НАЦІОНАЛЬНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ
імені М.П. ДРАГОМАНОВА

На правах рукопису

СЕМЕРІКОВ Сергій Олексійович

УДК 378.147:372.8004

ТЕОРЕТИКО-МЕТОДИЧНІ ОСНОВИ ФУНДАМЕНТАЛІЗАЦІЇ НАВЧАННЯ ІНФОРМАТИЧНИХ ДИСЦИПЛІН У ВИЩИХ НАВЧАЛЬНИХ ЗАКЛАДАХ

13.00.02 – теорія і методика навчання (інформатика)

Дисертація на здобуття наукового ступеня
доктора педагогічних наук

Науковий консультант:
ЖАЛДАК Мирослав Іванович
доктор педагогічних наук, професор,
дійсний член АПН України

Київ–2009

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

АПБ	Академія пожежної безпеки імені Героїв Чорнобиля
БДПУ	Бердянський державний педагогічний університет
БНТУ	Білоруський національний технічний університет
ВІППО	Волинський інститут післядипломної педагогічної освіти
ВНЗ	Вищий навчальний заклад
ДНДІАСБ	Державний науково-дослідний інститут автоматизований систем в будівництві
ЕОМ	електронна обчислювальна машина
ЕС	експертна система
ЄС	Європейський Союз
ЗІДМУ	Гуманітарний університет «Запорізький інститут державного і муніципального управління»
ІДГУ	Ізмаїльський державний гуманітарний університет
ІКТ	інформаційно-комунікаційні технології
ІКТН	інформаційно-комунікаційні технології навчання
ІСУЕП	Інститут соціального управління, економіки і права
КДПУ	Криворізький державний педагогічний університет
КЕІ	Криворізький економічний інститут
КНЕУ	Київський національний економічний університет імені Вадима Гетьмана
К-ПДУ	Кам’янець-Подільський державний університет
КПК	кишеньковий персональний комп’ютер
К-ПНУ	Кам’янець-Подільський національний університет імені Івана Огієнка
КТУ	Криворізький технічний університет
МДПУ	Мелітопольський державний педагогічний університет
МОН	Міністерство освіти і науки України
НДІ	науково-дослідний інститут

НДР	науково-дослідна робота
НМетАУ	Національна металургійна академія України
ООП	об’єктно-орієнтоване програмування
ОС	операційна система
ПАР	Південноафриканська республіка
ПВІЗ	Полтавський військовий інститут зв’язку
ПДПУ	Південноукраїнський державний педагогічний університет імені К.Д. Ушинського
ПЗ	програмне забезпечення
ПТНЗ	професійно-технічний навчальний заклад
СевНТУ	Севастопольський національний технічний університет
СКМ	система комп’ютерної математики
США	Сполучені Штати Америки
ТНПУ	Тернопільський національний педагогічний університет імені В. Гнатюка
ФРН	Федеративна Республіка Німеччина
ХДУ	Херсонський державний університет
ХНПУ	Харківський національний педагогічний університет імені Г.С. Сковороди
ЧНУ	Черкаський національний університет імені Богдана Хмельницького
1G	1st Generation («перше покоління», набір послуг аналогового мобільного зв’язку)
2.5G	second and a half Generation (2G-системи з технологією пакетної комутації каналів)
2D	2-Dimensional (двовимірний)
2G	2nd Generation («друге покоління», набір послуг цифрового мобільного зв’язку)
3,5G (4G)	4th Generation («четверте покоління», набір послуг мобільного зв’язку, що характеризується високою швидкістю передавання да-

них та підвищеною якістю голосового зв'язку)

3D	3-Dimensional (тривимірний)
3G	3rd Generation («третє покоління», набір послуг, що включає до себе як високошвидкісний мобільний доступ до послуг мережі Інтернет, так і технологію радіозв'язку)
ACM	Association for Computing Machinery (Асоціація обчислювальної техніки)
AJAX	Asynchronous JavaScript And XML (підхід до побудови інтерфейсів Web-додатків, за якого Web-сторінка, не перезавантажуючись, сама довантажує потрібні користувачу дані)
API	Application Programming Interface (прикладний програмний інтерфейс)
ASP	Active Server Pages (технологія, що дозволяє динамічно формувати автоматично обновлювані Web-сторінки з боку Web-сервера)
CDMA	Code Division Multiple Access (множинний доступ із кодовим розподілом каналів)
CD-RW	Compact Disc-ReWritable (перезаписуваний компакт-диск)
CLIPS	C Language Integrated Production System (інтегрована продукційна система мовою програмування С)
CMS	Content Management System (система керування вмістом)
COM	Component Object Model (модель об'єктних компонентів)
CORBA	Common Object Request Broker Architecture (загальна архітектура брокера об'єктних запитів)
CSS	Cascading Style Sheets (каскадні таблиці стилів)
DMA	Direct Memory Access (прямий доступ до пам'яті)
DVD-RW	Digital Versatile Disc Disc-ReWritable (перезаписуваний DVD)
E-Book	Electronic book device («електронна книга», пристрій для відображення тестових даних, поданих у електронному вигляді)
EGCS	Experimental/Enhanced GNU Compiler System (Експериментальна/Покращена збірка компіляторів GNU)

E-Ink	Electronic ink («електронні чорнила», технологія відображення даних, розроблена для імітації звичайного чорнила на папері)
EJB	Enterprise JavaBeans (специфікація технології написання та підтримки серверних компонентів, що містять бізнес-логіку)
EML	Education Modeling Language (мова моделювання навчання)
FPC	Free Pascal Compiler (компілятор мови програмування Паскаль з відкритими вихідними кодами)
FSF	Free Software Foundation (Фонд вільного програмного забезпечення)
FTP	File Transfer Protocol (протокол передачі файлів)
GCC	GNU Compiler Collection (Колекція компіляторів GNU)
GDB	GNU Debugger (налагоджувач GNU)
GNOME	GNU Network Object Model Environment («мережне об’єктне середовище GNU» – робоче середовище для UNIX-подібних операційних систем на основі бібліотеки GTK+)
GNU	«GNU's Not Unix» (проект створення вільної операційної системи)
GPL	GNU General Public License (Загальна громадська ліцензія GNU)
GPRS	General Packed Radio Service (загальна послуга пакетного радіопередавання)
GPS	Global Positioning System (система глобального позиціонування)
GSFL	Grid Service Flow Language (мова опису грід-сервісів)
GSM	Global System for Mobile communications (глобальна система мобільних комунікацій)
GTK+	The GIMP Toolkit (кросплатформенний набір віджетів для створення графічних інтерфейсів користувача)
HSDPA	High-Speed Downlink Packet Access (високошвидкісний пакетний доступ у напряму «донизу» – протокол високошвидкісного приймання пакетних даних стандарту мобільного зв’язку 3-го покоління)
HTML	HyperText Markup Language (мова розмітки гіпертекстових доку-

	ментів)
HTTP	Hyper Text Transfer Protocol (протокол передавання гіпертекстових документів)
HTTPS	Secure HTTP (об’єднання протоколів HTTP та SSL)
IDE	Integrated Development Environment (інтегроване середовище розробки)
IEC	International Electrotechnical Commission (Міжнародна електротехнічна комісія)
IEEE	Institute of Electrical and Electronics Engineers (Інститут інженерів з електроніки та електротехніки)
IP	Internet protocol (міжмережний протокол)
IPv4	протокол IP версії 4
IPv6	протокол IP версії 6
IrDA	Infrared Data Association (протокол обміну даними з використанням інфрачервоного випромінювання)
ISO	International Organization for Standardization (Міжнародна організація зі стандартизації)
IT	Information technology (інформаційні технології)
JSP	Java Server Pages (технологія, що дозволяє динамічно формувати автоматично обновлювані Web-сторінки з вбудованим Java-кодом)
KDE	K Desktop Environment (робоче середовище для UNIX-подібних операційних систем на основі бібліотеки Qt)
LCL	Lazarus Components Library (вільна бібліотека візуальних компонентів, подібна до VCL)
LGPL	GNU Lesser General Public License (Загальна громадська ліцензія обмеженого використання GNU)
LMS	Learning management system (система управління навчанням)
LO	Learning Object («навчальний об’єкт»)
LCMS	Learning content management system (система управління навчальним матеріалом)

MFC	Microsoft Foundation Classes (бібліотека класів Microsoft)
MID	Mobile Internet Device (мобільний Інтернет-пристрій)
MIDP	Mobile Information Device Profile (профіль для мобільного пристрою з інформаційними функціями)
MinGW	Minimalist GNU for Windows
MIT	Massachusetts Institute of Technology (Массачусетський технологічний інститут)
MLMS	Mobile Learning Management System (система управління мобільним навчанням)
MMORPG	Massively multiplayer online role-playing game (багатокористувацьке онлайнове ігрове середовище)
MMS	Multimedia Messaging Service (послуга мультимедійних повідомлень)
NASA	National Aeronautics and Space Administration (Національне управління з аеронавтики і космосу)
OLPC	One Laptom Per Child («Кожній дитині – по ноутбуку»)
OpenGL	Open Graphics Library (відкрита графічна бібліотека)
PAN	Personal area network (персональна мережа передавання даних)
PDA	Personal digital assistant (персональний цифровий помічник)
PEP	Palm Education Pioneers («Піонери навчання з Palm»)
PHP	PHP:Hypertext Preprocessor (PHP:гіпертекстовий препроцесор)
POSIX	Portable Operating System Interface (мобільний інтерфейс операційної системи)
RSS	Really Simple Syndication (сімейство форматів для публікації часто змінюваних даних)
RTL	Register Transfer Language (мова регістрового переносу)
SaaS	Software as a service («програмне забезпечення як послуга»)
SAC	Symbolic and Algebraic Computing (символьне й алгебраїчне опрацювання математичних виразів)
SAGE	Software for Algebra and Geometry Experimentation (програмне за-

	безпечення для алгебраїчних та геометричних досліджень)
SCORM	Sharable Content Object Reference Model (стандарт, розроблений для систем дистанційного навчання)
SDK	Software Development Kit (набір засобів розробки)
SFU	Microsoft Windows Services for UNIX (Сервіси Microsoft Windows для UNIX)
SMIL	Synchronized Multimedia Integration Language (мова інтеграції синхронізованого мультимедіа)
SMS	Short Message Service (Служба коротких повідомлень)
SOAP	Simple Object Access Protocol («простий протокол доступу до об'єктів» – протокол обміну структурованими повідомленнями в розподілених обчислювальних системах)
SRS	Student Response System (безпровідна система зворотного зв'язку зі студентами)
SSD	Solid State Drive (енергонезалежний перезаписуваний комп'ютерний запам'ятовуючий пристрій без рухомих механічних частин)
SSL	Secure Sockets Layer (протокол передавання зашифрованих повідомлень)
T9	Text on 9 Keys (система полегшеного введення тексту, що використовується в основному для написання SMS)
TCP/IP	Transmission Control Protocol / Internet Protocol (Протокол керування передачею / Протокол Internet)
UDDI	Universal Description Discovery and Integration (інструмент для розташування описів зовнішніх інтерфейсів Web-послуг для подальшого їх пошуку іншими організації та інтеграції у свої системи)
UMPC	Ultra-Mobile PC (ультрамобільний персональний комп'ютер)
UMTS	Universal Mobile Telecommunications System (універсальна мобільна телекомунікаційна система)
USB	Universal Serial Bus (універсальна послідовна шина)
VCL	Visual Component Library (бібліотека візуальних компонентів)

VHDL	VHSIC Hardware Description Language (мова опису апаратних засобів)
VLE	Virtual learning environment (віртуальне навчальне середовище)
VPN	Virtual Private Network (віртуальна приватна мережа)
VRML	Virtual Reality Modeling Language (мова моделювання віртуальної реальності)
WAP	Wireless Application Protocol (протокол бездротових програм)
Web-СКМ	система комп'ютерної математики з Web-доступом
Wi-Fi (WiFi)	Wireless Fidelity (назва стандарту бездротового зв'язку, який об'єднує декілька протоколів та ґрунтується на сімействі стандартів IEEE 802.11)
WiMAX	Worldwide Interoperability for Microwave Access (стандарт безпровідного зв'язку, що забезпечує широкосмуговий зв'язок на значні відстані зі швидкістю, порівняною з кабельними з'єднаннями)
WLAN	Wireless Local Area Network (бездротова локальна мережа)
WML	Wireless Markup Language (мова розмітки для пристроїв, що підтримують протокол обміну WAP)
WSDL	Web Services Description Language (мова опису зовнішніх інтерфейсів Web-послуг)
WWW	World Wide Web («всесвітня павутина»)
XHTML	Extensible Hypertext Markup Language (розширювана мова розмітки гіпертексту)
XML	Extensible Markup Language (розширювана мова розмітки)
XSI	X/Open System Interfaces Extension (X/Open-розширення системного інтерфейсу)

ВСТУП

Актуальність теми. Головні освітні тенденції 90-х рр. минулого століття – диференціація та спеціалізація навчання – виникли як відповідь на соціально зумовлену потребу ринкового суспільства знизити навчальне навантаження на студента та інтенсифікувати процес навчання у вищій школі з метою найшвидшого залучення молодої людини до суспільно-економічного життя. Проте в умовах ускладнення виробництва, прискорення науково-технічного прогресу та формування інформаційного суспільства вузькоспеціалізовані фахівці, підготовлені за скороченою програмою, швидко переставали б бути конкурентоспроможними на ринку праці. При цьому на початку XXI століття екстенсивними шляхом – подовженням терміну навчання та ускладненням навчального матеріалу – так і не вдалося розв'язати проблему швидкого застарівання знань, яка особливо гостро постала у сфері високих технологій – отримання нових матеріалів та здобування нових знань.

Аналіз вітчизняних та зарубіжних педагогічних досліджень показує, що на сучасному етапі інформатизації вищої освіти на перше місце виступають саме загальнотеоретичні, фундаментальні та міждисциплінарні знання, а не технологічні, утилітарні знання та вміння із застосування інформаційних технологій в навчальному процесі, як це мало місце в останні десятиліття.

Повернення до *фундаментальної освіти* в тому вигляді, в якому вона існувала в СРСР, є неможливим, оскільки змінилися соціально-економічні умови, роль знань у суспільстві, сама система освіти. Однак без фундаментальної освіти, без оволодіння системним знанням та без формування цілісної природничонаукової та інформаційної картини світу підготовка сучасного, здатного до навчання протягом всього життя фахівця також неможлива.

Розв'язанню протиріччя між радянським та сучасним підходами до визначення фундаментальної освіти сприяє чимало освітніх технологій – насамперед, це технології електронного і дистанційного навчання та тренінгові технології. Однак нова освітня парадигма, в основі якої лежить *фундаменталізація*

навчання, передбачає якісно нові цілі вищої освіти, нові принципи добору та систематизації знань: на базі цих принципів не стільки розширюється обсяг професійних та загальнонаукових знань, скільки визначаються їх зв'язки та способи формування і функціонування в практичній діяльності.

Спрямування системи освіти на особистість як головний соціальний орієнтир проявляється в різних напрямах, і провідним серед них є створення для будь-якого члена суспільства можливості отримання освіти будь-якого характеру та рівня в будь-який період його життя. Ця ідея знайшла відображення не лише за кордоном, в першу чергу – у економічно розвинених країнах, а й у вітчизняній системі освіти. Становлення особистісно орієнтованої системи освіти неможливе без підготовки для неї спеціалістів нового покоління – вчителів, здатних у своїй практичній діяльності реалізувати нову освітню парадигму.

Фундаменталізація предметної підготовки майбутніх вчителів інформатики та фахівців у галузі інформаційних технологій є актуальною задачею сучасної вищої освіти, оскільки характерною ознакою інформаційного суспільства є те, що в ньому «покоління речей та ідей змінюються швидше, ніж покоління людей» [119, 2]. Підготовка вчителів інформатики та інженерів-програмістів за суттю є професійною освітою, проте в сучасних соціально-економічних умовах традиційне протиріччя між фундаментальним та професійним навчанням набуває нового змісту: якщо в минулому вузька профілізація була показником високої соціальної захищеності, то сьогодні таким показником стає *мобільність*, якої може набути лише широко освічена людина, здатна гнучко реагувати на зміну технологій. Орієнтація на вузьких професіоналів, характерна для минулого століття, поступово зникає з виробничої сфери: у XXI столітті потрібен спеціаліст, здатний гнучко перебудовувати напрям та зміст своєї діяльності у зв'язку зі зміною життєвих орієнтирів чи вимог ринку. Досягнення мобільності (зокрема, навчальної та професійної) є однією з найважливіших задач Болонського процесу, розв'язання якої можливе лише за умови фундаментального характеру вищої освіти. Вузькопрофесійна підготовка поступово витісняється з системи вищої освіти. Проявом вказаної тенденції є заходи Міністерства освіти

та науки України, спрямовані на зближення вищої педагогічної та класичної університетської освіти.

Усунення існуючого протиріччя між соціальним замовленням суспільства, сучасними вимогами до підготовки фахівців у галузі інформаційних технологій, необхідністю підвищення їх фундаментальної підготовки та більш широкого використання мобільних технологій у освітній практиці з одного боку, та існуючою теорією і практикою навчання у ВНЗ, з іншого, є **суспільно значущою проблемою**.

Питанню фундаменталізації навчання у вищій школі присвячені роботи А.А. Аданнікова [2], С.І. Архангельського [7], О.В. Балахонова [12], С.А. Баляєвої [13], С.У. Гончаренка [55; 56], Г.Я. Дутки [75], О.В. Євця [80], І.В. Левченко [170], Л.С. Йолгіної [76], С.Я. Казанцева [113; 112], В.Г. Кінельова [121; 122], В.В. Кондратьєва [142; 143], С.В. Носирєва [219], А.Б. Ольневої [227], М.В. Садовнікова [289], О.В. Сергєєва [326], Н.Ф. Тализіної [363], В.Д. Шадрикова [238], М.О. Читаліна [425] та ін.

Методичні основи фундаментальної підготовки майбутніх учителів інформатики розглядали Т.О. Бороненко [28; 30], О.В. Горячов [60], А.П. Єршов [77; 78], М.І. Жалдак [98; 84], Т.П. Кобильник [135], К.К. Колін [138], О.І. Кухтенко [162], В.В. Лаптєв [164; 165], М.П. Лапчик [167], В.М. Монахов [199], Н.В. Морзе [210], В.Г. Разумовський [217], Т.М. Райхерт [269], Ю.С. Рамський [277], Н.І. Рижова [287; 286], І.О. Теплицький [258], Ю.В. Триус [147], С.І. Шварцбурд [136], М.В. Швецький [166; 427] та ін.

Становленню мобільного навчання присвячені роботи Д. Абернаті [449], Е. Вагнер [567], Р. Веттера [564], Т. Георгієва [495; 496], Дж. Еттевелла [458; 457], А. Кея [505], Д. Кігана [507], А. Кукульської-Хульме [509], Дж. Паско [529], О.П. Поліщука [391], Н. Рашбі [504; 540], П. Сеппала [544], І.О. Теплицького [308], Дж. Тракслера [519; 561], М. Шарплеса [546, 465], С.В. Шокалюк [393].

Фундаменталізація вищої інформатичної освіти впливає на всі компоненти методичної системи навчання інформатичних дисциплін: цілі, зміст, методи,

засоби, форми організації навчання. Це визначає два основних напрями модифікації методичної системи навчання інформатичних дисциплін. Перший – *фундаменталізацію змісту навчання* через надання йому властивостей стійкості, стабільності, збережуваності, тривалості. Другий – *підвищення мобільності* (навчальної, професійної, технологічної) через надання: навчанню властивості контекстності – чутливості до часу та місця; суб'єкту навчання більшої кількості «ступенів вільності» – вищої інтерактивності, більшої свободи руху, більшої кількості технічних засобів; засобам навчання властивостей відкритих систем – розширюваності, масштабованості, мобільності та «люб'язності».

Сказане зумовлює важливість проблеми фундаменталізації навчання інформатичних дисциплін у ВНЗ і **актуальність теми дослідження** «Теоретико-методичні основи фундаменталізації навчання інформатичних дисциплін у вищих навчальних закладах».

Зв'язок роботи з науковими програмами, планами, темами. Дисертаційне дослідження виконане в Національному педагогічному університету імені М.П. Драгоманова згідно з планом науково-дослідної роботи Інституту інформатики «Теоретичне обґрунтовування і розробка комп'ютерно-орієнтованих методичних систем навчання математики і інформатики в середніх загальноосвітніх і вищих педагогічних навчальних закладах» (код державної реєстрації 0198U001678) і «Комп'ютерно-орієнтовані методичні системи навчання природничих дисциплін в середніх загальноосвітніх і вищих педагогічних навчальних закладах» (код державної реєстрації 0101U002751).

Тема затверджена на засіданні Вченої ради Національного педагогічного університету імені М.П. Драгоманова 5 березня 2009 року (протокол №6) та узгоджена в Міжвідомчій раді з координації наукових досліджень з педагогічних і психологічних наук в Україні при АПН України 28 квітня 2009 року (протокол №3).

Метою дослідження є створення цілісної науково обґрунтованої методичної системи фундаментальної інформатичної підготовки майбутніх вчителів інформатики та фахівців у галузі інформаційних технологій.

Відповідно до мети було необхідно розв'язати наступні **завдання**:

1. Провести історичний та теоретико-методологічний аналіз сутності проблеми фундаменталізації вищої освіти;
2. Розкрити сучасні теоретико-методологічні підходи до фундаменталізації навчання інформатичних дисциплін у ВНЗ;
3. Розробити теоретичні засади фундаменталізації навчання інформатичних дисциплін на основі концепції мобільності;
4. Проаналізувати стан застосування мобільних технологій в процесі навчання, визначити організаційно-педагогічні, програмно-технічні і технологічні умови реалізації мобільного навчання;
5. Розробити методичну систему фундаментальної інформатичної підготовки у ВНЗ;
6. Експериментально перевірити результативність компонентів методичної системи фундаментальної інформатичної підготовки на прикладі курсів «Системне програмування», «Системне програмне забезпечення», «Подієорієнтоване програмування», а також розробити практичні рекомендації щодо їх впровадження і використання у ВНЗ;
7. Продіагностувати можливість застосування основних результатів дослідження та методичної системи фундаментальної інформатичної підготовки у практиці навчання студентів вищих навчальних закладів різного профілю.

Об'єктом дослідження є процес фундаменталізації навчання інформатичних дисциплін у вищих навчальних закладах III–IV рівнів акредитації.

Предмет дослідження – теоретичні та методичні основи фундаменталізації навчання інформатичних дисциплін у вищих навчальних закладах.

Для розв'язання поставлених завдань застосовувались такі **методи досліджень**:

а) *теоретичні* – аналіз чинних стандартів вищої освіти, навчальних програм, підручників і навчальних посібників, монографій, дисертаційних досліджень, статей і матеріалів науково-методичних конференцій з проблеми дослідження (розділ 1, розділ 2, розділ 4, п. 4.1); з питань інформатики та методики її

навчання (розділ 1, п. 1.3, розділ 3, пп. 3.1, 3.2); проблем застосування сучасних мережних та мобільних технологій в навчальному процесі ВНЗ (розділ 2, розділ 3, п. 3.3);

б) *емпіричні* – аналіз результатів навчання студентів у відповідності до проблеми дослідження, цілеспрямовані педагогічні спостереження, бесіди з викладачами та студентами, анкетування, тестування; аналіз досвіду роботи викладачів за основними положеннями дослідження (розділ 2, п. 2.5, розділ 3, п. 3.2, розділ 4, п. 4.1);

в) констатувальний, пошуковий та формувальний етапи педагогічного експерименту з наступним автоматизованим статистичним опрацюванням даних (розділ 4), якісним та кількісним аналізом результатів дослідження з метою з'ясування педагогічної ефективності компонентів методичної системи фундаментального навчання інформатичних дисциплін у вищій школі (розділ 4, п. 4.3).

Вибір методів дослідження визначався особливостями розв'язуваних завдань.

Наукова новизна одержаних результатів дисертаційного дослідження полягає у наступному:

– *вперше:*

1) розроблені, теоретично обґрунтовані і експериментально перевірені основні положення концепції фундаменталізації змісту та технологічної підсистеми методичної системи навчання інформатичних дисциплін у ВНЗ;

2) розроблені, теоретично обґрунтовані і експериментально перевірені основні положення технології мобільного навчання;

3) розроблені методичні основи застосування мобільних програмних засобів фундаменталізації навчання інформатичних дисциплін у ВНЗ;

– *удосконалено* модель регіонального інноваційного університетського комплексу як основи системи неперервної фундаментальної освіти;

– *дістало подальшого розвитку* положення про мікронавчання як основу технології мобільного навчання.

Практичне значення одержаних результатів дисертаційного дослідження полягає у наступному:

1) *обґрунтовано*:
   - цілі навчання і зміст предметів «Системне програмування», «Системне програмне забезпечення», «Подіє-орієнтоване програмування» та інших на основі інваріантності до операційної системи та мови програмування;
   - доцільність і ефективність використання середовища X Window для проектування мобільних мережних програм з графічним інтерфейсом та розроблено курс подіє-орієнтованого програмування в системі X Window;
2) *досліджено*:
   - програмно-апаратні та дидактичні можливості використання пристроїв класу «електронна книга» як інноваційних засобів мобільного навчання;
   - перспективи перенесення мобільного системного та прикладного програмного забезпечення у Web-середовища;
3) локалізовано:
   - оболонку експертних систем CLIPS та досліджено дидактичні можливості її використання при навчанні систем штучного інтелекту;
   - систему комп'ютерної математики Maxima та створено ряд нових інтерфейсів користувача до неї;
   - Web-СКМ SAGE та досліджено дидактичні можливості її використання при навчанні математичної інформатики;
4) розроблено комунікаційні бібліотеки для метакомп'ютингу, модулі компілятора Free Pascal для підтримки навчання системного програмування та методів розробки інтерфейсу користувача;
5) запропоновано структуру генераторів математичних текстів для систем дистанційного навчання.

Особистий внесок здобувача. У працях, опублікованих у співавторстві, автору належать такі результати:

1. Локалізовано оболонку експертних систем CLIPS, досліджено дидактичні можливості її використання при навчанні систем штучного інтелекту [82;

297; 309];

2. Локалізовано систему комп'ютерної математики Maxima, створено ряд нових інтерфейсів користувача до неї, розроблено на її основі факультативний курс «Комп'ютерні технології в наукових дослідженнях» [141; 174; 310; 325; 316; 317; 318; 442];
3. Розроблено низку Web-додатків для підтримки курсів «Штучний інтелект», «Операційні системи», «Комп'ютерні мережі», «Комп'ютерні технології в наукових дослідженнях», «Системне програмування» та ін. [68; 81; 313; 367];
4. Запропоновано структуру інформаційної системи вищого навчального закладу [148];
5. Досліджено методичні основи навчання системного програмування та об'єктно-орієнтованого моделювання засобами мобільних інтерпретованих мов програмування [176; 177; 256; 369];
6. Створено комунікаційні бібліотеки для метакомп'ютингу [181; 426];
7. Визначено напрями подальшого удосконалення інформатичних компетентностей студентів у галузі функціонального програмування [196; 323];
8. Створено модулі компілятора Free Pascal для підтримки навчання системного програмування та методів розробки інтерфейсу користувача [246; 248; 249];
9. Обґрунтовано доцільність і ефективність використання середовища X Window для проектування мобільних мережних програм з графічним інтерфейсом, розроблено курс подіє-орієнтованого програмування в системі X Window [247; 253; 255];
10. Розроблено спецкурс з сучасних методів і технологій розробки програмних засобів компонентної архітектури [259];
11. Досліджено програмно-апаратні та дидактичні можливості використання пристроїв класу «електронна книга» [296; 308; 393];
12. Запропоновано структуру генераторів математичних текстів для систем дистанційного навчання [301; 302];

13. Досліджено можливості застосування вільно поширюваного програмного забезпечення як фактора стабілізації курсів інформатики у вищих навчальних закладах [299; 374; 388].

У спільних роботах автору належить ідея написання посібника та його структура; написання всіх параграфів здійснено спільно з О.П. Поліщуком [248; 247; 249], І.О. Теплицьким [282; 341; 366] та В.М. Соловйовим [341].

У спільних статях [81; 148; 176; 244; 243; 245; 254; 259; 252; 253; 255; 304; 308; 309; 301; 313; 325; 317; 342; 368; 397; 390; 370; 376; 380; 392; 388; 393; 374; 371; 377; 379; 384; 437] автору належить постановка проблем, безпосередня участь у проведенні досліджень, формулювання їх основних результатів і здійснення загальної редакції. Автором визначено теми і зміст доповідей [15; 68; 82; 141; 174; 177; 181; 196; 236; 246; 241; 256; 258; 296; 306; 302; 323; 311; 316; 299; 303; 310; 324; 328; 343; 367; 369; 391; 394; 372; 426; 442; 436] на наукових конференціях, здійснено їх оприлюднення у переважній більшості випадків.

Апробація результатів дисертації. Основні положення і результати дослідження доповідались та обговорювались на наукових конференціях різного рівня: Першій міжнародній науково-практичній конференції «Інформоенергетичні технології адаптаційних процесів життєдіяльності на початку ІІІ тисячоліття» (Кривий Ріг, КДПУ, 15–17 березня 2001 р.); ІІІ-ій Всеукраїнській конференції молодих науковців «Комп'ютерне моделювання та інформаційні технології в науці, економіці та освіті» (Кривий Ріг, КДПУ, 26–28 квітня 2001 р.); Міжнародній науково-практичній конференції «Інформаційні технології та інформаційна безпека в науці, техніці та освіті» (Інфотех–2002) (Севастополь, СевНТУ, 30 вересня – 5 жовтня 2002 р.); Всеукраїнській науково-методичній конференції «Теорія та методика навчання фундаментальних дисциплін у вищій технічній школі» (Кривий Ріг, НМетАУ, 14–15 березня 2003 р.); V-ій Всеукраїнській науково-практичній конференції «Комп'ютерне моделювання та інформаційні технології в науці, економіці та освіті» (Черкаси, ІСУЕП, 21–23 квітня 2003 р.); Всеукраїнській науково-практичній конференції «Формування духовної культури особистості в процесі навчання математики в школі та вищому на-

вчальному закладі» (Луцьк, ВІППО, 22–24 травня 2003 р.); Четвертому міжнародному науково-методичному семінарі «Інформаційні технології в навчальному процесі» (Одеса, ПДПУ, 25–28 червня 2003 р.); ІІІ-ій Всеукраїнській конференції «Сучасні технології в науці та освіті» (Кривий Ріг, КДПУ, 12–13 вересня 2003 р.); Міжнародній науково-методичній конференції «Методологічні принципи формування фізичних знань учнів і професійних якостей майбутніх учителів фізики та астрономії» (Кам'янець-Подільський, К-ПДУ, 2–4 жовтня 2003 р.); IV-ій Всеукраїнській науково-практичній конференції «Теорія та методика навчання фундаментальних дисциплін у вищій школі» (Кривий Ріг, НМетАУ, 8–9 квітня 2004 р.); IV-ій Всеукраїнській конференції молодих науковців «Інформаційні технології в освіті, науці і техніці» (ІТОНТ–2004) (Черкаси, ЧНУ, 28–30 квітня 2004 р.); Міжнародній науково-технічній конференції «Інтегровані системи управління в гірничо-металургійному комплексі» (ІСГМК–2004) (Кривий Ріг, КТУ, 11–13 травня 2004 р.); Міжнародній науково-практичній конференції «Інформаційно-комунікаційні технології у середній і вищій школі» (Ізмаїл, ІДГУ, 27–29 травня 2004 р.); Всеукраїнській науково-практичній конференції «Інформатика та комп'ютерна підтримка навчальних дисциплін у середній і вищій школі» (Бердянськ, БДПУ, 23–26 червня 2004 р.); Міжнародній науково-практичній конференції «Інформаційні технології та інформаційна безпека в науці, техніці та освіті» (Інфотех–2004) (Севастополь, СевНТУ, 20–25 вересня 2004 р.); Всеукраїнському науково-методичному семінарі «Комп'ютерне моделювання в освіті» (Кривий Ріг, КДПУ, 29 березня 2005 р.); V-ій Всеукраїнській науково-практичній конференції «Теорія та методика навчання фундаментальних дисциплін у вищій школі» (Кривий Ріг, НМетАУ, 8–9 квітня 2005 р.); Міжнародній конференції «Проблеми прийняття рішень в умовах невизначеності» (Бердянськ, БДПУ, 12–16 вересня 2005 р.); міжнародній науковій конференції «Дидактика фізики в контексті орієнтирів Болонського процесу» (Кам'янець-Подільський, К-ПДУ, 22–24 вересня 2005 р.); Всеукраїнській конференції «Нові інформаційні технології навчання: психологічні проблеми» (Київ, Інститут психології ім. Г.С. Костюка АПН України, 11–

12 жовтня 2005 р.); Міжнародній науково-практичній конференції «Системний аналіз і управління» (Запоріжжя, ЗІДМУ, 27–28 жовтня 2005 р.); Всеукраїнському науково-методичному семінарі «Комп'ютерне моделювання в освіті» (Кривий Ріг, КДПУ, 26 квітня 2006 р.); VI-ій міжнародній науково-практичній конференції «Теорія та методика навчання фундаментальних дисциплін у вищій школі» (Кривий Ріг, НМетАУ, 27–28 квітня 2006 р.); V-ій Всеукраїнській конференції молодих науковців «Інформаційні технології в освіті, науці і техніці» (ІТОНТ–2006) (Черкаси, ЧНУ, 3–5 травня 2006 р.); Міжнародній науково-практичній конференції «Модернізація освіти: пошуки, проблеми, перспективи» (Київ–Переяслав-Хмельницький, Інститут педагогіки АПН України, 22–25 травня 2006 р.); Всеукраїнській науково-практичній конференції «Інформаційні технології в освіті» (Мелітополь, МДПУ, 24–26 травня 2006 р.); IV-ій міжнародній науково-технічній конференції «Комп'ютерні технології в будівництві» (Севастополь, ДНДІАСБ, 18–21 вересня 2006 р.); Міжнародному симпозіумі «Проблеми дидактики фізики та шкільного підручника фізики в світлі сучасної освітньої парадигми» (Кам'янець-Подільський, К-ПДУ, 9–11 листопада 2006 р.); Другій Всеукраїнській науково-практичній конференції «Інноваційні технології навчання в сучасній дидактиці вищої школи» (ІТНСДВШ – Полтава 2007) (Полтава, ПВІЗ, 13–16 березня 2007 р.); VII-ій Всеукраїнській науково-практичній конференції «Комп'ютерне моделювання та інформаційні технології в науці, економіці і освіті» (Кривий Ріг, КЕІ КНЕУ, 24–25 квітня 2007 р.); Всеукраїнській науково-практичній конференції «Нові технології навчання: психологічні аспекти» (Київ, Інститут психології ім. Г.С. Костюка АПН України, 15–16 травня 2007 р.); V-ій міжнародній науково-технічній конференції «Комп'ютерні технології в будівництві» (Севастополь, ДНДІАСБ, 18–21 вересня 2007 р.); Міжнародній Інтернет-конференції «Дидактика фізики і підручники фізики (астрономії) в умовах формування європейського простору вищої освіти» (Кам'янець-Подільський, К-ПДУ, 22–23 жовтня 2007 р.); II-ій міжнародній науково-методичній конференції «Вимірювання навчальних досягнень школярів і студентів: Гуманістичні, методологічні, методичні, технологічні аспекти»

(Харків, ХНПУ, 13–14 грудня 2007 р.); Міжвузівській науково-практичній конференції «Актуальні проблеми технічних, природничих та соціально-гуманітарних наук в забезпеченні цивільного захисту» (Черкаси, АПБ, 25 березня 2008 р.); VII-ій міжнародній науково-практичній конференції «Теорія та методика навчання фундаментальних дисциплін у вищій школі» (Кривий Ріг, НМетАУ, 17–18 квітня 2008 р.); VIII-ій Всеукраїнській науково-практичній конференції «Комп'ютерне моделювання та інформаційні технології в науці, економіці і освіті» (Кривий Ріг, КЕІ КНЕУ, 22–23 квітня 2008 р.); III-ому Всеукраїнському науково-методичному семінарі «Комп'ютерне моделювання в освіті» (Кривий Ріг, КДПУ, 24 квітня 2008 р.); VI-ій Всеукраїнській конференції молодих науковців «Інформаційні технології в освіті, науці і техніці» (ІТОНТ–2008) (Черкаси, ЧНУ, 5–7 травня 2008 р.); 2-гій Міжнародній Інтернет-конференції «Інновації в навчанні фізики та дисциплін технологічної освітньої галузі: міжнародний та вітчизняний досвід» (Кам'янець-Подільський, К-ПНУ, червень-вересень 2008 р.); VI-ій міжнародній науково-технічній конференції «Комп'ютерні технології в будівництві» (Севастополь, ДНДІАСБ, 9–12 вересня 2008 р.); Всеукраїнській науково-практичній конференції «Проектування освітніх середовищ як методична проблема» (Херсон, ХДУ, 16–19 вересня 2008 р.); Всеукраїнській науково-практичній конференції «Теоретичні та прикладні аспекти використання інформаційних технологій у вищій і загальноосвітній школах» (Тернопіль, ТНПУ, 25–27 вересня 2008 р.); Міжнародній науково-практичній конференції студентів та молодих науковців «Молодий науковець XXI століття» (Кривий Ріг, КТУ, 17–18 листопада 2008 р.).

Матеріали і результати дослідження обговорювалися на засіданнях і семінарах кафедри теоретичних основ інформатики Національного педагогічного університету ім. М.П. Драгоманова, кафедри інформатики та прикладної математики Криворізького державного педагогічного університету, кафедри технічної кібернетики Кременчуцького університету економіки, інформаційних технологій і управління, кафедри математичних методів та інформаційних техно-

логій в економіці Запорізького інституту економіки та інформаційних технологій, а також апробовані шляхом публікацій.

Впровадження результатів дисертаційного дослідження у педагогічну практику підтверджується довідками Криворізького державного педагогічного університету (№7/03 від 13.03.2009 р.), Криворізького технічного університету (№4 від 05.03.2009 р.), Черкаського національного університету імені Богдана Хмельницького (№2 від 13.03.2009 р.), Національної металургійної академії України (№10/2 від 23.10.2008 р.), Запорізького інституту економіки та інформаційних технологій (№1 від 15.09.2008 р.), Кременчуцького університету економіки, інформаційних технологій і управління (№2 від 05.10.2008 р.).

Публікації. Основні результати дослідження опубліковано у 95 науково-методичних працях загальним обсягом 105,6 д.а. (особистий внесок 67,79 д.а.), серед них: 1 монографія (15,95 д.а.), 9 навчальних посібників для студентів (63,20 д.а., особистий внесок 38,23 д.а.), 28 статей – у фахових виданнях (13,95 д.а., особистий внесок 7,45 д.а.), 17 статей у журналах та збірниках наукових праць (7,15 д.а., особистий внесок 3,56 д.а.), 40 тез доповідей – у матеріалах конференцій (5,35 д.а., особистий внесок 2,59 д.а.).

РОЗДІЛ 1
ТЕОРЕТИЧНІ ОСНОВИ ФУНДАМЕНТАЛІЗАЦІЇ ІНФОРМАТИЧНОЇ ОСВІТИ У ВИЩІЙ ШКОЛІ

1.1. Фундаментальність як основа університетської освіти

Вперше концепція *фундаментальної освіти* була сформульована Вільгельмом фон Гумбольдтом: в ній зазначалося, що предметом такої освіти мають бути саме ті фундаментальні знання, які сьогодні відкриває наука; більш того, освіта має бути вбудована в наукові дослідження. Вища школа XIX сторіччя переважно слідувала моделі Гумбольдта, згідно якої університет (з латини – «сукупність») – це елітарний навчальний заклад, в якому навчання та наукові дослідження знаходяться в нерозривній єдності, і головний акцент робиться на підготовку та виховання творчої особистості, здатної до саморозвитку.

1.1.1. Модель Гумбольдта в контексті Болонського процесу. У лютому 1809 р. Гумбольдт як директор зі справ культури й освіти міністерства внутрішніх справ у Берліні став свого роду міністром освіти й науки в Пруссії. І хоча вже через 13 місяців він подав прусському королю прохання про відставку й покинув службу, час перебування Гумбольдта на цій посаді сьогодні розглядається як один з найважливіших періодів німецької історії освіти [74].

Систематичне подання своєї концепції Гумбольдт здійснив у меморандумі 1810 р. «Про внутрішню й зовнішню організацію вищих наукових закладів у Берліні» [67], в якому він формулює свої пропозиції про необхідні філософсько-теоретичні й організаційні основи університету в порівнянні з іншими науковими установами та школами в Берліні. Найбільш важливим пунктом меморандуму є «ідеал самокерованої науки» як передумови будь-якої доцільної й успішної діяльності наукових установ як в галузі навчання, так і в галузі наукового дослідження.

Освіта студента повинна проходити при наявності зовнішнього керівництва, але самостійно та в тісному зв’язку з науковим дослідженням. У такий спо-

сіб студент повинен «створювати свій дух і характер» і цілеспрямовано готуватися до трудового життя, що у ті часи, як правило, означало державну службу.

«Самітність і свобода» вченого визначаються Гумбольдтом як головні принципи чистої науки й наукового дослідження. Хоча Гумбольдт поважає індивідуальне прагнення до пізнання й наукову свободу, він формулює в тому ж контексті також центральну для себе думку про те, що «розумова діяльність у людстві розвивається тільки як спільна діяльність» [67]. Гумбольдт вважав, що ця взаємодія необхідна не тільки для різних учених університету – вона сприятиме також взаємодії між університетськими дослідниками й викладачами з одного боку, і студентами з іншого.

Гуманізм Гумбольдта характеризується тим, що він вважає дискусію між ученим-викладачем і самостійно (та критично) міркуючим студентом справжнім «еліксиром наукового життя». Тому для Гумбольдта школу й університет необхідно чітко відокремити, розділити. Студент для нього більше не є школярем, що вдячно слухає повчання професора, а є додатковою інстанцією перевірки тез дослідника, а від діалогу між університетським викладачем і студентом виграють обидва.

У своєму меморандумі Гумбольдт визначає два завдання держави: 1) держава повинна «чітко й твердо» відокремити науково-навчальну установу (університет) від суто навчальної (школи у всіх її формах); 2) держава має піклуватися про те, щоб «завжди підтримувати діяльність», а саме автономну дослідницьку діяльність університету, «у самому живому й сильному життєвому стані». Держава повинна при цьому «завжди усвідомлювати, що не вона, насправді, домагається або може домогтися цього, а, що вона, скоріше, завжди є перешкодою, як тільки вона втручається, що без неї справа сама по собі пішла б набагато краще». Інакше кажучи, держава повинна знаходити фінансові засоби для університету, а втручатися якнайменше.

Гумбольдт вважає за необхідне залучати державу до відбору професорів та забезпечення «свободи їхньої діяльності», адже їй загрожує небезпека не тільки з боку держави, а й з університетських надр. Можливо, що існуючі там

школи мислення «приймають певний дух і мають тенденцію задушити розвиток іншого». Саме тому Гумбольдт переконаний у корисності свободи вченого для держави, він вимагає, щоб держава піклувалася про те, щоб професорами обиралися дійсно найкращі особи й щоб при цьому дотримувалися принципи загальнонаукових і професійних компетентностей й різноманіття думок.

Постійний науковий пошук Гумбольдт вважає головною характеристикою університетської освіти, а для його забезпечення пропонує «потрійне прагнення духу:

1) виводити все з єдиного первісного принципу (причому пояснення природи можуть бути підняті, наприклад, з механічних до динамічних, органічних і, нарешті, психічних пояснень у найширшому розумінні);

2) спрямувати все на єдиний ідеал;

3) зв'язувати згадані вище єдиний принцип і єдиний ідеал у єдину ідею» [67].

Саме цей «єдиний принцип» Гумбольдта став першою основою для поділу наук на фундаментальні та прикладні.

В своєму меморандумі Гумбольдт розрізняє навчання (як підготовку) та освіту, які він слідом за давньогрецькими філософами визначає через *bios prakticos* (примат соціальної корисності отриманого знання та професійної підготовки) та *bios theoreticos* (примат дозвілля та споглядання). Університет не може обмежуватися лише практичною підготовкою, *тому що в ньому відбувається не простий приріст знання, а й інтелектуальний розвиток студентів через універсальне навчання, вільну циркуляцію мислі, диспути та особисте спілкування*. Концепція університету Гумбольдта стала основою гуманістичного трактування університету, до ознак якого філософи науки та освіти відносили атмосферу думки (Дж. Ньюмен [220]), інтелігентність духу (Х. Ортега-і-Гассет [228]), інтелектуальну совість (К. Ясперс [448]), смак (П. Бурд'є [37]) та стиль (Ю. Хабермас [418]). Перед зростаючою небезпекою технократичного мислення вже в рамках своєї концепції Гумбольдт акцентував особливу увагу на проблемах гуманістичного виховання студентів.

Берлінський університет Фрідріха Вільгельма, створений на основі меморандуму Гумбольдта, став моделлю університету XIX–XX ст. (моделлю Гумбольдта [433]), фундаментальними принципами якого є *академічна свобода* (при одночасній відповідальності перед потребами держави та суспільства) і *єдність дослідження та навчання*.

Академічна свобода містила в собі: право на самоуправління під державним наглядом; ведення власного господарства; обмежене владою міністра право факультетів на комплектацію посад (що здійснювалося шляхом висування кандидатур на вакантну посаду, а також забезпечувалося правом надання *venia legendi* після захисту габілітаційної роботи, на основі чого надавалося звання приват-доцента); розділення іспитів на державні та незалежні від держави академічні; свобода подання навчального матеріалу для професорів і доцентів та лише формально обмежена свобода доступу для студентів (вимагалась лише наявність шкільного атестата). Єдність дослідження й навчання гарантувалася фігурою університетського викладача-дослідника, а також свободою вибору лекцій для студентів (на відміну від гімназії); інститутом семінарів («розплідників» науки); тісним зв'язком університету з іншими дослідницькими інститутами, що незабаром стали виникати у формі університетських інститутів.

Державна цільова програма «Наука в університетах» на 2008–2012 роки [283] фактично передбачає реалізацію моделі Гумбольдта в системі вищої освіти України через *дослідницькі університети*. Основою такого університету є наука та науково-педагогічні школи, при цьому в науці має переважати частка фундаментальних наукових досліджень, а в навчальному процесі передбачається поєднання природничо-наукових, гуманітарних дисциплін та фундаментальності освіти.

М.З. Згуровський зазначає, що «... об'єднана Європа, приділяючи величезну увагу масовій освіті, у подоланні науково-технологічного відставання ... найбільше покладається саме на дослідницькі університети. ... По-перше, їм, як ніколи доведеться підсилити фундаментальну складову як у навчанні, так і в наукових дослідженнях. ... Технології не можуть створюватися без глибинного

розуміння властивих їм фізичних, хімічних, біологічних процесів на молекулярному рівні та у наномасштабі. Нарешті, успішна ринкова реалізація наукоємної продукції потребує глибоких знань соціології, психології та економіки як фундаментальних дисциплін. ... По-друге, дослідницькі університети мають бути міждисциплінарними з менш вузькою спеціалізацією навчальних планів та з більш відкритою й мобільною системою навчання для студентів і викладачів» [103].

Основними характеристиками сучасного університету є:

– поліфункціональність – здатність як генерувати, так і забезпечувати трансфер знань;

– сильна орієнтація на наукові дослідження та розробки, насамперед – фундаментальні;

– наявність системи підготовки спеціалістів з науковими ступенями;

– орієнтація на сучасні напрями науки, високі технології та інноваційний сектор в економіці, науці та техніці;

– широкий набір спеціальностей та спеціалізацій;

– високий професійний рівень викладачів, прийнятих на роботу на конкурсній основі; наявність можливостей для запрошення провідних фахівців з різних країн світу на тимчасову роботу;

– високий ступінь інформаційної відкритості та інтеграція в міжнародну систему науки та освіти;

– сприйнятливість до світового досвіду та гнучкість у відношенні до нових напрямів наукових досліджень та методологій навчання;

– конкурсність та селективний підхід при наборі студентів;

– формування навколо університету особливого інтелектуального середовища;

– наявність корпоративної етики, що базується на етиці науки, демократичних цінностях та академічних свободах;

– формування навколо університету специфічного науково-технічного та економічного простору – технопарків;

– прагнення до лідерства у регіоні, країні та світовому науково-освітньому співтоваристві в цілому.

Велика Хартія Університетів [39], підписанням якої у 1988 р. розпочався Болонський процес, визначає наступні *основні принципи функціонування університетів*.

1. Університет є самостійною установою усередині суспільств із різною організацією, що є наслідком розходжень у географічній та історичній спадщині. Він створює, вивчає, оцінює і передає культуру за допомогою досліджень і навчання. Для задоволення потреб навколишнього світу його дослідницька і викладацька діяльність повинна бути морально й інтелектуально незалежною від будь-якої політичної й економічної влади.

2. Викладання і дослідницька робота в університетах повинні бути нероздільні для того, щоб навчання в них відповідало потребам, що змінюються, запитам суспільства і досягненням у науковому знанні.

3. Свобода в дослідницькій і викладацькій діяльності є основним принципом університетського життя. Керівні органи й університети, кожний у рамках своєї компетентності, повинні гарантувати дотримання цієї фундаментальної вимоги. Відкидаючи нетерпимість і будучи завжди відкритим для діалогу, університет є ідеальним місцем зустрічі викладачів, що здатні передавати свої знання і володіють необхідними засобами для їхнього удосконалювання за допомогою досліджень та інновацій, і студентів, що мають право, здатність і бажання збагатити свій розум цими знаннями.

4. Університет є хоронителем традицій європейського гуманізму. У здійсненні свого покликання він постійно прагне до досягнення універсального знання, перетинає географічні і політичні кордони і затверджує нагальну потребу взаємного пізнання і взаємодії різних культур.

Перші три принципи Хартії (та частково четвертий) повністю відповідають моделі Гумбольдта. Новим у четвертому принципі є *мобільність*, засобом реалізації якої розглядають взаємний обмін відомостями і документацією, збі-

льшення кількості спільних проектів для розвитку освіти, як основний елемент постійного прогресу знань [275, 78].

Таким чином, можна зробити висновок, що **концепція фундаментальності для вищої освіти є системоутворюючою**, а фундаменталізація вищої освіти є одним із пріоритетів Болонського процесу.

1.1.2. Фундаментальні знання та фундаменталізація освіти. Найважливішим напрямом реформування системи освіти справедливо вважають її фундаменталізацію. Спрямованість на фундаменталізацію освіти необхідна для того, щоб майбутній фахівець у процесі навчання зміг набути необхідні фундаментальні базові знання, сформовані в єдину світоглядну наукову систему на основі сучасних уявлень про науку та її методи. Даний підхід надасть можливість одержувати необхідні знання не тільки з обраної спеціальності, а й з усього комплексу пов’язаних з нею наук, включаючи природничо-наукові та гуманітарні знання, що формують не тільки професійні навички, але й особистісні потреби, відповідальність фахівця перед наукою й людством.

Фундаментальна наука завжди передувала виробництву: навіть найдавніше гончарне виробництво починалось зі знань про існування глини, її еластичності у вологому стані та затвердінні при висиханні та випалюванні тощо, і лише згодом сформувалась компетентність виробника. Аналогічно й інші прадавні виробництва виникли еволюційно в процесі повільного накопичення знань та досвіду. Саме тому в докапіталістичну епоху людина, що набула фахових компетентностей в процесі навчання, зберігала свої компетентності протягом всього життя.

В епоху капіталізму нові виробництва створювалися в короткі історичні терміни на основі наукових відкриттів. Так, на основі відкриттів термодинаміки виникло виробництво теплових двигунів, на основі досягнень електродинаміки – електродвигунів тощо. У цю епоху зародився процес, що надалі назвали *науково-технічним прогресом*, однак протікав він переважно за рахунок внутрішніх рушійних сил виробництва та був достатньо повільним для того, щоб

знання, отримані в процесі підготовки фахівця, не встигали істотно застарівати за час його професійної діяльності.

В інформаційному суспільстві темпи науково-технічного прогресу (які, на думку Ю.В. Триуса, і є одним з показників формування інформаційного суспільства [405, 33]) різко зростають, унеможливлюючи підготовку фахівців для негайного включення їх у технологічний ланцюжок або систему освіти, тому що неможливо точно передбачити стан технологій або системи освіти, що буде сформовано до моменту випуску фахівця: «ознакою прискорення технічного прогресу виступає **швидке скорочення проміжку часу між винаходом нового процесу та початком його використання в масовому виробництві:** якщо людству треба було 112 років для опанування фотографії і 56 років для організації широкого використання телефонного зв'язку, то відповідні терміни для радара, телебачення, транзистора й інтегральної мікросхеми складають відповідно 15, 12, 5 і 3 роки» [405, 50–51]. Звідси випливає наступний шлях вирішення проблеми: *навчати фахівця так, щоб він сам умів швидко адаптуватися в ситуації, що змінюється, дати йому знання, універсальні за своєю суттю, на основі яких фахівець зможе швидко змінити себе в новій сформованій обстановці*. Вихід з цієї критичної ситуації в системі освіти полягає у фундаменталізації освіти. Фундаменталізація освіти зумовлюється спрямованістю системи освіти на створення цілісного, узагальнюючого знання, яке було б ядром всіх отриманих студентом знань, що поєднувало б одержувані в процесі навчання знання в єдину світоглядну систему на базі сучасної методології.

Найбільш ефективною є освіта, що базується на єдності фундаментальності й професійної спрямованості навчання. Принцип професійної спрямованості навчання є найважливішим для вищої школи, тому що вища школа завжди була, є й принаймні найближчим часом буде професійною за своєю суттю та призначенням. І, незважаючи на запланований у новій редакції Закону України «Про вищу освіту» [101] перехід до узагальнених кваліфікацій, професійна складова у вищій освіті завжди буде матиме місце.

У методичній системі навчання повинні бути одночасно реалізовані обидва принципи: фундаментальності й професійної спрямованості. Розглянемо більш детально основні поняття, що лежать в основі фундаменталізації освіти, та визначимо роль професійної спрямованості у процесі фундаменталізації.

За В.Г. Кінельовим, фундаментальна освіта являє собою процес нелінійної діяльності людини в інтелектуальному середовищі і його впливі на особистість, в якому людина сприймає його для збагачення власного внутрішнього світу й завдяки цьому дозріває для примноження потенціалу самого середовища [120, 7]. Завдання фундаментальної освіти дослідник вбачає у забезпеченні сприятливих умов для виховання гнучкого й багатогранного наукового мислення, різних способів сприйняття дійсності, формування внутрішньої потреби в самореалізації й самоосвіті протягом усього життя.

З плином часу стрімко зростаючий обсяг різноманітних відомостей призвів до необхідності їх адекватного структурування та відображення в навчальних дисциплінах, що перетворило фундаментальну освіту у самостійну та найважливішу галузь інтелектуальної діяльності людини. Велику роль в цьому можуть відіграти курси, що містять найбільш фундаментальні знання, які є базою для формування загальної та професійної культури, швидкої адаптації до нових професій, спеціальностей та спеціалізацій [122].

С.І. Ожегов термін «*фундаментальний*» визначає як «1) великий та міцний; стійкий, глибокий; 2) основний, головний» [223, 789]. В.В. Ільїн визначає *фундаменталізм* як «допущення граничних унітарних основоположень, що утворюють для пізнавального різноманіття та розмаїтості непорушний моноліт центр-базис, який імплікує похідні від нього дистальні одиниці знання» [106, 6].

Фундаментальні знання – це найбільш стабільні та універсальні загальнотеоретичні знання, зміст яких відзначається максимальною узагальненістю, структурованістю, розкриває та визначає розмаїття внутрішніх та зовнішніх зв’язків даних [166]. Фундаментальні знання, будучи інструментом досягнення наукових компетентностей, орієнтовані на пізнання глибинних, сутнісних зв’язків між різноманітними процесами. «Фундаментальні знання формують

здатність особи опановувати нові знання, орієнтуватися у проблемах, що виникають, виконувати задачі діяльності, що прогнозуються. Фундаментальні знання є інваріантні у відношеннях: напрями підготовки до певної галузі освіти; спеціальності до напряму підготовки; спеціалізації спеціальності до спеціальності» [139, 18].

Н.В. Скоробогатова так визначає основну ціль навчання у вищих навчальних закладах: «формування висококваліфікованих фахівців ..., які мають фундаментальну теоретичну підготовку та здатні застосовувати набуті знання для творчого розв'язування практичних задач» [331, 14].

За визначенням В.Є. Соколова, до *фундаментальних наук* відносяться такі, основні визначення, поняття та закони яких первинні, не є наслідками інших наук та безпосередньо відображають, синтезують в закони й закономірності факти, явища природи та суспільства [339, 801]. При цьому, на думку Н.Ф. Тализіної, підготовка спеціалістів на базі фундаментальних наук не означає зменшення уваги до професійних видів діяльності, проте «... вивчення фундаментальних наук не повинно бути й рядоположено з професійними предметами: фундаментальні науки мають орієнтувати спеціаліста в своїй галузі, дозволяти йому не лише самостійно аналізувати наявні в ній нагромадження, а й передбачати її подальший розвиток» [363, 8]. Фундаментальна наука, матеріальне й інформаційне виробництво та освіта взаємозалежні між собою та утворюють складну систему.

Фундаментальна підготовка спрямована на посилення взаємозв'язків теоретичної й практичної підготовки молодого фахівця до професійної діяльності; спрямована на формування цілісної наукової картини навколишнього світу, на індивідуально-професійний розвиток студента, що в сукупності забезпечує високу якість освіти.

Аналіз сучасних підходів до визначення принципів функціонування системи освіти України показав домінування протягом останніх 15 років загальної гуманістичної спрямованості освіти, що виражається в її особистісній зорієнтованості та поєднанні гуманізації із професійною й культурологічною парадиг-

мами вищої освіти при збереженні базової ролі знаннєвої парадигми як орієнтації на системність і науковість знань, що визначають розвиток творчого потенціалу людини. У зв'язку із цим увагу дослідників привернув акмеологічний підхід, орієнтований на розвиток внутрішніх резервів і механізмів самовдосконалення людини в освітній діяльності: мотивації досягнень, саморозвитку, творчості, пріоритету духовно-моральних цінностей і цілей розвитку особистості, переважно орієнтованих на високий рівень професіоналізму й професійних досягнень [12, 15].

Прагматичне розуміння гуманітаризації виключно як збільшення частки гуманітарних дисциплін керівниками окремих навчальних закладів призвели до суттєвих розбіжностей як у навчальних планах, так й у рівні професійної підготовки випускників різних ВНЗ, що навчалися за однією спеціальністю. Виявлені розбіжності (див. п. 4.1) не лише були причиною різного рівня професійної підготовки випускників, що навчалися за однією спеціальністю, а й суттєво утруднювали перехід студента до іншого ВНЗ для продовження навчання через велику – в межах однієї спеціальності – академічну різницю.

Таким чином, наприкінці минулого століття виникла нагальна необхідність, по-перше, у прийнятті державних стандартів вищої освіти, які б визначали фахові компетентності випускника та зміст фахової підготовки, та, по-друге, в засобах забезпечення мобільності студентів в процесі навчання.

Перший напрям роботи був в цілому завершений у 2002 р. прийняттям галузевих стандартів вищої освіти з поширенням освітньо-кваліфікаційних характеристик та освітньо-професійних програм спеціальностей, які забезпечили уніфікацію держкомпонентів навчальних планів. Другий напрям роботи реалізується через впровадження системи залікових кредитів, єдиної для Європейської освітньої спільноти.

Разом з тим, необхідно відзначити, що вища інформатична освіта в значній мірі будується, як і раніше, на основі накопичувальної моделі нових знань, коли формуються вміння розв'язувати стандартні професійні завдання, діяти у відомих ситуаціях. Проте в умовах неодноразової зміни освітніх парадигм та

технологій навчання в процесі роботи викладача, апаратних платформ та технологій програмування в професійній діяльності педагога, інженера-програміста актуальними стають проблеми переходу від інформаційно-накопичувальної моделі вищої інформатичної освіти до методологічно орієнтованої моделі, що формує в майбутнього фахівця здатність до розв'язування нестереотипних професійних завдань, до творчого мислення на основі фундаментальних знань.

Розглядаючи теоретико-методологічні основи фундаменталізації університетської освіти, О.В. Балахонов пропонує визначення *фундаменталізації* як процесу якісної зміни вищої освіти на основі принципу її фундаментальності [12, 16–17]. У термінах експертів «Римського клубу» це означає необхідність переходу від «підтримуючої» до «випереджальної» інноваційної освіти.

О.Г. Ростовцева визначає фундаменталізацію як «впровадження в навчальний процес теорій високого ступеня узагальненості, що мають підвищену інформаційну ємність та універсальну застосовність» [284, 13]. І.Ю. Асманова уточнює, що фундаменталізація освіти має відбуватися «не шляхом розширення навчальних планів за рахунок включення нових дисциплін, міждисциплінарних теорій чи методологічних знань, а шляхом зміни способу вивчення ... дисципліни» [8, 168].

Аналізуючи вплив фундаменталізації на методичну систему навчання, М.В. Садовников вказує на те, що «фундаменталізація освіти як один з найважливіших зовнішніх факторів ... системи вищої педагогічної освіти справляє найбільший вплив на такі компоненти цієї системи, як цілі та зміст. Інші компоненти також знаходяться під впливом фундаменталізації, але в меншій степені» [289, 10].

Л.М. Харченко [420, 53] зазначає, що поняття *фундаменталізації навчання* має два основних трактування: «освіта вглиб» (поглиблена підготовка за заданим напрямом) [7; 9; 190] та «освіта вшир» (різностороння гуманітарна та природничо-наукова підготовка на основі оволодіння фундаментальними знаннями) [150; 157].

У більшості досліджень фундаменталізація освіти визначається як категорія освіченості людини. Її також розглядають як процес формування «фундаментально-знаннєвого» каркасу особистості (ядра системи знань індивіда), що забезпечить системність знань, цілісне сприйняття світу й людини в ньому, створення бази для професійної культури й майстерності [340].

В.В. Кондратьєв основою фундаменталізації освіти в технологічному університеті вважає неперервну математичну підготовку [142]. Д.Д. Ісхакова вважає, що важливим критерієм формування змісту процесу фундаменталізації вищої школи є його спрямованість на подолання технократичної асиметрії освіти, посилення її екологічної домінанти [107, 4]. Є.А. Тищенко умовою індивідуалізації інженерної підготовки вважає її фундаменталізацію на основі «взаємопроникнення, конвергенції гуманітарної та технічної культури» [401, 5].

На думку О.Г. Ростовцевої, фундаменталізації навчання сприятимуть міждисциплінарні зв’язки, науково-дослідна робота викладачів та студентів на стику фундаментальних та прикладних наук, введення у навчальні плани всіх спеціальностей природничо-наукових дисциплін [284, 13].

Ряд авторів [51; 191; 290; 420 та ін.] основою фундаменталізації вважають створення такої системи й структури освіти, пріоритетом якої є не прагматичні, вузькоспеціалізовані знання, а методологічно важливі, інваріантні знання, що сприяють цілісному сприйняттю наукової картини світу, інтелектуальному розвитку особистості та її адаптації у швидко мінливих соціально-економічних та інших умовах.

Е.Р. Соколова фундаментальну освіту трактує як освіту, засновану на фундаментальній природничо-науковій, гуманітарній, загальнопрофесійній та спеціальній підготовці, «що формує основи професійної та загальної культури сучасного фахівця, який володіє професійною мобільністю й креативним мисленням» [340].

На думку А.Б. Ольневої [227; 226], фундаменталізація навчання передбачає вивчення таких теоретичних відомостей різних наук, що пізніше, пройшовши випробування часом, стають ядром науки: «статус фундаментальності в на-

уці розпочинається з етапу розвитку науки «переднього краю», від гіпотези до статусу «ядра» науки. Наука «переднього краю» проходить апробацію на статус фундаментальної в розв'язанні прикладних задач, що мають різну професійну спрямованість. Окремо відзначаємо, що ... наявність спільної предметної області фундаментальної та варіативної складових змісту ... освіти призводить до появи основних нових професійних знань та вмінь майбутнього спеціаліста» [225, 10].

О.М. Новіков [216, 263] до *провідних напрямів фундаменталізації освіти* відносить:

– збереження ядра змісту, яке за своєю природою повинне бути консервативним;

– навчання базисних кваліфікацій – наскрізних умінь (базових компетентностей);

– посилення загальноосвітніх компонентів у професійних освітніх програмах;

– перехід до підготовки фахівців широкого профілю;

– пізню (на 2–3 курсі) профілізацію навчання;

– модульну будову змісту освіти;

– посилення наукового потенціалу навчальних закладів, створення науково-технологічних парків.

Підводячи підсумки, визначимо *основні ознаки* фундаменталізації освіти:

а) виділення універсальних, базових знань, виведенням їх на пріоритетні позиції та надання їм стрижневого значення для накопичення інших знань;

б) інтеграція освіти та науки;

в) перебудова процесу навчання на основі професійної та технологічної мобільності.

Визначаючи фундаменталізацію через сукупність взаємозалежних функцій (методологічної, професійно-орієнтувальної, розвивальної, прогностичної, інтегративної), можна виділити відповідні *шляхи її реалізації* в навчальному процесі:

– насичення змісту вищої освіти системними теоретичними знаннями, фундаментальними теоріями, концепціями, ідеями;

– домінування дослідницьких методів навчання, творчої діяльності, інтеграції ідей і методів науки, навчання й наукової творчості;

– саморозвиток студента як суб'єкта мобільної освітньої, професійної й науково-дослідної діяльності.

1.2. Інноваційність фундаментальної освіти

Основною метою реформування системи вищої освіти України є її орієнтація на науково-освітню інноваційну діяльність, в якій університет виступатиме як сучасний навчально-науковий інноваційний комплекс, що інтенсивно генерує та передає суспільству не лише нові знання, а й нові технології. В умовах інтеграції системи вищої освіти України у європейське та світове освітнє співтовариство саме функції трансферу знань та технологій разом із фундаменталізацією навчання створюють умови для експорту як знань, так і технологій.

Сучасний університет повинен мати власну інноваційну програму, ефективність якої визначається підготовкою спеціалістів, здатних комплексно аналізувати та розв'язувати актуальні проблеми соціально-економічного розвитку країни, поєднуючи свою діяльність із впровадженням інновацій. Інноваційні університети покликані розширити цілі функціонування вищої освіти від традиційних задач навчання та дослідження до розв'язування регіональних виробничо-економічних та кадрових проблем в загальному процесі становлення та розвитку національної інноваційної системи.

1.2.1. Вища освіта як фактор інноваційного розвитку. В Україні трансформаційні процеси в галузі вищої освіти, що розпочалися після здобуття незалежності, були обумовлені як особливостями суспільно-політичних процесів, так і складними економічними обставинами. Досить швидко стало зрозуміло, що від реформування вищої освіти буде залежати не тільки виживання вищих навчальних закладів, а й рівень розвитку самої держави. Тому було зроблено ряд спрямованих на перебудову вищої освіти кроків, які призвели до на-

ступних структурних перетворень, що впливають на сучасний стан системи українських вищих навчальних закладів:

1. Збільшення загального числа ВНЗ більш ніж у 2 рази. Якщо в 1990/1991 навчальному році їх було 149, то в 2002/2003 – вже 330 [557], а в 2007/2008 – 904 (з яких 553 – ВНЗ I–II рівня акредитації і 351 – ВНЗ III–IV рівня акредитації) [38].

2. Перехід на багатоступеневу систему підготовки. Сьогодні в Україні існують 4 освітньо-кваліфікаційних рівні: молодший фахівець, бакалавр, фахівець і магістр [101].

3. Введення чотирирівневої системи акредитації ВНЗ. Перший рівень – технікуми й училища, другий – коледжі, третій – інститути, четвертий – університети й академії [101]. За певних умов інститути можуть одержати й четвертий рівень акредитації.

4. Перехід на кредитно-модульну систему підготовки. Першим кроком цього переходу був проведений в 2004–2005 р. педагогічний експеримент із впровадження кредитно-модульної системи організації навчального процесу.

5. Орієнтація на інтеграцію в загальноєвропейський освітній простір через приєднання до Болонського процесу.

Проблеми, які постали перед вищими навчальними закладами у зв'язку з такою інтеграцією, змусили розглядати реформу вищої освіти головним чином через зміну методик навчання: «вищій освіті необхідно не тільки орієнтуватися на ринкові спеціальності, але й наповнювати зміст освіти новими матеріалами, впроваджувати сучасні технології навчання з високим рівнем інформатизації навчального процесу, виходити на творчі, ділові зв'язки із замовниками професіоналів» [100]. В результаті, як зазначає В.Ф. Паламарчук, у 2004 році більше половини (54%) всіх інновацій в галузі вищої освіти носили дидактичний характер [233].

Для подолання зазначених проблем Міністерством освіти та науки України у березні 2005 року було взято курс на «інноваційний трикутник»: освіта, науково-дослідна та виробнича, підприємницька діяльність [215].

Цінність вищих навчальних закладів як фактора економічного й соціального розвитку визначається не тільки їхніми освітніми завданнями, а й їхнім науковим потенціалом. Тому все більшого поширення (у тому числі – на державному рівні [124]) набуває питання про автономію університетів: у такий спосіб університети намагаються самостійно знайти шляхи участі в реальному інноваційному розвитку, тому що від цього залежить не тільки економічний потенціал суспільства, а й їхнє власне майбутнє. Справа в тому, вказує В. Нікітін [214], що автономія означає для вищих навчальних закладів не тільки фінансову незалежність, але й забезпечення можливостей:

– впливати на державну політику в сфері вищої освіти;

– розробляти власну політику розвитку взаємин з іншими вищими навчальними закладами як усередині країни, так і за її межами;

– розробляти й реалізовувати власні способи організації навчального процесу.

Як можна бачити, проблема автономії обговорюється не стільки як проблема незалежності, скільки як проблема активної участі університетів у суспільному й економічному розвитку. Університети повинні стати рівноправними учасниками інноваційних процесів, а не об’єктом державного регулювання з метою забезпечення цього інноваційного розвитку.

Міністерство освіти та науки України пропонує наступний ряд кроків з розвитку вищої освіти [100]:

– законодавчо розширити сферу працевлаштування випускників, підготовка яких проходила в рамках державних угод, на підприємствах, установах, організаціях будь-яких форм власності;

– розробити механізм економічного стимулювання підприємств, що створюють і бронюють робочі місця для молодих фахівців;

– підвищити ефективність цільової підготовки фахівців;

– посилити відповідальність підприємств, організацій і установ за забезпечення соціально-побутових умов для молодих фахівців;

– удосконалити механізм довгострокового пільгового кредитування навчання молоді у вищих навчальних закладах.

Вказані заходи багато в чому є реакцією на факт відтоку кваліфікованих кадрів з регіонів, особливо із сільської місцевості. Випускники не хочуть повертатися в регіони. У тому ж документі ([100]) фіксується, що розширення розмірів квот для абітурієнтів їхніх регіонів не вирішило проблему. Отже, держава повинна впровадити заходи (у першу чергу, законодавчі) для забезпечення повернення випускників у регіони, причому ці кроки повинні передбачати створення особливого середовища, у яке повинен потрапити випускник. Іншими словами, проблема не в тім, щоб допомогти випускникові й роботодавцеві «зустрітися», для чого необхідно побудувати механізми їхніх комунікацій, скільки в тім, щоб забезпечити випускника комфортним середовищем, звільнити його від необхідності пошуку на ринку праці.

Обговорюючи шляхи розв'язання проблем освітньої та наукової діяльності в Україні, що склалися на початок 2005 року, Л.М. Шульман [445] пропонує активізувати власний інноваційний потенціал, контролюючи при цьому не розробку того або іншого продукту, а його впровадження, доведення до виробництва й поширення на ринку. Цей підхід може бути застосований і до вищої освіти: необхідно здійснювати «супровід» процесу включення випускника вищого навчального закладу в професійну діяльність, а не контролювати його підготовку.

Ще один напрям перетворення вищої освіту в Україні на справжній фактор інноваційного розвитку суспільства, – це розширення розуміння інноваційного компонента. У більшості випадків під інноваціями розуміють економічні або технологічні інновації. Водночас у суспільному полі представлені інтереси різних груп, пов'язаних з установами вищої освіти: суспільної еліти, уряду, міністерств та відомств, професійних груп, бізнесу, навчальних закладів, національних груп, батьків, громадськості, студентів, людей з обмеженими можливостями доступу до освіти тощо. Різні групи по-різному оцінюють якість освіти, і з часом їхні пріоритети можуть змінюватися. Це означає, що установи вищої

освіти мають враховувати ці зміни й відповідним чином міняти свою політику [214].

Інноваційність вищої освіти в цьому випадку трактується не як випереджальна діяльність, а як процес постійних змін, тісно пов'язаних зі змінами в суспільстві та економіці, але не повністю ними обумовлених. Однак на сьогодні державна політика орієнтована на статичне уявлення про зв'язки суспільних груп та вищої освіти. Зокрема, основним елементом прив'язки змісту освіти до суспільних потреб вважаються державні стандарти, які покликані віддзеркалити ці потреби й бути основою організації навчального процесу. Держава замість ролі координатора відносин між системою вищих навчальних закладів і суспільством намагається відігравати роль ініціатора інновацій, визначаючи *інновації як «новостворені (застосовані) і (або) вдосконалені конкурентоздатні технології, продукцію або послуги, а також організаційно-технічні рішення виробничого, адміністративного, комерційного або іншого характеру, що істотно поліпшують структуру та якість виробництва і (або) соціальної сфери»* [139, 18].

На думку М.О. Зубрицької [104], вища освіта не зможе ефективно функціонувати без: 1) чітких економічних принципів її розвитку; 2) гнучких і результативних механізмів фінансового менеджменту; 3) стратегії підвищення соціально-економічного статусу педагогічних і наукових співробітників вищих навчальних закладів. В умовах дефіциту бюджету, з яким зіштовхується Україна, це означає необхідність надання вищим навчальним закладам можливості самим планувати свою економічну траєкторію, не очікуючи підтримки держави. На думку С.М. Ніколаєнка [215], для цього потрібно вжити наступних заходів:

– диверсифікувати джерела фінансування;

– комерціалізувати об'єкти інтелектуальної власності;

– поглибити співробітництво вищих навчальних закладів із промисловими підприємствами та із закордонними ВНЗ.

Виходячи з цих пропозицій, Інститут інноваційних технологій і змісту освіти визначив *інноваційну* діяльність як «*діяльність, що спрямована на вико-*

ристання і комерціалізацію результатів наукових досліджень та розробок і зумовлює випуск на ринок нових конкурентоздатних товарів і послуг» [139, 18]. Відповідно до даного трактування інноваційної діяльності вища освіта розглядається як одне із джерел інноваційного розвитку, а вищі навчальні заклади активно впливають на інноваційні процеси і тому обґрунтовано піднімаються питання якості підготовки фахівців, змісту освіти, доступності освіти, зв'язку освіти і науки тощо. У межах цього трактування центральне місце посідає орієнтація на майбутнє: інноваційність розуміється як постійне стратегічне планування, можливе в умовах фундаментальної освіти.

Враховуючи актуальність інновацій для досягнення цілей соціально-економічного розвитку, питання активізації інноваційної та інвестиційної діяльності є пріоритетними як для розвинених країн, так і для країн з перехідною економікою. Україна має досить потужний науково-технічний потенціал, значні досягнення в різних галузях науки, зокрема – у фундаментальних дослідженнях. Разом з тим, як і в багатьох країнах СНД, після 1991 року спостерігається зниження питомої ваги організацій, які здійснюють розробку та впровадження інновацій. Відкриття внутрішнього ринку для закордонних товарів та технологій призвело до падіння попиту на вітчизняну наукомістку продукцію. В існуючих умовах в якості однієї зі стратегій збереження наукового потенціалу вищої школи може розглядатися міжнародне співробітництво вищих навчальних закладів. Важливим напрямом міжнародного співробітництва може бути спільна діяльність ВНЗ в сфері підготовки кадрів для виробництва й експорту науково-технічної продукції. Разом з тим зрозуміло, що в довготривалій перспективі подібна стратегія є безперспективною: експорт технологічних розробок та передавання прав на них закордонним власникам сприяє посиленню технологічного лідерства потенційних конкурентів вітчизняних виробників на зовнішніх та внутрішніх ринках.

На основі аналізу досвіду країн СНД та Західної Європи, розглянемо основні напрями інноваційного розвитку вищої освіти в Україні.

1.2.1.1. Співробітництво в галузі підготовки фахівців для офшорного програмування. В останні роки серед фірм західних країн спостерігається тенденція делегування деяких операцій з розробки програмного забезпечення («аутсорсингу») у країни, що розвиваються, що забезпечує економію витрат за рахунок більш низької оплати праці.

За даними опитування 500 провідних акціонерних компаній США, включених у рейтинг Fortune 500, проведеного в 2004 році консалтинговою компанією Archstone разом із Центром міжнародної бізнес-освіти Університету Duke [537], очікувана мінімальна економія витрат у результаті аутсорсингу становила 20% для 88% опитаних компаній і 30% – для 55% компаній. В 66% випадків предметом аутсорсингу була розробка програмного забезпечення. За даними журналу Outsourcing [533], в 2005 році річний обсяг експорту програмного забезпечення оцінюється в суму порядку 20 млрд. доларів США, з яких більша частина припадає на Індію (близько 13 млрд.). Експорт програмного забезпечення інтенсивно розвивається й у країнах Східної Європи: Польщі (річний обсяг експорту – 22 млн. доларів США), Чехії (26 млн.), Росії (475 млн.), Україні (270 млн.), Білорусі (21 млн.). У 2005–2008 рр. темпи зростання ринку аутсорсингу складало в середньому 58%.

Якщо 3 роки тому основними центрами офшорного програмування були Київ, Харків, Дніпропетровськ, Донецьк, Одеса та Львів, то сьогодні спостерігається тенденція до відкриття невеликих за розміром представництв в інших обласних центрах та великих містах (зокрема, Херсоні, Луганську, Дніпродзержинську, Кривому Розі). Це пов’язано як з більш низьким рівнем вимог до заробітної платні, так і з поступовим вирівнюванням рівня підготовки за рахунок активного використання студентами регіональних ВНЗ можливості пройти магістерську підготовку або отримати другу вищу освіту в провідних ВНЗ України, де готують фахівців у галузі програмування.

Однією з особливостей офшорного програмування є досить гостра конкуренція за підготовлені кваліфіковані кадри, причому перевага віддається співробітникам до 30 років, що пов’язано з високим ризиком їх «професійного ви-

горяння». Тому інтенсивному розвитку експорту програмного забезпечення в Україні сприяє створення технопарків та програмістських колективів з широким залученням студентської молоді. Один із прикладів такої роботи започатковано в Херсонському державному університеті під керівництвом О.В. Співаковського: студенти-програмісти задовольняють при цьому не лише потреби зарубіжних, а й вітчизняних розробників (зокрема, МОН України) та внутрішні потреби ВНЗ. Організована у такий спосіб діяльність сприяє підвищенню практичної значущості навчання програмуванню.

Співробітництво у сфері підготовки кадрів для офшорного програмування може здійснюватися в процесі вивчення дисциплін спеціалізації. Конкретні форми взаємодії включають: проведення ярмарків вакансій, організацію виробничої практики студентів, ведення майстер-класів та читання спеціальних курсів співробітниками програмістських фірм.

Зазначимо, що надмірна популяризація офшорного програмування несе ряд ризиків як для системи освіти, так і для країни в цілому.

1. Відрив від потреб національної економіки в результаті надмірної орієнтації на потреби світового ринку: оскільки запити світової й національної економіки збігаються далеко не завжди, орієнтація на потреби зовнішніх споживачів може привести до ігнорування національних потреб.

Прагнення молодих людей до одержання високооплачуваної роботи призведе до підвищення конкурсів на відповідні спеціальності у ВНЗ та, відповідно, до розширення обсягів підготовки, в т.ч. – за рахунок держзамовлення. Проте широкі можливості впливу експортно-зорієнтованих ІТ-компаній на процес підготовки фахівців неминуче призведуть до переорієнтації вищих навчальних закладів на переважне врахування потреб цих компаній на шкоду потребам підприємств, зорієнтованим на внутрішній ринок.

2. Легальна діяльність в Україні фірм, що відрізняються не тільки високою прибутковістю, але й значними соціальними витратами: виробників програмного забезпечення для казино (в т.ч. – онлайнових) та комп’ютерних ігор.

Росту привабливості ігорного бізнесу в онлайні сприяє посилення державного регулювання традиційних казино в ряді країн. Практично повна заборона діяльності казино у Росії зумовила інтенсивний відтік відповідних фахівців до України та широкий розвиток мереж казино.

Участь подібних фірм у визначенні пріоритетів професійної підготовки у ВНЗ може створити серйозні проблеми як для держави, так і для вищої школи. Зокрема, в умовах лібералізації умов для здійснення грошових платежів за межами України онлайнові казино стають суттєвою соціальною загрозою.

3. Досвід інших країн Східної Європи в галузі офшорного програмування показує, що ІТ-фірми далеко не завжди готові вносити значний вклад у підготовку фахівців та виступати як довгострокові партнери в даній сфері. Наприклад, ряд офшорних розробників програмних продуктів, що діють у Польщі, Угорщині та Румунії, уже здійснюють перенесення власної діяльності до країн південно-східної Азії у зв'язку зі скороченням розриву в рівні оплати праці програмістів західних країн та країн східної Європи [476]. У країнах східної Європи зближення рівнів заробітних плат програмістів відбулося протягом 5–7 років, що призвело до ослаблення конкурентних переваг цих країн. Розвиток аналогічних процесів в Україні може привести до ослаблення зацікавленості розробників програмного забезпечення в партнерстві з національною вищою школою та зниженню потреби в національних кадрах.

1.2.1.2. Центри трансферу технологій при вищих навчальних закладах та їхня роль в експорті науково-технічних розробок. Відповідно до концепції розвитку науково-інноваційної діяльності в системі освіти Білорусі [145] відбувається перехід до нової моделі університету як навчально-науково-інноваційного комплексу, що об'єднує фундаментальну освіту, академічну науку з розвиненою мережею високотехнологічних інноваційних структур та малих підприємств.

Найважливішим структурним елементом білоруської інноваційної моделі є центри передавання технологій. Міжвузівський центр маркетингу науково-дослідних розробок підтримує постійно поновлювану централізовану базу да-

них розробок організацій Міністерства освіти Білорусі, веде базу даних інноваційних проектів, сприяє інноваційній діяльності. Метою діяльності центру є створення постійно діючої системи збирання та опрацювання маркетингових даних в галузі науково-дослідних розробок для класифікації, аналізу, оцінювання та поширення актуальних, своєчасних і точних повідомлень і даних, призначених для використання фахівцями з маркетингу НДР у ВНЗ при плануванні дій на ринку науково-технічної продукції [187].

За даними Міністерства освіти Білорусі [221], в 2006 році експорт науково-технічної продукції здійснювався 12 ВНЗ та Технопарком БНТУ «Метоліт» в 27 країн та склав близько 3 млн. доларів США. Зауважимо, що мова йде про досить значну для Білорусі суму, що перевищує загальний обсяг державного фінансування на відновлення матеріально-технічної бази вищої освіти (6,5 млрд. білоруських рублів) та еквівалентна одній шостій частині загального обсягу державного фінансування науки у вищих навчальних закладах. На жаль, наявних даних недостатньо для підрахунку частки вартості готової продукції, виробленої на основі інноваційних технологій, розроблених ВНЗ Білорусі. Непрямі дані говорять про те, що у високотехнологічному експорті переважають науково-технічні розробки низького ступеня завершеності, реалізовані за рубіж з передаванням значної частини або всіх прав інтелектуальної власності на неї. Це може бути викликано, зокрема, відсутністю у вищих навчальних закладів засобів для патентування об’єктів промислової власності за рубежем, низькою правовою культурою з використання об’єктів інтелектуальної власності. Все це призводить до зниження ефективності використання науково-технічного потенціалу вищої освіти та потенційно втраченому прибутку. Найчастіше експортовані в такий спосіб розробки реімпортуються згодом у вигляді готової високотехнологічної продукції.

І хоча експорт науково-технічних розробок для впровадження за рубежем може служити однією зі стратегій, що надасть вітчизняній вищій школі можливість зберегти свій науково-технічний потенціал, не затребуваний у національній економіці, дана стратегія не може вважатися ефективною навіть в середньо-

строковій перспективі, тому що вона неминуче призводить до поглиблення розриву між наукою та стратегічними національними інтересами.

Тому Департамент міжнародного співробітництва МОН України виділив ряд пріоритетних завдань з розвитку міжнародного співробітництва в галузі науково-інноваційної діяльності вищих навчальних закладів. Так, у діяльності ВНЗ на міжнародному ринку науково-технічної продукції першочергова увага приділяється стимулюванню експорту кінцевої продукції, заснованої на впровадженні передових технологій та постачанню науково-технічних розробок з високим ступенем завершеності. Таке державне регулювання надасть можливість максимально зберегти права інтелектуальної власності на розробки усередині країни.

У складі МОН України створено Департамент інновацій та трансферу технологій, цілями та завданнями якого є:

– участь у формуванні та забезпеченні реалізації державної політики у сфері інноваційної діяльності та трансферу технологій;

– розроблення нормативно-правових актів щодо забезпечення розвитку сфер інноваційної діяльності та трансферу технологій;

– створення сприятливих умов для інноваційної діяльності та діяльності у сфері трансферу технологій;

– формування стратегічних та середньострокових пріоритетних напрямів інноваційної діяльності, державних цільових програм та здійснення моніторингу їх реалізації;

– формування інноваційної інфраструктури та інфраструктури у сфері трансферу технологій;

– організація роботи технологічних парків, проведення експертизи, реєстрації інноваційних проектів та проектів технологічних парків, ведення Державного реєстру, забезпечення моніторингу і контролю реалізації проектів;

– сприяння процесу комерціалізації об’єктів інтелектуальної власності в інноваційній сфері;

– здійснення заходів щодо популяризації інноваційної діяльності;

– організація та методологічне забезпечення публічних заходів популяризації державної політики у сфері інновацій та трансферу технологій;

– координація діяльності відповідних структур і підрозділів міністерств та відомств з питань інноваційної діяльності та діяльності у сфері трансферу технологій.

Основним документом, яким керується Департамент інновацій та трансферу технологій, є Закон України «Про державне регулювання діяльності у сфері трансферу технологій» від 14 вересня 2006 року № 143-V. Відповідний законопроект, розроблений Міністерством освіти і науки України, було внесено Кабінетом Міністрів України до Верховної Ради ще у 2003 році. Закон після трьох вето було підписано Президентом України 2 жовтня 2006 року та введено в дію з 6 жовтня (з дня опублікування), крім статей 20 та 22, що набирають чинності з 1 січня 2007 року та статей 10, 12, 13 та 19, що набирали чинності з 6 квітня 2007 року.

Цим Законом визначаються правові, економічні, організаційні та фінансові засади державного регулювання діяльності у сфері трансферу технологій: він спрямований на забезпечення ефективного використання науково-технічного та інтелектуального потенціалу України, технологічності виробництва продукції, охорони майнових прав на вітчизняні технології на території держав, де планується або здійснюється їх використання, розширення міжнародного науково-технічного співробітництва у цій сфері.

Положення Закону спрямовані на *одночасний захист в процесі створення та передавання технологій інтересів:* держави, авторів технологій, підприємств, установ та організацій, які здійснюють передавання технологій; підприємств установ та організацій, які впроваджують інноваційні технології, що відповідають пріоритетним напрямам інноваційної діяльності загальнодержавного рівня.

Захист інтересів держави забезпечено завдяки:

1) визначенню випадків та процедури проведення обов’язкової державної експертизи технологій (для технологій, для яких суб’єктами їх трансферу пе-

редбачено отримання цільових субсидій, визначених цим Законом; для технологій, що плануються для використання в Україні за рахунок державних коштів, якщо сума їх закупівлі дорівнює або перевищує розмір суми, визначеної для відповідних процедур закупівель Законом України «Про закупівлю товарів, робіт і послуг за державні кошти») – стаття 12 Закону;

2) визначенню процедури державної реєстрації договорів про трансфер технологій на підставі зазначеної державної експертизи – стаття 13 Закону;

3) започаткування державної акредитації фізичних та юридичних осіб для здійснення на постійній та/або професійній основі посередницької діяльності у сфері трансферу технологій (технологічних брокерів) – стаття 15 Закону;

4) встановленню випадків обмеження щодо укладання договорів про трансфер технологій – стаття 18 Закону;

5) визначенню напрямів використання коштів, одержаних у результаті трансферу технологій, створених або придбаних за рахунок коштів Державного бюджету України, виключно для розвитку вітчизняної науково-технологічної сфери – стаття 20.

Захист інтересів авторів технологій має бути забезпечений завдяки реалізації положень статті 19, де передбачено вимоги щодо:

1) укладання договору між автором технології та підприємством, установою чи організацією, де створено технологію та/або куди здійснюється її трансфер, в якому мають бути визначені майнові права, що передаються за цим договором, умови виплати та розмір винагороди за передавання і використання майнових прав на технологію;

2) виплати винагороди автору технології у випадку отримання підприємством ліцензійних платежів від надання ліцензій на його технологію;

3) визначення розміру, ставки, порядку та умов виплати винагороди авторам технологій;

4) встановлення Кабінетом Міністрів України мінімальної ставки винагороди авторам технологій.

Стимулювання підприємств, установ та організацій, які здійснюють передавання технологій, та які впроваджують інноваційні технології, має бути забезпечено завдяки:

1) встановленню Кабінетом Міністрів України мінімальної ставки винагороди особам, які здійснюють трансфер технологій (пункт 5 статті 19);

2) цільовому субсидіюванню трансферу технологій (подібно до процедури, існуючої для проектів технологічних парків) – на суми податку на прибуток підприємств, одержаного від впровадження зазначених технологій, нарахованої за період та у порядку, що встановлені Законом України «Про оподаткування прибутку підприємств» та суми ввізного мита, що нараховується згідно з митним законодавством України при ввезенні в Україну для реалізації проектів трансферу технологій, устаткування, обладнання та комплектуючих, а також матеріалів, що не виробляються в Україні (стаття 22 Закону);

3) встановленню можливостей для надання державних гарантій щодо погашення кредитів комерційних банків, наданих для придбання технологій та їх складових, підприємствам, що належать до сфери управління центральних органів виконавчої влади, Національної та галузевих академій наук (стаття 21).

Державна політика у сфері інновацій та трансферу технологій в Україні сьогодні спрямована на заохочення інноваційної активності, організаційну модернізацію науково-технологічної сфери та формування мотивації суб'єктів господарювання до інновацій, вдосконалення системи державного регулювання сфери трансферу технологій, створення умов для ефективного і повноцінного розвитку інноваційної інфраструктури та інфраструктури трансферу технологій, сприяння комерціалізації результатів науково-технічних досліджень і розробок.

1.2.1.3. Створення інноваційних університетів. В проекті змін до Закону України «Про інноваційну діяльність» [102] визначаються такі форми інноваційної діяльності:

а) *інноваційне підприємство* – підприємство (об'єднання підприємств), що розробляє, виробляє і реалізує інноваційні продукти і (або) продукцію чи

послуги, обсяг яких у грошовому вимірі перевищує 70 відсотків його загального обсягу продукції і (або) послуг;

б) *технополіс* – створене на певній території об'єднання наукових, дослідно-конструкторських і технологічних організацій, дослідних виробництв, яке охоплює розробку, впровадження та виробництво інноваційного продукту, інноваційної продукції і надання послуг суб'єктам інноваційної діяльності;

в) *центр трансферу технологій* – підприємство або структурний підрозділ, створений при вищих навчальних закладах, наукових установах та інших підприємствах з метою надання послуг щодо передавання технологій та реалізації на їх основі інноваційних проектів;

г) *інноваційний бізнес-інкубатор* – складова інноваційної інфраструктури, створена для підприємців і малих новостворених підприємств, діяльність яких спрямована на впровадження інноваційних ідей та винаходів на початкових етапах їх комерціалізації і реалізації на цій основі інноваційних проектів, а також надання послуг суб'єктам інноваційної діяльності;

д) *інноваційно-технологічний кластер* – об'єднання географічно локалізованих підприємств, що пов'язані виробничими зв'язками з метою створення інноваційної продукції та надання послуг суб'єктам інноваційної діяльності;

е) *інноваційний центр* – це заклад інноваційної діяльності із розвиненою інноваційною інфраструктурою, призначений для сприяння розвитку інноваційної діяльності, співробітництво і кооперацію між дослідниками і виробництвом, надання послуг суб'єктам інноваційної діяльності, підвищення кваліфікації персоналу в галузі інноваційного менеджменту.

У відповідності до частини 2 статті 16 розділу III Закону, «інноваційне підприємство може функціонувати у вигляді інноваційного центру, інноваційного бізнес-інкубатора, технополісу, технопарку, центру трансферу технологій, інноваційно-технологічного кластеру тощо». Відкритість визначення інноваційного підприємства надає можливість пропонувати та досліджувати й інші його форми, такі як *інноваційний університет* – «університет, що сприяє прискореному розвитку соціуму, тобто функціонуванню інноваційного суспільства

за рахунок інтенсивного та масштабного передавання нових, згенерованих в університеті знань, включаючи технології в найрізноманітніших (природничо-наукових, технічних та соціально-гуманітарних) галузях людської діяльності» [357].

Визначальна риса (і одночасно перша функція) інноваційного університету – наявність системи трансферу знань, що включає організаційну підсистему трансферу технологій.

Відмінність від традиційної функції університету – генерації нових знань та введення їх в обіг – полягає в тому, що суспільство, засноване на знанні, не тільки вимагає значно більш швидкого використання досягнень науки в практиці, але й породжує технології, що прискорюють цей процес. Створення гнучкої, мобільної системи трансферу знань забезпечує умови для розвитку ланцюжка «університет – підприємства високих технологій – бізнес» з конкретними користувачами результатів наукових досліджень та зворотним зв'язком з університетом як з точки зору напряму досліджень, так і якості підготовки фахівців.

Друга найважливіша функція інноваційного університету, прямо пов'язана з першою, – функція інтегратора знань: університет стає фактично центром інтелектуального потенціалу суспільства. В умовах глобалізації функція трансферу знань для інноваційного університету набуває самостійного значення як можливість експорту вітчизняної освіти та технологій.

Функції інноваційного університету характеризуються в основному ринковими властивостями. Ймовірнісний характер ринкового попиту, що вимагає мобільної, гнучкої поведінки в діяльності підрозділів університету, найуспішніше може бути реалізований саме в міждисциплінарному полі, на рівні співробітництва з іншими кафедрами, факультетами та науково-дослідними структурами. Подібний механізм функціонального об'єднання найкращих ресурсів та інтелектуального потенціалу пропонується в пріоритетному національному проекті «Освіта», реалізацію якого розпочато в Росії в 2005 році. В основу змін

традиційного механізму управління освітою покладений проектно-орієнтований підхід.

Успішне функціонування інноваційного університету спрямоване, в остаточному підсумку, на підвищення якості освіти, що в останні роки все більше пов'язується з компетентнісним підходом.

1.2.1.4. Розробка стандартів вищої освіти на основі компетентнісного підходу. Сучасні процеси модернізації вищої школи України багато в чому сприймаються та оцінюються через призму компетентнісного підходу та пов'язаному з ним процесу розробки складових системи галузевих стандартів вищої освіти.

Указом Президента України від 4 липня 2005 року № 1013 «Про невідкладні заходи щодо забезпечення функціонування та розвитку освіти в Україні» (п. 7) визначено низку заходів, спрямованих на реалізацію в Україні положень Болонської декларації, зокрема, з розроблення та затвердження нових галузевих стандартів вищої освіти [411].

Концепція Державної програми розвитку освіти на 2006-2010 роки [146] одним із завдань визначає забезпечення доступу до високоякісної вищої освіти та мобільності випускників вищих навчальних закладів на ринку праці, шляхом інтеграції ВНЗ різних рівнів акредитації, наукових установ та підприємств, впровадження гнучких освітніх програм та інформаційних технологій навчання відповідно до вимог Болонської декларації.

Порівняльний аналіз сучасних зарубіжних освітніх систем і технологій та наукових розробок вітчизняних педагогів [365] надає можливість зробити висновок про те, що основними шляхами розвитку системи освіти є:

– постійне оновлення змісту вищої освіти з метою більш повного забезпечення потреб суспільства, у тому числі й майбутніх;

– орієнтація на забезпечення конкурентоспроможності випускників на ринку праці;

– формування у студента професійних та соціально-особистісних якостей, що надають йому можливість повністю реалізувати свій інтелектуальний потенціал;

– поглиблення автономії та забезпечення академічної незалежності закладів освіти, посилення їх зв'язків із роботодавцями, як основними замовниками фахівців;

– розширення академічної мобільності студентів, що надає можливість повніше реалізовувати їхній інтелектуальний потенціал.

Праця фахівця будь-якої спеціальності спрямована на певний об'єкт (предмет) діяльності й полягає у виконанні визначених виробничих функцій. Вона пов'язана з конкретною системою діяльності та реалізується за допомогою відповідної системи засобів цієї діяльності. Таким чином, праця фахівця пов'язана з конкретною технологією.

В умовах перманентної науково-технологічної революції життєвий цикл сучасних технологій стає меншим, ніж термін професійної діяльності фахівця. За цих умов «домінуючим в освіті стає формування здатності фахівця на основі відповідної фундаментальної освіти перебудовувати систему власної професійної діяльності з урахуванням соціально значущих цілей та нормативних обмежень – тобто формування особистісних характеристик майбутнього фахівця» [139, 4]. Якщо визначити за головне призначення системи вищої освіти підготовку такого фахівця, то процес навчання доцільно організовувати у такий спосіб, щоб забезпечувався гармонійний розвиток особистості майбутнього фахівця. Засобом формування особистості при цьому стають освітні технології, продуктом діяльності педагогічних колективів – особистість випускника ВНЗ, який повинен бути компетентним не лише в професійній галузі, але й мати активну життєву позицію, високий рівень громадянської свідомості, бути компетентним при вирішенні проблем, які ставить перед ним життя.

Отже, перехід до нового покоління галузевих стандартів вищої освіти на основі фундаменталізації навчання та компетентнісного підходу є необхідним етапом на шляху реформування системи вищої освіти в Україні, а застосування

компетентнісного підходу до створення галузевих стандартів вищої освіти створює умови для наближення фундаментальної освіти до потреб та вимог ринку праці, подальшого розвитку освітніх технологій та системи освіти в цілому.

Таким чином, можна зробити висновок, що ***компетентнісний підхід до навчання є одним із засобів його фундаменталізації***.

1.2.2. Фундаменталізація як основа розвитку інноваційної вищої освіти. У відповідності із дослідженнями А.А. Аданнікова [2], С.А. Баляєвої [13], А.Б. Ольневої [226], О.В. Сергєєва [326], розвиток вищої освіти має бути спрямований на:

– оновлення змістової бази навчання майбутніх фахівців природничо-математичних та технічних спеціальностей;

– розвиток здатності фахівця адаптуватися до високих темпів науково-технічного прогресу;

– формування у студентів творчого фахового мислення;

– розвиток здатності фахівця «згортати» наростаючі потоки професійно-значущих повідомлень до легко доступних для огляду обсягів;

– підвищення професійної мобільності випускника ВНЗ;

– уніфікацію змісту й рівня підготовки фахівців у різних ВНЗ.

Покажемо, що всі перераховані напрями розвитку вищої освіти вимагають фундаменталізації навчального процесу на основі інноваційних підходів.

1.2.2.1. Оновлення змістової бази навчання майбутніх фахівців природничо-математичних та технічних спеціальностей. Професійна знаннєва база навчання представлена загальнопрофесійними та спеціальними дисциплінами навчального плану. Кожна із цих дисциплін є адаптованою до певного контингенту слухачів інформаційною моделлю відповідної прикладної науки, яка, в свою чергу, є модифікованим варіантом тієї чи іншої фундаментальної науки. У ході такої модифікації фундаментальна наука переорієнтовується на частинні прикладні цілі, її основні закони відображаються у відповідні технології, а загальні рівняння перетворюються в розрахункові формули (наприклад, так з електродинаміки виник курс теоретичних основ електротехніки). Іноді прикладна нау-

ка являє собою цілий науково-технічний напрям і виникає на основі інтеграції кількох фундаментальних наук (наприклад, металургія поєднує фізику твердого тіла, фізику рідин, термодинаміку, хімію та ін.). Очевидно, що різні прикладні науки й навчальні дисципліни пов'язані з різними фундаментальними науками (наприклад, для інформатики важливі математичні основи її теорії та фізичні основи інструментальної бази, що забезпечують одержання, опрацювання, зберігання, подання, передавання різноманітних повідомлень).

Тому практично вся знаннєва база навчання фахівця з прикладних наук досить чутлива до досягнень фундаментальних наук: чим швидше включаються новітні досягнення відповідних фундаментальних наук у програми прикладних курсів, тим більш високою і сучасною буде підготовка фахівця за будь-якою спеціальністю.

1.2.2.2. Розвиток здатності фахівця адаптуватися до високих темпів науково-технічного прогресу. Однією з проблем сучасної вищої технічної освіти є відсутність механізмів, що забезпечують адекватність реалізованих освітніх програм поточним цілям і завданням підготовки фахівців, здатних брати активну участь у прискоренні науково-технічного прогресу. На жаль, більшість викладачів ВНЗ безпосередньо не беруть участь у процесі виробництва та не виконують наукові або конструкторські розробки зі свого фаху, лише зрідка прилучаючись до реального процесу розвитку техніки. Основна частина повсякденних науково-технічних досягнень забезпечується раціоналізаторською, винахідницькою, дослідницькою й конструкторською роботою професіоналів, що постійно займаються питаннями виробництва безпосередньо на виробництві, у профільних конструкторських структурах, галузевих НДІ, технопарках і т.д., тому викладач одержує повідомлення про ці досягнення з деяким запізненням.

Крім того, передати студентам новітні науково-технічні здобутки досить непросто: викладачеві необхідно відповідні повідомлення не тільки вчасно одержати й осмислити самому, але й перетворити їх у навчальний матеріал відповідного курсу, доступний для розуміння студентів. Для цього зазначений матеріал повинен бути несуперечливо вбудований у структуру діючого навчального

плану та забезпечений необхідними методичними розробками, навчально-методичною літературою, лабораторним устаткуванням тощо. Природно, що до моменту готовності всього перерахованого змістова частина розглянутого матеріалу вже застаріває, а це зумовлює постійне відставання підготовки фахівців від сучасного виробництва.

Сказане подано схематично на рис. 1.1 кривими II і III, що відображають нарощування новацій у виробництві $\xi(t)$ та оновлення навчального матеріалу $\eta(t)$ з часом t. Інтервал a_2–a_3 характеризує відставання навчання від виробництва в момент t_1. В сучасних умовах остаточна адаптація молодого фахівця до рівня виробництва відбувається вже на підприємстві, вимагає додаткового часу й засобів, що природно, не сприяє прискоренню науково-технічного прогресу.

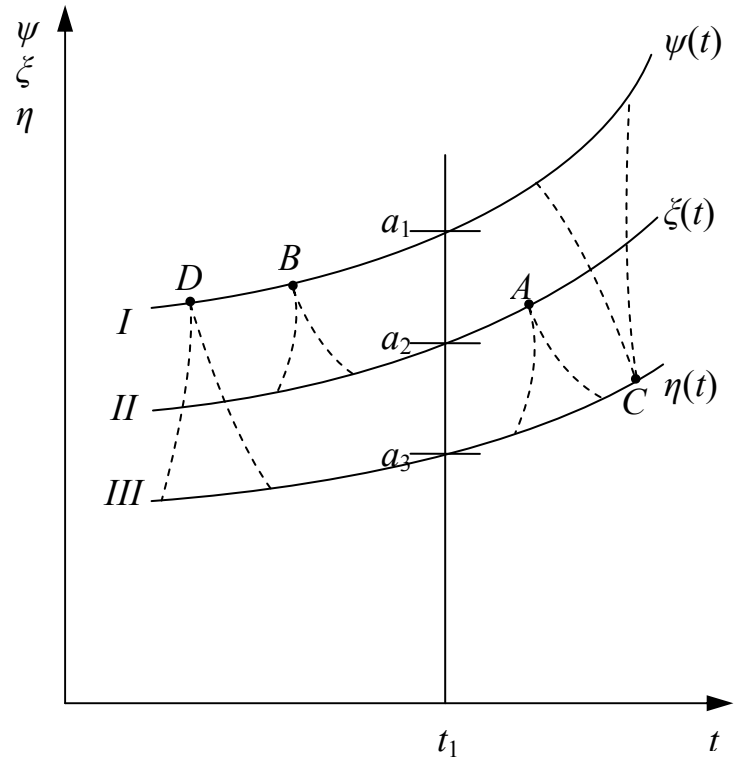


Рис. 1.1. Схематичне подання зростання обсягу досягнень фундаментальних наук $\psi(t)$ (*I*), новацій у виробництві $\xi(t)$ (*II*) та оновлення змісту навчального матеріалу у вищих навчальних закладах $\eta(t)$ (*III*)

Описана ситуація призводить до підготовки «фахівця вчорашнього дня», який легко адаптується до застаріваючого виробництва, однак не підготовлений до швидких радикальних змін у виробництві та ефективної участі у науково-

технічному прогресі. Вихід з цього «освітнього тупика» – у переході до інноваційної, випереджаючої освіти, що забезпечує фахівцеві можливості ефективно вирішувати принципово нові завдання. Для цього ВНЗ, крім підготовки висококваліфікованого фахівця повинні формувати широкоосвічену, творчу й системно мислячу особистість. Ця вимога нездійсненна без істотного посилення фундаментальної складової фахової підготовки.

Наповнення фундаментальних наук новими знаннями іде надзвичайно швидко, що умовно представлено залежністю $\psi(t)$ на рис. 1.1 (крива *I*). Не всі досягнення науки одразу ж використовуються у виробництві: інтервал a_1–a_2 характеризує відставання виробництва від науки в момент t_1. Криві на рис. 1.1 досить умовні – вони лише у середньому відображають головне у взаємозв'язках між фундаментальною наукою, виробничою практикою і вищою освітою. Криві II і III стикаються в тих точках, що відповідають випадкам впровадження у виробництво розробок, виконаних у деякому ВНЗ (точка *A*) – такий заклад має можливість не відстати від виробництва у висвітленні студентам власних розробок. Криві II і I стикаються в точках, які відповідають випадкам «миттєвого» впровадження результатів фундаментальних досліджень (точка *B*) – такі дослідження звичайно проводяться на замовлення виробництва. Ю.В. Триус вказує на особливу важливість точки *C*, що відповідає впровадженню результатів фундаментальних досліджень у навчальний процес, та точки *D*, яка характеризує проведення фундаментальних досліджень у дослідницькому ВНЗ.

Випереджаюча освіта має спиратися на те, що свідомо випереджає виробництво, – на фундаментальну науку. Таким чином, освіта, продовжуючи «підтягуватися» до рівня сучасного виробництва, повинна одночасно залучати до навчального процесу найсучасніші досягнення фундаментальних наук, досить глибоко знайомити з ними студентів і навчати студента «уловлювати» паростки нового в сфері своєї майбутньої професійної діяльності.

На рис. 1.2 показано взаємозв'язки між фундаментальними науками, виробництвом і освітою. У центральній частині рисунка стрілками позначені на-

прями циркуляції наукових досягнень, що породжує, стимулює та розвиває виробництво, науку та вищу освіту.

Рис. 1.2. Взаємозв'язок між фундаментальними науками, виробництвом та вищою освітою

Дві стрілки під номером 1 характеризують взаємозв'язки між потребами суспільства й потребами фундаментальних наук. У процесі постійної адаптації до умов навколишнього середовища люди досліджують оточуючий світ засобами фундаментальних наук, тому суспільство ставить все нові завдання перед

фундаментальними науками (стрілка 1, спрямована донизу). У цьому проявляється соціальне замовлення з боку суспільства. У свою чергу, розвиток фундаментальних наук надає людині можливість побачити нові проблеми, які варто поставити суспільству перед фундаментальними науками (стрілка 1, спрямована догори), що створює в суспільстві усвідомлення того, які наукові завдання є найбільш актуальні.

Стимульовані первинними запитами суспільства фахівці з фундаментальних наук досліджують різноманіття природних явищ і матеріальних структур. Суспільство через прикладні напрями фундаментальних наук установлює потенційну перспективність отриманих результатів. Взаємозв'язки між фундаментальними науками й прикладними напрямами фундаментальних наук позначені стрілками 2.

Припустимо, що у більшості випадків відкриття нового в науці (у момент часу t_0) випереджає усвідомлення (у момент t_1) його практичної значущості, тобто $t_1 > t_0$. Далі, у деякий момент часу t_2 естафету приймає «технічна наука» – модифікація відповідних розділів фундаментальних наук, орієнтована на розв'язання прикладних завдань. Для здійснення цієї модифікації потрібен певний час і тому завжди $t_2 > t_1$. Стрілки 3 на рис. 1.2 відображають взаємозв'язки між «технічними науками» та прикладними напрямами фундаментальних наук. У ВНЗ «технічні науки» перетворюються в один з навчальних курсів загальнопрофесійного або спеціального блоку. Цей процес вимагає певного часу й завершується з деяким запізненням $t_3 - t_2$.

В результаті цього повне навчально-методичне забезпечення підготовки висококваліфікованих фахівців (підтримуюча освіта) запізнюється в порівнянні із часом створення нової техніки. Однак, якщо в основу підготовки навчально-методичного матеріалу будуть покладені ті ж відомості, що породжують «технічні науки», то зазначений матеріал буде готовим до моменту t_3'. При цьому цілком досяжні умови, коли $t_3' < t_2$. Це означає, що при досить глибокій фундаменталізації вищої освіти вищі навчальні заклади зможуть адаптувати своїх випускників не тільки до сучасного, але й до майбутнього виробництва.

1.2.2.3. Формування у студентів творчого фахового стилю мислення. Здатність адаптуватися до високих темпів науково-технічного прогресу – необхідна, але недостатня умова для плідної участі людини в цьому процесі. Така здатність може ґрунтуватися на пасивному володінні фундаментальними знаннями, що лежать в основі технічного прогресу. Для активної участі в ньому важливо, щоб фахівець мав ще особливе професійне мислення, головними характеристиками якого є критичне ставлення до досягнутого, здатність запропонувати нове й уміння врахувати впливи всіх значимих внутрішніх і зовнішніх факторів, що забезпечують надійне функціонування запропонованого. Іншими словами, професійне мислення має включати в себе критичність, творчість, системність. Критичність розкриває потребу в новації, творчість її породжує, системність мислення гарантує якість і надійність новації. Крім того, всі етапи діяльності фахівця повинні перевірятися на відповідність законам фундаментальної науки. Знання цих законів також є обов'язковим атрибутом творчого фахового стилю мислення. Будь-яке протиріччя пропонованої новації якому-небудь із законів природи робить цю новацію принципово нереалізовною; перетворює інженерний проект на «прожект».

З.О. Решетова відзначає, що, характеризуючи професійне мислення, часто мають на увазі певні особливості мислення фахівця, що дозволяють йому успішно розв'язувати професійні задачі на високому рівні майстерності: швидко, точно та оригінально розв'язувати як ординарні, так й неординарні задачі [282]. Саме такий тип професійного мислення будемо назвати *творчим фаховим стилем мислення*.

Розвинений творчий фаховий стиль мислення вдосконалюється протягом всієї професійної діяльності, але його основи закладаються знаннями фундаментальних наук, в яких розроблено потужний арсенал методів вирішення складних проблем, що виникають в процесі пізнання: методи аналізу й синтезу, індукції й дедукції, реконструкції, моделювання і т.д.

1.2.2.4. Розвиток здатності фахівця «згортати» наростаючі потоки професійно-значущих повідомлень до легко доступних для огляду обсягів. Пото-

ки наукових і технічних повідомлень прискорено зростають і досягли вже неоглядних обсягів. Проте фахівець не може розраховувати на успіх, якщо він не здатен виявляти в зазначеному потоці професійно важливі повідомлення.

Стежити відразу за всім потоком відомостей можна, лише «згорнувши» його до доступних для огляду обсягів. До основних методів згортання повідомлень відносяться широко відомі методи систематизації й класифікації знань, концептуальний підхід та виявлення ознак ієрархічності структур, їхніх складових елементів тощо. Сьогодні на особливу увагу заслуговує об'єктно-орієнтований підхід як універсальний засіб дослідження складних систем (цікавий приклад його застосування до добору змісту навчання інформатики наводить О.Г. Степанов [353; 354]).

1.2.2.5. Підвищення професійної мобільності випускників вищих навчальних закладів. Під *професійною мобільністю* фахівця розуміється його здатність без великих часових та фінансових витрат змінювати спрямованість своєї професійної діяльності. У сучасному світі ця якість фахівця вирішальним чином визначає його життєве благополуччя, адже вузька спеціалізація в рамках підтримуючої освіти не може забезпечити випускникові вищого навчального закладу професійну мобільність.

Надмобільність – це здатність, переучуючись у короткі терміни, професійно функціонувати як за новим, так і за попереднім фахом. Такий рівень професійної мобільності досяжний для людини, котра володіє розвиненим фаховим мисленням та знаннями з фундаментальних наук.

Практично досяжна професійна мобільність рядового фахівця, що закінчив типовий вищий навчальний заклад із посиленою фундаментальною підготовкою, обмежується групою споріднених спеціальностей, що розрізняються за фундаментальними основами виробництва: виробництво інформаційних ресурсів, енергії, сировини, матеріалів, виробів, транспорт. Отже, кількасот існуючих спеціальностей обслуговують усього шість. Так, всі спеціальності інформаційної спрямованості спираються на математику (теорія інформатики) та на квантову фізику, фізику твердого тіла, теорію росту й розчинення кристалів, фізику

прискорювачів елементарних часток, хімію, термодинаміку тощо (інструментальна база інформатики). Знання фундаментальних основ спеціальностей, що обслуговують перераховані виробництва, полегшує переходи від однієї спеціальності до іншої в сфері кожного із цих виробництв: наприклад, якщо фахівець в галузі автоматизації добре опанував фундаментальні основи своєї галузевої автоматизації, то йому забезпечена міжгалузева мобільність, адже у цей час практично немає галузей, у яких би не знадобився фахівець із автоматизації.

1.2.2.6. Уніфікація змісту й рівня підготовки фахівців. Незважаючи на успішне впровадження в багатьох ВНЗ України кредитно-модульної системи навчання, випускники вітчизняних ВНЗ (так само, як і європейських) відстають від американських за професійною мобільністю та здатністю створювати трудову «самозайнятість». Одна із цілей Болонського процесу – ліквідувати це відставання, для чого передбачається уніфікація змісту й рівня підготовки випускників ВНЗ країн учасників Болонського процесу. Система залікових одиниць, на якій акцентувалась увага при впровадженні кредитно-модульної системи навчання, є лише засобом уніфікації кількісного обліку, а для забезпечення порівняльної якості освіти необхідно вводити взаємно визнані методології перевірки знань (сьогодні це питання активно розробляється Українським центром оцінювання якості освіти). Працює на уніфікацію й фундаментальний характер вищої освіти, який є одним із пріоритетів Болонського процесу (див. п. 1.1.1).

1.2.3. Регіональний інноваційний університетський комплекс як основа системи неперервної фундаментальної освіти. «Методичні рекомендації щодо розроблення середньострокових пріоритетних напрямів інноваційної діяльності галузевого та регіонального рівня», затверджених наказом Міністерства освіти і науки України, Міністерства економіки та з питань європейської інтеграції України, Міністерства промислової політики України, Міністерства фінансів України та Національної академії наук України від 9 липня 2003 р. № 442/279/180/298/449, визначають наступні принципи розроблення середньострокових пріоритетних напрямів інноваційної діяльності галузевого та регіонального рівня:

– *цілісності та взаємної узгодженості*, що передбачає формування середньострокових пріоритетних напрямів інноваційної діяльності галузевого та регіонального рівнів, узгоджених зі стратегічними та середньостроковими пріоритетними напрямами інноваційної діяльності загальнодержавного рівня, а також узгоджених між собою;

– *випереджаючого розвитку*, що означає розроблення середньострокових пріоритетних напрямів інноваційної діяльності галузевого та регіонального рівнів на основі аналізу і прогнозування можливостей та перспектив розвитку галузі та регіону з використанням науково-обґрунтованих підходів, а також з врахуванням перспективних потреб економіки і соціальної сфери на основі середньострокових прогнозів економічного і соціального розвитку;

– *відповідності*, що полягає у розробленні та формуванні такої системи пріоритетів, яка б відповідала конкретним умовам технологічної реструктуризації галузевих (регіональних) виробництв та можливостям ресурсного забезпечення реалізації пріоритетних напрямів. Останнє включає фінансово-економічні, інтелектуальні, матеріально-технічні ресурси;

– *гласності*, що полягає у забезпеченні обговорення на галузевому та регіональному рівнях відповідних середньострокових пріоритетних напрямів інноваційної діяльності, а затверджені середньострокові пріоритетні напрями інноваційної діяльності галузевого та регіонального рівнів мають бути доведені до суб'єктів інноваційної діяльності у галузі та в регіоні як орієнтири для розроблення ними власних планів, програм, та проектів.

У відповідності до вказаних рекомендацій, інноваційні процеси, що роблять вищу освіту фундаментом виробництва, зміни в характері й у змісті виробничої діяльності, підвищення ролі особистості в сучасному виробництві, швидка інфляція науково-технічних здобутків, підвищення конкурентоздатності на ринку праці викликали до життя нову форму навчально-наукового виробничого комплексу: *інноваційний університетський навчальний комплекс*.

Один з найбільш відомих регіональних інноваційних університетських комплексів – Криворізький навчально-науковий виробничий комплекс Націо-

нальної металургійної академії України, створений у 2004 р. До складу комплексу входять середні загальноосвітні заклади (зокрема, Криворізький металургійний ліцей, Саксаганський природничо-науковий ліцей, Криворізький колегіум №81), професійно-технічні училища, технікуми (коксохімічний, металургійний, гірничої електромеханіки), Криворізький металургійний факультет Національної металургійної академії України та Криворізький факультет Державного інституту підготовки та перепідготовки кадрів у промисловості (всього 16 навчальних закладів).

В дослідженні Н.Ю. Горбунової [58] визначено принципи побудови, структуру й основні напрями діяльності університетських комплексів як фактора розвитку регіональної системи безперервної технічної освіти.

Принципами побудови регіонального університетського комплексу є:

1) створення відкритого професійного середовища, в якому кожна людина може реалізувати свої здібності, потреби й можливості;

2) забезпечення багатоступеневої професійної освіти;

3) обґрунтований розподіл функцій між навчальними закладами, підприємствами й організаціями, що входять до університетського комплексу;

4) створення служб супроводу процесу безперервної професійної освіти (адаптаційних, діагностичних, науково-дослідних, психологічних, методичних центрів).

Як елементи до університетського комплексу входять: доуніверситетська підготовка, безперервна багаторівнева підготовка, філії, цільова підготовка фахівців, міжрегіональний інститут підготовки кадрів [58, 9].

Узагальнена схема комплексу показана на рис. 1.3.

Університетські навчальні комплекси надають можливість не тільки чітко й продуктивно координувати діяльність навчальних закладів, що входять до комплексу, незалежно від їхнього підпорядкування, а й сприяють досягненню нової якості освіти, забезпечуючи наближення загальноосвітніх і освітньо-професійних програм до потреб особистості, суспільства, регіону.

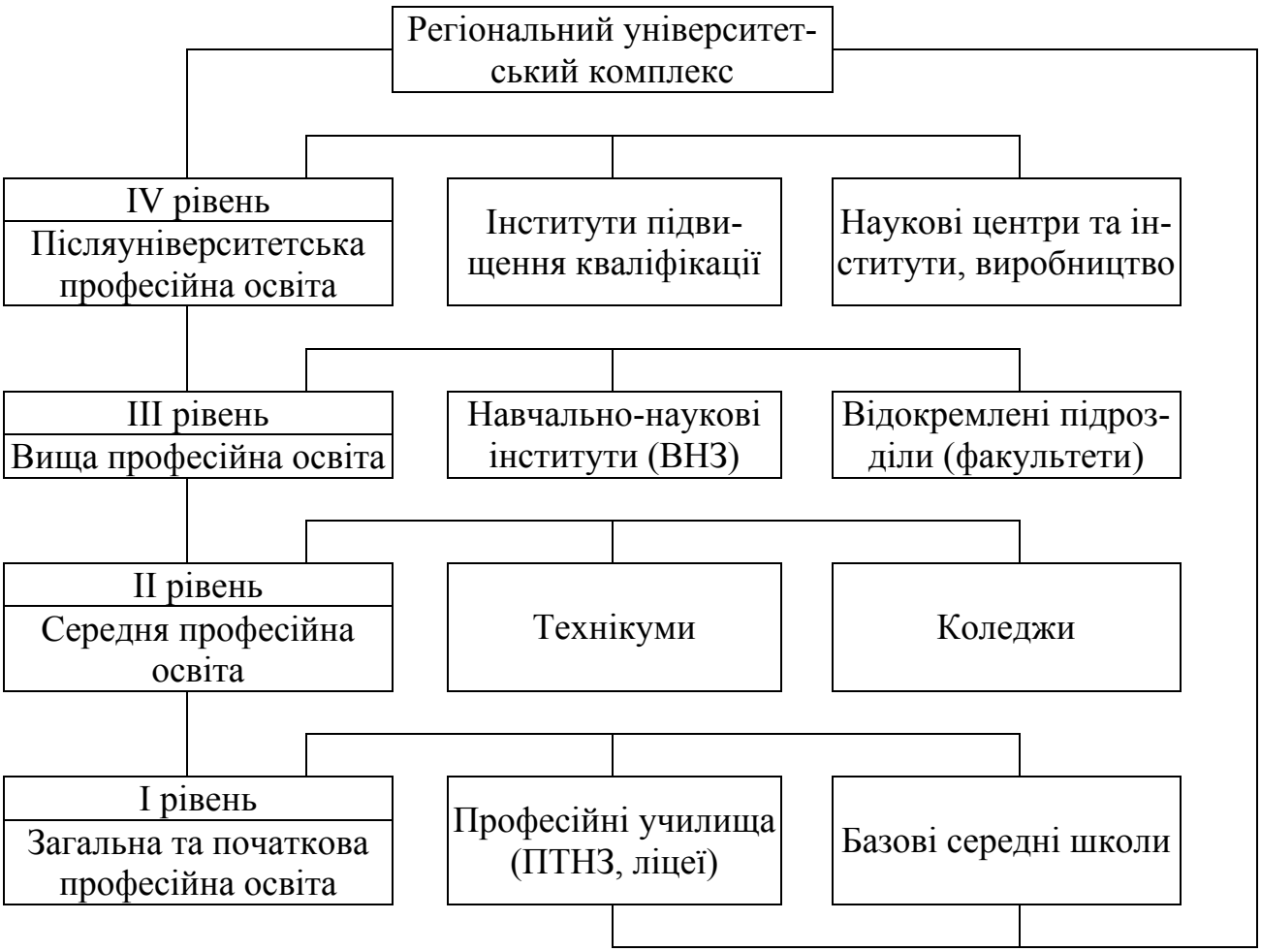


Рис. 1.3. Схема регіонального університетського комплексу

Спроби створення безперервної системи професійної освіти в Кривому Розі були неодноразові: так, у 1995 р. було створено комплекс у складі Криворізького державного педагогічного інституту, Криворізького обласного ліцею-інтернату для сільської молоді, Довгинцівської педагогічної гімназії №24, Жовтоводського та Нікопольського педагогічних училищ. Аналіз роботи комплексу показав, що, незважаючи на те, що на початковому етапі система неперервної освіти задовольняла потреби особистості й регіону, згодом від неї довелося відмовитися, тому що ця система була спрямована на підготовку «вузьких» фахівців (вчителів для сільських шкіл). Такий підхід не враховував розвиток, становлення й соціалізацію особистості в швидкозмінній кон'юнктурі ринку. Інакше кажучи, випускники цієї системи не могли мобільно реагувати на вимоги ринку й перекваліфікуватися відповідно до його запитів. Як показав аналіз навчальних планів та програм, у базових навчальних закладах простежувався

ухил у бік спеціалізації при відсутності широкої фундаментальної й загальнонаукової підготовки.

Наступною спробою створення безперервної системи професійної освіти стало введення багаторівневої системи – Криворізька вища металургійна школа, (1998 р.). Однак перехід до ринкових відносин вимагав прискореної соціальної й психологічної адаптації людини до нових умов, її професійної мобільності. Проблему ускладнювало те, що в середній школі не виявлялися схильності учнів до тих або інших видів діяльності. Як правило, випускники шкіл вступали до ВНЗ, не дуже добре орієнтуючись в обраній ними майбутній спеціальності. Система й методика навчання у ВНЗ істотно відрізнялася від звичної, шкільної системи. Крім того, великі навантаження ускладнювали адаптаційний процес. Найчастіше, під час навчання студенту приходило розуміння того, що спеціальність, на якій він навчається, «не його», і людині доводилося або припинити навчання, або починати навчання на іншій спеціальності «з нуля», тому що навчання було вузькоспеціальним (електромеханіка, коксохімічна технологія). Після закінчення вищого навчального закладу атестація випускника проводилася лише в рамках вузької професійної спеціалізації.

Таким чином, перераховані вище форми підготовки фахівців вищої кваліфікації не відповідали принципам неперервної освіти, а саме:

а) інтеграції освіти, науки й виробництва як єдиної комплексної системи здобування й використання нових наукових знань і технологій в освіті, економіці й соціальній сфері;

б) створення єдиного інформаційного середовища для освітньої, наукової та інноваційної діяльності;

в) створення єдиної системи підготовки, перепідготовки і підвищення кваліфікації фахівців для організацій різних форм власності й видів діяльності.

Для створення дієвої системи неперервної освіти в рамках університетського комплексу у 2001 р. свої зусилля об’єднали провідні фахівці Криворізького державного педагогічного університету та Криворізького металургійного факультету Національної металургійної академії України. Ґрунтуючись на кон-

цепції фундаменталізації навчання, були створені нові професійно-орієнтовані програми, що надавало можливість здобути фундаментальну політехнічну освіту за широким спектром спеціалізацій (механізація виробництва, автоматизований електропривод, економіка підприємства, металургійне виробництво, гнучкі комп'ютеризовані комплекси та робототехніка, коксохімічне виробництво та інші).

Включення всіх рівнів освіти до університетського комплексу підвищило рівень професійної орієнтації школярів; забезпечило навчання на основі погоджених навчальних планів і однакових вимог до оцінювання якості знань на всіх етапах навчання; підвищило рівень мотивації при вступі до ВНЗ; надало можливість готувати «свого» абітурієнта; забезпечило перехід учнів з одного рівня навчання на іншій у рамках єдиної системи; надало студентам можливість швидше адаптуватися до навчання у вищій школі; дозволило ліквідувати дублювання змісту освітніх програм; скоротило терміни навчання у ВНЗ випускників ліцеїв і коледжів; забезпечило ранню соціалізацію особистості.

Щорічним підсумком науково-педагогічного пошуку є міжнародні конференції «Теорія та методика навчання фундаментальних дисциплін у вищій школі» (Кривий Ріг, квітень-травень), «Стратегія якості в промисловості та освіті» (Варна, травень-червень), «Проблеми підготовки та перепідготовки фахівців у сфері інформаційних технологій» (Севастополь, вересень). Найдавніша з них – «Теорія та методика навчання фундаментальних дисциплін у вищій школі» – у 2008 році була проведена усьоме.

Саме орієнтація на фундаменталізацію навчання зумовила успішність роботи Криворізького навчально-наукового комплексу Національної металургійної академії України, метою якого є:

– підготовка фахівця широкого профілю й високої кваліфікації, здатного творчо вирішувати поставлені завдання, безупинно розвиватися, швидко адаптуватися до мінливих умов життя;

– організація підготовки фахівців як творчого процесу на основі принципу єдності навчання, науки та виробництва;

– виховання молодого фахівця як всебічно розвиненої, активної самодіяльної особистості, здатної оцінювати потреби й освоювати норми поведінки й систему цінностей соціуму;

– безперервність освіти та комплексний підхід до організації освітнього процесу.

В ході роботи комплексу було виявлено новий перспективний напрям дослідження – *фундаменталізація середньої професійної освіти*, об'єктивна потреба у розробці якого обумовлена динамічними змінами у техніці, пов'язаними зі збільшенням наукоємності виробничих процесів і систем управління, що приводить до подальшого ускладнення знарядь праці й професійної діяльності.

На думку Ю.В. Триуса, «для України найбільш раціональним є зосередження основних фінансових, матеріальних, людських ресурсів на розвитку високих технологій та конкурентоспроможного наукоємного виробництва» [405, 90], тому в цих умовах незмірно зростає роль фундаментальних знань і вмінь, що надають можливість фахівцеві швидко переучуватися і якісно освоювати нові виробничі й технологічні процеси. Отже, фундаменталізація професійної освіти стає неодмінною та провідною умовою підготовки майбутніх фахівців: саме на її основі найбільш ефективно можуть бути сформовані такі якості працівника сучасного виробництва, як широта професійного кругозору в поєднанні з його глибиною, професійна адаптація й мобільність, здатність до постійного саморозвитку й самоосвіти, здатність до гнучкого мислення й т.ін. [340]

Фундаменталізація змісту професійної освіти трактується як виділення інваріантних структурних одиниць змісту: основних компетентностей, оволодіння якими надає фахівцеві можливість бути професійно й академічно мобільним.

Підготовка фахівця технічного профілю в середньому спеціальному навчальному закладі (ПТНЗ, коледжі тощо) ґрунтується на циклах загальноосвітніх, загальнопрофесійних та спеціальних дисциплін. Загальнопрофесійні дисципліни, спираючись на загальнонаукові, є основою для прикладних і, у поєд-

нанні з останніми, є основою професійних компетентностей фахівця з вирішення виробничих завдань.

1.3. Фундаментальність інформатичної освіти

Підготовка вчителів інформатики та інженерів-програмістів за суттю є професійною освітою, проте в сучасних соціально-економічних умовах традиційне протиріччя між фундаментальним та професійним навчанням набуває нового змісту: якщо в минулому вузька профілізація була показником високої соціальної захищеності, то сьогодні таким показником стає мобільність, набути якої може лише широко освічена людина, здатна гнучко реагувати на зміну технологій. Вузькопрофесійна підготовка поступово вимивається із системи вищої освіти, переходячи у сферу професійно-технічної освіти та виробництва. Яскравим проявом вказаної тенденції є заходи Міністерства освіти та науки України, спрямовані на зближення вищої педагогічної та класичної університетської освіти.

1.3.1. Поняття фундаменталізації інформатичної освіти. **Інформатика – фундаментальна дисципліна**, *об'єктом* якої є інформаційні процеси в оточуючому світі, *предметом* – математичні структури, за допомогою яких моделюються інформаційні процеси, та комп'ютерні інформаційні моделі, що відображають математичні структури на архітектуру обчислювальних систем, *методологією* – обчислювальний експеримент. Віднесення інформатики до фундаментальних наук відображає загальнонауковий характер поняття «інформація» та процесів опрацювання повідомлень [166, 61].

О.Л. Семенов уточнює, що фундаментальна природничо-наукова частина інформатики будує теоретичні моделі процесів опрацювання, зберігання, передавання повідомлень: «За своїми об'єктами, поняттями, методами – це галузь математики. Предметом її вивчення є скінченні (конструктивні) об'єкти та алгоритмічно описані (конструктивні) процеси, що відбуваються в середовищі цих об'єктів» [295, 54]. Зазначену частину інформатики О.Л. Семенов називає *математичною інформатикою*. За М.І. Жалдаком, **математична інформатика**

є основою інформаційної технології, під якою він розуміє сукупність методів і прийомів збирання, зберігання, опрацювання, передавання та подання повідомлень, що розширює знання людей та розвиває їхні можливості з управління технічними та соціальними процесами [84; 95].

Як зазначає Р.Р. Фокін [413, 4–5], в розвитку інформатики як навчальної дисципліни існує ряд проблем та протиріч:

1. Виключно швидкий прогрес методології інформатики, її програмних та технічних засобів.

2. Навчальний матеріал швидко втрачає актуальність та постійно потребує заміни більш сучасним, причому застаріває не лише зміст, а й структура.

3. Існуюче методичне, програмне, технічне забезпечення інформатичних дисциплін швидко втрачає актуальність та застаріває.

Загальновизнано, що інформатика як наукова дисципліна розвивається надзвичайно швидко, що суттєво ускладнює розробку ефективних методичних систем її навчання. Практика навчання інформатики у вищій школі накопичила чимало методик та прийомів, що надають можливість досягти поставлених цілей навчання. Значна частина їх узагальнена у кількох десятках підручників та навчальних посібників з інформатики, випущених за останні 20 років. Зміст їх суттєво відрізняється та відображає як еволюцію методики навчання інформатики, так і об’єктивно існуюче різноманіття поглядів на сучасну інформатику. Впровадження нових державних стандартів дозволило дещо впорядкувати цей процес, проте не змінило його суті: *неусталеність методичних систем навчання інформатики була викликана її помилковим позиціонуванням як технологічної дисципліни*, вторинної в порівнянні з фундаментальними дисциплінами.

На думку О.О. Ракітіної, «основними факторами, що приводять до такого положення справ, є:

– високі темпи розвитку ... інформатики й технічних засобів інформатизації суспільства і, як наслідок, значний відрив змісту навчання інформатики від змісту інформатики як науки й сфери практичної діяльності;

– не усталена термінологія інформатики, наявність у її мові великої кількості іншомовних слів і словосполучень; відсутність стабільності у використанні термінології;

– нечіткість, розмитість вимог до навчальних і робочих програм курсу, що, по-перше, надає можливість педагогові самому вибирати найбільш важливі з його погляду теми й розділи, по-друге, призводить до значного розсіювання знань з предмету у випускників шкіл і вузів» [270, 20].

Ставлення до інформатики як до технологічної дисципліни породжує кричущі випадки, коли до навчальних планів спеціальностей «Прикладна математика», «Інформатика» тощо вводяться такі утилітарні скороминущі дисципліни, як «1С: Бухгалтерія та Підприємство», «Комп'ютерна графіка у Photoshop» і т.п. (нібито з метою залучення абітурієнтів чи підвищення практичної значущості навчання інформатики). Проте досвід зарубіжної вищої школи впевнено доводить, що *прагматизація є тупиковим напрямом в розвитку освіти* [289, 21], адже саме ґрунтовні теоретичні знання, широка загальна культура членів суспільства стимулюють соціальний, технічний, економічний прогрес. Необхідно чітко усвідомлювати, що **освіта тим краща з практичної точки зору, чим дальша вона від безпосередньої утилітарної корисності**. Тому відмова від принципу фундаментальності, який, як показано вище, визнається сьогодні у всьому світі головною умовою успішності функціонування вищої освіти, буде означати стрімкий рух нашої країни до освітнього колапсу, неминучого при ігноруванні тенденцій розвитку вищої освіти.

Досягнення професійної мобільності – одна з найважливіших задач Болонського процесу, розв'язання якої можливе лише за умови фундаментального характеру освіти. Так, за останні десятиліття, коли фундаментальні науки ставали рушійною силою виробництва, обсяг блоку математичних та природничонаукових дисциплін в технічних ВНЗ був скорочений у 1,5–2 рази. Тому, як зазначає О.В. Сергєєв, зниження рівня фундаменталізації технічної освіти може бути усунене лише на основі такої системи фундаментальної підготовки інженера, яка б охопила практично всі дисципліни навчального плану [326]. Перехід

на дворівневу академічну (замість вузькопрофесійної) освіту в рамках Болонського процесу, всупереч побоюванням, буде лише сприяти підвищенню рівня фундаменталізації підготовки спеціалістів [193].

Говорячи про фундаментальність інформатичної освіти [135], слід зазначити, що сьогодні в підготовці відповідних фахівців у США, країнах Західної Європи та Росії спостерігається зростання потреби в таких теоретичних знаннях, швидкість оновлення яких не настільки висока, як у прикладних, та які можна охарактеризувати в термінах доступності, збережуваності, універсальності та мінімізації вартості отримання знань. Все ці характеристики відносяться саме до фундаментальних знань.

Як свідчить досвід, фундаментальність у навчанні може бути досягнута, якщо в змісті навчання чітко виокремлені фундаментальні основи навчального предмета, що відповідають фундаментальним основам предметної галузі.

Н.Л. Стефанова уточнює, що фундаментальність полягає в тому, що «в змісті освіти розкривається не лише система певної галузі наукового знання, а й, можливо, поки що до кінця не сформована система знань про закономірності опанування та теоретичні основи побудови способів передавання багатовікового досвіду людства, об'єктивованого у сучасній системі знань» [355]. Тоді для забезпечення фундаментальності навчання проектування методичної системи навчання має спиратися на структуру поточного стану відповідної наукової дисципліни, що надає можливість врахувати сукупність зв'язків внутрішніх складових та визначає її зовнішні границі.

Як зазначає К.К. Колін, «саме це повинно надати людям можливість самостійно знаходити та приймати відповідальні рішення в умовах невизначеності, в критичних та стресових ситуаціях, а також у тих випадках, коли вони зіштовхуються з новими, вельми складними природними та соціальними явищами» [138].

Під терміном «*фундаменталізація інформатичної освіти*» будемо розуміти діяльність всіх суб'єктів освітнього процесу, спрямовану на підвищення якості фундаментальної підготовки студента, його системоутворюючих та інва-

ріантних знань і вмінь у галузі інформатики, що надають можливість сформувати якості мислення, необхідні для повноцінної діяльності в інформаційному суспільстві, для динамічної адаптації людини до цього суспільства, для формування внутрішньої потреби в безперервному саморозвитку та самоосвіти, за рахунок відповідних змін змісту навчальних дисциплін та методології реалізації навчального процесу.

Зміст інформатики сьогодні – це актуальна комплексна міждисциплінарна проблема, в розв'язанні якої однаково важливі як фундаментальні, так і прикладні дослідження. Тому для досягнення цілей фундаменталізації інформатичної освіти необхідно змістити увагу викладачів та студентів з проблеми набуття прагматичних знань на проблеми розвитку інформаційної культури та формування системного мислення на основі розуміння сутності інформаційних процесів [269].

1.3.2. Напрями фундаменталізації інформатичної освіти. Фундаменталізація, що передбачає поглиблення теоретичної, загальноосвітньої та загальнонаукової підготовки, є тенденцією, характерною для вищої професійної освіти Росії в цілому.

Г.О. Широких [431] виділяє два аспекти розв'язування проблеми фундаменталізації предметної підготовки вчителя інформатики: внутрішньопредметний та міжпредметний. З одного боку, пошук шляхів вирішення проблеми спрямований у внутрішньопредметну галузь інформатики з перенесенням акцентів на застосування формальних методів і відповідного математичного апарата (внутрішньопредметний аспект). З іншого боку, у дослідженнях В.П. Беспалька, В.І. Кагана, В.О. Сластьоніна та ін. підкреслюється, що подальша фундаменталізація підготовки фахівців повинна бути спрямована на педагогічну інтеграцію, подолання розриву між знаннями, отриманими студентами при вивченні різних навчальних дисциплін за рахунок істотного розвитку міжпредметних зв'язків (міжпредметний аспект).

Т.В. Мінькович, аналізуючи тенденції фундаменталізації шкільного курсу інформатики у Росії, пропонує шляхи *інтеграції теоретичної інформатики та*

інформаційних технологій засобами комп'ютерного моделювання [192]. Автор використовує апарат теорії управління для поєднання традиційної моделі комп'ютерної системи як сукупності апаратного та системного програмного забезпечення з моделлю інформаційних процесів через розв'язування таких задач інформатики, як подання повідомлень та описів інформаційних процесів, вивчення та організації кібернетичних систем, інформаційне моделювання реального світу [20].

Представники пермської школи дидактики інформатики М.О. Плаксін [237], Є.К. Хеннер [168], І.Г. Семакін [292], С.В. Русаков [230], Л.В. Шестакова [285] та інші серед шляхів фундаменталізації шкільного курсу інформатики виділяють:

– вивчення позакомп'ютерних аспектів інформатики, методів раціонального опрацювання повідомлень і даних без використання комп'ютера;

– побудову курсу інформатики на основі модельного підходу;

– вивчення основ системного аналізу, методів прийняття рішень в умовах невизначеності.

У звіті об'єднаної комісії ACM та IEEE Computer Science 2001 року [281, 62] зазначається, що однією з основних характеристик інформатики протягом всієї її відносно невеликої історії є дуже швидкий темп змін. Тому випускники повинні володіти глибокими фундаментальними знаннями, що допомагають їм виробляти нові необхідні навички в міру того, як еволюціонує область. Для досягнення цієї мети пропонується:

– застосовувати методики навчання, в яких підкреслюється різниця між навчальною діяльністю викладача та навчальною діяльністю студента, стимулюється незалежне мислення студентів;

– навчати студентів на творчих задачах та вправах, розв'язування яких розвиває їхню ініціативність;

– застосовувати методично узгоджені теоретичні та практичні курси, що забезпечують стабільність закріплення матеріалу;

– ознайомлювати студентів з інформаційними ресурсами та стратегіями оновлення своїх знань;

– постійно оновлювати обладнання та програмне забезпечення;

– заохочувати колективне навчання та застосування телекомунікаційних технологій для забезпечення взаємодії груп студентів;

– переконувати студентів у необхідності продовження професійного розвитку та самовдосконалення протягом усього життя [281, 64–65].

В якості *фундаментальних концепцій інформатики* цей документ визначає спільні ідеї, інваріантні по відношенню до виробників ПЗ, конкретних програмних пакетів та вузькоспеціалізованих вмінь, наводячи в якості прикладів теорію алгоритмів, архітектуру ЕОМ, способи подання інформації, моделювання та ін. «*Розуміння фундаментальних концепцій є виключно важливим для ефективної роботи з комп'ютером. Важливість специфічних навичок скороминуща, в той час, як знання фундаментальних концепцій будуть допомагати студентам протягом багатьох років, що особливо важливо з урахуванням сучасних темпів зміни інформаційних технологій*» [281, 68].

О.Є. Пупцев, визначаючи напрями формування змісту курсу інформатики в дванадцятирічній реформованій загальноосвітній середній школі Білорусі, носіями фундаментальних ідей визначає комп'ютерне моделювання, теорію алгоритмів, методологію і теорію програмування та ін. [268].

О.Г. Смолянінова виділяє наступний блок фундаментальних інформатичних дисциплін: «Теоретичні основи інформатики», «Програмування», «Дослідження операцій», «Інформаційні системи», «Теорія алгоритмів», «Основи мікроелектроніки та архітектура комп'ютерів» [336].

Н.В. Морзе до змісту фундаментальної підготовки вчителя інформатики відносить такі розділи: теоретичні основи інформатики, теорія алгоритмів, структури даних, технологія розробки програмного забезпечення, архітектура комп'ютерних систем, парадигми програмування (функціональне, продукційне, хорновське, об'єктно-орієнтоване), комп'ютерна графіка, операційні системи, інформаційні системи, теоретичні основи баз даних, бази даних і інформацій-

ний пошук, системи штучного інтелекту, комп'ютерне моделювання, аналіз і моделювання систем, дискретна математика, теоретичне програмування, соціальна інформатика, комп'ютерні комунікації і мережі, глобальна мережа Інтернет, гіпермедійний дизайн, програмна інженерія [211, 17].

М.П. Лапчик, досліджуючи структуру та методичну систему підготовки вчителів інформатики, вказує, що важливе місце в ній займає математична компонента фундаментальної освіти, призначення якої: отримання освіти в галузі основ математики, математичного моделювання, відсутність якого робить неможливим застосування інформатики для розв'язування прикладних задач; формування фундаментальних основ теоретичної (математичної) інформатики, що складають загальноосвітнє ядро цієї галузі знань [169].

Автори «Computing Curricula 2001: Computer Science», аналізуючи проблеми, що виникають при створенні основних курсів (п. 8.1 [281]), окремо виділяють дисципліни «Операційні системи» та «Системне програмування» (розділ «Побудова компіляторів») як «артефактні динозаври програмування» (за висловом М. Шоу [554]) через *високий ризик прив'язування змісту дисциплін до конкретних програм, виробників чи реалізацій, що робить знання студентів уразливими та швидко застаріваючими*. Саме тому в третьому розділі даного дослідження особлива увага звертається на фундаменталізацію цих дисциплін для всіх напрямів навчання, а в додатку Б наведені методичні основи вивчення теми «Операційні системи» у підготовці майбутнього вчителя.

М.В. Швецьким [165] сформульована концепція фундаменталізації інформатичної освіти, заснована на використанні в змісті навчання теорії, абстракції й реалізації. При цьому за допомогою вивчення відповідних математичних теорій, алгоритмів і структур даних конкретною мовою програмування передбачається домогтися формування фундаментальних знань з предмету.

Інша концепція фундаменталізації інформатичної освіти, сформульована Н.І. Рижовою [287], полягає у виділенні в змісті навчання світоглядних, філософських і математичних (та/або семіотичних) підстав навчального предмета й навчанні побудови формальної мови предметної галузі й формалізації теорій

предметної галузі за допомогою формальних мов із властивостями конструктивності. Ця концепція стала наслідком визначення інформатики як науки про семіотику формальних мов із властивостями конструктивності, призначених для опису інформаційних процесів за допомогою «формального використання» комп'ютера [288, 6].

Дослідник робить висновок, що фундаменталізація інформатичної освіти забезпечується включенням до змісту освіти:

– математичних основ інформатики, складовою яких є певна система формальних мов;

– питань формалізації сімейства як напівформальних, так і змістових мов, що використовуються в інформатиці.

Таким чином, фундаменталізація інформатичної освіти зводиться до посилення математичної складової. Безумовно, взаємозв'язок математики та інформатики дуже тісний: якщо на попередніх етапах розвитку інформатика розглядалась як елемент прикладної математики, то сьогодні, з появою поняття «комп'ютерна математика», на черзі дослідження й зворотного процесу – «як інформатика впливає на математику» [224, 30]. Яскраві приклади реалізації зворотного процесу – роботи М.І. Жалдака та Г.О. Михаліна з «комп'ютерної стохастики» [85], М.І. Жалдака та Ю.В. Триуса з «комп'ютерних методів оптимізації» [94]. У 2008 р. вийшли ще два посібники за редакцією М.І. Жалдака: «Системи комп'ютерної математики: Maple, Mathematica, Maxima» Т.П. Кобильника [134] та «Основи роботи в SAGE» С.В. Шокалюк [440].

В основі концепції, сформульованої С.Д. Каракозовим [116, 63], лежить трактування фундаменталізації інформатичної освіти як виділення в змісті навчання основ навчального предмета як сукупності базових прикладних завдань і навчання діяльності з їх розв'язування за допомогою обчислювальних систем (тобто навчання обчислювального експерименту). Зокрема, для вчителя інформатики система прикладних завдань повинна вибиратися на основі предметної галузі «Освіта».

Інший шлях розвитку фундаменталізації інформатичної освіти полягає в пошуку фундаментальних основ базової науки, що групуються навколо її центральної категорії – інформації. Це, у свою чергу, приводить до деяких змін профілю курсу інформатики. Можливість такої зміни цілком природна й полягає в самому змісті інформатики, що має інтегративний характер.

Даної концепції дотримуються такі вчені, як М.І. Жалдак, Н.В. Морзе, В.С. Ледньов, О.А. Кузнєцов, О.О. Ракітіна, Т.Б. Захарова та ін. Дослідники вважають, що фундаментальні основи інформатики обов'язково повинні включати уявлення про сутність понять інформація і повідомлення, закономірності протікання інформаційних процесів, про інформаційні моделі, інформаційні основи управління.

С.О. Бешенков указує, що найважливішою особливістю фундаменталізації інформатичної освіти є введення понятійного апарата, за допомогою якого можна було б розкрити зміст фундаментальної категорії інформації. Центральні поняття – поняття формалізації та інформаційної технології розв'язування задач. За допомогою цього апарата можна з максимальною повнотою розкрити зміст інформаційної діяльності не тільки в природничо-науковій, а й в гуманітарній сфері [22].

Таким чином, можливі два основні напрями фундаменталізації курсів інформатики:

1) математизація змісту навчання й розвиток формального компонента діяльності (центральними поняттями інформатики стають алгоритм і комп'ютер);

2) побудова курсів інформатики від феномена інформації та інформаційних процесів до методів їх вивчення за допомогою інформаційних моделей шляхом використання комп'ютера як засобу управління інформаційними процесами [197].

Ці два підходи цілком об'єктивні й відображають процеси, що відбуваються в усьому світі, але далеко не рівноправні з погляду знань, що формуються. Разом з тим, найбільш перспективним є курс, що об'єднує ці підходи на основі широкого застосування комп'ютерного моделювання. Однак переважна

більшість навчальних програм у ВНЗ орієнтовані на вивчення або апаратно-програмних засобів персонального комп'ютера та інформаційних технологій, або алгоритмізації та програмування, хоча задекларовані цілі курсу інформатики звичайно бувають набагато ширшими.

В роботі [258] визначені основні принципи застосування моделювання в якості засобу фундаменталізації інформатичної освіти. Так, в педагогічних ВНЗ моделювання традиційно вивчається на п'ятому курсі як закономірний підсумок навчання інформатики. Перехід до двоступеневої освіти бакалавр – магістр призводить до того, що колишній п'ятий курс за планом підготовки спеціаліста подвійних спеціальностей «Математика та основи інформатики», «Фізика та основи інформатики» стає першим курсом магістерської підготовки відповідних моноспеціальностей, в яких моделювання як навчальна дисципліна може просто зникнути. В підготовці ж бакалавра технологія моделювання розглядається переважно в курсі чисельних методів.

Такий підхід може призвести до формування у студентів хибного уявлення про те, що цілісний за своєю суттю курс інформатики є набором окремих, слабко пов'язаних між собою дисциплін. Враховуючи, що саме моделювання є тим «клеєм», який пов'язує усі інформатичні дисципліни, з метою їх інтеграції І.О. Теплицьким була розроблена система наскрізного навчання моделювання на всіх спеціальностях педагогічних ВНЗ, що мають спеціалізацію «Інформатика» [387].

Згідно з розробленою системою, початкове ознайомлення з моделюванням відбувається на І курсі при вивченні дисципліни «Шкільний курс інформатики» в розділі «Електронні таблиці». В якості практичних задач тут пропонуються задачі на побудову та аналіз *математичних моделей* з навчального посібника [373], особливістю яких є те, що при їх розгляді не вимагаються знання, що виходять за межі шкільного курсу інформатики. При розгляді розділу «Бази даних та інформаційні системи» відбувається ознайомлення з *інформаційним моделюванням*, яке є основою наступного курсу – «Інформаційні системи та технології».

При переході до навчання алгоритмізації та процедурного програмуванню розглядаються різні способи структурування даних як моделей об'єктів оточуючої дійсності. Зокрема, вивчення алгоритмів стискання даних у розділі «Теорія кодування» надає можливість ознайомитись з *моделюванням даних*, що не завжди можна зробити в узагальнюючому курсі моделювання. Процес алгоритмізації розглядається через побудову *алгоритмічних моделей*, природним продовженням яких є *програмні моделі*. У такий спосіб студенти приходять до необхідності природного поєднання даних та методів їх опрацювання, що надає можливість у природний спосіб перейти до об'єктно-орієнтованого програмування.

Розгляд різних технологій програмування вимагає побудови *процедурних*, *об'єктно-орієнтованих*, *функціональних* та *візуальних моделей* перебігу програм.

Вивчення об'єктно-орієнтованого програмування починається з огляду його основних складових – об'єктно-орієнтованого аналізу та проектування, які, у відповідності до [432], є основами *об'єктно-орієнтованого моделювання*. Для студентів, які навчаються за спеціальністю «Математика та основи інформатики», пропонуємо побудову моделей числових об'єктів різної природи як інтерпретації відповідних алгебраїчних структур – класів натуральних чисел (як символьних послідовностей, що задовольняють аксіомам Пеано), цілих, раціональних та комплексних чисел, реалізуючи при цьому класичні алгоритми «довгої арифметики». Вивчення класів поповнюваних одно- та двовимірних масивів конкретизується задачами на побудову векторних, поліноміальних та матричних об'єктів за [242]. Неминуча при цьому програмна реалізація методів лінійної алгебри створює передумови до вивчення чисельних методів у об'єктній методології. Такий підхід в курсі чисельних методів надає можливість із самого початку працювати з векторно-поліноміальними та матричними моделями, суттєво підвищуючи рівень наочності програм. Вивчені в курсі методи чисельного інтегрування диференціальних рівнянь застосовуються далі в курсі комп'ютерної графіки.

Застосування в курсі об'єктно-орієнтованого програмування сучасних систем проектування програмного забезпечення, побудованих на основі компонентних технологій, надає можливості для розгляду *візуальних моделей*. Інший спосіб введення такого типу моделей пропонується при розгляді систем комп'ютерної математики, в яких можлива повна реалізація процесу моделювання (від постановки задачі до аналізу результатів обчислювального експерименту).

В роботі [176] представлено структуру курсу об'єктно-орієнтованого програмування для майбутніх вчителів фізики та інформатики, в якому опанування концепціями об'єктного підходу відбувається через побудову різноманітних моделей фізичних явищ. Запропонована структура курсу передбачає початкове ознайомлення з *геометричним моделюванням* та його реалізацією в бібліотеці тривимірної графіки OpenGL, яка пропонується студентам у об'єктній реалізації.

Використання бібліотеки OpenGL (Mesa) при цьому надає можливості зручного моделювання 3D-об'єктів, а чисельні методи стають основою для побудови ілюстративних моделей фізичних процесів. Так, для реалізації пропонуються задачі кінематики, молекулярної динаміки та інші.

Вивчення основ автоматичного управління в курсі автоматики також відбувається на основі побудованих моделей математичних об'єктів [240]. Застосовувані при цьому алгебраїчні методи операційного числення надають можливість змоделювати процес управління різними технічними системами на основі активного використання *імітаційних моделей*.

Такий кібернетичний підхід розвивається далі в курсі «Системи управління базами даних», де основну увагу приділено реалізації розглянутих в шкільному курсі інформатики *інформаційних моделей*, зокрема – моделей реляційної алгебри [399].

Наступним етапом систематичного навчання моделювання є вивчення *моделей подання знань* в процесі побудови інтелектуальних систем у відповід-

ному курсі. Для цього пропонуються задачі на побудову експертних систем, систем символьної математики, нейронних мереж та систем логічного виведення.

За такого підходу інтегрований курс моделювання, що пропонується наприкінці навчання, може містити лише ті розділи, що не були розглянуті у попередніх дисциплінах: моделювання фрактальних об'єктів, моделі економічної динаміки тощо [341].

Досягнення поставленої цілі фундаменталізації інформатичної освіти можливе через організовану цілеспрямовану педагогічну діяльність, що забезпечує реалізацію виділених у п. 1.1 *функцій фундаменталізації освіти*:

– опанування методологічно важливими та інваріантними знаннями, що мають довгий термін життя, необхідними для професійної діяльності фахівця в галузі інформаційних технологій (*методологічна функція*);

– тісний зв'язок інформатичної освіти з професійною практичною діяльністю (*професійно-орієнтувальна функція*);

– розвиток творчої і пізнавальної активності та самостійності (*розвивальна функція*);

– розвиток методичних систем навчання інформатичних дисциплін з врахуванням перспектив розвитку «економіки знань» та інформаційного суспільства (*прогностична функція*);

– системність засвоєння інформатичних дисциплін на основі глибокого розуміння сучасних стану та існуючих проблем інформатики (*інтегративна функція*).

1.3.3. Професійна спрямованість фундаменталізації інформатичної освіти. За В.Д. Шадриковим, професійна спрямованість формується у мотиваційній сфері та являє собою «систему мотивів, що спонукають професіонала до виконання професійних завдань та задач професійного розвитку. В якості мотивів виступають потреби, інтереси, установки, переконання, ідеали та інші психологічні утворення людини. Головна їх особливість полягає в тому, що вони ... реалізуються в процесі виконання професійної діяльності чи розв'язування задач професійного розвитку» [267, 177].

В.О. Сластьонін вважає, що професійна спрямованість особистості, «представляючи собою вибіркове ставлення до дійсності та ієрархічну систему мотивів, ... пробуджує та мобілізує приховані сили людини, сприяє формуванню у неї відповідних здібностей, професійно важливих особливостей мислення, волі, емоцій, характеру» [332, 8].

В роботах Н.В. Кузьміної професійна спрямованість трактується як інтерес до професії та схильність нею займатися. Дослідник визначає професійну спрямованість як складне багатовимірне утворення, що має властивості об'єктності, специфічності, узагальненості, валентності, задоволеності, опірності, стійкості, центральності та ін. [158].

В.О. Якунін [446] підкреслює, що у міру навчання та опанування професійною діяльністю уявлення про різні її сторони змінюються. Формуванню професійної спрямованості особистості буде сприяти професійно-орієнтоване навчання, характерною рисою якого є суттєвий вплив на формування мотивації навчальної діяльності та розвиток інтересу до майбутньої професії. В навчальному процесі вищої школи професійна спрямованість навчання виступає як основний принцип, що надає можливість розв'язати протиріччя між теоретичним характером дисциплін, що вивчаються, та необхідністю практичного застосування знань в професійній діяльності.

Ідея послідовного систематичного наближення студента до майбутньої професійної діяльності із самого початку навчання у вищому навчальному закладі реалізована в роботах А.О. Вербицького. У [41] ним вводиться поняття контекстного навчання, яке характеризується моделюванням за допомого знакових засобів мовою начальних дисциплін предметного та соціального змісту майбутньої професійної діяльності. Одиницею роботи викладача та студента є не порція матеріалу, а ситуація, що несе в собі можливості розгортання змісту навчання в його динаміці.

Інтегративна функція фундаменталізації інформатичної освіти, що спонукує реалізацію принципу системності, полягає в такому об'єднанні фундаментальної та варіативної частин змісту інформатичних дисциплін, на основі якого

формується нова якість майбутнього спеціаліста – професіоналізм, що стає основою професійно-орієнтовної функції фундаменталізації інформатичної освіти. Також системність є необхідною умовою реалізації принципу науковості, на основі чого реалізуються методологічні функції фундаменталізації інформатичної освіти та створюються умови для реалізації її розвивальної функції.

Професійно-орієнтувальна функція фундаменталізації інформатичної освіти, введена під впливом ідей Г.О. Михаліна [195] та Н.В. Морзе [210], має наступні структурні компоненти: цільовий, змістовий, технологічний та підсумковий.

Враховуючи, що головною метою інформатичної підготовки студентів є формування професійних інформатичних компетентностей, основою *цільової компоненти* обрано суспільне замовлення, державні стандарти вищої освіти та особистий вибір студента. *Змістова компонента* містить специфічну інформатичну теорію, що відображає професіоналізацію за обраною спеціальністю. *Технологічною компонентою* визначається добір засобів, форм та методів реалізації задачі фундаменталізації інформатичної освіти. *Підсумкова компонента* для методичної системи навчання є діагностичною, в ній відображається рівень сформованості професійних інформатичних компетентностей студентів.

Зміст навчання є тим стрижнем, навколо якого поєднуються всі складові системи освіти, визначається їх послідовність та наступність. При формуванні змісту важливо встановити баланс між фундаментальністю та професійною спрямованістю інформатичної підготовки, реалізувавши виділений Г.О. Михаліним принцип диференційованої фундаментальності [194].

В.Д. Шадриковим у [267] в якості структуроутворюючого фактору проектування дидактичних систем математичної освіти студентів педагогічних ВНЗ запропоновано концепцію *фундирування*, спрямовану на подолання недоліків освітньо-професійних програм підготовки вчителів середньої школи в Росії. Виділимо серед них ті, що відносяться до інформатичної освіти Росії, Білорусі та України:

1. Недостатньо обґрунтовано визначається базове універсальне ядро змісту освіти, інваріантне для різних виконавців (педагогічних та класичних університетів).

2. Зміст професійно-предметної підготовки слабко узгоджується з державними стандартами загальної освіти, розв'язанням професійних завдань, розвитком здібностей і особистісних якостей майбутнього вчителя, іноді порушується розумний паритет складності змісту вимог і труднощів опанування студентами освітніх програм.

3. Співвідношення лекційних і практичних занять у реалізації навчальних програм як за обсягом, так і за часовими витратами не завжди відповідають реаліям сучасного інформаційного суспільства й потребам самореалізації студентів.

4. Зміст елементарного предмета (шкільний зміст предмета) не завжди забезпечує стійкість і варіативність освоєння навчальних елементів, слабко корелює з фундаментальними курсами, із сучасними досягненнями науки й техніки.

5. Не завжди виявляється цілісність і єдність предметного знання, генезис базових навчальних елементів і універсальних навчальних дій, історія предмета й предметної освіти.

6. Не виявляються цілісні механізми інтеграції фундаментальних і методичних знань і вмінь, їх спрямованість на професійну діяльність у світлі компетентнісного підходу.

Як зазначає Г.О. Михалін [194], педагогічний процес підготовки потрібно розглядати як формування цілісної системи професійної діяльності майбутнього педагога в напрямі відповідності професійному стандарту педагогічної діяльності. Для інформатичної освіти цей процес може бути розділений на три етапи:

I – *етап професіоналізації*. На цьому етапі формуються базові предметні знання й уміння, призначені для набуття базових інформатичних компетентностей (при підготовці інженерів-програмістів) та узагальнення базових навчальних елементів шкільного предмета (при підготовці вчителів інформатики).

II – *етап фундаменталізації*. На цьому етапі здійснюється глибоке теоретичне узагальнення знань та вмінь, набутих на попередньому етапі.

III – *етап технологізації*. На цьому етапі відбувається включення професіоналізованого та фундаменталізованого знання в структуру професійної діяльності як засіб самореалізації фахівця в галузі інформаційних технологій.

Автори [267] визначають *фундирування* як «процес становлення особистості педагога в опорі на поетапне розширення й поглиблення якостей особистості школяра, необхідне й достатнє для теоретичного узагальнення шкільної освіти, у напряму розвитку мислення, особистісних і професійних якостей майбутнього педагога».

За своєю основою, принцип фундирування є діалектичним: В.Д. Шадриков пропонує спіралеподібну схему моделювання базових знань, умінь, навичок предметної підготовки студентів. Починаючи зі шкільного предмета через пошарове фундирування його в різних теоретичних дисциплінах, обсяг, зміст і структура предметної підготовки повинні суттєво змінитися в напрямі практичної реалізації теоретичного узагальнення шкільного знання за принципом «бумерангу». Шкільні знання стануть виступати структуроутворюючим фактором, що надає можливість добирати теоретичні знання із предметної галузі більш високого рівня, через які відбувається фундирування шкільного знання. Інший шар фундирування може утворити вдосконалення й поглиблення практичних умінь, постановка експерименту, студентські дослідження, проектування орієнтувальної основи навчальної діяльності.

Таким чином, фундирування здійснюється на основі створення механізмів і умов (психологічних, педагогічних, організаційно-методичних, матеріально-технічних) для актуалізації й інтеграції базових загальноосвітніх навчальних предметів й вузівських знань з наступним теоретичним узагальненням і розширенням практичного досвіду майбутнього фахівця.

Концепція фундирування припускає розгортання в процесі предметної підготовки студентів наступних двох компонентів:

– визначення, аналіз і механізми реалізації змісту рівнів базових навчальних елементів і видів діяльності (знання, уміння, навички, математичні методи, ідеї, алгоритми й процедури, змістові лінії, характеристики особистісного досвіду);

– визначення й реалізація технології фундирування з урахуванням проектування індивідуальних освітніх траєкторій і розвитку самостійності студентів як основи конкурентоздатності на ринку праці (механізми управління пізнавальною й творчою діяльністю студентів, блоки формування професійної мотивації в освоєнні базових навчальних елементів і видів діяльності, варіативність способів розв'язання навчальних завдань).

Ключовим у концепції фундаменталізації є принцип *наскрізної інтеграції* навчальних дисциплін (навчального матеріалу) на основі формування інформатичних компетентностей (універсальних і професійних). Показником інтегративності навчальних дисциплін служить наступність у розгортанні змісту й структури навчальних дисциплін на основі фундаментальних концепцій інформатики.

На думку С.А. Ракова, цінність *фундаменталізації змісту* навчальної дисципліни полягає в переході від навчального елемента (універсальної навчальної дії) на рівні «даних» до його глибокого теоретичного узагальнення на рівні «сутності» для навчального процесу у ВНЗ та в майбутній професійній діяльності. Саме тому фундаменталізація змісту навчальної дисципліни надає можливість визначити стійке (інваріантне) ядро її змісту, а фундаментальність може бути досягнута, якщо в змісті навчання чітко виокремлені фундаментальні основи навчального предмета, що відповідають фундаментальним основам предметної галузі.

О.Х. Шень вказує на те, що «слід вчити фундаментальних сутностей, а не другорядних деталей, без яких можна обійтися. ... Сьогоднішні школярі – це навіть не завтрашні, а лише післязавтрашні програмісти. (Сьогодні їх найчастіше вчать вчорашнього (позавчорашнього?) програмування)» [137, 59].

Стабілізація ядра навчальних курсів на основі відокремлення їх фундаментальної складової від технологічної є одним з найбільш перспективних напрямів фундаменталізації інформатичних дисциплін. Так, зокрема:

1. Сучасні операційні системи базуються на принципах, закладених понад 20 років тому, і за своєю архітектурою є практично нерозрізненні. Визначальною є їх сумісність зі стандартом відкритих систем POSIX, за яким надається можливість використовувати в різних операційних системах єдиний програмний інтерфейс. Це дає можливість використовувати однакове (на рівні вихідних текстів) програмне забезпечення для різних операційних систем [299; 303; 305; 319; 374].

2. Інваріантність до мови програмування забезпечується створенням єдиного набору предметно-орієнтованих бібліотек. Так, в курсі чисельних методів математики може бути використана незалежна від ОС та середовища програмування бібліотека векторно-матричних об’єктів, реалізована мовами C++, Pascal, Python та Java [241; 242; 369]. Ще один варіант побудови цього курсу передбачає застосування системи комп’ютерної математики Maxima, що також функціонує на різному апаратному забезпеченні та під управлінням різних ОС [174; 298; 325; 318; 437]. Курс об’єктно-орієнтованого програмування має ядро, для підтримки якого застосовуються мови C++, Smalltalk, Java. Інший приклад – курс системного програмування, що охоплює спільні для POSIX-систем засоби та може викладатися із застосуванням мов C, Pascal та Python [246; 256].

3. Для підтримки таких курсів, як «Інтелектуальні та експертні системи» ([82; 297; 309; 324; 366]), «Паралельні та розподілені обчислення» ([181; 426]), «Моделювання» ([177; 311; 341; 380]) тощо доцільним є вибір стабільного програмного забезпечення, яке добре випробуване та оцінене в навчальній та науковій діяльності. Таке програмне забезпечення має тривалу історію, значний досвід успішного використання (не менше 15 років) але, в силу свого некомерційного характеру, як правило, є нелокалізованим.

На основі усталення змісту та засобів навчання інформатики через інваріантність відносно операційної системи та мови програмування з'являються широкі можливості:

– підвищення рівня теоретичної підготовки та формування компетентностей, необхідних для опанування сучасних інформаційних технологій;

– реалізації взаємозв'язків різних підходів (системного, діяльнісного та ін.), міжпредметної інтеграції та застосування методів суміжних наук (математики, фізики, філософії, природознавства);

– добору апаратних та програмних засобів навчання інформатичних дисциплін, зниження вартості цих засобів за рахунок використання ліцензійно чистого, вільно поширюваного, локалізованого програмного забезпечення;

– створення стабільних підручників.

Таким чином, **стабілізація курсів інформатики досягається поширенням на методичну систему навчання інформатики властивостей *відкритих систем*: розширюваності, масштабованості, мобільності, інтероперабельності та «люб'язності».**

Фундаменталізація змісту інформатичних дисциплін характеризується наступним компонентним складом:

1. *Освоєння сучасних галузей науки на основі виявлення генезису базових навчальних елементів і способів діяльності:*

– подання, опанування й генезис наукового знання й прийомів наукової діяльності: нанотехнології, фрактальна геометрія, вейвлети, моделі хаосу та ін. (за К.В. Корсаком [149]);

– розкриття історико-генетичних підстав значимості базових навчальних елементів розділу науки в інтегративному зв'язку з методикою навчання предмета (за В.Г. Бевз [18]);

– реалізація дослідницького підходу, у тому числі проектного методу на широкому спектрі сучасних досягнень науки й можливостей застосування в професійній діяльності (за Н.В. Морзе [209]);

– формування елементів наукового мислення й методологічної культури опанування елементів наукового пізнання в розв'язуванні навчальних і професійних завдань (за В.І. Клочком [130]).

2. *Наступність змістових ліній інформатичних дисциплін і варіативність способів розв'язування навчальних та практичних завдань на рівні міждисциплінарних взаємозв'язків:*

– визначення змістових складових (фундаментальної і технологічної) у навчальному предметі (знання, уміння, навички, ідеї, алгоритми й процедури);

– актуалізація передового досвіду предметної діяльності в інтерактивному режимі з використанням мобільних технологій, адекватних цілям навчання;

– інтеграція змісту, прийомів і методів опанування навчального матеріалу, міждисциплінарних взаємозв'язків на рівні системної інтеграції й змістовно-процесуального симбіозу.

3. *Створення умов (психологічних, педагогічних, організаційно-методичних, матеріально-технічних) для розвитку креативності, пошукової й творчої активності студентів у розв'язуванні навчальних і професійно-орієнтованих завдань:*

– освоєння фундаментального й технологічного знання з використанням інформаційно-комунікаційних технологій, зокрема технологій електронного, дистанційного та мобільного навчання;

– створення нових навчально-лабораторних комплексів, спеціальних курсів, навчальних дисциплін і методичних матеріалів, форм організації навчальної й наукової діяльності студентів на стику державних стандартів, інтересів регіону та корпоративного сектора;

– творче освоєння практико-орієнтованого поля майбутньої професійної діяльності.

У фундаменталізації змісту навчального предмета в контексті професійно-орієнтувальної функції фундаменталізації інформатичної освіти простежуються три лінії:

– визначення змісту навчального предмета, виходячи з його особливостей: добір, структура, етапи вивчення, інтегративні знання, співвідношення фундаментальної та технологічної складових тощо;

– наступності та теоретичного узагальнення базових навчальних елементів: посилення прикладного й діяльнісного компонентів навчання предмету, модульний принцип розгортання змісту навчального предмету й т.п.;

– врахування психологічних і педагогічних особливостей сприйняття, засвоєння, подання, застосування, аналізу й синтезу навчального матеріалу суб'єктом навчання: наочне моделювання, імітаційне моделювання, структурний аналіз базових навчальних елементів, посилення евристичного й гуманітарного компонентів, розвиток інтелектуальних і особистісних характеристик, варіативність розв'язування навчальних завдань і т.п. [35].

Ефективність опанування інформатичних дисциплін на основі концепції фундаменталізації змісту може бути визначена шляхом вимірювання (оцінювання):

а) рівня засвоєння базового знання (*професійно-предметний рівень*);

б) рівня засвоєння фундаментального знання (*фундаментальний рівень*);

в) рівня розвитку загальнонавчальних і професійних умінь, творчої активності студентів (*загальнопрофесійний рівень*);

г) рівня розвитку особистісних якостей та інтересів студентів (інтелектуальних, мотиваційних, оцінка рис особистості) (*рівень самореалізації*);

д) *рівня професійної ідентичності* особистості (професійна самооцінка, задоволеність професією, взаєминами, рівень тривожності й т.п.);

е) *рівня соціалізації* й взаємодії в процесі професійної діяльності.

1.3.4. Перспективи фундаменталізації шкільного курсу інформатики. Інформатична освіта має практико-орієнтовану спрямованість, яка полягає в тому, що практика є не лише джерелом нових задач, а й критерієм для вибору можливих напрямів досліджень. Це означає, що прогрес інформатики відбувається як під впливом внутрішніх потреб розвитку, так і під впливом запитів практики (задач, що виникають в математиці, економіці, природознавстві, ін-

женерії, всередині самої інформатики тощо). М.І. Жалдак наголошує, що фундаментальні знання мають важливе значення для прикладних досліджень, а потреби повсякденної виробничої практики викликають і стимулюють пізнавальну діяльність, спрямовану на розкриття законів фундаментального характеру, що в свою чергу є одним з аспектів гуманітаризації освіти [98]. Тому фундаменталізація інформатичної освіти у вищій педагогічній школі сприятиме проникненню ідей фундаменталізації й у шкільний курс інформатики. Як зазначає Ю.В. Триус та О.В. Копаєв, «... навіть з технологічної точки зору на уроках інформатики в середній школі необхідно знайомити учнів з найбільш загальними принципами функціонування систем, в тому числі – й програмних» [147]. М.П. Лапчик наголошує, що «... школі потрібен учитель інформатики з фундаментальними знаннями в галузі інформатики» [169].

Сьогодні більшість школярів, маючи високий рівень мотивації, постійний доступ до сучасних апаратних і програмних засобів, спеціалізованої літератури та ресурсам мережі Інтернет, досить добре володіють багатьма інформаційними технологіями. Нерідкою є ситуація, коли з ряду причин знання або вміння учнів з окремих тем курсу інформатики виявляються більш актуальними або навіть більш детальними, ніж у вчителя, тому навчання інформатики необхідно для систематизації знань школярів, підкреслення фундаментальної складової курсу, знайомства із загальними принципами та підходами до подання та обробки повідомлень.

Фундаментальний характер інформатики як науки суттєво впливає на розвиток змісту навчання інформатики в школі. На думку І.В. Левченко, фундаменталізація шкільного курсу інформатики «повинна означати не переорієнтацію на вивчення в школі основ фундаментальної інформатики як науки, а виділення фундаментальних основ науки та їх дидактичну переробку для освіти школярів за допомогою інформатики для оволодіння школярами соціального досвіду людства» [170, 12].

Фундаменталізація навчання інформатики в школі можлива при реалізації вчителем професійно-педагогічної діяльності з урахуванням ряду умов, в числі

яких адекватне відображення сучасного стану інформатики як фундаментальної науки, подання цілісного курсу інформатики на основі інтеграції змісту навчання навколо системоутворюючих стержнів, формування та розвиток мислення учнів, навчання ефективним способам роботи з інформацією, активне використання внутрішньопредметних і міжпредметних зв'язків курсу інформатики, навчання узагальнених способів застосування сформованих знань і умінь на практиці.

Важливою частиною змісту шкільного курсу інформатики є вивчення інформаційних і комунікаційних технологій (ІКТ), оволодіння школярами вміннями й навичками застосування засобів ІКТ для розв'язування навчальних і практичних завдань. Для багатьох учителів, методистів і авторів підручників інформатики формування вмінь використання засобів ІКТ – взагалі головна (якщо не єдина) ціль цього курсу, що не відповідає загальноосвітньому характеру цього курсу та суперечить державному стандарту. Витоки такої позиції знаходяться в історії введення інформатики в школу під гаслом необхідності «забезпечення комп'ютерної грамотності молоді», що саме по собі вже орієнтувало цей курс переважно на формування вмінь працювати з комп'ютером. Наприкінці 80-х – початку 90-х років XX ст., коли масовими загальнодоступними засобами ІКТ стали так звані «офісні пакети», зміст курсу інформатики орієнтувався більшою мірою на роботу із програмними засобами, що входять до складу цих пакетів. Підручники інформатики середини 90-х рр. усе більше нагадували збірник інструкцій для користувача.

Однак уже через кілька років знову постало питання про використання загальноосвітнього потенціалу інформатики, її внеску у світогляд, розвиток особистості, соціалізацію школярів і т.д. У стандарті [70] й типових програмах з інформатики [212; 96; 263; 110; 97], розроблених під керівництвом М.І. Жалдака, було чітко визначено місце ІКТ в інформатичній освіті, з'явилися питання єдності інформаційних процесів у біологічних, соціальних і технічних системах, інформаційного моделювання, соціальної інформатики та ін. Так, автори експериментального навчального посібника з інформатики для 7 класу [86] спеціаль-

но зосереджували увагу на основних типах та головних функціях програмного забезпечення замість детального розгляду певного програмного продукту, тому такий підручник є фундаментальним.

Відзначимо також, що прийняттю нових позицій щодо цілей і змісту курсу інформатики сприяла ще одна обставина. Критика радянської системи освіти, що розгорнулася на початку 90-х років XX ст., зараз уже перестала носити тотальний і безапеляційний характер, а стала більш конкретною та обґрунтованою. Зокрема, багато в чому справедливі докори відносно надмірно «академічного» характеру шкільної освіти та пропозиції відмовитися від принципів фундаментальності, системності й повноти його змісту, протиставлення їх компетентнісному підходу поступилися місцем більше стриманій позиції. Якщо компетентності – це обізнаність [40], «знання в дії» [358, 3], то діяльність, *дії не можуть бути ефективними, якщо вони не мають системного характеру, не відповідають вимогам повноти й не спираються на фундаментальні знання*. Так само актуальна зараз вимога мобільності освіти може бути реалізована тільки за рахунок фундаментальності освіти. Саме ця якість освіти дає можливість у короткий термін опановувати нові технології та способи діяльності, зробити людину мобільною, затребуваною на ринку праці.

Ці тенденції стають все більш виразними в останні роки, коли зміна поколінь засобів ІКТ відбувається настільки стрімко, що знання, уміння й навички в галузі конкретних версій цих технологій, одержувані в середній школі, втрачають свою актуальність і досить швидко стають незатребуваними.

Сьогодні вже не можна будувати вивчення ІКТ в основному на тренінгу типових умінь роботи з основними засобами цих технологій, орієнтуючи його на численні вправи та розв’язування завдань репродуктивного характеру. Не можуть залишитися осторонь і такі принципи навчання, як фундаментальність, системність, свідомість у навчанні. Повною мірою необхідно задіяти й внутрішньопредметні зв’язки шкільного курсу інформатики, особливо розділи, пов’язані з поданням повідомлень, формалізацією й моделюванням, властивостями алгоритмів тощо.

Відповідно до теорії діяльності, головним змістом навчання повинні бути загальні способи дій для розв'язування широких класів завдань, щоб діяльність учнів була спрямована на оволодіння цими загальними способами. П.Я. Гальперін відзначав, що всі надбання в процесі навчання можна розділити на дві нерівні частини: одну становлять нові загальні схеми речей, які обумовлюють нове їхнє бачення й нове мислення про них, іншу – конкретні факти й закони досліджуваної галузі, конкретний матеріал науки [50]. Освоєння загальних схем вимагає універсальних способів дій, у той час як конкретний матеріал пов'язаний з вузькопредметними, переважно виконавчими діями. Не заперечуючи необхідності формування конкретних дій, найбільшу увагу потрібно приділяти загальним способам дій, пов'язаним із використанням фундаментальних знань, які носять інваріантний характер.

Отже, доцільно було б побудувати зміст навчання ІКТ у курсі інформатики на основі виділення інваріантної частини (наукові основи ІКТ) та варіативної частини (навички роботи з конкретними версіями засобів ІКТ). При цьому варіативна частина могла б скласти основний зміст практичних (лабораторних) робіт із цього курсу. На думку Ю.В. Триуса, саме інваріантна частина повинна бути відображена у підручниках з інформатики, а варіативна – у посібниках, робочих зошитах, практикумах тощо.

Як основні *принципи добору змісту фундаментальних, наукових основ ІКТ* можна виділити такі*:* єдність подання повідомлень для всіх технологій; єдність у методах і засобах опрацювання повідомлень (і даних); побудова ІКТ на основі алгоритмів, що забезпечують автоматизацію опрацювання повідомлень (даних). Таким чином, в якості системоутворюючого поняття при фундаменталізації змісту навчання ІКТ може бути використане поняття «інформаційний процес». Відібрані дидактичні одиниці змісту навчання інформатики, в числі яких вимірювання і кодування інформації, інформаційне моделювання, основи логіки, алгоритмізації, управління та інші є значущими й для фундаменталізації методичної підготовки вчителя інформатики.

І.В. Левченко [170, 8–9] визначено *умови професійно-педагогічної діяль-*

ності вчителя в контексті фундаменталізації навчання інформатики: формування цілісного курсу інформатики на основі інтеграції змісту навчання навколо системоутворюючих стержнів, навчання ефективним способам роботи з інформацією, наповнення навчального матеріалу гуманітарною складовою, адекватне відображення в шкільному курсі сучасного стану фундаментальної науки інформатики, розкриття емоційно-ціннісних та моральних відносин, формування та розвиток мислення учнів, активне використання внутрішньопредметних і міжпредметних зв'язків курсу інформатики, навчання узагальнених способів застосування сформованих знань та вмінь на практиці. Націлювання методичної підготовки вчителів інформатики на створення умов для розкриття учням походження й сутності фундаментальних теоретичних знань, залучення учнів в різноманітні форми спілкування і діяльності, оволодіння узагальненими способами дій, самостійних відкриттів учнями в процесі пошукової діяльності, структурування розумових процесів, прояви багатства і складності розумових операцій (опора на власні думки, самостійність в узагальненнях, критична оцінка одержуваної інформації, активне застосування розумових операцій і т.п.) сприяє підвищенню ефективності навчання школярів інформатики.

Яскраві приклади фундаменталізації шкільного курсу інформатики наведені у роботах І.О. Теплицького, присвячених впровадженню інтегративного курсу моделювання [258; 373]. На жаль, кількість таких робіт досить незначна, тому фундаменталізація шкільного курсу інформатики в Україні все ще залишається практично нерозробленим напрямом, незважаючи на те, що збірник 1987 року «Вивчення основ інформатики та обчислювальної техніки в середній школі: досвід та перспективи» закінчувався тим, що «навчальні програми з усіх предметів в школах ФРН базуються ... на фундаментальних знаннях ..., і інформатика ... не є виключенням. Застосовуваний у ФРН ... підхід у навчанні інформатики заслуговує найпильнішої уваги. Необхідно відзначити також, що в США, ФРН та ... Великобританії спостерігається тенденція до виявлення фундаментальних понять ... у шкільних курсах інформатики. Аналіз зарубіжного досвіду навчання інформатики слід враховувати і при побудові курсу інформа-

тики та обчислювальної техніки у радянській школі» [444, 190–191].

Висновки до розділу 1

Теоретико-методологічний аналіз сутності проблеми фундаменталізації вищої освіти, теоретичний аналіз чинних стандартів вищої освіти, навчальних програм, підручників і навчальних посібників, монографій, дисертаційних досліджень, статей і матеріалів науково-методичних конференцій з проблеми фундаменталізації інформатичної освіти дозволив розробити теоретичні засади фундаменталізації навчання інформатичних дисциплін на основі концепції мобільності та зробити наступні висновки:

1. Концепція фундаментальності для вищої освіти є системоутворюючою.

2. Досягнення цілей фундаменталізації інформатичної освіти можливе через організовану цілеспрямовану педагогічну діяльність, яка забезпечує реалізацію методологічної, професійно-орієнтувальної, розвивальної, прогностичної та інтегративної функцій фундаменталізації освіти:

– опанування методологічно важливими та інваріантними знаннями з довготривалим терміном життя, необхідними для професійної діяльності фахівця в галузі інформаційних технологій;

– тісний зв’язок інформатичної освіти з професійною практичною діяльністю;

– розвиток творчої і пізнавальної активності та самостійності;

– розвиток методичних систем навчання інформатичних дисциплін з урахуванням перспектив розвитку «економіки знань» та інформаційного суспільства;

– системність засвоєння інформатичних дисциплін на основі глибокого розуміння сучасних проблем інформатики.

3. Фундаменталізація змісту навчальної дисципліни надає можливість визначити стійке (інваріантне) ядро змісту, а фундаментальність може бути досягнута, якщо в змісті навчання чітко виокремлені фундаментальні основи навчального предмета, які відповідають фундаментальним основам предметної галузі.

4. Компетентнісний підхід до навчання інформатичних дисциплін є одним із засобів їх фундаменталізації: ключовим у концепції фундаменталізації є принцип наскрізної інтеграції навчальних дисциплін на основі формування інформатичних компетентностей. Ознакою інтегративності навчальних дисциплін може слугувати наступність у розгортанні змісту й структури навчальних дисциплін на основі фундаментальних концепцій інформатики.

5. Інтегративність курсу інформатики визначається фундаментальністю самої науки інформатики та інтегративним характером основних об'єктів її вивчення. При цьому найбільш ефективним засобом інтеграції інформатичних дисциплін у педагогічних ВНЗ є моделювання, яке, крім того, є основою фундаменталізації підготовки майбутніх вчителів інформатики.

6. Перехід до нового покоління галузевих стандартів вищої освіти на основі фундаменталізації навчання та компетентнісного підходу є необхідним етапом на шляху реформування системи освіти в Україні, а застосування компетентнісного підходу до розробки галузевих стандартів вищої освіти створює умови для зближення фундаментальної освіти до потреб та вимог ринку праці, подальшого розвитку освітніх технологій та системи освіти в цілому.

7. Стабілізації курсів інформатики можна досягти поширенням на методичну систему навчання інформатики властивостей відкритих систем: розширюваності, масштабованості, мобільності, інтероперабельності та «люб'язності». Стабілізація ядра змісту та засобів навчання інформатики через інваріантність відносно операційної системи та мови програмування сприяє підвищенню рівня теоретичної підготовки, реалізує міжпредметну інтеграцію, відкриває широкі можливості добору апаратних та програмних засобів навчання інформатичних дисциплін, знижуючи їх вартість за рахунок використання ліцензійно чистого, вільно поширюваного, локалізованого програмного забезпечення.

Основні результати першого розділу опубліковані в роботах [196; 241; 245; 254; 258; 299; 303; 307; 311; 319; 320; 321; 322; 323; 341; 342; 343; 368; 374; 377; 379; 380; 384; 388; 398].

РОЗДІЛ 2
СУЧАСНІ ТЕХНОЛОГІЇ ФУНДАМЕНТАЛІЗАЦІЇ НАВЧАННЯ ІНФОРМАТИЧНИХ ДИСЦИПЛІН

2.1. Поняття про мобільне навчання

Характерною рисою останнього десятиріччя стало активне використання засобів мобільного зв'язку та різноманітних електронних пристроїв у суспільному житті. Сучасний мобільний телефон має функціональність, що не поступається комп'ютерам початкового рівня, а в деяких випадках – і середньої потужності. В першу чергу це стосується смартфонів та персональних комунікаторів (КПК із засобами зв'язку). Поширеність серед користувачів мобільного зв'язку смартфонів та персональних комунікаторів, на думку фахівців, складає біля 10% і має чітку тенденцію до зростання (в Україні на початок 2009 р. – 12,2%) [73].

Мобільне навчання – нова технологія навчання, що базується на інтенсивному застосуванні сучасних мобільних засобів та технологій. Мобільне навчання тісно пов'язане з навчальною мобільністю в тому сенсі, що студенти повинні мати можливість брати участь в освітніх заходах без обмежень у часі та просторі. Використання мобільних технологій відкриває нові можливості для навчання, особливо для тих, хто живе ізольовано або у віддалених місцях чи стикається з труднощами в навчанні. Можливість навчання будь-де та будь-коли, притаманна мобільному навчанню, сьогодні є загальною тенденцією інтенсифікації життя в інформаційному суспільстві.

2.1.1. Технології електронного навчання. Електронне навчання (E-learning) відноситься до великої науково-практичної галузі, що носить загальну назву **автоматизованого навчання** [140, 7]. Її розвиток можна поділити на три етапи:

Перший етап (20–50-ті рр. XX ст.) охоплює період з моменту появи електромеханічних комп'ютерів до широкого впровадження електронних комп'ютерів. Цей етап характеризується застосуванням різних механічних, еле-

ктромеханічних та електронних індивідуалізованих пристроїв, за допомогою яких подавався навчальний матеріал і виконувався контроль (та самоконтроль) знань (технологія ***програмованого навчання***). Як зауважує М.І. Жалдак, «програмоване навчання (в його розумінні біхевіористами 50-х рр. ХХ ст.) назавжди відійшло у минуле і ніколи не відродиться, оскільки окрім технічних в ньому є багато психологічних мінусів, зокрема відхід від цілісного подання і сприймання навчального матеріалу (що є одним з аспектів фундаменталізації навчання)».

Другий етап охоплює період 50–80-х рр. минулого століття та пов'язаний з широким впровадженням ЕОМ у практику, що не могло залишити осторонь фахівців у галузі освіти, тому спочатку з'являються ідеї навчання кібернетики в школі (В.С. Ледньов [172], В.М. Касаткін [117]), впровадженням елементів прикладної математики у навчальний процес (В.М. Монахов [199], М.І. Жалдак [93], Ю.С. Рамський [280]), з'являються комп'ютерно-орієнтовані середовища навчання, автоматизовані системи контролю знань та управління навчальним процесом (О.М. Довгялло [140], В.М. Глушков [52; 53], Ю.І. Машбиць [184; 186]). Ключовими термінами цього періоду стали *інтелектуальні навчаючі системи, комп'ютерно-орієнтовані системи навчання*, *комп'ютерна підтримка навчального процесу*, *комп'ютерні системи контролю знань*. В цей період була створена велика кількість спеціалізованого програмного забезпечення – автоматизованих навчальних систем PLATO, Coursewriter, Tutor та ін. Цьому сприяли очевидні переваги електронних комп'ютерів над електромеханічними – наявність пам'яті для зберігання навчальних матеріалів, висока швидкість опрацювання та розрахунків, більш широкі засоби для перегляду навчальних матеріалів та багато інших. Головним недоліком розробок цього періоду була їхня стаціонарність та автономність, пов'язана з використанням «великих» обчислювальних машин або, в кращому випадку, зв'язаних з ними терміналів. Також було важко реалізувати обмін освітніми ресурсами та послугами між великою кількістю користувачів.

Третій етап (з 80-х рр. минулого століття) розпочався з появою комп'ютерних мереж та персональних комп'ютерів. Виключно потужний ім-

пульс у розвитку освітніх технологій пов'язаний з використанням глобальної мережі Інтернет. Використання спільних та розподілених ресурсів, Web-технології, віддалений доступ до навчального контенту забезпечив суттєве підвищення ефективності професійної підготовки, її доступності та масовості. Мережні технології, висока якість та підвищення ефективності апаратного забезпечення уможливили створення професійних середовищ та систем для надання освітніх послуг і реалізації різних видів формальної (організованої) і неформальної (спеціально не організованої) освіти. Ключовими термінами цього періоду є *Інтернет*, *Web-курси*, *гіпертекст*, *віртуальне навчання*, *віртуальний університет*, *неперервна освіта*, *навчання протягом всього життя*, *дистанційне навчання*, *електронне навчання*, *мобільне навчання*.

Широке впровадження нових інформаційних технологій сприяє розвитку багатьох нових можливостей для управління процесом навчання, навіть у порівнянні з недавнім минулим. Сьогодні у повній мірі може здійснюватись ідея навчання у будь-який час і в будь-якому місці. Сьогодні в Інтернет налічується понад мільйон навчальних курсів, а кількість порталів та віртуальних навчальних закладів сягає 30000.

Багато практичних рішень доведені до рівня професійних систем. Так, системи на ядрі освітньої платформи WebCT використовують близько 5 млн. студентів; для WebCT створено сотні тисяч курсів, розроблених у 40000 університетів і коледжів з 50 країн. Платформа Blackboard – інше професійне широко використовуване середовище електронного навчання в різних предметних галузях. Програмна платформа «e-Learning Shell» (скорочено «eLSe») функціонує під управлінням Windows та Linux на базі безкоштовних продуктів, таких, як Web-сервер Apache з модулем PHP та СУБД MySQL. Ця платформа має функціональність, яка повністю відповідає вимогам Генерального директорату з питань освіти та культури Комісії Європейського Союзу. Прикладами інших середовищ є Acollab, ATutor, Authorware, Claroline, Click2learn, Colloquia, COSE, DodeboLMS, Dokeos, eCollege, ELEDGE, ILIAS, IntraLearn, LAMS, Learning Space, LearnLinc, LON-CAPA, LRN, OLAT, OpenACS, OpenLMS, SAKAI, The

Manhattan Virtual Classroom, Top Class, UniLearn, Virtual-U, Webalmir, WebBoard, Web Course in a Box, WebCT, Агапа, Веб-клас ХПІ, Прометей та багато інших.

Основні передумови та причини для широкого використання електронного навчання:

1. *Вплив інформаційного суспільства*. Інформаційне суспільство, в яке світ вступив з приходом XXI століття, – це суспільство знань. Якщо в індустріальному суспільстві досвід навчання обов'язково був пов'язаний зі школою (середньою або вищою), то в інформаційному суспільстві, коли в результаті розвитку високих технологій одні професії відмирають, інші змінюються, треті народжуються, рівень вимог до професійних якостей робітників та їхньої відповідальності збільшується, тому для того, щоб зберегти свої робочі місця в умовах великої конкуренції, зростати професійно, щоб впоратися зі зростаючими інформаційними потоками та вцілому успішно працювати в рамках динамічного інформаційного суспільства, люди повинні безперервно вчитися – вчитися протягом усього життя.

2. *Глобальність як характерна риса інформаційного суспільства*. Розвиток інформаційних технологій, Інтернету та досягнення в галузі комунікацій роблять суспільство більш відкритим, а його члени стають більш залежними один від одного і мусять постійно розширювати співробітництво. Цьому сприяє тенденція ринку праці до скасування національних кордонів і створення глобального ринку праці. Це неминуче приводить до глобалізації освіти та використання глобальних інформаційних ресурсів і стандартів.

3. *Стрімкий розвиток інформаційних та комунікаційних технологій*. Знаменитий емпіричний закон Мура розвитку електронної елементної бази (напівпровідникових мікросхем), згідно з яким ступінь інтеграції (кількість напівпровідників на одиницю площі кремнієвих пластин) подвоюється кожні 18 місяців, діє і для тактової частоти (швидкодії) мікропроцесорів, передавання даних через канали зв'язку та в інших місцях.

4. *Експоненціальне зростання накопичених людством знань і неможливість їх ефективного засвоєння за допомогою традиційних методів і підходів*. Це вимагає інтенсифікації процесів засвоєння знань, їхньої актуалізації та застосування на практиці.

5. *Практично вичерпані можливості традиційної підготовки кадрів для вирішення завдань нового часу.*

6. *Брак ІКТ-фахівців*: більш ніж 60% вакансій у ЄС – професії в галузі «інтенсивного знання та електронних навичок»; попит на ІКТ-фахівців буде перевищувати пропозицію приблизно на 12% щорічно протягом найближчих років. Одним із шляхів вирішення цієї проблеми є використання нових методів і методик інтенсивного навчання і підготовки, в тому числі електронного навчання.

7. *Освіта стає мета-індустрією*: так,

– у 2004 році глобальний ринок електронного навчання мав оборот понад 23,1 млрд. доларів США;

– до 2004 року США мали частку у 65,2% ринку, в той час як сьогодні Західна Європа є найбільш динамічним ринком освітніх послуг;

– у 2000 році частка електронного навчання в сфері ІКТ складала 24%, в той час як в 2005 році – вже 53,8%;

– у 2005 році електронне навчання є найбільш часто використовуваним у корпоративній ІТ-освіті;

– з’явилась нова, швидко зростаюча галузь – електронне навчання, – зі своїми виробниками контенту, видавцями електронних матеріалів, розробниками програмних засобів, послуг, порталів тощо;

– зростає тенденція до співробітництва університетів та великих ІТ-компаній, створення віртуальних університетів, віртуальних навчальних центрів.

Серед основних характеристик та особливостей електронного навчання можна виділити:

– можливість інтерактивної взаємодії між викладачем і студентом в режимі діалогу, що у деяких випадках може наближатися до діалогової взаємодії у традиційних навчальних технологіях;

– швидке розсилання/одержання навчальних матеріалів в електронному поданні;

– оперативний доступ до інформаційних ресурсів, що містяться в мережі Інтернет;

– можливість перевірки та контролю знань у дистанційному режимі;

– можливість організації лабораторних практикумів у віртуальному режимі через реалізацію віддаленого мережного доступу до реального лабораторного обладнання;

– створення «віртуальних груп» для оперативної взаємодії між учнями;

– можливість накопичення статистичних даних та на основі їх аналізу управляти навчанням;

– підвищення якості навчання та управління;

– впровадження автоматизованого управління якістю навчання;

– індивідуалізація професійної підготовки шляхом створення індивідуальних графіків навчання для окремих студентів.

Слід зазначити, що Україна сьогодні наближається до середньої стадії впровадження електронного навчання. Так, Державною програмою «Інформаційні та комунікаційні технології в освіті і науці» [260] на 2009-2010 роки передбачено наступні напрями роботи:

1. Розроблення, впровадження та легалізація програмного забезпечення:

– створення банку електронних документів нормативно-правового, науково-методичного, психолого-педагогічного, організаційного, програмно-технологічного та інформаційного забезпечення дистанційного навчання;

– створення та впровадження програмних засобів пілотної системи поточного і підсумкового контролю знань студентів у вищих навчальних закладах;

– створення та впровадження програмних засобів для уніфікованої системи дистанційного навчання.

2. Створення, зберігання та доступ до інформаційних ресурсів:

– створення Інтернет-порталу дистанційного навчання;

– створення Інтернет-порталу інформаційних ресурсів освіти і науки;

– забезпечення доступу до національних і світових інформаційних ресурсів;

– сертифікація та атестація програмних засобів та курсів дистанційного навчання.

3. Розроблення систем забезпечення інформаційної безпеки функціонування мереж та інформаційних ресурсів, зокрема – розроблення програмно-технічних систем забезпечення захисту інформаційних ресурсів від несанкціонованого доступу.

Вибір напрямів зумовлений тим, що впровадження інноваційних методів електронного навчання, зокрема, дистанційного, у практику діяльності ВНЗ ще є недостатньо широким. Так, опитування студентів криворізьких ВНЗ, в яких функціонують системи дистанційного навчання (Криворізький технічний університет, Криворізький державний педагогічний університет, Криворізький економічний інститут Київського національного економічного університету ім. В. Гетьмана, Криворізький металургійний факультет Національної металургійної академії України), показало, що лише 13% опитаних студентів використовують електронні ресурси навчальних закладів.

Опитування показало, що ефективність університетських Web-сайтів у найбільш важливому аспекті академічного спілкування – предметно-змістова підтримка навчальних інформаційних ресурсів (курсів) – є занадто низькою: лише 18% студентів використовували доступ до розміщених на сайтах курсів.

Дані про стан електронного навчання в нашій країні та в усьому світі свідчать про нагальну необхідність його стимулювання, щоб забезпечити динамічний і прогресивний розвиток та впровадження на всіх рівнях освіти, перш за все, – вищої, тому що *електронне навчання є інноваційною технологією*, спрямованою на професіоналізацію та підвищення мобільності тих, хто навчається, і

на сучасному етапі розвитку ІКТ воно може розглядатися як ***технологічна основа фундаменталізації вищої освіти***.

2.1.2. Мобільне навчання та рівний доступ до ІКТ. Розвиток інформаційних технологій призвів до появи нового соціального явища – *цифрового бар'єру* (digital divide; інші назви – цифрова нерівність, цифровий поділ): обмеженню можливостей соціальної групи через відсутність у неї доступу до сучасних засобів комунікації, тобто нерівний доступ членів суспільства до ІКТ. Виникнувши в середині 90-х рр. XX ст., даний термін спочатку характеризував лише можливість доступу до комп'ютерного обладнання, проте згодом став характеризувати інформаційні технології вцілому.

Подолання цифрового бар'єру в системі освіти можливо лише через забезпечення рівного доступу до неї за допомогою засобів ІКТ, тому цілком природно, що даний напрям є провідним у вітчизняній методиці навчання інформатики. Надання закладам освіти сучасних технічних засобів ІКТ створює умови для організації електронного навчання, а їх об'єднання засобами Інтернет – й для організації дистанційного навчання. Водночас поза увагою дослідників залишаються різноманітні електронні пристрої, насамперед, смартфони та персональні комунікатори, широко поширені серед учнів старшої школи та студентів. Проте наказом МОН України від 24.05.2007 №420 «Про використання мобільних телефонів під час навчального процесу» заборонено використання мобільних телефонів у загальноосвітніх та професійно-технічних навчальних закладах під час проведення навчальних занять.

На користь такої заборони наводяться наступні аргументи: «По-перше, наявність телефону дає можливість своєчасно повідомити батькові чи матері про те, що дитина закінчила навчання, та дає можливість оперативного зв'язку дитини з батьками. Але (уявіть ситуацію) коли йде урок і в класі сидить 30 учнів і через кожну хвилину дзвонить мобільний телефон в одного чи іншого учня. Чи можливо в таких умовах проводити повноцінний навчальний процес? По-друге, під час виконання самостійних, контрольних робіт в студентів та учнів часто виникає ідея використовувати мобільний телефон як джерело списуван-

ня. ... До цих всіх факторів додається ще одна проблема – більшість школярів користуються телефонами з відеокамерами, і пристроями доступу до мережі Інтернет. Ці телефони перетворюються на засіб поширення юнаками і дівчатами сцен насилля» [351].

Керівниками багатьох ВНЗ також заборонено *використання мобільних телефонів у навчальному процесі*, що призвело до створення унікальної ситуації – офіційної заборони потужного технічного засобу навчання, адже головним аргументом на користь такої заборони є те, що **мобільні пристрої є ефективним засобом ІКТ, не контрольованим викладачем** (мають доступ до Інтернет, можуть бути використані як джерело списування тощо).

Зауважимо, що рішення МОН України було викликане до життя тими самими причинами, які у 2000 р. спонукали Генеральний секретаріат Національної вчительської спілки Великобританії оприлюднити сумарну реакцію вчителів на мобільні пристрої в школі: «Мобільні телефони не призначені для використання протягом навчального дня – особливо під час уроків. Процес навчання порушується не лише у тієї особи, що телефонує або якій телефонують, а у всього класу» [520].

На початок 2008 р. в світі нараховувалося більше 1,5 мільярдів мобільних телефонів та комунікаторів – майже втричі більше, ніж комп'ютерів. Потужність більшості мобільних пристроїв перевищує потужності персональних комп'ютерів середини 90-х рр., які й досі використовуються в процесі навчання, тому нехтування потенціалом педагогічно виваженого і доцільного використання цього класу пристроїв є неприйнятним, так само, як їх застосування без відповідної методики навчання.

Проблему використання мобільних телефонів у навчальному процесі слід розглядати як один із проявів конфлікту між навчанням в аудиторії та поза нею:

повсюдне навчання	*аудиторне навчання*
– центральна особа – той, хто вчиться;	– центральна особа – той, хто вчить;
– індивідуалізоване;	– інституціоналізоване;
– спільне;	– індивідуальне;

повсюдне навчання	*аудиторне навчання*
– ситуативне;	– деконтекстуалізоване;
– у довільному місці;	– зафіксоване в місці;
– протягом всього життя.	– обмежене в часі.

Можна спробувати розширити аудиторне навчання до повсюдного: для цього достатньо використати записи лекцій, створити навчальні середовища для мобільних телефонів та засоби оцінювання навчальних досягнень, забезпечити доступ з дому до інтранет-мережі навчального закладу. Проте ці дії не забезпечують повноцінного повсюдного навчання.

Можна спробувати розширити і повсюдне навчання: для цього необхідно забезпечити мережну безпеку, заборонити персональні пристрої, залишивши право студенту самостійно обирати шлях навчання. Однак при цьому втрачається контроль з боку викладача.

На розв'язання цього конфлікту і спрямоване мобільне навчання, складовими якого є:

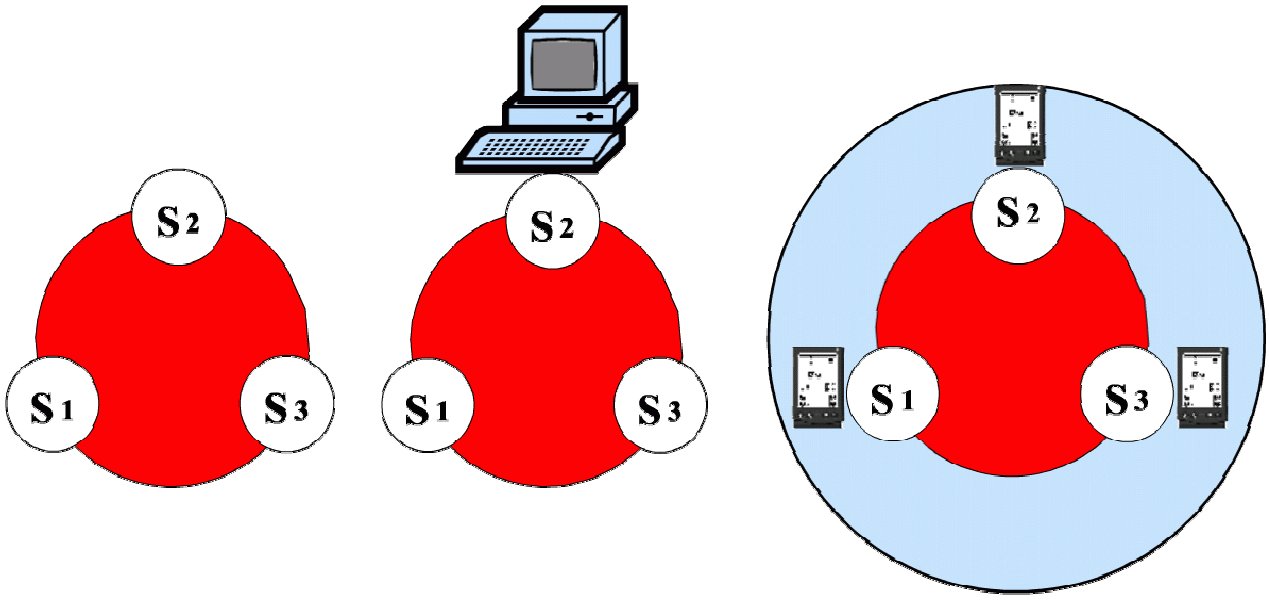
1) *навчання з використанням портативної техніки* (фокусується на технології і може бути реалізоване й у стаціонарному варіанті – наприклад, в класі);

2) *контекстно-чутливе навчання* (фокусується на суб'єкті навчання і може бути реалізоване за допомогою портативних або стаціонарних технологій);

3) *навчання в мобільному світі* (фокусується на мобільному суспільстві).

Тому можна розглядати *мобільне навчання* (mobile learning, M-Learning) як сучасний напрям розвитку систем дистанційного навчання із застосуванням мобільних телефонів, смартфонів, КПК, електронних книжок [308]. Технологія мобільного навчання передбачає наявність системи дистанційного навчання, яка включає в себе підсистему доступу до локального та віддаленого контенту. В порівнянні з традиційним в мобільному навчанні з'являється можливість моніторингу навчання в реальному часі та забезпечується висока насиченість контенту, що дозволяє розглядати його не лише як засіб навчання, а й як інструмент спільної роботи, призначений для підвищення якості навчання [180]

(рис. 2.1). Таким чином, **мобільне навчання є новим засобом подолання цифрового бар'єру**.



а *б* *в*

Рис. 2.1. Схема традиційного, комп'ютерно-орієнтованого та мобільного комп'ютерно-орієнтоване співробітництва

Слід зауважити, що запропоноване визначення мобільного навчання є частково *техноцентричним* – це провідний, проте не єдиний підхід до класифікації. Можна виділити ще принаймні три:

– *по відношенню до електронного навчання*. Цей підхід характеризує мобільне навчання як розширення електронного навчання (так, Т. Георгієв означає його як «нову стадію розвитку електронного навчання» [496, 1]), проте всеосяжність такого означення не допомагає визначити характерні властивості мобільного навчання. Дж. Тракслер взагалі стверджує, що в означеннях через техноцентричний підхід (або через електронне навчання) лише шукають місце «мобільного навчання десь в спектрі портативності електронного навчання» [465, 7];

– *по відношенню до формальної (спеціально організованої) освіти*. У літературі з мобільного навчання формальна освіта часто характеризується як навчання «віч-на-віч», проте форми дистанційної освіти (наприклад, дистанційне

заочне навчання) існували протягом понад 100 років [532], що породжує питання про місце мобільного навчання по відношенню до всіх форм традиційного навчання (під *традиційним* навчанням розуміємо організоване та кероване викладачем навчання «віч-на-віч» у групі, у визначений час та у певному місці);

– *по відношенню до суб'єкта навчання*. Цей напрям прослідковується в дослідженні з концептуального мобільного навчання М. Шарплеса, Дж. Тейлора, К. О'Маллі та їх колег. В ранніх роботах колективу, керованого М. Шарплесом, концепція мобільного навчання була тісно пов'язана з пристроєм [552] та сприятливими можливостями для безперервного навчання [553]. Однак із часом, у відповідності до визначених у п. 1.1.1 вимог мобільності навчання, увагу з пристрою було перенесено на учня. Це привело до розгляду мобільного навчання стосовно учня і визначення його як «будь-якого виду навчання, що відбувається, коли учень не має фіксованого, наперед визначеного місця, або навчання, що відбувається, коли учень використовує для навчання можливості пропонованих мобільних технологій» [465, 8]. В останніх роботах колективу (наприклад, [549]) розробляється контекстний аспект теорії мобільного навчання.

Таким чином, у нашому означенні мобільного навчання охоплені принаймні три з чотирьох напрямів визначення мобільного навчання. Для з'ясування ваги кожного з цих напрямів розглянемо місце мобільного навчання серед інших технологій.

2.1.3. Місце технологій мобільного навчання серед технологій автоматизованого навчання. Інтерес до загальних проблем і аспектів реалізації електронного навчання постійно зростає. Цей інтерес є очевидним як з боку працівників сфери освіти, які його безпосередньо організовують та реалізують, так і з боку студентів, котрі навчаються на педагогічних та інформатичних спеціальностях («Інформатика», «Комп'ютерні науки», «Інформаційні системи та технології» тощо). Кількість публікацій в галузі електронного навчання постійно зростає. Переважна більшість цих матеріалів розміщена в Інтернет. Якщо скористатися послугами пошукової системи Google, то при введенні ключо-

вих фраз «електронне навчання, Е-навчання» українською мовою отримаємо 142 посилання, російською – 376 посилань, англійською – близько 46 млн. посилань; при введенні ключових фраз «мобільне навчання, М-навчання» українською мовою отримаємо 17 посилань (з них 2/3 належать автору), російською – 171 посилання, англійською – близько 1,7 млн. посилань. Красномовним доповненням до кількості посилань на мобільне навчання у вітчизняних Інтернет-джерелах є розділ «Мобільне навчання» форуму Львівського державного університету безпеки життєдіяльності (http://ubgd.lviv.ua), що не містить жодного повідомлення.

Практично цілковита нерозробленість проблем мобільного навчання поєднується з високим інтересом до нього як з боку педагогічних працівників, так і з боку студентів. Проте єдина згадка про мобільне навчання з боку державних органів управління освітою відноситься до вже згадуваного вище наказу МОН України від 24.05.2007 №420 «Про використання мобільних телефонів під час навчального процесу», в четвертому пункті якого директору Українського державного центру позашкільної освіти спільно із операторами стільникового зв'язку необхідно було до 1 вересня 2007 року розробити Положення Всеукраїнського конкурсу на краще використання мобільних телефонів для отримання відомостей в усіх галузях знань. Станом на 1 грудня 2008 року таке положення все ще не розроблено.

Для визначення місця та ролі мобільного навчання у фундаменталізації навчання інформатичних дисциплін коротко розглянемо основні сучасні технології автоматизованого навчання.

2.1.3.1. Технології дистанційного навчання. Дистанційне навчання виникло у зв'язку з необхідністю забезпечення безперервного навчання протягом всього життя та завдяки специфічним можливостям розвитку найважливіших пізнавальних здібностей при його використанні.

Дистанційне навчання не є новою формою освіти: його історія налічує понад 150 років застосування та традицій. Його головною особливістю є просторова та часова відстань між викладачем та студентом.

Дистанційне навчання не слід ототожнювати із заочним навчанням, де студенти отримують навчальний план та програми курсів, певну кількість очних занять, далі готуються індивідуально та атестуються. Основна ідея дистанційного навчання – «дати» знання студенту, який, як правило, далеко від місця зберігання їх джерел. У минулому, коли ще не було сьогоднішніх засобів електронних комунікацій [6], для цього використовувалися друковані матеріали, що надсилалися звичайною поштою, а зворотний зв'язок відбувався через листування (кореспондентська освіта).

Не викликає запитань стосовно користі дистанційного навчання для суспільства: його застосування вивільняє діяльність студентів, щоб вони могли навчатися у будь-який час, в будь-якому місці і у такий спосіб, що відповідає їхній зайнятості. Незважаючи на ці переваги, перші 100 років дистанційного навчання були відзначені критикою. Великий прорив в якості та кількості наступив у 1970-х рр., коли в СРСР було широко запроваджена заочна форма навчання, а в Європі створені відкриті університети (зокрема, Відкритий університет Сполученого Королівства в Мілтон Кейнс [509] та Національний університет дистанційного навчання в Мадриді), які змінили статус дистанційного навчання, зробивши його міжнародно-визнаною університетською освітою.

За Д. Кіганом, дистанційна освіта має п'ять основних форм реалізації [507]:

1) дистанційне навчання – забезпечення освіти та навчання на відстані через відкриті університети, інститути дистанційного навчання та департаменти дистанційної освіти традиційних інститутів;

2) електронне навчання – навчання через Інтернет із застосуванням LMS;

3) синхронне електронне навчання – електронне навчання з активним зворотним зв'язком;

4) Інтернет-лекції – поширення відеолекцій засобами інтранет, соціальних мереж та WWW;

5) мобільне навчання – навчання за допомогою кишенькових комп'ютерів, смартфонів і мобільних телефонів.

2.1.3.2. Технології електронного навчання. У найширшому розумінні електронне навчання може розглядатися як навчання, що здійснюється та підтримується за допомогою електронних засобів та електронних середовищ. Ці електронні засоби можуть бути спеціалізованими або універсальними, такими як комп'ютер, що відноситься до класу складних електронних пристроїв. Таким чином, будь-яке навчання за допомогою комп'ютера може бути віднесене до електронного навчання.

В тлумаченні, запропонованому Європейською комісією, *під електронним навчанням розуміють процес формування знань, умінь та навичок за допомогою або повністю через Інтернет* [479].

У відповідності до тлумачення Європейської комісії, електронне навчання можна розглядати як різновид дистанційного навчання. З цієї точки зору воно є дистанційним навчанням, реалізованим у Інтернет-середовищі зі способом передавання навчальних матеріалів в електронному вигляді. Таке тлумачення включає в себе онлайн-навчання, Web-орієнтоване навчання, віртуальні університети та класи, «цифрову» співпрацю та технологічну підтримку дистанційного навчання. Проте саме ці складові Є.М. Смирнова-Трибульска [333] включає до сучасного тлумачення дистанційного навчання, тому в Україні більш поширеним є таке тлумачення: *електронне навчання* (e-learning) – це подання навчальних матеріалів та управління процесом навчання за допомогою нових інформаційних і телекомунікаційних технологій. Ми дотримуємося трактування електронного навчання як навчання, що підтримується та стимулюється застосуванням ІКТ.

Елементами системи електронного навчання, спільними з дистанційним навчанням, є:

– *змістові об'єкти*: навчальний матеріал поділений на модулі, що містять об'єкти різної природи – текст, графіку, зображення, аудіо, анімацію, відео тощо. Як правило, вони зберігаються в базі даних і доступні в залежності від потреб суб'єктів навчання. Результатом є індивідуалізація навчання – студенти отримують лише те, що їм потрібно, засвоюючи знання у бажаному темпі;

– *спільноти*: студенти можуть створювати Інтернет-спільноти для взаємодопомоги та обміну повідомленнями;

– *експертна онлайн-допомога*: викладачі-експерти доступні в мережі для проведення консультацій, надання відповідей на питання, організації обговорення;

– *можливості для співпраці*: за допомогою відповідного програмного забезпечення можна організувати онлайн-конференції, спільну роботу над проектом студентів, географічно віддалених один від одного;

– *мультимедіа*: сучасні аудіо- та відеотехнології подання навчальних матеріалів з метою стимулювання прагнення студентів до здобування знань та підвищення ефективності навчання.

Основні переваги електронного навчання:

– *індивідуалізація навчання*: засоби самонавчання надають можливість студентам, виходячи з власних можливостей, обирати тип, темп та спосіб отримання матеріалів на основі власних уподобань;

– *скорочення витрат на навчання*: в системі неформальної освіти студенти можуть суттєво знизити або навіть ліквідувати витрати на навчання – в усіх інших випадках вартість електронного навчання порівнянна чи навіть вища, ніж традиційного денного навчання;

– *швидкий та простий доступ до навчальних матеріалів*: користувачі можуть отримати доступ до навчального контенту з будь-якого місця, де є з'єднання з Інтернетом;

– *можливість спільного навчання* через обмін та спільне використання освітнього контенту кількома пов'язаними між собою користувачами;

– *звітність*: навчання, контроль знань, оцінювання та моніторинг навчального процесу, накопичення кредитів та проходження навчальних програм і планів та отримання сертифікату результати навчання автоматизовані. При цьому зберігаються різні дані, які можуть бути використані для адміністративного контролю за процесом навчання та формування різних звітів.

2.1.3.3. Технології мобільного навчання. В літературі пропонуються різні техноцентричні тлумачення мобільного навчання, спільним в яких є те, що за цієї технології навчання фізичне з'єднання з кабельною мережею є необов'язковим [519]. З цієї точки зору мобільне навчання може бути визначено як *підхід до навчання, при якому на основі мобільних електронних пристроїв створюється мобільне освітнє середовище, де студенти можуть використовувати їх у якості засобу доступу до навчальних матеріалів, що містяться в Інтернеті, будь-де та будь-коли*.

Мобільне навчання є, з одного боку, різновидом дистанційного навчання, а з іншого – електронного навчання (рис. 2.2). У порівнянні з електронним та дистанційним навчанням мобільне надає суб'єкту навчання більшу кількість «ступенів вільності» – вищу інтерактивність, більшу свободу руху, більшу кількість технічних засобів, основними з яких є UMPC – ультрамобільні ПК (Intel Classmate, Asus EEE, XO-1), Tablet PC – планшетні ПК, надпортативні ноутбуки, PDA (персональні цифрові помічники), аудіопрогравачі для запису та прослуховування лекцій, мультимедійні путівники музеями, мультимедійні ігрові консолі, електронні книжки, мобільні телефони, смартфони та багато інших [495].

Рис. 2.2. Співвідношення електронного, дистанційного та мобільного навчання

Д. Кіган, визначаючи мобільне навчання, виділяє мобільність та функціональність як критерії поділу технологій навчання:

функціональність		*мобільність*		
персональні комп'ютери	ноутбуки	кишенькові комп'ютери	смартфони	мобільні телефони
електронне навчання		**мобільне навчання**		

До особливостей мобільного навчання М. Шарплес відносить: спільну онлайнову роботу над проектом, моблоггінг (мобільний блоггінг), персоналізоване навчання, роботу у групах, онлайнові дослідження, рівний доступ до навчання [546].

Дж. Тракслер виділяє кілька напрямів реалізації мобільного навчання [561]:

– технологічно орієнтоване мобільне навчання – окремі конкретні технологічні інновації, впроваджені у навчальний процес для демонстрації технічних переваг та педагогічних можливостей;

– мініелектронне навчання – мобільні, бездротові і портативні технології, які використовуються для повторного впровадження рішень і підходів, що вже використовуються в «звичайних» електронних засобах навчання, можливо, перенесення деяких технологій електронного навчання, таких, як віртуальні навчальні середовища (VLE), на мобільні платформи, або використання мобільних технологій як гнучкої заміни статичних настільних технологій;

– поєднання мобільного навчання та навчання у класі – ті ж самі технології використовуються для підтримки спільного навчання в класі, можливо, в поєднанні з іншими технологіями, такими як сенсорні дошки;

– неформальне, персоналізоване, ситуативне мобільне навчання – ті ж технології з додатковою функціональністю, наприклад, залежні від місця розташування;

– мобільні тренінги – технології, що використовуються для підвищення продуктивності та ефективності мобільних працівників шляхом надання матеріалів та підтримки «точно у термін» і в контексті їхніх першочергових пріоритетів;

– віддалене (сільське) розвиваюче мобільне навчання – технології використовуються для вирішення екологічних та інфраструктурних проблем та підтримки освіти там, де «звичайні» електронні технології навчання не працюють.

Мобільне дистанційне навчання може реалізуватися за будь-яким з цих напрямів в залежності від інфраструктури (енергоживлення, поштові послуги,

Інтернет і т.д.), розрідженості комунікативного простору (нечасті особисті контакти, відсутність технічної підтримки і т.д.), розвиненості дистанційного навчання тощо.

Іноді окремо виділяють *віртуальне навчання*, під яким розуміють всі форми та підходи до навчання з використанням Інтернет [510]. Це електронне навчання за означенням Європейської комісії, або об'єднання мобільного та електронного навчання за нашим означенням.

Основне призначення мобільного навчання полягає в тому, щоб покращити знання людини в тій галузі, в якій вона бажає, і в той момент, коли їй це потрібно.

Завдяки сучасним технологіям мобільного зв'язку (взаємодія «студент–викладач» здійснюється в високошвидкісному середовищі обміну повідомленнями) через мобільне навчання забезпечується високий ступінь інтерактивності, що має вирішальне значення для навчання.

У мобільному навчанні пропонуються нові засоби дистанційного навчання, засновані на мобільному зв'язку, комп'ютерних та мережних технологіях. Це досягається за рахунок використання мобільних і портативних пристроїв, таких як КПК, смартфони, портативні комп'ютери та електронні записники. При цьому потрібно мати можливість під'єднуватися як до інших комп'ютерних пристроїв, так і до глобальної мережі Інтернет для надання навчальних матеріалів та здійснення двостороннього обміну повідомленнями між учасниками навчального процесу (студенти, викладачі, провайдери мобільних послуг та провайдери Інтернет).

В якості однієї з основних передумов для розвитку мобільного навчання слід вказати експоненціальний розвиток мобільного зв'язку та технологій. Наведемо лише деякі статистичні відомості на початок 2008 року, що відносяться до збільшення частки продаж мобільних телефонів, персональних цифрових помічників, смартфонів і збільшення кількості користувачів послуг мобільних комунікацій:

– більш 60% робочих місць у США є мобільними;

– у всьому світі продано понад 975 мільйонів стільникових телефонів з вбудованими засобами мережних комунікацій;

– продаж КПК і смартфонів виріс більш ніж на 200% у порівнянні з 2007 роком;

– кожного кварталу кількість компаній, що розробляють програмне забезпечення для мобільних пристроїв, збільшується на 1000;

– число абонентів мобільного зв'язку в Україні перевищило чисельність її населення;

– продажі багатофункціональних мобільних пристроїв перевищують кількість проданих персональних комп'ютерів.

За прогнозами аналітиків до 2010 року понад 1,5 мільярди користувачів будуть використовувати бездротовий Інтернет, а кількість користувачів мобільного зв'язку перевищить 4 мільярди.

Поряд з розвитком мобільного зв'язку зростає і потреба швидко отримувати різноманітні знання, з огляду на значну мобільність населення (понад 50% співробітників компаній витрачають 50% робочого часу поза межами офісу).

Основні проблеми, що стоять перед корпоративним мобільним навчанням:

1) для організацій:

– зняття бар'єрів для обміну досвідом і використання всіх наявних інформаційних ресурсів компанії;

– швидке створення нових інформаційних ресурсів.

2) для індивідуальних користувачів:

– отримання доступу до великого обсягу різноманітних інформаційних ресурсів з різних галузей діяльності людини;

– надання можливостей публікації своїх повідомлень і нових ідей.

2.1.4. Мікронавчання як психологічна основа мобільного навчання. Чарльз Вебер, розглядаючи «швидке навчання у швидкозмінному середовищі» [568], сформулював концепцію *мікронавчання* (microlearning), сутність якої полягає у вивченні порівняно невеликої частини навчального матеріалу (що цілком уміщується на екрані КПК чи мобільного телефону) та короткотермінового

навчання. Найчастіше цей термін використовується в галузі електронного навчання та суміжних галузях.

У широкому сенсі мікронавчання можна розуміти як метафору, яка відноситься до мікроаспектів різних навчальних моделей, концепцій і процесів. Тео Хуг відзначає, що «немає різниці, відноситься навчання до процесу створення та організації знань, зміни поведінки, відносин, цінностей, розумових здібностей, когнітивних структур, емоційних реакцій, дій, моделей або соціальних аспектів, у всіх випадках ми маємо можливість розглянути мікро-, мезо- і макроаспекти різних точок зору» [518, 4].

В залежності від сфери застосування, мікро-, мезо- і макроаспекти різняться. Наприклад, у контексті вивчення мови, можна було б ввести мікроаспекти з точки зору словників, фраз, пропозицій, і відрізняти їх від ситуацій та епізодів (мезоаспекти) та соціально-культурної специфіки або складної семантики (макроаспекти).

Як у навчальній технології, головна увага в мікронавчанні приділяється розробці мікронавчальних засобів на основі мікрокомпонент в цифрових медіасередовищах, що вже є повсякденною реальністю для сучасної освіти. Ці компоненти можуть бути включені у повсякденне життя – на відміну від традиційного електронного навчання, мікронавчання має тенденцію до застосування технологій просування контенту (подібних до тих, які використовуються в засобах масової інформації), що знижує когнітивне навантаження на студентів. Таким чином, вибір мікронавчальних об’єктів, а також темпів та термінів навчання за допомогою мікрозаходів має важливе значення й для методики електронного навчання.

Основні характеристики мікронавчання:

– мікронавчальні процеси часто випливають із особливостей роботи з мікроконтентом, розміщеним у середовищі електронного навчання або таких засобах, як Web-блог або закладки в соціальних мережах [518, 99];

– процес навчання може охоплювати від кількох секунд (наприклад, в мобільному навчання) до 15 хвилин або більше;

– мікронавчання також можна розуміти як процес часткової, «короткої» навчальної діяльності, тобто навчання через опрацювання об’єктів мікроконтенту за малий час.

Мікронавчання – це термін, який може бути використаний для опису неформального навчання та отримання знань в мікроконтентних, мікромедіа [472] чи багатозадачних середовищах, особливо тих, які ґрунтуються на Web 2.0 та бездротових Web-технологіях.

Для опису мікронавчальної діяльності використовуються наступні поняття:

а) *час*: порівняно короткі зусилля, оперативні витрати, рівень використання часу, вимір часу, суб’єктивний час і т.д.;

б) *контент*: дрібні або дуже дрібні частини навчального матеріалу, вузькі теми, прості питання і т.д.;

в) *навчальна програма*: мала частина програми, частина модуля, елементи неформального навчання і т.д.;

г) *форма*: фрагменти, частини, епізоди, елементи майстерності і т.д.;

д) *процес*: окремі, супровідні, ситуативні чи комплексні заходи, ітераційний метод і т.д.;

е) *середовище*: друковане, електронне, мономедіа, мультимедіа, інтермедіа і т.д.;

ж) *тип навчання*: повторювальне, активне, рефлективне, прагматичне, концептуальне, конструктивістське; також: класно-урочне навчання, корпоративне навчання і т.д.

Приклади мікронавчальної діяльності:

– читання абзацу тексту, електронної пошти або SMS;

– прослуховування короткого інформаційного подкасту або перегляд освітнього відеокліпу;

– перегляд флеш-карти;

– запам’ятовування слів, словосполучень, визначень або формул;

– вибір відповіді на запитання;

– сортування набору елементів мікроконтенту у (хроно)логічному порядку;

– відповідь на запитання відкритого типу;

– навчання в ході мікрогри;

– написання фрагменту програми (короткої функції);

– складання короткого вірша.

Приклади програмних засобів мікронавчання:

– зберігачі екрану, за допомогою яких користувачеві пропонується розв'язати невеликі серії простих завдань після певного періоду його неактивності;

– тести з багатоваріантним вибором на мобільних телефонах з використанням SMS або мобільних додатків (зокрема, мідлетів);

– щоденне «Слово дня» на RSS-каналі або електронною поштою;

– ПЗ на флеш-карті для запам'ятовування матеріалу з повторенням через певний інтервал часу.

Зазначимо, що мікронавчання не слід розглядати як спробу формалізації навчального процесу з максимально можливим усуненням суб'єктивного фактору безпосередньої взаємодії між викладачем та студентом – покладений в основу програмового навчання підхід, що не виправдав себе. Водночас за рядом ознак мікронавчання можна вважати сучасною реалізацією лінійного алгоритму програмованого навчання, пристосованого для використання у мобільному середовищі. Мікронавчання не обмежується традиційним навчальним процесом, тому за його допомогою можна «навчатися поза навчанням» (неформальне навчання), «навчатися, навчаючи» (за С. Пейпертом), «навчатися в процесі гри» тощо, більш активно та ефективно використовуючи час.

Мобільне навчання виступає одним із способів реалізації мікронавчання, надаючи можливість навчатися у будь-які малі фрагменти вільного часу, тому **мобільне навчання забезпечує більшу навчальну мобільність в порівнянні з електронним або традиційним навчанням**.

2.2. Історія мобільного навчання

2.2.1. Витоки мобільного навчання. Перша згадка про мобільне навчання зустрічається в роботі Дж. Дьюї «Демократія та освіта» (1916 р.): «Ми побачимо мобільне суспільство, насичене каналами поширення змін, що відбуваються будь-де, лише тоді, коли його члени будуть освічені, ініціативні та адаптивні» [482, 88]. За часів Дж. Дьюї такими каналами комунікації були міграційні потоки, трансатлантичний радіозв'язок і навіть – світова війна, сьогодні таким каналом є, насамперед, Інтернет. «Не тільки соціальне життя ідентичне комунікації, але й усі комунікації (і, отже, все справжнє соціальне життя) є освітніми. Для того, щоб стати суб'єктом комунікації, необхідно розширити та змінити свій досвід» [482]. Під комунікацією Дж. Дьюї розумів не лише передавання та отримання повідомлень (інформаційний аспект), а й обмін досвідом (освітній аспект). Згідно з Дьюї, комунікація є головним освітнім процесом. Концепція освіти як ліберального обміну досвідом породила філософські та методологічні питання, які набувають нового наповнення в епоху мобільного зв'язку. Викладач не має онтологічно привілейованого становища, а є просто одним з учасників процесу «навчального мовлення». Клас перетворюється на інформаційно-комунікаційне середовище, що регулюється із зовнішнього боку навчальним планом та системою випробувань.

Мобільність освіти є принциповою характеристикою єдиного освітнього простору, на формування якого спрямований, зокрема, й Болонський процес.

Свій початок комп'ютеризоване мобільне навчання бере з проекту Dynabook Алана Кея, який наприкінці 50-х рр. працював на Денверській військово-повітряній базі «Рендольф», де писав на машинному коді програми для ЕОМ Burroughs 220. Саме тоді він зіткнувся з проблемою передавання сформованих на цій ЕОМ даних на комп'ютери інших баз. Стандартних форматів та ОС для цих ЕОМ не існувало, тому А. Кею довелося створити мікропрограми, що містили всі необхідні коди та після їх запуску на інших машинах (через простий інтерфейс користувача) автоматично розгорталися необхідні дані. Такі програми А. Кей назвав модулями, в яких об'єднуються дані та код. У 1966 р.

він зайнявся науковою діяльністю в галузі молекулярної біології в Університеті штату Колорадо, де запропонував створити системи модулів (об'єктів), в яких об'єднуються дані та алгоритми їх опрацювання, взаємопов'язані один з одним і синхронно функціонуючі через визначені розробником інтерфейси. При цьому він активно використовував аналогії з біологічними об'єктами та механізмами взаємозв'язків і взаємовпливів клітин у живому організмі.

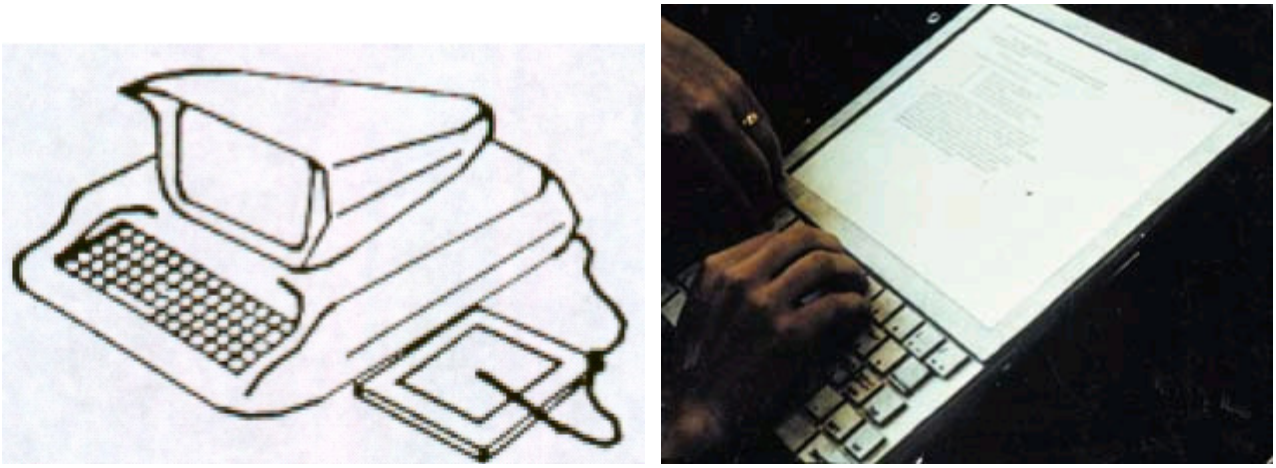
Пізніше А. Кей перейшов до Стенфордської лабораторії штучного інтелекту, а в 1972 р. – у відомий науковий центр Xerox PARC, де й реалізував ці ідеї в новій об'єктній мові Smalltalk (що, до речі, спочатку мала назву Biological System). Саме тоді він запропонував знаменитий термін «об'єктно-орієнтоване програмування» (ООП) [499].

В процесі роботи над Smalltalk А. Кей створив нову концепцію розробки програмного забезпечення – багатовимірне середовище сумісного функціонування об'єктів з асинхронним обміном повідомленнями. В результаті з'явилась можливість підтримки такого середовища за допомогою не одного, а багатьох комп'ютерів, об'єднаних у мережу. Працюючи над апаратною реалізацією ООП-системи (проект FLEX (рис. 2.3*а*) – повноцінний персональний комп'ютер, що базувався на об'єктах), А. Кей вивчав піонерські роботи Сеймура Пейперта та його колег з MIT з навчання дітей програмування мовою Лого. Розробники Лого досліджували дитячі уявлення про графіку та символи, запропонувавши «черепаху», що «малювала» на планшеті (екрані).

А. Кей бачив роль персонального комп'ютера як особистісне динамічне середовище (метамедіа), в якому об'єднувалися всі інші середовища: текст, графіка, анімація і навіть те, що ще не винайдено.

Подальшим розвитком FLEX став проект Dynabook (рис. 2.3*б*) – компактний комп'ютер, легко керований, оснащений клавіатурою і пером, безпровідною мережею тощо (в сучасних термінах Dynabook можна назвати планшетним портативним комп'ютером). В своїй статті 1972 р. А. Кей визначив ціль проекту як «персональний комп'ютер для дітей будь-якого віку». Smalltalk увібрав у себе багато з даного проекту – в ньому вперше були використані вікна, меню,

іконки та маніпулятор «миша» [499]. В Smalltalk містяться витоки Microsoft Windows, X Window та MacOS. Інакше кажучи, сучасні інтерфейси користувача еволюціонували паралельно з ООП, а їх формування відбувалося під впливом ідей Л.С. Виготського, Дж. Брунера та С. Пейперта.



а) *б)*

Рис. 2.3. *а* – концепція FLEX, *б* – прототип Dynabook

Сьогодні А. Кей – активний учасник проекту OLPC (One Laptop Per Child – «Кожній дитині – по ноутбуку»). Незважаючи на високу технологічну досконалість ідей проекту Dynabook – «батька» сучасних мобільних пристроїв, головним в ньому є все ж таки ідея «комп'ютера для конструктивістського навчання у динамічному світі», основою якого є положення соціального конструктивізму та конструкціонізму: особистісна зорієнтованість, висока інтерактивність, навчання через гру, спільне навчання, динамічне моделювання, навчання завжди та всюди.

Еволюцію концепції Dynabook показано у табл. 2.1.

Таблиця 2.1

Еволюція технології та навчальних концепцій

Роки	*Апаратура*	*ПЗ*	*Мережні засоби*	*Навчальна концепція*
1970-ті	Dynabook Alto	Smalltalk	Arpanet Ethernet	Навчання через відкриття
1980-	Xerox Star	C++	TCP/IP	Ситуативне навчання

Роки	*Апаратура*	*ПЗ*	*Мережні засоби*	*Навчальна концепція*
ті	Apple Lisa Apple Macintosh		Аналоговий стільниковий радіозв'язок	Конструктивістське навчання Навчання у співробітництві
1990-ті	ПК з Windows Ноутбуки КПК	Java	World Wide Web Цифровий стільниковий радіозв'язок Безпровідні локальні мережі	Проблемно-орієнтоване навчання Навчання протягом всього життя Соціально-конструктивістське навчання
2000-ні	Безпровідні КПК	CORBA	Bluetooth	Неформальне навчання Контекстуальне навчання

2.2.2 Піонерська освітня програма фірми Palm. У 90-х рр. минулого століття в ряді університетів Європи та Азії були розроблені системи мобільного навчання студентів. Значну роль в цьому відіграла піонер у створенні КПК – корпорація Palm, яка в рамках проекту PEP – Palm Education Pioneers (1999–2002 рр.) виділяла гранти на створення систем мобільного навчання під управлінням PalmOS.

У заключному звіті з проекту [527] були:

1) обґрунтовані нові типи навчальної активності, що виникають в процесі застосування КПК;

2) визначена роль КПК в тестовому контролі знань учнів;

3) сформульовані основні переваги персонального пристрою для навчання – підтримка самостійного навчання, підвищення відповідальності за результати навчання, посилення міжпредметних зв'язків;

4) визначені нові форми спільної роботи, в т.ч. – в сфері моделювання [571];

5) показані шляхи інтеграції мобільних та традиційних навчальних технологій.

Незважаючи на штучну прив'язку до використовуваної ОС, в рамках проекту PEP були сформульовані практично всі технологічні та педагогічні вимоги до застосування КПК в навчальному процесі [519, 501].

2.2.3. Виникнення мобільного навчання. Перші публікації, присвячені мобільному навчанню, з'являються у 2000 році. У квітневому випуску журналу «Computers and Education» М. Шарплес [553] охарактеризував нові можливості застосування мобільних технологій, що могло б підвищити ефективність андрагогічних навчальних програм та безперервної освіти. Багато ідей цієї статті М. Шарплеса розвиваються і є актуальними й сьогодні.

У 2001 р. Європейської комісія започаткувала проект MOBIlearn під керівництвом М. Шарплеса та Дж. Рікерка, які сформулювали основну ідею проекту – «що навчальне, те – мобільне» [546] – та визначили умови ефективності мобільного навчання:

1) конструктивність: навчання є конструктивним процесом пошуку розв'язку задач, що веде до утворення нового досвіду;

2) інтеріоризація результатів навчання;

3) діалектичність процесу контролю та відображення результатів навчальної діяльності у свідомості її суб'єкта.

В 2001 р. Д. Абернаті в статті [449] розглянула ділові застосування мобільного навчання, відзначивши, що впровадження засобів мобільного навчання не витісняє ПК у всіх його застосуваннях, а має доповнити корпоративне навчання новими інструментами. Можливості бізнес-застосування використання цієї технології полягають у збільшенні числа працівників і клієнтів, що взаємодіють в процесі навчання, та оперативному зворотному зв'язку для корпоративних відділів. Незважаючи на те, що на момент написання статті вже існував Wi-Fi, він не був настільки поширений, як сьогодні. Д. Абернаті точно зазначила, що нерозвиненість технології Wi-Fi може стати серйозним каменем спотикання для розвитку мобільного навчання.

Концепція мобільного навчання, запропонована Д. Кіганом у 2001 р., дістала розвиток у роботах Ф. Манг'яваччі, Р. Мейсона, Л. Родіна, М. Рончетті,

А. Трифонової та Д. Хойла (2002–2003 рр.). В 2002 р. в Канаді створено Консорціум мобільного навчання (The m-Learning Consortium), а в Австралії – державний стандарт на мобільне навчання [514].

Аналіз публікацій 2002–2006 рр. показує поширення засобів мобільного навчання у різних навчальних закладах. Так, у 2003 р. П. Сеппала і Х. Аламакі досліджували професійну підготовку та навчання фінських вчителів з використанням мобільних технологій в класі [544]. Зокрема, вони зазначають, що, з огляду на той факт, що у 2002 році 98% студентів Фінляндії мали мобільні телефони, впровадження мобільного навчання є наступним важливим кроком на шляху до цифрової революції в навчанні. Дослідники використовували службу коротких повідомлень для надсилання текстових фрагментів і цифрових фотографій, що збиралися в централізованому банку пам'яті, з якого кожен користувач може взяти цей матеріал в будь-який час для перегляду та вивчення. Викладачі отримують можливість вести записи в будь-який час і, в якості додаткової можливості, працювати над матеріалами під час своїх щоденних поїздок. П. Сеппала і Х. Аламакі прийшли до висновку, що мобільне навчання має багато переваг, і що ця технологія займе чільне місце в моделі навчання майбутнього.

У 2003 р. Дж. Еттевелл розглянув перспективи застосування мобільного навчання молодими викладачами та окремими категоріями роботодавців [457]. В деяких дослідженнях стверджується, що обмін SMS може бути шкідливим для граматичного розвитку студента (наприклад, щодо словникового запасу і написання) через те, що текст повідомлення, як правило, компактний, а іноді навіть пронизаний скороченнями або навмисно неправильною орфографією для прискорення набору. Дж. Еттевелл погоджується з тим, що ці питання є проблемними, і потребують подальшого дослідження, проте відкидає думку про те, що в аудиторії, де студенти зібралися разом для обміну повідомленнями та контентом за допомогою мобільних пристроїв, може виникнути ефект ізоляціонізму.

У 2004 р. Н. Вітсед розглянув появу мобільного навчання і мобільних обчислень [477] в галузі медицини. Так, за допомогою КПК можна отримати доступ до карток пацієнтів з будь-якої точки в лікарні; мобільні пристрої також надають можливість інтернам та медичним працівникам робити нотатки і записувати звук, який може бути вивчений і розглянутий пізніше. Ступінь вільності, яка забезпечується можливістю одержати доступ до необхідних матеріалів, документів «у будь-який час, в будь-якому місці», стає величезною перевагою, якщо взяти до уваги, скільки пацієнтів повинен оглянути лікар під час типового обходу. Н. Вітсед стверджує, що 28% американських лікарів вже використовують мобільні технології як частину їх повсякденного життя і що через переваги технології ця частка буде зростати.

В дисертації Фенг-Хуан Ю Янга (2004 р.) запропонована архітектура розподіленої системи мобільного навчання на основі грід-моделі [526], реалізація якої сприятиме створенню мобільних LMS, придатних для обслуговування великої кількості користувачів.

У 2004 р. корпорацією Intel було виконано пілотний проект «Навчання завжди та всюди», метою якого є надання кожному з учнів персонального доступу до мобільних комп'ютерних пристроїв та забезпечення безпровідного зв'язку у школах графства Ессекс. Міжнародним розвитком цього проекту є Intel World Ahead Education, розпочатий в Україні у 2008 р. під назвою «Мобільні технології – школам» (середовище електронного навчання «1 учень: 1 комп'ютер» [338]). Серед вітчизняних дослідників мобільного навчання слід відзначити М.А. Григор'єву, яка у 2004 р. запропонувала програму навчального курсу «Застосування мобільних освітніх систем» для студентів педагогічних ВНЗ [62], та І.Є. Мазурка, який у 2005 р. дослідив можливості застосування мобільних пристроїв у школі [180].

В період 2004–2005 рр. дослідники також вивчали, як широко поширені мобільні технології сьогодні і які тенденції їхнього розвитку. Так, у 2005 р. в США, за матеріалами Е. Вагнер та П. Вільсона [566], мобільних телефонів вже було більше, ніж стаціонарних, і інші бездротові пристрої набирають популяр-

ності за допомогою Wi-Fi-мереж. Автори визначають важливу відмінність між мобільним та електронним навчанням: вони стверджують, що нові пристрої і нові засоби доставляння навчальних матеріалів надають викладачам набагато більше варіантів для роботи із сучасними студентами, що є особливо важливим в умовах переходу від моделі «командування та контролю», типової для традиційних освітніх структур, до справжнього співробітництва у навчанні. Н. Рашбі [540] дослідив М-навчання з точки зору робітників. Він порівняв переваги свободи місця розташування з традиційними моделями електронного навчання багатьох компаній. Його робота показує, що мобільне навчання краще традиційних форм, зокрема, стосовно можливості працівників відстежувати та виявляти нові відомості з урахуванням кращих переваг їхнього стилю навчання. Стримуючими факторами цього виду освіти були обмежені пам'ять і швидкодія бездротових пристроїв недалекого минулого. Сьогодні завдяки таким корисним доповненням мобільних пристроїв, як GPS та покращені відео/аудіо-засоби, ще більше підвищуються дидактичні можливості застосування цих пристроїв. Цифрова гнучкість та придатність до сумісного використання цих нових технологій найкраще описуються поняттям медіаконвергенції, введеним Г. Дженкінсом, де весь діапазон нових технологій, використання яких надає можливість користувачам архівувати, коментувати, застосовувати медіаконтент, і в процесі використання цих технологій змінилися способи взаємодії із користувачами основних інститутів державного управління, освіти і комерції [503]. Н. Рашбі вважає, що мобільне навчання, швидше за все знайде застосування спочатку у бізнес-секторі. Він розглядає ризики та вигоди й інших організацій, таких як середні школи та університети.

П. Торнтон та К. Хаузер [560] у тому ж 2005 р. дослідили стан мобільного навчання в японських університетах. В Японії на базі Web підтримуються мобільні телефони, кишенькові комп'ютери та інші портативні медіапристрої, які надзвичайно поширені, а населення добре розбирається в тому, як їх використовувати. У цьому дослідженні оцінюються результати вивчення в аудиторії матеріалу за допомогою мобільних телефонів (засобами електронної пошти та з

використанням технології WAP). Важливо відзначити, що в Японії, тарифні плани на мобільний зв'язок набагато дешевші, ніж у Сполучених Штатах Америки, що надає можливість більшому числу студентів взяти участь у проектах мобільного навчання. Результати їхніх досліджень були особливо показовими. Дослідники відзначили покращення результатів тестування від 35% до 75% при використанні мобільного тестування у порівнянні з паперовим. Студентська реакція на нові можливості навчання була позитивною, хоча вона більше стосувалась КПК та смартфонів, ніж простих мобільних телефонів. Через широке поширення мобільних телефонів у Японії поширення навчальних матеріалів через них є порівняно легким процесом.

З появою у 2006 р. в США більш технологічно досконалих мобільних телефонів мобільне навчання стало ще більш привабливим для підприємств та навчальних закладів для покращення своїх навчальних середовищ. Е. Вагнер у [567] відзначає, що в той час як мобільні пристрої все ще залишаються найбільш поширеним інструментом для багатьох офісних працівників та студентів, технологічний ландшафт змінюється і стає все більш «прихильним» до мобільного навчання за рахунок розширення бездротових мереж та обладнання і зниження цін на доступ до безпровідної мережі. Е. Вагнер підкреслює, що, в той час, як пристрої мобільного навчання можуть бути надзвичайно корисними самі по собі, **саме навчальний матеріал має бути в центрі уваги педагогів, і що покращення контенту є одним з найкращих способів забезпечення ефективності мобільного навчання** для всіх його учасників.

С. Гомес у 2007 р. дослідив, як змінюються мобільні уроки та лекції в процесі розвитку мобільних пристроїв, запропонувавши застосування мобільного подкастингу [498].

Подкастинг (з англ. podcasting, від iPod та broadcasting [1] – повсюдне, широкоформатне мовлення) – процес створення і поширення звукових або відео-передач (подкастів) у Internet (зазвичай в форматі MP3, AAC або Ogg / Vorbis для звукових і Flash Video та інших для відео-передач). Як правило, подкасти мають певну тематику і періодичність видання, однак бувають і винятки.

Цільова аудиторія подкастингу – користувачі персональних або портативних комп'ютерів, а також власники портативних програвачів. Для зручного прослуховування подкастів створено багато програмних продуктів, таких як iTunes та AmaroK, що відслідковують оновлення подкаст-стрічок та автоматично завантажують новий матеріал.

Подкаст-термінал – це Web-сайт, що підтримує хостинг медіа-файлів та певною мірою автоматизує розміщення записів на сайті та підписку на оновлення. *Подкастом* називається або окремий файл, або регулярно оновлювана серія таких файлів, що публікуються за однією адресою в мережі. Поняттю подкастинга відповідає поняття аудіоблогу: під блогом зазвичай розуміють послідовність записів у вигляді звичайних Web-сторінок, а подкаст завжди забезпечує автоматичну перевірку оновлень за допомогою формату RSS.

Увага до мобільного навчання постійно зростає, що проявляється, в першу чергу, у зростанні кількості та частоти присвячених йому конференцій та семінарів. Так, в серії конференцій MLEARN перша, MLEARN 2002, відбулася в Бірмінгемі (Великобританія), MLEARN 2003 – в Лондоні (Великобританія), де зібрала більш ніж 200 делегатів з 13 країн, MLEARN 2004 – в Римі (Італія), MLEARN 2005 – в Кейптауні (ПАР), MLEARN 2006 – в Бенфі (США), MLEARN 2007 – в Мельбурні (Австралія), де зібрала 260 делегатів з 21 країни, MLEARN 2008 – у Волверхемптоні (Великобританія), MLEARN 2009 – в Орландо (США). Інша серія міжнародних конференцій – The International Workshop on Mobile and Wireless Technologies in Education (WMTE) – також регулярно проводиться з 2002 р. в Швеції, Китаї, Японії, Греції, США.

2.3. Засоби мобільного навчання

2.3.1. Апаратне забезпечення мобільного навчання. Впровадження мобільного навчання неможливе без відповідних мобільних пристроїв. М. Шарплес так сформулював вимоги до «ідеального мобільного пристрою для навчання» [553]:

– *надпортативний*, щоб буди доступним у будь-якому місці, де користувач потребує навчання;

– *індивідуальний*, адаптований до здібностей, знань та стилю навчання користувача, розроблений для підтримки особистісно-орієнтованого навчання, а не загальної роботи або розваг;

– *ненав'язливий*, такий, щоб студент міг захопитися процесом навчання;

– *доступний всюди* для спілкування з викладачами, експертами та колегами;

– *адаптовний* до контексту навчання та розвитку навичок і набуття знань студентами;

– *стабільний*, щоб за його допомогою можна було управляти навчанням протягом всього часу навчання, причому власні накопичення ресурсів і знань користувача мають бути доступні незалежно від змін в технології;

– *корисний*, придатний для потреб спілкування, роботи та навчання;

– *інтуїтивний*, для використання людьми без будь-якого попереднього досвіду роботи.

Існуючі мобільні пристрої суттєво відрізняються за своєю функціональністю, розмірами та ціною як між собою, так і в порівнянні зі звичайними технічними пристроями електронного навчання (стандартний ПК та периферійні пристрої). Основними рисами, що їх об'єднують, є мобільність та придатність для бездротового з'єднання. Основні види мобільних пристроїв, які використовуються в процесі навчання:

1) *переносні комп'ютери типу «ноутбук»*. З одного боку, вони мають функціональність, співрозмірну з функціональністю настільних ПК, в тому числі розширений діапазон зовнішніх пристроїв зберігання даних (CD-RW, DVD-RW та ін.), значний обсяг основної пам'яті, мультимедійні функції, великий екран. З іншого боку, у них невеликі розміри і забезпечується підтримка бездротового зв'язку. Сегмент ринку мобільних комп'ютерів в даний час є найбільш динамічним сектором ринку ПК;

2) *планшетні ПК (Tablet PC)* також мають повний спектр засобів персональних комп'ютерів. Деякі з них не мають клавіатури, проте мають сенсорні екрани і програмне забезпечення для розпізнавання рукописного тексту. Це відносно дорогі пристрої, що займають порівняно невелику частину ринку, проте їх популярність зростатиме з подальшим проникненням бездротових технологій;

3) *кишенькові комп'ютери (КПК, Pocket PC, PDA – Personal Digital Assistant, «надолонник»)* – збірна назва класу портативних електронних обчислювальних пристроїв, спочатку запропонованих до використання в якості електронних органайзерів. У англійській мові словосполучення «кишеньковий ПК» (Pocket PC) не є позначенням всього класу пристроїв, а є торговою маркою фірми Майкрософт, тобто, відноситься лише до одного з різновидів КПК. Англійське словосполучення Palm PC (надолонний комп'ютер) також асоціюється з конкретною торговою маркою. Для позначення всього класу пристроїв в англійській мові використовується словосполучення Personal Digital Assistant, PDA, що українською можна перекласти як «особистий цифровий секретар», проте в Україні цей термін не прижився.

КПК складається з процесора, пам'яті, звукової і відеосистеми, екрану, слотів розширення (за їх допомогою можна додати обсяг пам'яті або нову функціональність) та клавіатури. Кишенькові ПК мають невеликі розміри і значну потужність процесора. В нових моделях підтримується 65536 кольорів, розпізнавання рукописного тексту і мультимедійні функції. До КПК, оснащеного хост-контролером USB, можна безпосередньо під'єднувати різні USB-пристрої, зокрема клавіатуру, мишу, тверді диски і флеш-накопичувачі.

Для КПК характерні такі мобільні якості, як низька ціна, ефективність, зручність і компактність. Замість клавіатури в цих «особистих помічниках» використовуються, як і в планшетних ПК, стило та сенсорний екран, але на відміну від них вони значно менші і легші, а акумулятори працюють довше (до 10-12 годин). Основною перевагою КПК у порівнянні з ноутбуками є сенсорний екран, що усуває необхідність у застосуванні миші та інших пристроїв введення, а

також той факт, що їх дуже зручно застосовувати під час руху. Все це робить КПК придатними для використання в мобільному навчанні.

Деякі компанії пропонують кишенькові комп'ютери з можливостями їх використання в якості телефону (комунікатори).

4) *мобільні телефони*. Представники цього класу мобільних пристроїв можуть бути використані для голосового зв'язку, передавання і приймання текстових повідомлень. Найпростіші пристрої мають мало пам'яті, обмежену функціональність та низьку швидкість передавання даних. Мобільні телефони більш високого класу можуть бути використані для доступу до Інтернет через технології WAP (Wireless Application Protocol) та GPRS (General Packed Radio Service). Також вони можуть бути використані для передавання і приймання мультимедійних повідомлень (MMS);

5) *смартфони (смарт-телефони)* – клас гібридних пристроїв, що поєднують функції мобільних телефонів і КПК. Типовий смартфон не має повнорозмірної клавіатури, але може розпізнавати рукописний текст. Деякі нові моделі (особливо на базі Google Android) оснащуються висувною клавіатурою (рис. 2.4). «Розумні» телефони останнім часом набули настільки великого поширення, що стали поступово витісняти КПК та комунікатори. Ці пристрої мають практично ідентичні із звичайними КПК операційні системи з незначними відмінностями – додатковим програмним забезпеченням для роботи з мобільним зв'язком. У передових пристроях цього класу наявні вбудовані жорсткі диски, що робить їх більш придатними для зберігання великих обсягів даних та використання професійних прикладних програм. Вони важчі і споживають більше електроенергії, ніж традиційні мобільні телефони, однак, в силу їхньої очевидної переваги та вигоди ринок цих пристроїв розвивається дуже динамічно. Оцінки показують, що в 2009 році такі пристрої складатимуть майже чверть ринку мобільних телефонів і кишенькових комп'ютерів. Існує тенденція до переорієнтації користувачів на багатофункціональні пристрої для голосового зв'язку і передавання даних (що стосуються, крім мобільних телефонів, ще й смартфонів і навіть субноутбуків із засобами використання телефонії);

– в якості *додаткових мобільних пристроїв*, які можуть знайти застосування в мобільному навчанні, можна виділити мобільні принтери, Web-камери, картрідери для CompactFlash, Secure Digital, Memory Stick, Smart Media та інших типів карт, за допомогою яких можна переносити дані між різними типами пристроїв тощо.

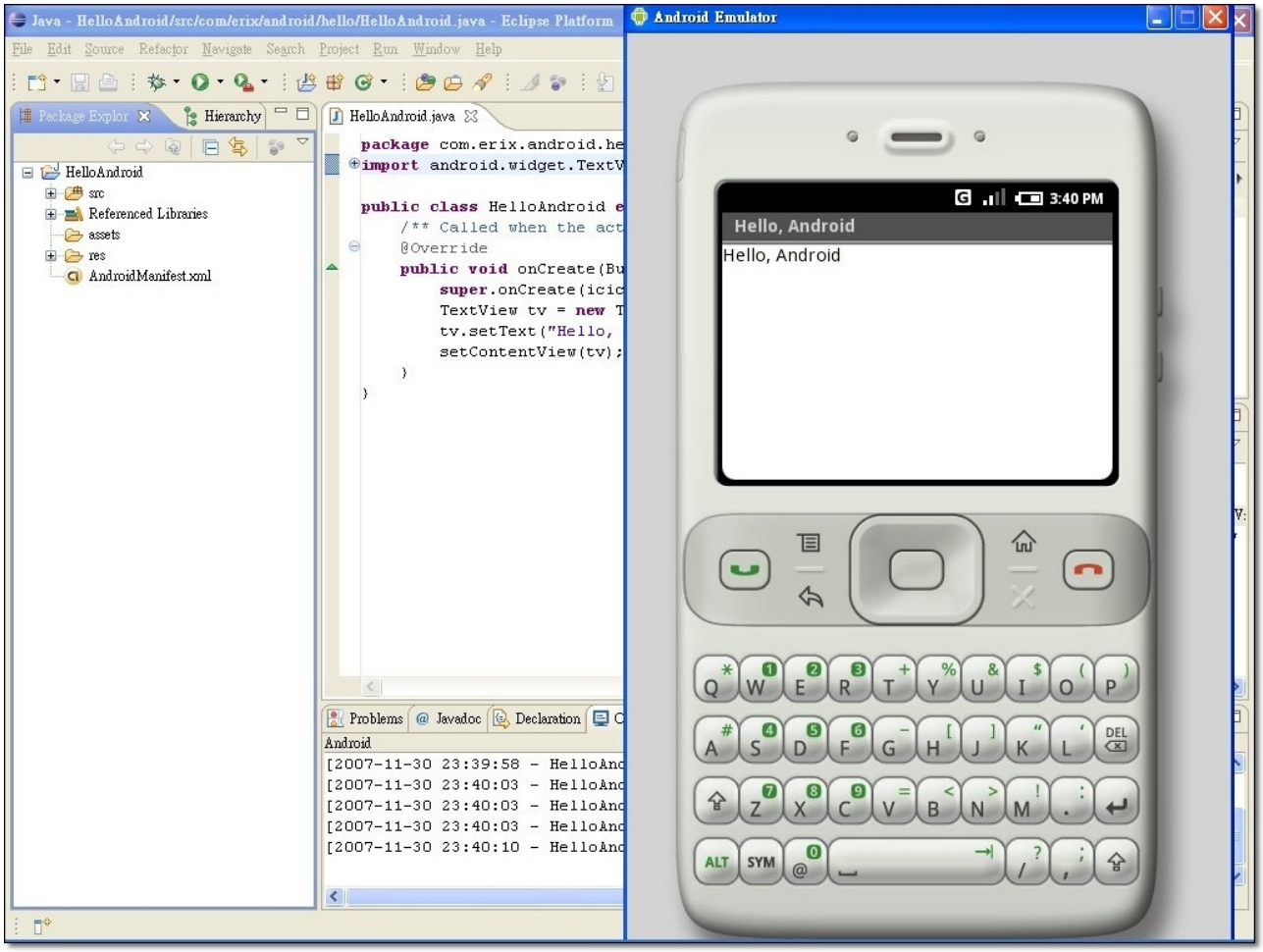


Рис. 2.4. Середовище розробки та емулятор Google Android

Наведена класифікація мобільних пристроїв є досить умовною, оскільки виробники постійно змішують їх відмінні риси, створюючи нові пристрої. Прикладом цього може бути поява пристроїв для введення рукописного тексту з повноцінною операційною системою Windows XP, розміри яких знаходяться у межах розмірів КПК. Іноді виробники додають функції мобільних телефонів до КПК чи, навпаки, вбудовують Web-браузер та функції КПК в мобільний телефон. У результаті пристрої істотно відрізняються, але всі вони можуть бути віднесені до смартфонів.

Порівняння типових параметрів основних мобільних пристроїв наведено у табл. 2.2.

Таблиця 2.2

Порівняння параметрів основних класів мобільних пристроїв

Пристрій *Параметри*	**Ноутбук**	**Планшетний ПК**	**Кишеньковий ПК**	**Мобільний телефон**	**Смартфон**
Вага	3 кг	1,5 кг	0,150 кг	0,100 кг	0,200 кг
Роздільна здатність екрану	Від 1024x768 пікселів	Від 1024x768 пікселів	240x320 піксе-лів	120x160 пі-кселів	200x300 пікселів
Пам'ять	2 Гб	2 Гб	512 Мб	4 Мб	32 Мб
Тривалість роботи від акумулятора	3 год.	4 год.	8 год.	18 год.	10 год.
Комунікаційні технології	IrDA, Wi-Fi, Bluetooth	IrDA, Wi-Fi, Bluetooth	IrDA, Wi-Fi, Bluetooth	WAP, GPRS, Bluetooth	GPRS, IrDA, Bluetooth

2.3.2. Програмно-комунікаційні засоби мобільного навчання. Сьогодні ще не існує єдиного стандарту мобільних програмних засобів, що стримує широке поширення мобільних технологій: практично кожен виробник має свою власну операційну систему і мультимедійні додатки. Так, операційні системи та ПЗ ноутбуків та планшетних комп'ютерів не відрізняються від відповідних засобів ПК. КПК та смартфони працюють переважно під управлінням ОС Windows Mobile, що містить таке стандартне ПЗ, як календар, контакти, голосовий запис, текстовий редактор, електронні таблиці, Pocket Internet Explorer, WindowsMedia Player 9, клієнти VPN, обміну повідомленнями, термінали тощо. Поширеними є платформи Symbian та Black Berry, перспективною є Google Android. У додатку А описані стандарти технологій мобільного навчання, рекомендовані Європейською комісією.

Дж. Беггелей у [459] зазначає, що поява *портативних програм* суттєво спрощує організацію мобільного навчання. Використання портативних (мобільних) програм надає можливість користувачу легко змінювати робоче місце без необхідності встановлювати потрібне ПЗ на кожному новому робочому комп'ютері. Електронна пошта, налаштування браузера, передавання файлів та інші функції можуть бути доступні без необхідності налаштування нового комп'ютера для них. Коли користувачі подорожують та є залежними від незнайомих комп'ютерів та Інтернет-з'єднань, подібні переналаштування можуть призводити до суттєвих витрат часу, якщо це взагалі можливо. Портативні програми, з іншого боку, подорожують разом з користувачем на зручних накопичувачах, таких як USB-диски. Перелік на сайті Portableapps.com охоплює кілька сотень портативних програм в більш ніж 50 категоріях, включаючи редагування документів, миттєвий обмін повідомленнями, робота в Інтернет, мережні послуги та захист програм і даних, а також мультимедійні презентації. Прикладами програм, що не потребують встановлення, є текстовий редактор Abiword, поштовий клієнт Thunderbird, портативний Web-браузер Mozilla Firefox, програма миттєвого обміну повідомленнями Gaim, FTP-клієнт FileZilla, медіапрогравач VLC Media Player, антивірус ClamWin.

Можна виділити два види програмного забезпечення для мобільних пристроїв: Web-орієнтоване та стандартне. Web-орієнтоване ПЗ легко інтегрується та налаштовується; для його роботи достатньо будь-якого стандартного Web-браузера. Стандартне програмне забезпечення є більш складним та вбудовується у мобільні пристрої.

При розробці програмного забезпечення для мобільних телефонів слід враховувати ряд обставин: мобільний телефон має малий екран, а його процесор і пам'ять мають обмежену функціональність. Це створює певні труднощі для програмістів, яким доводиться застосовувати спеціальні мобільні версії стандартних мов програмування, таких як C, C++, Java тощо. Так, компанія Sun розробила полегшену версію Java для пристроїв типу мобільних телефонів і КПК (Mobile Information Device Profile – MIDP). Macromedia розробила

Software Development Kit (SDK), що є середовищем для розробки мобільних мультимедійних додатків Flash Lite.

Розробка програмного забезпечення для кишенькових ПК не дуже складна, а для портативних комп'ютерів взагалі проста тому, що в них практично немає обмежень на обсяг пам'яті та швидкість процесора.

При реалізації мобільного навчання застосовуються різні технології та архітектури, побудовані на різних комунікаційних стандартах [497].

2.3.2.1. GSM. У 80-х роках минулого століття за участю багатьох європейських країн була створена група Conference Européenne des administration des Postes et des Télécommunications з метою розробки рекомендацій щодо покращення і вдосконалення мобільного зв'язку. Незабаром її було перейменовано в Group Speciale Mobile (GSM). Були розроблені технології для стільникового мобільного зв'язку на частотах від 900 МГц, а згодом (протягом 90-х років) – 1800 та 1900 МГц.

Пізніше абревіатура GSM набула значення Global System for Mobile communications (Глобальна система мобільних комунікацій).

GSM працює з аналогово-цифровими та цифрово-аналоговими перетворювачами, за допомогою яких людський голос та звук перетворюються у цифрову форму і навпаки.

Слід відзначити протокол WAP (Wireless Application Protocol), який використовується у поєднанні з технологією GSM. Він був створений для використання Інтернет у мобільних телефонах. Протоколом визначається набір технічних специфікацій для реалізації Інтернет-зв'язку та доставляння сучасних послуг телефонного зв'язку від та до мобільних телефонів, пейджерів, КПК та інших бездротових терміналів.

WAP є де-факто галузевим стандартом, що використовується багатьма постачальниками контенту. У WAP-пристроях може використовуватися мова WML (Wireless Markup Language), який є діалектом XML, оптимізованим для невеликих екранів.

2.3.2.2. GPRS. Технологія General Packet Radio System (GPRS) була розроблена і випробувана в період між 1999 і 2001 рр., а з 2002 р. впроваджена. В GPRS використовується режим пакетної комутації даних в процесі передавання. Пакетна комутація є швидким способом передавання даних, оскільки при передаванні практично кожного пакету використовуються свої власні засоби для досягнення мети (в порівнянні з комутацією каналів, що використовується кожен раз, щоб передати всі пакети в пункт призначення). Цей механізм забезпечує велику швидкість передавання даних. Застосування GPRS надає можливість досягти більш високої швидкості передавання даних в порівнянні з GSM. Максимальне значення цієї швидкості (теоретичне) – 171 Кбіт/с. На практиці швидкість залежить від обладнання та завантаження мережі (кількості включених абонентів) і складає 30-40 Кбіт/с.

На основі GPRS працює популярний сервіс обміну мультимедійними повідомленнями MMS. Оскільки GPRS побудований на протоколі TCP/IP і є прозорим для нього, це прекрасний інструмент для доступу до Інтернету та прийнятна основа для реалізації мобільного навчання.

Одна з головних переваг GPRS полягає в тому, що забезпечується високошвидкісний бездротовий Інтернет та інші комунікаційні послуги.

2.3.2.3. 3G та UMTS. Мережі третього покоління 3G функціонують на частотах дециметрового діапазону (близько 2 ГГц), швидкість передавання даних становить понад 2 Мбіт/с. Використання таких мереж надає можливість організувати відеозв’язок, дивитись на мобільному телефоні фільми та телепрограми тощо. В світі існує два стандарти 3G: UMTS (чи W-CDMA) та CDMA-2000. UMTS більш розповсюджений в основному в Європі, CDMA2000 – в Азії та США. За даними Wireless Intelligence, на кінець 2006 року в світі нараховувалось 364 млн. абонентів 3G, з них 93,5 млн. використовували мережу UMTS та 271,1 млн. – CDMA2000.

Термін 3G використовується для опису сервісів мобільного зв’язку стандартів наступного покоління, через які забезпечується більш висока якість звуку, а також високошвидкісний Інтернет-звязок та мультимедійні сервіси. Мобі-

льні мережі третього покоління (3G) відрізняються від мереж другого покоління (2G), таких як, наприклад, цифровий стандарт мобільного зв'язку GSM, зв'язок перехідного покоління (2.5G), GRPS із значно більшою швидкістю передавання даних, а також більш широким набором і високою якістю послуг, що надаються.

З січня 2007 року в Україні працює мережа 3G PEOPLEnet (стандарт CDMA2000 800 мГц). 1 листопада 2007 державне підприємство Укртелеком запустило мережу 3G Utel (стандарт UMTS 2100 з надбудовою HSDPA (3,5G)). В UMTS використовуються два різні канали на приймання і на передавання. Якщо в базовій версії UMTS забезпечується пікова швидкість від 2-х мегабіт за секунду для статичних об'єктів в околі соти та 384 Кб/с для мобільних абонентів, то для пристроїв, що підтримують HSDPA, швидкості теоретично можуть досягати 14,4 Мбіт/с. На практиці ж реальні швидкості майже не перевищують 3 Мбіт/с, а в умовах високих, щільних забудов та завантаженості мережі – ще менше. Використання такого зв'язку дає можливість здійснювати відеодзвінки, широкополосний доступ в Інтернет, а також переглядати потокове відео в онлайні.

На жаль, звичайні мобільні телефони не можуть функціонувати в мережі UMTS, проте багато телекомунікаційних компаній розробляють мобільні термінали для роботи в мережі 3-го покоління. З огляду на характеристики GPRS слід зазначити, що UMTS забезпечуються ще більш сприятливі умови для розвитку і використання мобільного навчання.

2.3.2.4. Wi-Fi (Wireless Fidelity) названий за аналогією з акустичним терміном Hi-Fi (High Fidelity – найвища точність відтворення). Wi-Fi – бездротова радіо-технологія для бездротової локальної мережі (WLAN). Цей стандарт об'єднує кілька протоколів та ґрунтується на сімействі стандартів IEEE 802.11 (Institute of Electrical and Electronic Engineers – міжнародна організація, що здійснює розробку стандартів у сфері електронних технологій). Найбільш відомим та поширеним на сьогодні є протокол IEEE 802.11b, що визначає функціонування бездротових мереж.

Наявність Wi-Fi-зон (точок) надає користувачеві можливість під'єднатися до точки доступу (наприклад, до офісної, домашньої або публічної мережі), а також підтримувати з'єднання кількох комп'ютерів між собою.

Максимальна дальність передавання сигналу у такій мережі складає 100 метрів, однак на відкритій місцевості вона може досягати до 300-400 м.

Окрім 802.11b, ще існує бездротовий стандарт 802.11a, де використовується частота 5 Ггц та забезпечується максимальна швидкість 54 Мбіт/сек., а також 802.11g, в якому використовується частота 2,4 Ггц і також забезпечується швидкість 54 Мбіт/сек. Крім цього, ведеться розробка стандарту 802.11n, за допомогою якого у майбутньому можна буде забезпечити швидкості до 320 Мбіт/сек.

Ядром бездротової мережі Wi-Fi є так звана точка доступу (Access Point), що під'єднуються до якоїсь наземної мережевої інфраструктури (каналів Інтернет-провайдера), через яку забезпечується передавання радіосигналу. Зазвичай, точка доступу складається із приймача, передавача, інтерфейсу для під'єднання до дротової мережі та програмного забезпечення для опрацювання даних. Навколо точки доступу формується окіл радіусом 50-100 метрів (її називають хотспотом або зоною Wi-Fi), на якій можна користуватися бездротовою мережею.

Для того, щоб під'єднатися до точки доступу та відчути всі переваги бездротового зв'язку, користувачу ноутбуку або мобільного пристрою із Wi-Fi-адаптером необхідно просто потрапити у вказаний окіл точки доступу. Усі дії із визначення пристрою та налаштування мережі у більшості операційних систем комп'ютерів та мобільних пристроїв здійснюються автоматично. Якщо користувач одночасно потрапляє в кілька Wi-Fi зон, то під'єднання здійснюється до точки доступу, що забезпечує найсильніший сигнал.

Wi-Fi технології дуже зручно використовувати для мобільного навчання. Розгортання достатньої кількості точок бездротового доступу до Інтернет на відповідних місцях в університетському кампусі (бібліотеці, читальних залах, кав'ярні і т.д.) забезпечує доступ до ресурсів Інтернет, в тому числі навчально-

го змісту практично з будь-якого місця на території ВНЗ. Таким чином, це раціоналізує використання вільного часу студентів.

Мобільні пристрої, в яких використовується Wi-Fi, повинні бути обладнані відповідним радіомодемами (бездротовими картками). Деякі пристрої (ноутбуки, КПК) вже продаються із вбудованими пристроями для під'єднання через Wi-Fi. Прогнозується, що 90% всіх мобільних пристроїв найближчим часом будуть оснащені модулем для Wi-Fi-з'єднання.

Wi-Fi-з'єднання, на думку деяких авторів, є наступною революцією в комп'ютерному світі з моменту появи WWW-браузера. Незважаючи на те, що зараз покриття мереж GSM та GPRS значно вище, ніж Wi-Fi, ситуація може змінитися.

2.3.2.5. 4G та WiMAX. 4G – четверте покоління мобільного зв'язку, що характеризується високою швидкістю передавання даних та підвищеною якістю голосового зв'язку. Як і GPRS, заснований на протоколах пакетного передавання даних. Для пересилання використовується протокол IP v6. Для передавання даних використовуються частоти 40 та 60 ГГц. Для чіткого приймання та передавання планують застосовувати адаптивні антени, які будуть автоматично налаштовуватися на конкретну базову станцію.

До мобільного зв'язку четвертого покоління (4G) відносять стандарти широкосмугового передавання даних, на основі яких забезпечуються швидкості передавання даних до 100 Мбіт/с (реально забезпечувані швидкості – від кількох до кількох десятків Мбіт/с). Ці стандарти групи WiMAX, зокрема протокол 802.16e (його називають «мобільний WiMAX»), а також південнокорейський WiBRO можуть функціонувати в діапазонах від 2,3 до 7,2 ГГц і вище. На ринку доступні абонентські термінали у вигляді карт передавання даних для ноутбуків. Голосові з'єднання в мережах 4G здійснюються за допомогою клієнтських програм типу Skype. Існує думка, що операторам немає сенсу розвивати стандарт 3G, який «застарів, не народившись», а слід відразу переходити до мереж четвертого покоління.

2.3.2.6. Bluetooth є бездротовою технологією радіозв'язку на коротких відстанях. Частотний діапазон – від 2402 до 24809 МГц; ширина смуги 1 МГц дає 79 каналів, що уможливлює передавання сигналів між розташованими поряд телефонами, комп'ютерами та іншими мобільними пристроями, спрощує комунікацію та синхронізацію між ними.

В даний час розробки в галузі Bluetooth ведуться групою Bluetooth SIG, до якої входять Lucent, Microsoft та інші компанії, чия діяльність пов'язана з мережними технологіями. Основне призначення Bluetooth – забезпечення економного (з точки зору витрачання струму) і дешевого радіозв'язку між різноманітними типами електронних пристроїв, таких як мобільні телефони та аксесуари до них, портативні та настільні комп'ютери, принтери та інші. При цьому велике значення приділяється компактності електронних компонентів, що дає можливість застосовувати Bluetooth у малогабаритних пристроях розміром з наручний годинник.

В інтерфейсі Bluetooth передбачено можливість передавання як голосу (зі швидкістю 64 Кбіт/сек), так і даних. Для передавання даних можуть бути використані асиметричний (721 Кбіт/сек в одному напряму і 57,6 Кбіт/сек в іншому) та симетричний (432,6 Кбіт/сек в обох напрямах) методи. Працюючи на частоті 2,4 ГГц, прийомопередавач (Bluetooth-чіп) може встановлювати зв'язок у межах від 10 до 100 метрів. У стандарті Bluetooth передбачене шифрування даних, що передаються з використанням ключа ефективної довжини від 8 до 128 біт і можливістю вибору односторонньої або двосторонньої аутентифікації. Додатково до шифрування на рівні протоколу може бути використано шифрування на програмному рівні.

Назва Bluetooth походить від прізвища середньовічного короля Данії Гаральда I Синьозубого (норв. Harald Blåtann), який вмів посадити за стіл переговорів ворогуючі партії, домовляючись з кожною партією окремо, тому назва Bluetooth стала відповідним ім'ям для технології, на основі якої можна узгоджено використовувати різні пристрої.

2.3.2.7. IrDA – це технологія, в якій використовуються електромагнітні хвилі в інфрачервоному діапазоні (довжина хвилі від 850 до 900 нм). Передавання та приймання даних здійснюється на невеликі відстані – до 1–2 метрів, наприклад, для застосування в PAN (персональних мережах передавання даних). Швидкість передавання даних між пристроями з IrDA становить 115,2 Кбіт/с або 4 Мбіт/с. В смартфонах, КПК, принтерах і портативних комп'ютерах часто використовуються протоколи IrDA. Розробка стандартів для цієї технології здійснюється Infrared Data Association.

IrDA є прикладом простого протоколу обміну даними в обмеженому просторі (стандартом визначається межа в 100 см). Шляхом обмеження дальності досягається безпека від прослуховування. Завдяки цьому також зменшується вартість приладів, однак передавання даних мусить відбуватись за умов прямої видимості між портами.

З початку IrDA був розроблений HP – через це, навіть зараз, можна знайти позначення HPSIR (HP-Serial-Infrared) для стандарту IrDA 1.0. До протоколу IrDA належать також IrLAP, IrLMP, IrIAS, IrIAP, IrLPT, IrCOMM, IrOBEX, IrMC та IrLAN. Порівняно недавно IrDA було розроблено стандарт IrFM (Infrared Financial Messaging), також відомого як «Вкажи та заплати» (Point and Pay). В останній час IrDA поступово витісняється Bluetooth.

Наведений вище короткий огляд показує, що нинішні мобільні технології охоплюють широкий спектр послуг, що робить їх зручними для використання в мобільному навчанні.

2.3.3. Електронні книги як інноваційний засіб мобільного навчання. Найбільш суттєвим недоліком КПК при використанні його в якості технічного засобу навчання є малий час автономної роботи, зумовлений, насамперед, застосуванням сенсорної панелі та кольорового екрану. Для подолання цього недоліку можна запропонувати системи з енергозберігаючими рефлективними екранами на основі технології «електронного паперу» («електронних чорнил» – E-Ink) [393].

Пристрої, в яких використовуються папероподібні екрани, позиціонуються переважно як електронні книжки (пристрої для читання – E-Book). Роздільна здатність E-Ink-екранів – 600×800 та вище, що дає можливість високоточного відтворення зображень з високим ступенем деталізації, а їх розмір (6 дюймів та вище) робить процес перегляду більш комфортним, ніж на КПК. Екран, виготовлений за технологією E-Ink, має властивість бістабільності: на підтримку зображення енергія не витрачається, тому, відкривши книгу, ви побачите ту сторінку, на якій вона була закрита. На сьогодні E-Ink – найбільш «зорозберігаюча» технологія, тому що зображення на екрані подається у відбитому світлі, найбільш природному для очей.

Електронна книга є лише носієм даних, тому традиційно складається з двох складових – носій та вміст. Носієм є електронний пристрій, який може бути пристосованим (наприклад, телефон, чиєю основною функцією є дзвонити) чи спеціалізованим. Вміст іноді називають «контентом» – це будь-яка форма зберігання повідомлень чи навчальних матеріалів (які теж є наборами повідомлень), наприклад текст, відео, аудіо та інші електронні форми. Найчастіше в якості вмісту електронної книги застосовується текст з ілюстраціями, як і в традиційній книзі. Автори [555], аналізуючи можливості застосування електронних книжок в дистанційному навчанні, головну увагу приділяють засобам обміну контентом. Проте такий підхід невиправдано звужує можливості застосування електронних книжок у порівнянні з КПК.

Сучасні електронні книги (Sony PRS-505, CyBook, IREX Iliad, lBook eReader V3 та інші) за будовою є потенційно універсальними пристроями, що функціонують під управлінням ОС Linux. В процесі завантаження системи ініціалізується стандартна графічна підсистема X Window, під управлінням якої завантажується головна програма, що надає користувачеві абстракцію книжкової полиці. Вибір файлів (у форматах pdf, djvu, doc, rtf, html, chm, lit, fb2 та ін.) приводить до запуску асоційованих програм.

У листопаді 2007 р. Інтернет-магазин Amazon представив власну електронну книгу – Kindle, яка має вбудовану клавіатуру, засоби зв’язку та необхідне

мережне ПЗ. На жаль, висока ціна, жорстка прив'язка до контент-провайдерів США та відсутність офіційних постачань в Україну не дозволяють сьогодні застосувати Kindle у вітчизняній системі освіти, тому для подальшої роботи було обрано вітчизняну розробку (компанія МУК, http://lbook.com.ua) – електронну книгу lBook eReader V3 (рис. 2.5).

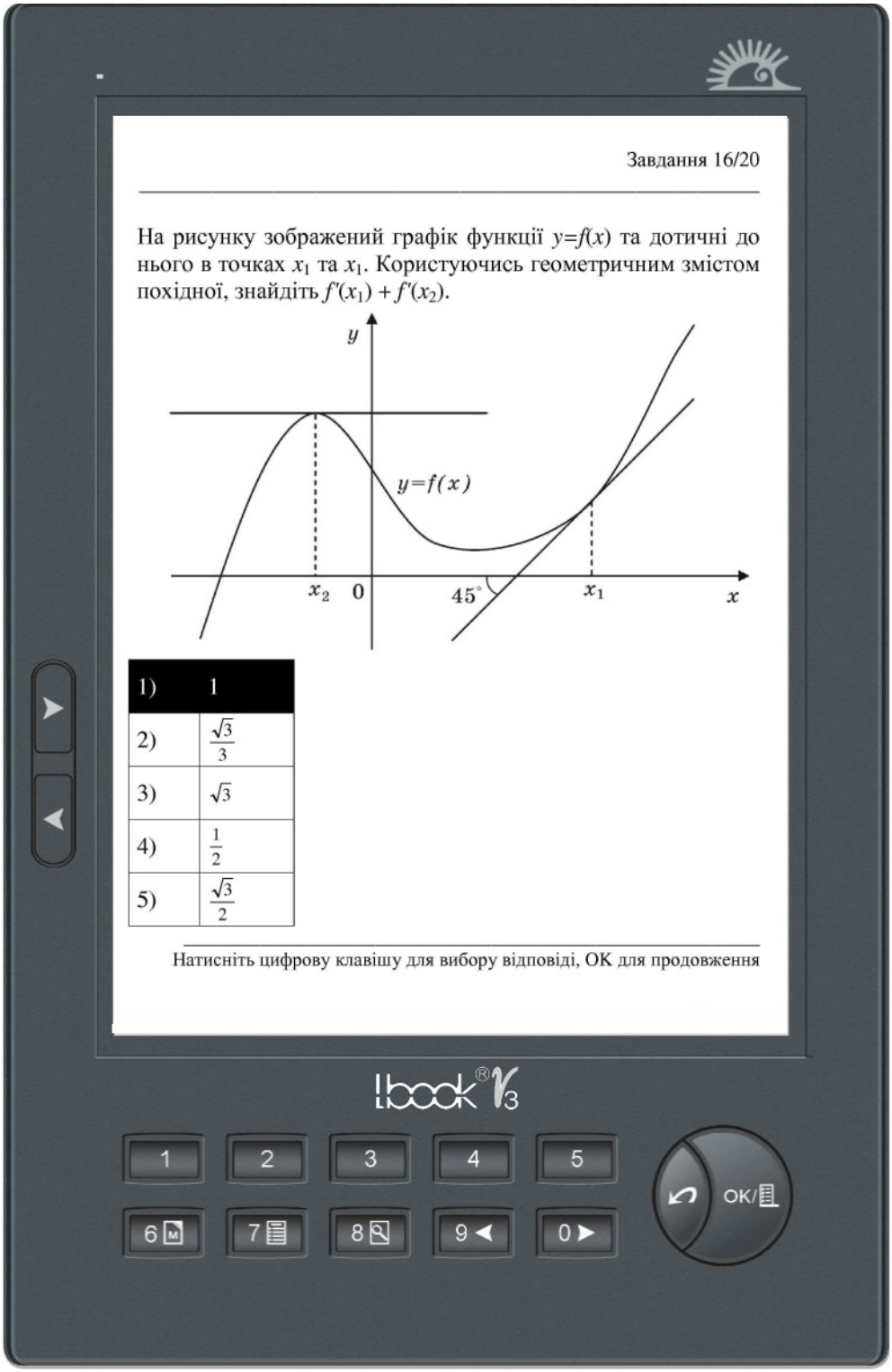


Рис. 2.5. Робота тестової системи на lBook eReader V3

Наявність відкритого пакету розробника для lBook eReader V3 спонукали нас до заміни стандартної книжкової полиці на універсальний файловий менеджер з можливістю запуску як стандартних програм для перегляду, так і завантажених користувачем. Тестування даного рішення виявило наступне:

– відсутність сенсорного екрану компенсується розробкою T9-подібного алгоритму введення тексту (за допомогою наявних 12 кнопок);

– відсутність вбудованих засобів для зв'язку компенсується встановленням карти розширення (за інтерфейсом SDIO);

– властива технології E-Ink низька реактивність екрану не дозволяє застосовувати анімацію зі швидкістю вище 4 кадри за секунду, проте для більшості навчальних демонстрацій цього цілком достатньо.

Це дозволяє розглядати lBook eReader V3 як нову програмно-апаратну платформу для дистанційного навчання. Для реалізації її потенціалу необхідні як заходи з портування програмного забезпечення (Web-браузера, мережних клієнтів, електронних таблиць, математичних пакетів тощо), так і розробка спеціалізованих SRS-клієнтів для Numina-подібних систем.

2.4. Умови застосування мобільного навчання

2.4.1. Переваги та недоліки мобільного навчання. У п. 2.1.3 було визначено місце мобільного навчання серед технологій автоматизованого навчання, що дозволяє визначити організаційно-технічні переваги мобільного навчання у порівнянні з електронним:

– *можливість навчання будь-де і будь-коли*;

– *портативність та мобільність*;

– *менші розміри та вага*;

– *нижча ціна мобільних пристроїв у порівнянні з ПК*: існує тенденція до постійного зниження цін при поліпшенні їх функціональності;

– *актуалізація навчання через «моду» на пристрої* залучає більше число студентів (особливо молоді) до мобільного навчання;

– *підвищена інтерактивність навчання*: основні операції виконуються в онлайн-режимі, тобто в режимі інтерактивної пізнавальної діяльності;

– *зручність* застосування послуг мобільного навчання в будь-який час і будь-якому місці. Занадто багато свого часу людина проводить в очікуванні, в процесі якого вона може урізноманітнити, поповнити та оновити свої знання і навички через виконання тестів, перегляд відеофільмів, і навіть під час гри;

– *розвинені засоби співробітництва*. Якісна освіта рідко отримується поодинці, і одним з найкращих способів успішного навчання є колективна робота, важлива для обміну ідеями. При роботі в онлайн-режимі можливо одночасно отримувати консультацію, нові ідеї та вести дискусії. Це створює умови для формування віртуальних груп із змінним кількісним та якісним складом;

– *безперервний доступ до навчальних матеріалів*. Мобільні пристрої можуть бути використані для підтримки реалізації конкретних заходів, як тоді, коли вони локалізовані, так і в русі. Наприклад, мобільні пристрої часто використовуються для технічного обслуговування і ремонту рухомих об'єктів (часто розташованих у важкодоступних місцях), причому оператор може отримувати довідки та консультації, доступ до професійної допомоги та навчальних матеріалів;

– *концентрований зміст*: навчальні об'єкти мобільного навчання більш тісно пов'язані один з одним на рівні мікронавчання;

– *сумісність*: наявність цілого ряду стандартів робить можливим використання різних форматів для подання навчального матеріалу в умовах застосування найрізноманітніших мобільних пристроїв.

До організаційно-технічних недоліків мобільного навчання можна віднести:

– *фрагментацію навчання*: навчання вимагає концентрації та роздумів, в той час як в процесі переміщення студенти знаходяться в ситуаціях, що можуть відволікати їх увагу;

– *відсутність у студентів добре розвинених навичок метапізнання* (здатності усвідомлювати та контролювати процес власного навчання) через віднос-

ну новизну мобільного режиму доставляння навчальних матеріалів та відповідних навчальних стратегій;

– *малий розмір екрану і труднощі з доступом до Інтернет*: мобільні телефони мають значно менші розміри екрану у порівнянні з традиційними ПК, а більшість Web-сайтів оптимізовано для екранів з високою роздільною здатністю;

– *висока вартість початкових вкладень у організацію мобільного навчання*: інвестиції у пристрої для кожного студента, організацію бездротової мережі, технічне обслуговування тощо;

– *проблеми забезпечення безпеки пристрою та даних у ньому*: через свої розміри і портативність пристрій легко втратити.

Частина зазначених недоліків мають технічну природу і зі зміною технології можуть зникнути. Подолання інших недоліків мобільного навчання вимагає зміни освітньої парадигми, що, в свою чергу, потребує зміни методів навчання і комунікацій між викладачем і студентом, а також серед самих студентів.

2.4.2. Мобільне навчання як інноваційна технологія. Мобільне навчання є новою освітньою парадигмою, на основі якої створюється нове навчальне середовище, де студенти можуть отримати доступ до навчальних матеріалів у будь-який час та в будь-якому місці, що робить сам процес навчання всепроникаючим та мотивує до безперервної освіти та навчання протягом всього життя.

Перехід від стаціонарного до мобільного навчання створює передумови для співробітництва, а також для неформальної взаємодії між студентами.

Мобільне навчання – це технологія навчання за допомогою мобільних пристроїв, комунікаційних технологій та інтелектуальних інтерфейсів користувача. Унікальними властивостями мобільного навчання є:

– придатність до одночасної взаємодії як з одним студентом, так і з групою (за відповідного покриття взаємодія є швидкою та надійною);

– можливість динамічного генерування освітнього контенту в залежності від місцезнаходження студентів, контексту та використовуваних мобільних пристроїв;

– можливість запису та зберігання окремих дискретних у часі дій студентів у будь-який час і в будь-якому місці;

– розмиття границь між соціумом та навчальним закладом завдяки можливості застосування мобільних пристроїв у навчанні, коли викладач ставиться в умови, за яких матеріалу, що раніше циркулював у межах аудиторії, може бути протиставлений матеріал ззовні, що функціонує без контролю з його боку.

До реалізації мобільного навчання існує два близькі підходи:

1. Мобільне навчання – це електронне навчання за допомогою мобільних пристроїв та безпровідних мереж. Після того, як домінуючим способом доступу до мережі Інтернет стануть бездротові мобільні пристрої, електронне навчання стане мобільним без будь-яких особливих змін у технології навчання.

Разом з тим застосування нових технологій (інформаційних, комунікаційних тощо) без зміни та адаптації до них змісту навчання та без урахування специфічних потреб цільових груп може призвести до компрометації ідеї, як це часто відбувалося в минулому.

2. Мобільне навчання є інноваційною педагогічною технологією [160], в якій сам навчальний процес є географічно та ситуаційно залежним, тобто контекстно пов’язаний з місцем та станом, в якому знаходиться студент.

За другого підходу враховується специфіка мобільних пристроїв, особливості цільової групи тих, хто навчається, та конкретизується практична необхідність. За такого підходу студент може спілкуватися безпосередньо з викладачем постійно за допомогою Інтернет – на відміну від традиційного навчання, де таке спілкування можливе лише у межах навчального закладу. Викладач відіграє роль консультуючого керівника, котрий спрямовує діяльність студента на отримання необхідних знань. Це надає можливість реалізувати проблемне навчання через обговорення дій, що допоможуть тому, хто навчається, оволодіти матеріалом, усвідомити необхідні результати та набути нові знання [391].

Впровадження мобільного навчання викликає зміни в усіх компонентах методичної системи навчання. У табл. 2.3 наводиться порівняння електронного та мобільного навчання за основними групами показників.

Таблиця 2.3

Порівняльний аналіз електронного та мобільного навчання

Електронне навчання	**Мобільне навчання**
Навчальний процес	
більша частина навчальних матеріалів – текстові та графічні	навчальні матеріали – текстові, графічні, голосові
Взаємодія між викладачем та студентом	
за допомогою електронної пошти з втратами часу на регулярну перевірку пошти	миттєве повідомлення про отримання електронної пошти
асинхронна пасивна комунікація	синхронна миттєва активна комунікація
	інтерактивність
	спонтанність
Комунікація між студентами	
безпосередня	безпосередня та опосередкована
через e-mail	через e-mail, SMS, MMS
в окремому приміщенні	миттєва, завжди
через точку доступу до Інтернет	без точки доступу до Інтернет
проблема організації позааудиторної роботи в групах	без географічних обмежень з використанням усіх засобів ІКТ
Зворотний зв'язок зі студентами	
опосередкований через електронну пошту, Web-сайти (форуми, чати тощо)	прямий через мобільні пристрої
асинхронний	синхронний та асинхронний
розподілений у часі	в реальному часі та у зручному режимі

Електронне навчання	Мобільне навчання
документально оформлений	частково задокументований
Оцінювання та контроль знань	
в аудиторії	в будь-якому місці
у визначений час	будь-коли
обмежений в часі	без обмежень в часі
стандартний тестовий	індивідуалізований (адаптований) тестовий
поганий зворотний зв'язок	насичений зворотний зв'язок
відкладений зворотний зв'язок	миттєвий зворотний зв'язок
тести фіксованої довжини	змінна довжина тесту/час на відповідь
тести та задачі переважно текстові	тести мультимедійні
Подання навчального матеріалу	
застосування однієї мови	автоматичне подання матеріалу різними мовами
аудиторне подання навчального матеріалу	індивідуальне подання матеріалу з розвиненими засобами комунікації
індивідуалізована, компонентно-орієнтована робота в групі	одночасна спільна робота в групі
отримання результатів екзаменів та контролю знань в твердій копії у визначений час	отримання результатів контролю знань в електронному вигляді у будь-який момент часу

В сучасному дистанційному навчанні панує асинхронний метод доставляння освітнього контенту, як правило, текстового матеріалу. Це буде змінюватися з впровадженням мобільного навчання. Об'єднання обчислювальних та комунікаційних засобів перетворює телефони і мобільні термінали на потужні мультимедійні пристрої. Наприклад, XML-подібна мова SMIL (Synchronized Multimedia Integration Language – Мова інтеграції синхронізованого мультимедіа) буде дуже корисна для поширення сучасного мультимедійного контенту. Форми керованого мультимедіа відкривають нові можливості для навчання, до-

сліджень та комунікації. Онлайнові мобільно-орієнтовані навчальні курси мають включати в себе більше мультимедійних тестів та завдань, адже загальновідомо, що при використанні різних каналів сприйняття обсяг сприйнятих та засвоєних людиною відомостей суттєво зростає.

Слід зазначити, що роль і значення стаціонарних комп'ютерів у навчанні зменшиться не так швидко – вони ще довго будуть використовуватися в якості засобу, використання якого надає можливість працювати протягом тривалого часу в автономному режимі. Роль стаціонарних ПК поступово будуть перебирати мобільні пристрої (з розширенням функцій і характеристик щодо подання і передавання повідомлень).

Концептуальні моделі традиційного та мобільного навчання в середній школі показані на рис. 2.6, 2.7 [391].

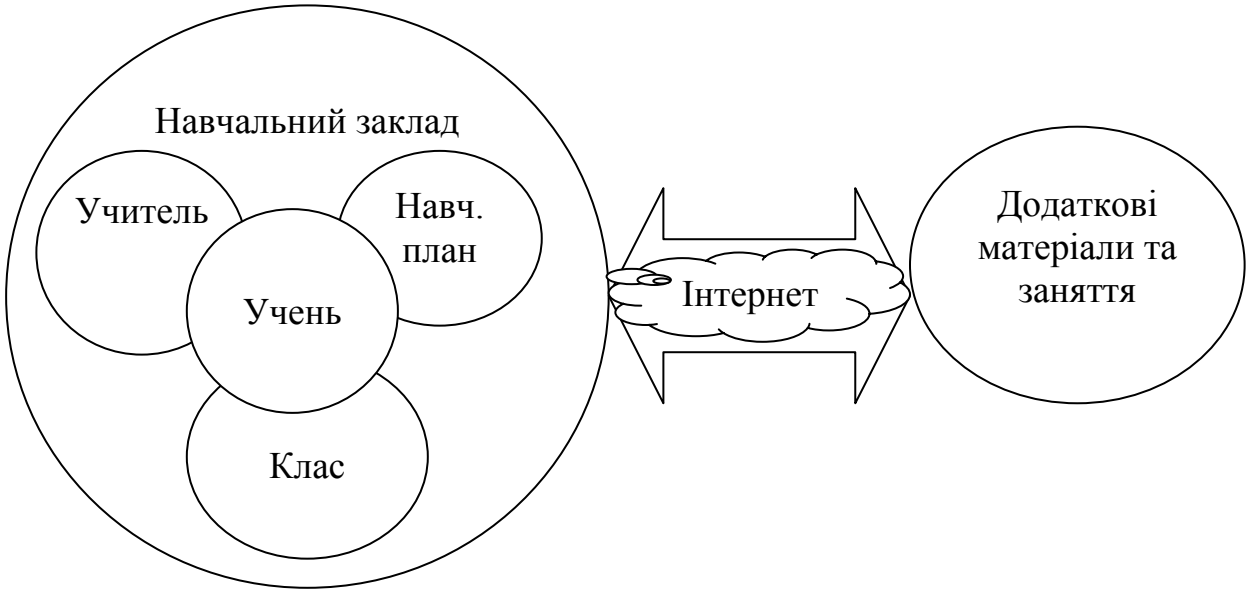


Рис. 2.6. Концептуальна модель традиційного навчання в середній школі

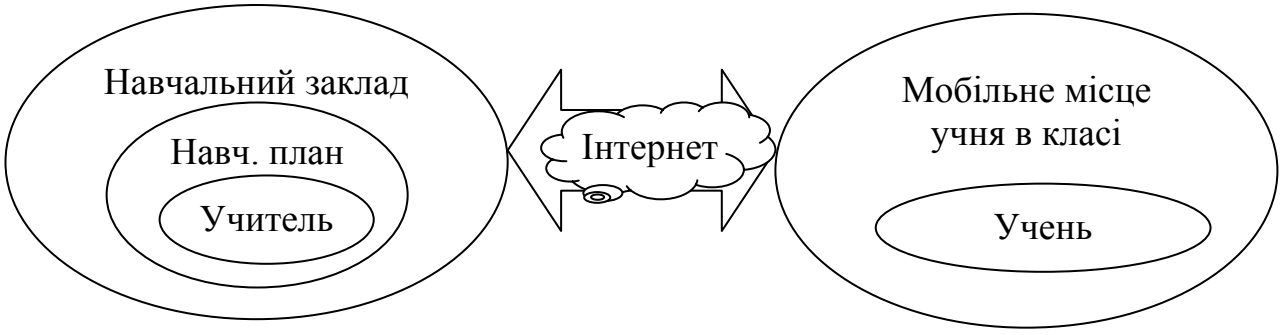


Рис. 2.7. Концептуальна модель мобільного навчання в середній школі

Для ефективної взаємодії у мобільному середовищі як викладачеві, так і студентові важливо усвідомлювати соціальну природу навчальних комунікацій, що є умовою якості навчання – інформаційно-комунікативні можливості тут є вирішальними в набутті знань, вмінь, навичок, досвіду.

Мобільне навчання відбувається не в класі, а у деякому іншому навчальному просторі. Проте концепція класу (навчальної групи) не зникає – класи перестають бути сталим утворенням і стають динамічними, формуючись на кожен предмет окремо. Час для навчання є питанням особистого вибору студентів і не обмежується університетом та розкладом занять. Студенти швидко усвідомлюють переваги динамічної інтерактивності мобільного навчання, ефективність та багатство комунікацій, якість керованого вчителем доступу до навчальних ресурсів.

Мобільне навчання не заважає соціалізації студентів, які є активними користувачами чатів, форумів і онлайнових співтовариств, побудованих на основі сервісів Web 2.0. Так, наприклад, MySpace, який описується як «місце для друзів», в якому надається онлайн-простір для особистих профілів, можливість знаходити інших людей зі спільними інтересами і брати участь у заходах, починаючи від онлайн-чатів до «живих» зустрічей, є одним з п’яти найбільш відвідуваних Web-сайтів у світі. Через засоби соціальних мереж, таких як MySpace, FaceBook, Bebo і Flickr або їхніх російськомовних аналогів «Вконтакте», «Одноклассники.ру», об’єднуючись у групи за інтересами, молоді люди в процесі обміну думками, навичками, текстами програм підвищують власну кваліфікацію в обраній предметній галузі. На основі соціальних мереж з використанням особистісних профілів і програмного забезпечення створюється мережне середовище соціальної солідарності, свободи слова і творчого спілкування.

Наказом МОН України №271 від 24.03.2009 р. «Про продовження Всеукраїнського експерименту щодо навчання вчителів ефективному використанню інформаційно-комунікаційних технологій у навчальному процесі та підвищення кваліфікації педагогічних працівників за програмою Intel® «Навчання для майбутнього» визначено, що методична та технічна допомога вчителям протягом

всього експерименту буде організована за допомогою використання соціальних сервісів Web 2.0. В програмі продовження педагогічного експериментального дослідження Н.В. Морзе пропонує розробити, обґрунтувати, експериментально перевірити та впровадити в систему післядипломної педагогічної освіти дистанційний курс «Використання ІКТ у навчальному процесі» (на основі використання сервісів Web 2.0); створити локалізовану та адаптовану 10-ту версію посібника для вчителів очної і дистанційної форм навчання на основі застосування соціальних сервісів Web 2.0 та відповідного програмного компакт-диску і матеріалів створеного навчально-освітнього порталу [264].

Методологічною основою впровадження соціального програмного забезпечення є теорія конструктивізму [32; 546], теорія соціального конструктивізму [274], теорія конструкціонізму [528], розмовна модель навчання [512], теорія соціального конструювання технологій [466], комбінована модель навчання [463], теорія коннективізму [556] та теорія соціальних мереж [494; 569].

Так, *конструктивізм* виходить з того, що навчання – це активний процес, в ході якого люди активно конструюють знання на основі власного досвіду («створення» знань замість їх отримання). Ідеї конструктивізму виражені і в теорії діяльності, згідно з якою діяльність та дії самої дитини є основою його психічного розвитку. Більш того, перехід від предметної дії до дії з моделями об’єктів пізнання є невід’ємною умовою формування розумових здібностей дитини. С.А. Раков до основних понять конструктивізму відносить дослідницьке навчання, навчання через діяльність, експериментування, навчання через відкриття та ін. [274]

На основі соціального конструктивізму (значимість якого в навчанні програмуванню особливо підкреслював основоположник мобільного навчання А. Кей) С. Пейпертом був розроблений новий напрямок – *конструкціонізм*, за яким до активної позиції соціального конструктивізму додається ідея про те, що люди створюють нове знання особливо ефективно, коли вони залучені до створення продуктів, наділених особистісним змістом: головне те, що люди в процесі конструктивної діяльності створюють щось важливе для них самих або

оточуючих. Конструкціонізм протиставляється інструкціонізму – процесу навчання як передавання знань від знаючого до незнаючого.

Різні електронні середовища, усередині яких учасники можуть створювати свої власні цифрові об'єкти, обмінюватися такими об'єктами, видозмінювати електронні об'єкти, є конструкціоністськими. До таких середовищ можна віднести багатокористувацькі світи, системи управління знаннями (наприклад, Moodle), різні сервіси Web 2.0. Чим складніше і цікавіше об'єкти, якими може обмінюватися співтовариство, тим більші можливості для навчання воно відкриває для своїх учасників.

Розвиваючи концепцію С.А. Ракова, можна виділити наступні *принципи організації мобільного навчання*:

1. Організація навчання як дослідження.
2. Створення навчально-дослідних співтовариств.
3. Особистісно орієнтоване навчання.
4. Співпраця в процесі навчання.
5. Насичення освітнього простору носіями знання.

Останній принцип, який ще можна назвати принципом надмірності, передбачає, що мобільне навчання потребує більше часу, ніж традиційне; крім того, суцільна віртуалізація навчання в школі може призвести до втрати соціальних контактів як між учнем та вчителем, так і між самими учнями. Лише комбінуючи традиційне та мобільне навчання, можна не лише дати професійні знання, а й сформувати загальну культуру особистості. Тому для тих осіб, котрі не мають особливих потреб, доцільно застосовувати мобільні технології дистанційного навчання як допоміжні в процесі традиційного навчання та як основні – в процесі позакласної (зокрема, факультативної) роботи.

До руйнівних організаційних проблем мобільного навчання (за влучними виразом М. Шарплес, «руйномобільного навчання» – disruptive mobile learning) відносять: захоплення мобільними іграми, кібер-знущання, втрату вчителем контролю, введення в оману на екзаменах та ін. І, хоча саме таке «вторгнення дому до школи» викликало до життя обговорювань у п. 2.1.2 наказ МОН Украї-

ни, існує й обернена тенденція «вторгнення школи до дому»: батьківській контроль за навчальною діяльністю дитини через шкільний інтранет, постійне відслідковування активності дитини через мобільний телефон та GPS-пристрої, оцінювання позашкільного навчання та ін.

Розв'язання вказаних проблем вимагає систематичної роботи з формування культури користування мобільними засобами як складової загальної культури особистості. МОН України разом з мобільним оператором «Київстар» підготовано посібник з формування культури користування мобільним зв'язком [414], що містить методичні основи організації навчально-виховної роботи з проблеми формування мобільної культури, напрямки до обговорення питання популяризації правил мобільного етикету серед молоді та методичні рекомендації батькам.

2.4.3. Об'єктно-орієнтоване середовище мобільного навчання. Технічно реалізація мобільного навчання можлива у кількох варіантах [308]:

а) WAP-інтерфейс [180];

б) клієнт-серверна система на основі однієї із систем дистанційного навчання;

в) статичні та динамічні Java-додатки (в т.ч. на основі технології Google Android).

При реалізації мобільного навчання використовуються наступні комунікаційні стандарти: GSM, GPRS, UMTS, Wi-Fi, Bluetooth. Інфрачервоний зв'язок можливий, проте не застосовується через малу (1–2 метри) максимальну відстань передавання сигналів. Технічні недоліки мобільних пристроїв обумовлені переважно сучасним станом розвитку технології: обмежений розмір пам'яті, менша (порівняно з ПК) потужність процесора, обмежений ресурс акумуляторів, обмежена роздільна здатність екрану.

Враховуючи, що традиційний WAP-інтерфейс поступово зникає, зосередимо увагу на клієнт-серверних мобільних технологіях, застосування яких в навчальному процесі дає можливість реалізувати концепцію мобільного освітнього середовища, визначальними особливостями якого є можливість завантажен-

ня і встановлення програмного забезпечення та наявність розвинених засобів отримання та опрацювання контенту. Головним компонентом такого середовища є мобільний портал дистанційного навчання (М-портал), вимоги до якого були визначені Ю.В. Триусом [410].

Основними видами діяльності в системі підготовки кадрів на основі мобільного навчання є:

– формування ринку освітніх послуг;

– розробка нового або адаптація (за необхідності) існуючого базового програмного забезпечення;

– розробка нового або адаптація (за необхідності) існуючих систем управління навчальним процесом;

– розробка навчальних курсів.

Якщо виключити перший компонент, то залишається єдине місце для зберігання навчальних матеріалів, доступу до них, підтримки та оновлення – М-портал.

М-портал – це Інтернет-сайт, користувачі якого після реєстрації та отримання певних прав можуть використовувати навчальні ресурси, створювати власні мікропортали, відвідувати мікропортали студентів, викладачів та інших користувачів в рамках онлайн-спільноти, мати доступ до модулів мобільного навчання та пов’язаних з ними систем управління навчанням. М-портал призначений для потенційних користувачів, які володіють необхідними практичними навичками.

Програмне забезпечення М-порталу має бути придатним для подання навчального контенту, трансляції лекцій, ведення дискусій та передавання повідомлень. Реалізація гнучкого зворотного зв’язку засобами М-порталу підвищує зацікавленість в довгостроковому навчанні та створює умови для включення суб’єктів навчання в планування, покращення та оцінювання самого навчального процесу.

Концепція М-порталу створює основу для низькорівневої «розмовної» моделі навчання, в якій молоді люди були б цілеспрямовані, їхня увага зосере-

джена на оволодінні навчальним матеріалом у процесі інтерактивного спілкування з викладачем у навчальному середовищі. В той же час на більш високому рівні студенти та викладачі можуть обмінюватися миттєвими повідомленнями в мережі та обговорювати в онлайні те, що зробив кожен з них, розглядати різні думки і погляди з обговорюваних питань, навчатися формулювати власні думки в ході обговорення.

Сьогодні в одній освітній установі, як правило, застосовуються гібридні мережі, що об'єднують як стаціонарні, так і мобільні пристрої (рис. 2.8). Г.Г. Швачич, аналізуючи результати впровадження Wi-Fi доступу в Національній металургійній академії України, зазначає, що це, в свою чергу, стимулює студентів до придбання ноутбуків або КПК [105]. Такий позитивний зворотний зв'язок створює можливість швидкого впровадження мобільного навчання.

Включення в традиційну мережу навчального закладу засобів мобільного навчання реалізується через систему управління навчанням (Learning Management System – LMS), що базується на Web-послугах з обміну XML-контентом за стандартами Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL), Universal Description Discovery and Integration (UDDI). Ці послуги також призначені для здійснення автоматичного розподілу XML-документів за виконуваними програмами, проектами, базами даних тощо. На їх основі створюються необхідні передумови для переходу від PC-центричних до розподілених мобільних систем, в яких з різних пристроїв (мобільні комп'ютери, PDA, Tablet PC, смартфони та ін.) можна здійснювати доступ до освітніх XML-ресурсів з будь-якого місця. Web-послуги є сучасною реалізацією розподілених обчислень. В них передбачається єдиний спосіб опису додатків, їх публікації та розташування в мережі.

При проектуванні архітектури мобільного освітнього середовища необхідно враховувати перспективи його розвитку, для чого доцільно застосовувати модульну інтеграцію його компонентів на основі стандартів. На рис. 2.9 показана архітектура системи Web-послуг комбінованої мережі, наповнення, інтеграція та зберігання даних в якій відбувається за стандартом UDDI, інтерфейс

описується за допомогою WSDL, а передавання – за допомогою SOAP, що дає користувачеві можливість користування зовнішніми додатками незалежно від використовуваних платформ, систем та стандартів. Тоді при виборі користувачем мобільного пристрою навчального курсу автоматично виберуться саме ті навчальні об’єкти, які підтримується на даному пристрої. Все це забезпечує функціональність для багаторазового використання об’єктів і послуг, що скорочує час розробки додатків.

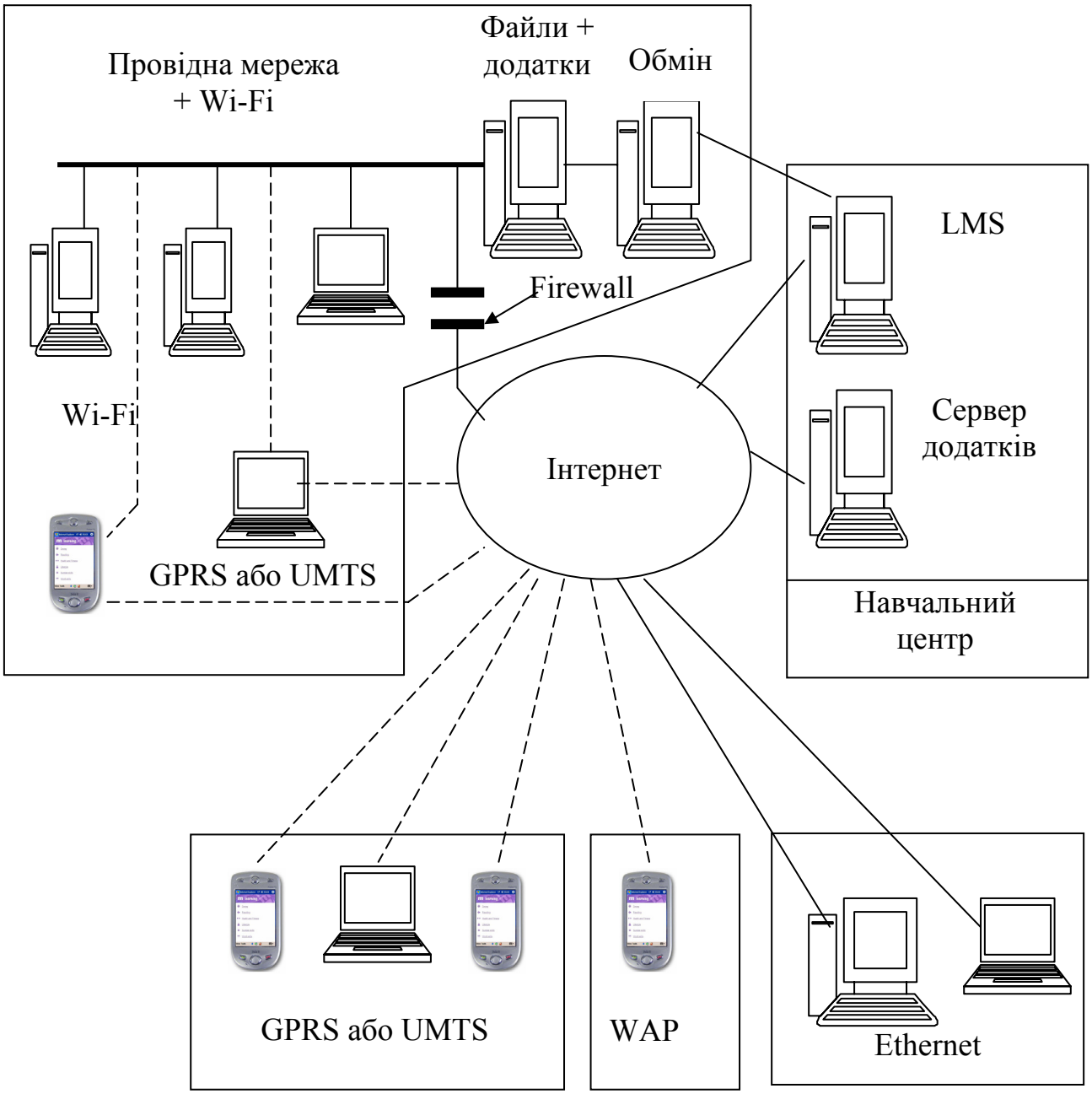


Рис. 2.8. Структура гібридної мережі навчального закладу

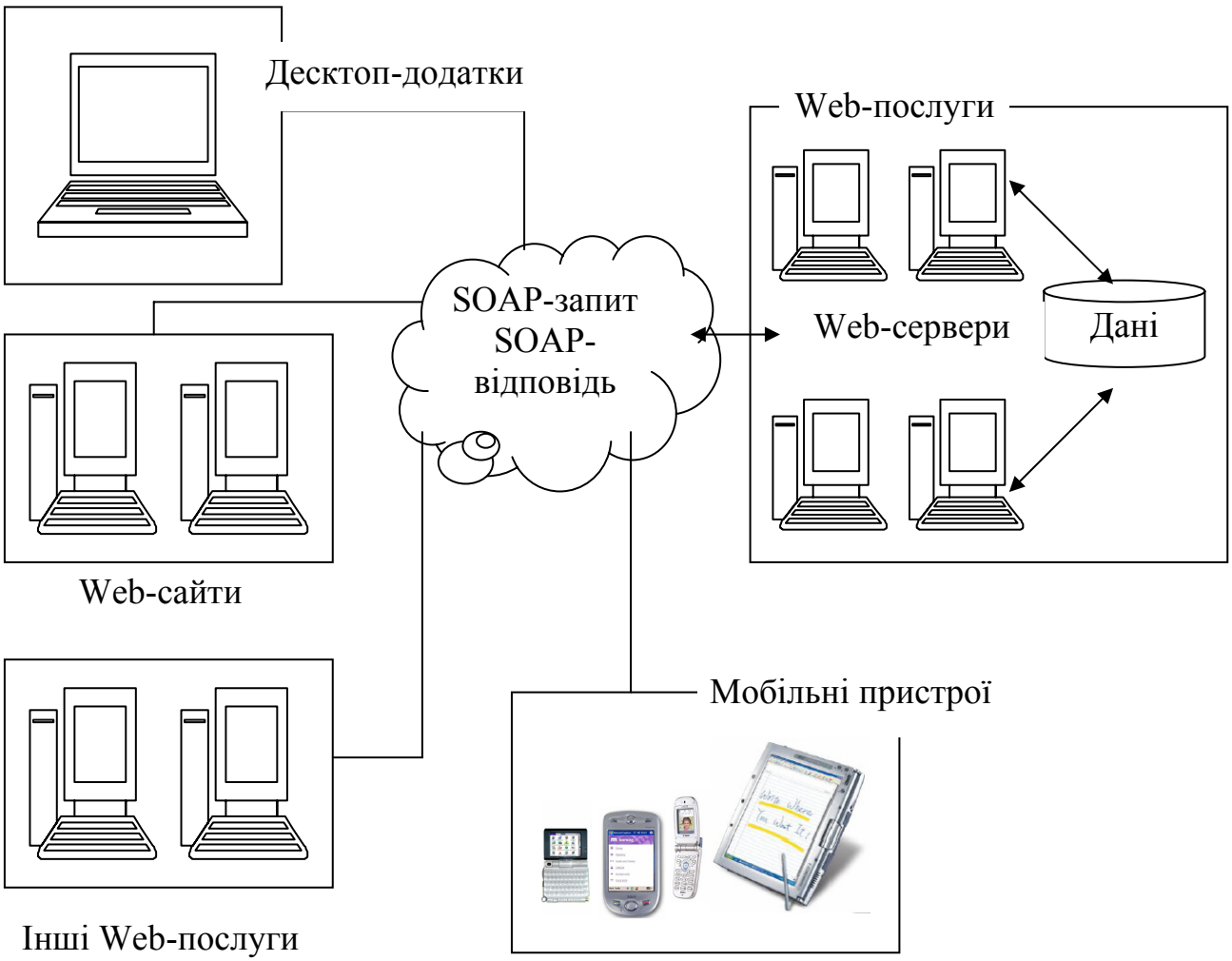


Рис. 2.9. Архітектура Web-послуг комбінованої навчальної мережі

Застосування стандартів надає можливість побудувати відкрите, модифіковане та масштабоване plug-and-play (самоналагоджуване) середовище мобільного навчання, що надаватиме широкий спектр освітніх послуг [44].

При використанні навчального курсу на мобільному пристрої в системі повинні збиратися необхідні об’єкти для завантаження, а потім направлятися на мобільний пристрій. Крім того, спосіб подання навчального матеріалу та електронних засобів навчання на мобільному пристрої має бути обраний з урахуванням характеристик цього пристрою. На основі цього можна запропонувати гнучку архітектуру послуг для мобільного навчання (рис. 2.10).

Мобільне навчання може включати широке коло навчальних матеріалів – від допомоги у виконанні конкретної роботи та автономних курсів, завантажуваних на КПК студента, до повністю мережних курсів з програмним забезпе-

ченням, що виконується на сервері. Запропонована архітектура мобільного навчання на основі мережних послуг є відкритою, масштабованою, глобальною та самоналагоджуваною. Для створення самоналагоджуваного середовища мобільного навчання необхідна підтримка різних рішень, що пропонуються виробниками, архітектура яких ґрунтується на стандартах. Відкритість і розширюваність архітектури сприяє її застосуванню у різних видах діяльності, забезпечуючи гнучкість і задоволення широкого кола освітніх потреб.

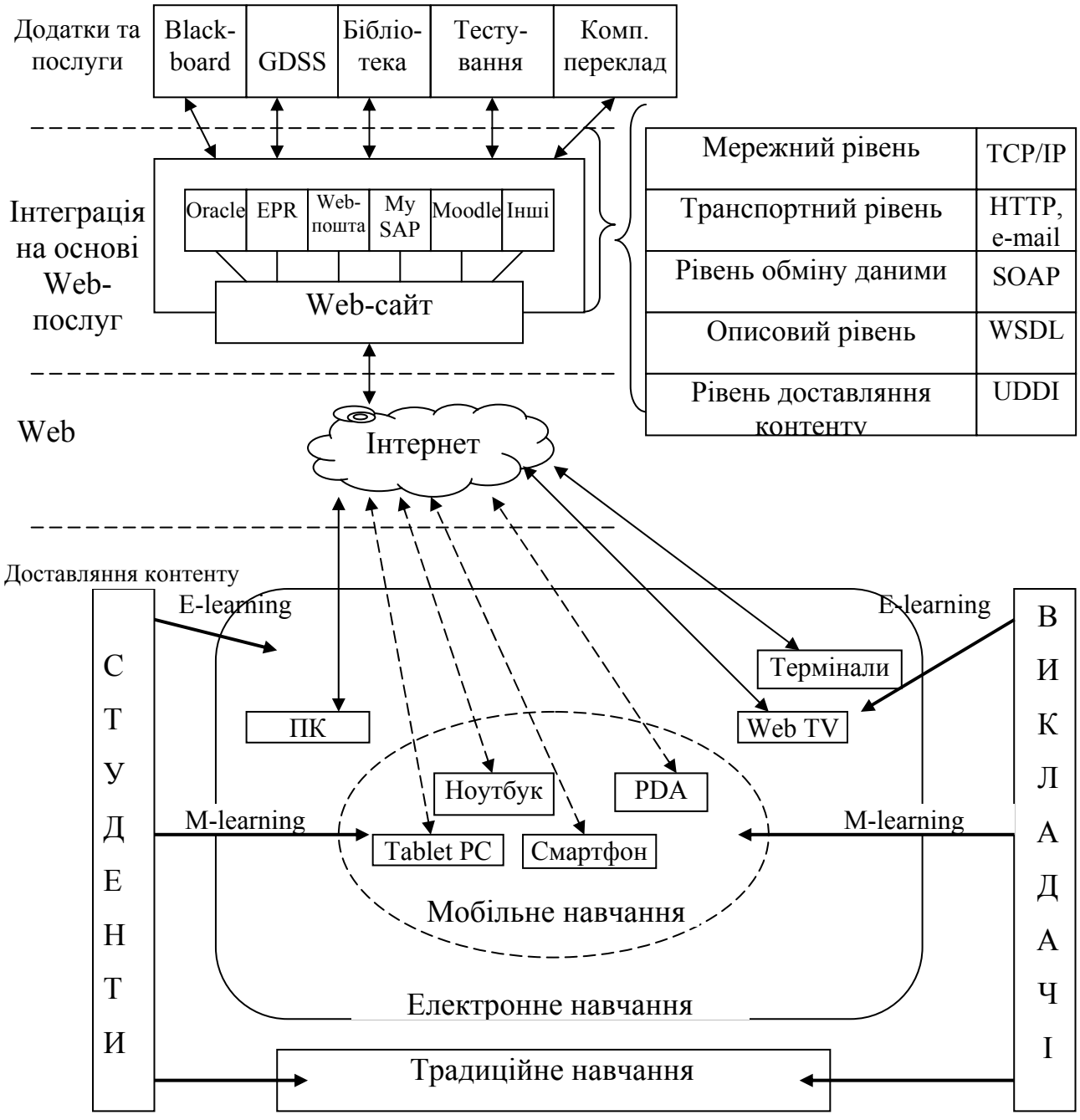


Рис. 2.10. Об’єктно-орієнтована архітектура середовища мобільного навчання

Застосування об’єктно-орієнтованої архітектури надає можливість інтегрувати найрізноманітніші системи (електронні бібліотеки, віртуальні та/або реальні лабораторії), здійснювати управлінням навчанням, надавати найрізноманітніші інформаційні послуги. З рис. 2.10 можна побачити, що в обговорюваній архітектурі поєднуються 4 рівні: додатки та послуги, інтегровані за допомогою Web-стандартів, власне Web та мобільне навчання.

Перший рівень – рівень взаємодії між викладачами та студентами – різні послуги, що надаються викладачам та студентам за допомогою додатків різних типів, як комерційних, так і вільно поширюваних (електронні бібліотеки, словники, перекладачі тощо).

Другий рівень – рівень Web-послуг, де інтегруються навчальний контент та програмні додатки, подані в різних форматах. В архітектурі, що використовується для цього, передбачається інтеграція та технології типу plug-and-play. На цьому рівні навчальний матеріал не залежить від застосовуваного мобільного пристрою, а його об’єктно-орієнтована структуризація дає можливість одночасного використання текстових даних, голосу, звуку, відео, тестів та різних виконуваних файлів. На цьому рівні також контролюється безпека, якість обслуговування та поширення навчальних матеріалів. На рівні інтеграції забезпечує доступ до всіх внутрішніх систем та інструментів для створення навчального контенту, що надає авторам можливість інтегрувати навчальні матеріали, включаючи текст, графіку, процедури оцінювання, відео та інші завантажувані компоненти.

Третій рівень – рівень доставляння контенту на різні мобільні та стаціонарні пристрої, що забезпечує доступ користувача до навчальних ресурсів будь-коли та будь-де. На цьому рівні підтримуються персональні комунікаційні системи, в тому числі багатофункціональні мобільні телефони, електронна пошта, ПК, мережні сервери, TV, AM/FM-радіо та GPS (Global Positioning System).

Четвертий рівень – рівень мобільного навчання – діяльність студентів, викладачів та адміністраторів М-порталу.

Феng-Хуан Ю Янгом [526, 40] запропоновано архітектуру середовища мобільного навчання на основі грід-сервісів – комбінації Web-сервісів з мовою опису правил їх функціонування, поширюваних за допомогою відповідних сервіс-провайдерів [513]. В лівій частині рис. 2.11 кілька сервісів об'єктів навчання (LO) підтримуються різними розробниками контенту. Ці сервіси не лише фізично розташовані у різних місцях, а й обслуговуються різними платформами. Кожен елемент, що містить LO-сервіс, представлений віртуальною установою, що може мати власну навчальну платформу, операційну систему, обладнання тощо. Різні LO-сервіси пов'язані один з одним через GSFL (Grid Service Flow Language – мову опису грід-сервісів). В центрі рис. 2.11 – грід-машина, у правій частині – грід-клієнти (різноманітні мобільні пристрої).

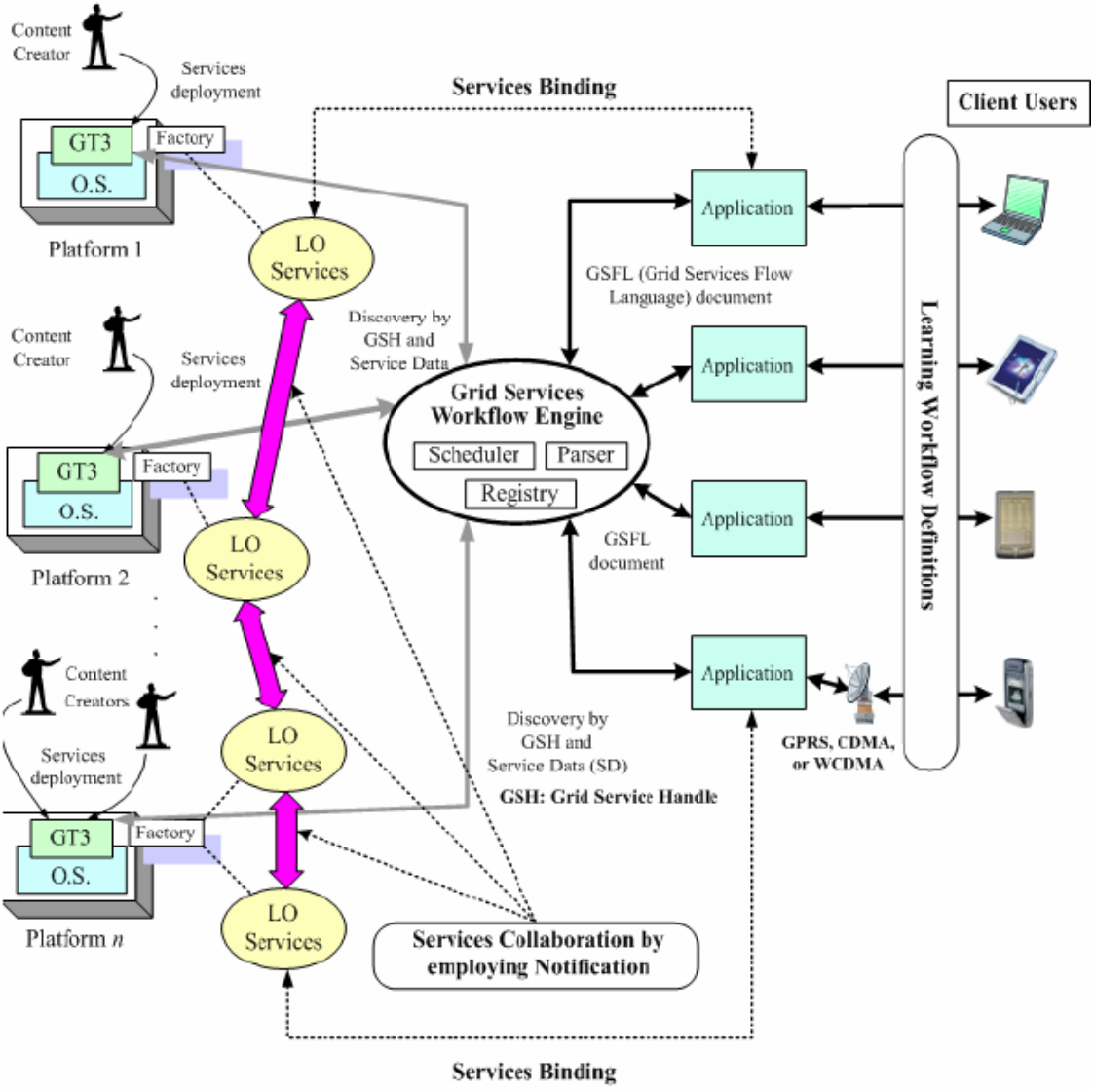


Рис. 2.11. Грід-архітектура середовища мобільного навчання (за Ф.-Х. Ю Янгом)

2.4.4. Система управління мобільним навчанням (Mobile Learning Management System – MLMS) прихована за М-порталом. MLMS є тією стороною мобільного навчання, яку користувач не бачить, але постійно використовує (рис. 2.12). MLMS розробляються у відповідності до потреб суб'єктів навчання, якими в нашому випадку виступають користувачі з мобільними пристроями. Через різноманітність таких пристроїв MLMS повинні бути гнучкими і автоматично пристосовуватися до пристроїв, тому що не всі мобільні телефони і КПК є однаковими – як було показано у п. 2.3.1, вони оснащені різними екранами, процесорами, пам'яттю та засобами введення.

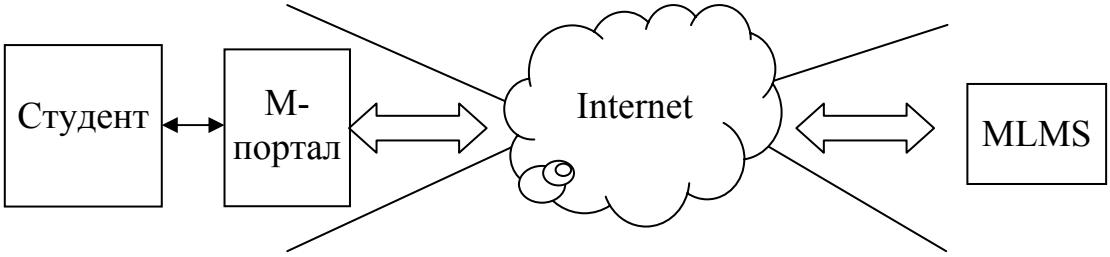


Рис. 2.12. Місце MLMS у М-порталі

MLMS являє собою систему, за допомогою якої на М-порталі опрацьовується запит від студента на надання йому необхідних навчальних матеріалів. Це платформа, через яку забезпечується мобільний доступ до навчальних матеріалів, послуг і моделей, адаптованих для використання в мобільному середовищі.

За допомогою системи управління навчанням надаються послуги трьом типам користувачів, з урахуванням специфіки мобільних пристроїв:

– *користувачі з HTML* працюють із системою мобільного дистанційного навчання за допомогою свого мобільного браузера. Такі користувачі можуть взаємодіяти з іншими студентами та викладачами за допомогою дискусійних форумів та інших спільних ресурсів Інтернет. Мережний (Web) зміст адаптується для малих екранів мобільних пристроїв;

– *користувачі без HTML* працюють із системою за допомогою SMS. Курси організовуються SMS-засобами мікронавчання з використанням короткотривалих (малих) навчальних об'єктів. У випадку тестування студент може використовувати SMS безпосередньо;

– третє, але не останнє місце посідає *голосова взаємодія*. Студент отримує доступ до системи за допомогою простого телефонного дзвінка і може управляти роботою з М-порталом за допомогою голосових команд. У відповідь на дзвінок синтезованим людським голосом читається спрощена версія контенту на порталі, а користувач може рухатися по ньому за допомогою введення команд з клавіатури.

MLMS є організаційно-технічною основою мобільного навчання. Перш ніж будувати таку систему, необхідно особливу увагу приділити вимогам гнучкості. При проектуванні MLMS використовуються стандартні підходи до проектування складних програмних продуктів, широко перевірених на практиці – об'єктно-орієнтований аналіз та проектування. В MLMS необхідно передбачити запис студентів на курс, реєстрацію, посилання на навчальні матеріали, ступінь покриття різних видів мобільних пристроїв, вхідні та вихідні дані, поповнення новими матеріалами, виправлення старих, забезпечення ходу навчання, способи контролю знань і забезпечення безпеки та захисту даних. Всі ці речі повинні бути враховані для того, щоб запропонувати користувачеві найкращий засіб навчання.

На рис. 2.13 показана спрощена модель роботи MLMS на прикладі створення тесту. Ця модель може бути розширена базою навчальних матеріалів, стратегіями управління навчання, електронними бібліотеками та ін.

В умовах мобільного навчання текстове подання навчальних матеріалів «підсилюється» голосом, відео, анімацією. Навіть в обмежених умовах для зв'язку і комунікації можна пройти навчання, отримавши в онлайн-режимі необхідні знання для забезпечення систематичного засвоєння матеріалу за допомогою постійного (always on) під'єднання або під'єднання на вимогу (on demand).

Врахування потреб суб'єктів мобільного навчання вимагає гнучкого подання навчального матеріалу з можливістю його доставляння у будь-якому вигляді. Для цього необхідно визначити таку модель змісту навчання (контенту), коли забезпечуватиметься одночасно його подання та навігація. Навчальний

матеріал має бути розроблений таким чином, щоб його можна було доставити незалежно від обраного способу подання, розділивши зміст та спосіб доставляння на обраний тип мобільного пристрою.

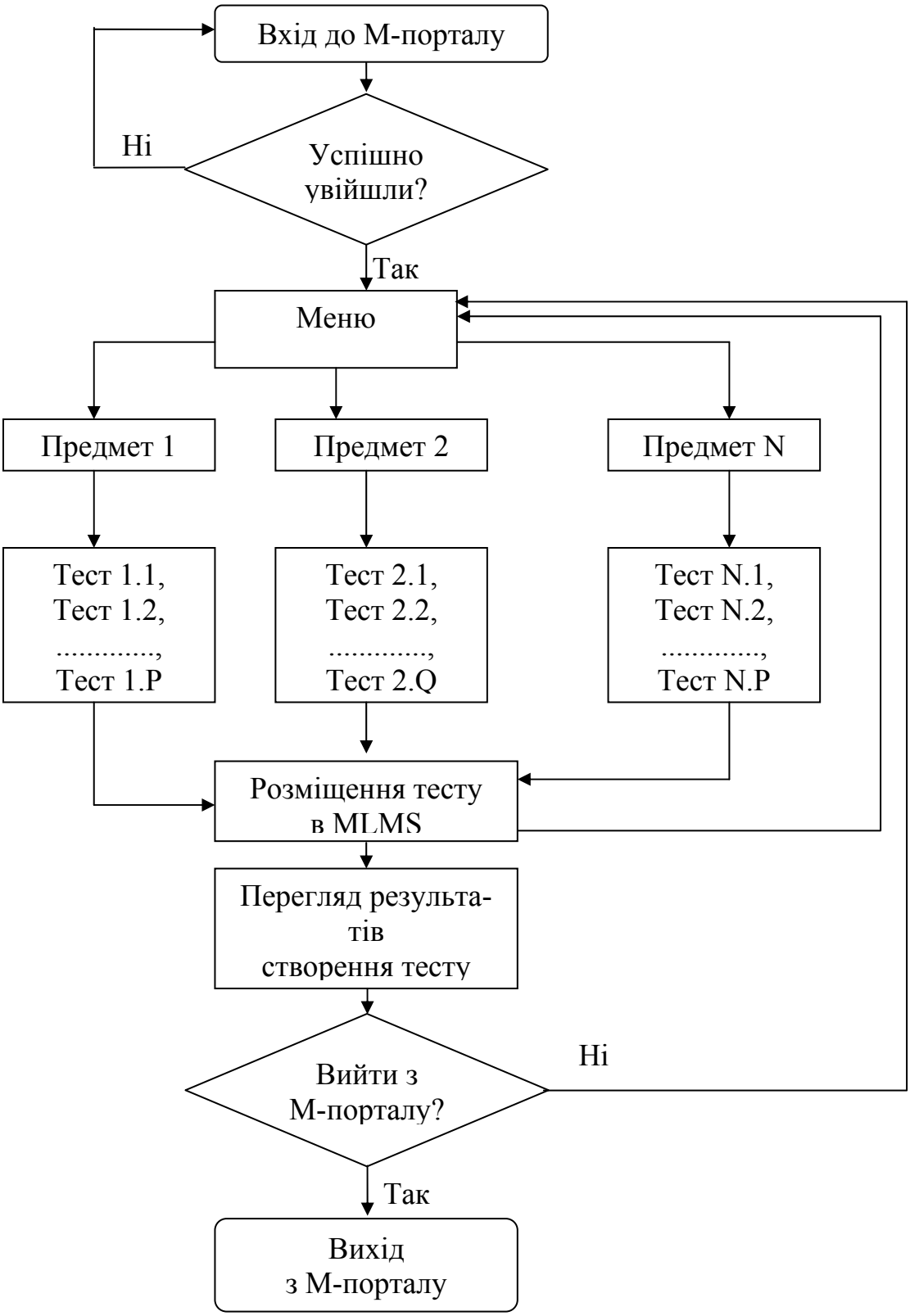


Рис. 2.13. Спрощена блок-схема роботи підсистеми створення тестів MLMS

На наступному рівні моделі необхідно додати засоби врахування контексту запиту і забезпечення його задоволення у відповідності до можливостей користувача. Це вимагає «інтелектуального» доставляння освітнього контенту, тому MLMS може містити «інтелектуальні» елементи. У цьому випадку вона буде містити ряд конкретних компонентів і, перш за все, моделі знань про конкретну предметну галузь, моделі навчання та «інтелектуальний» машинний інтерфейс (рис. 2.14).

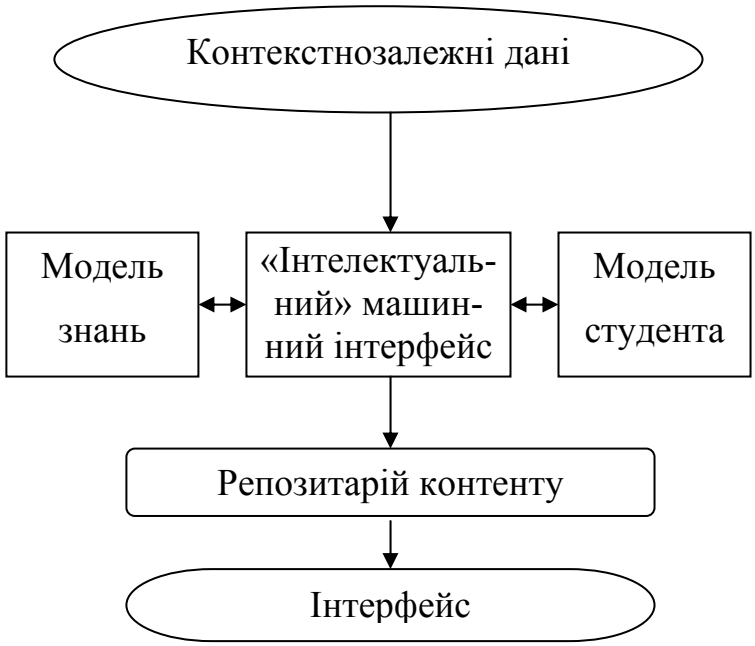


Рис. 2.14. Структура «інтелектуальної» MLMS

В найпростішому випадку управляти навчанням можна через добір контенту, а знаючи потреби користувача, умови його навчання та використовуваний пристрій, можна забезпечити доставляння:

– необхідних відомостей,

– необхідній особі,

– на необхідний пристрій,

– правильно,

– у потрібний час,

– у потрібному місці;

– у відповідному контексті.

Об’єктно-орієнтований стандарт SCORM (Sharable Content Object Reference Model), розроблений для систем дистанційного навчання, в поєднанні з Web-стандартами для гнучкого подання змісту на основі XML, служить основою для розробки змісту, незалежного від подання на екрані пристрою, і дозволяє використовувати правила форматування контенту для найкращого відображення. Поширення таких стандартів, як XML-мова моделювання навчання EML (Education Modeling Language) [468], дозволить розв’язувати відповідні освітні проблеми і у Web 2.0, підтримку якого стандартом SCORM заплановано ввести у жовтні 2009 р. На відміну від SCORM, за допомогою EML можна описувати не лише контент (тексти, вправи, тести тощо), а й ролі, стосунки, взаємодію студентів та викладачів.

2.4.5. Розробка навчальних матеріалів для мобільного навчання. Поширення електронного навчання і нових мобільних комунікаційних технологій веде до постійного збільшення кількості електронних підручників та навчальних матеріалів, що, в свою чергу, впливає на технології їх розробки та подання. Якщо раніше електронні підручники були набором не дуже добре ілюстрованих HTML-сторінок, то сьогодні вони створюються за допомогою великого набору інструментів комп’ютерно підтримуваного моделювання процесів, явищ, механізмів. Розробники активно використовують Java-аплети, включають до браузера модулі для відтворення анімації засобами Flash, VRML (X3D), AXEL, Viewpoint та ін. (див., наприклад, [415; 447; 483]). Все це приводить до підвищення інтерактивності навчання, візуальної привабливості курсу, підвищення ефективності навчання.

У розробці курсів для мобільного навчання можна виділити три покоління, пов’язані із застосовуваними програмними та технічними засобами [506].

Покоління 1. Розробка засобами WML для WAP 1.1. Це загальні телекомунікаційні технології, що не є специфічними для електронного навчання.

Покоління 2. Розробка засобами XHTML для WAP 2.0. Ці засоби орієнтовані на телефони з кольоровими великими екранами та функціональними брау-

зерами типу Opera. До них може бути додана функціональність відео (в т.ч. потокового).

Покоління 3. Розробка засобами Flash Lite. Нинішнє покоління мобільних навчальних курсів базується переважно на Flash Lite. Ця тенденція пояснюється тим фактом, що є тисячі розробників, які використовують Flash для створення систем електронного навчання та їх контенту, і їх педагогічні та технічні навички можуть бути за мінімальних змін використані у мобільному навчанні.

Слід зауважити, що стрімкий розвиток мобільних технологій призвів до того, що пристрої, на які орієнтоване кожне з поколінь, співіснують, тому навіть засоби першого покоління не можна відкидати як застарілі.

Відображення Web-сторінок на мобільних пристроях суттєво відрізняється від стаціонарного ПК. Якщо це не береться до уваги при форматування сторінок, їх зміст може змінитися іноді настільки, що втратить сенс. Тому при перенесенні навчальних матеріалів на мобільні пристрої доцільно враховувати наступні рекомендації з розробки навчальних матеріалів для мобільного навчання.

1. Якщо це можливо, використовувати зображення невеликих розмірів (не більше 150 пікселів в ширину для картинки, включеної у текст, що трохи більше половини екрана КПК).

2. Намагатись не використовувати графіку для основних елементів інтерфейсу на сторінці, тому що її відображення може бути відключеним; надписи на графічних кнопках дублювати текстом за допомогою атрибуту «Alt».

3. Користуватись правилами поділу матеріалу при мікронавчанні: скорочувати розмір сторінки так, щоб жоден абзац не виходив за її межі: у такий спосіб студент уникне постійного тривалого «прокручування» матеріалу на екрані, а час завантаження матеріалу скоротиться.

4. Не ділити сторінку на фрейми: вони будуть показані в порядку, що відрізняється від звичного при використанні стандартного монітора.

5. Не використовувати клієнтських сценаріїв відкривання сторінок у нових вікнах та спливаючих вікон: для користувачів мобільних пристроїв це досить незручно.

6. Не форматувати графіку за допомогою таблиць: в мобільних браузерах може трапитись перенесення клітинок таблиці, що призводить до перекручування і втрати сенсу навчального матеріалу.

7. При розміщенні тексту користуватися тим самим принципом (п. 6).

8. Не використовувати динамічне меню, оскільки його вміст може не поміститися на одному екрані, і деякі пункти будуть недоступні. Це також відноситься і до контекстно-залежної допомоги; краще відкрити її на окремій сторінці.

9. Розробники онлайнових курсів мають вказувати, як студенти можуть отримати допомогу (як в автономному режимі, так і в онлайні).

10. Навчальні матеріали слід подавати, за можливості, з різних точок зору і способів виконання завдання.

11. При розробці навчальних матеріалів та інтерфейсу користувача слід звернути увагу на їх подання в різних форматах для забезпечення доступу студентів, які мають проблеми зі здоров'ям.

12. Де можливо, застосовувати анімовані GIF-файли замість відеороликів.

Застосування нових технологічних засобів визначає нові особливості процесу навчання:

– інтерактивність;

– поділ контенту на окремі рівні за складністю подання;

– контекстно-залежні мікронавчальні об'єкти;

– використання «полегшеної» графіки, аудіо та відео;

– мультимедійність;

– часте використання повідомлень типу SMS та MMS.

Функціональність браузерів для КПК все ще досить обмежена у порівнянні з браузерами ПК: хоча всі індустріальні стандарти (CSS, SSL, HTML, JavaScript, Flash) в них підтримуються, ступінь підтримки суттєво відрізняється. Для браузерів ПК для підтримки нових функцій досить завантажити останню версію браузера. На КПК не можна це зробити так просто – як правило, вона розробляється для конкретної операційної системи (ОС), а завантаження нової

ОС на старі КПК в більшості випадків неможливе. Така ж проблема і в смартфонах. Природно, що незабаром ці проблеми будуть вирішені і весь стандартний набір специфікацій та характеристик для настільних браузерів буде працювати і в мобільних пристроях.

Все це вимагає при розробці навчальних матеріалів, за можливістю, на рівні М-порталу контролювати використовувану мобільну платформу та використовуване програмне забезпечення для того, щоб в залежності від них користувач отримував навчальний матеріал, при поданні якого найбільш повно враховуються характеристики саме його пристрою. Якщо ж такої можливості немає, то можна надати користувачеві можливість самостійно обирати необхідну конфігурацію (підтримку CSS, JavaScript, Flash тощо).

Це аж ніяк не додає зручності для Web-дизайнера, програміста та викладача, які завжди хочуть, щоб їхній курс був привабливим, легким для перегляду та засвоєння. Вирішенням цієї дилеми є використання таких засобів розробки навчальних матеріалів, в яких підтримується експорт в мобільні формати. Наприклад, в Macromedia Flash MX 2004 вбудовані шаблони для різних мобільних пристроїв – від КПК до мобільних телефонів. При створенні зображення враховується не тільки фактичний розмір екрану, а й спосіб його відображення на мобільному пристрої. Це допомагає краще розташувати елементи інтерфейсу і навігації. Так, компанія Trivantis пропонує модулі розширення для своєї системи розробки курсів Lectora Publisher, використання яких можна створювати навчальні матеріали, придатні для відтворення на платформи Pocket PC або PalmOS. Застосування інструментів для створення моделей Adobe RoboDemo 5 надає можливість імітувати роботу «настільних» програм на КПК або будь-якому іншому мобільному пристрої з Macromedia Flash Player.

В рамках проекту M-learning [458] розроблено ряд простих у використанні інструментів, призначених для створення свого власного контенту: MyLearning author являє собою набір простих у використанні програмних інструментів, за допомогою яких можна створити динамічне діяльнісне навчальне середовище для роботи студентів на своїх кишенькових комп’ютерах;

MyLearning resources – бібліотека ресурсів, об'єднаних у mediaBoard; використання SMS quiz author надає можливість створити автоматичну SMS-відповідь на тест із кількома варіантами відповідей.

Деякі компанії, такі як Opera Software, розробили спеціальні алгоритми для опрацювання Web-сторінок, відформатованих для відображення на стандартному моніторі. В браузері Opera Mobile 9.5 використовується режим Small-Screen Rendering, призначений для мобільних пристроїв з екраном шириною 128 пікселів або менше. Весь текст на сторінці розташовується в один стовпчик з можливістю «прокрутки» лише вгору та вниз. Довгі списки та панелі автоматично стискаються (функція «content folding» – в цьому режимі зображення звужується так, що не перевищує 70 % від розміру екрану в довільному напрямі). В настільній версії браузера (Opera Mini) існує можливість перегляду Web-сторінки в цьому режимі, що надає авторам можливість швидко тестувати навчальні матеріали для мобільного навчання.

В більшості мобільних браузерів використовується движок WebKit. Розробники Mozilla Corporation планують найближчим часом випустити версію движка Gecko та браузера Mozilla Firefox, оптимізовану для мобільних пристроїв.

Компанією Hot Lava Software (http://www.hotlavasoftware.com) розробляються два продукти: пакет розробника Learning Mobile Author (LMA) та засіб публікації контенту Mobile Delivery and Tracking System (MDTS) на PocketPC, Windows Mobile, BlackBerry, Nokia, PalmOS, мобільних телефонах, смартфонах та будь-яких нових Інтернет-пристроях [517] (рис. 2.15). В LMA підтримуються формати Flash (версії 7 та 8), Flash Lite (версії 1.1 та 2.2), SVG, аудіо (mp3, amr, mid, wav, aiff, mmf, jar тощо), відео (3gp, mp4, mov, jar і т.п.), подання зображень, тестів, опитувань, тексту та ін. Нажаль, обидва ці продукти є комерційними, і, хоча й дозволяється безоплатне їх використання протягом двох тижнів, вартість повнофункціонального пакету сягає 1000$.

MyMLE (http://mle.sourceforge.net/) – вільно поширюваний аналог LMA, що надає можливість створення навчальних матеріалів для будь-яких пристроїв

з підтримкою J2ME та для СДН Moodle (за допомогою плагіну MLE-Moodle). MLE – відкрита та розширювана система, що активно розвивається.

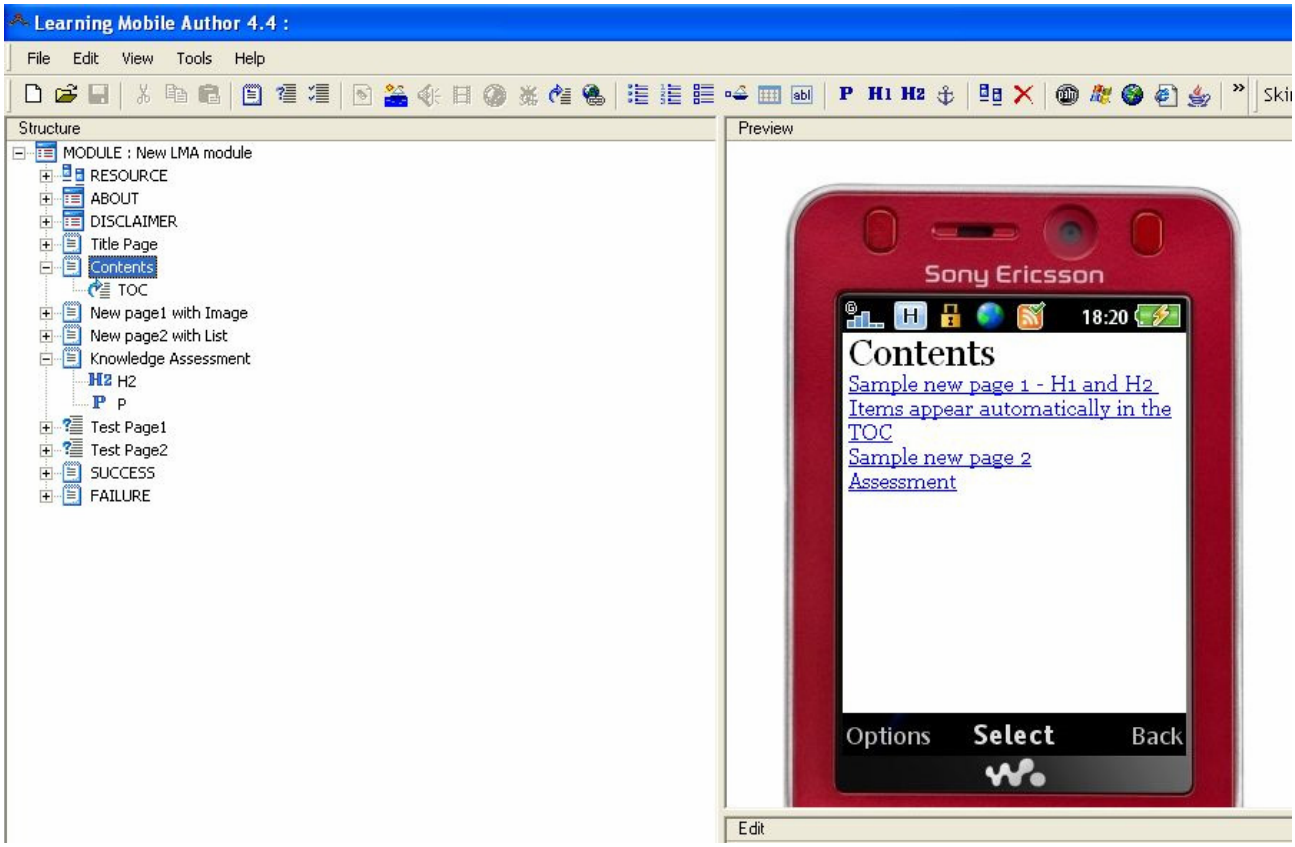


Рис. 2.15. Learning Mobile Author 4.4

2.5. Приклади застосування мобільного навчання

2.5.1. Пілотні проекти мобільного навчання

1. Восени 2002 року близько 300 студентів, котрі навчалися на першому курсі NAIT і Seneca College (Канада), отримали бездротовий доступ до навчальних матеріалів курсу «Вступ до бухгалтерського обліку» за допомогою КПК HP iPAQ у мережі Bell Mobility network [475]. Деякі навчальні матеріали містили пов’язаний з ними аудіо-відео контент. Цей проект надав можливість NAIT і Seneca College вивчити: 1) як мобільне навчання допомагає студентам? і 2) чи впливатиме ця навчальна технологія на можливість залучення абітурієнтів? Успішність пілотного проекту оцінювалася на основі ефективності та дієвості з трьох напрямків: покращення результатів навчання студентів; розширення доступу студентів коледжу до освітніх послуг та розширення методів та засобів навчання для викладачів.

2. Кілька інших коледжів також використовують мобільні технології як для навчання, так і для управління:

а) адміністрація і студенти коледжу Філіпа Морана використовують КПК iPAQ для допомоги у навчанні (наприклад, як портативний розширюваний словник) та в управлінні (наприклад, для доступу до даних про студентів) [559, 110];

б) студенти магістратури університету Бірмінгема були оснащені КПК iPAQ, за допомогою яких вони отримували простий бездротовий доступ до інформації про різні курси, електронної пошти, Web-серфінгу, ресурсів та навчальних матеріалів Інтернет [548].

3. В рамках європейської програми «Leonardo da Vinci» (http://learning.ericsson.net/leonardo/), за якою фінансуються проекти в галузі професійної підготовки і безперервного навчання, за підтримки компанії «Ерікссон» і кількох європейських онлайн-університетів в 2003 році був реалізований проект «Від електронного навчання до мобільного». У рамках цього проекту розроблено MLMS для управління мобільним навчанням та пілотні навчальні курси. Через навчальне середовище на основі Microsoft Reader Works надається кожному студентові, який має програмне забезпечення Microsoft Reader, можливість відображення контенту. Повний текст курсу обсягом до 1000 сторінок А4 легко розміщується в пам'яті стандартного КПК, такого як HP Compaq iPAQ серії 5000.

Цей проект має і другий етап – «Мобільне навчання: Наступне покоління навчання». На даному етапі учасники реалізували переваги від конкретних технологій, характерних для мобільних пристроїв в процесі навчання – потокове аудіо та відео, обмін мультимедійними повідомленнями тощо. В якості інструменту створення контенту використовується Macromedia Dreamweaver MX.

4. Європейський проект «m-Learning» спрямований на залучення до навчання молоді у віці від 16 до 24 років, які найбільш схильні до ризику соціальної нерівності: тих, котрі не встигають у школі, мають серйозні проблеми з вираженням думок у письмовій формі, погані знання з математики і т.п. [458].

Ідея проекту полягає в тому, що ці молоді люди більшу частину часу приділяють спілкуванню зі своїми друзями через мобільні телефони. Хоча в даний час значна їх частина обмінюються лише текстовими повідомленнями і грають у прості ігри, розробники покладаються на те, що багато хто з них будуть замінювати свої мобільні пристрої на більш сучасне обладнання з розвиненими мультимедійними характеристиками. Саме з урахуванням цього факту розроблена LMS із модулем «мікропортал», через який буде надаватися доступ до навчальних матеріалів. Але й ті, хто має старі мобільні телефони, не залишаються осторонь: для них розроблені спеціальні модулі для перетворення мови в текст, тексту в мовлення, доставляння освітнього контенту та відповіді на SMS. Таким чином, партнери проекту мають намір залучити молодих людей до процесу навчання з використанням звичного середовища, засобів комунікації і навіть стилю спілкування.

5. У Санкт-Петербурзькому державному інституті точної механіки та оптики (ІТМО) була розроблена і впроваджена система тестування на основі КПК фірми Palm [352]. Тестування студентів та учнів проходить з використанням КПК і центрального сервера як в онлайні, так і в автономному режимі. Один КПК може використовуватись кількома учнями – на початку тестування необхідно ідентифікуватися.

6. Яскравим прикладом позакласної роботи є проект MyArtSpace, реалізований у 2006 р. М. Шарплесом [562], метою якого було застосування мобільних телефонів для навчання дітей у музеях шляхом створення власної інтерпретації екскурсії до музею. В класі перед екскурсією вчитель обирає її тему. В музеї, кожен експонат якого позначений дволітерним кодом, учні застосовують мультимедійні мобільні телефони для збирання даних: вводять код об’єкту, записують звук, відео, нотатки, роблять фотографії. В процесі збирання всі повідомлення автоматично відсилаються та розміщуються на персональному Web-сайті учня, що являє собою дитячу інтерпретацію екскурсії. Після повернення з екскурсії до класу учні обмінюються зібраним контентом та застосовують його для створення презентацій.

7. У 2006 р. відділення Tribal Learning and Publishing (http://www.m-learning.org/) досліджувало застосування мобільного навчання у сімейному навчанні та вихованні. Метою проекту було експериментальне використання мобільних технологій для розвитку навичок сімейного навчання. Первісна оцінка виконувалась за допомогою опитування учнів для виявлення попередніх знань з використання мобільних телефонів. Далі пропонувалась SMS-вікторина: учні відсилали свої відповіді на вказаний номер і за допомогою майже миттєвого зворотного зв'язку отримували повідомлення від вчителя, наскільки добре вони відповіли. В ході проекту пропонувався ряд заходів, спрямованих на вивчення мови, що використовується в тексті повідомлення, в тому числі через прослуховування, тлумачення абревіатур, обговорення теми з використанням звичайних телефонів або кишенькових комп'ютерів з відправленням повідомлень. Заходи супроводжувалися PDA-вікторинами та включали пошук слів, написання ігор та навчальні дискусії: а) *створи розповідь*: спільна діяльність батьків і дітей з використанням функцій на КПК для створювання розповіді, що містить текст, відео, фотографії тощо; б) *загадки та жарти*: учні можуть розв'язувати головоломки, записувати реакцію на жарти, і навіть створювати короткі відеоролики; в) *алітерація*: вивчення алітерації і створення рими.

8. Важливість роботи, що проводилася в Університеті Преторії (ПАР), полягає в тому, що вона стосується основних університетських курсів. Університет Преторії почав використовувати мобільні телефони у 2002 році для підтримки трьох програм заочної освіти [565], оскільки більш ніж 99% сільських студентів мали мобільні телефони. Профіль цих студентів у 2002 році розподілився наступним чином:

– більшість живе в сільських районах;

– 100% мають повну зайнятість;

– 77,4% володіють англійською як другою мовою;

– 83,8% мають вік між 31 і 50 роками;

– 66,4% складають жінки;

– 0,4% мають доступ до електронної пошти;

– 99,4% мають мобільний телефон.

Більшість з цих студентів живуть у віддалених сільських районах, де малорозвинена або відсутня провідна телекомунікаційна інфраструктура. Мобільний телефон для підтримки дистанційного навчання застосувався як засіб передавання SMS: а) всім студентам; б) окремим студентам для мотивації та підтримки індивідуального навчання; в) конкретним групам студентів для узгодження спільної роботи.

9. У 2005–2007 рр. у Норвезькому університеті науки та технологій (NTNU) був реалізований пілотний проект із застосування мобільних телефонів в курсі біології [536], в якому найцікавіше полягає в отриманні студентами додаткових матеріалів до лекції, записаних у вигляді відео-звіту та розміщеному в університетській LMS. Норвезький університет науки і технологій, розташований у Тронхеймі, є найбільш технологічно розвиненим навчальним ВНЗ Норвегії, в якому широко використовується бездротовий широкосмуговий доступ в приміщеннях університету та в студмістечку. Мобільні телефони є невід'ємною частиною соціального життя студентів NTNU, а можливість широкосмугового доступу до Інтернет на території кампусу спонукає їх до придбання мобільних телефонів з WLAN/3G (Nokia N80 і т.п.). Використання університетської LMS також вже є невід'ємною частиною їхнього навчального процесу, однак використання мобільних телефонів для доступу до LMS є ще порівняно новим для всіх студентів. Студенти використовували WLAN/3G-мобільні телефони або ноутбуки для перегляду відео. В цілому, всі учасники експерименту вітали появу нових можливостей для навчання, а його хід та висновки свідчать про те, що використання відео та мобільних телефонів робить позитивний внесок у навчальну діяльність студентів.

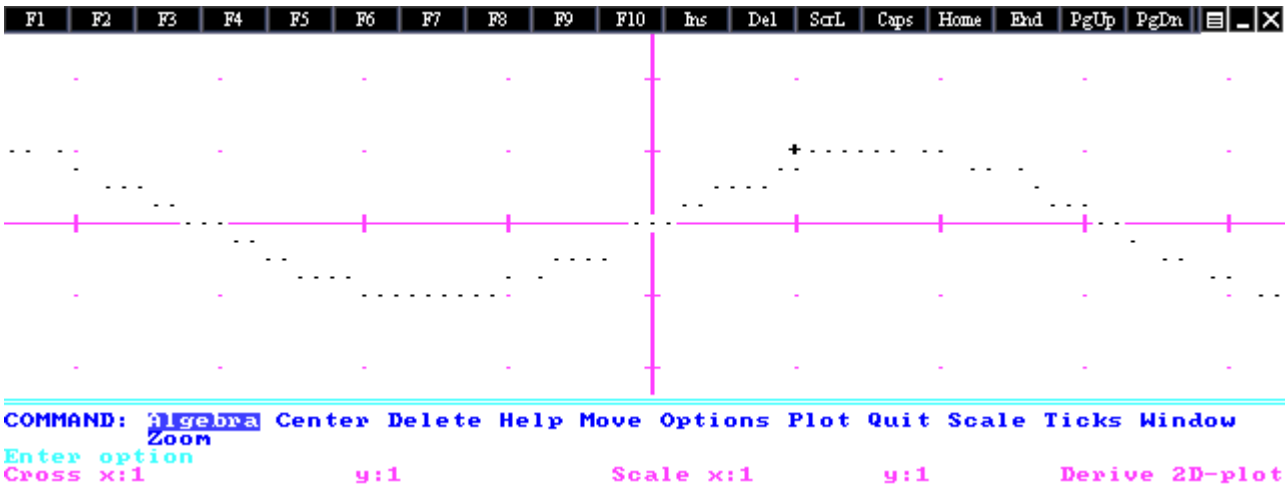
2.5.2. Мобільне навчання інформаційних технологій математичного призначення. У 2006-2007 н.р. проводився педагогічний експеримент із впровадження елементів мобільного навчання інформаційних технологій математичного призначення у старших класах шкіл нового типу м. Кривого Рогу [308]. На початку експерименту в якості платформи для організації дистанцій-

ного навчання були обрані комунікатори з операційною системою на основі Windows CE (WinCE, відомою також як Windows Mobile), що включає розвинені засоби розробки програм та широку підтримку серед виробників. Вибір персональних комунікаторів базується на їх зростаючій поширеності серед учнівської та студентської молоді, ефективних комунікаційних засобах та наявності стандартного ПЗ для роботи з документами в різних форматах.

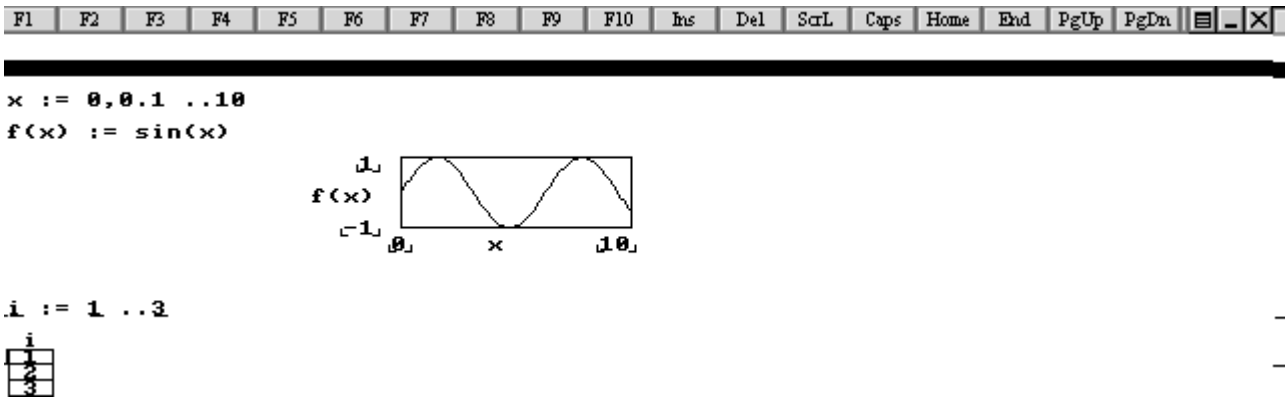
Стандартне ПЗ КПК не включає систем комп'ютерної математики (СКМ), які інколи вважають занадто «важкими» для даного класу пристроїв. Проте багаторічний досвід експлуатації інженерних калькуляторів з СКМ Derive (класу TI-Nspire CAS) показує її ефективну роботу при суттєво менших обчислювальних ресурсах, ніж ті, що наявні на сучасних КПК.

Тому проблема практично повної відсутності СКМ для WinCE розв'язувалась на основі обчислювальних потужностей КПК – шляхом запуску ПЗ, розробленого для MS DOS, під управлінням відповідного емулятора цієї операційної системи. Хоча застосовані DOS-версії Derive 1.53 та MathCAD 2.5 (рис. 2.16) і були випущені більше 15 років тому, вважати застарілими їх лише через це не варто: символьне ядро цих систем було розроблено вже давно і за останні роки суттєво не змінилося, тому як наукова, так і освітня цінність цих систем не була втрачена лише тому, що MS DOS відійшла у минуле. Використання в КПК дає друге життя цим компактним та ефективним системам.

В якості мобільної СКМ було обрано вільно поширювану систему Maxima. Наявний WinCE-порт цієї системи не розвивається з 2001 р. та не задовольняє сучасним вимогам до ергономіки інтерфейсу користувача (реалізований лише режим командного рядка з текстовим поданням результатів обчислень). Це спонукало нас до його переробки. По-перше, текстовий інтерфейс користувача був замінений на графічний шляхом перенесення на платформу WinCE інтерфейсу wxMaxima. По-друге, була виконана локалізація інтерфейсу за технологією GetText, що дало можливість вільного вибору мови інтерфейсу (російської, української, англійської). І, нарешті, була виконана оптимізація вихідних текстів Maxima з метою прискорення її роботи.



а)



б)

Рис. 2.16. СКМ Derive 1.53 (*а*) та MathCAD 2.5 (*б*) на КПК HP Jornada 720

Крім розробленої мобільної версії Maxima (рис. 2.17), при навчанні математики на платформі WinCE можна застосувати графічний аналізатор Math Xpander (рис. 2.18, а), середовище динамічної геометрії Euclid (рис. 2.18, б), СКМ Formulae 1 (рис. 2.18, в), теоретико-числовий пакет PARI-GP та інші.

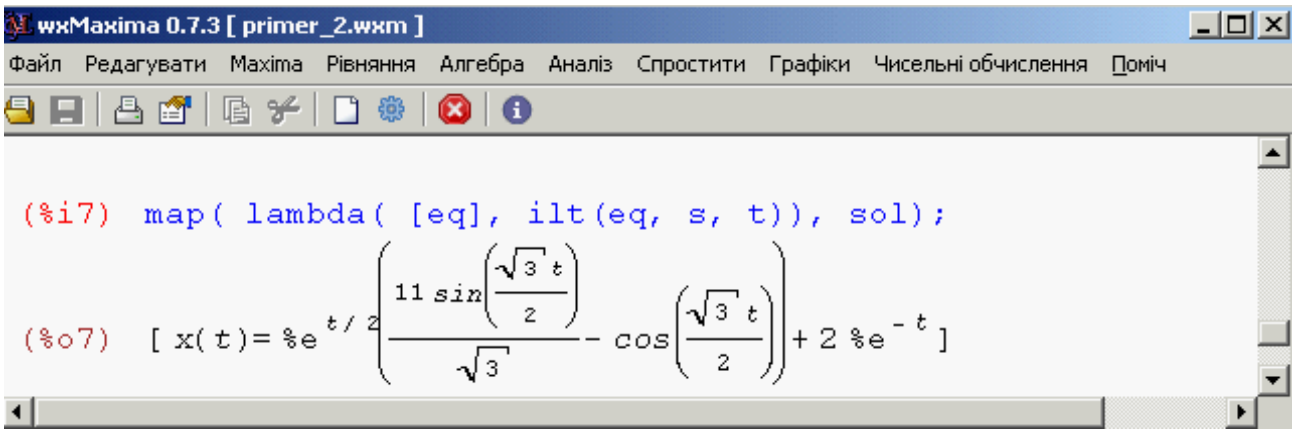
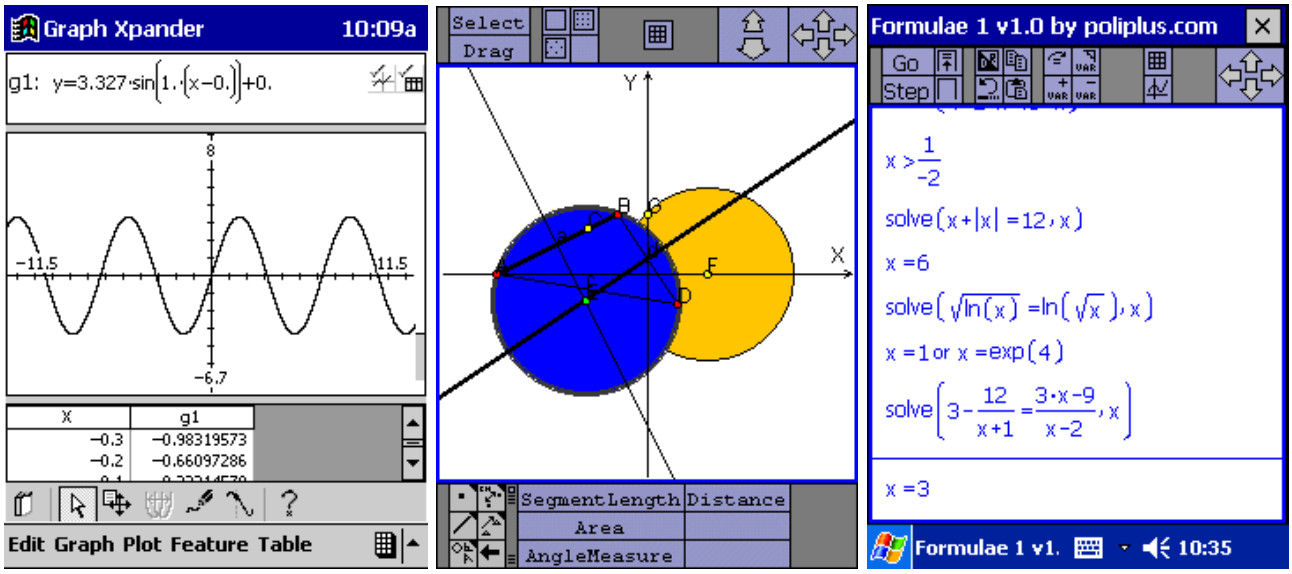


Рис. 2.17. КПК-версія СКМ Maxima 5.13



а) б) в)

Рис. 2.18. Математичне ПЗ на КПК Fujitsu Loox

2.5.3. Системи зворотного зв'язку. За даними на початок 2005 р., 78% університетів та коледжів США і Канади використовували безпровідні мережі, багато з них направляють значні ресурси на впровадження мобільного навчання [500, 46]. Наприклад, в технологічному університеті Північної Альберти (Канада) реалізується програма, за якою для 17 тисяч студентів будуть закуплені близько 12 тис. мобільних комп'ютерів і КПК HP iPAQ. Адміністрація має намір використовувати бездротовий зв'язок для студентів, що виконують великий обсяг робіт за межами університету, на відкритому повітрі (зокрема, студенти факультету лісового господарства). Студенти використовують ноутбуки для завантаження даних від своїх викладачів, записують дані під час роботи на відкритому повітрі, створюють карти та орієнтуються в лісі за допомогою програмного забезпечення, систем глобального позиціонування та географічних інформаційних систем. Викладачі використовують ноутбуки для зв'язку з Інтернетом та отримання базових даних від супутників GPS і передавання студентам, які отримують їх за допомогою Wi-Fi-доступу. Тому вони можуть витратити приблизно два тижні на місцях, навчаючись під дистанційним моніторингом викладача за ходом навчання та отриманими результатами.

Зазначимо, що для вітчизняних навчальних закладів побудова комбінованих мереж на основі провідних та безпровідних технологій сьогодні вже є більш економічно вигідним, ніж розгортання традиційних провідних мереж [105]. Врахування цієї тенденції надає можливість створити такі педагогічні технології, в яких мобільні пристрої стануть основою нової освітньої інфраструктури школи та вищого навчального закладу, а не перешкодою в навчанні. Інтеграція в навчальний процес (замість адміністративних обмежень) передбачає не лише добір відповідного ПЗ для індивідуальної роботи, а й активне використання засобів колективної роботи з виконання навчальних проектів та оцінювання навчальних досягнень.

Сьогодні, в лекційних аудиторіях роль студента залишається переважно пасивною: окремі прийоми (запитання до аудиторії, бліц-контрольні роботи тощо) не дозволяють підтримувати активність всіх студентів протягом всієї лекції. Перспективним засобом активізації навчальної діяльності є системи зворотного зв'язку (Student Response System – SRS) [486], використання яких дозволяє застосовувати комбінацію з безпровідних мереж, КПК та мультимедійного проектора для подання відповідей в процесі тестування. Прикладом такої системи є Numina SRS, що застосовується в Північно-Каролінському університеті (м. Вілмінгтон, США) при навчанні математики, фізики та хімії (рис. 2.19) [564].

В типовому сеансі роботи Numina SRS студенти використовують комунікатори, щоб відповісти на питання викладача. В SRS зберігаються їхні відповіді (у віддаленій базі даних), а узагальнені результати відображаються на мультимедійній дошці або екрані проектора. Оскільки SRS є серверними Web-додатками, то відсутня необхідність у спеціальному ПЗ на стороні клієнта – лише Web-браузер (рис. 2.20). Викладач використовує закриті тести, що розміщуються на локальному Web-ресурсі.

При застосуванні систем зворотного зв'язку на лекції:

1) майже 100% студентів беруть участь у тестуванні (фактором підвищення кількості опитуваних є непублічність та анонімність відповідей) на від-

міну від типових 2-3% студентів, суттєво знижуючи при цьому позанавчальну активність на занятті;

2) викладач одразу отримує статистику розуміння студентами лекційного матеріалу;
3) викладачі приймають обґрунтовані рішення на основі оперативних результатів зі зміни темпу та організації подання матеріалу;
4) майже 100% студентів, що працюють в SRS, надають їй перевагу перед традиційними засобами тестового контролю.

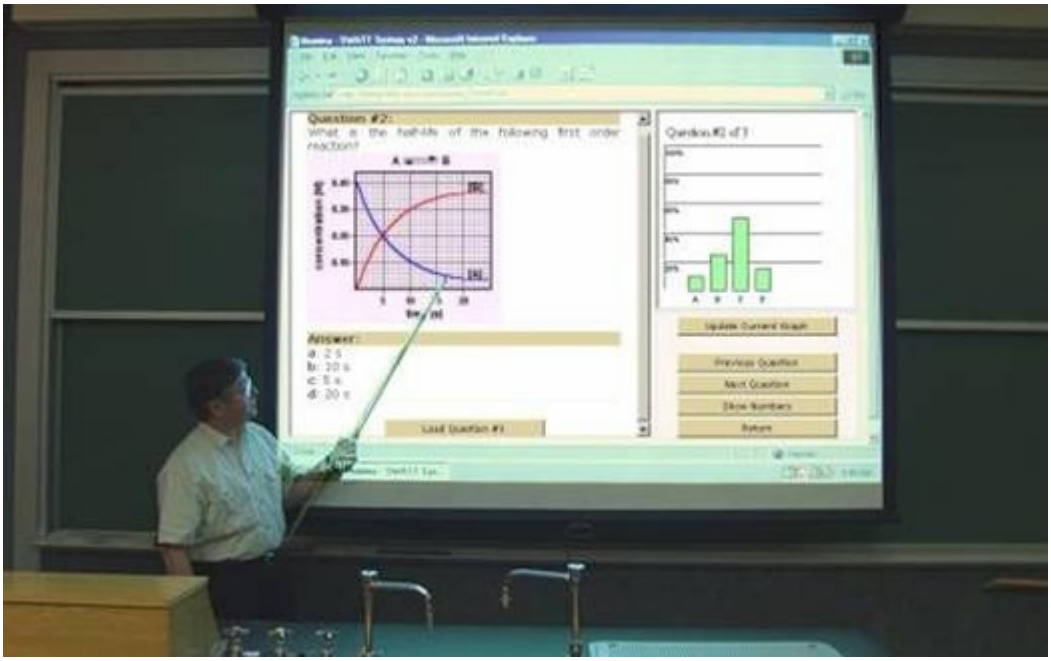


Рис. 2.19. Викладач застосовує Numina SRS у класі

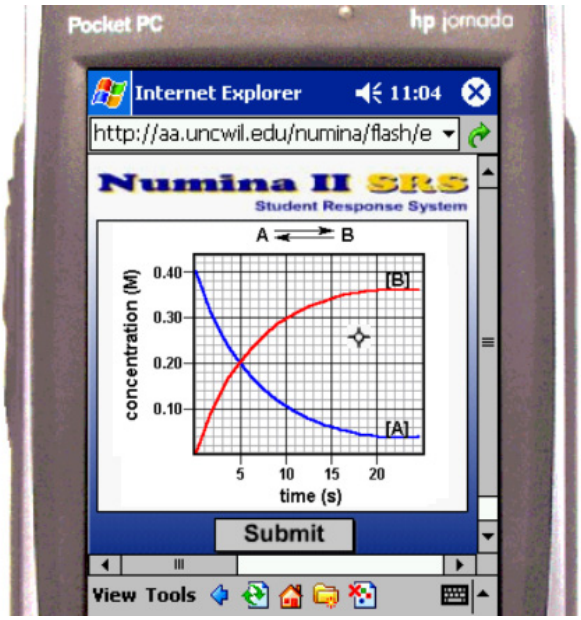


Рис. 2.20. Онлайн-тест на КПК

Системи зворотного зв'язку можуть використовуватися не лише з персональними комунікаторами, а й з більш простими мобільними пристроями, проте на комунікаторах можна виконувати Flash-додатки, що надає можливість використання мультимедійних тестів.

В Папському католицькому університеті Чилі (Pontificia Universidad Católica de Chile) розроблено проект Eduinnova (http://www.eduinnova.com) підтримки спільного навчання за допомогою мобільних технологій:

1. Викладач завантажує необхідні навчальні об'єкти на свій КПК.

2. В аудиторії він поширює ці об'єкти на студентські КПК.

3. Викладач ініціює спільну діяльність команд студентів над проектом.

4. В ході роботи викладач на своєму КПК відслідковує, оцінює та обговорює індивідуальну та групову діяльність (рис. 2.21).

5. Результати роботи збираються на КПК викладача та публікуються в Інтернет.

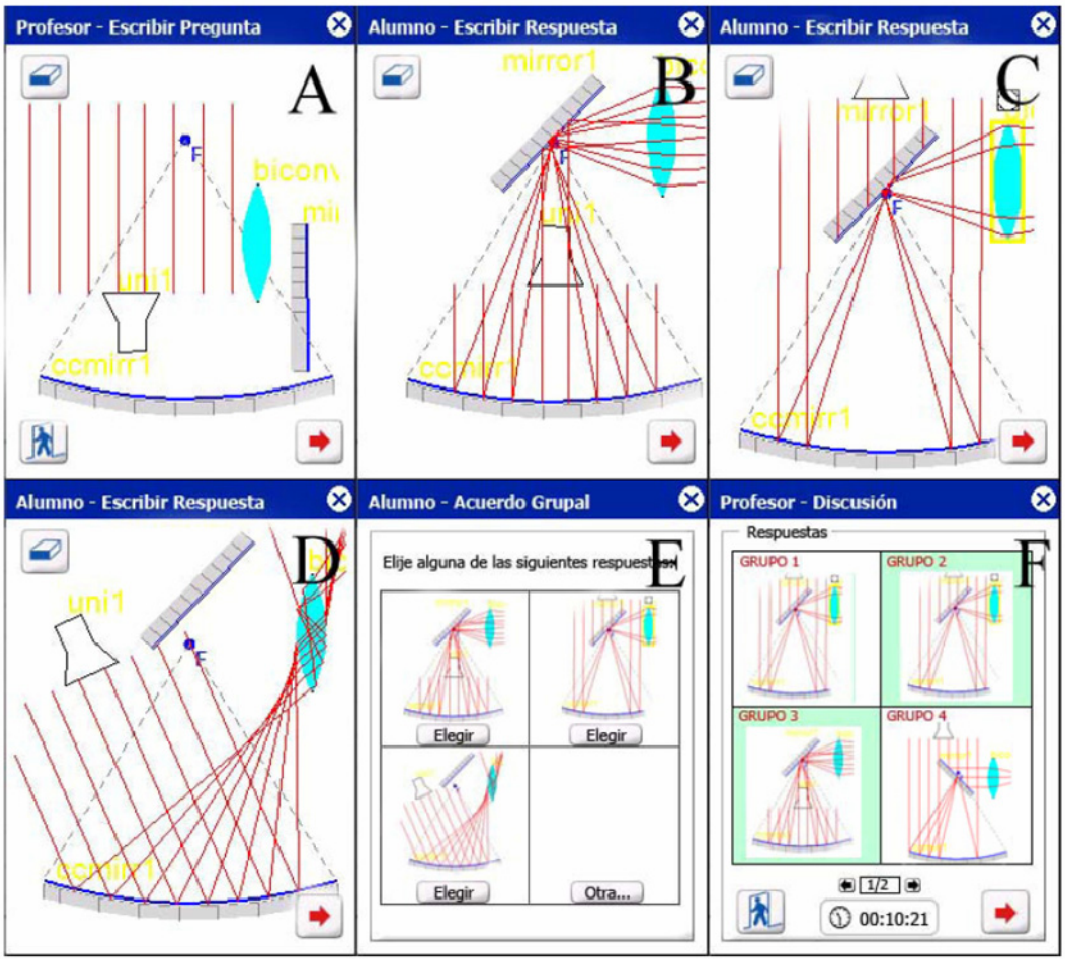


Рис. 2.21. Спільна робота в середовищі Eduinnova

SRS є гарним прикладом реалізації концепції мобільного освітнього офісу [308]. Перетворення мобільного освітнього офісу на *мобільне освітнє середовище* вимагає переходу від застосування розрізнених послуг (електронної пошти, чату, Web, FTP, Telnet) до інтегрованих середовищ навчання (Moodle, WebCT) та колективної роботи (FirstClass, NetMeeting) на основі застосування:

1) різних пристроїв та платформ, об'єднаних як провідними, так і безпровідними мережами;

2) клієнт-серверних Інтернет-технологій;

3) об'єктно-орієнтованої компонентної архітектури;

4) стандартизованих способів обміну даними;

5) відкритості та масштабованості [232] (рис. 2.22).

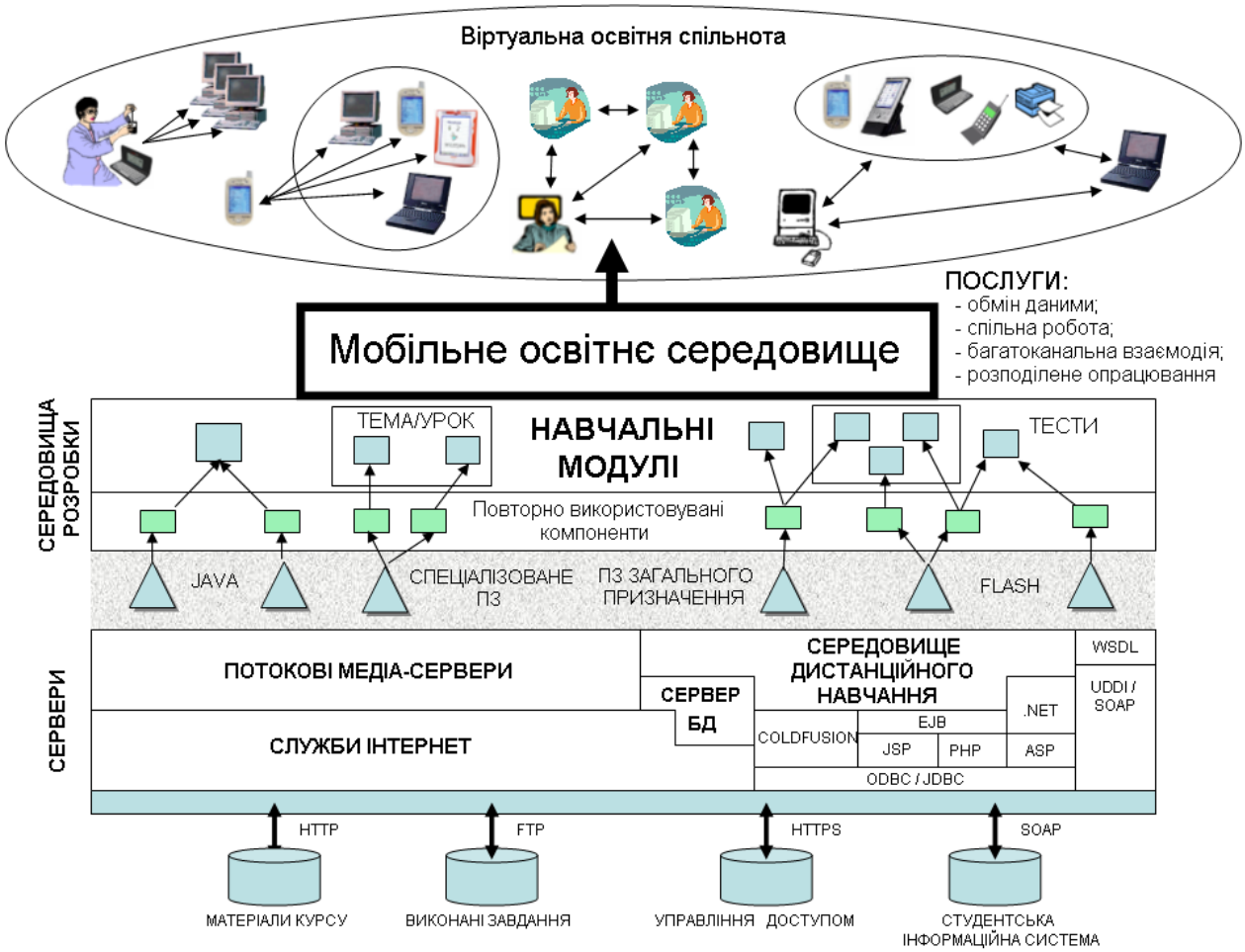


Рис. 2.22. Структура мобільного освітнього середовища

2.5.4. Застосування засобів мобільного зв'язку для підготовки до лекцій. Перехід системи вищої освіти України до навчання за кредитно-

модульною системою вимагає розробки нових підходів до таких традиційних форм навчання, як лекції. У п. 2.5.3 були розглянуті засоби активізації навчальної діяльності студентів в ході лекції за допомогою систем зворотного зв'язку, проте впровадження SRS не розв'язує питання готовності студента до лекції, тому актуальною є проблема розробки ефективних засобів для заохочення підготовки студентів до майбутньої лекції. Автори роботи [453] зазначають, що підготовка до лекцій може бути досить ефективним засобом пробудження інтересу та розширення участі студентів в слуханні і аналізові лекції, а, отже, сприятиме підвищенню їх успішності. Результати досліджень також свідчать про те, що студенти, котрі мають деякі попередні знання з поточної лекції, вчаться більш ефективно, ніж непідготовлені студенти [454].

Використання мобільних технологій надає нові можливості для студентів при підготовці до лекцій. Як зазначалося у п. 2.5.1, студенти-першокурсники біологічного факультету Норвезького університету науки і технологій використовували мобільні телефони для доступу до навчальних матеріалів через університетську LMS.

У 2007/2008 н.р. в курсі «Комп'ютерні технології в наукових дослідженнях» на фізико-математичному факультеті Криворізького державного педагогічного університету було апробовано матеріали навчального посібника [440], відповідно до яких студенти мали, використовуючи університетську LCMS MOODLE, опанувати основи роботи у новій Web-СКМ SAGE. Зміст посібника носить описовий характер і не вимагає високого ступеня узагальнення та осмислення. Однак є багато матеріалів, які в ході опанування курсу мали бути визначені, згадані і систематизовані. Візуалізація результатів досліджень та зразки діяльності мають важливе значення в цьому питанні.

Лекції та лабораторні заняття були основними компонентами навчання. Лекції були традиційними в тому сенсі, що викладач розглядав теми курсу, використовуючи різні засоби наочності. Паралельно із лекціями проводилися лабораторні роботи, де студенти мали вивчити засоби СКМ, що використовувалися для ілюстрації розглянутого теоретичного матеріалу. Студенти також мали

розв’язати завдання, розміщені у LMS, а також мали доступ до відеоматеріалів курсу.

У ході експерименту коротке відео про майбутню лекцію (від 6 до 9 хвилин) розміщувалося в університетській LMS, як правило, за один день до початку лекції. У відео викладач вказував основні теми і деякі ключові елементи, які студенти повинні вивчити до наступної лекції. Запис відео відбувався захватом з екрану комп’ютера слайдів презентації, що містили текст та графіку, та зразків роботи у Web-СКМ SAGE, з подальшим накладанням субтитрів, виносок та звуку (рис. 2.23).

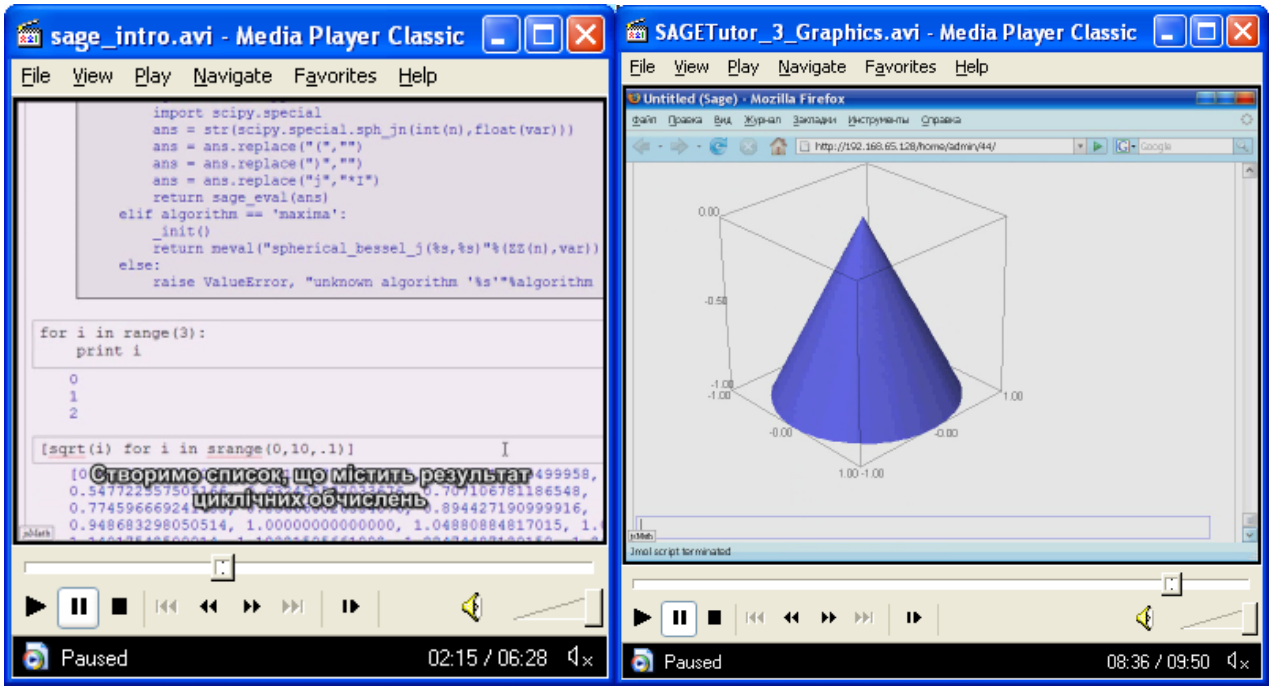


Рис. 2.23. Приклади відеоматеріалів до лекцій

Запис зберігався у контейнерах AVI (для мультимедійного проектору) та 3GP (для мобільного телефону). Відео було розміщене у LMS в трьох версіях: одна для персональних комп’ютерів і дві – для різних моделей мобільних телефонів.

Після вирішення деяких початкових технічних проблем більшість функції LCMS MOODLE стали доступні в мобільному варіанті (рис. 2.24), тому LMS-контент автоматично переформатовувався для використання на мобільному телефоні. Студенти мали можливість обирати, чи хочуть вони отримувати повід-

омлення, вирішувати завдання або переглядати відео на комп'ютері чи на мобільному телефоні.

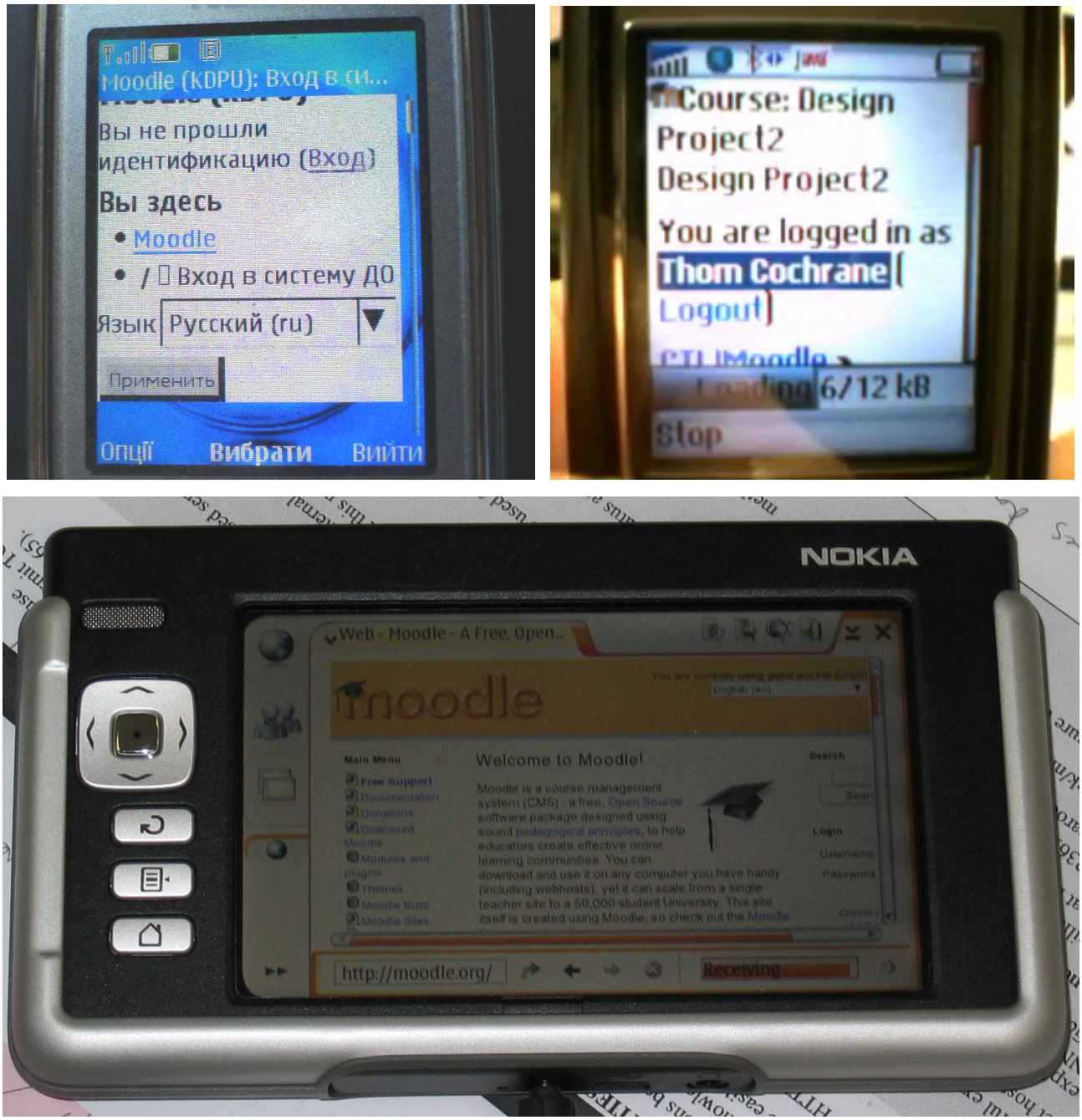


Рис. 2.24. Мобільний варіант LCMS MOODLE

Для збирання даних про те, як студенти використовували мобільні технології в процесі підготовки до майбутньої лекції, були поєднані методи спостереження та опитування. Відвідування лекцій дало з перших рук дані про те, як був проведений відеоінструктаж, і надало можливість визначити, як саме його матеріали були використані на лекції. Всього до уваги бралося 11 лекцій з 12,

передбачених програмою курсу (перша лекція була вступною). Одразу ж після лекції опитувались окремі студенти, а наприкінці семестру – кожний студент. Також хід експерименту висвітлювався і обговорювався на університетському форумі.

У ході експерименту наше припущення полягало в тому, що навчання відбувається у взаємодії між студентами та викладачами на основі використовуваних технологій. Аналіз результатів опитування показав, що студенти були в цілому задоволені новими можливостями для навчання, які забезпечувалися Інтернет-природою мережної СКМ SAGE, та можливістю її використання на мобільних телефонах. Спостереження і бесіди також свідчать про те, що студенти готувалися до лекцій і що вони використовували для цього відео. Всі студенти використовували мобільні телефони для перегляду відео, в т.ч. й в аудиторії. Деякі використовували його на регулярній основі, а інші ні. Студенти стверджували, що перегляд відео до відвідування лекції покращує їхню інформованість про майбутні проблеми і, можливо, сприяє активній участі в слуханні і аналізові матеріалу лекції. Використання мобільних засобів навчання також надавало більше варіантів стосовно того, коли й де можна готуватися до лекцій.

В ході аналізу результатів навчання були виділені три рівні використання мобільних телефонів для підтримки навчання:

I – студенти використовували мобільний телефон для перегляду відео без подальшої підготовки, тобто просто як спосіб зорієнтуватися у найближчій темі;

II – студенти використовували мобільний телефон для перегляду відео, перш ніж опановувати матеріал лекції та виконувати завдання, запропоновані викладачем;

III – студенти використовували мобільні телефони для навчання у LMS.

На першому рівні студенти використовували мобільний телефон для перегляду відео безпосередньо перед лекцією без подальшої підготовки, стверджуючи, що брак часу є фактором того, що вони використовували свій мобільний телефон у такий спосіб. Вони також переглядали відео у їдальні, в транспорті (якщо відео було попередньо завантажено) або під час перерви. Очевидно,

що тут застосування мобільного телефону надає можливість підготуватися навіть «у останню хвилину» (з урахуванням того, що студентам незвично готуватися до лекції): не дивлячись на те, що такі студенти використовують мобільний телефон для перегляду відео без подальшої підготовки, є підстави вважати, що вони приходять на лекцію з деякими попередніми знаннями про неї, адже перегляд відео дає уявлення про деякі концепції і загальні риси майбутньої лекції.

На другому рівні студенти використовували мобільні телефони для більш ґрунтовної підготовки до лекцій, витрачаючи час не лише на перегляд відео, а й на застосування посібника, інших книг, Інтернет тощо. Це був саме тип студентської активності, яку ми бажали пробудити засобами відео. Такі студенти при опитуванні зазначали, що вони виконали відеоінструкції, і це позитивно позначилося на сприйнятті лекції. Думки з приводу застосування мобільних телефонів у процесі підготовки розділилися: якщо всі студенти використовували свої мобільні телефони для перегляду відео, то отримання допоміжних навчальних матеріалів відбувалося переважно у традиційний спосіб, за допомогою комп’ютера та книг.

На третьому рівні студенти включалися власне до процесу мобільного навчання, використовуючи клієнтську частину LCMS MOODLE на мобільному телефоні. Крім перегляду відеофільмів студенти використовували мобільні телефони для виконання тестових завдань та отримання динамічних навчальних матеріалів, що з’являлися на сторінці курсу. Студенти високо оцінили можливість отримувати оновлення з курсу в будь-який час і в будь-якому місці. Під’єднання до LMS мобільних телефонів надало їм нові можливості для навчання.

Висновки до розділу 2

Теоретичний аналіз навчальних програм, підручників і навчальних посібників, монографій, дисертаційних досліджень, статей і матеріалів науково-методичних конференцій з проблем застосування сучасних мережних та мобі-

льних технологій в навчальному процесі ВНЗ, аналіз результатів навчання студентів та досвіду роботи викладачів, цілеспрямовані педагогічні спостереження, анкетування, узагальнення власного досвіду з впровадження інноваційних технологій та авторських програмних засобів дозволив визначити організаційно-педагогічні, програмно-технічні і технологічні умови реалізації мобільного навчання та зробити наступні висновки:

1. Електронне навчання є інноваційною технологією, спрямованою на професіоналізацію та підвищення мобільності тих, хто навчається, і на сучасному етапі розвитку ІКТ воно може розглядатися як технологічна основа фундаменталізації вищої освіти.

2. Удосконалення апаратних характеристик перетворило мобільні пристрої на потужні інтерактивні мультимедійні технічні засоби мобільного навчання – сучасного напряму розвитку систем дистанційного навчання із застосуванням мобільних телефонів, смартфонів, КПК та електронних книжок. Мобільне навчання – це специфічний вид навчання, в якому сам навчальний процес є географічно та ситуаційно залежним.

3. Мобільне навчання виступає одним із способів реалізації мікронавчання, надаючи можливість навчатися у будь-які малі фрагменти вільного часу, тому мобільне навчання забезпечує більшу навчальну мобільність в порівнянні з електронним навчанням.

4. В порівнянні з традиційним у мобільному навчанні забезпечується можливість моніторингу навчання в реальному часі та висока насиченість контенту, що надає можливість розглядати його не лише як засіб навчання, а й як інструмент спільної роботи, спрямованої на підвищення якості навчання.

5. До визначальних характеристик мобільного навчання відносяться:

– можливість динамічного генерування навчального матеріалу в залежності від місцезнаходження студента, типу мобільного пристрою та способу його застосування;

– розмиття границь між соціумом та навчальним закладом завдяки можливості застосування мобільних пристроїв у навчанні, коли викладач опиняєть-

ся в умовах, за яких матеріалу, що раніше циркулював у межах аудиторії, може бути протиставлений матеріал ззовні, що функціонує без контролю з його боку.

6. Впровадження елементів мобільного навчання в навчальний процес середньої та вищої школи надасть можливість уникнути негативних наслідків неконтрольованого використання мобільних пристроїв через їх активне залучення до процесу навчання замість адміністративних заборон.

7. Паралельно з використанням традиційних навчальних технологій мобільне навчання сприятиме забезпеченню якості освіти, підвищуючи гнучкість процесу навчання та задовольняючи вимоги безперервної освіти та навчання протягом усього життя. Мобільне навчання може також забезпечити поліпшення можливостей отримання освіти для осіб з особливими потребами, пропонуючи їм більшу гнучкість і вибір часу і місця навчання через доставляння контенту на їхні мобільні пристрої у відповідності до їхніх потреб.

8. Перспективними напрямами розвитку мобільного навчання є: тестування, навчальні дослідження та навчання в процесі роботи; контекстне навчання, чутливе до часу та місця; мобільні навчальні соціальні мережі; мобільні навчальні ігри; голосовий мобільний подкастинг з інтерактивним оцінюванням.

Основні результати другого розділу опубліковані в роботах [148; 252; 296; 298; 301; 302; 306; 308; 313; 325; 314; 315; 316; 317; 321; 367; 376; 390; 391; 393; 394; 426; 436].

РОЗДІЛ 3
МЕТОДИЧНІ ОСНОВИ ФУНДАМЕНТАЛІЗАЦІЇ ІНФОРМАТИЧНОЇ ОСВІТИ У ВИЩІЙ ШКОЛІ

3.1. Принципи проектування та розвитку методичної системи фундаментальної інформатичної підготовки

В методичній та науковій літературі зустрічаються різні підходи до визначення поняття «методична система». Виходячи з досліджень М.І. Жалдака [84], Н.В. Морзе [210], Ю.В. Триуса [405], було сформульовано визначення *методичної системи як сукупності взаємопов'язаних структурних та функціональних компонентів, що визначає діяльність суб'єктів навчально-виховного процесу, підпорядковану цілям виховання, освіти та навчання, зорієнтовану на запланований кінцевий результат.*

За Н.В. Кузьміною [158], структурні та функціональні компоненти методичної системи визначимо наступним чином: *структурні компоненти* – це основні базові характеристики методичної системи, *функціональні компоненти* – це стійкі базові зв'язки основних структурних компонентів, що виникають в процесі діяльності суб'єктів навчально-виховного процесу та зумовлюють розвиток методичної системи.

В якості структурних компонентів методичної системи виділимо базові: *цілі*, *зміст*, *методи*, *засоби*, *форми організації навчання* та розширені: *принципи*, *контроль*, *студента*, *викладача*, *результати навчання*.

В якості функціональних компонентів методичної системи виділимо:

– *гностичний* – включає дії, пов'язані з процесом накопичення нових знань про цілі системи та засоби їх досягнення, про станах суб'єктів педагогічної взаємодії;

– *проектувальний* – включає дії, пов'язані з перспективним плануванням задач та способів їх розв'язання;

– *конструктивний* – включає дії з добору та композиційної побудови змісту навчання;

– *комунікативний* – включає дії, пов'язані з організацією педагогічної взаємодії;

– *організаторський* – включає дії, пов'язані з реалізацією методичної системи.

Враховуючи визначену у другому розділі роль сучасних комп'ютерних технологій в процесі фундаменталізації навчання інформатики у вищій школі, розроблювана концептуальна модель методичної системи фундаментальної інформатичної підготовки повинна реалізуватися *комп'ютерно-орієнтованою методичною системою навчання*, під якою, за Ю.В. Триусом [405, 257], будемо розуміти *методичну систему навчання, яка забезпечує цілеспрямований процес формування інформатичних компетентностей суб'єкта навчання на основі широкого використання технологій електронного навчання.*

При проектуванні методичних систем навчання необхідно враховувати їх особливості: *цілісність* – залежність кожного елемента системи від його місця і функцій в системі; *структурність* – функціонування системи зумовлене не стільки особливостями її окремих елементів, скільки властивостями її структури; *взаємозалежність* системи і середовища – система формується і проявляє свої властивості в процесі взаємовпливів із середовищем; *ієрархічність* – кожний елемент системи в свою чергу може розглядатися як система, а система, що досліджується в цьому випадку, сама є елементом більш широкої системи; *множинність описів* – внаслідок принципової складності кожної системи її адекватне пізнання вимагає побудови множини різних моделей, кожна з яких описує лише певний аспект системи [405, 251–252].

Модель методичної системи фундаментальної інформатичної підготовки повинна враховувати *принципи розвитку та вдосконалення методичної системи навчання*, виділені А.М. Пишкало [337]:

– *принцип цілеспрямованості*, який полягає в тому, що результати розвитку методичної системи навчання в цілому ті її компонентів мають бути адекватні цілям навчання студентів;

– *принцип взаємозв'язності*, який полягає в тому, що при зміні компонентів методичної системи необхідно визначати впливи, які будуть викликані цими змінами, на всі інші елементи і враховувати їх;

– *принцип повноти*, полягає в тому, що при вдосконаленні методичної системи навчання необхідно повністю враховувати усі взаємозв'язки елементів системи.

Наведені принципи пов'язані зі структурними компонентами методичної системи навчання та описують її внутрішні взаємозв'язки. До зовнішніх зв'язків методичної системи навчання відносяться принципи:

– *врахування класичних філософських законів розвитку*: діалектичного синтезу, переходу кількісних змін у якісні, діалектичного протиріччя;

– *професіонального розвитку*, який полягає в тому, що розвиток методичної системи навчання повинен здійснюватися в напряму посилення професійної спрямованості усіх її компонентів;

– *фундаментальності підготовки*;

– *наступності*, який полягає в тому, що розвиток методичної системи навчання повинен базуватися на існуючій системі навчання та органічно доповнювати її.

3.2. Організаційні форми та методи навчання інформатичних дисциплін у вищій школі

Фундаменталізація інформатичної освіти впливає на всі компоненти методичної системи навчання: зміна цілей та змісту навчання природно веде до зміни технологічної складової методичної системи – методів, засобів, організаційних форм навчання. В першому розділі були визначені *цілі* навчання та напрями фундаменталізації *змісту* навчання, у другому – інноваційну *технологію мобільного навчання* як складові методичної системи фундаментального навчання інформатичних дисциплін у вищій школі.

Враховуючи, що Н.В. Морзе [205; 209] та Ю.В. Триусом [405; 407] дано докладну характеристику організаційних форм, методів та засобів навчання ін-

форматики у середній та вищій школі, коротко розглянемо ті з них, що зберігаються та набувають подальшого розвитку у методичній системі фундаментального навчання, та більш детально опишемо деякі нові.

3.2.1. Форми організації навчання – цілеспрямована, чітко організована, змістовно насичена й методично забезпечена система пізнавального та виховного спілкування, взаємодії, співпраці викладачів та студентів [155, 316].

В.Г. Кирсько розподіляє форми організації навчання на *навчально-планові* (урок, лекція, семінар, домашня робота, екзамен та ін.), *позапланові* (бригадно-лабораторні заняття, консультації, конференції, гуртки, екскурсії, заняття за поглибленими та допоміжними програмами) і *допоміжні* (групові та індивідуальні заняття, групи вирівнювання, репетиторство) [155].

Взаємодія учасників навчального процесу є основою поділу організаційних форм навчання на три групи: 1) індивідуальні заняття, у тому числі – самонавчання; 2) колективно-групові заняття; 3) індивідуально-колективні заняття.

Найпоширенішою в навчанні інформатики є *лекційно-лабораторна* форма, що витримала випробування життям і, незважаючи на критику, зберігається дотепер в усьому світі.

Характерними її ознаками є: постійний склад навчальних груп; строге визначення змісту навчання; певний розклад навчальних занять; поєднання індивідуальної й колективної форм роботи; провідна роль викладача; систематична перевірка й оцінювання знань [168].

Загальні форми організації навчання поділяються на фронтальні, колективні, групові, парні, індивідуальні, а також зі змінним складом студентів [422]. В основу поділу загальних форм навчання покладено характеристики особливостей комунікативної взаємодії як між викладачем та студентами, так і між самими студентами.

Фронтальне навчання застосовується при роботі всіх студентів над одним і тим самим змістом або при засвоєнні одного й того самого виду діяльності та передбачає роботу викладача з усією групою (потоком, підгрупою) в єдиному темпі, із спільними завданнями. Ця організаційна форма широко використову-

ється на лабораторних заняттях на початку вивчення предмету (теми) при реалізації словесного, наочного й практичного методів, а також у процесі контролю знань.

Як відзначає О.І. Бочкін, важливість використання комп'ютера проявляється в можливості негайного наслідування зразка діяльності, що демонструється викладачем [31]. Слід зазначити, що в міру засвоєння загальних способів дій робота студентів стає усе більш індивідуальною та незалежною від зовнішньої допомоги та вказівок викладача.

Колективна форма навчання відрізняється від фронтальної тим, що студентська група розглядаються як цілісний колектив зі своїми лідерами й особливостями взаємодії.

У *групових* формах навчання студенти працюють у групах, створюваних на різній основі й на різний термін. Це досить типова форма навчання інформатичних дисциплін при *роботі над проектами*, що відображає реальний поділ праці в колективі програмістів, які працюють над одним завданням.

При навчанні в складі групи в ній виникає інтенсивний обмін різноманітними повідомленнями, тому групові форми ефективні в групах з учасниками різного рівня підготовки й мотивації.

У *парному* навчанні основна взаємодія відбувається між двома студентами, котрі можуть обговорювати завдання, здійснювати взаємонавчання або взаємоконтроль. Парні форми організації навчання, так само, як і групові, відносяться до *гнучких форм*, конкретизацією яких в процесі навчання є групове та парне (екстремальне) програмування.

Парне програмування – форма розробки програмного забезпечення, за якої програма для розв'язування поставленої задачі створюється парою програмістів, котрі працюють за одним робочим місцем. Суть парного програмування полягає у наступному: один програміст працює над написанням коду, а інший сидить поряд, і спостерігає за його роботою, таким чином контролюючи його роботу, і уявляє проект в цілому. За домовленістю, вони міняються місцями.

К. Бек [19] визначає наступні переваги цієї форми організації діяльності:

– покращується трудова (навчальна) дисципліна;

– отримується якісніший код;

– якщо пари міняються досить часто, розробники знайомі з великою кількістю частин проекту, тому у випадку, якщо один розробник покине проект, його досить швидко може замінити інший (інтеграція парного навчання з колективним);

– покращується мораль розробників;

– молоді програмісти досить швидко отримують практичні знання;

– при парному програмуванні розробники швидше знайомляться один з одним і краще налагоджуються хороші взаємостосунки у колективі.

Досвід зарубіжних розробників програмного забезпечення показав, що **при парному програмуванні програмісти показують більш, ніж у двічі більшу продуктивність, в порівнянні з тим, коли вони працюють поодинці**.

Головним недоліком цієї форми К. Бек вважає необхідність узгоджувати стиль програмування, проте в процесі навчання це є, навпаки, перевагою.

За дистанційної форми організації навчання парне програмування реалізується через *віддалене парне програмування* – спосіб реалізації парного програмування, при якому обидва розробники, що складають пару, фізично знаходяться у різних місцях, і працюють за допомогою партнерського редактора реального часу, спільної розподіленої стільниці або спеціального модуля IDE для віддаленого парного програмування (Sangam, MoonEdit і т.п.).

Індивідуальна форма навчання передбачає взаємодію викладача з одним студентом. Особливого поширення ця форма набуває у розподіленій освіті [422].

В умовах комп’ютерного класу управляти індивідуальною діяльністю студентів досить складно: ситуація за кожним комп’ютером практично унікальна. Вихід полягає в тому, щоб залучити до навчання сильних студентів (у тому числі в рамках парної роботи) та, за висловом А.П. Єршова, «автоформалізувати власний педагогічний досвід». Сучасна реалізація цієї форми знайшла своє відображення в методі *учіння через навчання* [515].

В навчанні інформатики можна говорити про індивідуальне навчання при контакті з колективним знанням, що реалізується у формі «студент і комп'ютер» [424]. Працюючи один на один з комп'ютером (точніше, з навчальною програмою), студент у своєму темпі здобуває знання, сам вибирає індивідуальний маршрут вивчення навчального матеріалу в рамках заданої теми. «Радикальна відмінність цієї форми від класичної самостійної форми роботи в тім, що програма є зручним для використання «зліпком» інтелекту й досвіду її автора» [46].

Застосування ЕОМ сприяє інтеграції кращих сторін індивідуальної та фронтальної форм навчання – так, за рахунок тиражування педагогічних програмних засобів, навчальних курсів, використання ресурсів Інтернет зберігається й перевага фронтальних форм: можливість вчитися у кращих викладачів, використовувати різні джерела навчальних матеріалів. Це допомагає реалізувати одне з найважливіших завдань викладача вищої школи – розвиток у студентів самостійної пізнавальної активності.

Зовнішні форми організації навчання інформатики позначають певний вид заняття: лекція, семінар, практичне заняття, лабораторне заняття, практикум, факультативне заняття, екзамен, предметні гуртки, студентські наукові співтовариства й т.д.

Лекція – усне систематичне та послідовне подання матеріалу з певної проблеми, методу, теми, питання й т.д. У вищій школі ця форма є основною в процесі навчання і має два змісти: це і форма, і метод. Лекція завжди фронтальна. При наявності у студентів підготовлених на комп'ютері конспектів (наприклад, у вигляді гіпертексту або презентації) підсилюється самоуправління пізнавальною діяльністю, знімається острах не записати щось важливе. Студенти можуть одержати й роздруківку конспекту. При цьому, як відзначає О.І. Бочкін, оптимальна форма конспекту передбачає наявність у лівій частині сторінки тезисно поданих основних моментів, а праворуч – місце для коментарів [31].

Додаткові (*консультаційні*) форми організації навчання розраховані на окремих студентів або групу з метою заповнення пробілів у знаннях, вироблен-

ня вмінь і навичок, задоволення підвищеного інтересу до навчального предмета [234]. Так, на консультаціях можуть бути роз'яснені окремі питання, організоване повторне пояснення теми і т.п.

Для задоволення пізнавального інтересу та поглибленого вивчення предмета з окремими студентами проводяться заняття, на яких розв'язуються завдання підвищеної складності, обговорюються наукові проблеми, що виходять за рамки програми, даються рекомендації із самостійного опанування проблем, що цікавлять студентів.

Розрізняють поточні, тематичні й узагальнюючі (наприклад, при підготовці до екзаменів або заліків) консультації. Консультації найчастіше є груповими (від 5 студентів), що однак не виключає й індивідуальних консультацій.

Навіть в найпершій програмі курсу ОІОТ [265] передбачалися три основних види організаційного використання кабінету обчислювальної техніки на уроках – демонстрація, фронтальна лабораторна робота й практикум. Ці ж форми застосовуються й у вищій школі.

Демонстрація. Використовуючи демонстраційний екран (мультимедійні дошку, проектор тощо), викладач показує різні навчальні елементи змісту курсу (елементи інтерфейсу, фрагменти програм, схеми, тексти й т.п.). При цьому викладач сам працює на комп'ютері, а студенти спостерігають за його діями або відтворюють їх. У деяких випадках викладач пересилає демонстрації на студентські комп'ютери (мобільні пристрої), а студенти працюють із ними самостійно. Зростання ролі й дидактичних можливостей використання комп'ютерних демонстрацій пояснюється покращенням мультимедійних характеристик комп'ютерів. Основна дидактична функція демонстрації – повідомлення студентам нового навчального матеріалу.

Лабораторна робота (фронтальна) є основною формою роботи в комп'ютерному класі. Діяльність студентів може бути як синхронна, так і асинхронна. Нерідко відбувається швидке «розтікання» фронтальної діяльності навіть при спільному вихідному завданні. Роль викладача під час фронтальної ла-

бораторної роботи – спостереження за роботою студентів (у тому числі через мережу) та надання їм оперативної допомоги.

Дидактичне призначення використовуваних програмних засобів може бути різним: опанування нового матеріалу (наприклад, за допомогою програми навчального призначення), закріплення нового матеріалу (наприклад, за допомогою програми-тренажера), перевірка рівня засвоєння навчального матеріалу або операційних навичок (наприклад, за допомогою програм автоматизованого контролю або тестування).

Індивідуальний практикум – більш високорівнева форма роботи в порівнянні із фронтальними лабораторними роботами, що характеризується різнотипністю завдань як за рівнем складності, так і за рівнем самостійності; більшою опорою на підручники, довідковий матеріал, ресурси Інтернет тощо.

Студенти одержують індивідуальні завдання від викладача на одне, два або більше занять. Як правило, такі завдання видаються для відпрацьовування знань та вмінь, що відповідають певному розділу (темі) курсу.

Лабораторно-обчислювальний практикум (за типом «занурення») – форма, за якою передбачається інтенсивна концентрована робота студентів у комп'ютерному класі з відривом від інших занять протягом 1–2 тижнів. У ході занурення може бути опрацьований матеріал з окремого курсу або сукупності тем.

Семінари та *практичні заняття* є перехідною формою від фронтальної до індивідуальної роботи. В навчанні інформатики необхідно виробляти ряд немашинних та домашинних навичок і вмінь (наприклад, розв'язування завдань з теоретичних основ інформатики, розробка та обговорення алгоритму, моделі тощо). Практичне заняття – найбільш адекватна форма роботи для колективного осмислення того, що треба зробити або вже зроблено на комп'ютері, і чому такі результати отримані.

Важливим інтелектуальним умінням є здатність до розгорнутого прогнозу результатів, отриманих за допомогою комп'ютера на основі накопиченого

досвіду роботи з ним. Для його формування доцільно застосовувати семінарські заняття.

Студентам корисно знати, що саме зараховується як результат роботи на семінарі, адже при вивченні суспільно-гуманітарних дисциплін це є лише виступи, доповнення та участь у дискусії. На семінарах з інформатики можливі контрольовані результати:

– текст алгоритму, готовий для введення;

– таблиця виконання алгоритму, складена без застосування комп'ютера;

– проект роботи із програмою;

– відповіді на питання інструкції;

– інструкція до власної або чужої програми;

– коментарі до своєї або чужої програми;

– опис очікуваних результатів роботи з програмою.

Проектна форма навчань. В основі проектної форми лежить творча діяльність студента. Ознаками проектної форми навчання є:

– наявність організаційного етапу підготовки до проекту – самостійний вибір і розробка варіанту виконання, вибір програмних і технічних засобів, вибір джерел потрібних відомостей;

– вибір із числа учасників проекту лідера (організатора, координатора), розподіл ролей;

– наявність етапу самоекспертизи й самооцінки (рефлексії), захисту результату та оцінювання рівня виконання;

– кожна група може займатися розробкою окремого проекту або брати участь у втіленні колективного проекту.

3.2.2. Методи навчання. *Метод* (з грец. μέθοδος – «шлях через») – систематизована сукупність кроків, які треба здійснити для розв'язування певної задачі, досягнення мети [71]. *Метод навчання* – впорядковані способи взаємопов'язаної діяльності викладача та студента (їх взаємосприяння), спрямовані на досягнення цілей навчання [166, 87].

За методом навчання визначається, що і як саме студенти повинні робити з навчальним матеріалом, які властивості і зв'язки між об'єктами необхідно розкривати. Метод є центральною ланкою детермінації процесу навчання зовнішніми обставинами.

Поряд з поняттям «метод навчання» у теорії й педагогічній практиці використовуються поняття «прийом навчання», «методичний прийом». Прийнято вважати, що метод як спосіб діяльності складається із прийомів або окремих дій, спрямованих на розв'язування педагогічних завдань.

У методах навчання можна виділити змістову і формальну сторони. Змістова сторона включає такі компоненти:

1) зміст, різні моделі, аналогії, алгоритми, використання яких дає змогу засвоїти сутність навчальних предметів;

2) розумові, передусім мислительні, дії, потрібні для засвоєння змісту навчальних предметів і додаткового змісту (загальнологічні дії, а також дії, через які розкриваються принципи побудови навчального матеріалу тощо);

3) співвідношення між цілями навчання, з одного боку, та прямими і непрямими його продуктами, з іншого.

Формальна сторона методів навчання характеризується співвідношенням активності викладача та студентів, характером поєднання колективних та індивідуальних форм навчальної роботи, співвідношенням зорових та слухових форм подання навчального матеріалу, кількість і складність завдань, які стоять перед студентами, мірою допомоги, що надається їм тощо. При цьому діяльність викладача, з одного боку, обумовлена метою навчання, закономірностями засвоєння й характером навчальної діяльності студентів, а з іншого боку – вона сама обумовлює діяльність студентів, реалізацію закономірностей засвоєння й розвитку.

Оскільки *загальні методи навчання* численні й мають багато характеристик, їх можна класифікувати за кількома напрямами:

1. *За характером взаємної діяльності викладача та студентів* – система загально-дидактичних методів навчання І.Я. Лернера та М.М. Скаткіна [71]: ре-

продуктивний метод, пояснювально-ілюстративний метод, метод проблемного подання навчального матеріалу, частково-пошуковий або евристичний метод, дослідницький метод.

2. *За основними компонентами діяльності викладача* – система методів Ю.К. Бабанського [11], що включає три великі групи методів навчання: а) методи організації й здійснення навчальної діяльності (словесні, наочні, практичні репродуктивні й проблемні, індуктивні й дедуктивні, самостійної роботи та роботи під керівництвом викладача); б) методи стимулювання й мотивації навчання (методи формування інтересу: пізнавальні ігри, аналіз життєвих ситуацій, створення ситуацій успіху; методи формування обов'язковості й відповідальності в навчанні: роз'яснення суспільної й особистісної значимості навчання, пред'явлення педагогічних вимог); в) методи контролю й самоконтролю (усний і письмовий контроль, лабораторні й практичні роботи, машинний і безмашинний програмований контроль, фронтальний і диференційований, поточний і підсумковий).

Частково-дидактичні методи навчання можна класифікувати:

– за особливостями подання та характером сприймання матеріалу – система традиційних методів (Є.Я. Голант [45], І.Т. Огороднiков [222] та ін.): словесні методи (розповідь, бесіда, лекція та ін.); наочні (показ, демонстрація та ін.); практичні (лабораторні роботи, твори та ін.);

– за ступенем взаємодії викладача та студентів: подання матеріалу, бесіда, самостійна робота;

– в залежності від конкретних дидактичних завдань (Б.П. Єсіпов [79]): підготовка до сприймання, пояснення, закріплення матеріалу й т.д.;

– за принципом розчленовування або з'єднання знань: аналітичний, синтетичний, порівняльний, узагальнюючий, класифікаційний;

– за характером руху думки від незнання до знання: індуктивний, дедуктивний.

М.П. Лапчик [168], О.І. Бочкін [31] та Н.В. Морзе [205], крім загальнодидактичних та частково-дидактичних, виділяють ще *спеціальні методи на-*

вчання інформатики, до яких відносять *метод доцільно дібраних задач* та *метод демонстраційних прикладів*.

До спеціальних методів навчання інформатики відносяться обчислювальний експеримент та програмування. Це пов'язано з наступними обставинами:

1) обчислювальний експеримент є методологією інформатики як науки, тому його можна віднести до принципів (методології) наукових методів учіння [166, 91];

2) цілі навчання інформатики у вищій школі включають необхідність засвоєння як певної сукупності наукових фактів, так і методів отримання цих фактів, які використовуються в самій науці, а програмування відображає метод пізнання, що застосовується в інформатиці. При цьому під терміном «програмування» розуміється діяльність, яка у вузькому сенсі зводиться до простого кодування відомого алгоритму, а в широкому – співпадає з методологією інформатики, тобто є тотожною обчислювальному експерименту [166, 92].

Частина назв форм навчання інформатики виступають і в якості назв методів навчання: це, насамперед, лекція, метод проектів та лабораторно-обчислювальний практикум (за методом «занурення»).

Лекція як метод навчання відноситься до словесних методів. У вищій школі лекції звичайно практикуються при поданні нового досить об'ємного і складного матеріалу з використанням прийомів активізації навчально-пізнавальної діяльності студентів, у тому числі через привчання їх до конспектування лекційного матеріалу.

В ході лекції студентом сприймається навчальний матеріал, потім у свідомості відбувається його аналіз, після чого цей матеріал знову виражається словами (у вигляді конспекту лекції). Конспект є вже фіксацією продуктів мислення студента, що вимагає від нього значної розумової напруги, тому уміння слухати й конспектувати лекцію виробляється поступово.

Метод проектів, незважаючи на свої давні витоки – один з основних сучасних інноваційних методів активного навчання. В навчанні інформатики цей метод широко впроваджується в освітню практику (теоретичні основи впрова-

дження методу проектів розроблені в працях Є.С. Полат [218]). Проекти можуть бути індивідуальними й груповими, локальними та телекомунікаційними. Під навчальним телекомунікаційним проектом Є.С. Полат розуміє таку форму навчання, яка передбачає спільну навчально-розвивальну діяльність учасників, які можуть бути територіально віддаленими, для досягнення значущої для них мети (результату) узгодженими методами, що вимагають застосування засобів комп'ютерних телекомунікацій. Характерними ознаками навчальних телекомунікаційних проектів є самостійна дослідницька діяльність їх учасників, пов'язана з розв'язанням цікавої проблеми, що має на меті отримання практичного результату та спирається на більшості або на кожному своєму етапі на використання засобів комп'ютерних телекомунікацій.

Занурення відноситься до методів концентрованого навчання інформатики. Відповідно до дослідження А.О. Остапенко, існують різні моделі занурення [231]:

1. «Занурення» як модель інтенсивного навчання із застосуванням сугестивного впливу.

2. Занурення як модель тривалого заняття одним або кількома предметами:

– занурення в предмет (однопредметне занурення);

– двопредметна система занурення;

– тематичне занурення (занурення в образ);

– евристичне (метапредметне) занурення;

– занурення в порівняння (міжпредметне занурення);

– занурення в культуру («діалог культур»);

– занурення як компонент колективного способу навчання;

– виїзне занурення;

– циклова (конвеєрна) система навчання.

Найпоширеніша форма реалізації занурення в навчанні інформатики – лабораторно-обчислювальний практикум.

Відповідно до вищенаведеного, спрощена схема класифікації методів навчання інформатики може мати вигляд, показаний на рис. 3.1 [166].

Рис. 3.1. Типологія методів навчання інформатики

При виборі та поєднанні методів навчання необхідно керуватися наступними *критеріями*:

– відповідність цілям і завданням навчання, виховання й розвитку;

– відповідність змісту досліджуваного матеріалу (складність, новизна, характер, можливість наочного подання матеріалу);

– відповідність реальним навчальним можливостям студентів: рівню підготовленості (навченості, розвиненості, вихованості, ступінь володіння інформаційними й комунікаційними технологіями), особливостям групи;

– відповідність наявним технічним умовам та відведеному для навчання часу;

– відповідність ергономічним умовам (час за розкладом, наповнюваність аудиторії, тривалість роботи за комп'ютером і т.д.);

– відповідність індивідуальним особливостям і можливостям самих викладачів (риси характеру, рівень володіння тим чи іншим методом, стосунки з групою, попередній досвід, рівень психолого-педагогічної, методичної та інформаційно-технологічної підготовки).

Прикладом комбінованого методу навчання є *учіння через навчання* (з німецької *Lernen durch Lehren*), що активно пропагується Ж.-П. Мартаном [515] (Католицький університет Айхштетт-Інгольштадт, Німеччина). Це метод навчання, при якому студенти самі – за допомогою викладача – готовлять і проводять заняття (це може стосуватися і його окремих частин). Основа методу не є новою: ще у Давньому Римі існувала приказка «Docendo discimus» – «навчаючи, учимося самі». В XIX столітті ця ідея стала частиною Белл-Ланкастерської системи взаємного навчання. Широкого поширення цей метод набув завдяки заснованій в 1987 році Ж.-П. Мартаном мережі, що охоплює кілька тисяч учителів, а з 2001 року «Учіння через навчання» переживає особливий підйом у зв'язку зі шкільними реформами в Німеччині.

Для інформатичної освіти цей метод цікавий насамперед своїм кібернетичним трактуванням, згідно якого навчальні комунікації моделюються нейронною мережею. У природних нейронних мережах навчання відбувається в головному мозку, при цьому нейрони утворюють стабільні, тривалі з'єднання. В нейронних мережах продукуються знання, інтегруючись і створюючи в рамках цих взаємозв'язків нові більш ефективні з'єднання (емергенції [291]).

Як можна перенести цю модель на організацію й проведення заняття? Викладач повинен подбати про те, щоб студенти інтенсивно спілкувалися й

створювали довгострокові, пов’язані з матеріалом контакти, тобто викладач повинен піклуватися про те, щоб студенти колективно продукували знання. Це відбувається найкраще в рамках невеликих дослідницьких проектів, в тому числі – телекомунікаційних (саме тому даний метод відноситься до комбінованих).

Які необхідні умови для функціонування колективу людей? Ж.-П. Мартан вказує, що «учні повинні отримувати радість від свого завдання й відчувати, що ця колективна робота відбувається з наміром поліпшити світ» [515] (етична мотивація). Комунікації повинні бути вільними: комунікативні бар’єри мають бути усунуті (чим простіші й швидші комунікації, тим краще). Викладач повинен добре знати кожного студента для того, щоб сприяти їхній активній і продуктивній взаємодії один з одним (орієнтація на ресурси). Чим компетентніші окремі члени колективу, чим компетентніші викладачі, тим краще функціонує колектив.

Учіння розглядається як органічна продуктивність мозку й ґрунтується на погодженості молекулярних, клітинних і системних нейронних процесів у суміжних підсистемах моторики, сенсорики та асоціації. Встановлено, що:

1) учіння відбувається в контурах регулювання, які селективно стабілізуються шляхом структурного й функціонального узгодження;

2) учіння відбувається за певними правилами, що відповідають індивідуальній мотиваційній та емоційній динаміці й якими обумовлюється успіх в навчанні;

3) сенсомоторні та асоціативні контури регулювання включаються, підсилюючись, у навчальний процес, що й приводить до учіння через навчання.

Метод учіння через навчання ґрунтується на конструюванні комунікативних умінь студентів та вимагає від них відкритості, дружелюбності, концентрації, для чого, зокрема, заохочується демократична поведінка.

В процесі структурування групи комунікації стають усе інтенсивнішими, тому викладач повинен звикнути до того, щоб з кожного повідомлення відразу ж розпізнавати основне висловлення й співвідносити його з іншими повідомленнями. Він стає організатором колективного міркування й повинен обережно

направляти розумові потоки, не втручаючись занадто часто. Він не повинен випустити з уваги зміст, втручаючись насамперед у сам процес, щоб комунікація між студентами відбувалася безупинно. Як зазначає М.Ю. Кондратьєв, здатність до комунікації у колективі стає основною якістю студентів [144].

Викладач як організатор колективного самоаналізу повинен піклуватися про те, щоб він вів до однієї мети, а саме до доведення нового матеріалу до всіх студентів. На початку заняття ще панує змістова невизначеність (відсутність лінійності), проте шляхом спільної роботи, крок за кроком повинна виникнути ясність (лінійність на основі досвіду). Базою підготовки до впровадження методу викладачем може бути його діяльність як модератора форумів, де з хаотично поступаючих повідомлень конструюються знання.

Існує паралель між процесом конструювання за методом учіння через навчання і способом наповнення Інтернет-енциклопедії. Той факт, що знання за методом учіння через навчання презентуються студентами, які не мають статусу експертів, привертає увагу одногрупників. У такий спосіб всі студенти закликаються працювати над поліпшенням ще незавершеного знання. Так само і з Інтернет-енциклопедією: користувачі тільки тому готові критично працювати спільно над текстами, що вони не визнають переваг в знаннях авторів статей. Тільки через рівноправність всіх користувачів стає можливим, що наявне – можливо, спочатку дилетантське – знання буде занесене до енциклопедії. Ця нова форма конструювання знання позначає перехід до суспільства знань, у якому всі рівноправно беруть участь у колективному конструюванні знань.

Учіння через навчання ґрунтується на трьох компонентах: педагогічно-антропологічному, навчально-теоретичному та систематичному, предметно-спрямованому та змістовому.

З точки зору *педагогічно-антропологічного аспекту* учіння через навчання посилається в основному на піраміду потреб А. Маслоу. Завдання формування у інших знань повинно задовольняти потреби в надійності (за А. Маслоу – в структурі самосвідомості [182]), соціальному контакті й соціальному визнанні, а також у самореалізації й змісті (трансцендентність).

Навчально-теоретичний та систематичний аспект протипоставлений традиційному способу подання навчального матеріалу. У той час, як на занятті, центром якого є викладач, відбувається, як правило, рецептивне сприйняття навчального матеріалу, знання за методом учіння через навчання затребувані самими студентами. Виходячи з підготовленого, але ще не систематизованого на занятті матеріалу, перед студентами постає завдання здобути із цього матеріалу шляхом оцінювання, зважування та систематизації відповідні знання (Linearitaet a posteriori). Цей процес може відбуватися лише за інтенсивної комунікації.

З погляду *предметно-спрямованого та змістового аспекту* (в оригінальній методиці Ж.-П. Мартана спрямованого на навчання іноземних мов) цей метод повинен усунути віддавна існуюче протиріччя між звиканням (біхевіористичний компонент), співвідношенням матеріалу (когнітивний компонент) та аутентичною взаємодією (комунікативний компонент). У змістовому плані застосування методу вимагає, щоб навчальний матеріал став основою для міркувань. При роботі з підручником його зміст подається студентам. Якщо робота з підручником закінчена, то передбачається, що вони самі в рамках методу проектів виробляють нові знання й формують їх у всіх інших. На цьому етапі мотивація студентів дуже сильно залежить від якості змісту: вони повинні відчувати, що таке обговорення є для них професійно значущим (трансцендентне відношення: потреба в змісті).

Перед розглядом нової теми викладач розподіляє матеріал малими дозами між групами студентів (максимально три студенти); кожна група одержує окрему частину матеріалу, а також завдання повідомити цей зміст всім іншим. Студенти, котрі одержали завдання, дидактично підготовляють матеріал. Під час такої підготовки, що відбувається на занятті, викладач підтримує окремі групи, надає імпульси та поради. Швидко виявляється, що студенти без проблем справляються із цим завданням, адже вони могли спостерігати, які прийоми застосовує сам викладач.

Відразу ж потрібно звернути увагу на те, що учіння через навчання у жодному разі не повинно розумітися як фронтальне заняття, проведене студентами: вони повинні постійно відповідними засобами переконуватися, що матеріал зрозумілий тим, кому він адресований (коротко запитувати, узагальнювати, залучати до партнерської роботи). Тут викладач повинен втручатися, якщо він бачить, що комунікація не вдається або що застосовувані студентами прийоми мотивації не спрацьовують.

Переваги методу:

1. Матеріал опрацьовується інтенсивніше, а студенти виявляються істотно активнішими.

2. Студенти набувають додатково до предметних знань таких ключових умінь:

– здатність працювати в команді;

– здатність до планування;

– надійність;

– презентація й коментування;

– самосвідомість.

До недоліків методу відносять більші часові витрати (у порівнянні з іншими методами навчання).

Метод учіння через навчання знаходить своє застосування у всіх предметах. Так, у Німеччині він рекомендується як відкритий метод активізації навчально-пізнавальної діяльності, він може бути застосований і як метод підвищення кваліфікації за позааудиторною формою.

Студентам цей метод дає можливість тренувати мислення, щоб самим продукувати знання, гармонійно поєднуючи дослідження й навчання (що відповідає ідеалам університетського навчання В. фон Гумбольдта). Цей метод виявився особливо ефективним для стимулювання та обмежування традиційно багаторазової деталізації матеріалу. Учіння через навчання можна застосовувати і у великих групах (з кількістю учасників від 15 до 35). Саме така форма була використана в лекційному курсі «Операційні системи» у 2007/2008 н.р.

3.3. Мобільне програмне забезпечення навчання інформатичних дисциплін у вищій школі

Як було показано у першому розділі, фундаменталізація навчання виступає насамперед інструментом стабілізації змісту навчання засобами, адекватними предметній галузі навчання в умовах швидких темпів її розвитку. Так, у п. 1.1.3 зазначено, що стабілізація курсів інформатики досягається поширенням на методичну систему навчання інформатики властивостей *відкритих систем*: розширюваності, масштабованості, мобільності; інтероперабельності та «люб'язності».

Л.Г. Хоменко зазначає, що в останні десятиліття «центр уваги змістився ... на мобільність програмного продукту як міру уніфікації та подовження терміну його життя, можливість використання наступними поколіннями ЕОМ» [421, 395]. У першому розділі даної роботи було показано, що стабілізація програмного забезпечення разом з усталенням змісту навчання веде до фундаменталізації навчання інформатичних дисциплін у вищій школі, саме тому метою цього пункту є розгляд стабільного мобільного програмного забезпечення, що виступає технічним засобом фундаменталізованого навчання у пропонованій методичній системі.

В.О. Петрушин визначає *мобільність програмного забезпечення* як властивість, що полягає у можливості його перенесення з ЕОМ одного типу на ЕОМ іншого з низькими працевитратами [140, 328]. Властивість мобільності важлива при створенні програмного забезпечення для електронного навчання. Під *мобільністю пакета прикладних програм* будемо розуміти можливість перенесення його з одного операційного середовища в інше. Мобільність пакета прикладних програм є «найважливішою властивістю пакетів прикладних програм ... [в комп'ютерній технології навчання], оскільки сприяє спрощенню їх тиражування, супроводу, а також полегшує навчання роботі з ними (не виникає необхідності повторного навчання при зміні технічної бази комп'ютерної технології навчання)» [140, 329].

3.3.1. Мобільні операційні системи

3.3.1.1. Історія створення й поточний статус стандарту POSIX. Один із загальноприйнятих способів підвищення мобільності (кросплатформенності, портабельності) програмного забезпечення – стандартизація програмного оточення: програмних інтерфейсів, утиліт тощо [485]. На рівні системних сервісів подібне оточення описується в стандарті POSIX (Portable Operating System Interface – мобільний інтерфейс операційної системи); назва запропонована відомим фахівцем, засновником Фонду вільно поширюваного програмного забезпечення Ричардом Столменом.

Розглянемо сучасну версію стандарту POSIX у редакції 2003 р., яку можна назвати «потрійним стандартом», а саме: стандартом IEEE Std 1003.1, Технічним стандартом Open Group та міжнародним стандартом ISO/IEC 9945.

Історія створення цієї версії така. На початку 1998 р. представники трьох організацій – Комітету зі стандартів мобільних додатків IEEE, Open Group та робочої групи 15 підкомітету 22 спільного технічного комітету 1 (JTC1/SC22/WG15) Міжнародної організації зі стандартизації (ISO) – почали консультації з питання злиття й розвитку курованих ними стандартів інтерфейсів до системних сервісів: IEEE Std 1003.1, IEEE Std 1003.2, Базових специфікацій від Open Group, ISO/IEC 9945-1, ISO/IEC 9945-2. У вересні того ж року в місті Остін, штат Техас, в офісі корпорації IBM відбулося організаційне засідання групи, сформованої для досягнення поставленої мети.

Основним документом для переглянутого стандарту, перший проект якого був представлений у липні 1999 року, стали Базові специфікації від Open Group, оскільки вони включали положення стандартів IEEE і ISO/IEC. В 2001 році, після завершення підготовчої роботи, стандарт містив наступні чотири частини:

– основні означення (терміни, концепції та інтерфейси, спільні для всіх частин);

– опис прикладного програмного С-інтерфейсу до системних сервісів;

– опис інтерфейсу до системних сервісів на рівні командної мови й службових програм;

– детальне роз'яснення положень стандарту, обґрунтування ухвалених рішень.

Далі в ISO, IEEE і Open Group в 2001-2002 рр. пройшло формальне затвердження нового стандарту POSIX. Тим часом накопичувалися відносно дрібні виправлення, враховані в редакції 2003-го року.

З розвитком стандарту розширювалося й трактування терміну «POSIX». Спочатку він відносився до документа IEEE Std 1003.1-1988, де описувався прикладний програмний інтерфейс ОС класу Unix. Після стандартизації інтерфейсу на рівні командної мови й службових програм більш правильно розуміти під словом «POSIX» стандарт у цілому, позначаючи перераховані вище частини 2 і 3 через POSIX.1 і POSIX.2 відповідно до нумерації документів IEEE і ISO/IEC.

3.3.1.2. Основні ідеї стандарту POSIX. В стандарті POSIX описується множина базових системних сервісів, необхідних для функціонування прикладних програм. Доступ до них надається за допомогою інтерфейсу, специфікованого для мови С, командної мови й загальновикористовуваних службових програм.

У кожного інтерфейсу є дві сторони: та, що викликає, і та, що викликається. Стандарт POSIX орієнтований у першу чергу на ту, що викликає. Його призначення – зробити програми мобільними на рівні вихідної мови. Це значить, зокрема, що при перенесенні С-програм на іншу операційну платформу буде потрібна перекомпіляція. Про мобільність виконуваних програм чи об'єктних файлів у стандарті мова не йде – це забезпечується додатковими засобами (середовища .NET та Mono для C#, інтерпретатори байт-коду для Java, Python тощо).

Стандарт POSIX аж ніяк не обмежений рамками Unix-середовища: практично для всіх сучасних операційних системи існують розширення (наприклад, для Windows – Cygwin, MinGW, SFU і т.п.), через які надаються необхідні сер-

віси і тим самим підтримується виконання POSIX-сумісних програм. Можна стверджувати, що підтримка стандарту POSIX полегшує перенесення прикладних програм практично на будь-яку скільки-небудь поширену операційну платформу. Додаткові зусилля стосовно підвищення мобільності, прикладені на етапі розробки, безумовно, окупаються незалежністю POSIX-програм від обраної операційної системи.

Разом з визначенням інтерфейсу системних викликів, в POSIX залишається за рамками розгляду їх реалізація. Зокрема, не розрізняються системні виклики й бібліотечні функції. Не є об'єктом стандартизації засоби адміністрування, апаратні обмеження та функції, необхідні тільки адміністратору, що ще раз підкреслює спрямованість стандарту POSIX на прикладні програми, а не на операційні системи.

POSIX нейтральний стосовно системної архітектури й розрядності процесора. Це дуже важливий аспект мобільності програм.

Орієнтація на міжнародний стандарт мови С визначила не тільки стиль опису функцій, але й, певною мірою, напрям розвитку специфікацій POSIX у плані синхронізації обох стандартів. Як відомо, у затвердженій в 1999 р. редакції специфікацій мови С узаконений комплексний тип даних, що викликало відповідне поповнення POSIX-функцій.

У стандарті POSIX виконаний поділ на обов'язкові та додаткові функції, причому обов'язкове ядро зроблене, за можливості, компактним. Особлива увага приділяється способам реалізації стандартизованих функцій як в «класичному» Unix-середовищі, так і на інших операційних платформах, у мережних і розподілених конфігураціях.

Розробники нової версії стандарту POSIX дбайливо віднеслися і до його передісторії, і до передісторії Unix-систем, і, головне, до додатків, що задовольняли більше раннім версіям стандарту. Існуючі інтерфейси намагалися зберігати; у процесі розвитку дотримувався принцип зворотної сумісності; нові інтерфейси додавалися так, щоб вони не конфліктували з попередніми. Повністю уникнути внесення змін у додатки не вдалося із цілком зрозумілих причин: тре-

ба було усунути протиріччя між різними вихідними специфікаціями, а також відмовитися від підтримки традиційного варіанту мови С і перейти на його міжнародний стандарт.

3.3.1.3. Основні поняття стандарту POSIX. Стандарт POSIX у редакції 2003-го року – досить великий, багатогранний документ, де докладно розглядаються наступні категорії системних компонентів:

– засоби розробки;

– мережні засоби;

– засоби реального часу;

– потоки управління;

– математичні інтерфейси;

– пакетні сервіси;

– інтерфейсні файли;

– успадковані інтерфейси.

Саме такий (на верхньому рівні, далеко не повний) спектр послуг повинен забезпечуватися операційною системою для роботи програм.

Найважливішим є поняття відповідності стандарту POSIX. Вище вже відзначалось, що будь-який інтерфейс розглядається з двох сторін: тієї, що викликає, і тією, яку викликають. Дві сторони є й у POSIX-відповідності: відповідність реалізації (операційної системи) і програми.

В реалізації (операційній системі), що відповідає стандарту POSIX, повинні підтримуватися всі обов’язкові службові програми, функції, інтерфейсні файли із забезпеченням вказаних в стандарті вимог.

Через операційну систему можуть надаватися можливості, позначені в стандарті в якості додаткових, а також нестандартні послуги. Якщо стверджується, що системою підтримується деяке розширення, це повинно робитися неуперечливо, для всіх необхідних частин і так, як описано в стандарті.

Для мінімізації розмірів ОС і програм стандартом POSIX передбачена досить дрібна гранулярність необов’язкових засобів (усього їх сорок). З іншого

боку, виконано об’єднання взаємозалежних необов’язкових засобів у групи, що в багатьох випадках рятує від аналізу великої кількості опцій. Групи ці такі:

– шифрування;

– засоби реального часу;

– розвинені засоби реального часу;

– потоки реального часу;

– розвинені потоки реального часу;

– трасування;

– потоки;

– успадковані засоби.

Наприклад, у групу «засоби реального часу» входять засоби чотирнадцяти видів, у тому числі планування на основі пріоритетів, асинхронне уведення/висновок, семафори, таймери й т.п.

У документації на операційну систему мають бути відображені питання відповідності стандарту POSIX, описані підтримувані додаткові та нестандартні засоби.

Для програм поняття відповідності стандарту POSIX багатше нюансами. Передбачено строгу відповідність, головна ознака якої – обмеження кола використовуваних засобів межами стандарту. Розглядається й відповідність із застосуванням розширень; у цьому випадку документація на додаток повинна містити опис необхідних нестандартних засобів. Бажано, щоб використовувані розширення POSIX-засобів описувалися міжнародними чи національними стандартами. (Слід відзначити, що реалізація поняття строгої POSIX-відповідності неможлива, тому що не буває операційних систем без засобів адміністрування, а вони не описуються даним стандартом.)

Профілем називають набір опцій, що описують необов’язкову функціональність. Відповідність профілю означає відповідність стандарту POSIX і підтримку заданої функціональності. Розумним чином обрані профілі надають можливість враховувати потреби представницьких класів користувачів та додатків.

Допускається існування підпрофілів, що описують підмножини стандартних засобів. Реалізація, що відповідає підпрофілю, може функціонувати на апаратних платформах з обмеженими ресурсами або використовуватись при роботі зі специфічними програмами.

Стандарт POSIX – це існуючий багато років організм, що розвивається, у якому з кожною новою редакцією щось з'являється, а щось втрачається. Застарілими називаються засоби, які ще підтримуються різними реалізаціями, але в майбутньому вони, імовірно, зникнуть. В нових програмах вони не повинні використовуватися; для кожного з них в стандарті передбачено адекватну за функціональністю сучасну заміну.

3.3.1.4. Основні поняття операційних систем, що відповідають стандарту POSIX:

– користувач;

– файл;

– процес;

– термінал;

– хост;

– вузол мережі;

– час;

– мовно-культурне середовище.

Це первинні поняття. Їх не можна строго означити, але можна пояснити за допомогою конкретних прикладів. Для кожного з виділених понять будуть описані їх характеристики – властиві їм атрибути й застосовні до них операції.

У тексті стандарту POSIX подані наступні характеристики основних понять разом з посиланнями на атрибути й операції.

У *користувача* є ім'я й числовий ідентифікатор.

Файл – об'єкт, стосовно якого допускається читання, запис, та який має такі атрибути, як права доступу до нього та тип (звичайний файл, символьні й блокові спеціальні файли, канал, символьне посилання, сокет і каталог). В реалізації можуть підтримуватися й інші типи файлів.

Процес – адресний простір разом з виконуваними в ньому потоками управління, а також системними ресурсами, необхідні для їх виконання.

Термінал (або термінальний пристрій) – символьний спеціальний файл, що задовольняє специфікації загального термінального інтерфейсу.

Мережа – сукупність взаємозалежних вузлів.

Мовно-культурне середовище – частина оточення користувача, що залежить від мовних та культурних домовленостей.

Для роботи з великою кількістю об'єктів завжди надаються механізми групування й побудови ієрархій. Існує ієрархія файлів та каталогів, групи користувачів і процесів, підмережі й т.п.

Для написання програм, в яких використовуються сутності POSIX-сумісних систем, застосовуються командний інтерпретатор (мова shell) або компільована мова С. У першому випадку в прикладній програмі можуть бути використані службові програми (утиліти), у другому – функції. Функціональний інтерфейс операційних систем природно вважати первинним, оскільки більшість службових програм призначені для виклику тієї або іншої функції.

Основними операціями, застосовними до об'єктів операційної системи, є читання, запис і виконання. За допомогою механізму прав доступу можна вибірково дозволяти й забороняти виконання подібних операцій. Раніше в стандарті фігурувало поняття суперкористувача – адміністратора, для якого немає контролю прав доступу. В POSIX-2001 обрано більш гнучке формулювання – «той, хто має відповідні привілеї», що відображає прогрес у розвитку засобів адміністрування в реалізації операційної системи.

В POSIX-сумісних операційних системах визначені об'єкти, які можна назвати допоміжними; їх використання допомагає організувати взаємодію між основними сутностями. Особливо широкий спектр засобів обміну даними між процесами (IPC – Inter-process Communication).

Процеси виконуються в певному оточенні, частиною якого є мовно-культурне середовище (locale), утворене такими категоріями, як символи та їхні властивості, формати повідомлень, дати й часу, числові та грошові величини.

Як правило, із процесом асоційовані принаймні три файли – стандартне введення, стандартне виведення та стандартне протоколювання. Звичайно стандартне введення пов’язане із клавіатурою терміналу, а стандартне виведення та стандартне протоколювання – з екраном. З стандартного введення поступають команди й (за потребою) вхідні дані для них. На стандартне виведення надходять результати виконання команд. До стандартного протоколювання виводяться діагностичні повідомлення.

До операційних систем можуть пред’являтися вимоги, наприклад, підтримки реального часу: забезпечення необхідного сервісу протягом заданого відрізка часу.

3.3.1.5. Середовище компіляції POSIX-сумісних програм. Часто (хоча це не завжди усвідомлюється) розробка програм ведеться в крос-режимі, тобто платформа розробки (еквівалентний термін – інструментальна платформа) не збігається із платформою виконання (що зветься також цільовою платформою). На інструментальній платформі створюється середовище компіляції програм, а результат компіляції може бути перенесений для наступного виконання на цільову платформу. Саме у такий спосіб було створено програмне забезпечення для платформи lBook eReader V3, описане у п. 2.3.3.

Найважливіша частина середовища компіляції – інтерфейсні файли, що містять прототипи функцій, визначення символічних констант, макросів, типів даних, структур і т.п. Для кожної описаної в стандарті POSIX функції визначено, які інтерфейсні файли повинні бути включені для компіляції програми (найчастіше потрібен один файл).

В стандарті POSIX передбачено симетричний механізм перевірки функціональності, що надає можливість програмам в разі потреби надсилати запити на одержання доступу до певних прототипів і імен.

3.3.1.6. Мобільність POSIX-сумісних програм принципово досяжна завдяки двом основним факторам. По-перше, це наявність величезного числа стандартизованих системних сервісів, а по-друге, можливість динамічного з’ясування

характеристик цільової платформи й налаштовування програми під них. (Природно, мається на увазі мобільність у рамках, регламентованих стандартом.)

Програми, що відповідають стандарту POSIX, можуть бути одно- і багатопроцесними, з динамічною адаптацією конфігурації до властивостей цільової платформи. Стандартизовано засоби породження й завершення процесів, зміни їхніх програм, опитування та зміни різноманітних характеристик. Процеси можна призупиняти й активізувати в заданий час. За допомогою механізму сигналів можна сповіщати про події й завершувати процеси ззовні. Для їхнього групування передбачені засоби управління завданнями. Програми оснащені регуляторами для управління плануванням і пріоритетами процесів. Наявний широкий спектр засобів управління пам’яттю та обміну даними між процесами: черги повідомлень, поділювана (спільно використовувана) пам’ять, семафори. Нарешті, у межах процесу можна організувати кілька потоків управління.

Необхідний ступінь детермінізму виконання досягається завдяки засобам підтримки реального часу (до них відносяться управління дисципліною виділення процесорів, сигнали реального часу, утримання сторінок в оперативній пам’яті, точні таймери і т.п.).

Функції для роботи з файлами призначені для виконання запитів від програм на читанні й запис даних тривалого зберігання, захист таких даних від несанкціонованого доступу. Механізм блокування фрагментів файлів дозволяє забезпечити атомарність транзакцій. Наявність асинхронного введення/виведення дозволяє об’єднувати операції обміну, оптимізуючи виконання програми. У стандарті POSIX ретельно пророблені питання доступу до зовнішніх пристроїв, під’єднаних через послідовні лінії, особливо до терміналів.

Стандартизована командна мова shell – адекватний засіб для написання невеликих мобільних процедур і їх швидкого автоматизованого налагодження. Виділимо механізм конвеєрів, що надає можливість поєднувати команди в ланцюжки з фільтрацією проміжних результатів. Службові програми утворюють розвинене середовище виконання для shell-процедур. За рахунок фонового ре-

жиму можна організувати одночасне виконання кількох програм і роботу з ними за допомогою звичайного термінала навіть без багатовіконного режиму.

В POSIX стандартизується інтерфейс командного рядка. У принципі, він достатній, у міру зручний і, що важливо, виникає мінімум проблем стосовно мобільності. Імовірно, у майбутніх версіях стандарту буде регламентований і графічний інтерфейс.

Мовно-культурне середовище – одне з найважливіших понять стандарту POSIX стосовно мобільності. В програмі можна визначати потрібне їй середовище (локалізацію), що дозволяє адаптуватися до потреб користувачів.

Для багатокористувацьких систем необхідна організація взаємодії великої кількості людей. В POSIX ця проблема вирішується через регламентацію засобів безпосереднього й поштового обміну повідомленнями.

В стандарті POSIX передбачені базові засоби підтримки розробки програм (у першу чергу – для мови С), що, звичайно, не знижує потреби в спеціалізованих системах.

Для роботи з програмами надаються стандартизовані засоби для з'ясування як «крупноблочних» характеристик цільової системи (наприклад, спектру підтримуваних необов'язкових засобів), так і більш дрібних характеристик (наприклад, поточний розмір вільного дискового простору).

Проблеми мобільності програм надзвичайно складні, і було б перебільшенням стверджувати, що в стандарті POSIX-2001 вони вирішуються повністю. За його рамками залишаються такі найважливіші питання, як графіка, багатовіконний інтерфейс і цілий ряд інших. Саме тому нами був розроблений описаний у [248; 247] курс проектування додатків з графічним інтерфейсом, що надає можливість створювати мобільні мережні віконні додатки.

3.3.1.7. Підтримка стандарту POSIX у сучасних операційних системах. Повністю POSIX-сумісними (такими, що мають відповідний сертифікат сумісності) є наступні операційні системи: A/UX, BSD/OS, HP-UX, IBM AIX, INTEGRITY, IRIX, LynxOS, Mac OS X, Minix, MPE/iX, OpenSolaris, OpenVMS, QNX, RTEMS, Solaris, UnixWare, velOSity, VxWorks.

POSIX-сумісними (такими, що офіційно не сертифіковані, але відповідні стандарту) є BeOS, FreeBSD, GNU/Linux, NetBSD, Nucleus RTOS, OpenBSD, RTEMS, Sanos, SkyOS, Syllable, VSTa.

Серед перелічених систем найбільш поширеними є:

Mac OS X – POSIX-сумісна операційна система фірми Apple Inc, що базується на мікроядрі Mach та підсистемі BSD-UNIX Каліфорнійського університету в Берклі, випускається для комп'ютерів Macintosh на базі процесорів PowerPC та Intel.

Minix – вільно поширювана POSIX-сумісна мікроядерна операційна система, що поширюється за ліцензією BSD. Ендрю Таненбаум створив першу версію Minix в 1987 в якості ілюстрації до підручника [364].

QNX – комерційна POSIX-сумісна операційна система реального часу. QNX призначена в першу чергу для вбудованих систем. Вважається однією з найкращих реалізацій концепції мікроядерних операційних систем.

Linux – монолітне ядро, що використовується для створення POSIX-сумісних операційних систем. Це один із найвидатніших прикладів розробки з відкритими джерельними кодами та вільно поширюваного програмного забезпечення. Спершу розроблювалася та використовувалася індивідуальними ентузіастами на персональних комп'ютерах, з тих пір Linux завдяки підтримці таких компаній, як IBM, Sun Microsystems, Hewlett-Packard, Novell та інших набув неабиякої популярності як серверна операційна система. Linux портовано на велику кількість апаратних платформ. Тепер вона досить успішно використовується як на суперкомп'ютерах, так і на мобільних телефонах. Значна кількість спеціалізованих дистрибутивів Linux, що розробляються та підтримуються різними спільнотами, забезпечує широкий вибір програмного забезпечення.

В операційні системи сімейства *Windows* включена підсистема Microsoft POSIX (рис. 3.2), в якій реалізовано одну з перших версій стандарту POSIX. POSIX-сумісні програми функціонують як процеси-нащадки posix.exe. Обмеженість реалізації не дає можливості створювати в них потоки управління, вікна, використовувати засоби обміну даними між процесами тощо.

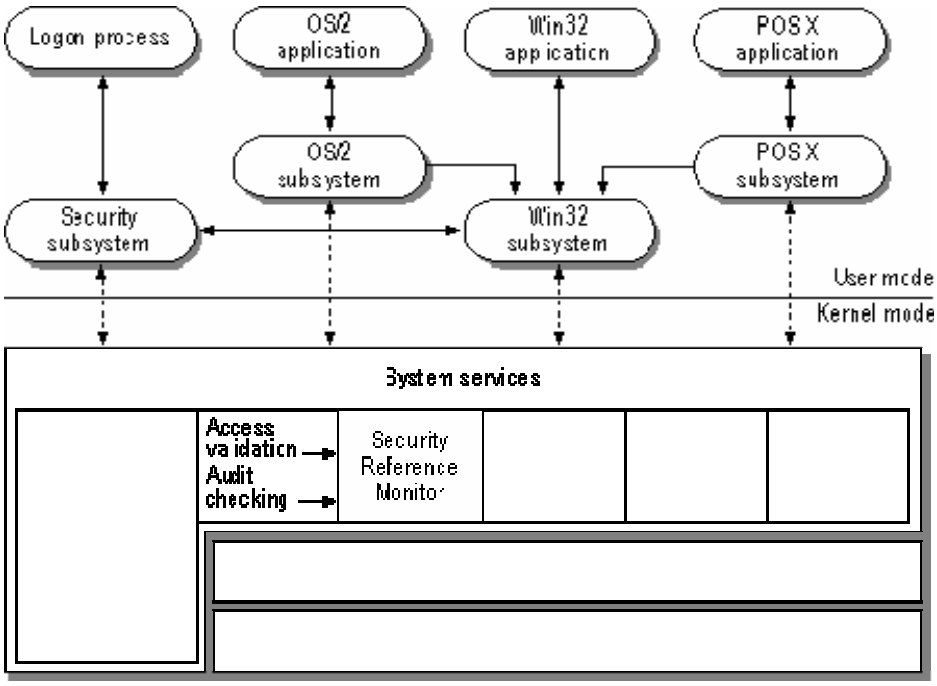


Рис. 3.2. Стандартні підсистеми виконання операційної системи Windows

Для забезпечення повної POSIX-сумісності застосовуються наступні програми:

1. *Cygwin* – набір вільно поширюваних програмних інструментів, розроблених фірмою Cygnus Solutions (входить до складу Red Hat), застосування яких надає можливість перетворення Windows на POSIX-сумісну систему. Cygwin розроблявся як середовище для переносу програм з POSIX-сумісних операційних систем у Windows. Cygwin містить бібліотеку, що реалізує інтерфейс прикладного програмування POSIX на основі системних викликів Win32. Крім того, Cygwin містить у собі інструменти розробки GNU для виконання основних завдань програмування, а також і деякі прикладні програми, еквівалентні базовим програмам UNIX. В 2001 році до складу Cygwin був включений пакет X Window System.

2. *Microsoft Windows Services for UNIX* (Сервіси Microsoft Windows для UNIX, SFU) – програмний пакет Interix, розроблений компанією Microsoft, що забезпечує POSIX-сумісність Windows (рис. 3.3). Windows Vista Enterprise і Ultimate Editions також містять елементи SFU, перейменовану в підсистему для додатків UNIX (Subsystem for UNIX-based applications, SUA).

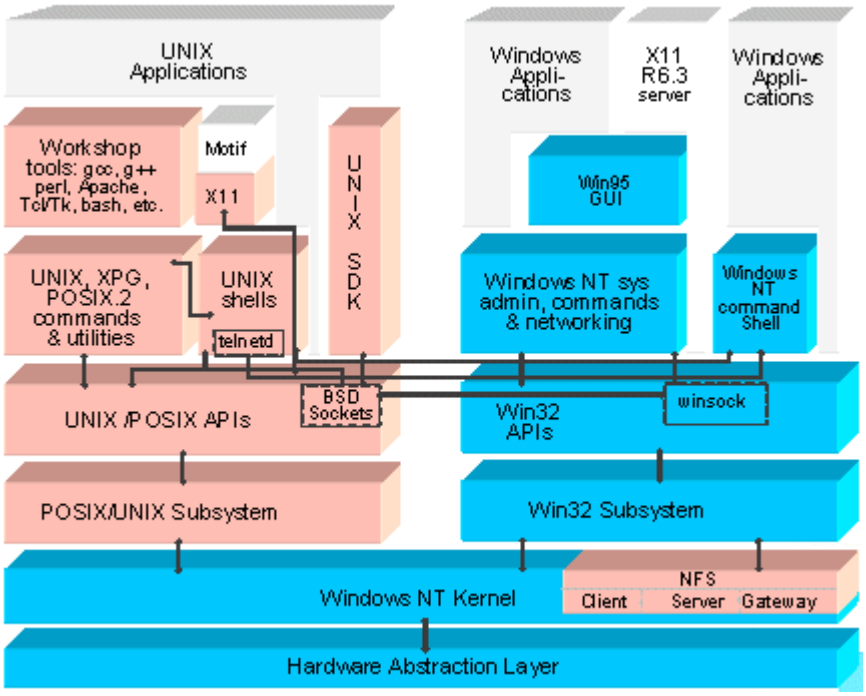


Рис. 3.3. Структура Microsoft Windows Services for UNIX

3. *MinGW* або Mingw32 (Minimalist GNU for Windows) – колекція вільно розповсюджуваних заголовних файлів і бібліотек у сполученні з набором інструментів GNU (компілятор GCC і ін.), що надають можливість створення програми Windows, в яких не використовуються сторонні динамічні бібліотеки. Спочатку MinGW створювався як відгалуження Cygwin для роботи з бібліотекою Microsoft MSVCRT (Windows API); бібліотека MinGW менш вимоглива до обсягу оперативної й дискової пам'яті, поширюється під більше вільною ліцензією й може використовуватися з будь-яким програмним забезпеченням, але функціональність специфікації POSIX реалізовані в ній не так повно, як в Cygwin.

4. *GnuWin32* – проект із портування програм за вільними ліцензіями на платформу Windows. Програми компілюються для безпосереднього виконання в Windows, не вимагаючи запуску в емуляторах Unix-оточення, таких як Cygwin або MSYS.

5. *CoLinux* (Cooperative Linux) – технологія, за допомогою якої можна запускати Linux у Windows. В CoLinux використовується модифікований Linux і спеціальний драйвер Windows для відображення системних викликів Linux у

виклики Windows. Пам’ять програми використовується як системна пам’ять операційної системи. Використовуючи цю технологію, можна запускати один або кілька екземплярів Linux у середовищі Windows без втрати швидкості (для користувача екземпляри Linux виглядають як запущені на іншому комп’ютері та доступні через мережу).

Таким чином, можна зробити висновок, що **POSIX-сумісність є засобом уніфікації операційних систем, а дотримання стандартів POSIX при розробці програмного забезпечення – засобом уникнення залежності від використовуваної операційної системи.**

3.3.2. Мобільні компілятори

3.3.2.1. GCC. Колекція компіляторів GNU (GNU Compiler Collection, GCC) – набір компіляторів для різних мов програмування. GCC – вільно поширюване програмне забезпечення, що розробляється Фондом вільного програмного забезпечення (FSF) під ліцензією GNU GPL та GNU LGPL, і є ключовою складовою набору інструментів розробки GNU (GNU development toolchain). Це стандартний набір компіляторів для POSIX-сумісних операційних систем [64].

GCC започаткований Ричардом Столменом у 1985 році як компілятор мови C (GNU C Compiler) для проекту GNU. Перша версія випущена навесні 1987 року, у 1988 році з’явилася підтримка C++. GCC був першим незалежно створеним (не базувався на препроцесорі CFront Б’ярна Страуструпа) та першим власне компілятором (а не препроцесором у C) мови C++.

У 1997 група розробників, незадоволена повільним темпом і закритістю офіційної розробки GCC, створила проект EGCS (Experimental/Enhanced GNU Compiler System – Експериментальна/Покращена збірка компіляторів GNU), в якому об’єднано кілька експериментальних відгалужень GCC. Розробка EGCS з часом виявилась більш життєздатною, ніж GCC, і у квітні 1999 року EGCS був оголошений офіційною версією GCC.

GCC сьогодні розробляється широкою групою розробників зі всього світу. Він перенесений на більшу кількість типів процесорів та операційних систем,

ніж будь-який інший компілятор. Версія GCC для Windows забезпечується проектами MinGW та Cygwin, під DOS – проектом DJGPP (лише C/C++).

У версії 4.3.3 (випущеній у січні 2009 року), у типовій збірці підтримуються наступні мови: Ada, C, C++, Fortran 95, Java, Objective-C, Objective-C++. В додаткових проектах підтримуються мови програмування Pascal, Modula-2, Modula-3, Mercury, VHDL, PL/I та D.

За допомогою GCC версії 4.3 створюється код для таких процесорних архітектур: Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC і VAX.

Менш відомі серед підтримуваних процесорів включають A29K, ARC, ETRAX CRIS, D30V, DSP16xx, FR-30, FR-V, Intel i960, IP2000, M32R, 68HC11, MCORE, MMIX, MN10200, MN10300, Motorola 88000, NS32K, ROMP, Stormy16, V850, Xtensa та AVR32.

Окремими проектами підтримуються D10V, LatticeMico32, MeP, Motorola 6809, MicroBlaze, MSP430, Nios II та Nios, PDP-10, TIGCC (варіант для m68k), Z8000 та PIC24/dsPIC.

Зовнішній інтерфейс GCC є стандартом для компіляторів на платформі Unix. Користувач викликає управляючу програму, яка називається gcc. Вона інтерпретує аргументи командного рядка, визначає і запускає для кожного вхідного файлу свої компілятори потрібної мови, запускає, якщо необхідно, асемблер і компонувальник.

Компілятор кожної мови є окремою програмою, до якої подається вхідний текст, на основі якого породжується вихідний опис програми мовою асемблера. Всі компілятори мають загальну внутрішню структуру: front end, за допомогою якого здійснюється синтаксичний аналіз і породжується абстрактне синтаксичне дерево, і back end, за допомогою якого дерево конвертується в Register Transfer Language (RTL), виконуються різні оптимізації, потім породжується опис програми мовою асемблера на основі архітектурно-залежного зіставлення зі зразком.

Головним інструментом для налагодження програм, скомпільованих за допомогою GCC, є GNU Debugger (gdb). Існують також вузькоспеціалізовані засоби для налагодження: Valgrind для пошуку втрат пам'яті (memory leaks) та GNU Profiler (gprof) для визначення, скільки часу йде на виконання тієї або іншої частини програми (так зване «профілювання програми»).

Слід зазначити, що компілятори GCC включають десятки опцій і користуватися ними напряму не зовсім зручно, тому для спрощення роботи рекомендуємо використовувати оболонки або інтегровані середовища розробки – Code::Blocks, Dev-C++, KDevelop, NetBeans, Eclipse (рис. 3.4).

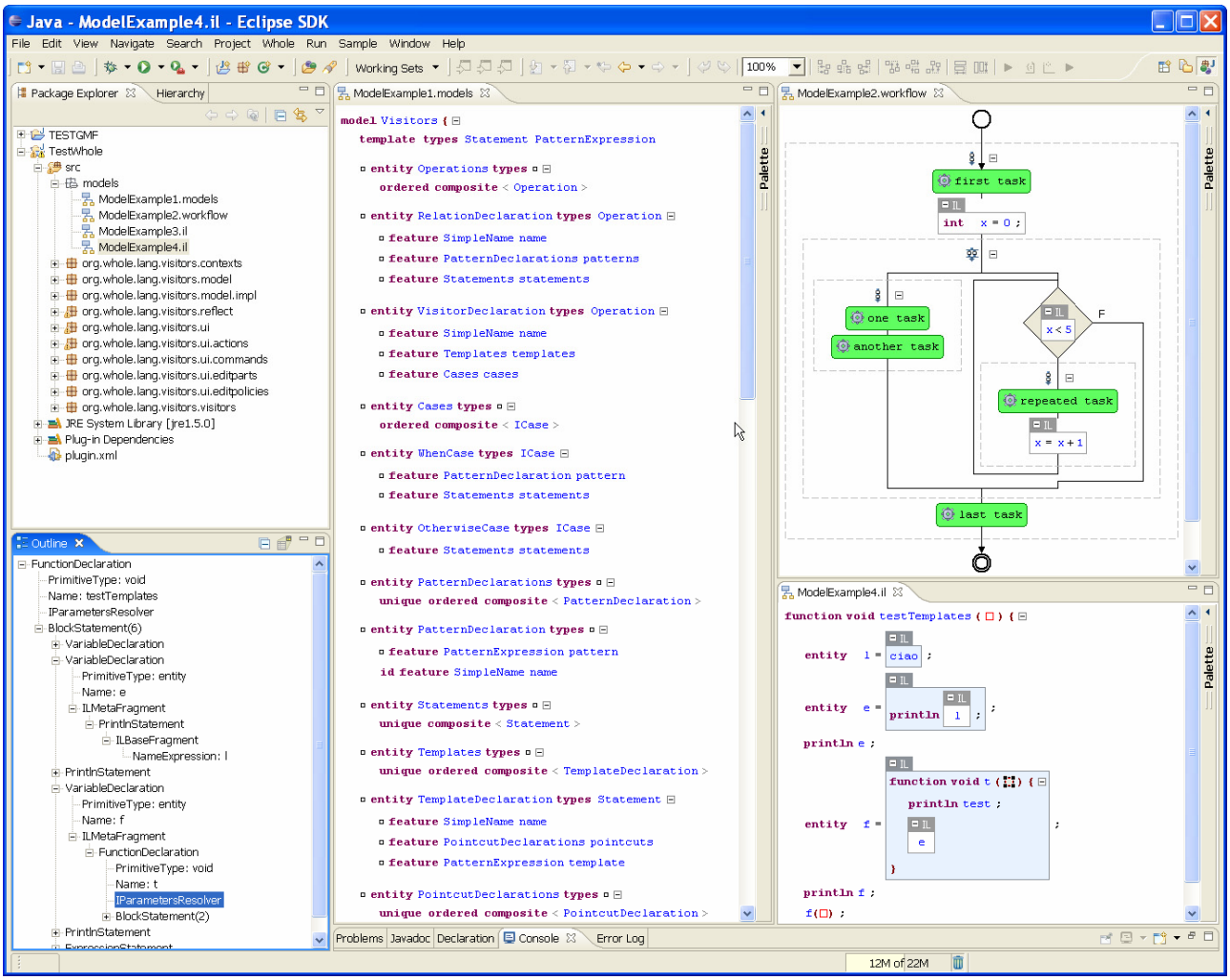


Рис. 3.4. Інтегроване середовище Eclipse

3.3.2.2. Free Pascal (повна назва Free Pascal Compiler, FPC) – це вільно поширюваний компілятор програм, описаних мовою програмування Паскаль.

FPC – кросплатформенний інструмент, призначений для різних апаратних платформ, зокрема i386, x86-64 (тільки Linux і Windows), PPC, PPC64 (тільки Linux), SPARC (тільки Linux), ARM. Важливою особливістю даного компілятора, на відміну, наприклад, від GNU Pascal, є орієнтація на поширені комерційні діалекти мови Object Pascal і Delphi, які широко застосовуються у вітчизняній системі освіти (рис. 3.5). Поряд з цим FPC включає ряд додаткових засобів, наприклад, підтримується перевизначення операторів.

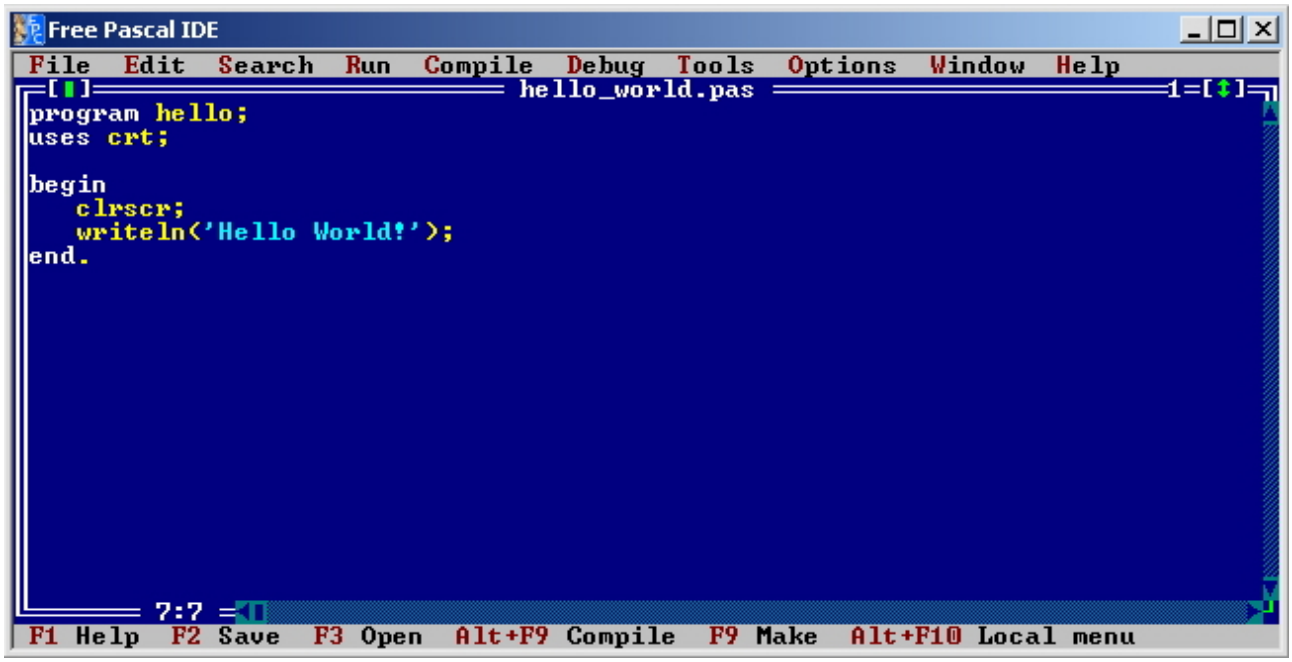


Рис. 3.5. Інтегроване середовище Free Pascal

В останні роки у рамках проекту Free Pascal також розробляється Lazarus (рис. 3.6) – вільно поширюваний аналог середовища розробки Delphi і Lazarus Components Library (LCL) – вільно поширювана бібліотека візуальних компонентів, аналогічна VCL у Delphi.

У Free Pascal підтримується компіляція в кількох режимах, якими забезпечується сумісність із різними діалектами й реалізаціями мови:

TP	режим сумісності з Turbo Pascal: сумісність практично повна, за винятком кількох моментів, пов'язаних з тим, що за допомогою FPC компілюються програми для захищеного режиму процесора, в якому неможливо пряме звертання до пам'яті, портів і т.д.

FPC	власний діалект: відповідає попередньому, розширеному додатковими засобами, такими як, наприклад, перевизначення операторів
DELPHI	режим сумісності з Borland Delphi: включає підтримку класів і інтерфейсів [114]
OBJFPC	поєднано об’єктно-орієнтовані засоби Delphi і власні розширення мови
MACPAS	режим сумісності з Mac Pascal
GNU	режим часткової сумісності з GNU Pascal

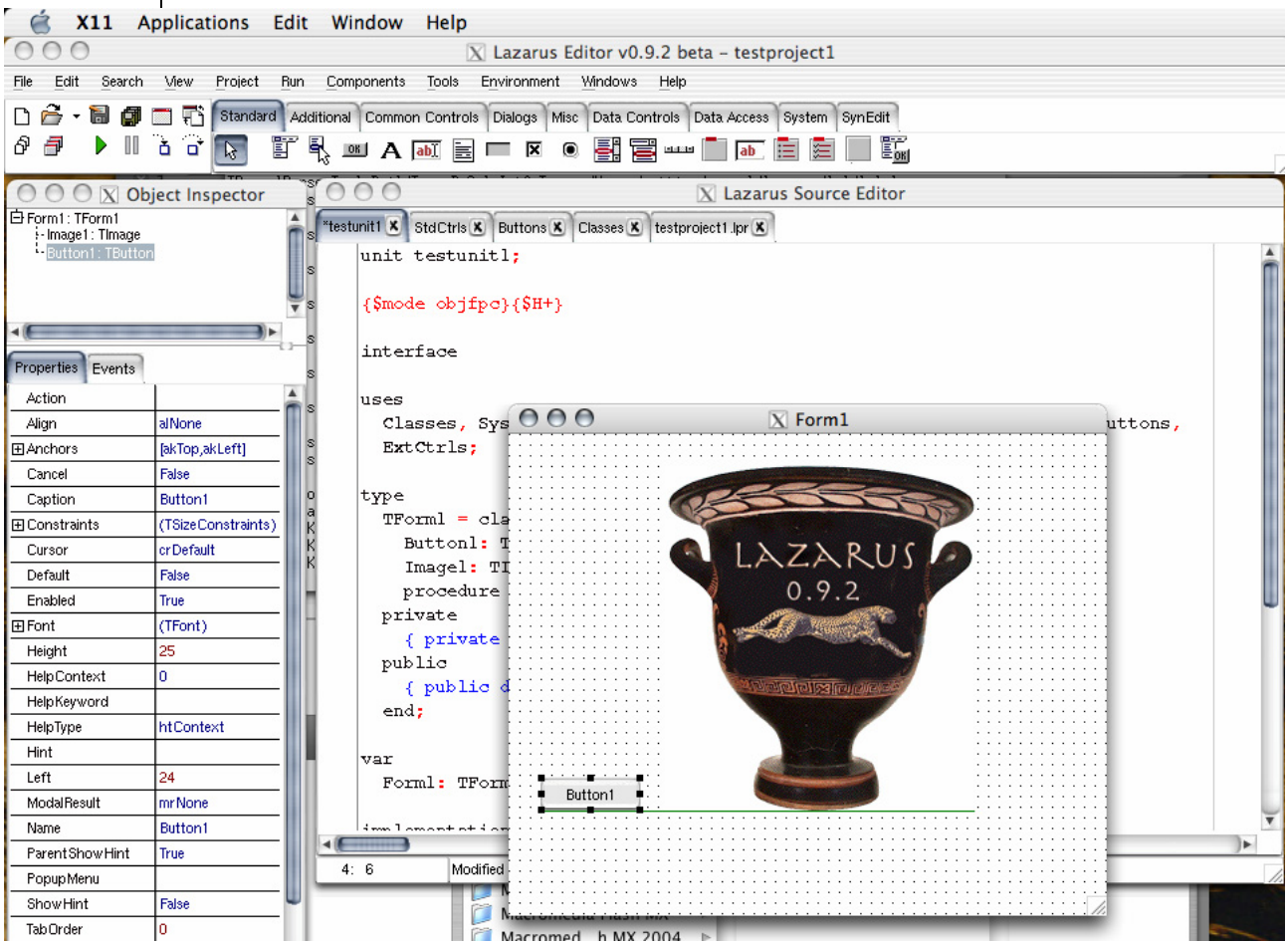


Рис. 3.6. Інтегроване середовище Lazarus

Застосування Free Pascal надало можливість, з одного боку, підтримати існуючі методики навчання на основі Borland Pascal 7 та Kylix/Delphi, а з іншого – розширити сферу застосування компіляторів Pascal в навчанні POSIX-сумісних операційних систем [248; 249].

Завершуючи огляд, можна зробити висновок, що **застосування мобільних компіляторів є засобом уникнення залежності від використовуваного середовища програмування.**

3.3.3. Мобільні інтерпретовані мови програмування. Як було показано вище, застосування мобільних компіляторів виступає засобом стабілізації таких компільованих мов програмування, як C, C++, Java, Pascal та ін. У той же час слід зазначити, що у вищій школі в навчанні інформатичних дисциплін все більшого застосування набувають *мобільні інтерпретовані мови загального призначення*. Яскравим прикладом такої мови є мова Python, застосування якої надало можливість створити незалежні від використовуваної операційної системи курси комп'ютерного моделювання [177; 369] та системного програмування [256].

Python – інтерпретована об'єктно-орієнтована мова програмування високого рівня з динамічною семантикою, розроблена Гвідо ван Россумом в 1990 р. [538]. Структури даних високого рівня разом із динамічною семантикою та динамічним зв'язуванням роблять її привабливою для швидкої розробки програм, а також в якості засобу інтеграції існуючих компонент (саме це надало можливість створити новий графічний інтерфейс до системи комп'ютерної математики Maxima [442]). В Python підтримуються модулі та пакети модулів, що сприяє модульності та повторному використанню коду. Інтерпретатор Python та стандартні бібліотеки доступні на всіх основних платформах, тобто сам інтерпретатор є мобільним. В Python підтримується кілька парадигм програмування, зокрема, об'єктно-орієнтована, процедурна, функціональна та аспектно-орієнтована.

Серед основних *переваг* Python можна назвати такі [23; 344]:

– «чистий» синтаксис (для виділення блоків використовуються відступи);

– мобільність програм (внаслідок інтерпретованої природи мови);

– стандартний дистрибутив має велику кількість корисних модулів (включаючи модулі для розробки графічного інтерфейсу);

– можливість використання в діалоговому режимі (корисне для експериментування та розв’язування простих задач);

– стандартний дистрибутив має просте, але разом із тим досить потужне середовище розробки;

– зручність для розв’язування математичних задач (містить засоби роботи з комплексними числами, цілими числами довільної довжини, у режимі безпосереднього введення може використовуватись як потужний символьний калькулятор [173]).

До *недоліків* Python відносять [359]:

– порівняно низьку швидкодію (внаслідок інтерпретованої природи мови; поступово усувається в нових реалізаціях інтерпретатора);

– відсутність статичної типізації (внаслідок того, що будь-який метод класу є віртуальною функцією);

– неможливість модифікації вбудованих класів чисел, рядків та списків (з метою збільшення швидкодії);

– глобальне блокування інтерпретатора (локальна проблема окремих реалізацій);

– неузгодженість операторів порівняння (при порівнянні необхідно зводити порівнювані величини до одного типу).

Елегантний синтаксис Python, динамічне опрацювання типів, ефективні структури даних високого рівня, простий, але ефективний підхід до об’єктно-орієнтованого програмування, а також те, що це інтерпретована мова, роблять її ідеальною для швидкої розробки та прототипування програм.

Програма мовою Python може бути розширена функціями та типами даних, створеними іншими мовами (C, C++, Pascal тощо).

Python портований на всі відомі платформи – від КПК до мейнфреймів. Існують порти під Windows, всі варіанти UNIX (включно з Linux), Plan 9, Mac OS і Mac OS X, Palm OS, OS/2, Amiga, AS/400 і навіть OS/390 і Symbian [178]. Таким чином, **Python може виступати засобом забезпечення мобільності програм, створених на POSIX-несумісних платформах.**

При цьому, на відміну від багатьох портованих систем, на кожній платформі в реалізації Python підтримуються характерні для даної платформи технології (наприклад, Microsoft COM). Крім того, існує спеціальна версія Python для віртуальної машини Java – Jython, що надає інтерпретатору можливість виконуватися на будь-якій системі, де підтримується Java, класи Java можуть безпосередньо використовуватися з Python і навіть бути написаними на Python. Нещодавно почалася розробка системи, направленої на більш повну інтеграцію з платформою .NET – Iron Python.

Програми, написані на Python, легко читаються. Мова має чіткий і послідовний синтаксис, продуману модульність і масштабованість. Однією з цікавих синтаксичних особливостей мови є виділення блоків програми з допомогою відступів (пропусків чи табуляцій), у Python відсутні операторні дужки (begin/end або фігурні дужки, як у Pascal та C. Враховуючи, що у більшості стилів форматування операторні дужки займають цілий рядок, цей «трюк» надає можливість помітно зменшити кількість рядків програми:

Програма мовою C	*Програма мовою Python*

`int factorial(int x) {`
`  if (x == 0) {`
`  return 1;`
`  } else {`
`  return x * factorial(x-1);`
`  }`
`}`
	`def factorial(x):`
	`  if x == 0:`
	`  return 1`
	`  else:`
	`  return x * factorial(x-1)`

Отже, виконання і навіть коректність програми може залежати від початкових відступів у тексті. Деякі критики мови вважають такі особливості не інтуїтивними, проте досвід застосування Python в якості другої мови програмування показав, що такий підхід виробляє у студентів навички структурування програмного коду.

3.3.4. Відкриті математичні системи

3.3.4.1. Maxima – це відома алгебраїчна система, розробка якої почалася в Массачусетському технологічному інституті (MTI) в 60-х роках минулого сто-

ліття у рамках проекту MAC. Спочатку дослідження символьного й алгебраїчного опрацювання математичних виразів (symbolic and algebraic computing, SAC) було пов’язане зі штучним інтелектом, однак незабаром вони перетворилася на окрему самостійну галузь досліджень, що відноситься зараз більше до математики, ніж до штучного інтелекту.

Maxima – одна з програм для виконання математичних обчислень, символьних перетворень, а також для побудови різноманітних графіків. Складні обчислення оформляються у вигляді окремих процедур, що можуть потім використовуватися при розв’язуванні інших задач. Система поширюється під ліцензією GPL і доступна як користувачам ОС Linux, так і користувачам Windows.

Застосування Maxima надає можливість розв’язувати велику кількість достатньо складних задач, не вдаючись у тонкощі програмування. Завдяки цьому програма одержала широке поширення у фізиці, біології, економіці тощо.

Автоматичне виконання аналітичних перетворень – одна з головних переваг цієї програми. За допомогою Maxima можна перетворювати і спрощувати алгебраїчні вирази, диференціювати, обчислювати визначені і невизначені інтеграли, обчислювати скінченні і нескінченні суми і добутки, розв’язувати алгебраїчні і диференціальні рівняння та їх системи, а також розкладати функції в ряди, знаходити границі. Крім того, Maxima має стандартні доповнення для аналітичних розрахунків.

Для розв’язування задач, які неможливо розв’язати аналітично, до системи Maxima включно велику кількість ефективних алгоритмів для проведення чисельних розрахунків. За допомогою Maxima можна розв’язувати задачі оптимізації (лінійного програмування, знаходження екстремумів функцій), а також задачі математичної статистики. У Maxima реалізовано адаптивний контроль точності, заснований на виборі внутрішніх алгоритмів, що дозволяє її максимізувати.

В пакет вбудовано довідкову систему з прикладами використання тих чи інших функцій.

Система настільки гнучка й універсальна, що її використання може надати неоціненну допомогу в розв'язуванні математичних задач як школяреві, котрий осягає основи математики, так і майбутньому науковцеві, котрий використовує математичні методи для розв'язування різних прикладних задач.

Історія розробки Maxima поділяється на три періоди: науково-дослідний проект у MTI, проект під керівництвом Вільяма Шелтера і поточний проект Maxima.

MACSYMA (Проект Mac's SYmbolic MAnipulation System) була розроблена групою Mathlab у лабораторії комп'ютерних наук MTI (спочатку відомої як Проект MAC) у 1969-1972 р. Ця робота була підтримана грантами NSG 1323 NASA, N00014-77-C-0641 Дослідницького агентства ВМС, ET-78-C-02-4687 Міністерства енергетики США і F49620-79-C-020 ВПС США. Macsyma була потім модифікована для використання під операційною системою UNIX (на комп'ютерах DEC VAX і робочих станціях Sun) Ричардом Фейтманом і його колегами з Каліфорнійського університету (Берклі); ця версія Macsyma відома як VAXIMA.

Ліцензування в 70-ті рр. XX ст. програмних кодів Macsyma призвело до створення інших систем комп'ютерної математики – Maple фірми Waterloo Maple Inc. та Mathematica фірми Wolfram Research. Спільність цих програмних продуктів виражається як у схожому синтаксисі (табл. 3.1), так і в спільних алгоритмах.

Таблиця 3.1

Порівняння команд Maxima, Maple та Mathematica

	Maxima	*Maple*	*Mathematica*
границя	`limit(x-7,x,3);`	`limit(x-7,x=3);`	`Limit[x-7,x->3]`
розгортка виразу	`expand((a+b)^3);`	`expand((a+b)^3);`	`Expand[(a+b)^3]`
розклад на множники	`factor(%);` `ezgcd(num, denom);`	`factor(%); normal(%);`	`Factor[%]`
розв'язуван-	`solve(a*x^2=4,x);`	`solve(a*x^2=4,x);`	`Solve[a x^2==4,x]`

	Maxima	*Maple*	*Mathematica*
ня рівнянь			
3D-графіка	`plot3d(sin(x*y),[x,-2,2],[y,-1,1]);`	`plot3d(sin(x*y),x=-2..2,y=-1..1);`	`Plot3D[Sin[x y],{x,-2,2},{y,-1,1}]`
параметри відображення	`set_plot_option([plot_format,gnuplot]);`	`plotsetup(x11);`	`Display["math.eps", %, "EPS"]`
оточення	`plot_options;`	`plotsetup();`	`$DisplayFunction`
інтегрування	`integrate(x^2*sin(alpha*x),x,0,beta);`	`int(x^2*sin(alpha*x), x=0..beta);`	`Integrate [x^2 Sin[alpha x], {x,0,beta}]`
розклад цілого числа на множники		`ifactor(%);`	`FactorInteger(%)`
квадратний корінь	`sqrt(3);`	`sqrt(3);`	`Sqrt[3]`
числове значення	`ev(%,numer);`	`evalf(%);`	`N[%,10]`
підстановка	`ev(%,x=1,y=2);` або `at(%,[x=1,y=2]);`	`eval(%,[x=1,y=2]);`	`ReplaceAll[%,{x->1,y->2}]`
часткова сума ряду	`sum((1+i)/(1+i^4), i,1,10);`	`sum((1+i)/(1+i^4),i=1..10);`	`Sum[(1+i)/(1+i^4),{i, 1,10}]`
відкладене обчислення часткової суми ряду	`'sum((1+i)/(1+i^4),i,1,10);`	`Sum((1+i)/(1+i^4),i=1..10);`	`:=`
частковий добуток ряду	`product((i^2+3*i-11)/(i+3),i,0,10);`	`product((i^2+3*i-11)/(i+3),i=0..10);`	`Product[(i^2+3*i-11)/(i+3),{i,0,10}]`
відкладене обчислення	`'product((i^2+3*i-11)/(i+3),i,0,10);`	`Product((i^2+3*i-11)/(i+3),i=0..10);`	`:=`

	Maxima	*Maple*	*Mathematica*
часткового добутку ряду			
нескінченність	`INF`	`infinity`	`Infinity`
комплексні числа	`%I`	`I`	`I`
	`rectform(%);`	`convert(%,rect);`	
	`polarform(%);`	`convert(%,polar);`	
	`realpart(%);`	`Re(%);`	`Re[%]`
	`imagpart(%);`	`Im(%);`	`Im[%]`
	`abs(%);`	`abs(%);`	`Abs[%]`
	`carg(%);`	`argument(%);`	`Arg[%]`
тригонометрична форма	`trigsimp(%);` `trigrat(%);`	`simplify(%);`	`Simplify[%],` `TrigSimp[%],` `TrigReduce[%]`
	`trigexpand(%);`		`TrigExpand[%]`
	`trigrat(%);`		`TrigFactor[%]`
визначення функції	`f(x):=x^2+1/2;` або `define(f(x),x^2+1/2);`	`f:=x->x^2+1/2;` або `f:=unapply(x^2+1/2,x);`	`f[x_]=x^2+1/2` або `f=Function[x,x^2+1/2]`
похідна	`diff(f(x),x,2);`	`diff(f(x),x&2);`	`D[f[x],{x,2}]`
відкладене обчислення похідної	`'diff(f(x),x,2);`	`Diff(f(x),x&2);`	`:=`
масиви	`array([x, y], 300);`	`x := array(1..300);`	`Array[x, 300]`
поділ на складові вирази	`pickapart(%, 4);`	`addressof(%);`	`FullForm[%];`

	Maxima	*Maple*	*Mathematica*
		`disassemble(%);`	`[[3]]`
		`pointto(a[2]);`	`ReplacePart[Out[32], Expand[Out[54]], 1]`
		`rhs(solutions[2]);`	
кінець команди	`;` або `$`	`;`	новий рядок або `;`
діапазон	`[x,-2,2]`	`x=-2..2`	`{x,-2,2}`
множення	`*`	`*`	пропуск або `*`
результат останньої операції	`%`	`%`	`%`
надання значення	`:`	`:=`	`=`
перевірка на рівність	`=`	`=`	`==`

Таким чином, Macsyma фактично стала родоначальником всього напряму програм символьної математики.

«Академічність», неінтуїтивний інтерфейс користувача Macsyma у 80-ті роки XX ст. суттєво звузили сферу її використання, до того ж лобіювання інтересів інших фірм, що виробляли подібні програмні продукти, призвели до фактичної зупинки роботи над нею.

Новий етап у розвитку Maxima настав у 1999 році, коли минув термін дії патенту, і права на Maxima повернулись до одного з її авторів – Вільяма Шелтера, який виконав повну переробку системи та залучив до її відкритої розробки провідних спеціалістів.

Вільям Шелтер розробляв і підтримував цю версію Maxima із самого початку проекту до своєї передчасної кончини в 2001 році. З листопада 2001 року проект Maxima підтримується роботою команди на чолі з Джеймсом Амундсоном і Ричардом Фейтманом.

Сьогодні проект продовжує активно розвиватися, і участь в ньому є кращою візитною карткою для математиків та програмістів з усього світу. Завдяки зусиллям інтернаціональної команди розробників Maxima набула ряд особливостей, що дозволяють використовувати її у вітчизняній системі освіті: система повністю відкрита, ліцензійно чиста і безкоштовна, незалежна від використовуваної операційної системи й апаратної платформи; невелика за розміром, невимоглива до апаратних ресурсів; надає користувачеві широкий вибір інтерфейсів.

Система комп'ютерної математики Maxima адаптована до потреб різних категорій користувачів. Текстове ядро системи (рис. 3.7) надає можливість побудувати такі спеціалізовані інтерфейси за допомогою технології «програмних обгорток» [141; 302; 316; 301; 310; 317; 442].

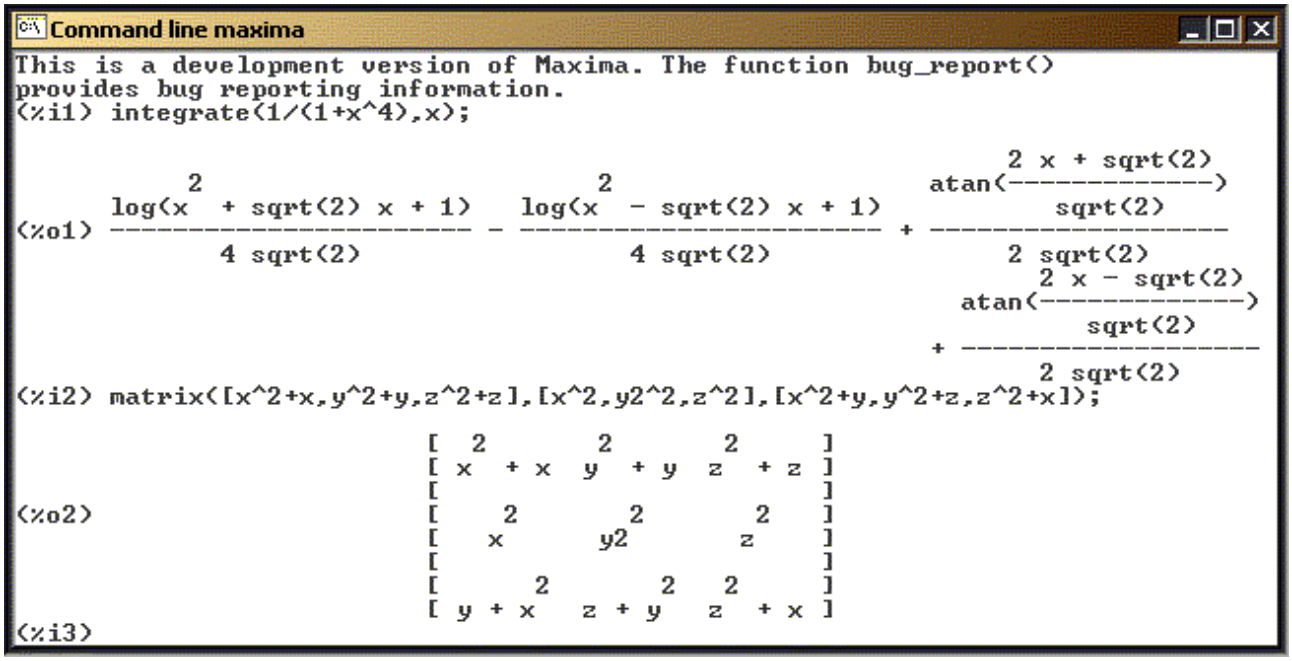


Рис. 3.7. Консольний інтерфейс Maxima

Графічний інтерфейс xmaxima (рис. 3.8) додає до ядра СКМ засоби побудови графіків та виклику довідкової системи. Сьогодні цей інтерфейс повністю локалізований і рекомендується до застосування на комп'ютерних системах з мінімальними ресурсами.

Найбільш придатним для початківців виявився інтерфейс wxMaxima (рис. 3.9), що включає розвинені засоби конструювання вхідних виразів. Інтерфейс TeXmacs відповідає концепції WYSIWYW (What You See Is What You

Want) та спрямований на використання науковцями, являючи собою уніфіковане середовище для створення структурованих документів з різними типами об'єктів (текстових, графічних, математичних, керованих тощо). Для відображення результатів використовується система TeX (рис. 3.10).

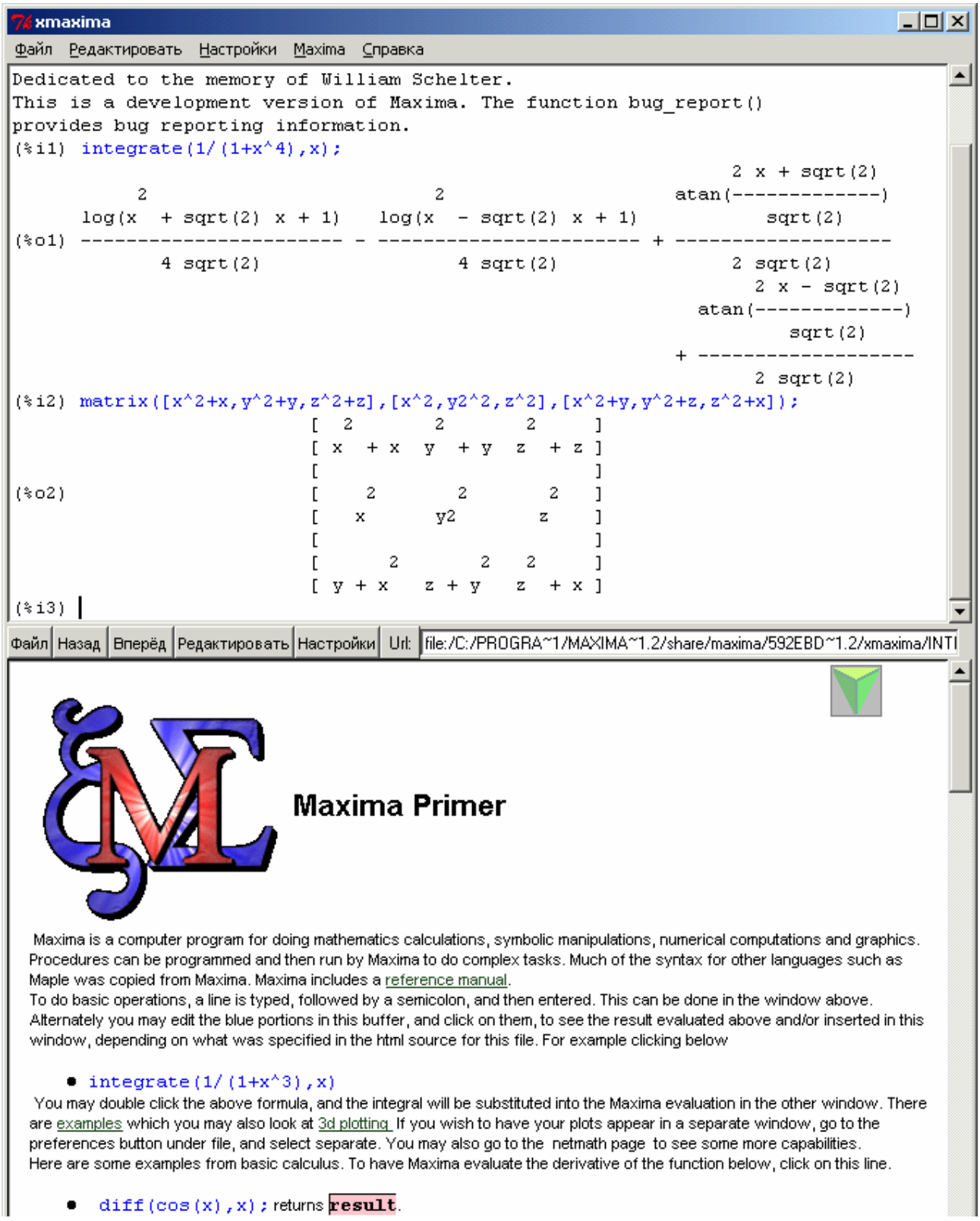


Рис. 3.8. Графічний інтерфейс xmaxima

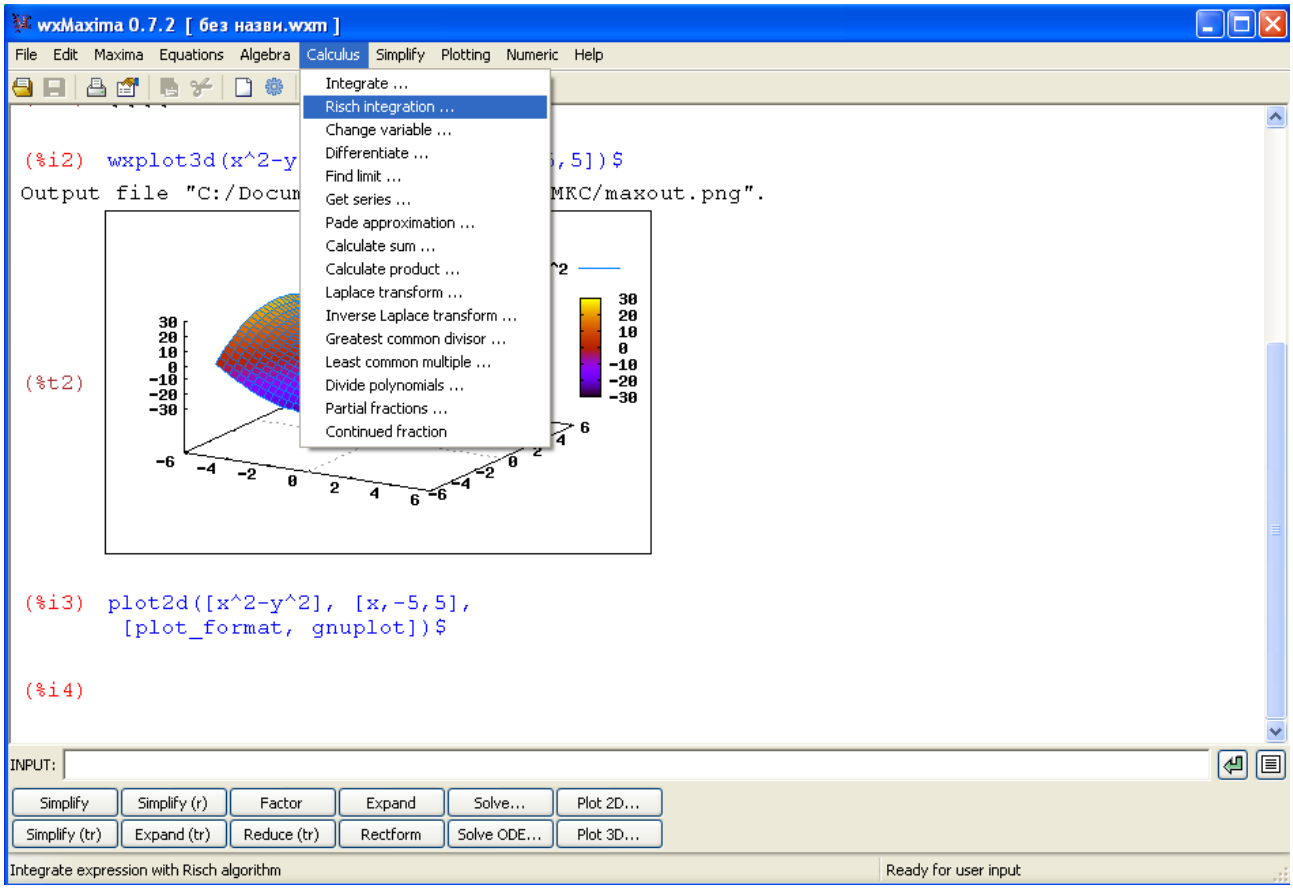


Рис. 3.9. Графічний інтерфейс wxMaxima

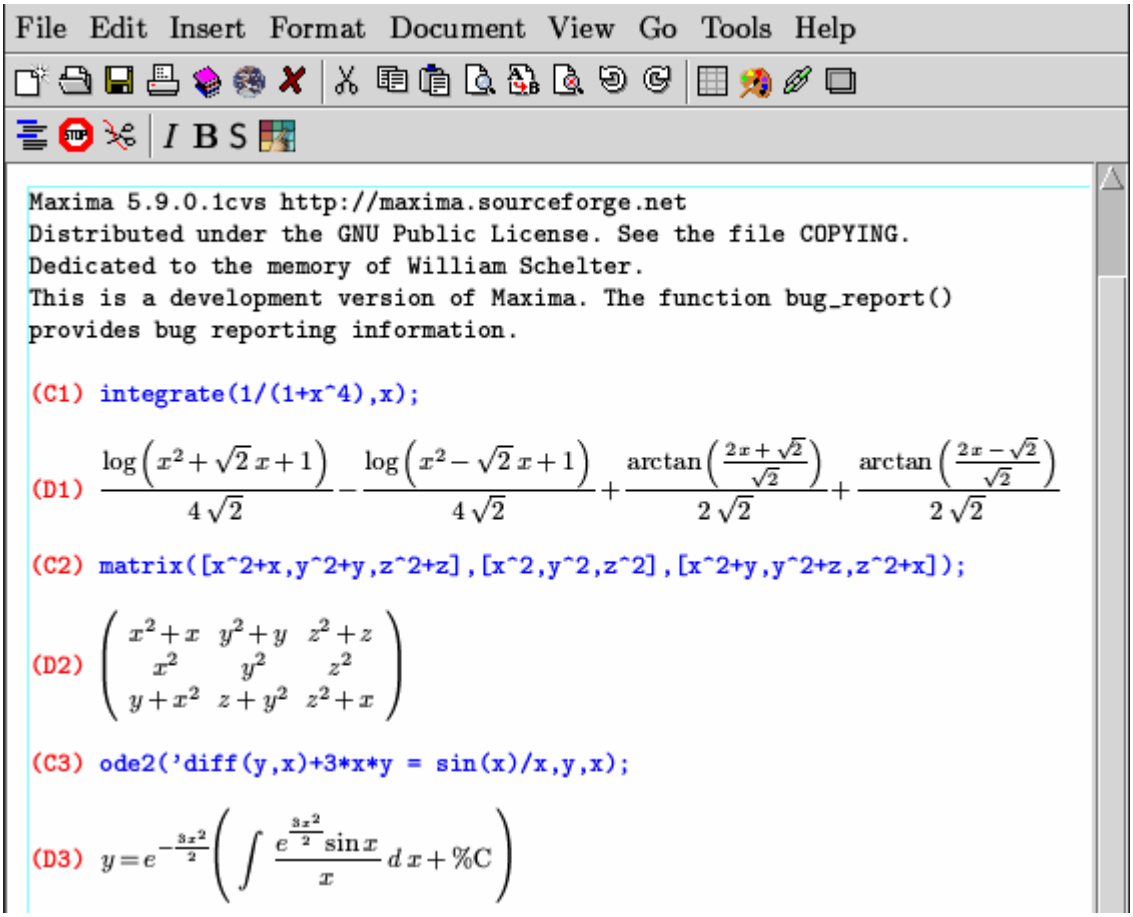


Рис. 3.10. Інтерфейс TeXmacs

Інтерфейс Symaxx/2 (рис. 3.11) орієнтований на подання задачі у вигляді набору взаємопов'язаних текстових, графічних і обчислювальних об'єктів та спрямований на застосування студентами інженерних спеціальностей.

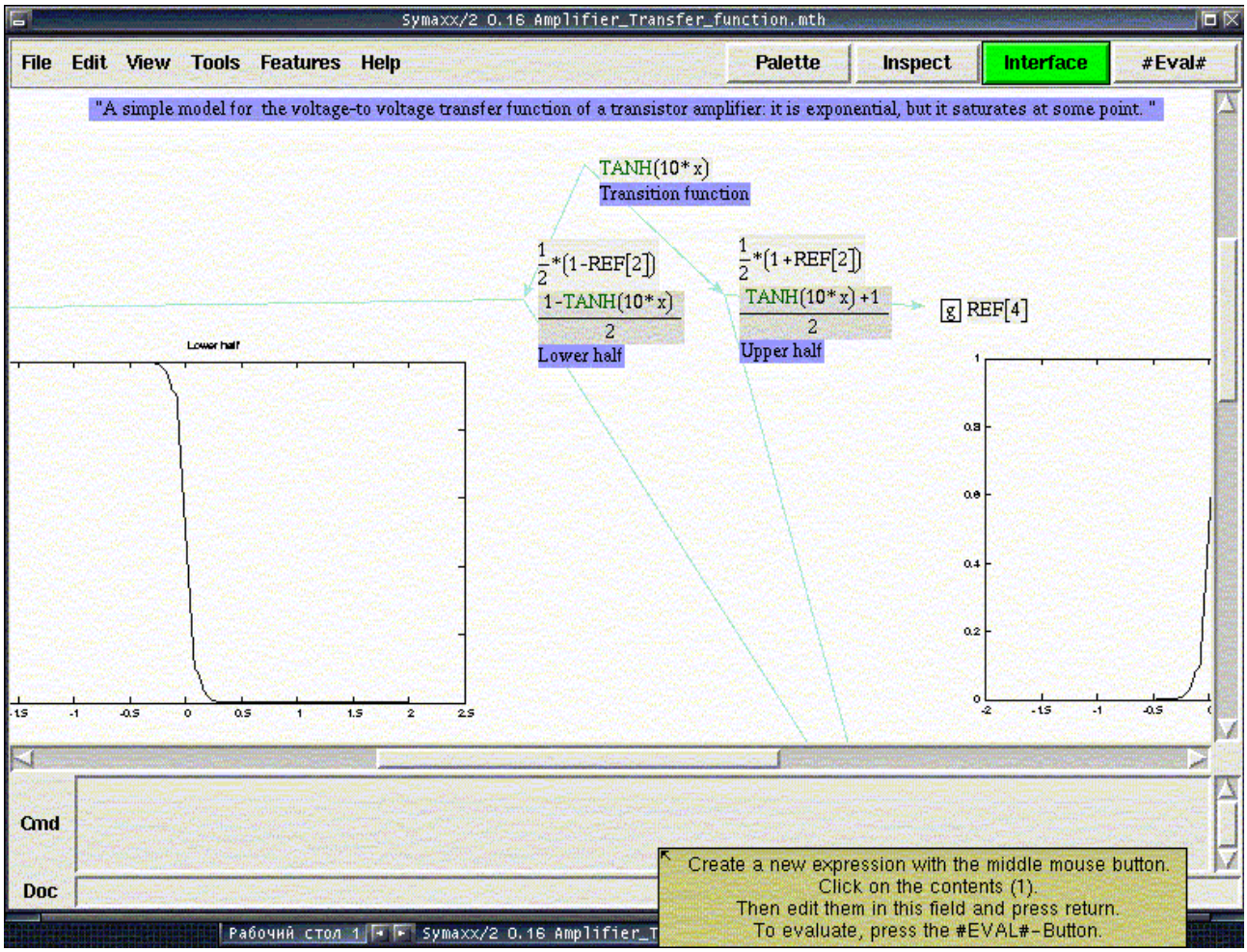


Рис. 3.11. Інтерфейс Symaxx/2

Головним недоліком існуючих інтерфейсів Maxima була відсутність їх локалізованих версій, що є необхідною умовою популяризації даної системи серед вітчизняних користувачів. Адже для того, щоб вільно працювати у розвиненому середовищі комп'ютерної математики, необхідно володіти не тільки англійською, а й її «математичним діалектом».

Від моменту входження у групу розробників Maxima в 2002 році, нами ведеться цілеспрямована робота з її локалізації. Результатами цієї роботи є створення російського варіанту інтерфейсу xmaxima (2004 р.), російського (2006 р.) та українського (2007 р.) варіантів інтерфейсу wxMaxima (рис. 3.12).

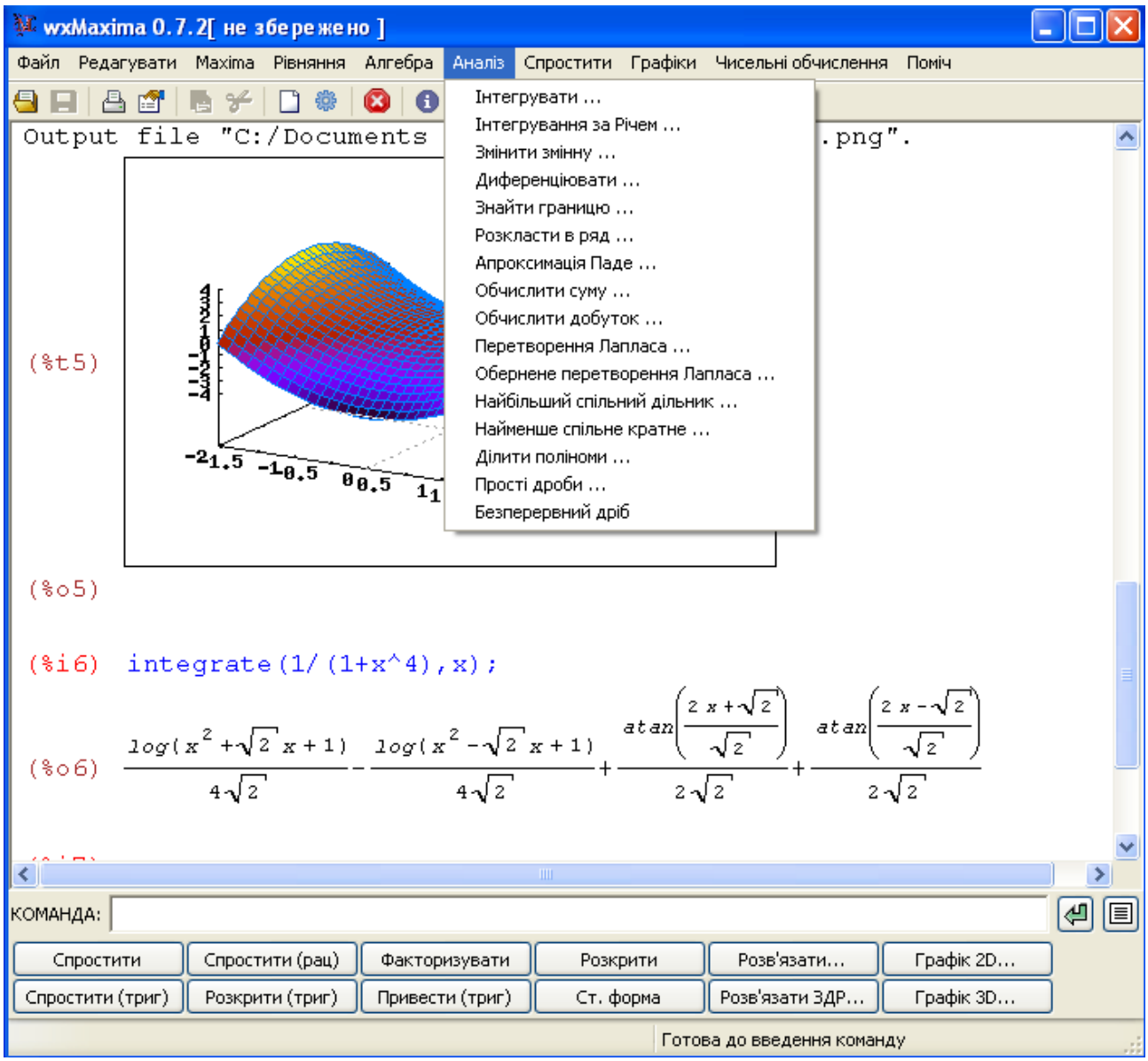


Рис. 3.12. Локалізований інтерфейс wxMaxima

Починаючи з серпня 2007 р., результати виконаної роботи входять у стабільну версію системи, що доступна для завантаження за адресою:

`http://heanet.dl.sourceforge.net/sourceforge/maxima/maxima-5.18.1.exe`

Тривала історія розвитку системи, оптимізовані алгоритми, POSIX-сумісність, невимогливість до ресурсів, спільні з іншими СКМ засоби роботи, ліцензійна чистота, безоплатність, різноманітність інтерфейсів користувача та локалізація – все це дає можливість рекомендувати СКМ Maxima до широкого впровадження у вітчизняній системі освіти в якості стабільного програмного забезпечення математичного призначення.

3.3.4.2. Scilab – пакет наукових програм для чисельних обчислень, що створює потужне відкрите оточення для інженерних і наукових розрахунків. Розробка розпочата у 1980-ті рр.; з 1994 року вільно поширюється через Інтернет. З 2003 р. Scilab підтримується компанією Scilab Consortium. У ній зараз 25 учасників, зокрема Mandriva, INRIA та ENPC (Франція).

Scilab містить сотні математичних функцій з можливістю додавання нових, написаних різними мовами (C, C++, Fortran та ін.). В Scilab підтримуються різноманітні структури даних (списки, поліноми, раціональні функції, лінійні системи) в інтерпретованій мові високого рівня. Scilab був спроектований так, щоб бути відкритою системою, в яку користувачі можуть додавати свої типи даних і операції над цими даними шляхом перевизначення.

У системі доступна наступна функціональність (рис. 3.13):

– 2D- і 3D-графіка, анімація;

– лінійна алгебра, розріджені матриці;

– поліноміальні та раціональні функції;

– інтерполяція, апроксимація;

– моделювання;

– Scicos: гібрид системи моделювання динамічних систем і симуляції;

– диференціальні і недиференціальні оптимізації;

– опрацювання сигналів;

– паралельне програмування;

– статистика;

– робота з клітковими автоматами;

– інтерфейс до Fortran, Tcl/Tk, C, C++, Java, LabVIEW [5].

Scilab включає схожу з MATLAB мову програмування: в складі пакету є утиліта, за допомогою якої можна конвертувати документи з Matlab у Scilab. До складу пакету також входить Scicos – інструмент для редагування блокових діаграм і симуляції, який є аналогом пакету Simulink з MATLAB (рис. 3.14) [474]. Існує можливість спільного використання Scilab з програмою LabVIEW.

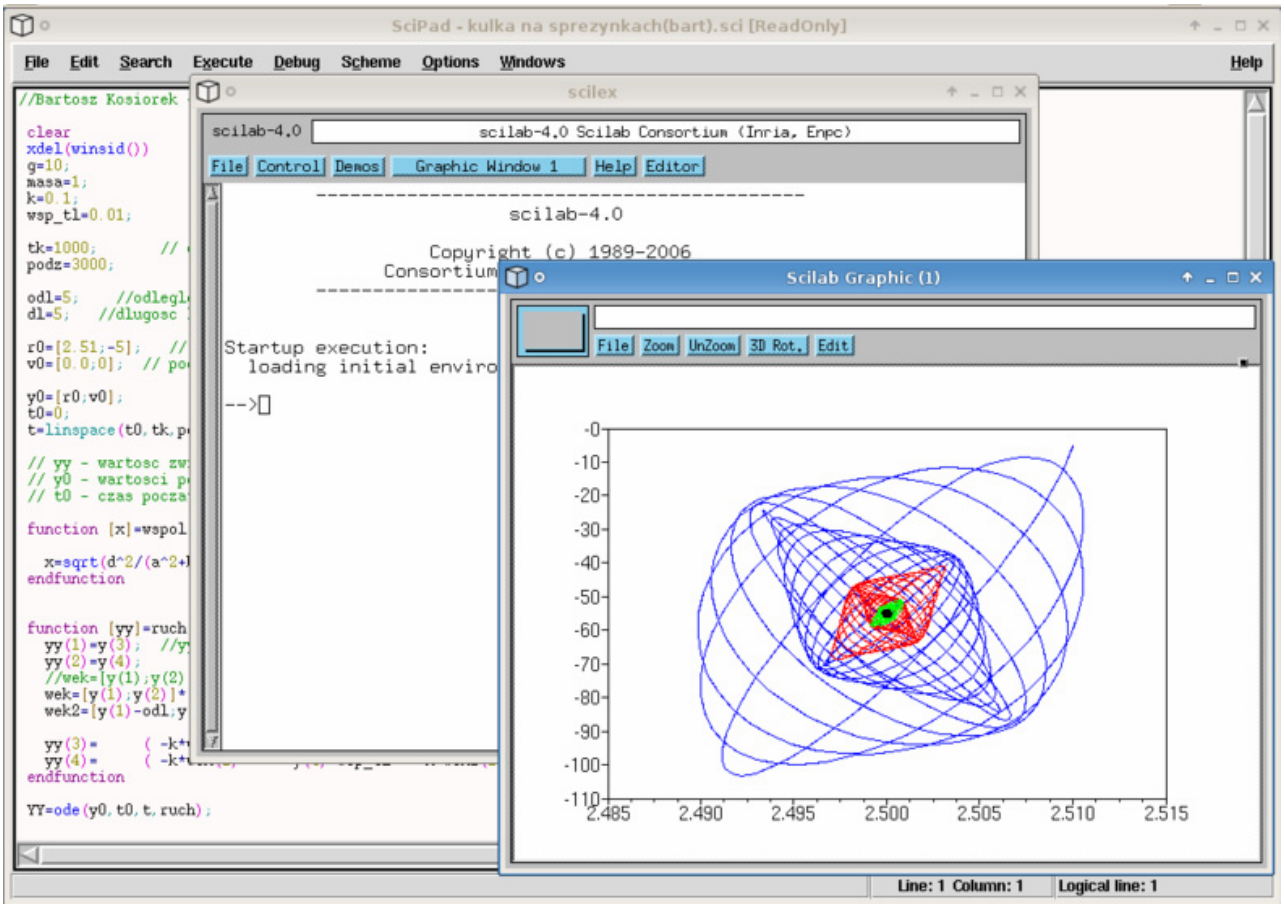


Рис. 3.13. Інтерфейс Scilab

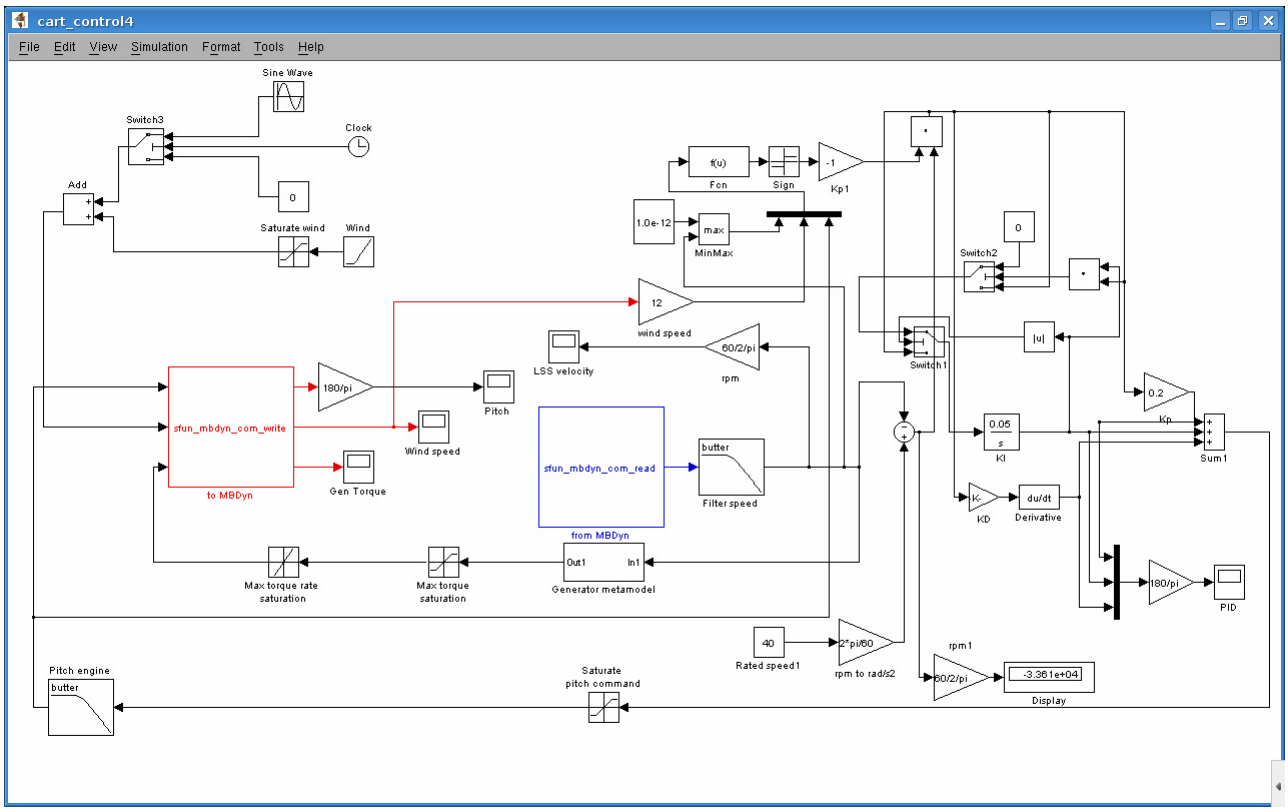


Рис. 3.14. Інтерфейс пакету Scicos

За допомогою Scilab можна опрацьовувати елементарні і велику кількість спеціальних функцій (Бесселя, Неймана, інтегральні функції), система включає потужні засоби роботи з матрицями, поліномами (у тому числі і символьні перетворення), проводити чисельні обчислення і розв'язувати задачі лінійної алгебри, оптимізації і симуляції, потужні статистичні функції, а також засоби для побудови і роботи з графіками. Для чисельних розрахунків використовуються бібліотеки Lapack, LINPACK, ODEPACK, Atlas та інші.

Scilab доступний для різних POSIX-сумісних операційних систем, включаючи Linux та Windows.

3.3.5. Спеціалізовані предметні середовища

3.3.5.1. Оболонка експертних систем CLIPS. Робота з програмним інструментарієм – один із головних напрямів прикладної інформатики, оскільки це дає можливість ознайомити студентів з інформаційними технологіями, з можливими сферами їх застосування в навчанні та в майбутній професійній діяльності. Традиційно вивчення теми «Штучний інтелект. Експертні системи» (за відсутності в навчальному плані окремого предмету «Інтелектуальні системи») починається з подання прикладів штучного інтелекту, бо історично першим науковим напрямом, в зв'язку з яким появився термін «штучний інтелект», була імітація на ЕОМ творчих процесів: складання віршів і музики, доведення теорем, гра в шахи та шашки, переклад текстів тощо [278].

Сучасні експертні системи (ЕС) – це складні програмні комплекси, в яких акумулюються знання та емпіричний досвід фахівців у конкретних предметних галузях і через які поширюється цей емпіричний досвід для консультування інших користувачів. Розробка експертних систем спрямована на використання ЕОМ для опрацювання даних в тих галузях науки і техніки, де традиційні методи моделювання малопридатні.

Саме тому при розгляді теми «Штучний інтелект. Експертні системи» головну увагу приділяють готовим програмним засобам – оболонкам експертних систем FirstClass та Visual Expert. Головною особливістю цих оболонок є візуальний інтерфейс до бази знань та жорстко зафіксовані правила виведення. Це, з

одного боку, надає можливість швидко побудувати експертну систему, але з іншого, – звужує клас систем, що можуть бути побудовані.

У візуальних оболонках ЕС реалізується найвищий рівень абстракції, тоді як в функціональних та декларативних мовах програмування – найнижчий. Середній рівень абстракції забезпечується в оболонках ЕС, що, поряд із візуальним інтерфейсом, включають мови для створення баз знань та побудови запитів до них.

Останні оболонки є цілком придатними для навчання ЕС у ВНЗ як педагогічного, так і технічного спрямування: лише трохи програючи у простоті роботи з суто візуальними оболонками, їх використання надає студентам можливості глибше опанувати розглядувану тему. При виборі засобу навчання ЕС доцільно застосовувати табл. 3.2.

Таблиця 3.2

Вибір засобу навчання ЕС в залежності від навчального часу

Навчальний час, год.	Середовище навчання ЕС	Рівень абстракції	Засіб навчання
< 6	Суто візуальна оболонка	найвищій	FirstClass, Visual Expert
6–12	Візуальна оболонка із вбудованою мовою	середній	CLIPS, BESS
> 12	Функціональна чи декларативна мова програмування	низький	Lisp, Prolog, Scheme

Серед багатьох існуючих оболонок ЕС однією з найбільш популярних є оболонка CLIPS, що має понад 20 років історії розвитку та широку інсталяційну базу [423]. Ця оболонка вигідно відрізняється від інших швидкодією та відкритістю, проте для не англомовного користувача її зручність обмежена неможливістю подання фактів та правил рідною мовою.

З метою подолання цього обмеження була розроблена локалізована версія CLIPS 6.23 (рис. 3.15), що може експлуатуватися на POSIX-сумісних платформах [309].

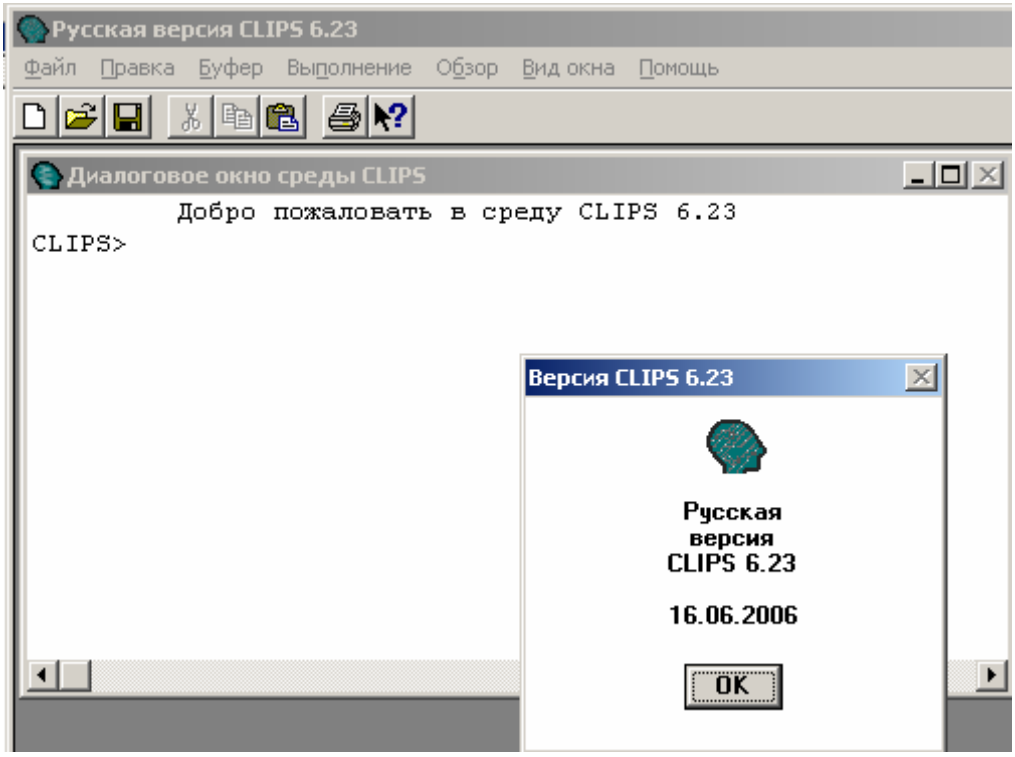


Рис. 3.15. Головне вікно CLIPS

Спочатку абревіатура CLIPS була назвою мови (мова С, інтегрована з продукційними системами), зручної для розробки баз знань і макетів експертних систем. Мова CLIPS була створена в Центрі космічних досліджень NASA у 1984 році.

Сьогодні CLIPS являє собою сучасний інструмент, призначений для створення експертних систем. CLIPS складається з експертної оболонки зі своїм способом подання знань, гнучкої і потужної мови і кількох допоміжних інструментів.

Головна перевага CLIPS у тому, що використання мови і середовища CLIPS дають змогу користувачам швидко створювати ефективні, компактні експертні системи. При цьому користувач застосовує арсенал вже готових інструментів (механізм управління базою знань, механізм логічного виведення, менеджери різних об’єктів CLIPS) і конструктори (упорядковані факти, шаблони, правила, функції, родові функції, класи, модулі, вбудовану мову COOL для об’єктно-орієнтованого програмування).

Локалізовані інтерфейс користувача, системні повідомлення, синтаксис та довідкова систему CLIPS (рис. 3.16) дають наступні переваги:

а) зручність у використанні та при написанні експертних систем;

б) якщо виникає помилка, видаються повідомлення рідною мовою, що полегшує розуміння змісту помилки;

в) при написанні програми користувач має можливість застосовувати російські та українські позначення фактів, змінних, правил, процедур тощо [297].

Рис. 3.16. Головний файл допомоги

Для демонстрації прикладів можна використовувати російську Windows-версію CLIPS 6.23, що цілком сумісна з базовою специфікацією мови.

Основним методом роботи з CLIPS є застосування командного рядка. Після появи в головному вікні CLIPS запрошення – *CLIPS*> – команди користувача можуть вводитися в середовище безпосередньо з клавіатури. Команди можуть бути викликами системних функцій чи функцій користувача, конструкторами різних даних CLIPS тощо. У випадку виклику користувачем деякої функції вона негайно виконується, а результат її роботи відображається на екрані.

Зауважимо, що всі операції, вказівки про виконання яких задаються у командному рядку, відображені у відповідних пунктах меню.

Для того щоб запустити експертну систему на виконання, необхідно ввести команду *reset* і команду *run*. Після цього система готова до використання. Для повторного запуску експертної системи необхідно ще раз виконати команди *reset* і *run*.

Відразу після запуску середовища CLIPS на виконання на екрані з'являється запрошення, в якому користувач сповіщається про те, що він працює з інтерпретатором. У режимі інтерпретатора користувач може використовувати безліч команд: створювати нові факти, правила, описувати функції, використовувати конструктори, об'єкти CLIPS тощо.

Факти – одна з основних форм подання даних в CLIPS. У CLIPS фактом є список неподільних значень примітивних типів даних. В системі CLIPS підтримується два типи фактів – упорядковані факти і неупорядковані факти чи шаблони. Факти можна додавати, вилучати, змінювати і дублювати, вводячи відповідні команди з клавіатури або з програми.

Правила в CLIPS призначені для визначення евристик так званих «емпіричних правил», за якими визначається набір дій, що виконуються при виникненні деякої ситуації. Правила (продукції) складаються з умов (Якщо) і наслідку (То). Ліва частина правила задається набором умовних елементів, що звичайно складаються з умов. Всі умови в лівій частині правила поєднуються за допомогою неявного логічного оператора *and*. Права частина правила містить

список дій, виконуваних при реалізації правила через механізм логічного виведення. Для відокремлення правої і лівої частини правил використовується символ =>. Дії, вказані в правилі, виконуються послідовно, але тоді і тільки тоді, коли всі умовні елементи в лівій частині цього правила задовольняють іншим. Якщо в правій частині правила не визначена жодна дія, правило може бути проаналізоване, але при цьому нічого не відбудеться.

Функцією в CLIPS називається частина коду, що має ім'я і при виконанні якої повертається результат виконаної дії (наприклад, екранне подання повідомлення). Функції, результат виконання яких не повертається, але при цьому виконуються певні дії, називаються командами. В CLIPS визначено кілька типів функцій – визначені користувачем *зовнішні функції, системні (внутрішні) функції,* функції, визначені в середовищі CLIPS за допомогою конструктора *deffunction* та *родові функції*.

Функції можуть вводитися з клавіатури або використовуватися в правилах, повідомленнях, визначених користувачем в функціях чи родових функціях. CLIPS містить великий набір функцій, використання яких може задовольнити будь-які потреби користувача, серед яких логічні і математичні функції, функції роботи з рядками і складеними величинами, функції введення/виведення, процедурні функції, функції для роботи з методами родових функцій, функції для роботи з конструкторами.

В меню допомоги наведено приклад експертної системи, призначеної для діагностування загального стану хворого, визначення типу захворювання і надання користувачеві рекомендацій стосовно першої медичної допомоги (рис. 3.17).

Щоб запустити приклад ЕС, його слід зберегти в текстовому файлі з розширенням *.clp* та за допомогою команди *load* завантажити програму в середовище CLIPS. Далі треба виконати команди (reset) і (run). Після цього повідомляється про наявність помилок у програмі, а якщо вони відсутні, то програма готова для демонстрації. Далі відбувається робота користувача з системою, при

цьому користувачеві видаються певні рекомендації, що генеруються в системі автоматично за визначеними у ЕС правилами.

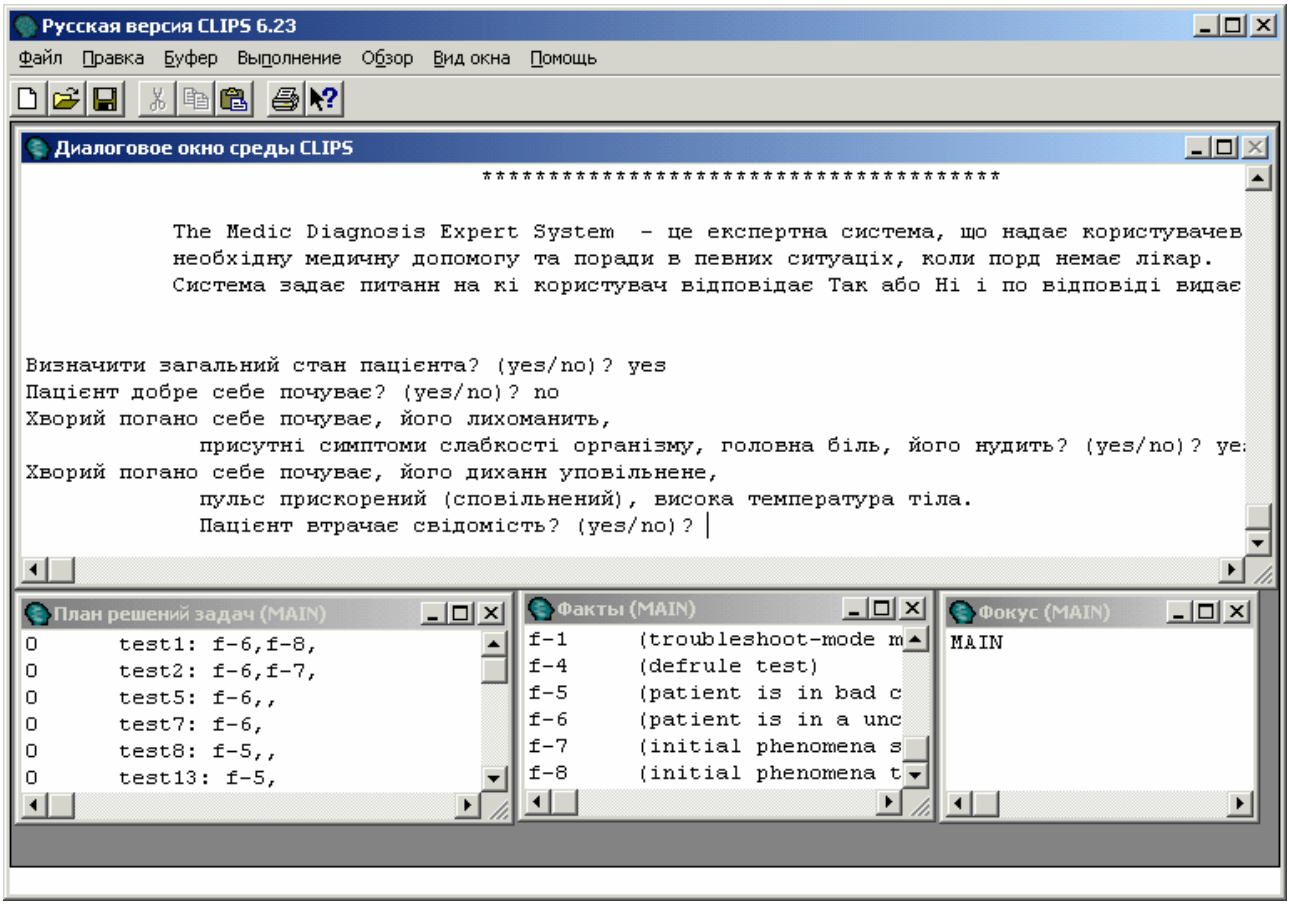


Рис. 3.17. Робота з експертною системою в CLIPS

Однією з основних характеристик CLIPS є її продуктивність, тобто швидкість одержання результату і його вірогідність (надійність). В CLIPS передбачено надання пояснень, чому запропоновано саме таке рішення, і його обґрунтування. Система не вимагає інсталяції, мала за розміром (не більше 825 кілобайт) та невимоглива до апаратних ресурсів.

CLIPS – це вільно поширюваний продукт, локалізована версія якого створена для всіх сучасних операційних систем. Це дає можливість стабілізувати засоби навчання та зміст теми «Штучний інтелект. Експертні системи», зробивши її незалежними від операційної системи.

3.3.5.2. Мультимедійне об'єктно-орієнтоване середовище Squeak. Мова Smalltalk була розроблена як програмна частина проекту Алана Кея Dynabook (п. 2.2.1). Smalltalk є одночасно і мовою програмування, і середовищем розроб-

ки програм. Це чисто об’єктно-орієнтована мова, у якій абсолютно все розглядається як об’єкти. Як зазначає один з її розробників Д. Інгаллс, «мета проекту Smalltalk – зробити світ інформаційних ресурсів доступним для дітей будь-якого віку. Всі труднощі полягають у тому, щоб знайти й застосувати досить прості й ефективні метафори, що дозволить людині вільно оперувати найрізноманітнішими даними – від чисел і текстів до звукових і зорових образів» [499]. В основу мови покладені дві прості ідеї: 1) усе є об’єктами; 2) об’єкти взаємодіють, обмінюючись повідомленнями.

Більш глибокий аналіз Smalltalk показує, що це ретельно продумана фундаментальна розробка, яка не має прямих аналогів у традиційній практиці виробництва програмної продукції, оскільки охоплює на єдиній концептуальній основі всі відомі програмно-апаратні рівні віртуальної машини користувача. При цьому мікропроцесорна (апаратна) реалізація основних системних класів може не тільки значно випередити сучасний рівень розвитку ЕОМ, але й забезпечити ефективну реалізацію подальших поколінь ЕОМ [123].

Світову популярність набула версія Smalltalk-80, комерційні реалізації якої вийшли в 1981 р. Smalltalk увібрав у себе багато чого з проекту Dynabook: у ньому вперше використано растрову графіку з вікнами, що перекриваються, меню, іконками й маніпулятор «миша». В Smalltalk закладено основи сучасного графічного інтерфейсу користувача, на яких безпосередньо базуються інтерфейси Macintosh, Windows і Motif.

В 1995 р. А. Кей, Д. Інгаллс і Т. Кьохлер працювали в Apple, будучи усе ще зацікавленими у своєму баченні Dynabook як середовища розробки для побудови освітнього програмного забезпечення, яке зможуть використовувати (і навіть програмувати) не лише технічні фахівці. На жаль, у комерційних реалізаціях Smalltalk, що одержали поширення на той час, зникли багато ідей проекту Dynabook, тому, вирішивши, що «правильного» Smalltalk не існує, А. Кей з колегами почали створення Squeak («Скрип») – відкритого, вільно поширюваного середовища розробки. У вересні 1996 р. Squeak став доступний в Інтернет.

За минулі роки він був успішно перенесений на різні варіанти ОС UNIX, Windows і навіть Windows CE.

Сьогодні розробка Squeak продовжується тією ж групою в Walt Disney Imagineering – дане середовище використовується у багатьох диснеївських проектах.

Як і Dynabook, Squeak реалізує концепцію конструкціонізму, розроблену С. Пейпертом. Найближчою до неї є концепція навчання через дослідження, що активно розвивається С.А. Раковим. Squeak пропонує спільну для викладача та учня «гру з побудови нових знань»: чим більші можливості відкриває оболонка для самостійного побудови та конструювання нових об’єктів, тим з більшим інтересом до неї відносяться користувачі. Крім Squeak, конструкціонізм помітно вплинув на педагогічний дизайн та реалізацію середовища Лого та похідного від Squeak середовища Scratch, що на більш високому рівні реалізує ідеї середовища Лого.

В мультимедійному об’єктно-орієнтованому середовищі Squeak з’являється усе більше властивостей проекту Dynabook – потужна 2D- і 3D-графіка, багатоголосий і синтезований звук, підтримка анімації й відео, засоби для роботи з різноманітними медіа-форматами тощо (рис. 3.18). Squeak сьогодні – практичний Smalltalk, у якому дослідник, викладач або зацікавлений студент може переглянути вихідний код для будь-якої частини системи, включаючи графічні примітиви й саму віртуальну машину, і виконати будь-які зміни без необхідності використання мови, відмінної від Smalltalk (рис. 3.19).

Squeak включає в себе ряд інтерфейсів користувача: Morphic (основний інтерфейс), eToys (мова візуальних сценаріїв, що базується на Morphic), новий експериментальний інтерфейс Tweak та MVC (наслідуваний від початкового інтерфейсу користувача Smalltalk-80).

Squeak використовується як основний компонент в новій операційній системі Es (рис. 3.20). Багато розробників Squeak співпрацюють у проекті Croquet (Крокет) – надбудовою Squeak, метою якої є створення мережної операційної системи реального часу, що утворює спільний робочий простір з 2D- та 3D-

засобами між кількома користувачами. В ньому забезпечується гнучка структура, в якій більшість концепцій інтерфейсу користувача можуть бути швидко прототиповані і розгорнуті в потужне середовище моделювання. Додатки, створені за допомогою програмного забезпечення для розробників (SDK), можуть бути використані для підтримки високомасштабованої спільної візуалізації даних, віртуального навчання та вирішення проблем навколишнього середовища, 3D-Wiki, онлайнових ігрових середовищ (MMORPG), взаємопов'язані багатокористувацькі віртуальні середовища та ін. (рис. 3.21, 3.22).

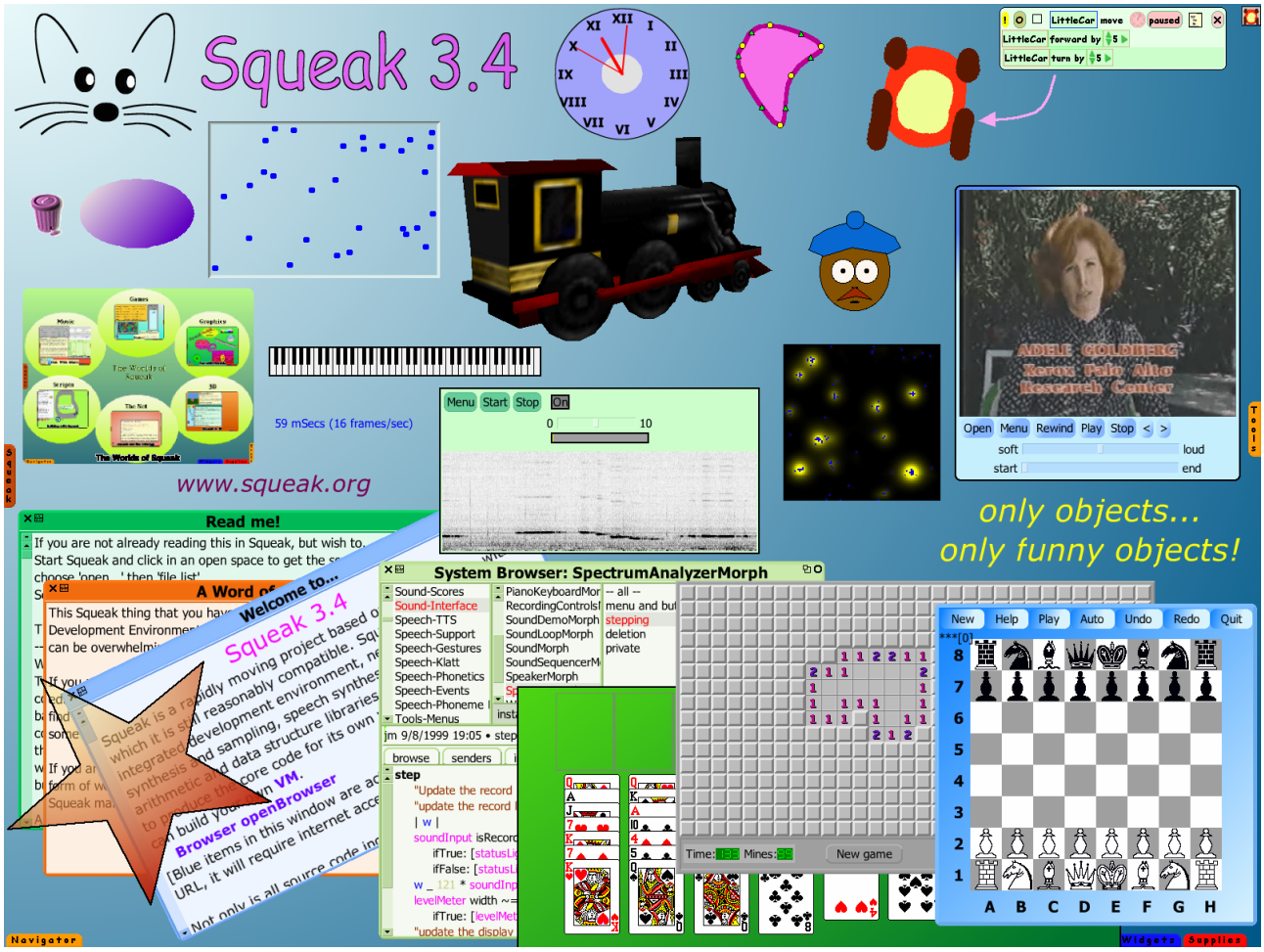


Рис. 3.18. Мультимедійне об'єктно-орієнтоване середовище Squeak

Таким чином, Squeak є розвиненим об'єктно-орієнтованим мультимедіа-середовищем мови Smalltalk, в якому реалізовані основні концепції мобільного навчання та програмування як «другої грамотності» [77]. Вибір Squeak в якості основного компоненту проекту OLPC дає можливість говорити про його високу

стабільність, наявність мобільних реалізацій (рис. 3.23) – про мобільність, а реалізація об’єктного підходу – про фундаментальність даного середовища.

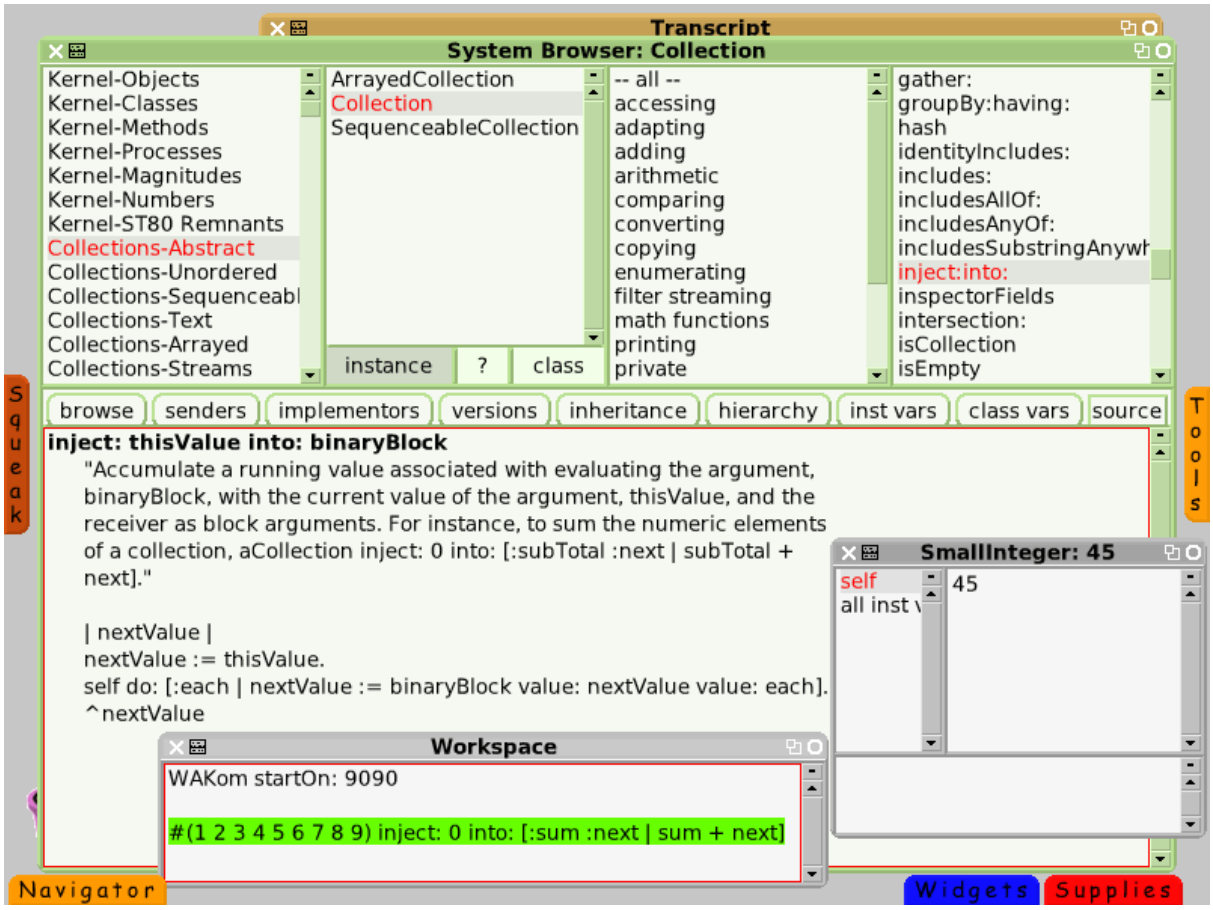


Рис. 3.19. Браузер вихідного коду Squeak

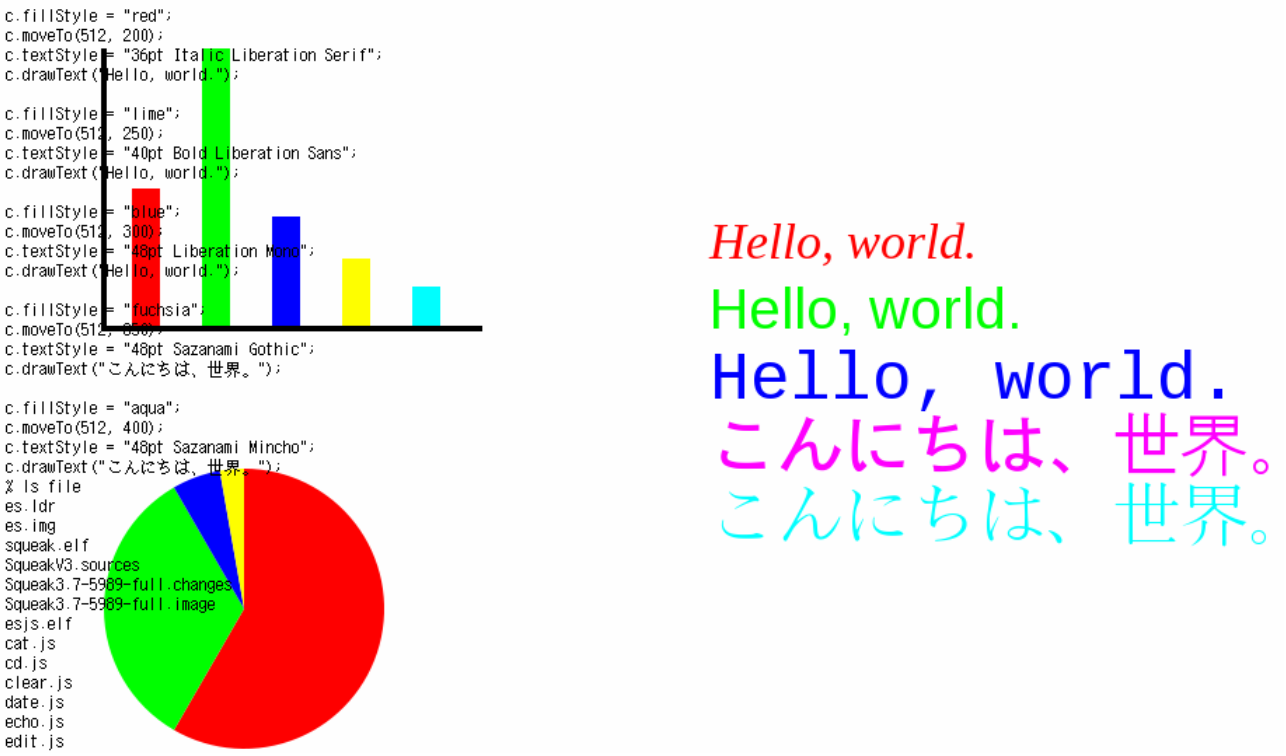


Рис. 3.20. Операційна система Es

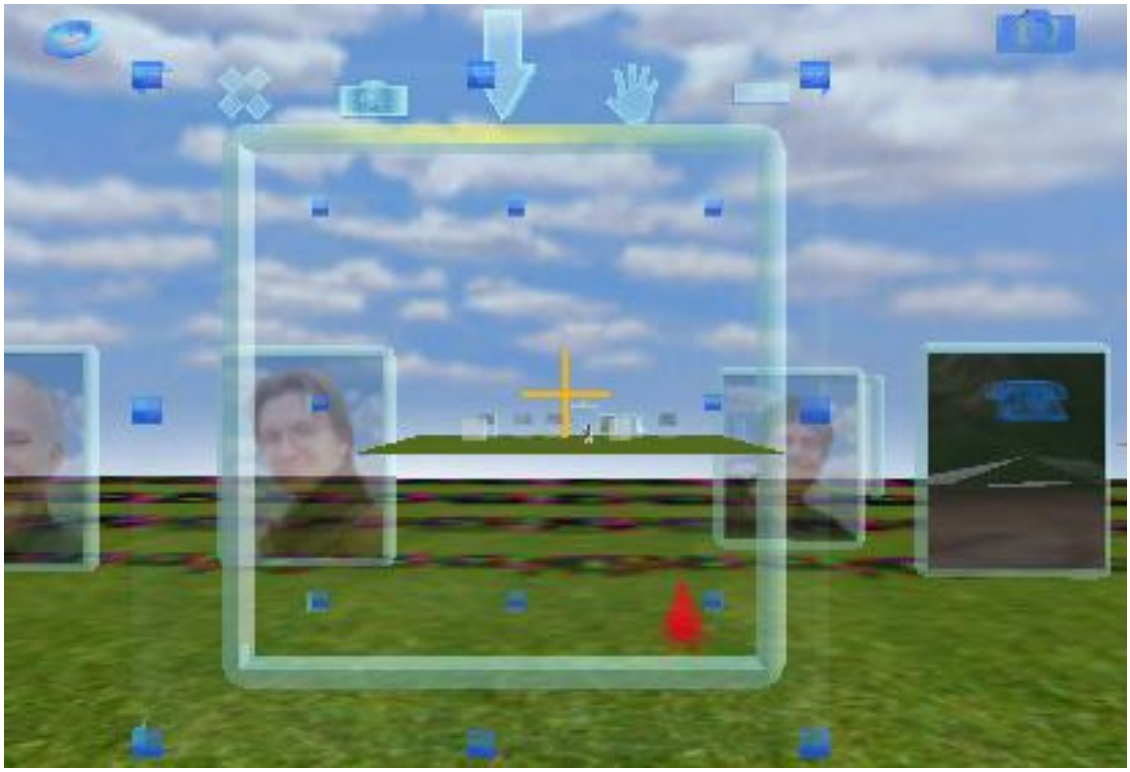


Рис. 3.21. 3D-карта в середовищі Croquet

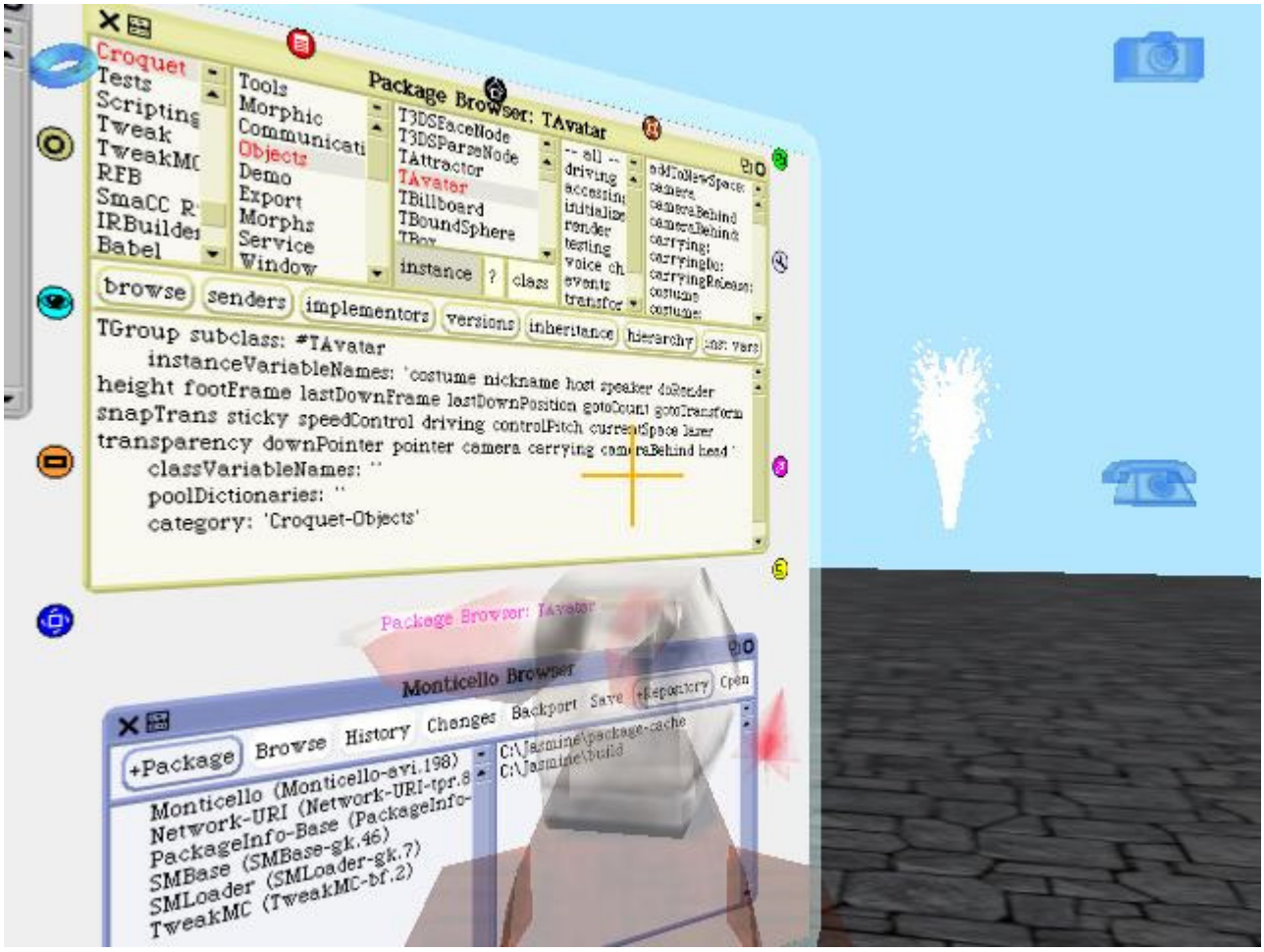


Рис. 3.22. Редагування вихідних кодів Croquet у 3D-середовищі

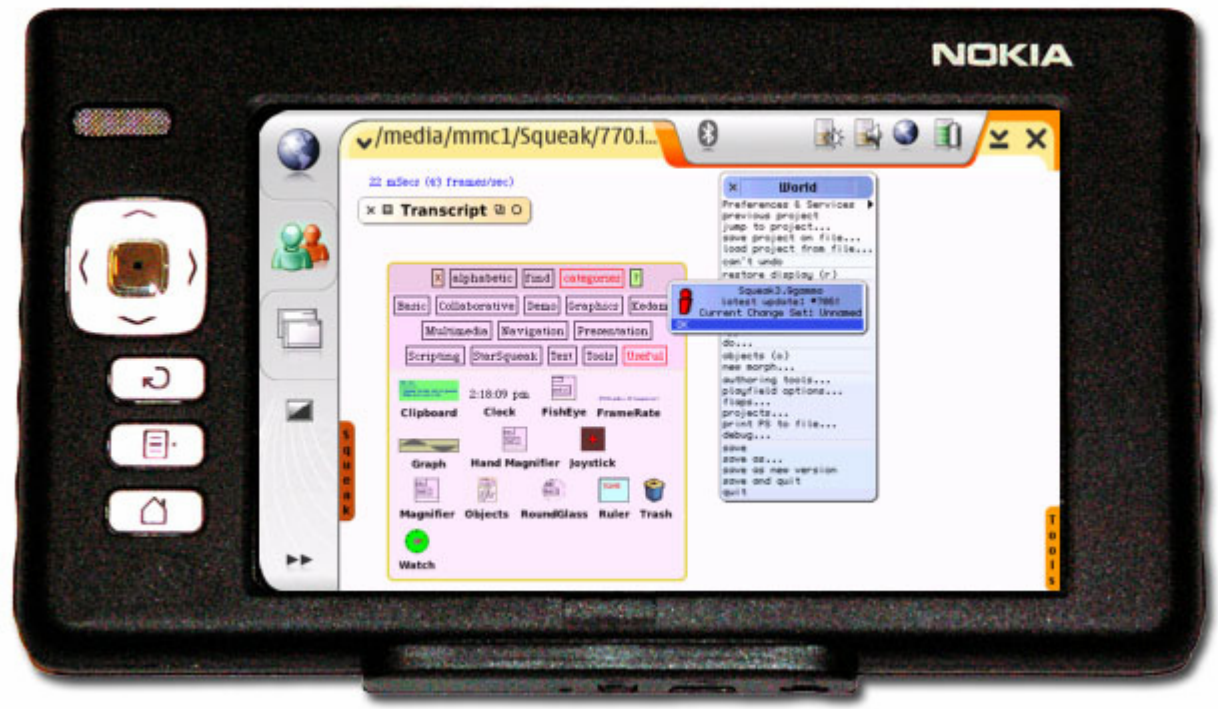


Рис. 3.23. Мобільний Squeak на комунікаторі Nokia 770

3.3.6. Web-середовища. Всесвітні програми Intel World Ahead Education та One Laptop per Child, спрямовані на насищення освітнього ринку недорогими комп'ютерами для забезпечення рівного доступу до засобів ІКТ, викликали до життя концепцію *нетбука* (з англ. netbook) – невеликого ноутбуку, призначеного для доступу до Інтернет та роботи з офісними додатками. Нетбуки відрізняються компактними розмірами (діагональ екрану 7–10 дюймів), невеликою вагою, низькими енерговитратами і відносно невисокою вартістю.

Дані пристрої займають проміжне положення між мобільними Інтернет-пристроями (MID) та КПК «знизу» і субноутбуками «зверху». Через високу вартість швидкісної флеш-пам'яті вони оснащені твердотільними жорсткими дисками (SSD) відносно малого розміру (порядку 4 Гб), що, поряд із невисокою швидкодією та малим обсягом оперативної пам'яті, суттєво ускладнює застосування таких ресурсоємних додатків, як розвинені середовища програмування, системи комп'ютерної математики і т.п.

Для розв'язання цієї проблеми пропонуємо перейти до *мережецентричної моделі* навчання, за якої ресурсоємні додатки працюють на Інтернет-серверах, а головною клієнтською програмою виступає Web-браузер.

3.3.6.1. Онлайн-IDE. Software as a service (SaaS) – *програмне забезпечення як послуга* – це модель пропозиції програмного забезпечення користувачеві, при якій постачальник розробляє Web-додаток, розміщує його і управляє ним (самостійно або через третіх осіб) з метою і можливістю використання замовниками через Інтернет. Замовники платять не за володіння програмним забезпеченням як таким, а за його використання (через API, що доступний через Web і в якому часто використовуються Web-служби). Близьким до терміну SaaS є термін «додаток за запитом» (On-Demand).

Принциповою відмінністю моделі SaaS від інших (Hosted Applications і Application Service Provider (ASP)) є те, що отримується саме послуга і інтерфейс (призначений для користувача або програмний), тобто деяка функціональність без жорсткої прив'язки до способу її реалізації.

У моделі SaaS:

– додаток пристосований для віддаленого використання;

– одним додатком користується кілька клієнтів;

– модернізація додатку відбувається плавно і прозоро для клієнтів;

– для платних послуг: оплата здійснюється як щомісячна абонентська плата або на основі об'єму транзакцій, а підтримка додатку входить до складу оплати.

Тенденція розташування настільних додатків в мережі та надання їх у якості послуг не є новою: так, з 2006 р. на ринку онлайнових текстових процесорів працюють Google Docs (рис. 3.24) та Zoho Writer. У 2007 р. в Інтернет з'явились і перші онлайнові інтегровані середовища програмування.

Більшість Інтернет-IDE має досить специфічний характер (на відміну від IDE загального призначення, таких як Eclipse): IDE не є послугою, що надається, а скоріше інструментом для користувачів, які використовують інші послуги. Прикладами таких послуг є Coghead, Zoho Creator, Bungee Builder, Microsoft

PopFly і Yahoo Pipes. Всі ці сервіси є пропрієтарними, в деяких з них використовуються свої власні мови, і вимагається розміщення всіх послуг виключно на їх серверах.

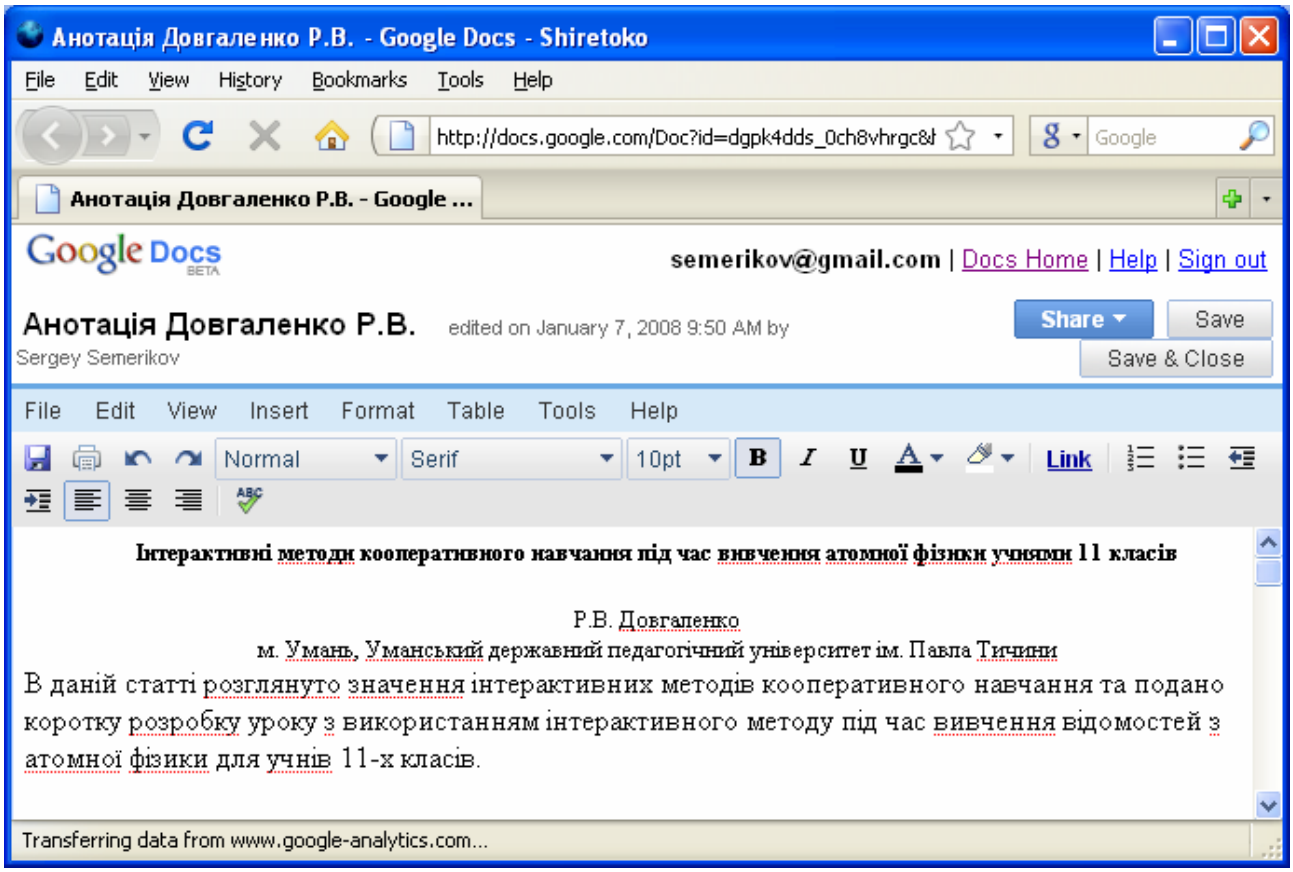


Рис. 3.24. Редагування документа Microsoft Word в Google Docs

Проте існує кілька служб, які мають більш загальний характер, та базуються на стандартних мовах. Наприклад, в Heroku застосовується мова Ruby і він може бути використаний для розробки та розгортання Ruby-додатків. Найближчим часом на основі Google AppEngine планується розгортання Python-орієнтованих сервісів.

CodeIDE є повноцінною онлайн-IDE, що підтримує мови BASIC (рис. 3.25), Pascal, C, C++, Perl, Java, JavaScript (рис. 3.26), HTML, MySQL та LISP. На нашу думку, це середовище є гарним інструментом для залучення до програмування: його використання надає користувачеві можливість швидко почати кодування і миттєво отримати результати. Хоча CodeIDE не є повноцін-

ним інтегрованим середовищем, робота з ним дає уявлення про те, як в цілому може виглядати онлайн-IDE.

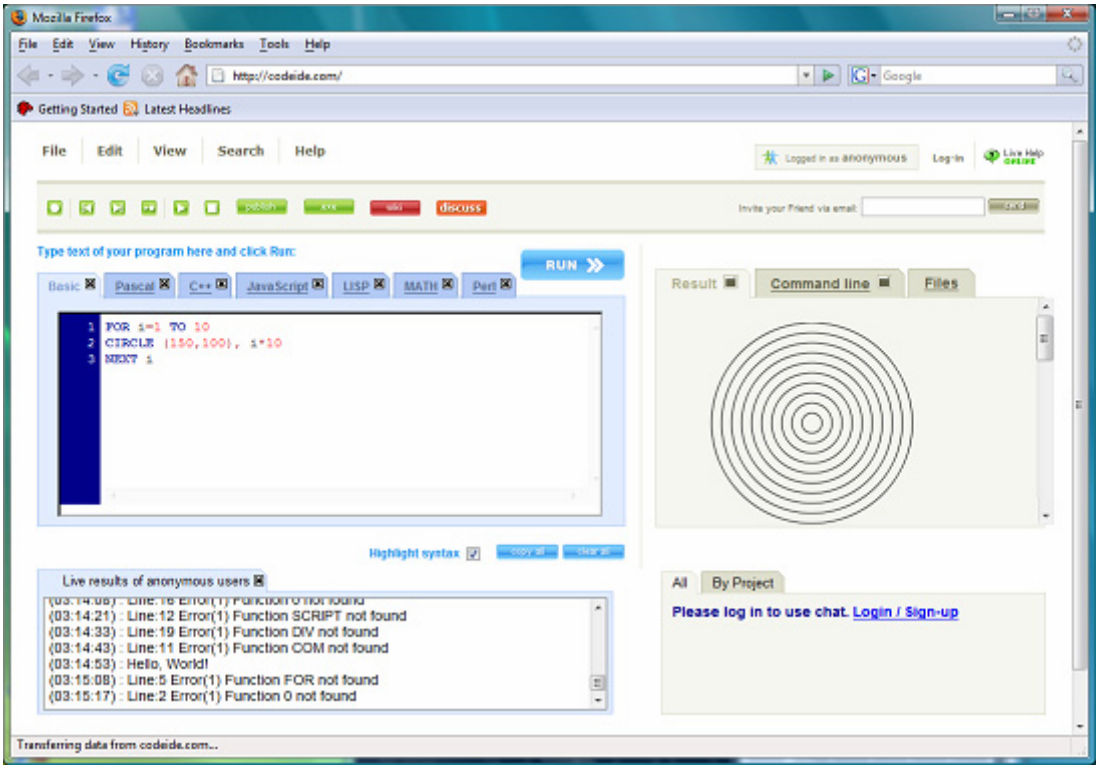


Рис. 3.25. CodeIDE в режимі виконання програми мовою BASIC

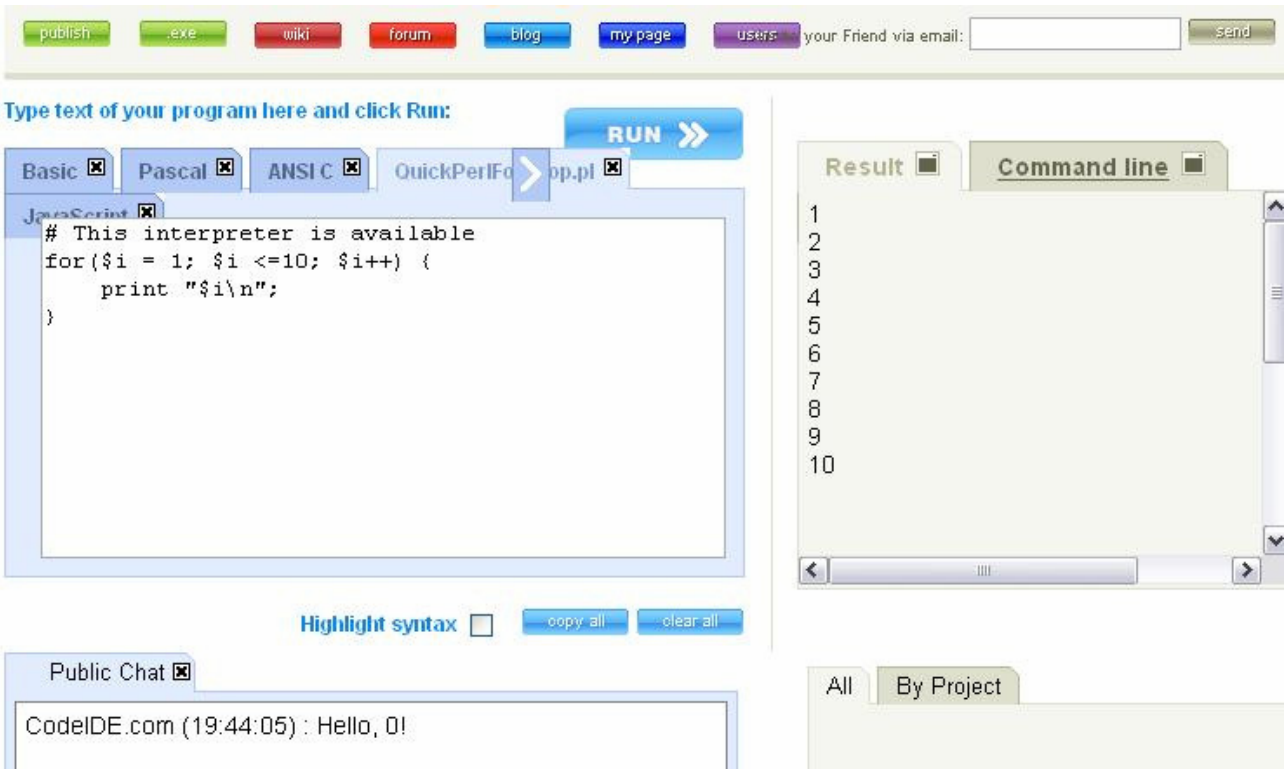


Рис. 3.26. CodeIDE в режимі виконання програми мовою JavaScript

При роботі з онлайн-IDE від їх користувача не вимагається нічого, крім Web-браузера, хоча деякі з них (особливо ті, що оснащені розвиненим інтерфейсом користувача) є орієнтованими на конкретну платформу. На рис. 3.27 наведено зовнішній вигляді онлайн-IDE CodeRun, а на рис. 3.28 – Compilr IDE.

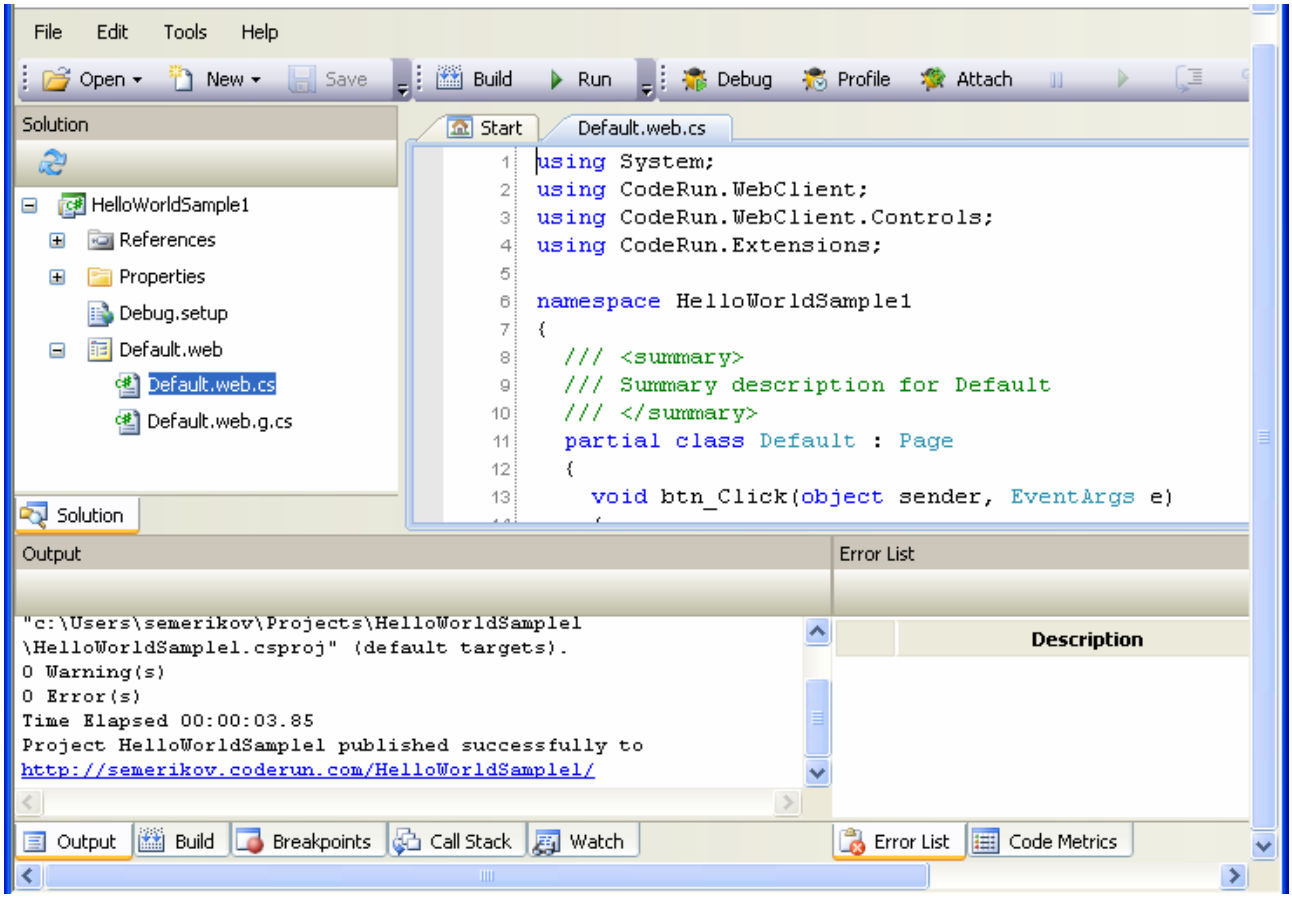


Рис. 3.27. CodeRun в режимі виконання програми мовою C#

Поки що в жодній з онлайн-IDE не надається такого багатого набору функцій, як в «справжніх» (настільних) IDE, такі як Eclipse або Visual Studio. Цей розрив не є унікальним для IDE – достатньо порівняти інструментальні панелі Google Docs та Microsoft Word: Google Docs містить всі основні функції, тим не менше він ще далекий від повномасштабного настільного текстового процесора.

Можна очікувати, що ця ситуація зміниться. Як стверджують аналітики Gartner, перехід до серверних додатків поки що, як і раніше, знаходиться в зародковому стані, але незабаром ця тенденція стане провідною. Так, якщо сьогодні для запуску 70–80% корпоративних додатків потрібна Windows, то до 2011

року більшість цих програм будуть незалежними від операційної системи та існувати у формі, наприклад, Web-додатків [543].

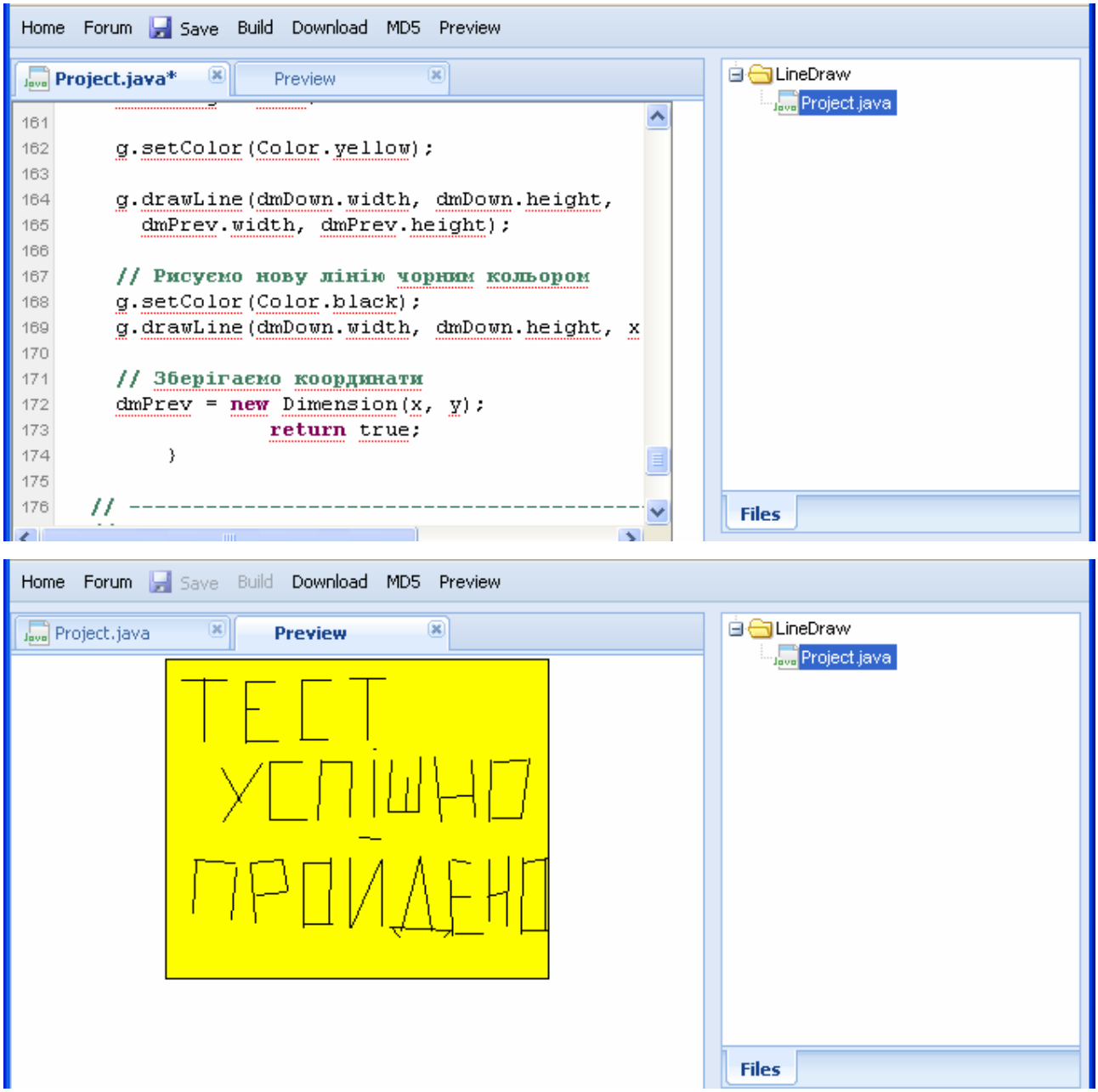


Рис. 3.28. Створення та виконання програми мовою Java в Compilr IDE

Безпосередніми перевагами онлайн-IDE є відсутність необхідності у їх інсталяції та миттєве розгортання проекту. Крім того, їх застосування відкриває нові можливості для обміну навчальними матеріалами та співробітництва (досить згадати про можливість віддаленого парного програмування та спільної роботи над проектами).

Перспективним напрямом роботи є розробка своїх власних спеціалізованих онлайн-IDE та онлайнових плагінів до існуючих середовищ (зокрема, Eclipse). Яскравим прикладом таких спеціалізованих середовищ є онлайн-IDE для розробки AVR-додатків (http://www.online-gcc.com).

3.3.6.2. Математичний Web-інтегратор SAGE. Однією з проблем, що постають в процесі навчання математичної інформатики [133], є вибір середовища для роботи. Як комерційні, так і вільно поширювані системи суттєво різняться за функціональністю (загального призначення, спеціалізовані), інтерфейсом (командного рядка, графічним), розміром (від кількасот кілобайт до кількох гігабайт), вбудованою мовою програмування тощо. Безальтернативне ознайомлення лише з однією системою комп'ютерної математики (навіть такої розвиненої, як Maxima) неминуче впливатиме на подальшу професійну діяльність, обмежуючи клас розв'язуваних задач особливостями конкретного програмного продукту.

Як в педагогічній, так і в інженерній та науково-дослідницькій роботі діє єдиний принцип: *вибір інструмента визначається задачею*, тому обійтися лише однією системою комп'ютерної математики не вдається, та й не потрібно.

Ю.В. Триус вказує основні причини необхідності знання основ роботи з кількома математичними системами: «необхідність раціонального вибору математичної системи з урахуванням особливостей задачі, що розв'язується; необхідність розв'язування складних задач за допомогою різних систем, щоб перевірити правильність результатів, не покладаючись на одну систему (збільшити вірогідність одержаного результату); необхідність підготовки математичних документів (статей, звітів, книг, навчальних занять і т.д.) підвищеної якості. Останнє говорить на користь інтеграції математичних систем між собою і з іншими програмами, що може розглядатися як один з перспективних напрямів розвитку систем комп'ютерної математики» [405, 364–365].

Фактором, що ускладнює вивчення та застосування різних СКМ, є синтаксичні відмінності у застосуванні одних і тих самих команд, що можуть змінюватися навіть в межах однієї СКМ (наприклад, при виході нової версії). Інша

проблема – те, що досить часто в системах загального призначення не вистачає функціональності, що є в спеціалізованих системах, та навпаки. Зауважимо, що частково ця проблема може бути розв'язана через застосування сучасних об'єктних технологій інтеграції програмного забезпечення, таких як COM та CORBA. В результаті для роботи буває необхідним цілий набір СКМ, встановлених на одному комп'ютері, що в умовах загальноосвітньої школи та ВНЗ реалізувати практично неможливо через ліцензійні чи адміністративні перешкоди.

Проблема вибору СКМ та підтримки великої інсталяційної бази може бути розв'язана через застосування мережних технологій, коли користувач за допомогою спеціалізованого клієнтського програмного забезпечення звертається до серверної частини СКМ, де виконуються команди користувача та повертається результат до клієнтського ПЗ. Такі послуги надаються, зокрема, Matlab Web Server, webMathematica та wxMaxima. І, хоча далеко не у всі СКМ включені вбудовані мережні засоби, для тих з них, в яких поряд з візуальним підтримується командний інтерфейс, можливе створення мережної надбудови.

Логічним наступним кроком буде створення оболонки, що інтегрує в собі послуги різних СКМ за допомогою клієнт-серверних технологій, надаючи користувачеві рівний доступ до різних СКМ за допомогою єдиної командної мови (а за необхідності – легко переходити до мови будь-якої СКМ). Тоді на клієнтському комп'ютері не буде потреби у встановленні та налагодженні спільного функціонування різних СКМ – достатньо звернутися до математичного сервера засобами Web-браузера.

Web-СКМ – новий перспективний напрям розвитку СКМ; перші представники таких систем з'явились лише на початку XXI століття. Інтеграція СКМ у єдине мережне середовище – найкраща ілюстрація сучасної концепції «комп'ютер – це мережа», що знаходить своє відображення у перенесенні прикладного ПЗ (навіть «робочих столів») у Web-середовища.

Для користувача за рахунок цього надається можливість мобільного доступу до програм та даних, для адміністратора комп'ютерного класу знімаються проблеми підтримки великої інсталяційної бази та ліцензування ПЗ, для ви-

кладачів – суттєво розширюється спектр використовуваного ПЗ, а для учнів та студентів створюються умови для дистанційного навчання.

SAGE (*Software for Algebra and Geometry Experimentation* – програмне забезпечення для алгебраїчних та геометричних досліджень) – це безкоштовне вільно поширюване середовище математичних обчислень для виконання символьних, алгебраїчних та чисельних розрахунків. Його інтерфейс написаний потужною і досить популярною мовою програмування Python. В SAGE об'єднано послуги популярних вільно поширюваних математичних програм та бібліотек, таких як PARI, GAP, GSL, Singular, MWRANK, NetworkX, Maxima, Sympy, GMP, Numpy, mathplotlib та багатьох інших засобами Python, Lisp, Fortran 95 та C/C++ [440, 6].

SAGE є серверним ПЗ, що базується на відомому Python-CMS Zope. Розвинена функціональність Web-додатку забезпечується широким застосуванням технології AJAX, що є основою більшості продуктів Web 2.0, а адекватність відображення математичних даних – браузерними математичними шрифтами (jsMath).

В SAGE є власне символьне ядро (рис. 3.29), проте основним його призначенням є інтеграція різних систем [443] та надання до них єдиного Web-інтерфейсу (рис. 3.30). Можливість виконання на Web-сторінках, генерованих SAGE, програм мовами Fortran, Python, Lisp, Java та ін., надає їм надвисокого рівня керованості, порівняного з традиційними СКМ, без суттєвих вимог до апаратних ресурсів комп'ютера користувача (необхідні лише браузер та мережне з'єднання).

Перша версія SAGE з'явилася в лютому 2006 року, останньою на сьогодні є версія 4.1. Найновіша версія SAGE завжди доступна за адресою http://www.sagemath.org/.

Розвинений Web-інтерфейс, безкоштовність та відкритість середовища SAGE – це основні, але не єдині переваги цього засобу. Слід вказати ще на такі особливості SAGE [440, 7]:

– невимогливість до апаратної складової обчислювальної системи;

– індиферентність до використовуваного браузера та операційної системи;

– підтримка інтерфейсів комерційних систем комп'ютерної математики – Maple, Magma, Mathematica, Matlab та ін.;

– подання математичних виразів у природний спосіб не вимагає встановлення додаткового програмного забезпечення;

– публікація робочих аркушів (Worksheets) у мережі Internet;

– підтримка технології Wiki [472];

– потужний інструментарій для побудови статичних та динамічних графічних зображень (на площині та у просторі).

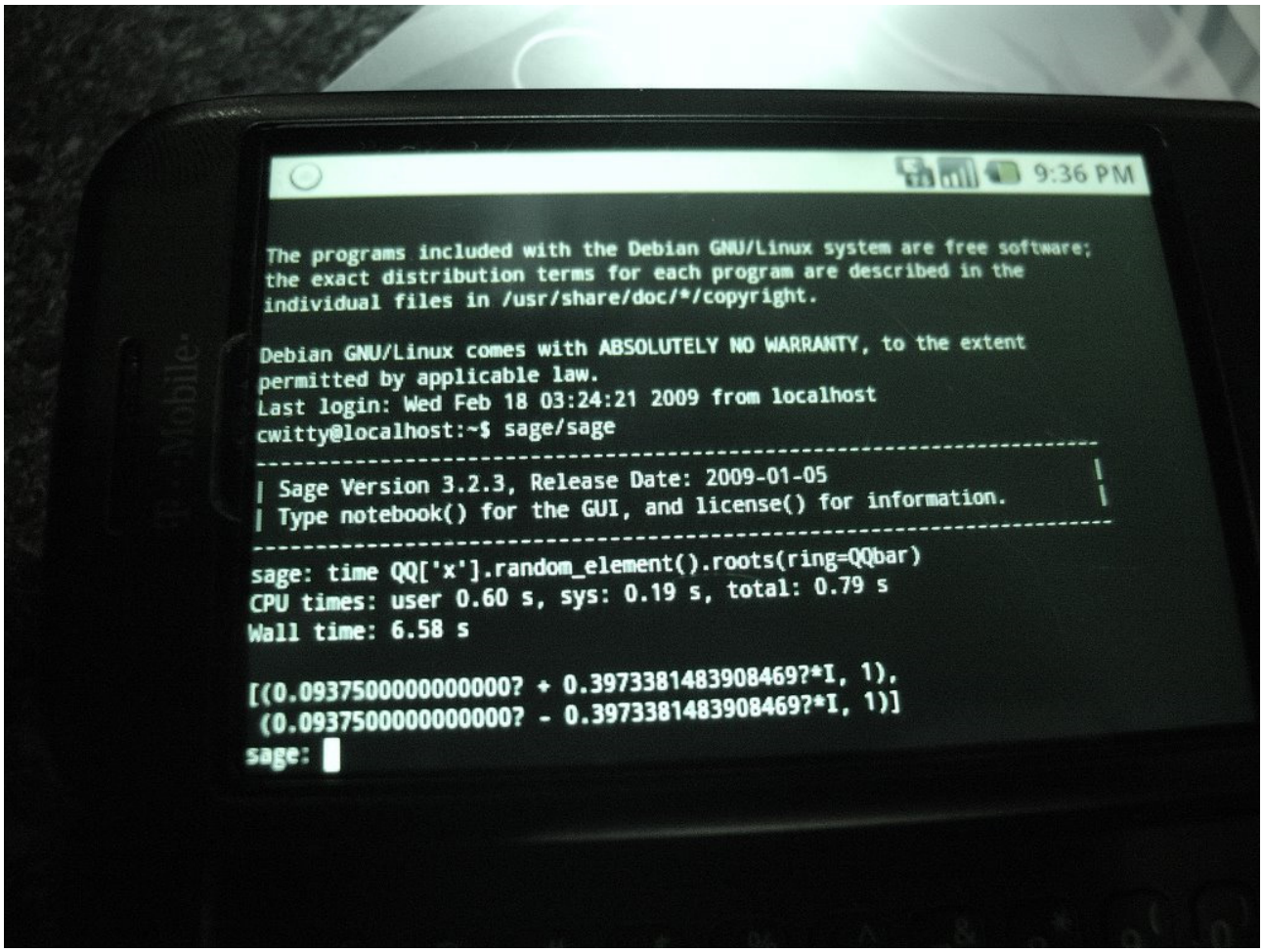


Рис. 3.29. Консольна версія SAGE на мобільному телефоні T-Mobile G1

Для організації роботи у локальній мережі достатньо встановити SAGE на будь-якому комп'ютері (а не лише на сервері). Для цього необхідно встановити вільно поширюваний засіб для віртуалізації комп'ютера Vmware Player та розгорнути архів з образом Linux, у якому встановлено SAGE. У 2009 році плану-

ється випуск версії SAGE, що здатна виконуватись під управлінням ОС Windows без додаткового програмного забезпечення.

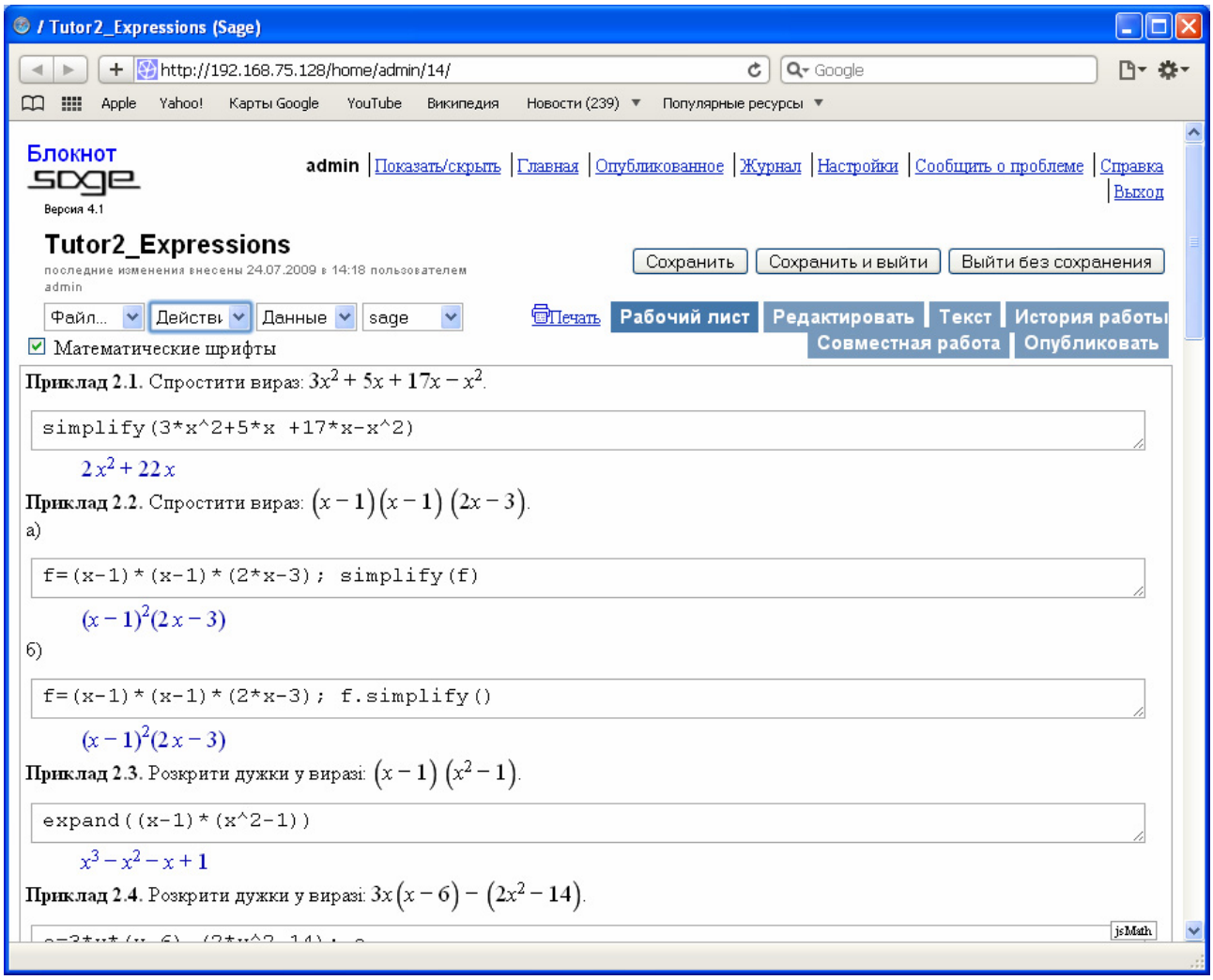


Рис. 3.30. Web-інтерфейс SAGE

Починаючи з 2008 р. SAGE використовується у Криворізькому державному педагогічному університеті в процесі навчання теорії алгоритмів, моделювання (рис. 3.31), методів оптимізації, чисельних методів, теорії кодування [251], паралельних та розподілених обчислень [257], розпізнавання образів та інших навчальних дисциплін.

Також SAGE може бути використаний в процесі навчання елементарної та вищої математики, у тому числі лінійної та вищої алгебри, геометрії, математичного аналізу, методів математичної фізики, теорії чисел, комбінаторики, теорії графів та багатьох розділів математичної інформатики (додаток В). З метою підвищення зручності встановлення, налагодження та роботи системи у 2009 р.

нами було створено інсталяційний пакет SAGE для ОС сімейства Windows, що містить локалізований Web-інтерфейс SAGE, адаптований до роботи у середовищі браузер Opera Mini на мобільних телефонах та смартфонах (рис. 3.32).

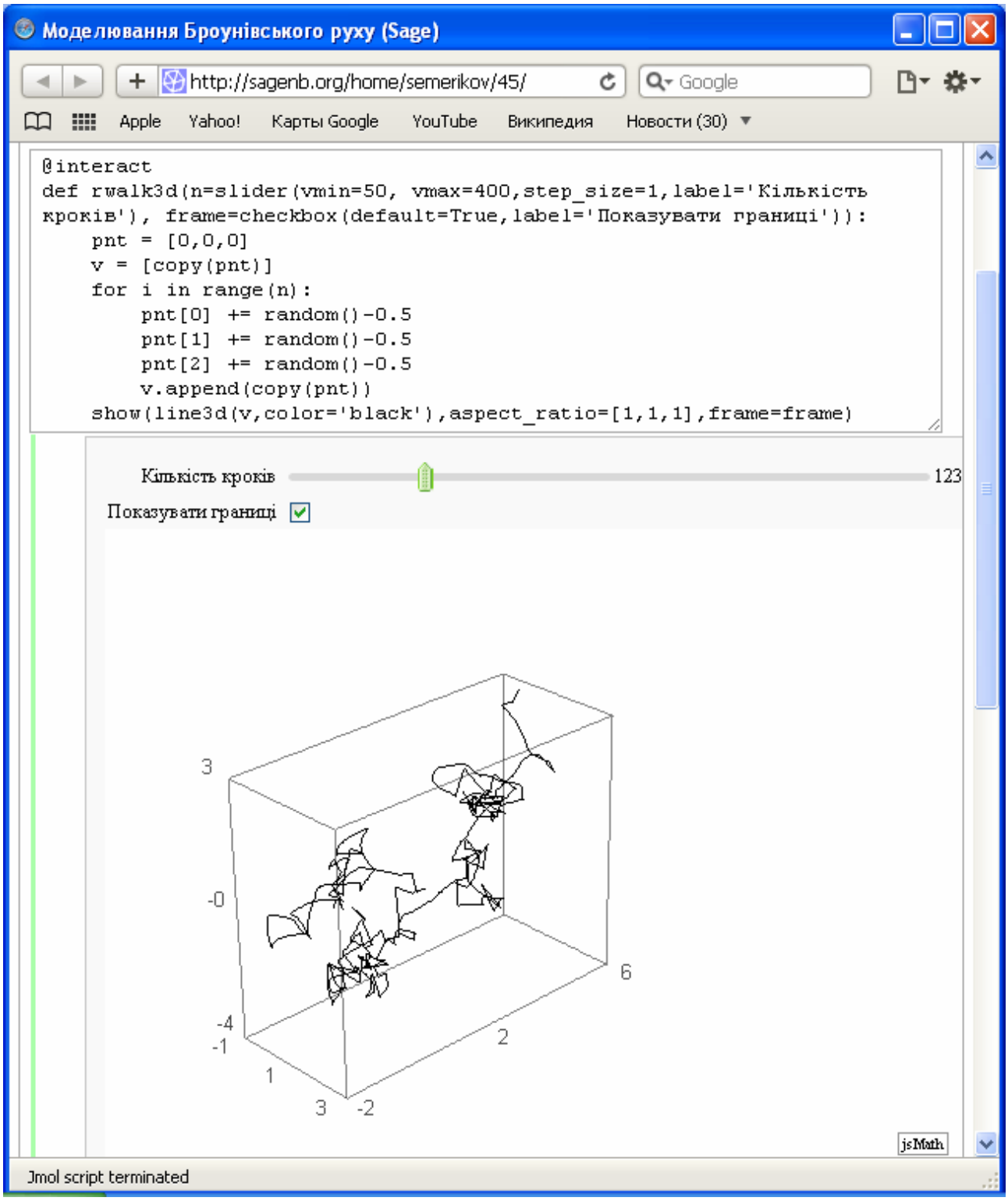


Рис. 3.31. Модулювання броунівського руху у середовищі SAGE

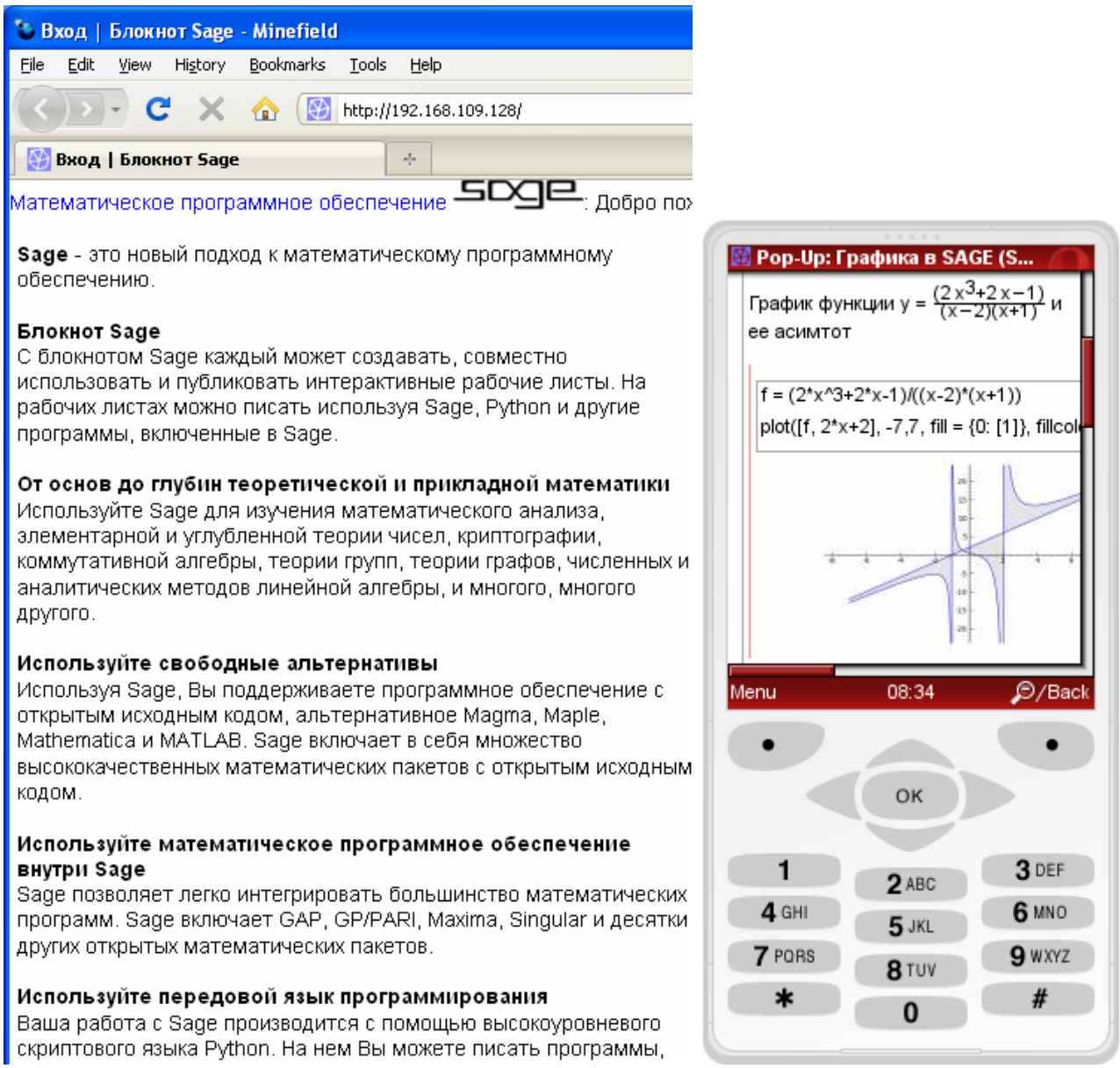


Рис. 3.32. Локалізована мобільна версія SAGE

3.4. Фундаменталізація змісту навчання інформатичних дисциплін

Стабілізація програмних засобів надає широкі можливості для варіювання програмного забезпечення навчання інформатичних дисциплін (замість штучної прив'язки до окремих програмних продуктів), що вимагає виділення в усіх курсах фундаментальної та варіативної складової.

В останніх роботах вітчизняних дослідників в галузі теорії та методики навчання інформатики тенденція пошуку фундаментальних інваріант, вивчення яких дозволило б абстрагуватися від конкретних інформаційних технологій і ресурсів, стає провідною. Так, Л.В. Гришко [65; 408] виділена фундаментальна

частина курсу основ програмування та розроблена варіативна частина, в якій підтримуються дві мови програмування та понад десять компіляторів в різних операційних системах. Дослідження Т.П. Кобильника [135; 132] спрямоване, з одного боку, на виділення фундаментальних основ інформатики (як математичної дисципліни), а з іншого – на розкриття можливостей застосування різних систем комп'ютерної математики.

Покажемо, як відбувається фундаменталізація змісту навчання, на прикладі дисциплін, що традиційно відносяться до технологічних: «Системне програмування», «Системне програмне забезпечення» та «Подіє-орієнтоване програмування».

3.4.1. Системне програмування

Як зазначає В.В. Гриншкун [63], підготовка вчителів з будь-яких дисциплін завжди базується на принципі, згідно з яким педагог завжди повинен бути фахівцем-професіоналом у своїй предметній галузі. Тому підготовку майбутніх вчителів інформатики як фахівців-професіоналів у галузі інформатики та інформаційних технологій (за напрямком підготовки 0403 «Системні науки та кібернетика») ніяк не можна вважати повноцінною без вивчення найбільш загальних методів системного програмування у сучасних операційних системах.

У п. 3.3.1 було показано, що сучасні операційні системи можуть бути описані в межах єдиного стандарту мобільних операційних систем POSIX, що надає можливість курс системного програмування, незважаючи на відносну прив'язку до специфіки роботи апаратного забезпечення комп'ютерів, побудувати таким чином, що його зміст залишається інваріантним стосовно постійного вдосконалення елементної бази комп'ютерів і розробки все більш досконалого програмного забезпечення. Тоді системні програми, проектування та розробка яких здійснюється студентами під час вивчення курсу, будуть працювати як на найсучасніших комп'ютерах, так і на комп'ютерах, вік яких вимірюється десятками років.

3.4.1.1. Основа стабілізації курсу. Основне призначення стандарту POSIX – зробити програми мобільними на рівні вхідної мови стосовно до зміни апара-

тно-програмної платформи. POSIX нейтральний стосовно системної архітектури та розрядності процесора. Це дуже важливий аспект мобільності програм.

Як було показано у попередньому пункті, стандарт POSIX не обмежений рамками Unix-середовища. Тому можна стверджувати, що дотримання стандарту POSIX полегшує перенесення програм практично на будь-яку операційну платформу.

У стандарті POSIX проведено поділ на обов'язкові та додаткові системні сервіси, причому обов'язкове ядро зроблене досить компактним. Для розгляду в курсі було вибрано в основному обов'язкові послуги; інші (серед яких вибиралися стабільні, наявні в багатьох історично сформованих реалізаціях) залучалися лише тоді, коли явно не вистачало функціональності для створення програм.

Основне призначення курсу – розгляд прийомів і методів використання стандартизованих службових програм і функцій. Не ставилася мета висвітлити всі тонкощі реалізації ОС, всі можливі коди помилок і т.п. Головне – відчути «дух» стандарту, навчитися застосовувати закладені в ньому мобільні засоби.

Значне місце в курсі як за обсягом, так і за роллю, відведено прикладам програм. Багато положень стандарту (пов'язані, наприклад, з опрацюванням помилкових ситуацій) розглядаються не в основному тексті, а в цих прикладах. За можливості вони компілювалися й виконувалися на кількох POSIX-сумісних апаратно-програмних платформах. Програму курсу наведено у додатку Г.

3.4.1.2. Засоби для обслуговування поняття користувача. В операційній системі, що відповідає стандарту POSIX, має бути передбачена підтримка *бази даних користувачів*, у якій про кожного з них зберігається принаймні наступні дані:

– ім'я користувача;

– числовий ідентифікатор користувача;

– числовий ідентифікатор групи, до якої входить користувач;

– початковий робочий каталог;

– початкова програма-оболонка користувача.

Над базою даних користувачів визначені операції пошуку за ідентифікатором або іменем користувача, реалізовані, відповідно, за допомогою функцій `getpwuid` і `getpwnam`.

Користувачів об'єднують в групи; кожний є членом принаймні однієї групи. Для груп, як і для користувачів, існує *база даних груп*, записи якої містять, як мінімум, наступні поля:

– ім'я групи;

– числовий ідентифікатор групи;

– список користувачів, яким дозволено ставати членами даної групи.

Функції для пошуку в базі даних груп – `getgrgid` і `getgrnam`.

Для отримання асоційованих з іменем користувача даних може бути використана службова програма `id`.

Про вхідне ім'я поточного користувача можна довідатися також за допомогою утиліти `logname` і функції `getlogin`.

Для зміни поточної групи користувача призначена службова програма `newgrp`.

Щоб довідатися, які користувачі та за якими терміналами в цей момент працюють у системі, можна скористатися службовою програмою `who`.

За допомогою утиліт `write`, `talk` і `mesg` можна в обмеженій формі організувати взаємодію між користувачами.

Базовим засобом забезпечення поштової взаємодії, відповідно до стандарту POSIX-2001, є службова програма `mailx`.

3.4.1.3. Засоби роботи з файлами. У трактуванні стандарту POSIX поняття *файлу* охоплює все, за допомогою чого можна зберігати, використовувати й/або поставляти дані. Файл має такі атрибути, як тип, ім'я й режим.

У стандарті зафіксовані наступні *типи файлів*:

– звичайний файл;

– каталог;

– канал;

– символьний спеціальний файл;

– блоковий спеціальний файл;

– символьне посилання;

– сокет.

Файли разом зі службовими даними, що зберігаються в об’єктах, які називаються дескрипторами файлів, поєднуються в ієрархічну структуру, що має назву файлової системи.

У межах файлової системи кожний файл має унікальний *ідентифікатор* (порядковий номер – він же номер файлу).

Відповідно до стандарту, з кожним файлом пов’язана принаймні наступні службові дані:

– режим – об’єкт, що містить біти режиму й тип файлу;

– числовий ідентифікатор власника-користувача;

– числовий ідентифікатор групи, якій належить файл.

По відношенню до конкретного файлу всі користувачі діляться на три категорії:

– власник файлу;

– члени групи, якій належить файл;

– інші користувачі.

Для кожної із цих категорій за режимом доступу визначаються *права на операції з файлом*, а саме:

– право на читання;

– право на запис;

– право на виконання (для каталогів – право на пошук).

Для виконання більшості операцій з файлами їх необхідно відкрити. Відкритому файлу відповідає *файловий дескриптор* – невід’ємне ціле число, унікальне в межах процесу й використовуване для доступу до файлу. Дескриптор є посиланням на опис відкритого файлу, де зберігається позиція файлового вказівника, його статус і режими доступу.

Для одержання даних про файли і файлові системи, а також для зміни їх атрибутів призначені наступні службові програми й функції:

– утиліта pwd і функція getcwd призначені для опитування абсолютного маршрутного імені поточного каталогу;

– утиліта ls і функції сімейства stat (stat, fstat, lstat) призначені для надання даних про файли;

– утиліта df і функції fstatvfs і statvfs призначені для надання даних про файлові системи;

– службова програма du призначена для надання даних про сумарний обсяг простору, зайнятого ієрархіями файлів;

– утиліта cd і функція chdir призначені для зміни поточного каталогу;

– утиліти chown і chmod, функції chown, fchown, chmod, fchmod призначені для зміни таких атрибутів файлів, як власник і режим доступу;

– утиліта touch призначена для модифікації часу останнього доступу або зміни файлу.

Для створення звичайних файлів використовується функція creat, для створення каталогів – утиліта mkdir і однойменна функція, для створення каналів – утиліта mkfifo і однойменна функція.

Нові посилання на файл (жорсткі або символьні) створюються за допомогою службової програма ln, а також функцій link і symlink.

Для вилучення файлів служать утиліти rm і rmdir, функції unlink, rmdir і remove.

Копіювання файлів виконується за допомогою службової програми cp, переміщення (перейменування) – за допомогою програми mv або функції rename.

Для обходу файлової ієрархії й систематичне опрацювання її елементів призначена утиліта find.

Однією з форм обходу й опрацювання файлової ієрархії можна вважати архівування. В стандарті POSIX передбачено для цього службову програму pax.

У стандарті POSIX-2001 виділені дві основні групи функцій для обслуговування операцій введення/виведення:

– функції нижнього рівня, при роботі з якими використовуються цілочисельні файлові дескриптори;

– функції більш високого рівня, за допомогою яких здійснюється буферизоване введення/виведення із застосуванням потоків.

Для відкривання файлів і формування нових описів відкритих файлів, файлових дескрипторів і потоків служать функції нижнього рівня `open` і `pipe`, а також функції буферизованого введення/виведення `fopen`, `fdopen`, `freopen`.

Для закривання файлів призначені функції `close` і `fclose`.

Досить корисною із практичної точки зору є функція створення й відкривання тимчасових файлів `tmpfile`.

Для читання даних з файлу призначені функції `read` і `fread`, для запису – функції `write` і `fwrite`.

Символьні посилання доводиться читати особливим чином, за допомогою функції `readlink`.

Для символьного буферизованого введення/виведення призначені функції `fgetc` і `fputc`, рядки рекомендується вводити за допомогою функції `fgets`, а виводити за допомогою функцій `fputs` і `puts`.

Індикатор поточної позиції файлу (файловий вказівник) може бути опитаний або переміщений за допомогою функції нижнього рівня `lseek`, а також функцій буферизованого введення/виведення `fseek`, `ftell`, `ftello`, `fgetpos`, `fsetpos`, `rewind`.

Для виконання різноманітних операцій над відкритим файлом призначена функція `fcntl`.

Особливий клас операцій з файлами зі своєю системою понять становлять блокування, хоча вони також оформляються як команди функції `fcntl`.

Функції `setbuf`, `setvbuf` і `fflush` призначені для управління буферами потоків.

3.4.1.4. Засоби опрацювання структурованих даних. Найпростіший різновид структурованих даних – *текстові файли*. У свою чергу, найпростішою опе-

рацією з файлами є їх виведення у стандартний потік виведення. Для цього служить утиліта `cat`.

Перегляд великих текстових файлів зручно здійснювати за допомогою службової програми `more`.

Для перегляду нетекстових файлів рекомендується службова програма `od`.

Щоб переглянути початок файлу, слід скористатися службовою програмою `head`.

За допомогою службової програми `tail` можна переглянути кінець файлу.

Службова програма `pr` є фільтром для друкування й оформлення сторінок.

Для підрахунку числа символів, слів і рядків у файлах служить утиліта `wc`.

За допомогою службової програми `sort` залежно від заданих опцій виконується одна з трьох можливих дій:

– впорядковуються рядки всіх вхідних файлів із записом результату у вихідний файл;

– здійснюється злиття всіх вхідних (попередньо впорядкованих) файлів і записується результат у вихідний файл;

– перевіряється, чи дійсно вихідний файл є впорядкованим.

За допомогою утиліти `diff` порівнюється вміст вхідних файлів і видається на стандартне виведення список змін, які необхідно зробити, щоб перетворити один файл в іншій.

Якщо потрібно перевірити на співпадіння два файли, доцільно скористатися більш простою і швидкою службовою програмою `cmp`.

Ще одним засобом виявлення розходжень (і збігів) текстових файлів є службова програма `comm`.

Для контролю цілісності файлів призначена службова програма `cksum`.

Поняття регулярного виразу (РВ) – одне з найважливіших для програм опрацювання текстових файлів. Відповідно до стандарту POSIX-2001, *регулярний вираз* – це шаблон, що служить для вибірки певних ланцюжків символів з множини подібних ланцюжків. Говорять, що обрані ланцюжки задовольняють (успішно зіставляються з) РВ.

Розрізняють базові (БРВ) і розширені (РРВ) регулярні вирази.

Найбільш часто механізм регулярних виразів використовується в службовій програмі `grep`.

У командних файлах опрацювання текстів часто виконується за допомогою потокового редактора `sed`.

Ще один популярний засіб опрацювання текстових файлів – службова програма `awk`. За її допомогою виконуються програми, написані однойменною мовою програмування.

На рівні функцій робота з регулярними виразами підтримується сімейством `regex`: `regcomp`, `regexec`, `regfree`, `regerror`.

Ідейно простий, але досить потужний і корисний засіб опрацювання текстових файлів – службова програма перетворення символів `tr`.

Службова програма `uniq` надає можливість звести однакові рядки, що розташовані підряд, до одного.

В стандарті POSIX-2001 передбачені *аналоги реляційних операцій для текстових файлів*:

`cut` – виконується операція проекції відношень;

`paste` – здійснюється горизонтальне об'єднання;

`join` – виводиться на стандартне виведення результат об'єднання двох відношень.

Каталог також можна розглядати як набір структурованих даних. Опрацювання каталогів починається з їхнього відкривання. Для цього призначена функція `opendir`.

Після відкривання поточним стає перший елемент каталогу. Якщо надалі знадобиться знову позиціонуватися на перший елемент, можна скористатися функцією `rewinddir`.

Для читання елементів каталогу призначена функція `readdir`.

Після завершення роботи з каталогом його слід закрити за допомогою функції `closedir`.

Нерідко поєднується читання елементів каталогу й зіставлення із шаблоном імен файлів. Подібне зіставлення реалізується за допомогою функції `fnmatch`.

Для генерації маршрутних імен, що задовольняють заданому шаблону, в стандарті POSIX-2001 передбачено функцію `glob`.

Виділення з маршрутного імені файлу простого імені виконується за допомогою утиліти `basename`. Маршрутний префікс виділяється за допомогою службової програми `dirname`.

3.4.1.5. Процеси. Відповідно до стандарту POSIX-2001, *процес* – це адресний простір разом із потоками управління, що в ньому перебігають, а також системними ресурсами, які пов'язані з цими потоками.

Кожний процес характеризується цілим рядом атрибутів. Найважливішим серед них є *ідентифікатор процесу* – додатне ціле число, що однозначно ідентифікує процес протягом часу його існування.

Процес, від якого походить даний, називається батьківським.

З кожним процесом асоціюється ідентифікатор користувача, від імені якого було створено процес. Цей атрибут називається реальним ідентифікатором користувача процесу.

Для визначення прав доступу процесу (у тому числі прав доступу до файлів) використовують діючі (з англ. effective) ідентифікатори користувача й групи, які в загальному випадку можуть відрізнятися від реальних.

Перебіг процесу визначається за виконуваною в його рамках програмою.

Для отримання даних про процеси служить утиліта `ps`.

Отримання ідентифікаторів процесу, батьківського процесу й групи процесів виконується за допомогою функцій `getpid`, `getppid` і `getpgrp`.

Група процесів – це сукупність процесів з узгодженим доставлянням сигналів. Для встановлення ідентифікатора групи процесів призначена функція `setpgid`.

Для створення сеансу і встановлення ідентифікатора групи процесів призначена функція `setsid`.

Доступ до реальних і діючих ідентифікаторів користувача й групи виконуваного процесу здійснюється за допомогою функцій `getuid`, `geteuid`, `getgid`, `getegid`.

Функція `getgroups` призначена для одержання ідентифікаторів додаткових груп виконуваного процесу.

Перевизначити діючий ідентифікатор користувача виконуваного процесу можна за допомогою функцій `setuid` і `seteuid`. Аналогічні функції для перевизначення ідентифікаторів групи процесу називаються `setgid` і `setegid`.

Опитування та зміна маски режиму створення файлів виконуваного процесу здійснюється за допомогою службової програми `umask` і однойменної функції.

Нові процеси створюються за допомогою функції `fork`.

Зазвичай в процесі-нащадку, з використанням функції сімейства `exec`, підмінюється програма, за якою визначається перебіг процесу, і передається їй управління та список аргументів. До числа функцій цього сімейства належать `execl`, `execv`, `execle`, `execve`, `execlp`, `execvp`.

В батьківському процесі реалізується очікування завершення процесів-нащадків: до нього доходять дані про статус завершення від функцій сімейства `wait` – `wait` і `waitpid`.

В процесі може бути реалізоване власне завершення за допомогою функцій сімейства `exit` – `exit`, `_Exit`, `_exit`.

За допомогою функції `atexit` можна зареєструвати функції, які будуть викликатися, якщо процес завершується зверненням до `exit` або після завершення програми.

Для припинення процесів ззовні призначена службова програма `kill`.

3.4.1.6. Засоби обміну даними між процесами реалізують високопродуктивний, детермінований обмін даними між процесами в межах однієї системи.

До числа найбільш простих і водночас найчастіше використовуваних засобів обміну даними між процесами належать *канали*, що представляють собою

файли відповідного типу. В стандарті POSIX-2001 розрізняються іменовані й безіменні канали.

Обміну даними між процесами через канал може бути встановлений у такий спосіб: в одному із процесів за допомогою функцій `popen` або `pipe` створюється канал і передається до іншого відповідний відкритий файловий дескриптор. Після цього між процесами відбувається обмін даними через канал за допомогою функцій `read` і `write`.

У порівнянні з `pipe` функція `popen` є більш високорівневою. За її допомогою виконується одразу кілька операцій: породжується процес, забезпечується виконання заданої команди в його межах, організовується канал між виконуваним та породженим процесами й формуються необхідні потоки для цього каналу.

Канал залишається відкритим доти, поки не буде викликана функція `pclose`.

Відповідно до стандарту POSIX-2001, під *сигналом* розуміється механізм, за допомогою якого в процесі або потоці управління генерується повідомлення про деяку подію, що відбулася в системі, або реалізується вплив цієї події на перебіг процесу.

У кожному процесі визначені реакції на всі передбачені в системі сигнали.

У кожному потоці управління є *маска сигналів*, за якою визначається набір блокованих сигналів.

Із сигналом можуть бути асоційовані дії одного із трьох типів:

– виконати дії, пов’язані з сигналом;

– ігнорувати сигнал;

– опрацювати сигнал, виконавши власну функцію.

До засобів генерації сигналів відносяться службова програма `kill` й однойменна функція.

Із процесу (потоку управління) можуть посилатися сигнали до цього ж процесу за допомогою функції `raise`.

За допомогою функції `abort` викликається аварійне завершення процесу.

Отримати доступ й змінити спосіб опрацювання сигналів можна за допомогою функції `sigaction`; на рівні командного інтерпретатора – за допомогою спеціальної вбудованої команди `trap`.

До технічних аспектів можна віднести роботу з наборами сигналів. Її здійснюють за допомогою функцій `sigemptyset`, `sigfillset`, `sigaddset`, `sigdelset`, `sigismember`.

Функція `sigprocmask` призначена для доступу або зміни маски сигналів процесу, якою визначається набір блокованих сигналів.

За допомогою функції `sigpending` можна з'ясувати набір блокованих сигналів, що очікують доставляння до виконуваного процесу (потоку управління). Дочекатися появи подібного сигналу можна за допомогою функції `sigwait`.

При зверненні до функції `pause` очікується сигнал.

За допомогою функції `sigsuspend` поточна маска сигналів виконуваного процесу спочатку замінюється, а потім відбувається перехід процесу у стан очікування. Як правило, парою функцій `sigprocmask` і `sigsuspend` обрамляють критичні секції.

Черги повідомлень, семафори й поділювані сегменти пам'яті віднесені до необов'язкової частини стандарту POSIX-2001 (X/Open-розширення системного інтерфейсу – XSI).

Кожна черга повідомлень, набір семафорів і поділюваний сегмент однозначно ідентифікується додатним цілим числом, що повертається як результат звернення до функцій `msgget`, `semget` і `shmget`.

Для одержання ідентифікаторів засобів обміну даними між процесами використовується ще одна сутність – ключ, для його генерації призначена функція `ftok`.

Отримати доступ до наявних у даний момент у системі (поточних) засобів обміну даними між процесами можна за допомогою службової програми `ipcs`.

Для вилучення із системи поточних засобів обміну даними між процесами призначена службова програма `ipcrm`.

За допомогою механізму *черг повідомлень* забезпечується обмін даними між процесами дискретними порціями – повідомленнями. При перебігу процесу над повідомленнями виконуються дві основні операції – приймання і відправлення.

Для роботи із чергами повідомлень у стандарті POSIX-2001 передбачені наступні функції: `msgget` – одержання ідентифікатора черги повідомлень, `msgctl` – управління чергою повідомлень, `msgsnd` – відправлення повідомлення, `msgrcv` – приймання повідомлення.

Відповідно до стандарту POSIX-2001, *семафор* – це мінімальний примітив синхронізації, що служить основою для більш складних механізмів синхронізації.

Семафор визначається цілим числом у діапазоні від 0 до 32767.

Семафори створюються за допомогою функції `semget` і опрацьовуються за допомогою функції `semop` певними наборами (масивами), причому операції над наборами для програм є атомарними. У рамках групових операцій для будь-якого семафора з набору можна збільшити значення, зменшити значення, дочекатися обнуління.

Для виконання різних дії над семафорами призначена функція `semctl`.

У стандарті POSIX-2001 *поділюваний об'єкт пам'яті* визначається як об'єкт, що представляє собою пам'ять, яка може бути паралельно відображена в адресний простір більш ніж одного процесу.

Робота з поділюваною пам'яттю починається з того, що одному із процесів за допомогою функції `shmget` створюється поділюваний сегмент.

Для одержання доступу до поділюваного сегмента його потрібно приєднати за допомогою функції `shmat`, тобто розмістити сегмент у віртуальному просторі процесу. Після приєднання, відповідно до прав доступу, із процесів можуть читатися дані із сегмента й записуватися до нього (можливо, з синхро-

нізацією перебігу за допомогою семафорів). Коли поділюваний сегмент стає непотрібним, його слід від'єднати за допомогою функції `shmdt`.

Передбачено можливість виконання дій над поділюваними сегментами (функція `shmctl`).

3.4.1.7. Загальний термінальний інтерфейс. У стандарті POSIX-2001 *термінал* або термінальний пристрій – це спеціальний символьний файл, що задовольняє специфікаціям загального термінального інтерфейсу.

З кожним термінальним пристроєм асоційовані черги введення й виведення.

Введення може відбуватися в канонічному й неканонічному режимах. Канонічний режим означає обов'язкову буферизацію введення в системі, а також природне опрацювання символів вибою й знищення рядка.

У неканонічному режимі вхідні дані не піддаються попередньому опрацюванню.

Деякі символи відіграють спеціальну роль при введенні та виведенні. Наприклад, при введенні символу `intr` генерується сигнал переривання (`sigint`), що надсилається до всіх процесів, які перебігають під управлінням терміналу.

За допомогою службової програми `tty` на стандартне виведення видається ім'я термінала, відкритого як стандартне введення.

Довідатися, чи асоційований відкритий файловий дескриптор з термінальним пристроєм, а також одержати ім'я цього пристрою можна за допомогою функцій `isatty` і `ttyname`.

Кожний термінал має ряд характеристик, про які можна дізнатися або змінити, використовуючи утиліту `stty`.

Деякі, хоча й досить обмежені, можливості управління терміналами можна здійснювати за допомогою службової програми `tput`.

Опитування й зміна характеристик термінала на рівні функцій поділені на два сімейства: `tc*` і `cf*`. До першого входять функції `tcgetattr` – опитування, `tcsetattr` – зміна, `tcflow` – припинення або поновлення термінального введення/виведення, `tcflush` – скидання черги введення, `tcdrain` – очікування

фізичного закінчення виведення, `tcsendbreak` – розрив з'єднання, `tcgetpgrp` – одержання ідентифікатора асоційованої з терміналом групи процесів переднього плану, `tcsetpgrp` – встановлення ідентифікатора групи, `tcgetsid` – опитування ідентифікатора групи процесів лідера сеансу, який перебігає під управлінням терміналу.

Функції сімейства `cf*` – `cfgetispeed`, `cfgetospeed`, `cfsetispeed`, `cfsetospeed` – служать для вибірки/зміни даних про швидкості термінального уведення/виведення.

Для одержання маршрутного імені термінала служить функція `ctermid`.

3.4.1.8. Опитування характеристик хостів. *Хост* – комп'ютер (вузол мережі), під'єднаний до мережі Інтернет. Кожен хост має унікальну IP-адресу.

Найбільш загальні характеристики хосту можна одержати за допомогою службової програми `uname` і однойменної функції.

Звернення до функція `gethostname` надає можливість отримати ім'я хосту.

Значення конфігураційних параметрів – найважливіша характеристика хосту, а налагодження на конфігурацію цільової системи – обов'язковий елемент мобільного програмування додатків.

До числа найбільш важливих характеристик належать `_POSIX_VERSION` (підтримувана версія системного інтерфейсу для мови С стандарту POSIX) і `_POSIX2_VERSION` (підтримувана версія інтерфейсу до системних сервісів на рівні командної мови й службових програм).

За допомогою окремої групи констант описуються підтримувані необов'язкові засоби стандарту POSIX-2001. Серед них `_POSIX_IPV6` (в реалізації підтримується IP v6), `_POSIX_REGEXP` (в реалізація підтримується опрацювання регулярних виразів), `_POSIX_SHELL` (в реалізація підтримується стандартний командний інтерпретатор), `_POSIX_V6_ILP32_OFF32` (в реалізація надається середовище компіляції С-програм з 32-бітними типами `int`, `long`, `off_t` і такими ж вказівниками) і т.д.

За допомогою трьох константи задаються номери файлових дескрипторів для стандартних потоків введення (`STDIN_FILENO`=0), виведення (`STDOUT_FILENO`=1) і протоколювання (`STDERR_FILENO`=2).

В стандарті визначаються мінімально припустимі значення для різного роду лімітів на кількість і розміри, які повинні підтримуватися в реалізаціях й бути доступними програмам.

При створенні мобільних додатків не слід покладатися на якісь конкретні значення лімітів і вимагати більше ресурсів, ніж передбачено мінімально припустимими значеннями.

Для отримання значень системних параметрів під час виконання програми призначена службова програма `getconf`, а також функції `sysconf`, `confstr`, `fpathconf` і `pathconf`, за допомогою першої з яких опитуються обмежуючі конфігураційні параметри, що мають числові значення, за допомогою другої отримуються конфігураційні ланцюжки символів, а за допомогою двох останніх – конфігураційні значення, що відносяться до файлів.

3.4.1.9. Мережні засоби. В стандарті POSIX-2001 *мережа* означається як сукупність взаємозалежних хостів. Під мережною адресою розуміється доступний у межах мережі ідентифікатор, що використовується для позначення мережних елементів.

Надання мережної адреси мережному елементу називається зв'язуванням, або прив'язуванням, а зворотна дія – звільненням або скасуванням прив'язування.

Найчастіше в якості мережних елементів виступають апаратні мережні інтерфейси, за допомогою яких передаються та приймаються дані, однак з таким інтерфейсом, як шлейфовий (loopback), ніякої апаратури не асоційовано.

Дані передаються в мережі у вигляді послідовності октетів (восьмибітових беззнакових кодів). Якщо деякий елемент даних (наприклад, адреса або номер порту) складається більш, ніж із восьми бітів, для його передавання і зберігання потрібно кілька октетів. Мережним називається порядок октетів (байтів),

при якому перший (з найменшою адресою) октет містить старші (найбільш значимі) біти.

При обміні даними між процесами в ролі мережних елементів виступають *сокети*, які в стандарті POSIX-2001 трактуються як окремий тип файлів.

Під *адресою* сокету розуміють структуру, що включає ідентифікатор адресного сімейства й специфічні для даного сімейства адресні дані.

Адресне сімейство відповідає певному середовищу обміну даними між процесами. В стандарті POSIX-2001 визначено три таких сімейства: `AF_UNIX` (обмін даними між процесами в межах однієї системи), `AF_INET` (обміну даними за протоколами IP v4), `AF_INET6` (обміну даними за протоколами IP v6).

У межах кожного адресного сімейства можуть існувати сокети кількох типів. В стандарті POSIX-2001 передбачено чотири типи: `SOCK_STREAM` (надійні, упорядковані, повнодуплексні потоки октетів у режимі із встановленням з'єднання), `SOCK_SEQPACKET` (аналог `SOCK_STREAM` з додатковим збереженням границь між записами), `SOCK_DGRAM` (передавання даних у вигляді датаграм у режимі без встановлення з'єднання), `SOCK_RAW` (аналог `SOCK_DGRAM` з додатковою можливістю доступу до протокольних заголовків та інших даних нижнього рівня).

Для кожного адресного сімейства кожний тип сокету може підтримуватися одним або кількома протоколами.

Загальна логіка роботи із сокетами полягає в наступному. Сокети створюються за допомогою функції `socket`, до якої передаються адресне сімейство, тип сокету й протокол, а одержується відкритий файловий дескриптор. Потім за допомогою функції `bind` сокету надається локальна адреса. Якщо сокет орієнтований на режим із встановленням з'єднання, то його слід позначити як готового приймати з'єднання, для чого знадобиться функція `listen`. Реальне приймання з'єднань виконується за допомогою функції `accept`, що створює для кожного з них новий сокет за образом та подобою «слухаючого». У свою чергу, на хості, що ініціює з'єднання, викликається функція `connect` (у режимі без

встановлення з'єднання за допомогою функції `connect` можна вказати адресу відправника).

Для приймання даних, що надійшли до сокету, можна скористатися універсальною функцією низькорівневого введення/виведення `read` або спеціалізованим сімейством функцій `recv*`, а для передавання – функцією `write` або сімейством `send*`. Крім того, за допомогою функцій `select` та `poll` можна опитати наявність даних для приймання або можливість відправлення чергової порції даних.

Завершується обмін даними між хостами зверненням до функції `shutdown`.

Дані про хости як вузли мережі зберігаються в мережній базі, послідовний доступ до якої обслуговується функціями `sethostent`, `gethostent` і `endhostent`.

За допомогою функції `getaddrinfo` за іменем вузла мережі або імені мережного сервісу отримуються адреси набору сокетів і асоційовані дані, що дає можливість створити сокет для звертання до заданого сервісу.

Функція `freeaddrinfo` носить технічний характер, і призначення для звільнення пам'яті, зарезервованої при зверненні до функції `getaddrinfo`.

Функцію `getnameinfo` можна вважати оберненою до `getaddrinfo`. За її допомогою за адресою сокету отримуються дані про ім'я вузла й сервісу.

Технічну роль відграють і функції перетворення IP-адрес із текстового подання в числове й навпаки: `inet_addr`, `inet_ntoa`, `inet_pton`, `inet_ntop`. За допомогою перших двох опрацьовуються тільки адреси IP v4: за `inet_addr` перетворюється текстовий ланцюжок на цілочисельне значення, придатне для використання в якості IP-адреси, за `inet_ntoa` виконується обернене перетворення. Друга пара функцій по суті аналогічна до першої, але має більш загальний характер, тому що призначення для перетворення адрес у форматі IP v6.

Для перетворення значень типів `uint16_t` і `uint32_t` з хостового порядку байт у мережний служать функції `htons` і `htonl`; функції `ntohs` і `ntohl` при-

значені для здійснення оберненого перетворення.

Поряд з базою даних хостів підтримується база даних мереж з аналогічною логікою роботи й набором функцій: `setnetent`, `getnetent`, `getnetbyaddr`, `getnetbyname`, `endnetent`.

За допомогою функції `getnetent` здійснюється послідовний доступ до бази, за допомогою `getnetbyaddr` здійснюється пошук за адресним сімейством й номером мережі, а за допомогою `getnetbyname` вибирається мережа із заданим (офіційним) ім'ям.

Такий самий програмний інтерфейс надається в базі даних мережних протоколів: `setprotoent`, `getprotoent`, `getprotobyname`, `getprotobynumber`, `endprotoent`.

Ще один прояв тієї ж логіки роботи – база даних мережних сервісів: `setservent`, `getservent`, `getservbyname`, `getservbyport`, `endservent`.

Для створення сокетів, крім функції `socket`, може бути використана функція `socketpair`, за допомогою якої створюються пари сокетів із встановленим між ними з'єднанням. Вона звичайно використовується для адресного сімейства `AF_UNIX`; підтримка інших сімейств не гарантується.

Дізнатися про отриману локальну адресу (її іноді називають ім'ям сокету) можна за допомогою функції `getsockname`.

Із сокетами можуть бути асоційовані опції, що впливають на їхнє функціонування. Опитати або змінити значення цих опцій можна за допомогою функцій `getsockopt` і `setsockopt`.

Функція `getpeername` призначена для отримання адреси (імені) сокету, з яким встановлене з'єднання.

Після прив'язування сокету до локальної адреси й, можливо, встановлення з'єднання та задання значень опцій, можна приступати до відправлення та приймання даних через сокет. Для цього служать функції `recvfrom`, `recv`, `recvmsg`, `sendto`, `send`, `sendmsg`.

3.4.1.10. Управління часом. Відповідно до стандарту POSIX, за початок відліку часу приймається нуль годин, нуль хвилин, нуль секунд першого січня 1970-го року всесвітнього часу.

Всесвітнім називають поясний час нульового годинного пояса, що являє собою місцевий середній сонячний час гринвіцького меридіана. За стандартну одиницю виміру астрономічного часу в POSIX-2001 прийнята секунда.

Годинниками називають програмні або апаратні об'єкти, що можуть бути використані для вимірювання часу. Покази годинника можна запитати й, можливо, установити (у припустимих для них межах).

Реальним (або астрономічним) називається час, вимірюваний за системними годинниками безвідносно до того, який процес (потік управління) в даний час перебігає.

Під часом виконання (процесорним часом) розуміють час, затрачуваний на перебіг процесу (потоку управління), включаючи використовувані системні сервіси.

Годинником процесорного часу називається годинник, за допомогою якого вимірюється час перебігу конкретного процесу або потоку управління.

Під віртуальним часом процесу розуміють час, вимірюваний системними годинниками, поки перебіг процесу не закінчився.

Таймер – це механізм, призначений для введення до системи управління процесом (потоком управління) повідомлення про закінчення заданого проміжку часу (інтервальний таймер) або про досягнення (перевищення) годинниками заданих показань (абсолютний таймер). Відповідна подія називається спрацьовуванням таймера.

Таймером процесорного часу називається таймер, асоційований з годинниками процесорного часу.

Завести (зарядити) – значить запустити таймер, за допомогою якого контролюється плин часу, що надає можливість введення до системи управління процесом повідомлення (сигнал) про настання заданого моменту.

Таймер знімається із зведення (розряджається), коли припиняється контроль за плином часу за допомогою даного таймера.

Найпростішим засобом запиту та зміни поточних дати і часу є службова програма `date`.

Запит на отримання поточного часу (у секундах від початку відліку) виконує за допомогою функції `time`.

Якщо тип `time_t` реалізований як 32-розрядне ціле зі знаком, то в 2038-му року наступить переповнення (так звана проблема 2038-го року).

З функцією `time` асоційована функція `difftime`, за допомогою якої обчислюється різниця в секундах між двома моментами часу.

Довідатися про поточний час із більшою точністю можна за допомогою функції `gettimeofday`.

У необов'язкову частину стандарту POSIX-2001, за якою регламентується робота з таймерами, входять функції, за допомогою яких можна опитати й встановити покази заданих годинників, а також довідатися про їхню точність: `clock_gettime`, `clock_settime`, `clock_getres`.

Годинники можуть бути загальносистемними, тобто доступними для всіх процесів, або стосуватися лише певного процесу. Всі реалізації повинні підтримувати загальносистемні годинники реального часу, за допомогою яких вимірюється час від початку відліку та які мають ідентифікатор `clock_realtime`.

В стандарті POSIX-2001 передбачено кілька способів подання даних про час. Для виконання перетворень між різними поданнями даних про час служать функції `gmtime`, `localtime`, `mktime`, `strftime`, `strptime`, `getdate`.

Для врахування даних про годинний пояс і сезонні виправлення використовуються зовнішні змінні `tzname`, `timezone`, `daylight`, значення яких встановлюється за змінною оточення `TZ` за допомогою функції `tzset`.

Базовим засобом для роботи з годинниками процесорного часу є функція `clock`. При зверненні до неї повертає як результат видається процесорний час, витрачений на перебіг процесу з якогось моменту, що залежить від реалізації й пов'язаного тільки з його (процесу) запуском.

Щоб перевести час, що отримується при зверненні до функції clock, у секунди, його слід поділити на константу CLOCKS_PER_SEC, що визначена рівною 1 мільйону.

Для вимірювання часу виконання простої команди можна скористатися службовою програмою time.

Більш розвинену функціональність має спеціальна вбудована в shell команда times, за якою видається на стандартне виведення час, витрачений на виконання операцій командного інтерпретатора і перебіг породжених від нього процесів.

Реалізація утиліт time і times спирається на функцію times, за якою опитуються дані про час перебігу процесу й породжених процесів. За допомогою функції times вимірюється час в тактах. Відповідно, для переведення результатів, отриманих за допомогою times, у секунди їх потрібно ділити на sysconf(_SC_CLK_TCK), а не на CLOCKS_PER_SEC.

Пряме маніпулювання годинниками процесорного часу можливо при зверненні до функції clock_getcpuclockid, що дозволяє з'ясувати їх ідентифікатори.

Для зміни асоційованих з файлами даних про час служить функція utime.

Функція sleep дозволяє призупинити виконання процесу (потоку управління) на задане число секунд.

Більш сучасний аналог цієї функції, nanosleep, дозволяє позбутися від обмеження на максимальну тривалість припинення виконання й одержати можливість задавати цю тривалість із значно вищою точністю.

Описувані далі засоби для роботи з інтервальними таймерами входять у необов'язкову частину стандарту POSIX-2001 – XSI.

В реалізації кожному процесу повинні надаватися принаймні три інтервальні таймери, що мають наступні ідентифікатори: ITIMER_REAL (таймер реального часу, за допомогою якого генерується сигнал SIGALRM), ITIMER_VIRTUAL (таймер віртуального часу процесу, за допомогою якого генерується сигнал

SIGVTALRM), ITIMER_PROF (таймер профілювання, за допомогою якого генерується сигнал SIGPROF).

Відповідно до стандарту POSIX-2001 інтервальні таймери обслуговуються за допомогою функцій getitimer і setitimer. При зверненні до функції getitimer запам'ятовуються поточні характеристики таймера, а при зверненні до setitimer зводиться або знімається таймер із зведення, встановлюються нові характеристики й запам'ятовуються попередні.

До інтервальних таймерів відноситься функція alarm, що надає можливість в ході перебігу процесу передбачити подання сигналу SIGALRM через задане число секунд реального часу.

3.4.1.11. Мовно-культурне середовище. Відповідно до стандарту POSIX-2001, мовно-культурне середовище – це частина оточення користувача, що залежить від мовних і культурних узгоджень.

Під налагодженням на мовно-культурне середовище розуміють процес формування даних, специфічних для підтримки конкретних природних мов, місцевих налаштувань і кодувань символів. Іноді подібний процес називають *локалізацією*, на противагу *інтернаціоналізації* – процесу підготовки додатків, придатних для налаштування на різні мовно-культурні середовища.

Мовно-культурне середовище формується з даних кількох іменованих категорій, кожна з яких призначена для управління певними аспектами функціонування компонентів системи. Імена й призначення категорій відповідають наступним змінним оточення: LC_CTYPE (класифікація символів, перетворення регістра), LC_COLLATE (порядок алфавітного порівняння символів), LC_MONETARY (форматування грошових величин), LC_NUMERIC (форматування числових величин), LC_TIME (формати дати й часу), LC_MESSAGES (формати повідомлень).

Категорії діляться на дрібніші елементи, засобом іменування яких служать ключові слова.

У кожній реалізації визначені одне або кілька мовно-культурних середовищ. Підтримка POSIX-середовищ з іменами-синонімами «POSIX» і «C» є обов'язковою.

В стандарті POSIX-2001 передбачені дві змінні програмного оточення, за якими визначаються всі категорії мовно-культурного середовища: `LC_ALL` (значення цієї змінної враховується в першу чергу) та `LC_LANG` (її значення враховується в останню чергу).

Загальна логіка використання засобів інтернаціоналізації та локалізації полягає в наступному. Користувач, надаючи відповідних значень описаним вище змінним програмного оточення, ідентифікує своє мовно-культурне середовище. При перебігу програми дані про цільове середовище можуть бути тримані за допомогою функції `setlocale`.

При створенні нового мовно-культурного середовища його визначення у вихідному форматі міститься у файлах, які повинні бути опрацьовані за допомогою утиліти `localedef`. Для одержання відомостей про мовно-культурні середовища призначена службова програма `locale`. Функція `setlocale` служить для встановлення або опитування всього мовно-культурного середовища процесу або його окремих категорій. Одержати детальні відомості про категорії `LC_MONETARY` і `LC_NUMERIC` поточного мовно-культурного середовища можна за допомогою функції `localeconv`.

Для перетворення грошових величин у ланцюжок символів відповідно до налаштувань поточного мовно-культурного середовища можна скористатися функцією `strfmon`, що входить до XSI-розширення стандарту POSIX-2001.

Якщо потрібно одержати детальні відомості про всі аспекти мовно-культурного середовища, слід скористатися функцією `NL_LANGINFO`, також віднесену в стандарті POSIX-2001 до розширення XSI.

За допомогою функції `strerror` номери (коди) помилок перетворюються у текстові повідомлення, що залежать від мовно-культурного середовища.

Розвинені засоби для роботи з каталогами повідомлень (діагностичних, інформаційних), що використовуються в програмах, винесені в розширення XSI стандарту POSIX-2001. Ідея полягає в тому, щоб в інтернаціоналізованих програмах фігурували не самі повідомлення, а їх ідентифікатори в каталозі, який для кожного мовно-культурного середовища може бути своїм.

В стандарті не фіксується формат каталогів повідомлень, але пропонується службова програму `gencat` для їх генерації за вхідним описом.

У прикладній програмі робота з каталогами повідомлень здійснюється за допомогою функцій `catopen`, `catgets` і `catclose`. При зверненні до функції `catopen` відкривається каталог повідомлень і повертається його дескриптор, що потім використовується у виклику `catgets` для читання повідомлення. При зверненні до функції `catclose` каталог повідомлень закривається.

3.4.1.12. Приклад 1: реалізація лабораторного практикуму з системного програмування в POSIX-системах засобами Free Pascal. Однією зі складових системного програмування є сукупність програмних засобів інтерфейсу програміста для роботи з операційною системою. Інтерфейс системних викликів ОС UNIX відрізняється стабільністю (майже 40 років розробки), компактністю (менше 1000 базових системних викликів) і універсальністю (один системний виклик застосовується до різних ситуацій). Наприкінці 80-х рр. XX ст. інтерфейс системних викликів ОС UNIX став основою міжнародного стандарту мобільного відкритого програмного забезпечення POSIX. Таким чином, опанування системного програмування в ОС UNIX дає можливість надалі створювати мобільне програмне забезпечення під будь-яку сучасну операційну систему, що задовольняє вимоги стандарту POSIX.

Традиційно в якості мови системного програмування використовується мова С. Це, зокрема, обумовлено тим, що становлення мови С та ОС UNIX відбувалося в один і той самий час у одному й тому ж авторському колективі. Довгий час мова С вигідно відрізнялася від конкуруючих процедурних мов гнучкістю, простотою, мобільністю й компактністю опису програм. У той же час на ринку освітніх послуг частіше використовувалася мова Pascal, яка відрізнялася високим ступенем формалізації й структурування програм. Як у мові для початкового навчання програмування, в її синтаксисі чітко реалізувалися основні алгоритмічні конструкції, в якості ключових містилися англомовні слова, були наявні самоочевидні імена процедур і функцій.

У процесі розвитку мова С стала більш чітко структурованою, менш «поблажливою» до вільної роботи з вказівниками, що більш повно відповідає парадигмам структурного програмування. У свою чергу, до мови Pascal було додано обмежену адресну арифметику, засоби гнучкого управління циклами, модульну структуру та інші запозичені з мови С засоби. Таке взаємне збагачення мов-конкурентів привело до того, що поступово обидві вони стали застосовуватися і в системному, і в прикладному програмуванні.

Найвищий ступінь гнучкості мови Pascal, еквівалентний С, але зі збереженням традиційного Pascal-синтаксису, реалізований в мобільному компіляторі Free Pascal.

Наявність загальних програмних бібліотек для мов С та Pascal, реалізована в сімействі компіляторів GCC і Free Pascal, надає можливість вирішувати будь-які завдання однаковими або хоча б однотипними засобами, залишаючи розходження лише в синтаксисі.

Це надало можливість розробити новий варіант лабораторного практикуму з системного програмування, обравши в якості мов програмування С та Pascal. Для реалізації цієї мети було:

– побудовано інтерфейсну бібліотеку до ядра операційної системи UNIX мовою Pascal, що розширює функціональність стандартних модулів `linux` і `sysutils`;

– створено аналоги функцій бібліотеки `stdio` мови С для доступу до файлів і процесів;

– укладено систематичні настанови для програміста стосовно роботи з модулями `linux`, `sysutils` і `stdio`, проілюстровані прикладами системних утиліт.

У результаті було створено настанови системного програміста в UNIX у середовищі Free Pascal [249], що включає розділи, які повністю відповідають наведеній вище концепції курсу:

1. Основні поняття й термінологія.

2. Файл.

3. Робота з файлами.

4. Каталоги, файлові системи й спеціальні файли.

5. Процес.

6. Сигнали та їх опрацювання.

7. Обмін даними між процесами за допомогою програмних каналів.

8. Додаткові методи обміну даними між процесами.

9. Термінал.

10. Сокети.

11. Стандартна бібліотека введення/виведення.

12. Різні додаткові системні виклики й бібліотечні процедури.

Особливістю обраних засобів реалізації поставлених завдань є їх відкритість, низька вартість і мобільність, що дозволяє розроблене програмне забезпеченню використовувати під управлінням багатьох операційних систем [246].

Великий інтерес студентів викликають завдання на створення аналогів стандартних команд ОС. Такі завдання, як правило, вимагають застосування відомостей з різних розділів курсу, що сприяє формуванню уявлень про взаємозв’язок різних складових стандарту POSIX.

Наступний приклад показує реалізацію команди для перегляду вмісту каталогу, що вимагає спільного застосування засобів для обслуговування поняття користувача (імені користувача, числового ідентифікатора користувача, числового ідентифікатора групи користувача, імені групи), засобів для роботи з файлами (режиму доступу до файлу, числового ідентифікатора власника-користувача та групи, типу файлу), засобів опрацювання структурованих даних (відкриття, читання та закриття каталогу), засобів управління часом (отримання даних про час створення, останнього читання та останнього запису), засобів стандартної бібліотеки введення/виведення.

Вправа 13.28. Напишіть аналог команди `ls -l`.

```
uses linux,strings,sysutils,stdio; (*для системних викликів та роботи з
рядками PChar*)

function ctime(var time_t:longint):pchar;cdecl;external 'c';
```

```
function gettype(t:word):char;forward; (*тип об'єкту ф.с. в форматі
команди ls*)

(*тип об'єкту ф.с. в форматі команди ls*)
function gettype(t:word):char;
begin
  if S_ISDIR(t) then            (*перевірка на каталог*)
    gettype:='d'
  else
    if S_ISREG(t) then          (*перевірка на звичайний файл*)
      gettype:='-'
    else
      if S_ISBLK(t) then        (*перевірка на блочний пристрій*)
        gettype:='b'
      else
        if S_ISCHR(t) then      (*перевірка на символьний пристрій*)
          gettype:='c'
        else
          if S_ISFIFO(t) then (*перевірка на іменований програмний
канал*)
            gettype:='p'
          else
            if S_ISLNK(t) then  (*перевірка на символьне посилання*)
              gettype:='l'
            else
              gettype:='?';
end;


function getrights(r:word):string;
var
  u,           (*права для власника*)
  g,           (*права для групи*)
  o,           (*права для всіх інших*)
```

```
  s,           (*спеціальні права*)
  i:integer;
  res:string;  (*права в символьній формі*)
const
  o7777=(1 shl 12)-1; (*константа = всі 12 бітів прав задано *)
  o10  =8;            (*010  *)
  o100 =64;           (*0100 *)
  o1000=512;          (*01000*)
  symrights:array [0..7] of string=( (*базові комбінації прав у
символьній формі*)
       '---', (*0 = 000*)
       '--x', (*1 = 001*)
       '-w-', (*2 = 010*)
       '-wx', (*3 = 011*)
       'r--', (*4 = 100*)
       'r-x', (*5 = 101*)
       'rw-', (*6 = 110*)
       'rwx'  (*7 = 111*)
     );
  spec='tss';          (*масив спеціальних прав доступу*)
begin
  (*обрізаємо старші біти, що не відносяться до прав доступу (тип файлу
і т.п.)*)
  r:=r and o7777;(* константа 10000-1==1*8^4-1==1*(2^3)^4-1==2^12-1 *)
  (*виділяємо числові права для власника, групи, інших + спеціальні*)
  o:=r mod o10;
  s:=r div o1000;
  u:=(r div o100) mod o10;
  g:=(r mod o100) div o10;
  res:=symrights[u]+symrights[g]+symrights[o];(*формуємо символьні
права з базових трійок*)

  for i:=1 to 3 do   (*цикл перевірки наявності спеціальних прав*)
    if s and (1 shl (i-1)) <> 0 then (*якщо право встановлено*)
      if res[12-3*i]='x' then  (*якщо є звичайне право на виконання*)
        res[12-3*i]:=spec[i]   (*заносимо маленьку букву*)
```

```
      else
        res[12-3*i]:=upcase(spec[i]); (*інакше - велику*)

  getrights:=res; (*повертаємо результат - 9-тисимвольне подання 12-
тибітних прав*)
end;


var
  d:^TDir;              (*вказівник на запис для роботи з каталогом*)
  elem:^Dirent; (*вказівник на запис, де зберігається один элемент
каталогу*)
  tekkat,                          (*рядок для зберігання імені каталогу*)
  fullpath                        (*повний шлях до элемента каталогу*)
          :array [0..1000] of char;
  st:stat;          (*для зберігання даних про файл або каталог*)

begin
  if paramcount=0 then  (*якщо в командному рядку не вказано каталог,*)
    strcopy(tekkat,'.') (*то в якості каталогу використовуємо поточний*)
  else
    tekkat:=paramstr(1); (*інакше використовуємо каталог з командного
рядка*)

  if not access(pchar(tekkat),F_OK or R_OK) then  (*F_OK - перевірка
існування об’єкта ф.с.*)
  begin
    writeln('Каталог ', tekkat, ' не існує або недоступний для
читання');  (*діагностика*)
    halt(1);                    (*повернення у попередню програму*)
  end;

  if not fstat(pchar(tekkat),st) then    (*спроба отримати дані про
файл чи каталог*)
  begin
```

```
    writeln('Помилка отримання даних про каталог ', tekkat);
(*діагностика*)
    halt(1);                      (*повернення у попередню програму*)
  end;

  if not S_ISDIR(st.mode) then    (*перевірка на каталог*)
  begin
    writeln(tekkat, ' - не каталог');  (*діагностика*)
    halt(1);                      (*повернення у попередню програму*)
  end;

  d:=opendir(tekkat);       (*спроба відкрити каталог для читання*)

  if d=nil then                   (*якщо спроба невдала*)
  begin
    writeln('Помилка виклику opendir для каталогу ', tekkat);
(*діагностика*)
    halt(1);                      (*повернення у попередню програму*)
  end;

  elem:=readdir(d);               (*спроба читання элемента каталогу*)
  while elem<>nil do
  begin
    (*формування повного імені елемента каталогу*)
    strcopy(fullpath,tekkat);     (*копіюємо ім'я поточного каталога у
початок повного імені*)
    if strcomp(tekkat,'/')<>0 then(*якщо поточний каталог - не
кореневий*)
    begin
      if fullpath[strlen(fullpath)-1]='/' then (*якщо в кінці імені
каталогу слеш*)
        fullpath[strlen(fullpath)-1]:=#0;      (*замінюємо його ознакою
кінця рядка*)
      strcat(fullpath,'/');       (*додаємо після імені каталогу слеш-
роздільник*)
    end;
```

```
    strcat(fullpath,elem^.name);  (*та ім'я елемента каталогу*)

    if not fstat(pchar(fullpath),st) then    (*спроба отримання даних
про файл чи каталог*)
    begin
      writeln('Помилка отримання даних про ', fullpath);
(*діагностика*)
    halt(1);                      (*повернення у попередню програму*)
    end;
    {gmtime_r(st.mtime,mytm);}
    writeln(gettype(st.mode),getrights(st.mode),st.nlink:5,
     '  ',st.size:10,' ',ctime(st.mtime), elem^.name);  (*виведення
імені елемента каталогу*)
    elem:=readdir(d);               (*спроба читання елемента каталогу*)
  end;

  closedir(d);               (*закривання відкритого каталогу*)
end.
```

3.4.1.13. Приклад 2: реалізація лабораторного практикуму з системного програмування в POSIX-системах засобами Python. Одним з утруднень, що виникають у процесі вивчення системного програмування, є прив'язка цього курсу до використовуваних інструментальних засобів – операційної системи та мови програмування. Для подолання даного утруднення доцільним є стабілізація курсу системного програмування на основі інваріантності стосовно операційної системи і мови програмування. Це стає можливим при виконанні двох умов:

1) сумісності між операційними системами на рівні програмних інтерфейсів;

2) виконання програм під управлінням різних систем без перекомпіляції.

Перша умова сьогодні виконується на всіх POSIX-сумісних операційних системах. Виконання другої умови передбачає використання інтерпретованої мобільної мови програмування.

Одна з них – описана вище мова Python – це інтерпретована, об'єктно-орієнтована, високорівнева мова програмування, що має легкий для вивчення

синтаксис. Інтерпретатор мови Python і велика стандартна бібліотека модулів доступні безкоштовно і можуть вільно поширюватися. При цьому в складі бібліотеки є ряд модулів, в яких реалізується стандарт POSIX. Багатство засобів доступу до ядра ОС, надане в даних модулях, у поєднанні з простотою мови, робить мову Python привабливим для використання в процесі навчання системного програмування.

Для підтримки цього курсу було створено довідник системного програміста мовою Python, що містить у собі 10 розділів, які охоплюють усю множину системних викликів POSIX-сумісних операційних систем. Довідник поділяється на дві частини – управління файлами і управління процесами (структура довідника є еквівалентною [249]).

Приклад:

Вправа 4.11. Використовуючи POP3-з'єднання, отримайте листи з вашої поштової скриньки.

```
import poplib, email
# Облікові дані користувача:
SERVER = "pop.server.com"
USERNAME = "user"
USERPASSWORD = "secretword"

p = poplib.POP3(SERVER)
print p.getwelcome()
# етап ідентифікації
print p.user(USERNAME)
print p.pass_(USERPASSWORD)
# етап транзакцій
response, lst, octets = p.list()
print response
for msgnum, msgsize in [i.split() for i in lst]:
  print "Повідомлення %(msgnum) s має довжину %(msgsize)s" % vars()
  print "UIDL =", p.uidl(int(msgnum)).split()[2]
  if int(msgsize) > 32000:
    (resp, lines, octets) = p.top(msgnum, 0)
```

```
  else:
    (resp, lines, octets) = p.retr(msgnum)
  msgtxt = "\n".join(lines)+"\n\n"
  msg = email.message_from_string(msgtxt)
  print "* Від: %(from)s\n* Кому: %(to)s\n* Тема: %(subject)s\n" % msg
  # msg містить заголовки повідомлення або все повідомлення (якщо воно
невелике)

# етап оновлення
print p.quit()
```

3.4.2. Системне програмне забезпечення. Вивчення будь-якої мови програмування тісно пов'язане з інструментальним програмним забезпеченням для розробки програм – оболонкою, текстовим редактором та компілятором, що звичайно відносять до системного програмного забезпечення. Інтеграція системного програмного забезпечення у середовища для розробки програм та сучасні операційні системи поступово викликають розмивання поняття системного програмного забезпечення. Так, проведене у 2003 р. опитування студентів I–II курсів спеціальності «Інформатика» Криворізького державного педагогічного університету показало, що 62% студентів недостатньо чітко розрізняють основні компоненти системного програмного забезпечення (з них 40% їх плутають), 24% утруднюються з визначенням функцій компілятора, 13% не дали жодного означення.

У процесі навчання мов програмування поняття компілятора поступово збагачується, наповнюючись досвідом практичної роботи, тому виникає необхідність його формалізації.

На жаль, у багатьох студентів поняття компілятора залишається досить розмитим, і цьому є об'єктивні причини. По-перше, компілятори розроблені для великої кількості мов програмування і мають різні форми та методи застосування. По-друге, в багатьох системах компіляція є процесом, інтегрованим у середовище розробки. По-третє, продовжується інтенсивний розвиток усіх напрямів досліджень, що відносяться до системного програмного забезпечення: в

будові компіляторів з’явилися нові важливі компоненти, технології та розробки [419].

Незважаючи на таку різноманітність систем та постійні зміни, фундаментальні положення процесу компіляції залишаються незмінними, і саме на них варто зосередити увагу студентів. Для цього пропонуємо вивчення теми «Основи компіляції» за таким планом:

1. Основні поняття процесу компіляції.
2. Типова структура компілятора.
3. Інтегровані середовища розробки.
4. Вимоги до розробки компіляторів.

1. Основні поняття процесу компіляції

Програмне забезпечення можна створювати за допомогою багатьох мов програмування з різними парадигмами (процедурною, об’єктно-орієнтованою, функціональною, візуальною тощо). Призначення компілятора полягає в перетворенні описів програм, орієнтованих на користувача, в машинно-орієнтовані, що використовуються безпосередньо при виконанні програми за допомогою комп’ютера. Компілятори – це спеціалізовані системи опрацювання тексту, що мають багато спільного з іншими інструментальними засобами опрацювання текстів, написаними мовою програмування або природною мовою.

Робота компілятора звичайно розглядається на двох етапах.

1. *Етап аналізу*, на якому аналізується вхідний текст.
2. *Етап синтезу*, на якому генерується машинно-орієнтоване подання.

Вхід етапу аналізу називається *вхідним текстом* чи *вхідним кодом*, а вихід етапу синтезу – *цільовим текстом* чи *цільовим кодом*. Перетворення вхідного коду в цільовий звичайно називається *процесом компіляції*. Процес компіляції здійснюється за допомогою компілятора. Компілятор мови можна також назвати *реалізацією* мови. Породжений компілятором цільовий код може мати вид машинного коду для деякої машини (комп’ютера) чи деякого проміжного коду, що надалі буде перетворений (вже за допомогою інших інструментальних

засобів) у машинний код. Можливий також варіант, коли проміжний код безпосередньо використовується за допомогою *інтерпретатора*.

Процес компіляції являє собою перетворення тексту однієї мови на іншу, перехід від *вхідного коду* до *цільового коду*.

Процес компіляції також включає третю мову – мову *реалізації*, під якою розуміють мову написання компілятора (зазвичай, це мова С). Нею може бути та ж мова, що і вихідний чи цільовий код, але це необов'язково.

2. Типова структура компілятора

Логічно процес компіляції поділяється на *етапи*, що, у свою чергу, діляться на *фази*. Фізично процес компіляції поділяється на *проходи*.

Основними етапами компіляції є *аналіз* (визначення структури і значення вхідного коду) і *синтез* (побудова цільового коду). Крім того, може бути етап попереднього опрацювання (*препроцесінгу*). Цей етап в основному пов'язаний з мовами С та С++.

Етап аналізу ділять на три окремі фази:

1. Лексичний аналіз.

2. Синтаксичний аналіз.

3. Семантичний аналіз.

Лексичний аналіз – це відносно проста фаза, у якій формуються *символи* (чи лексеми) мови. Слова мови, наприклад, if, for, do чи ідентифікатори, наприклад, count, name, чи послідовності знаків, наприклад, ++, ==, зручно сприймати як один символ, як це робиться на етапі аналізу. Призначенням фази лексичного аналізу чи *лексичного аналізатора* є перехід від набору знаків до символів мови, які надалі будуть опрацьовуватися на синтаксичній і семантичній фазах. Такий підхід копіює поведінку людини при читанні програми: адже ми сприймаємо текст програми не як простий набір знаків, а як набір символів (слів), що складаються з цих знаків.

Тут важливо відзначити, що за допомогою лексичного аналізатора усього лише формуються символи – їх набір і порядок не має для нього ніякого зна-

чення, тобто, лексичний аналізатор звичайно не використовується для роботи з контекстом.

У процесі *синтаксичного аналізу* визначається загальна структура програми, що включає розуміння порядку проходження символів у програмі. Це означає, що *синтаксичний аналізатор* повинен мати дані про контекст, у якому він працює. Після застосування синтаксичного аналізатора отримується подання програми в деревоподібній формі, що називається *синтаксичним деревом*. Наприклад, вираз (*a* + *b*) * (*c* – *d*) може бути поданий у вигляді:

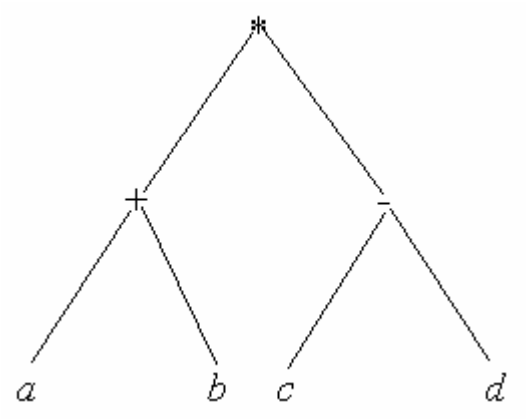


Рис. 3.33. Абстрактне синтаксичне дерево

Це подання називається *абстрактним синтаксичним деревом*. У такий спосіб уся програма може бути подана за допомогою абстрактних синтаксичних дерев.

Фаза синтаксичного аналізу є ключовою на етапі аналізу. Вона безпосередньо взаємопов’язана з лексичною фазою, а отримані після проходження цієї фази результати надалі будуть використовуватися при реалізації семантичної фази. Синтаксичним аналізатором зчитуються символи в програмі зліва направо. У процесі зчитування визначається, чи може бути послідовність вже прочитаних символів початком програми. Наприклад, перших п’ять вхідних символів можуть бути початком деякої програми, а перші шість – ні. У цьому випадку подальші дії будуть залежати від прийнятого способу *відновлення після помилок*.

Деякі властивості мов програмування не можуть бути перевірені простим скануванням зліва направо без створення таблиць довільного розміру. Перевір-

ка таких властивостей мов програмування (що називають *статичною семантикою*) виконується в *семантичній фазі аналізу*.

Етап синтезу процесу компіляції складається з наступних основних фаз.

1. Генерація машинно-незалежного коду.
2. Оптимізація машинно-незалежного коду.
3. Розподіл пам’яті.
4. Генерація машинного коду.
5. Оптимізація машинного коду.

В окремих випадках деякі з цих фаз можуть бути відсутніми. Наприклад, якщо за допомогою компілятора безпосередньо компілюється машинний код, перші дві фази можуть опускатися. Оптимізація коду може відбуватися на рівні машинно-незалежного коду, на рівні машинного коду, на обох рівнях чи не відбуватися на жодному рівні.

Існують причини, чому спочатку необхідно створювати *машинно-незалежний код*: це сприяє мобільності компілятора й служить для відокремлення залежності від мови та залежності від машини. Багато компіляторів також генерують деякі проміжні коди, що можуть бути незалежні від вхідної мови, машинної мови чи від обох. Прикладами таких проміжних мов є наприклад, P-код для Pascal, Diana для Ada, байт-код для Java.

Оптимізація програмного коду – це перетворення програми засобами компілятора з метою поліпшення їх характеристик, таких як продуктивності або компактності, – без зміни функціональності. Оптимізація – не обов’язковий, але важливий етап компіляції. Вона може відбуватися неявно під час трансляції програми, але, як правило, оптимізацію програми виділяють як окрему фазу функціонування компілятора. Компонувальник також може виконувати частину оптимізацій, таких як видалення зайвих підпрограм або їх перевпорядкування підпрограм. Розрізняють низько- та високорівневу оптимізацію. Низькорівнева оптимізація перетворює програму на рівні елементарних команд, наприклад, інструкцій процесора. Високорівнева оптимізація здійснюється на рівні структурних елементів програми, таких як розгалуження та цикли.

Низькорівнева оптимізація включає в себе такі техніки, як оптимальний вибір (заміна, об'єднання, поділ) інструкцій; перевпорядкування інструкцій; розподіл регістрів; видалення ланцюжків переходів; векторизація; пониження сили операцій.

Високорівнева оптимізація включає в себе такі техніки, як видалення зайвого («мертвого») коду і зайвих присвоєнь; підгонка, звернення циклів; оптимізація розгалуження; розгортання, згортання, об'єднання і поділ циклів; обчислення інваріантів циклів, винесення загальних підвиразів і коду в розгалуження, винесення розгалуження з циклів; перемикання, об'єднання і поділ розгалужень; попередній відбір даних; перевпорядкування функцій; вбудовування і вилучення функцій.

Якщо оптимізується лише окрема підпрограма, то таку оптимізацію називають *локальною*, інакше – повною або *глобальною*.

Потреба в оптимізації генерованого коду може бути різною. Якщо потрібен ефективний код, необхідно забезпечити значну оптимізацію. У той же час в багатьох середовищах швидкість виконання програми не є критичним параметром, отже, необхідна лише незначна оптимізація. Деякі види оптимізації реалізувати просто, і тому їх часто реалізують у компіляторах, тоді як інші форми оптимізації, особливо глобальні (на відміну від локальних), трудомісткі і вимагають значних витрат часу при компіляції, а тому застосовуються рідко [118]. Багато компіляторів надає користувачу можливість самому визначити, що саме потрібно оптимізувати.

У фазі *розподілу пам'яті* кожній сталій та змінній в програмі ставиться у відповідність зарезервоване місце в пам'яті для зберігання їх значень. Дана область пам'яті може бути одного з трьох типів:

1) *статична пам'ять*, якщо час зберігання значення змінної дорівнює часу виконання програми (зарезервоване місце в пам'яті для зберігання значення змінної не може бути звільнене до завершення виконання програми);
2) *динамічна пам'ять*, якщо час зберігання значення змінної дорівнює часу виконання визначеного блоку, функції чи процедури (зарезервоване місце в

пам’яті для зберігання значення змінної може бути звільнене після виконання даного фрагмента програми);

3) *глобальна пам’ять*, якщо на момент компіляції час зберігання значення змінної невідомий, а пам’ять повинна виділятися і звільнятися в процесі виконання.

Результатом роботи фази розподілу пам’яті є створення *таблиці адрес*.

Якщо логічно процес компіляції складається з *етапів* і *фаз*, фізично він складається із *проходів*. Компілятор здійснює прохід щоразу при зчитуванні вхідного коду чи його подання. Ранні компілятори були *багатопрохідними* через недостатній обсяг пам’яті комп’ютерів того часу. Сучасні компілятори є переважно *однопрохідними*, тобто повний процес компіляції цілком виконується при однократному зчитуванні коду. У цьому випадку різні описані фази будуть виконуватися паралельно (це, як правило, є найбільш зручним), що усуває необхідність складного зв’язку між різними проходами.

3. Інтегровані середовища розробки

Сучасні компілятори часто є не окремими, автономними інструментальними засобами, а являють собою частину *інтегрованих середовищ розробки* (IDE), які іноді називають середовищами програмування. Крім засобів компіляції, сучасні IDE пропонують засоби мовно-орієнтованого редагування, налагодження, визначення робочих профілів програми, управління конфігурацією і т.д. Прикладом такого середовища є IDE Eclipse, в якому передбачено такі основні групи операцій:

– *редагування* із засобами *вирізання*, *вставляння*, *скасування операції* тощо;

– *пошук* із засобами заміни тексту та локалізації функцій в процесі налагодження;

– *перегляд* різних вікон, що містять засоби діагностики та інші дані, пов’язані з поточним проектом (точки переривання програми, зміст регістрів, розташування змінних, використання класів і т.ін.);

– *управління проектом*, включаючи запуск нових проектів, компіляцію і зв'язування різних компонентів проекту;

– *налагодження* з можливістю запуску програми в режимі покрокового виконання, задання точок переривання, відстеження значень виразів, перегляду таблиць символів і т.д.

– *засоби виконання*, пов'язані з IDE.

4. Вимоги до розробки компіляторів

Загальна структура компілятора багато в чому залежить від його фазової структури і структури синтаксичного аналізатора, а структура синтаксичного аналізатора відображає властивості вихідної мови. Як правило, при проектуванні компілятора керуються такими вимогами: 1) ефективна компіляція; 2) мінімальний розмір компілятора; 3) мінімальна довжина цільового коду; 4) створення ефективного цільового коду; 5) мобільність; 6) простота використання; 7) практичність.

Одночасно задовольнити всі ці вимоги неможливо, тому доводиться віддавати перевагу деяким з них. У навчальних середовищах, наприклад, ефективність компіляції й зручні засоби діагностики можуть бути більш важливими, ніж створення ефективного цільового коду, тоді як для вбудованих систем першочергове значення має розмір і ефективність цільового коду. Багато компіляторів розроблені з розрахунком на те, що користувач сам повинен визначати режим роботи компілятора – ступінь оптимізації, виконання перевірок часу виконання і т.д.

Наведена вище структура вивчення теми надає можливість наповнити поняття компілятора змістом на основі вивчення загальних властивостей процесу компіляції, що найбільш повно відповідає вимогам до фундаментальної теоретичної підготовки із системного програмного забезпечення [304].

Виходячи з того, що студенти постійно користуються компіляторами для створення власних програм, доцільно на практичному занятті із вказаної теми перейти від розгляду інтерфейсу конкретного середовища програмування до програмної реалізації наступної задачі:

Створити програму для обчислення арифметичних виразів, що містять цілі числа, з'єднані операціями додавання та множення; вирази можуть бути згруповані за допомогою дужок.

В цій задачі пропонується створити найпростіший інтерпретатор арифметичних виразів. Вибір цілих чисел та лише двох операцій зумовлений їх мінімальністю. Подальші модифікації програми (зміна типу даних, реалізація операцій віднімання, ділення тощо) є суто косметичними і можуть бути запропоновані як домашнє завдання.

Для визначення того, з яких компонентів буде складатися програма, спочатку пропонуємо розглянути кілька арифметичних виразів: 1+2+3*4, 76*(4+8)*2, -123 тощо. В процесі виконання цієї роботи звертаємо увагу на те, що вказані вирази містять зображення цілих чисел, знаків «+», «*», «(», «)», після чого робимо висновок про те, що саме ці елементи будуть головними лексичними одиницями програми. При цьому операції та дужки задаються лише одним знаком, тоді як запис числа може містити кілька знаків.

Дамо таке визначення: ціле число – це «0» або скінченний набір цифр, що починається з будь-якого елемента множини [«1»..«9»], за яким слідує нуль або більше будь-яких елементів множини [«0»..«9»]. Цілому числу може передувати необов'язковий знак «–».

Використовуючи регулярні вирази, ціле число можна було б означити так:

```
[-]?([1-9][0-9]*|0)
```

Тут «?» означає необов'язковий елемент, круглі дужки використовуються для групування, «*» (зірочка Кліні) означає повторення попереднього елемента нуль та більше разів, «|» – операцію «або».

У якості ознаки закінчення введення виразу та необхідності початку його обчислення оберемо клавішу «Enter», а для завершення програми використаємо будь-яке доцільне слово (наприклад, «quit» – «вихід»). Нарешті, звертаємо увагу на те, що користувачем програми можуть бути введені й будь-які інші символи, що не повинні входити до запису арифметичного виразу. Ці три випадки також вважатимемо окремими лексемами.

Таким чином, перша частина програми – лексичний аналізатор – повинна розпізнавати 8 типів лексем.

Написання лексичного аналізатора є хоч і простою за структурою, проте трудомісткою роботою [244], яка в студентів-першокурсників відніме значний час. З метою його економії (адже написання програми необхідно вкластися у 2 академічні години) застосуємо програмний засіб для генерації лексичних аналізаторів – flex (lex), на вхід якого подамо файл наступної структури:

```
%{
  enum TOKENS {INTEGER=1, PLUS, UMNOJ, OTKSK, ZAKSK, ENDSTR, QUIT,
NERASP};
%}

int       [-]?([1-9][0-9]*|0)
%%
{int}     return INTEGER;
"+"       return PLUS;
"*"       return UMNOJ;
"("       return OTKSK;
")"       return ZAKSK;
"quit"    return QUIT;
\n        return ENDSTR;
[\t ]     ;
.         return NERASP;
%%

int yywrap()
{
  return 1;
}
```

TOKENS є переліком 8 визначених лексем: ціле число, додавання, множення, відкриваюча та закриваюча круглі дужки, ознаки кінця рядка і програми та нерозпізнаного символу. Ціле число означуємо правилом розпізнавання {int}, що містить відповідний регулярний вираз, усі інші лексеми – відповідними на-

борами символів: «+», «*», «(», «)», «quit», «\n» (клавіша Enter), «.» (позначення будь-якого іншого символу, відмінного від описаних вище). Символи пропуску та табуляції ігноруватимемо як незначущі.

Після опрацювання наведеної лексичної структури за допомогою flex студенти отримують файл з текстом функції лексичного аналізу yylex(), після кожного звернення до якої повертається одна лексема, для зберігання якої в програмі передбачимо відповідну змінну:

```
int lexema;
```

Наступній частині роботи – побудові синтаксичного аналізатора – передує повернення до розгляду записаних арифметичних виразів. 1+2+3*4, 76*(4+8)*2 тощо з метою визначення правил їх обчислення. Для цього спочатку визначаємо пріоритет виконання кожної з операцій (дужки – вищий, множення – середній, додавання – нижчий), а потім зображаємо процес обчислення вказаних виразів у вигляді синтаксичних дерев (рис. 3.34).

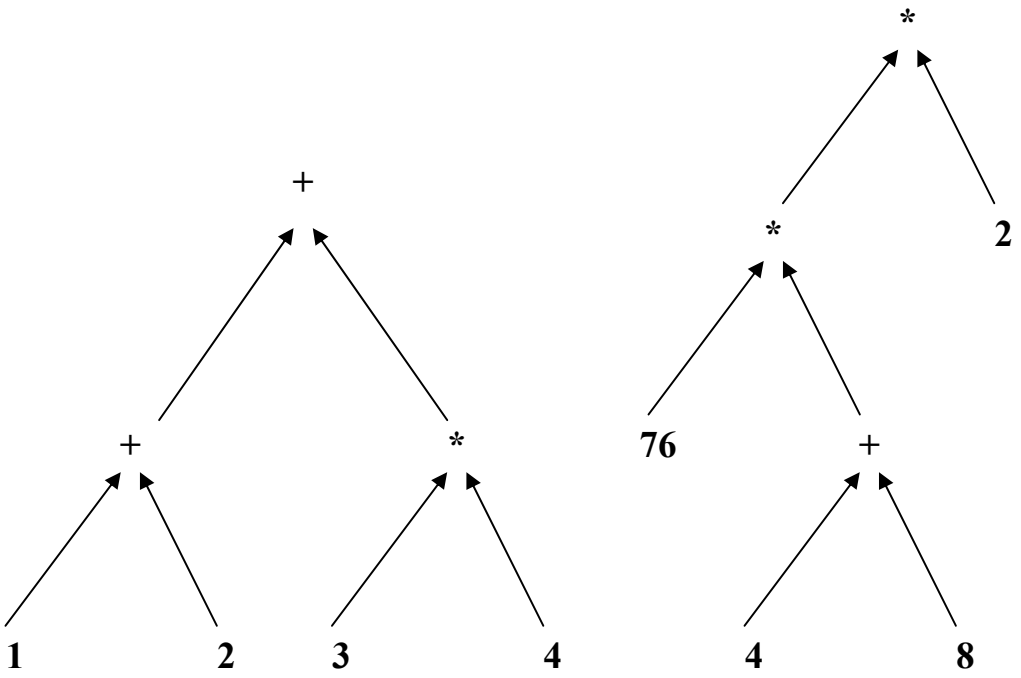


Рис. 3.34. Синтаксичні дерева для виразів
1+2+3*4 (ліворуч) та 76*(4+8)*2 (праворуч)

З рисунків видно, що обчислення починається з найнижчих гілок дерева і виконується у порядку, в якому читається арифметичний вираз – зліва направо.

Дужки на синтаксичному дереві не показуються, проте в процесі його побудови взятий у дужки вираз розташовується на рівень нижче.

Будь-який арифметичний вираз E можна подати у вигляді суми:

E=T+T+T+..., де T – інший вираз, що складається з множників:

T=F*F*F*..., де F – або число, або будь-який арифметичний вираз у дужках:

F=(E),

F=число.

Доданки та множники у виразах E і T повторюються щонайменше один раз, тому в термінах регулярних виразів правила обчислення (граматика) будь-якого арифметичного виразу можуть бути записані так:

1) E → T(«+» T)*

2) T → F(«*» F)*

3) F → «(» E «)»

4) F → «число»

Тут для уникнення плутанини зірочки Кліні та зірочки-множення, дужок групування та дужок в запису арифметичного виразу усі лексеми взяті у лапки.

Визначивши правила обчислення, наводимо приклади їх застосування для синтаксичного аналізу наведених вище прикладів, на кожному кроці застосовуючи лише одне правило, та порівнюємо цей процес із побудовою синтаксичного дерева:

$E \Rightarrow T + T + T \Rightarrow T + T + F * F \Rightarrow F + T + F * F \Rightarrow F + F + F * F \Rightarrow$ 1+2+3*4

$E \Rightarrow T \Rightarrow F * F * F \Rightarrow F * (E) * F \Rightarrow F * (T + T) * F \Rightarrow F * (F + T) * F \Rightarrow$

$\Rightarrow F * (F + F) * F \Rightarrow$ 76*(4+8)*2

Формалізація запису процесу обчислення за допомогою правил 1–4 дає підстави стверджувати, що його програмна реалізація – синтаксичний аналізатор являтиме собою впорядкований набір звернень до функцій E, T та F. Процес обчислення починається з виклику функції E, тому організуємо в програмі наступний цикл:

```
виконувати дії
|  отримати наступну лексему
|  викликати функцію E та отримати результат обчислення виразу
|  роздрукувати результат обчислення виразу
до тих пір, поки не буде введена ознака завершення програми
```

Програмна реалізація цього циклу матиме такий вигляд:

```
main()// головна функція програми
{
  int result; // змінна для зберігання результату

  printf("Калькулятор: +,*,(,)\n\n"); // заставка програми
  do { // виконувати дії
    lexema=yylex(); // отримати наступну лексему
    result=E(); // викликати функцію E та отримати результат обчислення
    printf("%d\n",result); // роздрукувати результат обчислення виразу
  }while(lexema!=QUIT) // поки не буде введена ознака завершення
програми
}
```

У відповідності до правила 1 функція E призначена для виконання наступних дій:

Алгоритм	Програмна реалізація
	`int E()`
	`{`
`викликати функцію T та зберегти результат`	`  int result=T();`
`поки зчитана лексема – знак "+"`	`  while(lexema==PLUS)`
`|  отримати наступну лексему`	`  {`
`|  викликати функцію T, додавши результат`	`    lexema=yylex();`
`|    до попереднього`	`    result=result+T();`
`-`	`  }`
`повернути результат звернення до функції`	`  return result;`
	`}`

У відповідності до правила 2 функція T призначена для виконання наступних дій:

Алгоритм	Програмна реалізація
	int T() {
викликати функцію F та зберегти результат	int result=F();
поки зчитана лексема – знак "*"	while(lexema==UMNOJ)
| отримати наступну лексему	{
| викликати функцію T, помноживши	lexema=yylex();
| результат на попередній	result=result*F();
-	}
повернути результат звернення до функції	return result;
	}

За допомогою циклу while у функціях E і T реалізуються операції додавання та множення цілих чисел. Якщо ці операції відсутні, то маємо справу або з виразом у дужках, або просто з числом, що й реалізується у функції F.

За допомогою функції F реалізуються одразу 2 правила – 3 та 4. У правилі 3 стверджується, що, якщо вираз починається з відкриваючої дужки, слід звернутися до функції E, після чого перевірити, чи є наступна лексема закриваючою дужкою. Порушення парності дужок вважатимемо помилкою. До правила 4 переходимо у випадку, коли правило 3 не виконується. Єдиний допустимий тип лексеми в цьому правилі – ціле число, усі інші вважатимемо помилковими. За будь-яких обставин помилку необхідно діагностувати та, за можливістю, відновити аналіз виразу (наприклад, як це пропонується у [430]).

Алгоритм	Програмна реалізація
	int F() {
	int result;
якщо зчитана лексема знак "(", то перейти до правила 3:	if(lexema==OTKSK)
	{
| отримати наступну лексему	lexema=yylex();
| викликати функцію E	result=E();
| якщо зчитана лексема – знак ")"	if(lexema==ZAKSK)
| | отримати наступну лексему	lexema=yylex();

Алгоритм	Програмна реалізація

<pre>| інакше
| | помилка парності дужок
| -
інакше - перейти до правила 4
| якщо зчитана лексема – ціле число
| | отримати результат зі змінної yytext,
| | що визначена функцією yylex(),
| | перетворивши yytext на ціле число
| | отримати наступну лексему
| |
| інакше
| |
| | помилка введення нечислової лексеми
| | там, де вона очікувалася
| -
 -
повернути результат звернення до функції</pre>
	<pre> else
	    puts("Немає)");
	}
	else
	{
	 if(lexema==INTEGER)
	 {
	   result=atoi(yytext);
	   lexema=yylex();
	 }
	 else
	 {
	   puts("Не число");
	   result=0;
	 }
	}
	return result;
	}</pre>

Для унаочнення процесу обчислення виразу можна запропонувати доповнити кожну з наведених функцій налагоджувальною діагностикою. Для цього доцільно визначити макропідстановку DEBUG, в залежності від якої виконуватимуться директиви умовної компіляції #ifdef – #endif. Тоді, наприклад, функція F може мати такий вигляд:

```
int F()
{
  int result; // змінна для зберігання результату

#ifdef DEBUG
  puts("F почалась");
#endif
  if(lexema==OTKSK) // якщо зчитана лексема знак "(",
                    // то перейти до правила 3:
  {
```

```
    lexema=yylex(); // отримати наступну лексему
#ifdef DEBUG
    puts("Виклик E з F");
#endif
    result=E(); // викликати функцію E
    if(lexema==ZAKSK) // якщо зчитана лексема – знак ")"
      lexema=yylex(); // отримати наступну лексему
    else
      puts("Помилка: немає )"); // помилка парності дужок
  }
  else // інакше - перейти до правила 4
  {
    if(lexema==INTEGER) // якщо зчитана лексема – ціле число
    {
#ifdef DEBUG
      printf("ЦІЛЕ ЧИСЛО: %s\n",yytext);
#endif
      result=atoi(yytext); /* отримати результат зі змінної yytext, що
        визначена функцією yylex(),перетворивши yytext на ціле число */
      lexema=yylex(); // отримати наступну лексему
    }
    else
    { // помилка введення не числової лексеми
      puts("Помилка: не число");
      result=0;
    }
  }
#ifdef DEBUG
  printf("F закінчилась, результат=%d\n",result);
#endif
  return result; // повернути результат звернення до функції
}
```

Приклад роботи з побудованою програми у налагоджувальному режимі:

```
76*(4+8)*2
E почалась
    Виклик T з E
        T почалась
            Виклик F з T
                F почалась
                    ЦІЛЕ ЧИСЛО: 76
                F закінчилась, результат=76
            Виклик F з T
                F почалась
                    Виклик E з F
                        E почалась
                            Виклик T з E
                                T почалась
                                    Виклик F з T
                                        F почалась
                                            ЦІЛЕ ЧИСЛО: 4
                                        F закінчилась, результат=4
                                T закінчилась, результат=4
                            Виклик T з E
                                T почалась
                                    Виклик F з T
                                        F почалась
                                            ЦІЛЕ ЧИСЛО: 8
                                        F закінчилась, результат=8
                                T закінчилась, результат=8
                        E закінчилась, результат=12
                F закінчилась, результат=12
            Виклик F з T
                F почалась
                    ЦІЛЕ ЧИСЛО: 2
                F закінчилась, результат=2
        T закінчилась, результат=1824
E закінчилась, результат=1824
1824
quit
```

Відповідна схема звернень до функцій в процесі синтаксичного аналізу показана на рис. 3.35.

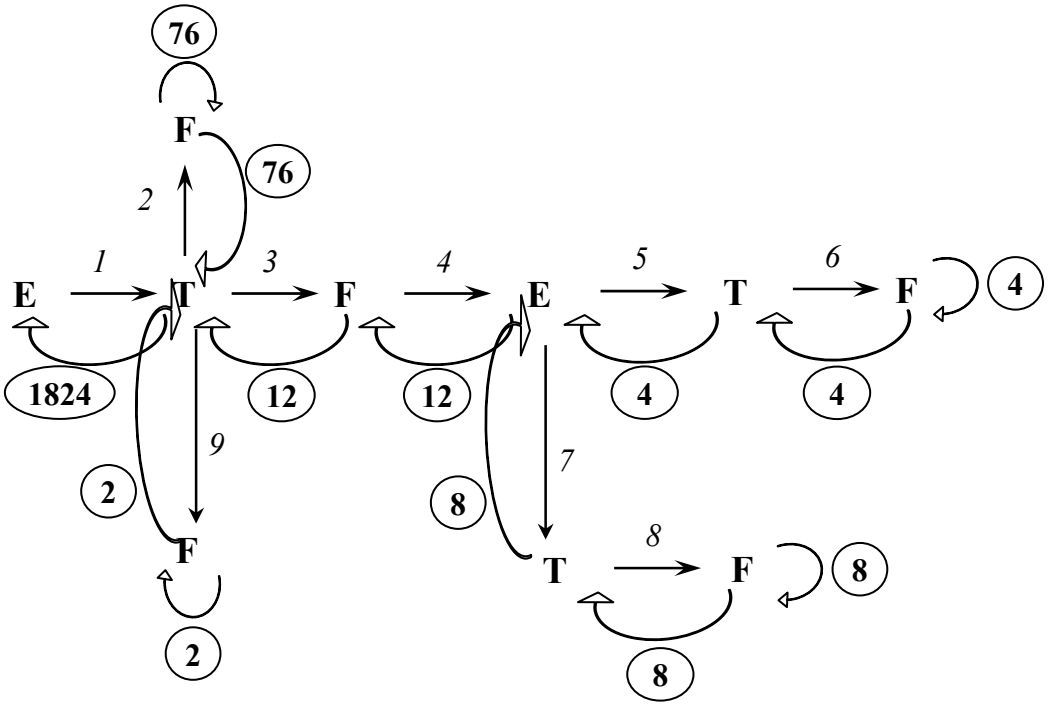


Рис. 3.35. Схема звернень до функцій в процесі обчислення виразу 76*(4+8)*2

Умовні позначення:

E – ім’я функції, що викликається;

1 – номер виклику наступної функції;

8 – значення, що повертається у попередню функцію

У подальшому створена програма може бути використана як основа для створення компілятора з мови програмування, системи символьної математики та як елемент інтегрованої системи для обчислення значень виразів, що задаються користувачем. Зауважимо, що для мови Pascal існує аналог flex – plex, а відповідна програма матиме такий вигляд:

```
%{
uses lexlib;
const
  _INTEGER=1;
```

```
  PLUS=2;
  UMNOJ=3;
  OTKSK=4;
  ZAKSK=5;
  QUIT=6;
  ENDSTR=7;
  NERASP=8;
%}
int   [-]?([1-9][0-9]*|0)
%%
{int}begin yydone:=true; yyretval:=_INTEGER; end;
"+"  begin yydone:=true; yyretval:=PLUS; end;
"*"  begin yydone:=true; yyretval:=UMNOJ; end;
"("  begin yydone:=true; yyretval:=OTKSK; end;
")"  begin yydone:=true; yyretval:=ZAKSK; end;
"quit"    begin yydone:=true; yyretval:=QUIT; end;
\n   begin yydone:=true; yyretval:=ENDSTR; end;
[\t ] ;
.    begin yydone:=true; yyretval:=NERASP; end;
%%

function yywrap:integer;
begin
   yywrap:=1;
end;

var
  lexema:integer;


function T:integer;forward;
function F:integer;forward;


function E:integer;
var
```

```
  result:integer;
begin
  result:=T;
  while lexema=PLUS do
  begin
    lexema:=yylex;
    result:=result+T;
  end;
  E:=result;
end;


function T:integer;
var
  result:integer;
begin
  result:=F;
  while lexema=UMNOJ do
  begin
    lexema:=yylex;
    result:=result*F;
  end;
  T:=result;
end;


function F:integer;
var
  result,code:integer;
begin
  if lexema=OTKSK then
  begin
    lexema:=yylex;
    result:=E;
    if lexema=ZAKSK then
      lexema:=yylex
```

```
    else
      writeln('Помилка: немає )');
  end
  else
  begin
    if lexema=_INTEGER then
    begin
      val(yytext,result,code);
      lexema:=yylex;
    end
    else
    begin
      writeln('Помилка: не число');
      result:=0;
    end
  end;
  F:=result;
end;


var
  result:integer; { змінна для зберігання результату }
begin
  writeln('Калькулятор: +,*,(,)');writeln; { заставка програми }
  while true do { виконувати дії... }
  begin
    lexema:=yylex; { отримати наступну лексему }
    if lexema=QUIT then { ... поки не буде введена ознака завершення
програми }
      break;
    result:=E; { викликати функцію E та отримати результат обчислення }
    writeln(result); { роздрукувати результат обчислення виразу }
  end;
end.
```

Наведений програмний код показує універсальність обраного підходу до реалізації синтаксичних аналізаторів: відмінності між програмами зумовлені лише синтаксисом використовуваної мови програмування, тому аналогічна структура програми збережеться при застосуванні Python та інших мобільних мов програмування.

Запропонований підхід до побудови практичного заняття надає можливість на початку вивчення курсу інформатики наповнити практичним змістом поняття інтерпретації, лексичного та синтаксичного аналізу. Застосування генератора лексичних аналізаторів при цьому суттєво полегшує процес написання програми, дозволяючи максимально наблизити текст програмної реалізації до запису алгоритму [312].

3.4.3. Подіє-орієнтоване програмування. Будь-яка комп'ютерна програма виконується у такий спосіб: після розміщення в пам'яті глобальних змінних і конструювання глобальних об'єктів класів починається виконання команд, що містяться в головній підпрограмі. Як правило, спочатку виконується ряд ініціалізаційних дій – аналіз параметрів командного рядка, формування інтерфейсу користувача тощо. Після виконання таких робіт можливі два принципово різних методи виконання програми – або вона виконується далі послідовно оператор за оператором (команда за командою), із зверненням в заданій послідовності до необхідних підпрограм, аж до завершення, або після виконання ініціалізаційних дій відбувається перехід у режим очікування подій, після відбування яких потрібно виконати відповідні дії, і після того, як подія відбудеться, викликається на виконання підпрограма – обробник цієї події, продовжуючи одночасно аналіз надходження інших подій і т.д. у циклі, поки не відбудеться подія, за якою генерується сигнал про необхідність залишити цикл і завершити програму. При цьому момент настання тієї чи іншої події і їх послідовність не регламентовані, але події можуть ранжуватися за пріоритетами їх обслуговування і обслуговування більш пріоритетних подій може переривати обслуговування менш пріоритетних на час обслуговування більш пріоритетних. Перервана менш пріоритетна подія (стосовно, тієї, що обслуговується в поточний мо-

мент) ставиться в чергу на обслуговування і її опрацювання буде здійснене, коли надійде його черга. Методи складання такого типу програм і назвемо *подіє-орієнтованим програмуванням* [250].

В кожній прикладній програмі можуть бути передбачені різні реагування на одну й ту саму подію (наприклад, натискання комбінації клавіш чи кнопок миші); крім того, необхідно мати можливість при виконанні програми імітувати певні події для виконання відповідних дій (для виклику потрібного обробника події), тому одна з основних задач операційної системи полягає в перетворенні сигналу, що надійшов, в інформаційне повідомлення (це, як правило, структура з необхідним набором полів), визначення адресата – обробника події і виклик цього обробника з передаванням йому в якості одного з аргументів повідомлення, що відповідає події. Щоб з операційної системи міг бути здійснений виклик необхідного обробника, структура прикладної програми повинна бути узгоджена з характеристиками і особливостями тієї операційної системи, під управлінням якої вона буде виконуватися.

Одним з найбільш яскравих прикладів реалізації подіє-орієнтованого програмування є задача побудови інтерфейсу користувача, елементи якого обмінюються один з одним повідомленнями. Набір функцій (класів) для створення інтерфейсу утворює бібліотеку. В операційній системі MS-DOS прикладами таких бібліотек є Turbo Vision та Graphics Vision (за допомогою першої створюється текстовий інтерфейс, за допомогою другої – графічний). В операційних системах сімейства Windows є власні механізми опрацювання подій, об’єднані в програмний інтерфейс Win32 (Win64), над яким надбудовуються об’єктно-орієнтовані бібліотеки MFC, VCL та ін. На жаль, застосування цих бібліотек в процесі навчання подіє-орієнтованого програмування ускладнене, адже всі вони (так само, як і згадані операційні системи) вимагають значних ліцензійних витрат, що для більшості навчальних закладів України досить проблематично.

В зв’язку з наведеним виникає актуальна проблема: *створення курсу подіє-орієнтованого програмування графічного інтерфейсу користувача на осно-*

ві POSIX-сумісних операційних систем та бібліотек, що не вимагають ліцензійних відшкодувань.

Графічні інтерфейси POSIX-систем мають давню історію. Стандартом стала розподілена система X Window (рис. 3.36), за допомогою якої можна виводити на екран дисплея графічні зображення, в ній підтримується концепція вікон і уніфікується робота з різними пристроями введення-виведення на основі бібліотеки Xlib [524]. Щоб полегшити програмування із застосуванням Xlib і спростити створення інтерфейсів користувача, створено кілька пакетів, серед яких найбільш поширені X Toolkit Intrinsics, Athena і Motif [492]. В останні роки з'явилися два нових пакети: GTK+ і Qt, які покладені в основу популярних графічних інтерфейсів GNOME (рис. 3.37) і KDE (рис. 3.38).

Рис. 3.36. X Window System в Mac OS X

X Window (чи просто X) – це система для створення графічного інтерфейсу користувача на комп'ютерах, що функціонують під управлінням POSIX-сумісних операційних систем. В X Window System не визначаються деталі інтерфейсу користувача – відповідальність за них покладається на менеджери вікон. Із цієї причини зовнішній вигляд програм у середовищі X Window може дуже сильно розрізнятися; в різних програмах можуть використовуватися зовсім не схожі інтерфейси.

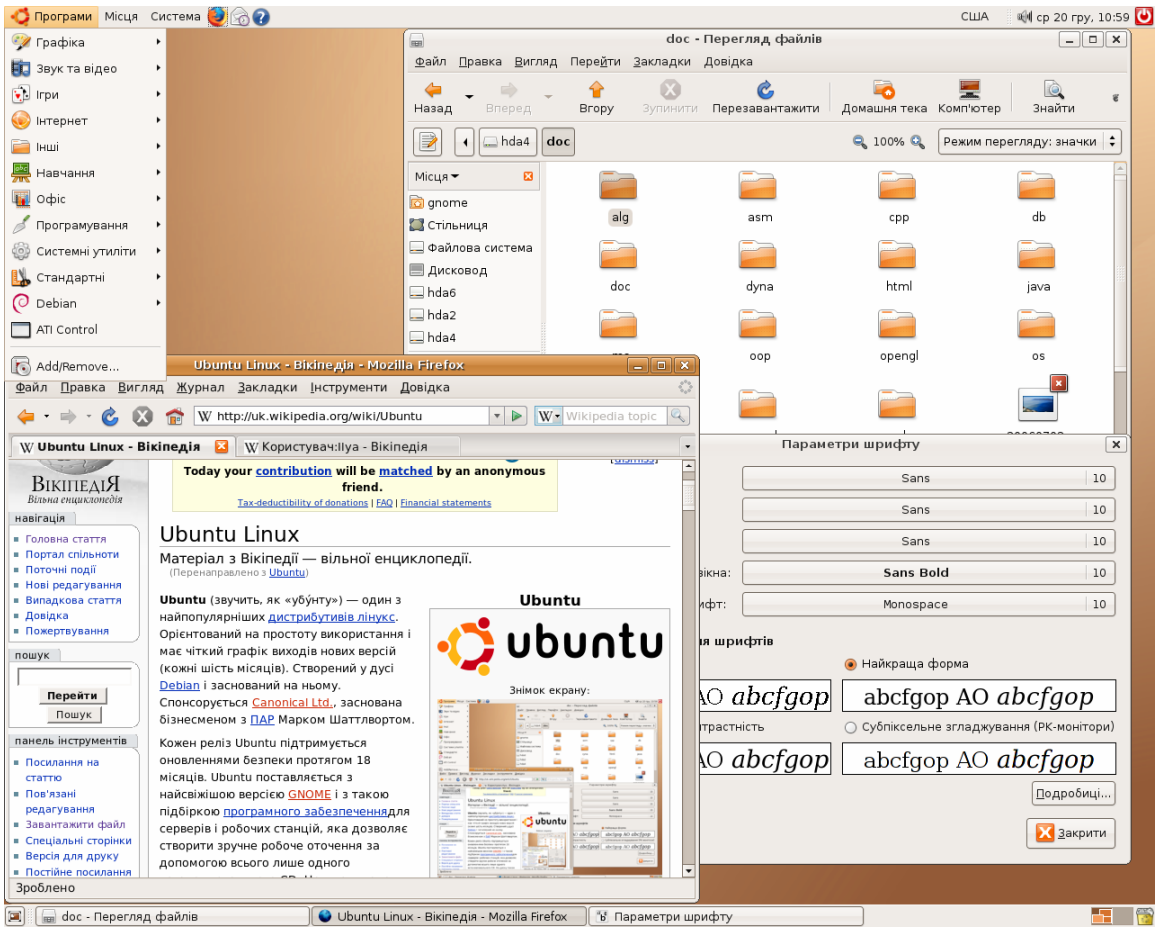


Рис. 3.37. GNOME 2.20 в середовищі X Window System

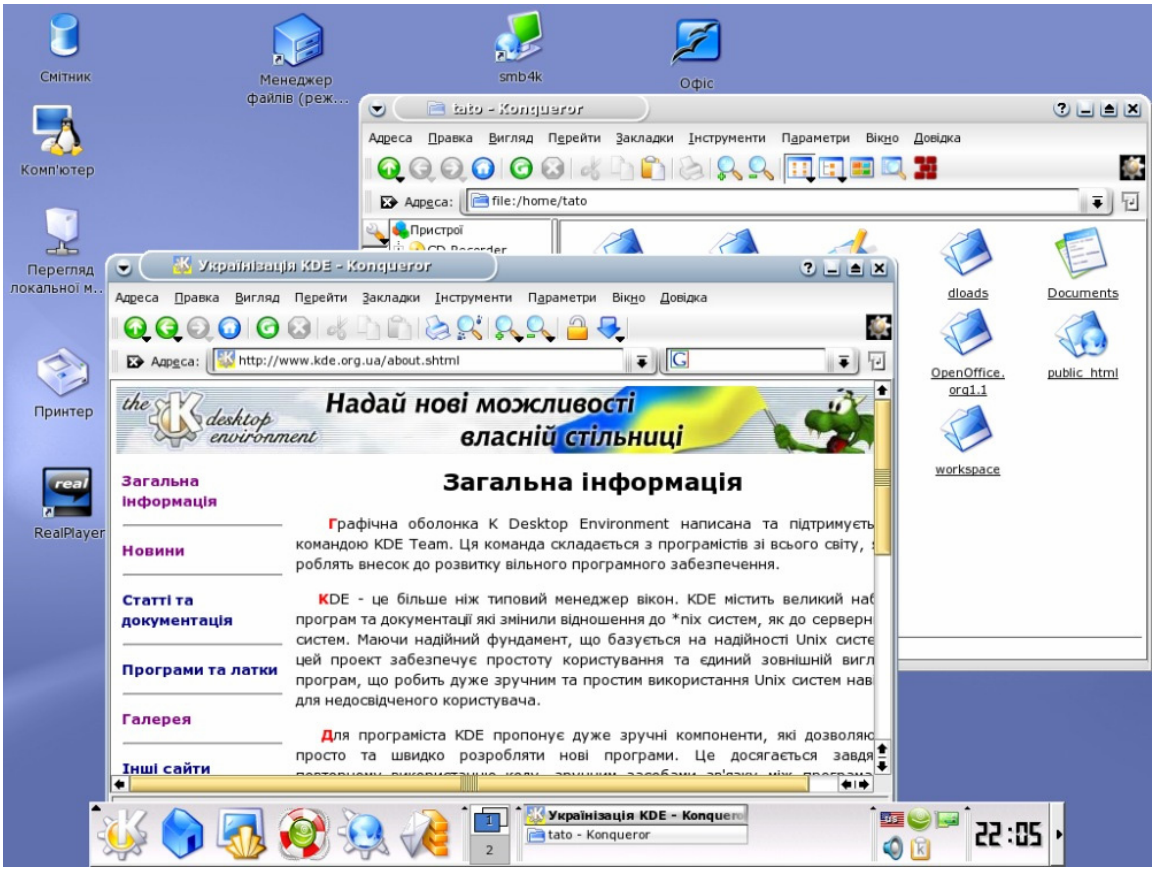


Рис. 3.38. KDE 4 в середовищі X Window System

За допомогою віконного менеджеру здійснюється управління розміщенням і зовнішнім виглядом вікон. Це надає можливість створення інтерфейсу, подібного до Microsoft Windows або Macintosh (наприклад, засобами віконних менеджерів Kwin в KDE і Metacity в GNOME), або зовсім інший стиль (наприклад, у фреймових віконних менеджерах, таких, як Ion). Віконний менеджер може бути простим і мінімалістичним (як twm – базовий віконний менеджер, що поставляється з X), а може пропонуватися функціональність, наближена до повноцінного робочого середовища (наприклад, Enlightenment).

X була створена в Массачусетському технологічному інституті (США) в 1984 році. Нинішня версія протоколу – X11 – з'явилася у вересні 1987 року (один цей факт підкреслює високу стабільність протоколу та системи).

Особливістю системи є те, що вона може використовуватись для роботи як на окремій ЕОМ, так і в мережі. Це означає, що програма, перебіг якої відбувається на одному комп'ютері, може за допомогою X Window використовуватися людиною, що працює за іншою машиною: через систему забезпечується виведення графічних даних на екран його машини, сприймання сигналів від зовнішніх пристроїв, таких як клавіатура і миша, і передавання їх до програми. Пристрій виведення може мати кілька екранів – в X забезпечується відображення на будь-якому з них. Все це: екран (екрани), а також пристрої введення (клавіатура та миша) в термінах X Window називають *дисплеєм*.

Використовуючи X, можна працювати з багатьма програмами одночасно. Щоб виведення з них не змішувалося, в системі створюються на екрані дисплея «віртуальні» підекрани – *вікна*. Кожній програмі, як правило, відповідає своє вікно чи вікна. В X є набір засобів для створення вікон, їх переміщення на екрані і зміни їх розмірів.

Особливістю X Window є те, що за її допомогою можна організувати обмін даними між самими програмами і між програмами та зовнішнім середовищем шляхом розсилання подій. *Подія* – це набір даних, за допомогою яких ідентифікуються дії, які виконуються в системі, та додаткові дані про них.

В X Window передбачена мережна прозорість: графічні програми можуть виконуватися на іншій машині в мережі, а їх інтерфейс при цьому буде передаватися через мережу й відображатися на локальній машині користувача. У контексті X Window терміни «клієнт» і «сервер» мають незвичне для багатьох користувачів значення: сервер означає локальний дисплей користувача (дисплейний сервер), а клієнт – програму, при роботі якої цей дисплей використовується (вона може виконуватися на віддаленому комп'ютері).

Система X Window представляє собою сукупність програм і бібліотек. Серцем її є окремий UNIX-процес, що функціонує на комп'ютері, до якого приєднаний дисплей. Саме *сервер* враховує особливості конкретної апаратури: що треба зробити, щоб зафарбувати піксель на екрані, намалювати лінію чи інший графічний об'єкт, як опрацювати сигнали, що приходять від клавіатури і миші.

Сервер обмінюється даними з програмами-клієнтами. Якщо перебіг сервера і клієнта відбувається на різних машинах, то дані пересилаються через мережу, якщо ж комп'ютер один, то для передавання даних використовується внутрішній канал. Наприклад, якщо сервер виявляє, що натиснуто кнопку миші, то він готує відповідний пакет і надсилає його тому клієнту, у вікні якого знаходиться курсор миші. І навпаки, якщо в ході програми потрібно що-небудь вивести на екран дисплея, створюється необхідний пакет даних, який надсилається до сервера.

Склад пакетів і їх черговість визначаються за спеціальним протоколом. Але щоб програмувати для X, зовсім не обов'язково знати деталі реалізації сервера і протоколу обміну. В системі є бібліотека процедур Xlib, за допомогою яких з програм здійснюється доступ до послуг X на високому рівні.

Протокол, за допомогою якого узгоджується функціонування сервера і клієнта, є прозорим для мережі: клієнт і сервер можуть перебувати як на одній і тій самій машині, так і на різних. Зокрема, вони можуть функціонувати на різних апаратних архітектурах під управлінням різних операційних систем – результат буде однаковим. Клієнт і сервер можуть з'єднуватися через Інтернет за допомогою туннельованого з'єднання через зашифрований мережний сеанс.

X-сервери можуть також існувати всередині інших графічних середовищ. В OpenVMS – операційній системі компанії Hewlett-Packard – як стандартне середовище робочого стола використовується версія X разом з CDE, відома як DECwindows. До складу операційних систем Mac OS X 10.3 (Panther) і вище від Apple входить X11.app на основі XFree86 4.3 і X11R6.6. Windows не містить у собі підтримку X, але існують численні сторонні реалізації: як вільно поширювані (Cygwin/X, Xming, X-Deep/32, WeirdMind, Weird), так і комерційні (Xmanager, Wired, Exceed, X-Win32).

Коли X Window використовується всередині іншої віконної системи (наприклад, віконної підсистеми Windows або Mac OS), вона звичайно функціонує в режимі без кореневого вікна. Це означає, що кореневе вікно (тло екрана й пов’язані з ним меню) управляється за допомогою зовнішньої віконної системи, а не самою X Window. При цьому через зовнішню віконну систему також здійснюється управління геометрією X-вікон, створюваних усередині неї. Однак деякі сервери (наприклад, Exceed, Xming та Cygwin/X) здатні створювати й кореневе вікно – у цьому випадку клієнти відображаються в окремому вікні в зовнішній системі.

Незалежність від апаратури і відокремлення клієнтів від серверів впливає на продуктивність системи – у минулому це істотно знижувало її (у порівнянні з Windows і Mac OS, де віконна підсистема була інтегрована глибоко в саму операційну систему). Для нормального функціонування X Window рекомендувалося від 4 до 8 Мб оперативної пам’яті – значно більше (в ті часи), ніж для Windows або Mac OS.

В поточних версіях Windows і Mac OS X передбачено внутрішній поділ графічної підсистеми, схожий на клієнт-серверний поділ в X, і потрібні приблизно ті ж ресурси, що й для X з KDE або GNOME. Більша частина накладних витрат в X тепер припадає на затримку при передаванні даних через мережу між клієнтом і сервером.

На жаль, кількість як перекладених, так і оригінальних видань, присвячених подіє-орієнтованому програмуванню в X Window з використанням Xlib,

дуже мала, що не в останню чергу зумовлено комерційною орієнтацією на застосування засобів швидкої розробки програм (Kylix, KDevelop, Eclipse) та відповідних об’єктно-орієнтованих бібліотек візуальних компонентів. Застосування таких засобів спрощує процес програмування, проте приховується сама основа подіє-орієнтованого програмування – безпосереднє опрацювання подій, що негативно вливає на рівень розуміння студентами відповідних механізмів та суттєво звужує коло розв’язуваних ними задач.

Це змусило нас вдатися до розробки навчальних посібників (орієнтованих на застосування мобільних компіляторів Free Pascal [248] та GCC [247]) та лабораторного практикуму з подіє-орієнтованого програмування в X Window, побудованого за принципом поступового зростання рівня абстракцій. В першому розділі посібника описується X протокол та основи Xlib – віконна структура, виведення тексту та графіки, робота з зовнішніми пристроями та програмними ресурсами, засоби обміну даними між клієнтами. В другому розділі розглядаються бібліотека Xt та її нащадки як найпростіший засіб автоматизації опрацювання повідомлень.

Апробація курсу подіє-орієнтованого програмування надала можливість зробити такі висновки:

1. Застосування POSIX-сумісних систем та графічної підсистеми X Window надають можливість організувати процес навчання на основі локалізованого, ліцензійно чистого, стабільного та мобільного програмного забезпечення.

2. З метою кращого розуміння студентами основ подіє-орієнтованого програмування вивчення методів побудови інтерфейсу користувача доцільно починати не з об’єктно-орієнтованих бібліотек візуальних компонентів та засобів швидкої розробки програм, а з розгляду базових механізмів опрацювання повідомлень та клієнт-серверної взаємодії на рівні X протоколу.

3. Процедурна природа Xlib, Xt, Athena та Motif надає можливість розпочати опанування подіє-орієнтованого програмування користувацького інтерфейсу одразу після вивчення будь-якої мобільної мови процедурного програму-

вання (C, Pascal і т.п.), що дає додаткову мотивацію студентам, роблячи їх програми більш наочними без суттєвого ускладнення структури та необхідності раннього переходу до об’єктно-орієнтованої технології. В той же час об’єктно-орієнтована бібліотека Qt (стандарт де-факто в програмуванні графічного інтерфейсу вільно поширюваних UNIX-систем) може бути використана в якості ілюстрації в процесі вивчення об’єктно-орієнтованого програмування мовами C++, Object Pascal, Python та ін.

4. Послідовне опанування засобів бібліотек X Window надає можливість організувати процес навчання за принципом поступового зростання рівня абстракцій з поверненням на кожному новому рівні до засвоєного на попередньому етапі матеріалу, реалізовуючи у такий спосіб фундаменталізацію змісту курсу.

Висновки до розділу 3

Синтез методичної системи фундаментальної інформатичної підготовки фахівців у галузі інформаційних технологій у вищих навчальних закладах, аналіз монографій та дисертаційних досліджень з питань інформатики та методики її навчання, узагальнення досвіду застосування сучасних мережних та мобільних технологій в навчальному процесі ВНЗ, власного досвіду з впровадження авторських навчальних програм, посібників, підручників та програмних засобів дозволили зробити наступні висновки:

1. Фундаменталізація інформатичної освіти вимагає посилення ролі обчислювального експерименту та програмування:

– обчислювальний експеримент є методологією інформатики як науки, тому його можна віднести до принципів (методології) наукових методів учіння;

– цілі навчання інформатики у вищій школі включають необхідність засвоєння як певної сукупності наукових фактів, так і методів отримання цих фактів, які використовуються в самій науці, а програмування відображає метод пізнання, що застосовується в інформатиці. При цьому під терміном «програмування» розуміється діяльність, яка у вузькому сенсі зводиться до простого ко-

дування відомого алгоритму, а в широкому – співпадає з методологією інформатики, тобто є тотожною обчислювальному експерименту.

2. До інноваційних методів навчання інформатичних дисциплін відноситься парне програмування – форма розробки програмного забезпечення, за якої програма для розв'язування поставленої задачі створюється парою програмістів, котрі працюють за одним робочим місцем. У парному програмуванні основна взаємодія відбувається між двома студентами, котрі можуть обговорювати завдання, здійснювати взаємонавчання або взаємоконтроль. Даний метод є також і формою організації навчальної діяльності, за якої два студенти-програмісти показують більш, ніж у двічі більшу продуктивність, в порівнянні з тим, коли вони працюють поодинці. За дистанційної форми навчання парне програмування реалізується через віддалене парне програмування – спосіб реалізації парного програмування, при якому обидва розробники, що складають пару, фізично знаходяться у різних місцях, і працюють за допомогою партнерського редактора реального часу, спільної розподіленої стільниці або спеціального модуля IDE для віддаленого парного програмування.

3. Фундаменталізація навчання інформатичних дисциплін вимагає стабілізації засобів навчання через надання їм властивостей відкритих систем. Стабілізація програмних засобів надає широкі можливості для варіювання програмного забезпечення навчання інформатичних дисциплін замість штучної прив'язки до окремих програмних продуктів. До стабільного програмного забезпечення навчання інформатичних дисциплін у вищій школі відносяться мобільні операційні системи, мобільні компілятори, мобільні інтерпретовані мови програмування, відкриті математичні системи, спеціалізовані предметні середовища та Web-середовища.

4. Один із загальноприйнятих способів підвищення мобільності програмного забезпечення – стандартизація програмного оточення: програмних інтерфейсів, утиліт тощо. На рівні системних сервісів подібне оточення описується в стандарті POSIX, підтримка якого полегшує перенесення прикладних програм практично на будь-яку скільки-небудь поширену операційну платформу. Мобі-

льність програм, що відповідають стандарту POSIX, принципово досяжна завдяки наявності великої кількості стандартизованих системних сервісів та можливості динамічного з'ясування характеристик цільової платформи й налаштовування програми під них. POSIX-сумісність є засобом уніфікації операційних систем, а дотримання стандартів POSIX при розробці програмного забезпечення – засобом уникнення залежності від використовуваної операційної системи.

5. Застосування мобільних компіляторів є засобом уникнення залежності від використовуваних середовища програмування (через потужний інтерфейс командного рядка та легкість інтеграції у IDE), операційної системи (через забезпечення POSIX-сумісності) та мови програмування (через надання спільних бібліотек).

6. Застосування мобільних інтерпретованих мов загального призначення є засобом забезпечення мобільності програм, створених на POSIX-несумісних платформах.

7. Відкриті вільно поширювані мобільні математичні системи відзначаються тривалою історією розвитку, оптимізованими алгоритмами, POSIX-сумісністю, невимогливістю до ресурсів, ліцензійною чистотою, безоплатністю, різноманітністю інтерфейсів, локалізованістю та іншими перевагами, які дозволяють застосовувати їх в якості стабільного програмного забезпечення математичного призначення.

8. Добір спеціалізованих предметних середовищ навчання інформатичних дисциплін виконується з відкритих мобільних програмних систем навчального призначення, що мають широку інсталяційну базу та придатні для локалізації.

9. Використання мобільних пристроїв з невисокою швидкодією та малим обсягом оперативної пам'яті суттєво ускладнює застосування таких ресурсоємних програм, як середовища програмування, системи комп'ютерної математики і т.п. Для розв'язання цієї проблеми доцільно перейти до мережецентричної моделі, за якої ресурсоємні програми працюють на Інтернет-серверах, а головним клієнтським додатком виступає Web-браузер. Перенесення прикладного програмного забезпечення у Web-середовище (онлайн-IDE, Web-СКМ ті ін.)

створює нові можливості для обміну навчальними матеріалами та організації співробітництва між усіма учасниками навчального процесу:

– для будь-якого користувача за рахунок цього надається можливість мобільного доступу до програм та даних;

– для адміністратора комп'ютерного класу знімаються проблеми підтримки великої інсталяційної бази та ліцензування програмного забезпечення;

– для викладачів суттєво розширюється спектр використовуваного програмного забезпечення, а для студентів – використовуваних засобів мобільного навчання.

Основні результати третього розділу опубліковані в роботах [68; 81; 82; 141; 174; 176; 177; 181; 188; 189; 243; 244; 246; 247; 248; 249; 250; 253; 255; 256; 259; 297; 298; 300; 304; 305; 309; 310; 312; 313; 325; 316; 317; 318; 321; 323; 324; 328; 341; 366; 369; 370; 371; 372; 374; 379; 380; 384; 388; 426; 436; 437; 442].

РОЗДІЛ 4
ОЦІНЮВАННЯ ЕФЕКТИВНОСТІ ФУНДАМЕНТАЛІЗАЦІЇ НАВЧАННЯ ІНФОРМАТИЧНИХ ДИСЦИПЛІН У ВИЩИХ НАВЧАЛЬНИХ ЗАКЛАДАХ

4.1. Констатувальний експеримент

У 1999 р. за ініціативою декана фізико-математичного факультету Криворізького державного педагогічного університету В.М. Соловйова автором був проведений порівняльний аналіз навчальних планів спеціальності «Математика та основи інформатики» Національного педагогічного університету імені М.П. Драгоманова (НПУ), Кіровоградського державного педагогічного університету імені В. Винниченка (КДПУ), Рівненського державного педагогічного інституту (РДПІ), Тернопільського державного педагогічного університету та Криворізького державного педагогічного університету 1995 (КР95) та 1999 (КР99) років (табл. 4.1).

Таблиця 4.1

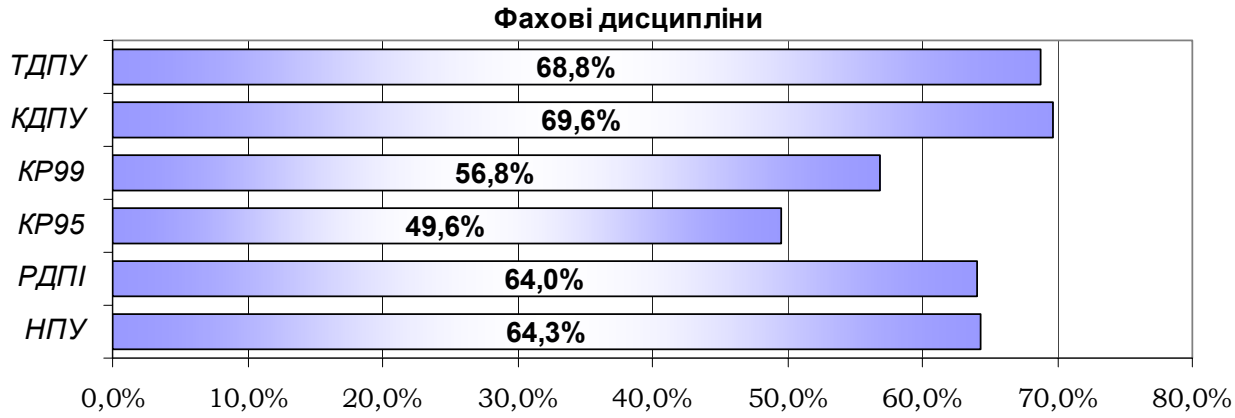
Порівняльний аналіз навчальних планів спеціальності «Математика та основи інформатики» (станом на 1999 р.)

Блок	*НПУ*	*РДПІ*	*КР95*	*КР99*	*КДПУ*	*ТДПУ*	max Δ
Фахові дисципліни та дисципліни спеціалізації	64,3%	64,0%	49,6%	56,8%	69,6%	68,8%	20,0%
Психолого-педагогічні дисципліни	16,4%	17,5%	25,5%	21,4%	12,1%	11,4%	13,4%
Гуманітарні та соціально-економічні дисципліни	19,4%	18,5%	24,9%	21,8%	18,3%	19,9%	6,6%

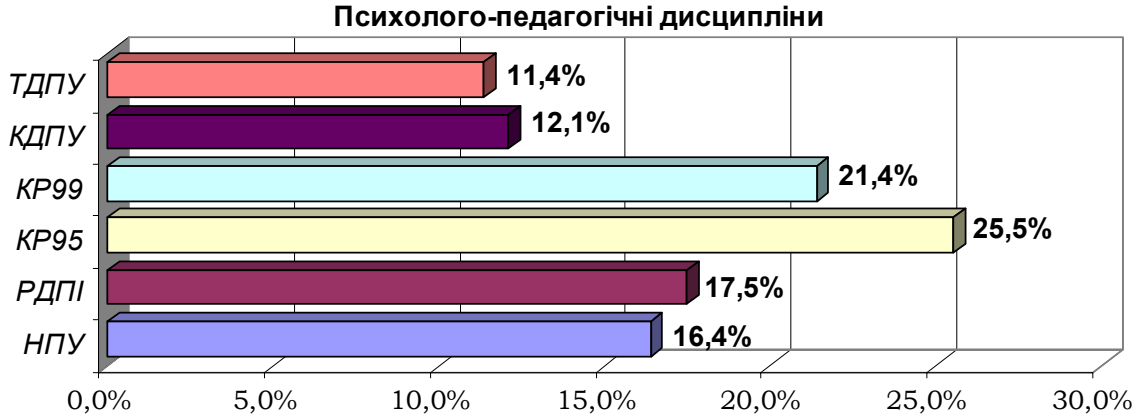
Графічну інтерпретацію результатів аналізу подано на рис. 4.1, 4.2.

Виявлені розбіжності у долі блоків фахових дисциплін та дисциплін спеціалізації (від 4,5% до 20%) в навчальних планах не лише були причиною різного рівня професійної підготовки випускників, що навчалися за однією спеціальністю, а й суттєво ускладнювали студенту перехід до іншого ВНЗ для про-

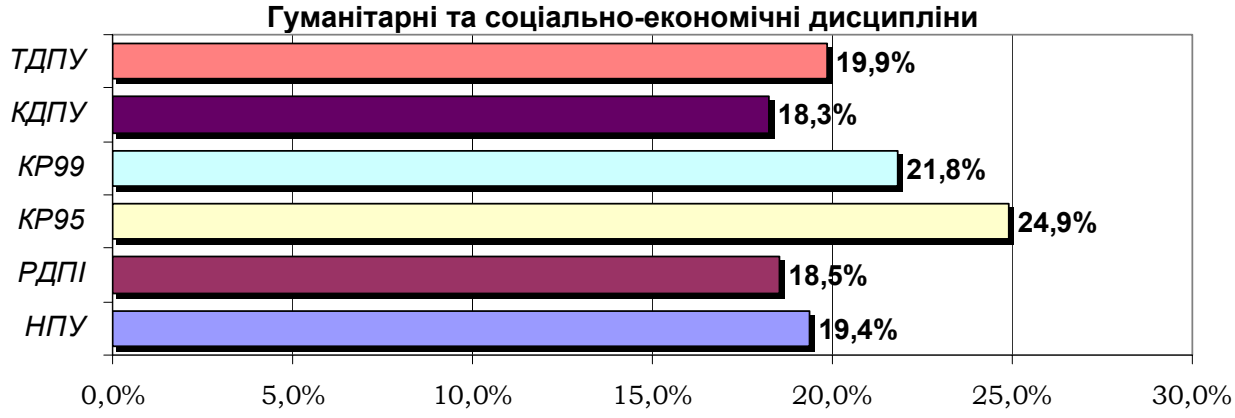
довження навчання через велику – в межах однієї спеціальності – академічну різницю.



а)



б)



в)

Рис. 4.1. Частка блоків у навчальних планах спеціальності «Математика та основи інформатики»: *а* – фахові дисципліни та дисципліни спеціалізації; *б* – психолого-педагогічні дисципліни; *в* – гуманітарні та соціально-економічні дисципліни

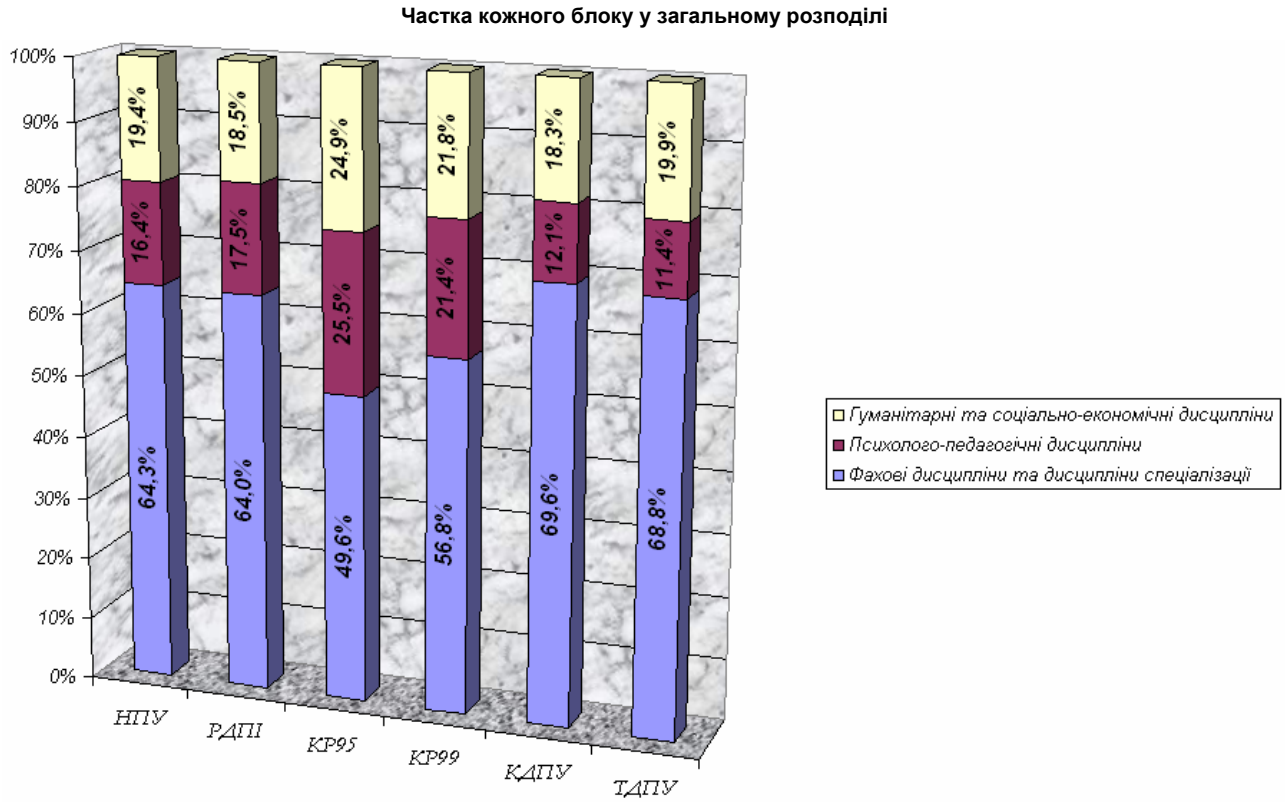


Рис. 4.2. Частка блоків у загальному розподілі в навчальних планах спеціальності «Математика та основи інформатики»

Таким чином, наприкінці минулого століття виникла нагальна необхідність, по-перше, у прийнятті державних стандартів вищої освіти, в яких би визначалися фахові компетентності випускника та зміст фахової підготовки, і, по-друге, у засобах забезпечення мобільності студентів у процесі навчання.

Виявлена соціальна потреба у розробці стандартів інформатичної освіти та підвищенні рівня навчальної мобільності студентів і професійної мобільності випускників спонукала до початку констатувального експерименту (2001–2002 рр.), завданням якого було вивчення існуючого стану досліджуваного явища та виділення вихідних положень дослідження.

Для реалізації поставленого завдання було організовано та проведено три всеукраїнські конференції з проблем застосування інформаційних технологій і комп’ютерного моделювання в освіті та методики навчання фізико-математичних та інформатичних дисциплін:

1) III Всеукраїнська конференція молодих науковців «Комп'ютерне моделювання та інформаційні технології в науці, економіці та освіті» (26–28 квітня 2001 року, м. Кривий Ріг, Криворізький державний педагогічний університет);

2) Всеукраїнська науково-методична конференція «Теорія та методика навчання математики, фізики, інформатики» (26–28 квітня 2001 року, м. Кривий Ріг, Криворізький державний педагогічний університет);

3) II Всеукраїнська науково-методична конференція «Теорія та методика навчання математики, фізики, інформатики» (25–26 квітня 2002 року, м. Кривий Ріг, Національна металургійна академія України).

Аналіз матеріалів учасників перших двох конференцій показав спільність проблем, що виникають в процесі навчання інформатичних дисциплін у вищих навчальних закладах різного профілю, та необхідність застосування спільних підходів до навчання споріднених дисциплін – математики, інформатики, фізики. Залучення до експерименту вищого технічного навчального закладу (НМетАУ) підтвердило це припущення, надавши можливість перейти від розгляду окремих методик навчання у ВНЗ різного профілю до розгляду більш загальних методик навчання фундаментальних дисциплін у вищій школі.

Результати констатувального експерименту виявили наступне:

1. В розвитку інформатичних дисциплін існує ряд проблем та протиріч: виключно швидкий прогрес програмних та технічних засобів; навчальний матеріал швидко втрачає актуальність та постійно потребує заміни більш сучасним, причому застаріває не лише зміст, а й структура; існуюче методичне, програмне, технічне забезпечення інформатичних дисциплін швидко втрачає актуальність та застаріває.

2. Неусталеність методичних систем навчання інформатичних дисциплін викликана її помилковим позиціонуванням як технологічних дисциплін, вторинних в порівнянні з фундаментальними дисциплінами.

3. Нечіткість, розмитість вимог до змісту навчальних і робочих програм інформатичних дисциплін призводить до значного розсіювання знань у студентів та знижує їх навчальну мобільність.

У результаті стало зрозумілим, що є системна, комплексна проблема, сутність якої полягає у невідповідності між підготовкою фахівців з інформатики її фундаментальному характеру і гіперболізації ролі програмно-технічних засобів навчання, з одного боку, та потенціалом фундаментальної підготовки і мобільних технологій, з другого боку.

4.2. Пошуковий експеримент

Процес пошуку шляхів розв'язання поставленої комплексної проблеми та результати експериментальної роботи (2003–2005 рр.) відображені у матеріалах п'яти всеукраїнських конференцій і семінарів з комп'ютерного моделювання в освіті та методики навчання фундаментальних дисциплін у вищій школі, організованих автором та при його безпосередній участі:

1) Всеукраїнська науково-методична конференція «Теорія та методика навчання фундаментальних дисциплін у вищій технічній школі» (14–15 березня 2003 року, м. Кривий Ріг, Національна металургійна академія України);

2) V Всеукраїнська науково-практична конференція «Комп'ютерне моделювання та інформаційні технології в науці, економіці та освіті» (21–23 квітня 2003 року, м. Черкаси, Інститут соціального управління, економіки і права);

3) IV Всеукраїнська науково-практична конференція «Теорія та методика навчання фундаментальних дисциплін у вищій школі» (8–9 квітня 2004 року, м. Кривий Ріг, Національна металургійна академія України);

4) Всеукраїнський науково-методичний семінар «Комп'ютерне моделювання в освіті» (29 березня 2005 року, м. Кривий Ріг, Криворізький державний педагогічний університет);

5) V Всеукраїнська науково-практична конференція «Теорія та методика навчання фундаментальних дисциплін у вищій школі» (8–9 квітня 2005 року, м. Кривий Ріг, Національна металургійна академія України).

В ході пошукового експерименту були отримані такі основні результати:

1. Розроблено концепцію курсу подіє-орієнтованого програмування в системі X Window, незалежну від операційної системи та мови програмування [253;

255]; створено два навчальні посібники: «Програмування в X Window» [247] та «Програмування в X Window засобами Free Pascal» [248], апробовані під управлінням різних операційних систем в процесі навчання дисциплін «Методи математичного моделювання» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики»), «Візуальне програмування» (спеціальність «Математика та основи інформатики»), «Подіє-орієнтоване програмування» (спеціальність «Інформатика»), «Проектування програмних інтерфейсів» (спеціальність «Комп'ютерні системи управління та автоматики»).

2. Розроблено методичні основи навчання операційних систем майбутніх вчителів інформатики, узагальнені для всіх сучасних операційних системи, що відповідають стандарту POSIX [305]; апробовані під управлінням різних операційних систем в процесі навчання дисциплін «Операційні системи» (спеціальність «Інформатика»), «Сучасні операційні системи» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики»), «Функціональні можливості сучасних операційних систем» (спеціальність «Математика та основи інформатики»), «Сучасні операційні системи та мережі» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики»).

3. Узагальнено досвід використання вільно поширюваного програмного забезпечення у підготовці майбутнього вчителя інформатики, визначено шляхи швидкого переходу на вільне програмне забезпечення еволюційним шляхом без втрати набутих раніше загальних методів та прийомів роботи, показано перспективи впровадження вільно поширюваного програмного забезпечення у підготовку майбутніх інженерів-програмістів [374].

4. Узагальнено досвід розробки динамічних Web-додатків навчального призначення [68; 81; 313; 328; 367], апробованих в процесі навчання дисципліни «Web-програмування» (спеціальність «Комп'ютерні системи та мережі»).

5. Розроблено методичні основи навчання системного програмного забезпечення у підготовці майбутнього вчителя інформатики та інженера-програміста [244; 304; 312], апробовані під управлінням різних операційних систем засобами різних мов програмування в процесі навчання дисциплін «Сис-

темне програмування» (спеціальності «Гнучкі комп'ютеризовані системи та робототехніка», «Комп'ютерні системи управління та автоматики»), «Системне програмне забезпечення» (спеціальності «Інформатика», «Комп'ютерні системи та мережі»), «Обчислювальна техніка» (спеціальність «Трудове навчання та основи інформатики»), «Елементи професійного програмування» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики»).

6. Досліджено психолого-педагогічні та методичні засади навчання майбутніх вчителів інформатики технології комп'ютерного моделювання [258; 311; 342; 343; 370; 375; 378; 379; 380; 381; 382; 383; 384; 385; 386; 389; 392; 395; 396; 398], реалізовані в навчальному посібнику з лабораторного практикуму «Методи математичного моделювання» [341], апробованому в процесі навчання дисциплін «Імітаційне моделювання» (спеціальність «Комп'ютерні системи та мережі»), «Комп'ютерне моделювання» (спеціальності «Комп'ютерні системи та мережі», «Математика та основи інформатики», «Фізика та основи інформатики»), «Методи математичного моделювання» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики»), «Математичне моделювання» (спеціальність «Інформатика»), «Математичне моделювання та системний аналіз» (спеціальність «Інформатика»), «Моделювання» (спеціальність «Математика та основи інформатики»), «Об'єктно-орієнтоване програмування» (спеціальності «Інформатика», «Фізика та основи інформатики», «Хімія та основи інформатики»), «Шкільний курс інформатики» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики»).

7. Розроблено систему дистанційного тестування знань у мережах Fidonet [314; 315] та Інтернет [313].

8. Досліджено дидактичні можливості мови Лісп як засобу побудови інтелектуальних систем в курсі інформатики [324; 371; 372], реалізовані у навчальному посібнику з лабораторного практикуму «Вступ до програмування систем штучного інтелекту мовою Лісп» [366], апробованому в процесі навчання дисциплін «Методи математичного моделювання» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики»), «Елементи функці-

онального програмування» (спеціальність «Інформатика»), «Розпізнавання образів» (спеціальність «Інформатика»), «Інтелектуальні та експертні системи» (спеціальності «Інформатика», «Математика та основи інформатики»), «Комп'ютерні системи штучного інтелекту» (спеціальності «Комп'ютерні системи та мережі», «Економічна кібернетика»).

9. Розглянуто філософські та технологічні аспекти переходу до інформаційного суспільства [377].

10. Розглянуто принципи розробки та впровадження СКМ Maxima [174; 318], розроблено гіпертекстовий довідник із системи Maxima [316; 317], апробований в процесі навчання дисциплін «Теорія автоматичного управління» (спеціальності «Гнучкі комп'ютеризовані системи та робототехніка», «Комп'ютерні системи управління та автоматики»), «Теорія управління» (спеціальність «Інформатика»), «Комп'ютерні технології в наукових дослідженнях» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики»).

11. Запропоновано підхід до розробки навчальних планів з інформатичних дисциплін у вищій школі на основі наскрізного навчання комп'ютерного моделювання як засобу інтеграції різних дисциплін [245; 258].

12. Розроблені методичні основи застосування метакомп'ютерних систем в процесі навчання математичних та інформатичних дисциплін [181; 426], апробовані в процесі навчання дисципліни «Паралельні і розподілені обчислення» (спеціальність «Інформатика»).

Результати пошукового етапу експерименту дали можливість виявити наступні напрями фундаменталізації змісту навчання інформатичних дисциплін:

1. Чітке виділення в змісті навчання фундаментальних основ навчального предмета, що відповідають фундаментальним основам предметної галузі, через посилення ролі фундаментальної природничо-наукової частини інформатики – математичної інформатики, що є теоретичною основою інформаційної технології.

2. Зміщення уваги викладачів та студентів з проблеми набуття прагматичних знань на проблеми розвитку інформаційної культури та формування сис-

темного мислення на основі розуміння сутності інформаційних процесів, побудова курсів інформатики від феномена інформації та інформаційних процесів до методів їх вивчення за допомогою інформаційних моделей.

3. Інтеграція математичної інформатики та інформаційних технологій на основі комп'ютерного моделювання.

4.3. Формувальний експеримент

Реалізація виявлених напрямів фундаменталізації навчання інформатичних дисциплін та результати експериментальної роботи (2006–2008 рр.) відображені у матеріалах дев'яти всеукраїнських та міжнародних конференцій і семінарів, організованих автором та за його безпосередньої участі:

1) Всеукраїнський науково-методичний семінар «Комп'ютерне моделювання в освіті» (26 квітня 2006 року, м. Кривий Ріг, Криворізький державний педагогічний університет);

2) VI Міжнародна науково-практична конференція «Теорія та методика навчання фундаментальних дисциплін у вищій школі» (27–28 квітня 2006 року, м. Кривий Ріг, Національна металургійна академія України);

3) IV Міжнародна науково-технічна конференція «Комп'ютерні технології в будівництві» (18–21 вересня 2006 року, м. Севастополь, Державний науково-дослідний інститут автоматизованих систем у будівництві);

4) VII Всеукраїнська науково-практична конференція «Комп'ютерне моделювання та інформаційні технології в науці, економіці і освіті» (24–25 квітня 2007 року, м. Кривий Ріг, Криворізький економічний інститут);

5) V Міжнародна науково-технічна конференція «Комп'ютерні технології в будівництві» (18–21 вересня 2007 року, м. Севастополь, Державний науково-дослідний інститут автоматизованих систем у будівництві);

6) VII Міжнародна науково-практична конференція «Теорія та методика навчання фундаментальних дисциплін у вищій школі» (17–18 квітня 2008 року, м. Кривий Ріг, Національна металургійна академія України);

7) VIII Всеукраїнська науково-практична конференція «Комп'ютерне моделювання та інформаційні технології в науці, економіці і освіті» (22–23 квітня 2008 року, м. Кривий Ріг, Криворізький економічний інститут);

8) III Всеукраїнський науково-методичний семінар «Комп'ютерне моделювання в освіті» (24 квітня 2008 року, м. Кривий Ріг, Криворізький державний педагогічний університет);

9) VI Міжнародна науково-технічна конференція «Комп'ютерні технології в будівництві (КОМТЕХБУД–2008)» (9–12 вересня 2008 року, м. Севастополь, Державний науково-дослідний інститут автоматизованих систем у будівництві).

Розширення тематики заходів з комп'ютерного моделювання в освіті та методики навчання фундаментальних дисциплін у вищій школі до комп'ютерного моделювання та інформаційних технологій в освіті, методики навчання фундаментальних дисциплін у вищій школі та проблем підготовки та перепідготовки фахівців у галузі інформаційних технологій була викликана необхідністю реалізації задач дослідження, пов'язаних із фундаменталізацією навчання інформаційних технологій та впровадженням технологій мобільного навчання. Основну увагу в ході формувального експерименту було приділено технологічним та методичним засобам стабілізації навчання інформатичних дисциплін, подовження терміну «життя» знань, підвищенню професійної мобільності майбутніх фахівців.

Основними результатами даного етапу дослідження є наступні:

1. Розроблено курс системного програмування в POSIX-системах з використанням мобільного програмного забезпечення та мобільних інтерпретованих мов програмування [246; 256], реалізований в навчальному посібнику [249], апробованому в процесі навчання дисциплін «Системне програмування» (спеціальності «Гнучкі комп'ютеризовані системи та робототехніка», «Комп'ютерні системи управління та автоматики», «Інформатика», «Комп'ютерні системи та мережі», «Програмне забезпечення автоматизованих систем»), «Операційні системи» (спеціальності «Інформатика», «Системи управління і автоматики»),

«Сучасні операційні системи» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики»), «Функціональні можливості сучасних операційних систем» (спеціальність «Математика та основи інформатики»), «Сучасні операційні системи та мережі» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики»).

2. Розроблено методику застосування мобільних інтерпретованих мов програмування в процесі навчання об'єктно-орієнтованого моделювання [176; 177; 369], апробовану в процесі навчання дисциплін «Комп'ютерне моделювання» (спеціальності «Комп'ютерні системи та мережі», «Математика та основи інформатики», «Фізика та основи інформатики»), «Комп'ютерні технології в навчанні» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики»), «Моделювання» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики», «Хімія та основи інформатики», «Програмне забезпечення автоматизованих систем»), «Об'єктно-орієнтоване програмування» (спеціальності «Інформатика», «Фізика та основи інформатики», «Хімія та основи інформатики»), «Обчислювальний практикум» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики»).

3. Запропоновано нову складову інформаційної культури – інформаційну безпеку [376].

4. Запропоновано шляхи легалізації програмного забезпечення в галузі вищої освіти [388].

5. Локалізовано оболонку експертних систем CLIPS [82], розроблено методичні основи її застосування як засобу вивчення експертних систем [309], реалізовані в навчальному посібнику «CLIPS: локалізована оболонка експертної систем для вітчизняної системи освіти» [297] та апробовані в процесі навчання дисциплін «Інтелектуальні та експертні системи» (спеціальності «Інформатика», «Математика та основи інформатики», «Професійне навчання»), «Інтелектуальні системи» (спеціальності «Інформатика», «Гнучкі комп'ютеризовані системи та робототехніка», «Комп'ютерні системи управління та автоматики», «Про-

грамне забезпечення автоматизованих систем»), «Комп'ютерні системи штучного інтелекту» (спеціальність «Комп'ютерні системи та мережі»), «Шкільний курс інформатики» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики»).

6. Досліджено дидактичні можливості пристроїв класу «електронна книга» як нового технічного засобу навчання [393], створено рекомендації з розробки програмного забезпечення для них [296].

7. Завершено локалізацію СКМ Maxima [325; 437], розроблено ряд нових мобільних інтерфейсів користувача до неї [141; 310; 442], виконано інтеграцію Maxima у СДН MOODLE [301; 302], створено довідник користувача до Maxima версії 5.13 [298], апробований в процесі навчання дисциплін «Комп'ютерні технології в наукових дослідженнях» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики»), «Методи обчислень» (спеціальності «Математика та основи інформатики», «Фізика та основи інформатики», «Інформатика», «Комп'ютерні системи та мережі», «Програмне забезпечення автоматизованих систем»).

8. Розроблено спецкурс із сучасних методів і технологій розробки програмних виробів компонентної архітектури [254; 259], елементи якого апробовані в процесі навчання дисциплін «Об'єктно-орієнтоване програмування» (спеціальності «Інформатика», «Фізика та основи інформатики», «Гнучкі комп'ютеризовані системи та робототехніка»), «Сучасні об'єктні технології» (спеціальність «Інформатика»).

9. Розроблено психолого-педагогічні та технологічні засади мобільного навчання [252; 306; 308; 321; 390; 391; 394; 436].

У п. 2.5.2 наведені основні результати експерименту із впровадження елементів мобільного навчання інформаційних технологій математичного призначення у старших класах шкіл нового типу на основі КПК, а в п. 2.5.4 – експерименту із застосування Web-СКМ SAGE та мобільного варіанту LCMS MOODLE. В апробації навчально-методичних матеріалів і програмних продуктів, створених у межах даного дослідження, брали участь викладачі і студенти

Криворізького державного педагогічного університету та інших ВНЗ: Криворізького технічного університету, Черкаського національного університету імені Богдана Хмельницького, Запорізького інституту економіки та інформаційних технологій, Кременчуцького університету економіки, інформаційних технологій і управління, Національної металургійної академії України та інших.

Експериментом на різних його етапах було охоплено понад 1200 студентів спеціальностей:

– педагогічних – «Математика та основи інформатики», «Фізика та основи інформатики», «Хімія та основи інформатики», «Трудове навчання та основи інформатики», «Професійне навчання»;

– економічних – «Економічна кібернетика»;

– інформатичних – «Інформатика», «Прикладна математика», «Комп'ютерні системи та мережі», «Програмне забезпечення автоматизованих систем», «Системи управління і автоматики», «Гнучкі комп'ютеризовані системи та робототехніка».

Результати педагогічного експерименту були статистично опрацьовані і за відповідними правилами прийняття рішень зроблено висновки про те, що розроблені методичні системи навчання є ефективнішими за традиційні не лише в напрямі формування у студентів фундаментальних знань та узагальнених навичок роботи, а й у напрямі підвищення технологічної та професійної мобільності.

Ефективність розробленої методичної системи навчання інформатичних дисциплін підтверджується також результатами написання під керівництвом автора понад 60 кваліфікаційних робіт бакалавра, спеціаліста та магістра, у переважній більшості з яких студентами застосовувались мобільні операційні системи та середовища програмування, розроблялися частини програмних проектів (в т.ч. – з локалізації та портування програмного забезпечення, розробки програмних інтерфейсів, інтеграції різних засобів тощо). Більшість студентів-випускників одержали за свої роботи відмінні оцінки державних екзаменаційних комісій Криворізького державного педагогічного університету, Криворізь-

кого технічного університету, Запорізького інституту економіки та інформаційних технологій та Кременчуцького університету економіки, інформаційних технологій та управління. Деякі з тем робіт, що стосуються окремих аспектів дослідження, наведені у додатку Е.

Майже всі студенти-дипломники, які працювали під керівництвом автора, вели наукові дослідження у галузі прикладної математики, комп'ютерних наук, методики навчання фізики, математики та інформатики, мали наукові публікації, виступали на наукових конференціях різного рівня. Багато хто з них одержав рекомендацію до аспірантури, навчався і навчається в ній, при цьому 3 з них (зокрема 1 під керівництвом автора) підготували до захисту кандидатські дисертації і успішно працюють в Криворізькому державному педагогічному університету та в інших ВНЗ м. Кривого Рогу.

Враховуючи комплексний характер роботи з фундаменталізації навчання інформатичних дисциплін, оцінювання її ефективності відбувалося за результатами державного екзамену зі спеціальностей «Інформатика» (І), «Комп'ютерні системи та мережі» (КСМ) та «Гнучкі комп'ютеризовані системи та робототехніка» (ГКС). Можливість використання результатів державного екзамену на різних спеціальностях забезпечувалась спільним блоком дисциплін, що виносяться на екзамен: «Операційні системи», «Системне програмування», «Системне програмне забезпечення», «Архітектура комп'ютерів», «Комп'ютерні мережі» та «Методи обчислень».

Контрольні та експериментальні групи формувалися наступним чином:

– до контрольних груп (КГ) відносилися студенти четвертих курсів (2003–2005 років випуску), що завершували навчання за освітньо-кваліфікаційним рівнем бакалавра за традиційною методичною системою без широкого використання фундаменталізованих інформатичних курсів та технологій мобільного навчання;

– до експериментальних груп (ЕГ) відносилися студенти четвертих курсів (2006–2008 років випуску), що завершували навчання за освітньо-кваліфікаційним рівнем бакалавра за створеною автором методичною системою

з широким упровадженням фундаменталізованих інформатичних курсів та технологій мобільного навчання (табл. 4.2, 4.3).

Таблиця 4.2

Розподіл студентів по групах в ході формувального експерименту

Потік	Кількість студентів	Рік випуску	Група
ГКС–99	47	2003	КГ
ГКС–00	46	2004	КГ
ГКС–01	46	2005	КГ
ГКС–02	35	2006	ЕГ
ГКС–03	40	2007	ЕГ
ГКС–04	48	2008	ЕГ
І–99	29	2003	КГ
І–00	27	2004	КГ
І–01	17	2005	КГ
І–02	20	2006	ЕГ
КСМ–119	10	2003	КГ
КСМ–110	1	2004	ЕГ
КСМ–111	7	2005	ЕГ
КСМ–112	2	2006	ЕГ
Разом:	375		

Таблиця 4.3

Розподіл студентів за роками випуску

Групи	Кількість студентів (за роками випуску)						Всього
	2003	2004	2005	2006	2007	2008	
Контрольні	86	73	63				222
Експериментальні		1	7	57	40	48	153
Всього	86	74	70	57	40	48	375

Дві групи 2004 та 2005 років випуску загальною кількістю 7 студентів були віднесені до експериментальних через те, що саме на них було розпочато відпрацювання пропонованої методики.

У табл. 4.4 показано розподіл оцінок державного екзамену в контрольних та експериментальних групах. Одержані дані було піддано статистичному опрацюванню з використанням кутового перетворення Фішера [61; 330], призначеного для співставлення двох емпіричних вибірок. В якості показника спостережуваного ефекту обрано якість навчання.

Таблиця 4.4

Розподіл оцінок державного екзамену
в контрольних та експериментальних групах

Потік	Група	Загальна кількість студентів	Кількість студентів, які отримали оцінку				Відсоток студентів, які отримали оцінку			
			2	3	4	5	2	3	4	5
ГКС–99	КГ	47	0	30	5	12	0,0%	63,8%	10,6%	25,5%
ГКС–00	КГ	46	0	21	13	12	0,0%	45,7%	28,3%	26,1%
ГКС–01	КГ	46	0	30	11	5	0,0%	65,2%	23,9%	10,9%
І–99	КГ	29	1	9	5	14	3,4%	31,0%	17,2%	48,3%
І–00	КГ	27	1	14	6	6	3,7%	51,9%	22,2%	22,2%
І–01	КГ	17	1	6	3	7	5,9%	35,3%	17,6%	41,2%
КСМ–119	КГ	10	0	5	2	3	0,0%	50,0%	20,0%	30,0%
Разом по КГ:		222	3	115	45	59	1,4%	51,8%	20,3%	26,6%
ГКС–02	ЕГ	35	0	12	14	9	0,0%	34,3%	40,0%	25,7%
ГКС–03	ЕГ	40	2	18	9	11	5,0%	45,0%	22,5%	27,5%
ГКС–04	ЕГ	48	0	19	6	23	0,0%	39,6%	12,5%	47,9%
І–02	ЕГ	20	0	8	6	6	0,0%	40,0%	30,0%	30,0%
КСМ–110	ЕГ	1	0	0	1	0	0,0%	0,0%	100,0%	0,0%
КСМ–111	ЕГ	7	0	1	3	3	0,0%	14,3%	42,9%	42,9%
КСМ–112	ЕГ	2	0	0	0	2	0,0%	0,0%	0,0%	100,0%
Разом по ЕГ:		153	2	58	39	54	1,3%	37,9%	25,5%	35,3%
Разом:		**375**	**5**	**173**	**84**	**113**	**1,3%**	**46,1%**	**22,4%**	**30,1%**

В табл. 4.5 показані узагальнені дані успішності та якості знань з державного екзамену за освітньо-кваліфікаційним рівнем бакалавра.

Таблиця 4.5

Результати державного екзамену
в контрольних та експериментальних групах

Групи	Успішність	Якість	Середній бал
Контрольні	98,6%	46,8%	3,72
Експериментальні	98,7%	60,8%	3,95

Для розрахунку кутового перетворення Фішера враховується наступне: в контрольних групах ефект «якість навчання» спостерігався у 104 студентів, не спостерігався – у 118; в експериментальних групах ефект «якість навчання» спостерігався у 93 студентів, не спостерігався – у 60 (табл. 4.6).

Таблиця 4.6

Розподіл студентів в контрольних та експериментальних групах
за спостережуваним ефектом

Групи	Якість навчання спостерігається	Якість навчання не спостерігається	Разом
Експериментальні	93 (60,8%)	60 (39,2%)	153 (100%)
Контрольні	104 (46,8%)	118 (53,2%)	222 (100%)
Разом:	197	178	375

Експериментальні дані повністю задовольняють обмеження, що накладаються кутовим перетворенням Фішера:

а) жодна з часток, що порівнюються, не дорівнює нулю;

б) кількість спостережень у обох вибірках більше 5, що дозволяє будь-які співставлення.

Сформулюємо гіпотези:

H_0: Частка студентів, які склали державний екзамен на «відмінно» або «добре», у експериментальних групах не більше, ніж у контрольних.

H_1: Частка студентів, які склали державний екзамен на «відмінно» або «добре», у експериментальних групах більше, ніж у контрольних.

За формулою φ=arcsin$\sqrt{P}$ (де P – відсоткова доля) обчислимо значення кутів для кожної з груп: φ_1(60,8%)=1,789; φ_2(46,8%)=1,507.

Розрахуємо емпіричне значення $\varphi^*_{емп}$ за формулою

$$\varphi^*_{емп}=(\varphi_1-\varphi_2)\sqrt{\frac{n_1 n_2}{n_1+n_2}},$$

де n_1=153 – кількість спостережень у експериментальних групах, n_2=222 – кількість спостережень у контрольних групах.

Критичні значення φ^* для рівнів статистичної значущості 0,01 та 0,05:

$$\varphi^*_{кр}=\begin{cases}1{,}64\,(p\le 0{,}05)\\ 2{,}31\,(p\le 0{,}01)\end{cases}$$

Результат розрахунку $\varphi^*_{емп}$=2,682>2,31 знаходяться у зоні значущості (рис. 4.3), що дає підстави для відхилення гіпотези H_0 і прийняття H_1, тобто вищий рівень результатів державного екзамену з інформатики є результатом запропонованої методики.

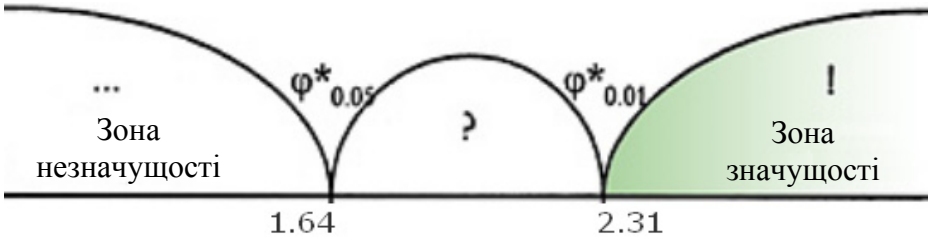


Рис. 4.3. Вісь значущості

Підвищення якості навчання – головна, проте не єдина ознака, за якою можна оцінювати пропоновану методичну систему фундаментальної інформатичної підготовки. Переконливим прикладом підвищення професійної мобільності студентів ще в процес навчання є регулярна результативна участь студентів спеціальності «Інформатика» педагогічного університету (КДПУ) у другому етапі Всеукраїнської студентської олімпіади з економічної кібернетики, що проводиться серед студентів економічних спеціальностей, та Всеукраїнської студентської олімпіади з системного програмування, що проводиться серед студентів технічних університетів (в додатку Д наведені приклади тестів, що

використовувалися для визначення рівня теоретичних знань студентів в процесі підготовки до олімпіади).

Інший показник – широке застосування випускниками мобільного програмного забезпечення, яке, на їхню думку, є засобом підвищення конкурентоспроможності на ринку праці.

Висновки до розділу 4

Аналіз результатів навчання студентів з метою з'ясування педагогічної ефективності розробленої методичної системи фундаментальної інформатичної підготовки, констатувального, пошукового та формувального етапів педагогічного експерименту дозволив зробити наступні висновки:

1. В сучасних умовах, у зв'язку з стрімким розвитком прикладної інформатики, майбутні вчителі інформатики та фахівці у галузі інформаційних технологій потребують фундаментальної інформатичної підготовки: на перше місце виступають саме загальнотеоретичні, фундаментальні та міждисциплінарні знання, а не технологічні, утилітарні знання та вміння із застосування нових інформаційних технологій.

2. Разом з тим у вищій інформатичній освіті спостерігаються певні негативні явища. Як показали наші дослідження і дослідження інших фахівців, сьогодні рівень фундаментальності інформатичних дисциплін досить низький, що призводить до швидкого застарівання набутих студентами в процесі навчання знань і умінь, зниження рівня фахової підготовки та ролі університетської інформатичної освіти у формуванні професійних інформатичних компетентностей студентів.

3. Для подолання зазначених негативних явищ в умовах інформаційного суспільства в основу методичних систем навчання інформатичних дисциплін повинні бути покладені принципи фундаментальності та мобільності. В процесі фундаменталізації змісту навчання інформатичних дисциплін пріоритетними стають методологічно важливі, інваріантні знання, що сприяють цілісному сприйняттю наукової картини світу, інтелектуальному розвитку особистості та

її адаптації у швидко мінливих соціально-економічних та інших умовах. При цьому технологічною основою фундаменталізації навчання інформатичних дисциплін виступають навчальна, технологічна та професійна мобільність.

4. Проведений педагогічний експеримент підтвердив гіпотезу про те, що фундаменталізація навчання інформатичних дисциплін сприяє підвищенню основних показників ефективності навчання студентів, зростанню рівня фундаментальності знань, розвитку узагальнених вмінь і навичок щодо використання мобільного програмного забезпечення в навчальній та виробничій діяльності, підвищенню конкурентоспроможності випускників інформатичних спеціальностей ВНЗ на ринку праці.

ВИСНОВКИ

У відповідності до поставленої мети та завдань дисертаційного дослідження в ході вивчення наукової проблеми і впровадження розроблених компонентів методичної системи навчання інформатичних дисциплін отримані такі основні **результати**:

– з'ясовано стан теоретичної розробленості проблеми в науковій літературі та її практичної реалізації в системі підготовки майбутніх фахівців у галузі інформаційних технологій у вищих навчальних закладах III–IV рівнів акредитації;

– розроблені теоретичні засади фундаменталізації навчання інформатичних дисциплін на основі концепції мобільності; визначені організаційно-педагогічні, програмно-технічні і технологічні умови реалізації мобільного навчання;

– розроблена методична система фундаментальної інформатичної підготовки;

– експериментально перевірена результативність розробленої методичної системи фундаментальної інформатичної підготовки у практиці навчання студентів вищих навчальних закладів III–IV рівнів акредитації різного профілю.

Результати проведеного дослідження теоретичних, технологічних та методичних основ фундаменталізації навчання інформатичних дисциплін у ВНЗ III–IV рівнів акредитації дають підстави зробити такі висновки:

1. Концепція фундаментальності для вищої освіти є системоутворюючою, тому процес її фундаменталізації є як поверненням до витоків сучасної університетської освіти, так і рухом до інтеграції у загальноєвропейський освітній простір.

2. Досягнення цілей фундаменталізації інформатичної освіти можливе через організовану цілеспрямовану педагогічну діяльність учасників освітнього процесу, що забезпечує реалізацію методологічної, професійно-орієнтувальної, розвивальної, прогностичної та інтегративної функцій фундаменталізації освіти:

– опанування методологічно важливими та інваріантними знаннями з довготривалим терміном життя, необхідними для професійної діяльності фахівця в галузі інформаційних технологій;

– тісний зв'язок інформатичної освіти з професійною практичною діяльністю;

– розвиток творчої і пізнавальної активності та самостійності студентів;

– розвиток методичних систем навчання інформатичних дисциплін з урахуванням перспектив розвитку «економіки знань» та інформаційного суспільства;

– системність засвоєння інформатичних дисциплін студентами на основі глибокого розуміння сучасних проблем інформатики.

3. Фундаменталізація інформатичної освіти впливає на всі компоненти методичної системи навчання інформатичних дисциплін: цілі, зміст, методи, засоби, форми організації навчання. Це визначає два основних напрями модифікації методичної системи навчання інформатичних дисциплін. Перший – фундаменталізація змісту навчання: надання йому властивостей стійкості, стабільності, збережуваності, тривалості. Другий – підвищення мобільності через надання: навчанню властивості контекстності (чутливості до часу та місця); суб'єкту навчання більшої кількості «ступенів вільності» (вищої інтерактивності, більшої свободи руху, більшої кількості технічних засобів); засобам навчання властивостей відкритих систем (розширюваності, масштабованості, мобільності та «люб'язності»).

4. Фундаменталізація змісту навчальної дисципліни надає можливість визначити стійке (інваріантне) ядро змісту, а фундаментальність може бути досягнута, якщо в змісті навчання чітко визначені фундаментальні основи навчального предмета, які відповідають фундаментальним основам предметної галузі. Компетентнісний підхід до навчання інформатичних дисциплін є одним із засобів їх фундаменталізації: ключовим у концепції фундаменталізації є принцип наскрізної інтеграції навчальних дисциплін на основі формування інформатичних компетентностей.

Ознакою інтегративності навчальних дисциплін є наступність у розгортанні змісту й структури навчальних дисциплін на основі фундаментальних концепцій інформатики. Інтегративність інформатичних дисциплін визначається фундаментальністю самої науки інформатики та інтегративним характером основних об'єктів її вивчення. При цьому найбільш ефективним засобом інтеграції інформатичних дисциплін у педагогічних ВНЗ у педагогічних ВНЗ є моделювання, яке, крім того, є основою фундаменталізації підготовки майбутніх вчителів інформатики.

Перехід до нового покоління галузевих стандартів вищої освіти на основі фундаменталізації навчання та компетентнісного підходу є необхідним етапом на шляху реформування системи освіти в Україні, а застосування компетентнісного підходу до розробки галузевих стандартів вищої освіти створює умови для зближення фундаментальної освіти до потреб та вимог ринку праці, подальшого розвитку освітніх технологій та системи освіти в цілому.

5. На сучасному етапі розвитку засобів ІКТ технологічною основою фундаменталізації вищої освіти стає електронне навчання – інноваційна технологія, спрямована на професіоналізацію та підвищення мобільності тих, хто навчається. Удосконалення апаратних характеристик перетворило мобільні пристрої на потужні інтерактивні мультимедійні технічні засоби мобільного навчання, яке є сучасним напрямом розвитку дистанційного навчання із застосуванням мобільних телефонів, смартфонів, КПК та електронних книжок.

Мобільне навчання – це специфічний вид навчання, в якому сам навчальний процес є географічно та ситуаційно залежним. В порівнянні з традиційним у мобільному навчанні забезпечується можливість моніторингу навчання в реальному часі та висока насиченість контенту, що надає можливість розглядати його не лише як засіб навчання, а й як інструмент спільної роботи, спрямованої на підвищення якості навчання.

До визначальних характеристик мобільного навчання відносяться:

– можливість динамічного генерування навчального матеріалу в залежності від місцезнаходження студента, типу мобільного пристрою та способу його

застосування;

– розмиття границь між соціумом та навчальним закладом завдяки можливості застосування мобільних пристроїв у навчанні, коли викладач опиняється в умовах, за яких матеріалу, що раніше циркулював у межах аудиторії, може бути протиставлений матеріал ззовні, що функціонує без контролю з його боку.

Впровадження елементів мобільного навчання в навчальний процес середньої та вищої школи надасть можливість уникнути негативних наслідків неконтрольованого використання мобільних пристроїв через їх активне використання в процесі навчання замість адміністративних заборон. Використання технологій мобільного навчання паралельно з традиційними навчальними технологіями сприятиме забезпеченню якості освіти, підвищуючи гнучкість процесу навчання та задовольняючи вимоги безперервної освіти та навчання протягом усього життя. Мобільне навчання може також забезпечити поліпшення можливостей отримання освіти для осіб з особливими потребами, пропонуючи їм більшу гнучкість, вибір часу і місця навчання через доставляння контенту на їхні мобільні пристрої у відповідності до їхніх потреб.

6. Фундаменталізація інформатичної освіти вимагає посилення ролі обчислювального експерименту та програмування:

– обчислювальний експеримент є методологією інформатики як науки, тому його можна віднести до принципів (методології) наукових методів учіння;

– цілі навчання інформатики у вищій школі включають необхідність засвоєння як певної сукупності наукових фактів, так і методів отримання цих фактів, які використовуються в самій науці, а програмування відображає метод пізнання, що застосовується в інформатиці. При цьому під терміном «програмування» розуміється діяльність людини, яка у вузькому сенсі зводиться до простого кодування відомого алгоритму, а в широкому – співпадає з методологією інформатики, тобто є тотожною обчислювальному експерименту.

7. До інноваційних методів навчання інформатичних дисциплін відноситься парне програмування – форма розробки програмного забезпечення, за якої програма для розв’язування поставленої задачі створюється парою програ-

містів, котрі працюють за одним робочим місцем. У парному програмуванні основна взаємодія відбувається між двома студентами, котрі можуть обговорювати поставлену задачу і свої дії, здійснювати взаємонавчання або взаємоконтроль. Даний метод є також і формою організації навчальної діяльності, за якої студенти-програмісти показують більш, ніж у двічі більшу продуктивність, в порівнянні з тим, коли вони працюють поодинці. За дистанційної форми навчання парне програмування реалізується через віддалене парне програмування – спосіб реалізації парного програмування, при якому обидва розробники, що складають пару, фізично знаходяться у різних місцях, і працюють за допомогою партнерського редактора реального часу, спільної розподіленої стільниці або спеціального модуля IDE для віддаленого парного програмування.

8. Стабілізація ядра змісту та засобів навчання інформатики через інваріантність відносно операційної системи та мови програмування сприяє підвищенню рівня теоретичної підготовки, реалізує міжпредметну інтеграцію, відкриває широкі можливості добору апаратних та програмних засобів навчання інформатичних дисциплін, знижуючи їх вартість за рахунок використання ліцензійно чистого, вільно поширюваного, локалізованого програмного забезпечення.

Стабілізація програмних засобів надає можливості для варіювання програмного забезпечення навчання інформатичних дисциплін замість штучної прив’язки до окремих програмних продуктів. До стабільного програмного забезпечення навчання інформатичних дисциплін у вищій школі відносяться мобільні операційні системи, мобільні компілятори, мобільні інтерпретовані мови програмування, відкриті системи комп’ютерної математики, спеціалізовані предметні середовища та Web-середовища:

1) один із загальноприйнятих способів підвищення мобільності програмного забезпечення – стандартизація програмного оточення: програмних інтерфейсів, утиліт тощо. На рівні системних сервісів подібне оточення описується в стандарті POSIX, підтримка якого полегшує перенесення прикладних програм практично на будь-яку скільки-небудь поширену операційну платформу. Мобі-

льність програм, що відповідають стандарту POSIX, принципово досяжна завдяки наявності великої кількості стандартизованих системних сервісів та можливості динамічного з'ясування характеристик цільової платформи й налаштовування програми під них. POSIX-сумісність є засобом уніфікації операційних систем, а дотримання стандартів POSIX при розробці програмного забезпечення – засобом уникнення залежності від використовуваної операційної системи;

2) застосування мобільних компіляторів є засобом уникнення залежності від використовуваних середовища програмування (через потужний інтерфейс командного рядка та легкість інтеграції у IDE), операційної системи (через забезпечення POSIX-сумісності) та мови програмування (через надання спільних бібліотек);

3) застосування мобільних інтерпретованих мов загального призначення є засобом забезпечення мобільності програм, створених на POSIX-несумісних платформах;

4) відкриті вільно поширювані мобільні системи комп'ютерної математики відзначаються тривалою історією розвитку, оптимізованими алгоритмами, POSIX-сумісністю, невимогливістю до ресурсів, ліцензійною чистотою, безоплатністю, різноманітністю інтерфейсів, локалізованістю та іншими перевагами, що дозволяють застосовувати їх в якості стабільного програмного забезпечення математичного призначення;

5) добір спеціалізованих предметних середовищ навчання інформатичних дисциплін виконується з відкритих мобільних програмних систем навчального призначення, що мають широку інсталяційну базу та придатні для локалізації;

6) використання мобільних пристроїв з невисокою швидкодією та малим обсягом оперативної пам'яті суттєво ускладнює застосування таких ресурсоємних програм, як середовища програмування, системи комп'ютерної математики і т.п. Для розв'язання цієї проблеми доцільно перейти до мережецентричної моделі, за якої ресурсоємні програми працюють на Інтернет-серверах, а головним клієнтським додатком виступає Web-браузер. Перенесення прикладного програмного забезпечення у Web-середовище (онлайн-IDE, Web-СКМ ті ін.)

створює нові можливості для обміну навчальними матеріалами і співробітництва між усіма учасниками навчального процесу:

– для будь-якого користувача за рахунок цього надається можливість мобільного доступу до програм та даних;

– для адміністратора комп'ютерного класу знімаються проблеми підтримки великої інсталяційної бази та ліцензування програмного забезпечення;

– для викладачів суттєво розширюється спектр використовуваного програмного забезпечення, а для студентів – використовуваних засобів мобільного навчання.

Таким чином, фундаменталізація навчання інформатичних дисциплін у вищій школі сприяє підвищенню рівня теоретичної підготовки та формуванню професійних інформатичних компетентностей студентів; реалізації міжпредметної інтеграції та застосуванню методів суміжних наук; надає широкі можливості вибору апаратних та програмних засобів навчання; надає можливість створювати стабільні підручники з інформатичних дисциплін.

Сукупність результатів, отриманих у дисертаційному дослідженні, в опублікованих дисертантом роботах, дозволяє кваліфікувати виконану роботу як теоретичне узагальнення здобутків науково-методичних досліджень, які проводились як в Україні, так і за її межами, власних наукових напрацювань дисертанта, досвіду роботи вищих навчальних закладів із підготовки фахівців у галузі інформаційних технологій. Пропоноване дослідження є певним внеском у розв'язання актуальної проблеми в галузі методики навчання інформатики у вищій школі та відкриває новий напрям у розробці методичних систем навчання інформатичних дисциплін, що надасть можливість суттєво підняти рівень підготовки фахівців у галузі інформатики та інформаційних технологій.

Отримані результати надають можливість вказати деякі напрями подальших досліджень:

1) дослідження перспективних напрямів розвитку мобільного навчання та використання його технологій у вищій школі;

2) розвиток концепції мережецентричних обчислень у навчальній і науково-дослідній діяльності студентів;

3) розширення можливостей Web-середовища SAGE в напрямі підтримки навчальних досліджень у природничих науках;

4) фундаменталізація шкільного курсу інформатики.

Над цими проблемами під керівництвом дисертанта працює творчий колектив із студентів, аспірантів, здобувачів та викладачів КДПУ, КЕІ КНЕУ, НМетАУ.

ДОДАТКИ

Додаток А

Стандарти технологій мобільного навчання

ЄС визначає стандартизацію як ключовий стратегічний напрям і рекомендує розширювати зв'язки розробників засобів навчання з офіційними організаціями. Дотримуючись цих рекомендацій і враховуючи важливу роль, яку відіграє у справі стандартизації зростання ринку освітніх послуг, проект MOBIlearn приєднався до наступних міжнародних стандартів і специфікацій взаємодії: ISO/IEC JTC1/SC36, ADL SCORM, CEN/ISSS WSLT, IEEE LTSC, IMS Global Learning Consorcium, W3C, ITU, DCMI Education Working Group. Зокрема, наступні технології і стандарти були визначені в якості ключових: XML, DVB-MHP, 3GPP, SIP, ISO13407.

ISO/IEC JTC1/SC36

Призначення SC36 – стандартизація у галузі інформаційних технологій для навчання, освіти і професійної підготовки, для підтримки окремих осіб, груп чи організацій і надання можливості сумісності і повторного використання ресурсів та інструментів. Найбільш цікаві задачі проекту розробляються наступними робочими групами:

SC36/WG2 Технологічна співпраця;

SC36/WG3 Навчальні матеріали;

SC36/WG4 Управління та надання послуг з навчання, освіти і професійної підготовки.

ADL SCORM

Призначення ADL (Advanced Distributed Learning) – забезпечити доступ до якісних навчальних матеріалів, які можуть бути пристосовані до індивідуальних потреб учня. В перспективі ADL базуватиметься на технологіях динамічного навчання, в яких контент збирається та доставляється учням з урахуванням їхніх особистих темпів і потреб. В процесі роботи над ADL створено

об’єктну модель розподіленого контенту SCORM, в якій об’єднується багато нових стандартів в єдину контентну модель.

CEN/ISSS WSLT

CEN (Європейський комітет зі стандартизації), CENELEC (Європейський комітет з електротехнічної стандартизації) та ETSI (Європейський інститут телекомунікаційних стандартів) були запрошені для виявлення потреб в європейських ініціативах зі стандартизації в галузі технологій навчання та освітніх мультимедійних програм. Ними були розроблені рекомендації зі спільного навчання.

IEEE LTSC

Метою робочих груп IEEE LTSC є розробка технічних стандартів, рекомендацій та посібників для компонентів програмного забезпечення, інструментів, технологій і методів розробки, на основі яких полегшується розробка, розгортання, підтримка і взаємообмін комп’ютерних реалізацій навчальних компонентів і систем. Особливо цікаві напрями:

– метадані об’єктів навчання (IEEE P1484.12);

– визначення компетентностей (IEEE P1484.20);

– семантика та обмін прив’язками (IEEE P1484.14);

– локалізація протоколів обміну даними (IEEE P1484.15);

– архітектура та еталонна модель (IEEE P1484.1).

IMS Global Learning Consortium

IMS Global Learning Consortium займається розробкою та заохоченням відкритих специфікацій для спрощення онлайнової розподіленої навчальної діяльності, такої як знаходження і використання освітнього контенту, відстеження ходу навчання та звітування про його продуктивність, а також обмін записами про студентів між адміністративними системами.

IMS має дві основні цілі: 1) визначення технічних вимог до сумісності додатків і служб у розподіленому навчанні та 2) підтримки включення специфікацій IMS у продукти та послуги. IMS прагне заохочувати широке впроваджен-

ня специфікацій розподілених середовищ навчання, за допомогою яких кілька авторів можуть працювати разом.

World Wide Web Consortium (W3C)

W3C контролює технічний розвиток Інтернет. За сім років W3C розробив понад сорок технічних специфікацій для Web-інфраструктури. Однак, Web досі є молодою технологією, тому є ще багато роботи, яку належить зробити, особливо в об'єднанні комп'ютерів, телекомунікацій та мультимедійних технологій. Для того, щоб задовольнити зростаючі очікування користувачів і використати збільшення потужності машини, W3C вже заклав основи для наступного покоління Інтернету. Використання W3C-технологій дозволить зробити Web надійною, масштабованою та адаптивною інфраструктурою для інформаційного суспільства.

Міжнародна телекомунікаційна спілка (ITU)

Її цілями є:

1) підтримка і розширення міжнародного співробітництва між усіма її країнами-членами з метою вдосконалення і раціонального використання всіх видів електрозв'язку;

2) заохочення та активізація участі підрозділів і організацій у діяльності спілки, а також сприяння плідній співпраці і партнерству між ними та країнами-членами для досягнення спільних цілей;

3) заохочення та надання технічної допомоги в галузі телекомунікацій країнам, що розвиваються, а також сприяння мобілізації матеріальних, людських і фінансових ресурсів, необхідних для покращення доступу до телекомунікаційних послуг в таких країнах;

4) сприяння розвитку технічних засобів та їх найбільш ефективній експлуатації з метою підвищення ефективності телекомунікаційних послуг, підвищення їх корисності, щоб зробити їх, наскільки це можливо, широко доступними для громадськості;

5) заохочення розширення переваг нових телекомунікаційних технологій для всіх жителів світу;

6) сприяння використанню телекомунікаційних послуг для підтримки мирних відносин;

7) гармонізація дій країн-членів ITU для подальшої плідної і конструктивної співпраці і партнерства між ними;

8) сприяння на міжнародному рівні більш широкому підходу до проблем електрозв'язку в глобальній інформаційній економіці та суспільстві шляхом співробітництва з іншими світовими та регіональними міжурядовими організаціями та неурядовими організаціями, що займаються питаннями телекомунікацій.

DCMI Education Working Group

Метою робочої групи Dublin Core Metadata Initiative (DCMI) є розробка пропозицій з використання метаданих дублінського ядра для опису освітніх ресурсів, що можуть використовуватись в дошкільній, шкільній, подовженій і вищій освіті, професійно-технічній підготовці, навчанні протягом усього життя. Робоча група продовжує свою роботу з розробки класифікаторів для DCMES та доменних елементів, кваліфікаторів елементів та значень для опису навчальних матеріалів.

XML

XML використовується в якості стандартного галузевого формату. Він буде використовуватися також для навчальних об'єктів. Використання XML надає можливість відділити семантику змісту від його подання. За допомогою XML навчальні об'єкти можуть бути використані в бібліотеках компонентів для створення навчальних продуктів.

DVB-MHP

MHP (Multimedia Home Platform) являє собою загальний API, що є повністю незалежним від апаратної платформи, на якій він функціонує. MHP – це відкрита стандартна платформа. Online-віщання та Інтернет-контент від різних провайдерів можна отримати за допомогою одного пристрою, що використовує цей загальний DVB-MHP API. В MHP визначається загальний інтерфейс між прикладними програмами і терміналами, на яких ці програми виконуються. Ви-

користання MHP (DVB-MHP TS101-812) розширює вже існуючі відкриті DVB-стандарти мовлення та послуг у всіх передавальних мережах, включаючи супутникові, кабельні, наземні та мікрохвильові.

3GPP

В мобільному мультимедійному доступі будуть використовуватися стандарти, рекомендовані 3GPP. 3GPP з самого початку була створена для виробництва глобально застосовних технічних специфікацій і технічних звітів для мобільних систем 3-го покоління. Для забезпечення сумісності при передаванні мультимедійної інформації будуть використовуватися стандартні кодеки. Ці стандарти рекомендовані робочою групою №4 3GPP SA, яка займається специфікаціями кодеків для передавання мовлення, аудіо-, відео- та мультимедіа.

SIP

Session Initiation Protocol (SIP) виник як протокол вибору для встановлення конференційних, телефонних, мультимедійних та інших типів сеансів зв'язку через Інтернет. SIP також може бути використаний для нових видів зв'язку, таких як миттєвий обмін повідомленнями і мобільними програмами в різних мережах, включаючи безпровідні.

ISO13407: Human centred design processes for interactive systems

Цей стандарт є інструментом для управління розробкою процесів і містить вказівки на джерела даних і стандарти, що відносяться до людиноцентричного підходу, як міждисциплінарної діяльності, в якій враховуються ергономічні знання і технології з метою підвищення дієвості та ефективності, поліпшення умов праці людини, та боротьби з можливими негативними впливами на здоров'я людини, її безпеку та продуктивність.

Додаток Б

Методичні основи вивчення теми «Операційні системи» у підготовці майбутнього вчителя

Стрімкий розвиток інформатики як науки, вдосконалення засобів обчислювальної техніки та програмного забезпечення призводить до швидкого застарівання навичок роботи з конкретними програмними продуктами. Відомий вчений у галузі програмування Дональд Кнут, відмічаючи цю тенденцію, підкреслює необхідність навчання концептуальних основ інформатики, її математичної та алгоритмічної складових, що формують загальні прийоми та методи роботи майбутнього фахівця.

На жаль, значна частина навчальної літератури з інформатики не відповідає цим вимогам, пропонуючи варіант вивчення курсу інформатики без ознайомлення з її теоретичними основами та переносячи центр уваги на заучування інтерфейсу окремих прикладних програм. Це призводить до такого негативного явища, як заміна вивчення інваріантної частини курсу інформатики варіативною, а, отже, штучного «розмиття» вимог до знань та вмінь з предмету.

Операційна система (ОС) є першим, з чим стикається будь-який учень на початку роботи з комп'ютером, проте означення операційної системи в курсі інформатики вводиться не одразу. Так, М.І. Жалдак та Н.В. Морзе у підручнику «Інформатика–7» спочатку вводять поняття інформаційної (обчислювальної) системи як системи пристроїв і описів правил управління ними, а сукупність описів правил управління пристроями інформаційної системи називають її програмною частиною або програмним забезпеченням [86, 24].

В процесі вивчення курсу введені поняття збагачуються, наповнюючись досвідом практичної роботи, тому виникає необхідність формалізації поняття операційної системи, як це було зроблено, наприклад, у навчальному посібнику М.І. Жалдака та Ю.С. Рамського [87].

На жаль, у багатьох студентів поняття операційної системи залишається досить розмитим, і цьому є об'єктивні причини. По-перше, операційні системи

розроблені для великої кількості різноманітних комп'ютерних систем: однокористувацьких робочих станцій та персональних комп'ютерів, багатокористувацьких систем середнього масштабу, універсальних обчислювальних машин та суперкомп'ютерів, систем реального часу тощо. По-друге, неперервно продовжується інтенсивний розвиток усіх напрямів досліджень, що відносяться до комп'ютерів: в будові операційних систем з'явилися нові важливі компоненти, технології та розробки [356]. Незважаючи на таку різноманітність систем та постійні зміни, деякі фундаментальні положення залишаються незмінними, і саме на них варто зосередити увагу студентів. Для цього пропонуємо вивчення теми «Операційні системи» за таким планом:

1. **Поняття операційної системи. Багатошарова архітектура комп'ютерної системи**

В цьому питанні доцільно розглянути *операційну систему* як розподілювач ресурсів, сукупність програм, призначених для управління апаратними, програмними і інформаційними ресурсами інформаційної системи, засіб підвищення ефективності інформаційної системи та полегшення праці програміста.

При розгляді багатошарової архітектури комп'ютерної системи використовуємо запропоновану в [356] схему розташування шарів:

прикладне програмне забезпечення
компілятори, редактори, командні інтерпретатори (системне програмне забезпечення)
операційна система
машинні мови
мікропрограми
фізичні пристрої

2. **Етапи розвитку операційних систем**

На кожному етапі виділятимемо його характерні ознаки:

Ранні системи (1945–1955) – відсутність операційних систем та захисту програм.

Друге покоління систем (1956–1965) – системи пакетного опрацювання даних, канали введення-виведення, буферизація, переривання, бібліотеки, мови управління завданнями.

Третє покоління систем (1966–1980) – багатозадачність, розподіл часу, віртуальна пам’ять, багатопроцесорність.

Четверте покоління та подальший розвиток операційних систем (з 1981) – масивно-паралельні системи, комп’ютерні мережі, розподілені обчислення.

3. Концепції операційної системи

Кожна з концепцій операційної системи (а це, насамперед, програми, процеси, потоки, файли, системні виклики, оболонки, ядро, пам’ять, управління пам’яттю) може бути предметом детального розгляду, проте нашою задачею є означення їх суттєвих властивостей.

Під *програмою* розумітимемо:

– набір інструкцій та даних, що зберігаються;
– файл, що має атрибут «виконуваний»;
– файл, що містить впорядковану послідовність інструкцій ядра;
– файл з текстом програми мовою програмування;
– об’єктний файл (текст програми, відтрансльований в машинні коди);
– виконуваний файл, що містить об’єктний код з приєднаними бібліотеками.

Процес розглядатимемо як перебіг виконання програми в середовищі її виконання, що може бути призупинений ОС та складається з трьох компонентів: коду програми, даних користувача та системних даних (зокрема, тих, що відсутні у програмі).

Потік означимо як послідовність інструкцій, що виконуються всередині процесу.

Концепцію *файлу*, якою студенти вже володіють, доцільно розширити введенням поняття дескриптора файлу як унікального цілого числа, що однозначно ідентифікує файл в операціях введення-виведення та запропонувати серію прикладів застосування дескрипторів файлів для операцій із регулярними файлами, каталогами, каналами, символьними та блочними пристроями тощо.

Системні виклики розглядатимемо як сукупність розширених інструкцій операційної системи, що забезпечують інтерфейс (взаємозв'язки та узгодження функціонування) між програмами та ядром системи.

При розгляді *оболонок (інтерпретаторів команд)* слід звернути увагу на те, що вони є звичайними програмами, які можуть бути замінені іншими, тому до оболонок формулюються такі вимоги:

– підтримка запрошення до роботи;
– перенаправлення введення-виведення;
– можливість виконання фонових задач;
– підтримка щонайменше трьох типів команд: виконуваних файлів, сценаріїв та вбудованих команд оболонки.

Найбільш містким з розглядуваних є концепція *ядра* – коду, який постійно знаходиться в основній пам'яті і за допомогою якого

– контролюється перебіг процесів шляхом їх створення, завершення, призупинення та обміну даними;
– планується виконання процесів на центральному процесорі;
– розподіляється пам'ять для виконуваних процесів;
– обслуговується файлова система;
– надається процесам контрольований доступ до периферійних пристроїв;
– прозоро надається ряд системних послуг, що узагальнює концепцію файлу.

Пам'ять пропонується характеризувати розміром, швидкістю доступу та вартістю. Це дозволяє впорядкувати різні типи пам'яті у порядку зменшення вартості:

Тип	*Розмір*	*Швидкість доступу*
Регістрова пам'ять	Байти	Майже зі швидкістю процесора
Кеш-пам'ять	Кілобайти	Наносекунди
Оперативна пам'ять	Мегабайти	Мікросекунди
Магнітні диски	Гігабайти	Мілісекунди
Магнітні стрічки /Оптичні диски	Без обмежень	–

Окремо розглядаються віртуальна та дискова кеш-пам'ять.

Управління пам'яттю є однією з головних послуг операційної системи, що забезпечує: ізоляцію процесів; автоматичне виділення та контроль пам'яті; підтримку модульного програмування; контроль захисту та доступу; довготермінове зберігання.

4. Структура операційної системи

Для кожного типу систем виділятимемо їх характерні ознаки.

Мінімальні операційні системи – найбільш прості операційні системи, що являють собою безструктурні набори незахищених підпрограм, які важко адаптувати до мультипрограмного та багатокористувацького середовища.

Багатошарові системи являють собою ієрархію шарів, що розташовані один над одним, або серію концентричних кіл, в яких рівень привілеїв зростає до центра.

Віртуальні машини, в яких для кожного процесу утворюється віртуальний центральний процесор з наданням мінімального набору операцій, що утворюють трирівневу ієрархію:

– найнижчий рівень – віртуалізація центрального процесора для всіх процесів;
– середній рівень – віртуалізація пам'яті для всіх процесів;
– вищий рівень – віртуальні пристрої введення-виведення.

Системи клієнт-сервер мають мінімальне ядро, що забезпечує узгоджене функціонування та обмін даними між клієнтом (користувацьким процесом) та сервером (обробником запитів).

5. Засоби введення-виведення

Розгляд цього пункту доцільно почати з найпростішого *програмованого введення-виведення*, при якому через центральний процесор безпосередньо здійснюється управління контролерами пристроїв. Далі переходимо до *введення-виведення, керованого перериваннями*, коли через центральний процесор здійснюється управління процесом введення-виведення за допомогою надсилання команд до відповідних модулів. Завершується пункт розглядом *каналів прямого доступу до пам'яті (DMA – Direct Memory Access)*, коли від централь-

ного процесора передається управління до модуля DMA для забезпечення контролю за закріпленим за ним блоком пам’яті.

6. Мультипрограмування. Мультипрограмні ОС та їх задачі. Операційні системи як віртуальні машини

В останньому пункті коротко характеризуємо *апаратні засоби підтримки мультипрограмування* (механізми пріоритетних переривань, захисту коду та даних, динамічний перерозподіл адрес) та *вимоги до мультипрограмних ОС*, зумовлені ними.

Наведена структура вивчення теми надає можливість наповнити поняття операційної системи змістом на основі вивчення загальних для усіх систем властивостей, що найбільш повно відповідає вимогам до фундаментальної підготовки майбутнього вчителя інформатики.

Додаток В

Web-СКМ SAGE як засіб навчання математичної інформатики

За визначенням Т.П. Кобильника, математична інформатика – напрям наукових досліджень, що знаходиться на межі математики та інформатики і, з одного боку, є складовою теоретичної інформатики, де математичні моделі і засоби використовуються для моделювання та дослідження інформаційних процесів у різних сферах діяльності людини, а, з іншого боку, займається використанням інформаційних систем і технологій для розв'язування складних математичних задач [133, 6]. Розглядаючи СКМ як основний засіб навчання математичної інформатики, він виділяє наступні класи задач, що можуть бути реалізовані за їх допомогою: 1) символьні перетворення, лінійна алгебра, аналітична геометрія, теорія графів, математичний аналіз, математична статистика; 2) моделювання інтелекту, математичне моделювання, теорія кодування, криптологія, розпізнавання образів. Для розв'язання цих задач пропонуються різні системи, що можуть бути об'єднані в єдине Web-середовище для досліджень в галузі математичної інформатики – SAGE [251; 257; 440; 443]. Розглянемо приклади реалізації у Web-СКМ SAGE динамічних моделей з різних розділів математики та інформатики, застосування яких надає можливість суттєво підвищити рівень наочності в процесі навчання математичних та інформатичних дисциплін.

1. Чисельний аналіз

Знаходження коренів методом дихотомії

```
def bisect_method(f, a, b, eps):
    try:
        f = f._fast_float_(f.variables()[0])
    except AttributeError:
        pass
    intervals = [(a,b)]
    two = float(2); eps = float(eps)
    while True:
        c = (a+b)/two
        fa = f(a); fb = f(b); fc = f(c)
        if abs(fc) < eps: return c, intervals
        if fa*fc < 0:
            a, b = a, c
        elif fc*fb < 0:
```

```
            a, b = c, b
        else:
            raise ValueError, "f має змінювати знак на проміжку
(%s,%s)"%(a,b)
        intervals.append((a,b))
@interact
def _(f = cos(x) - x, a = float(0), b = float(1), eps=(-3,(-16..-1))):
     eps = 10^eps
     print "точність = %s"%float(eps)
     try:
         time c, intervals = bisect_method(f, a, b, eps)
     except ValueError:
         print "f повинна мати різні знаки на кінцях інтервалу"
         show(plot(f, a, b, color='red'), xmin=a, xmax=b)
     else:
         print "корінь =", c
         print "f(c) = %r"%f(c)
         print "кількість ітерацій =", len(intervals)
         P = plot(f, a, b, color='red')
         h = (P.ymax() - P.ymin())/ (1.5*len(intervals))
         L = sum(line([(c,h*i), (d,h*i)]) for i, (c,d) in
enumerate(intervals) )
         L += sum(line([(c,h*i-h/4), (c,h*i+h/4)]) for i, (c,d) in
enumerate(intervals) )
         L += sum(line([(d,h*i-h/4), (d,h*i+h/4)]) for i, (c,d) in
enumerate(intervals) )
         show(P + L, xmin=a, xmax=b)
```

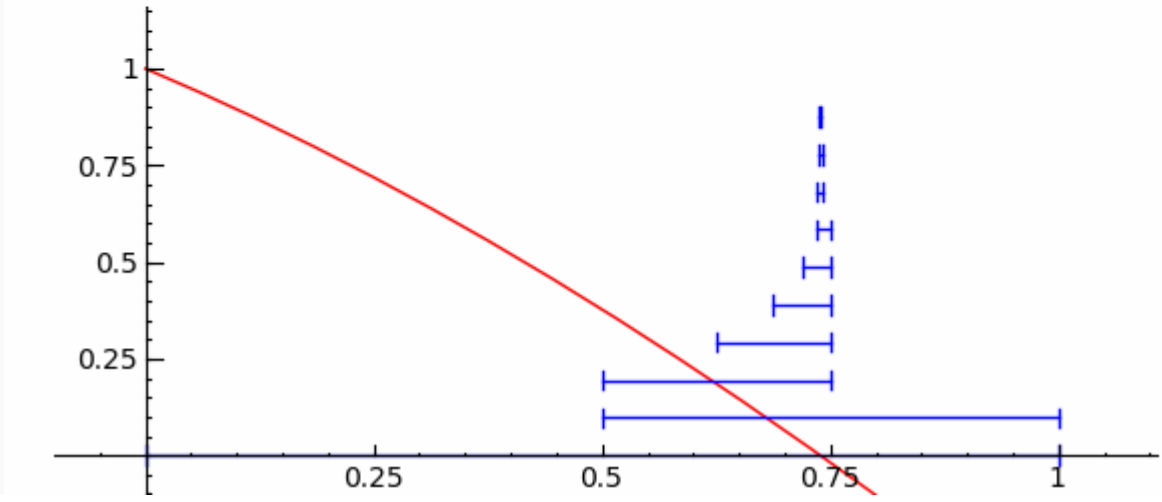
f -x + cos(x)

a 0.0

b 1.0

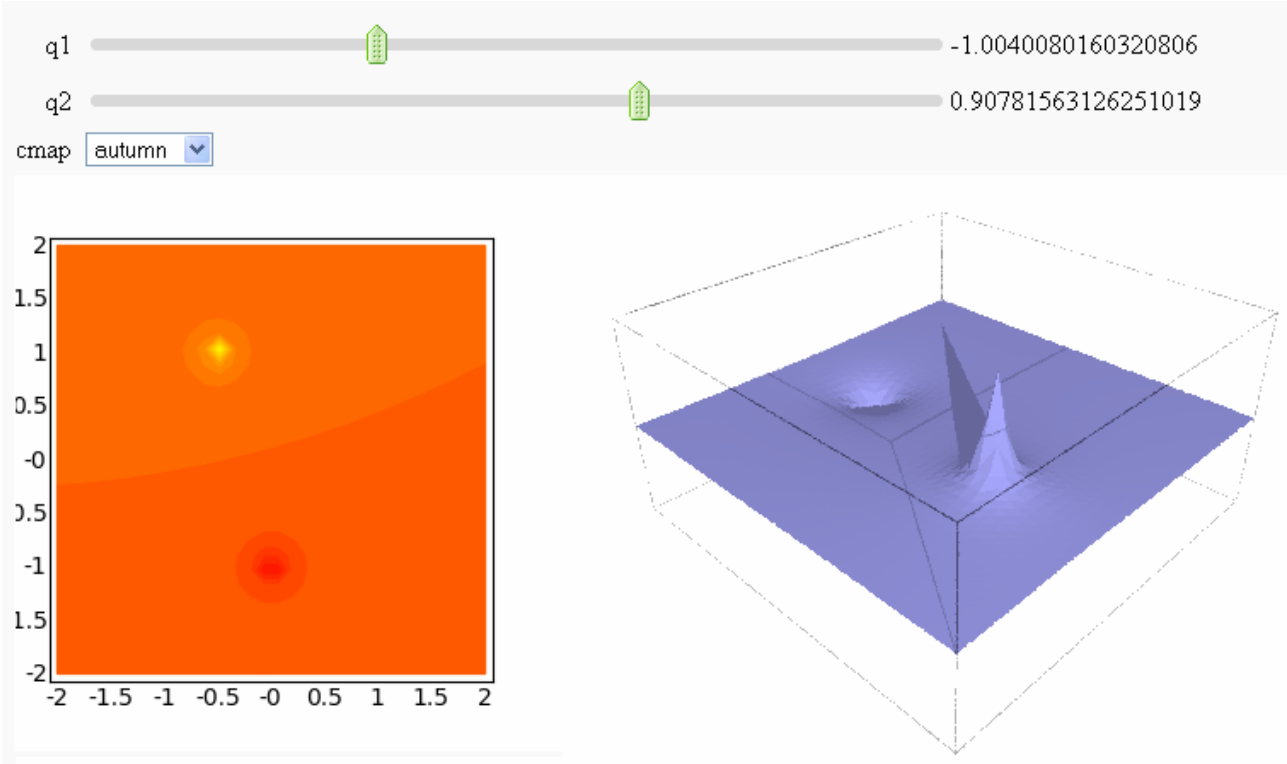
eps -3

```
точність = 0.001
Time: CPU 0.00 s, Wall: 0.00 s
корінь = 0.7392578125
f(c) = cos(0.739257812500000) - 0.739257812500000
кількість ітерацій = 10
```



Контурна карта та 3D-графік двох взаємно обернених функцій

```
@interact
def _(q1=(-1,(-3,3)), q2=(-2,(-3,3)),
      cmap=['autumn', 'bone', 'cool', 'copper', 'gray', 'hot', 'hsv',
            'jet', 'pink', 'prism', 'spring', 'summer', 'winter']):
     x,y = var('x,y')
     f = q1/sqrt((x+1)^2 + y^2) + q2/sqrt((x-1)^2+(y+0.5)^2)
     C = contour_plot(f, (-2,2), (-2,2), plot_points=30, contours=15,
cmap=cmap)
     show(C, figsize=3, aspect_ratio=1)
     show(plot3d(f, (x,-2,2), (y,-2,2)), figsize=5, viewer='tachyon')
```



Метод Ньютона

```
def newton_method(f, c, eps, maxiter=100):
    x = f.variables()[0]
    fprime = f.derivative(x)
    try:
        g = f._fast_float_(x)
        gprime = fprime._fast_float_(x)
    except AttributeError:
        g = f; gprime = fprime
    iterates = [c]
    for i in xrange(maxiter):
       fc = g(c)
       if abs(fc) < eps: return c, iterates
       c = c - fc/gprime(c)
       iterates.append(c)
    return c, iterates
html("<h1> Знаходження кореня за методом Ньютона з подвійною
точністю</h1>")
@interact
```

```
def _(f = x^2 - 2, c = float(0.5), eps=(-3,(-16..-1)),
interval=float(0.5)):
     eps = 10^(eps)
     print "точність = %s"%float(eps)
     time z, iterates = newton_method(f, c, eps)
     print "корінь = ", z
     print "f(c) = %r"%f(z)
     n = len(iterates)
     print "кількість ітерацій =", n
     html(iterates)
     P = plot(f, z-interval, z+interval, rgbcolor='blue')
     h = P.ymax(); j = P.ymin()
     L = sum(point((w,(n-1-float(i))/n*h),
rgbcolor=(float(i)/n,0.2,0.3), pointsize=10) + \
             line([(w,h),(w,j)],rgbcolor='black',thickness=0.2) for i,w
in enumerate(iterates))
     show(P + L, xmin=z-interval, xmax=z+interval)
```

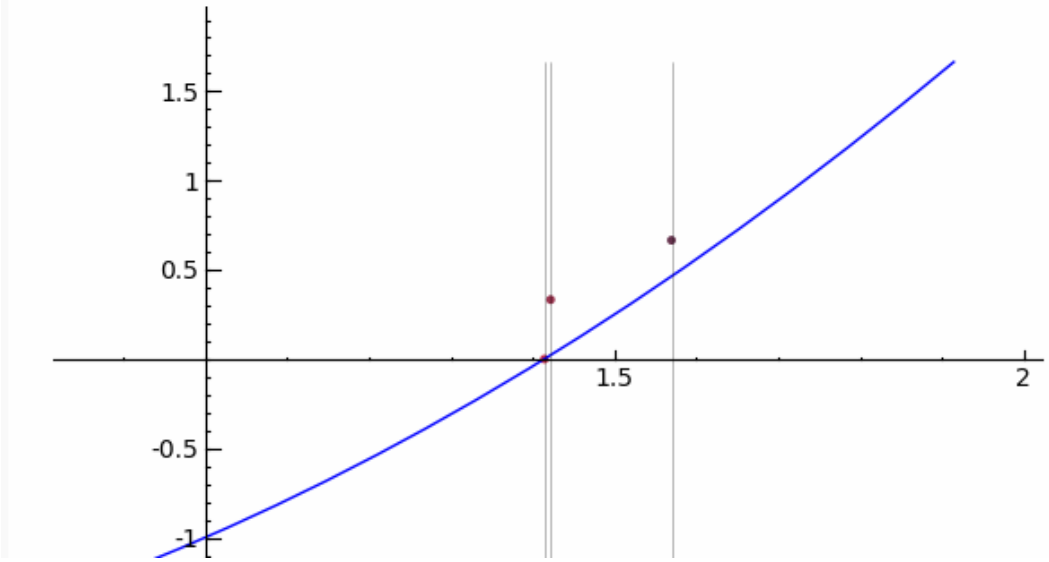
f x^2-2

c 0.5

eps -3

interval 0.5

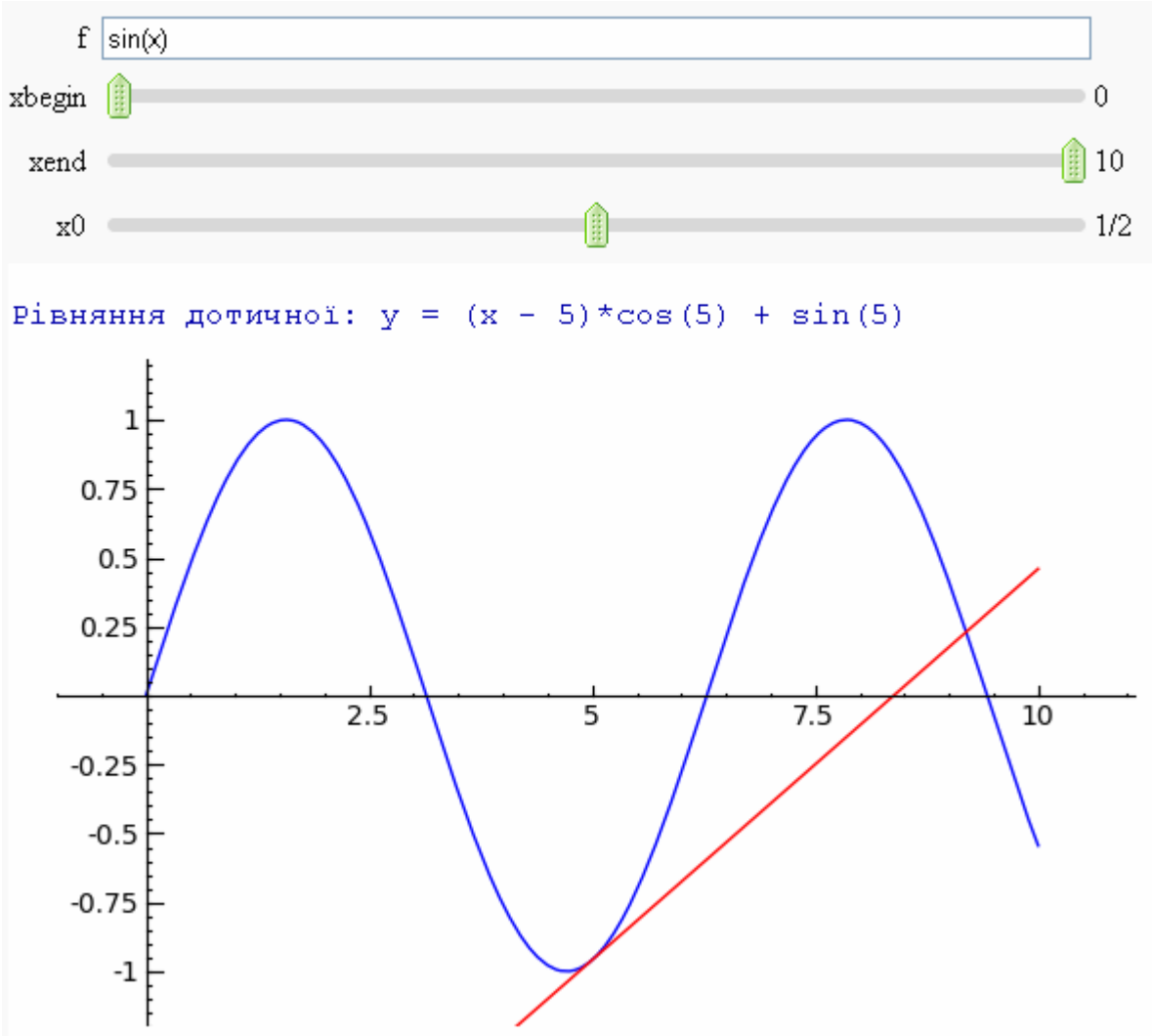
```
точність = 0.001
Time: CPU 0.01 s, Wall: 0.00 s
корінь = 1.41423428594
f(c) = 0.00005861552842922 89
кількість ітерацій = 5
[0.5, 2.25, 1.5694444444444444, 1.4218903638151426, 1.4142342859400734]
```



Побудова дотичної

```
@interact
def tangent_line(f = input_box(default=sin(x)), xbegin =
slider(0,10,1/10,0), xend = slider(0,10,1/10,10), x0 = slider(0, 1,
1/100, 1/2)):
    prange = [xbegin, xend]
    x0i = xbegin + x0*(xend-xbegin)
```

```
    var('x')
    df = diff(f)
    tanf = f(x0i) + df(x0i)*(x-x0i)
    fplot = plot(f, prange[0], prange[1])
    print 'Рівняння дотичної: y = ' + tanf._repr_()
    tanplot = plot(tanf, prange[0], prange[1], rgbcolor = (1,0,0))
    fmax = f.find_maximum_on_interval(prange[0], prange[1])[0]
    fmin = f.find_minimum_on_interval(prange[0], prange[1])[0]
    show(fplot + tanplot, xmin = prange[0], xmax = prange[1], ymax =
fmax, ymin = fmin)
```



Чисельне інтегрування за методом середніх прямокутників

```
var('x')
@interact
def midpoint(n = slider(1,100,1,4), f = input_box(default = "x^2", type
= str), start = input_box(default = "0", type = str), end =
input_box(default = "1", type = str)):
    a = N(start)
    b = N(end)
    func = sage_eval(f, locals={'x':x})
    dx = (b-a)/n
    midxs = [q*dx+dx/2 + a for q in range(n)]
    midys = [func(x_val) for x_val in midxs]
    rects = Graphics()
```

```
    for q in range(n):
        xm = midxs[q]
        ym = midys[q]
        rects = rects + line([[xm-dx/2,0],[xm-
dx/2,ym],[xm+dx/2,ym],[xm+dx/2,0]], rgbcolor = (1,0,0)) + point((xm,ym),
rgbcolor = (1,0,0))
    min_y = find_minimum_on_interval(func,a,b)[0]
    max_y = find_maximum_on_interval(func,a,b)[0]
    html('<h3>Чисельне інтегрування за методом середніх
прямокутників</h3>')
    html('$\int_{a}^{b}{f(x) dx} {\\approx} \sum_i{f(x_i) \Delta x}$')
    print "\n\nЧисельний результат Sage: " +
str(integral_numerical(func,a,b,max_points = 200)[0])
    print "Очикуваний результат за методом середніх прямокутників: " +
str(RDF(dx*sum([midys[q] for q in range(n)])))
    show(plot(func,a,b) + rects, xmin = a, xmax = b, ymin = min_y, ymax
= max_y)
```

n 4

f x^2

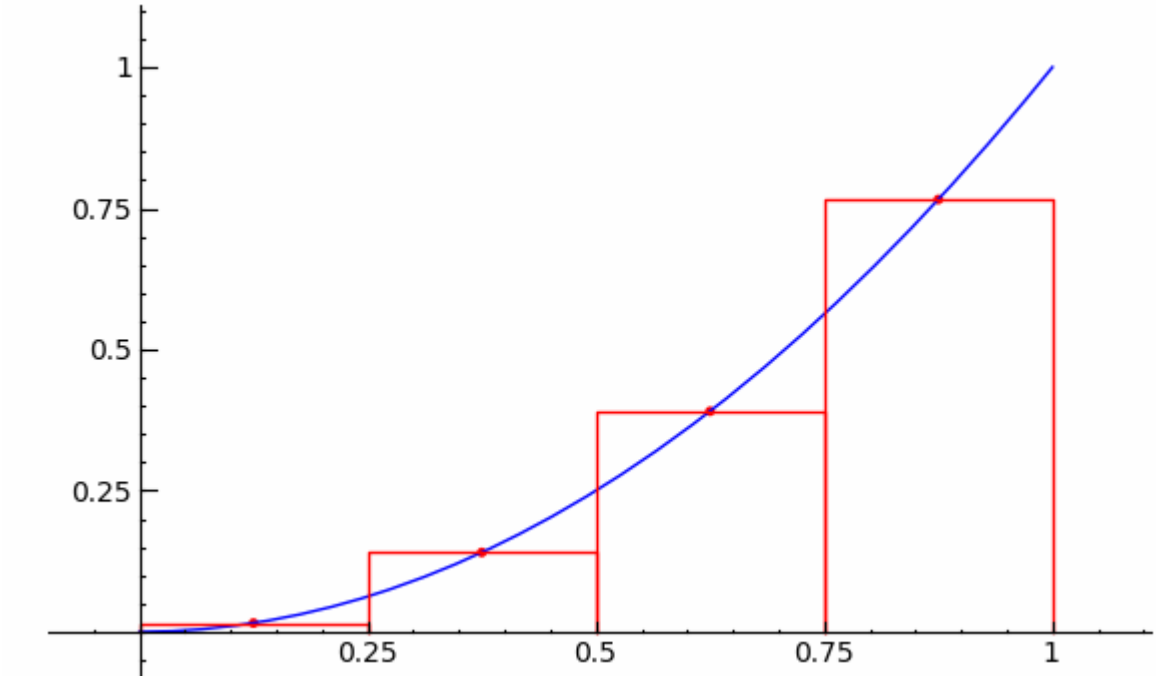
start 0

end 1

Чисельне інтегрування за методом середніх прямокутників

$$\int_a^b f(x)dx \approx \sum_i f(x_i)\Delta x$$

```
Чисельний результат Sage: 0.333333333333
Очикуваний результат за методом
середніх прямокутників: 0.328125
```



Знаходження коренів методом Ньютона

```
State = Data = None
@interact
def newtraph(f = input_box(default=8*sin(x)*exp(-x)-1, label='f(x)'),
xmin = input_box(default=0), xmax = input_box(default=4*pi), x0 =
input_box(default=3, label='x0'), show_calcs = ("Показувати
обчислення",True), step = ['Наступний','Попередній', 'Скинути'] ):
    global State, Data
    state = [f,xmin,xmax,x0,show_calcs]
    if (state != State) or (step == 'Скинути'):
        Data = [ 1 ]
        State = state
    elif step == 'Наступний':
        N, = Data
        Data = [ N+1 ]
    elif step == 'Попередній':
        N, = Data
        if N > 1:
            Data = [ N-1 ]
    N, = Data
    df = diff(f)
    theplot = plot( f, xmin, xmax )
    theplot += text( '\n$x_0$', (x0,0), rgbcolor=(1,0,0),
vertical_alignment="bottom" if f(x0) < 0 else "top" )
    theplot += points( [(x0,0)], rgbcolor=(1,0,0) )
    Trace = []
    def Err( msg, Trace=Trace ):
        Trace.append( '<font color="red"><b>Помилка: %s!!</b></font>' %
(msg,) )
    def Disp( s, color="blue", Trace=Trace ):
        Trace.append( """<font color="%s">$ %s $</font>""" %
(color,s,) )
    Disp( """f(x) = %s""" % (latex(f),) )
    Disp( """f'(x) = %s""" % (latex(df),) )
    stop = False
    is_inf = False
    xi = x0
    for i in range(N):
        fi = RR(f(xi))
        fpi = RR(df(xi))
        theplot += points( [(xi,fi)], rgbcolor=(1,0,0) )
        theplot += line( [(xi,0),(xi,fi)], linestyle=':',
rgbcolor=(1,0,0) )
        Disp( """i = %d""" % (i,) )
        Disp( """~~~~x_{%d} = %.4g""" % (i,xi) )
        Disp( """~~~~f(x_{%d}) = %.4g""" % (i,fi) )
        Disp( """~~~~f'(x_{%d}) = %.4g""" % (i,fpi) )
        if fpi == 0.0:
            Err( 'Похідна дорівнює 0 на кроці %d' % (i+1,) )
            is_inf = True
            show_calcs = True
        else:
```

```
            xip1 = xi - fi/fpi
            Disp( r"""~~~~x_{%d} = %.4g - ({%.4g})/({%.4g}) = %.4g""" %
(i+1,xi,fi,fpi,xip1) )
            if abs(xip1) > 10*(xmax-xmin):
                Err( 'Похідна надто близька до 0!' )
                is_inf = True
                show_calcs = True
            elif not ((xmin - 0.5*(xmax-xmin)) <= xip1 <= (xmax +
0.5*(xmax-xmin))):
                Err( 'значення x поза діапазоном; можливе розходження!')
                stop = True
                show_calcs = True
        if is_inf:
            xl = xi - 0.05*(xmax-xmin)
            xr = xi + 0.05*(xmax-xmin)
            yl = yr = fi
        else:
            xl = min(xi,xip1) - 0.01*(xmax-xmin)
            xr = max(xi,xip1) + 0.01*(xmax-xmin)
            yl = -(xip1-xl)*fpi
            yr = (xr-xip1)*fpi
            theplot += text( '\n$x_{%d}$' % (i+1,), (xip1,0),
rgbcolor=(1,0,0), vertical_alignment="bottom" if f(xip1) < 0 else
"top" )
            theplot += points( [(xip1,0)], rgbcolor=(1,0,0) )
        theplot += line( [(xl,yl),(xr,yr)], rgbcolor=(1,0,0) )
        if stop or is_inf:
            break
        epsa = 100.0*abs((xip1-xi)/xip1)
        nsf = 2 - log(2.0*epsa)/log(10.0)
        Disp( r"""~~~~~~~~\epsilon_a = \left|(%.4g -
%.4g)/%.4g\right|\times100\%% = %.4g \%%""" % (xip1,xi,xip1,epsa) )
        Disp( r"""~~~~~~~~num.~sig.~fig. \approx %.2g""" % (nsf,) )
        xi = xip1
    show( theplot, xmin=xmin, xmax=xmax )
    if show_calcs:
        for t in Trace:
            html( t )
```

$f(x) = 8\,e^{-x}\sin\left(x\right) - 1$

$f'(x) = -8\,e^{-x}\sin\left(x\right) + 8\,e^{-x}\cos\left(x\right)$

$i = 0$

$x_0 = 3$

$f(x_0) = -0.9438$

$f'(x_0) = -0.4505$

$x_1 = 3 - (-0.9438)/(-0.4505) = 0.9051$

$\epsilon_a = \left|(0.9051 - 3)/0.9051\right| \times 100\% = 231.5\%$

$num.\,sig.\,fig. \approx -0.67$

$i = 1$

$x_1 = 0.9051$

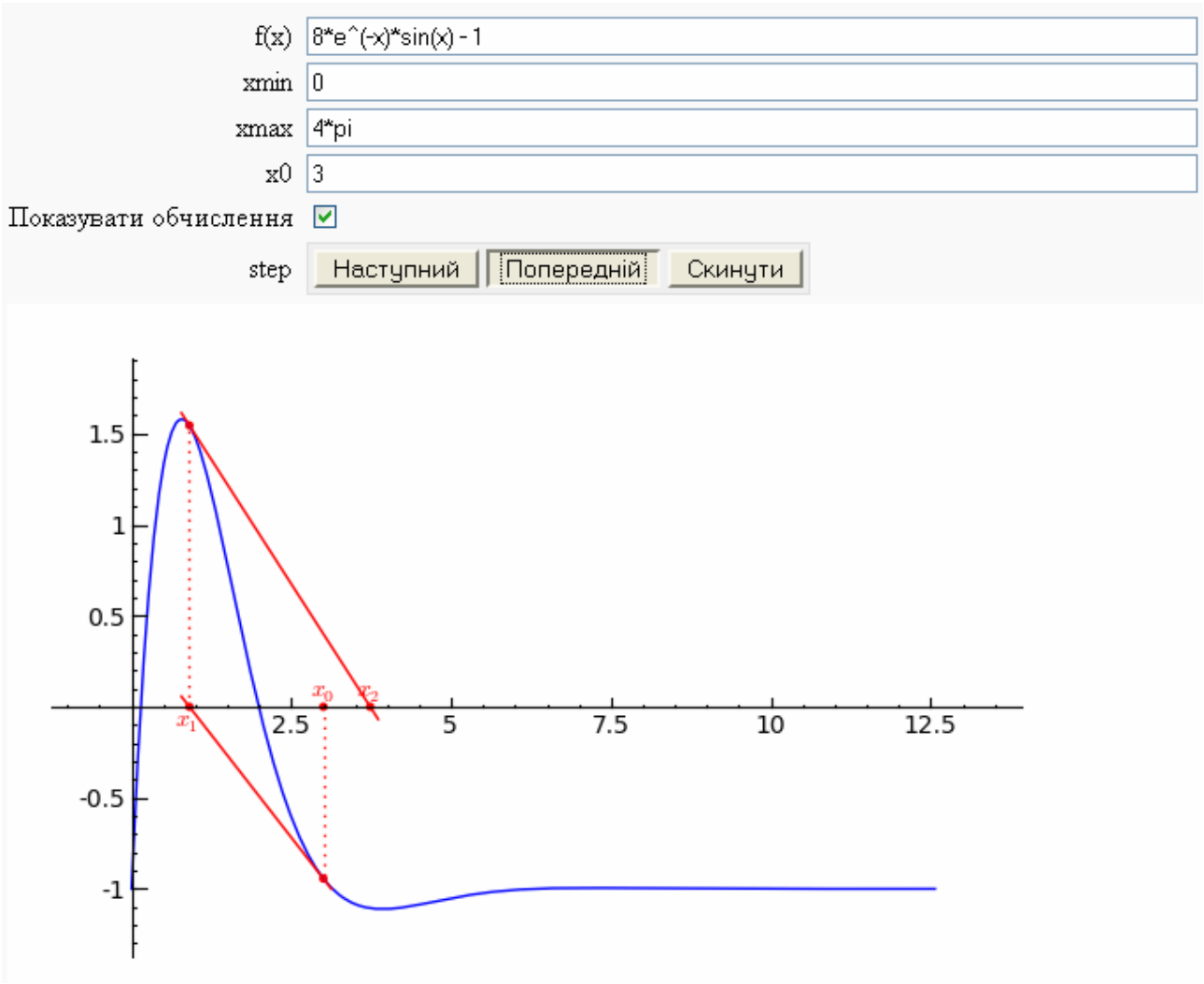
$f(x_1) = 1.545$

$f'(x_1) = -0.5465$

$x_2 = 0.9051 - (1.545)/(-0.5465) = 3.732$

$\epsilon_a = \left|(3.732 - 0.9051)/3.732\right| \times 100\% = 75.75\%$

$num.\,sig.\,fig. \approx -0.18$



Побудова функцій

```
x = var('x')
@interact
def _(f=sin(x), g=cos(x), xrange=input_box((0,1)), yrange='auto', a=1,
action=selector(['f', 'df/dx', 'інтеграл від f', 'чисельник f',
'знаменник f', '1/f', 'обернена до f', 'f+a', 'f-a', 'f*a', 'f/a',
'f^a', 'f(x+a)', 'f(x*a)', 'f+g', 'f-g', 'f*g', 'f/g', 'f(g)'],
width=15, nrows=5, label="h = "),
      do_plot = ("Зобразити графіки", True)):
    try:
        f = SR(f); g = SR(g); a = SR(a)
    except TypeError, msg:
        print msg[-200:]
        print "f, g або a не є символьними виразами."
        return
    if not (isinstance(xrange, tuple) and len(xrange) == 2):
        xrange = (0,1)
    h = 0; lbl = ''
    if action == 'f':
        h = f
        lbl = 'f'
    elif action == 'df/dx':
        h = f.derivative(x)
```

```
    lbl = '\\frac{df}{dx}'
elif action == 'інтеграл від f':
    h = f.integrate(x)
    lbl = '\\int f dx'
elif action == 'чисельник f':
    h = f.numerator()
    lbl = '\\text{numer(f)}'
elif action == 'знаменник f':
    h = f.denominator()
    lbl = '\\text{denom(f)}'
elif action == '1/f':
    h = 1/f
    lbl = '\\frac{1}{f}'
elif action == 'обернена до f':
    h = solve(f == var('y'), x)[0].rhs()
    lbl = 'f^{-1}(y)'
elif action == 'f+a':
    h = f+a
    lbl = 'f + a'
elif action == 'f-a':
    h = f-a
    lbl = 'f - a'
elif action == 'f*a':
    h = f*a
    lbl = 'f \\times a'
elif action == 'f/a':
    h = f/a
    lbl = '\\frac{f}{a}'
elif action == 'f^a':
    h = f^a
    lbl = 'f^a'
elif action == 'f^a':
    h = f^a
    lbl = 'f^a'
elif action == 'f(x+a)':
    h = f(x+a)
    lbl = 'f(x+a)'
elif action == 'f(x*a)':
    h = f(x*a)
    lbl = 'f(xa)'
elif action == 'f+g':
    h = f+g
    lbl = 'f + g'
elif action == 'f-g':
    h = f-g
    lbl = 'f - g'
elif action == 'f*g':
    h = f*g
    lbl = 'f \\times g'
elif action == 'f/g':
    h = f/g
    lbl = '\\frac{f}{g}'
elif action == 'f(g)':
```

```
        h = f(g)
        lbl = 'f(g)'
    html('<center><font color="red">$f = %s$</font></center>'%latex(f))
    html('<center><font color="green">$g =
%s$</font></center>'%latex(g))
    html('<center><font color="blue"><b>$h = %s =
%s$</b></font></center>'%(lbl, latex(h)))
    if do_plot:
        P = plot(f, xrange, color='red', thickness=2) +  \
            plot(g, xrange, color='green', thickness=2) + \
            plot(h, xrange, color='blue', thickness=2)
        if yrange == 'auto':
            show(P, xmin=xrange[0], xmax=xrange[1])
        else:
            yrange = sage_eval(yrange)
            show(P, xmin=xrange[0], xmax=xrange[1], ymin=yrange[0],
ymax=yrange[1])
```

f: sin(x)
g: x*cos(x)
xrange: (0, 2*pi)
yrange: (-3, 3)
a: 2

h =

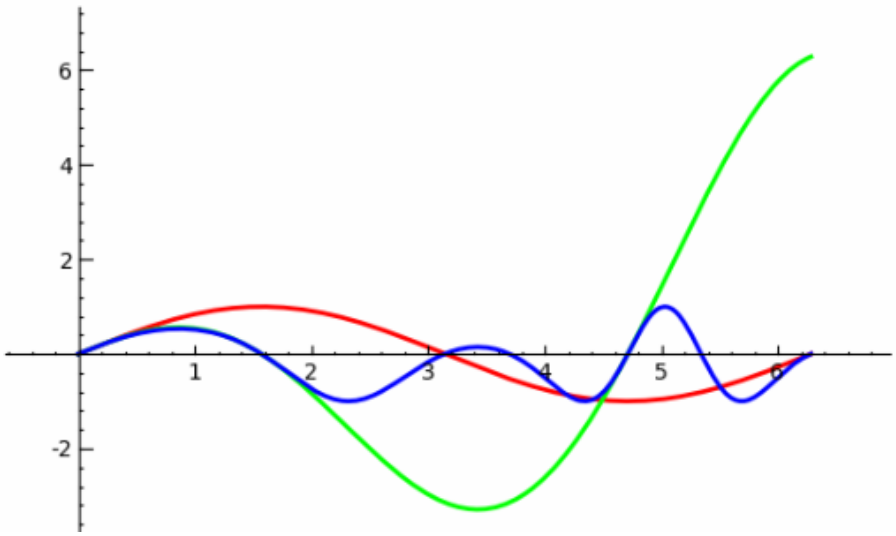
f	df/dx	інтеграл від f	чисельник f
знаменник f	1/f	обернена до f	f+a
f-a	f*a	f/a	f^a
f(x+a)	f(x*a)	f+g	f-g
f*g	f/g	f(g)	

Зобразити графіки ☑

$$f = \sin(x)$$

$$g = x\cos(x)$$

$$h = f(g) = \sin(x\cos(x))$$



Перетворення координат

```
var('u v')
from sage.ext.fast_eval import fast_float
from functools import partial
@interact
def trans(x=input_box(u^2-v^2, label="x=",type=SR), \
         y=input_box(u*v+cos(u*v), label="y=",type=SR), \
         t_val=slider(0,10,0.2,6, label="Довжина кривих"), \
         u_percent=slider(0,1,0.05,label="<font color='red'>u</font>",
default=.7),
         v_percent=slider(0,1,0.05,label="<font color='blue'>v</font>",
default=.7),
         u_range=input_box(range(-5,5,1), label="u-лінії"),
         v_range=input_box(range(-5,5,1), label="v-лінії")):
     thickness=4
     u_val = min(u_range)+(max(u_range)-min(u_range))*u_percent
     v_val = min(v_range)+(max(v_range)-min(v_range))*v_percent
     t_min = -t_val
     t_max = t_val
     g1=sum([parametric_plot((i,v), t_min,t_max, rgbcolor=(1,0,0)) for
i in u_range])
     g2=sum([parametric_plot((u,i), t_min,t_max, rgbcolor=(0,0,1)) for
i in v_range])
     vline_straight=parametric_plot((u,v_val), t_min,t_max,
rgbcolor=(0,0,1), linestyle='-',thickness=thickness)
     uline_straight=parametric_plot((u_val, v),
t_min,t_max,rgbcolor=(1,0,0), linestyle='-',thickness=thickness)


(g1+g2+vline_straight+uline_straight).save("uv_coord.png",aspect_ratio=
1, figsize=[5,5], axes_labels=['$u$','$v$'])
     xuv = fast_float(x,'u','v')
     yuv = fast_float(y,'u','v')
     xvu = fast_float(x,'v','u')
     yvu = fast_float(y,'v','u')
     g3=sum([parametric_plot((partial(xuv,i),partial(yuv,i)),
t_min,t_max, rgbcolor=(1,0,0)) for i in u_range])
     g4=sum([parametric_plot((partial(xvu,i),partial(yvu,i)),
t_min,t_max, rgbcolor=(0,0,1)) for i in v_range])
     vline=parametric_plot((partial(xvu,v_val),partial(yvu,v_val)),
t_min,t_max, rgbcolor=(0,0,1), linestyle='-',thickness=thickness)
     uline=parametric_plot((partial(xuv,u_val),partial(yuv,u_val)),
t_min,t_max,rgbcolor=(1,0,0), linestyle='-',thickness=thickness)
     (g3+g4+vline+uline).save("xy_coord.png", aspect_ratio=1,
figsize=[5,5], axes_labels=['$x$','$y$'])
     print jsmath("x=%s, \: y=%s"%(latex(x), latex(y)))
     print "<html><table><tr><td><img
src='cell://uv_coord.png'/></td><td><img
src='cell://xy_coord.png'/></td></tr></table></html>"
```

x= u^2 - v^2

y= u*v

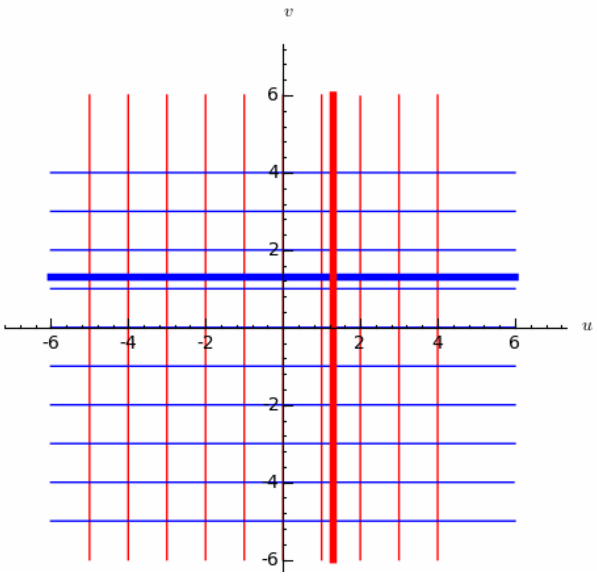
Довжина кривих 6.000000000000000

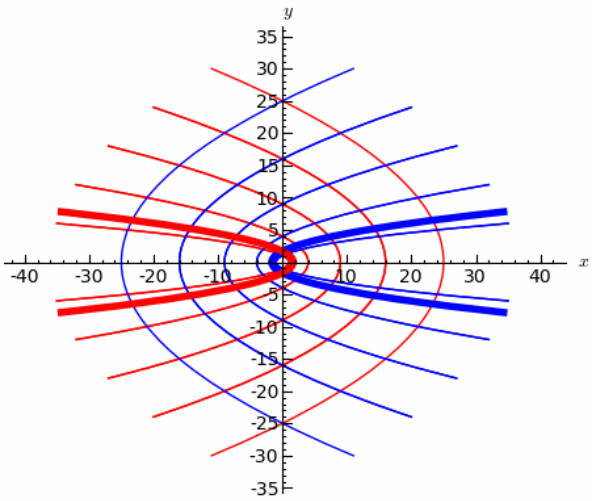
u 0.7000000000000000

v 0.7000000000000000

u-лінії [-5, -4, -3, -2, -1, 0, 1, 2, 3, 4]

v-лінії [-5, -4, -3, -2, -1, 0, 1, 2, 3, 4]



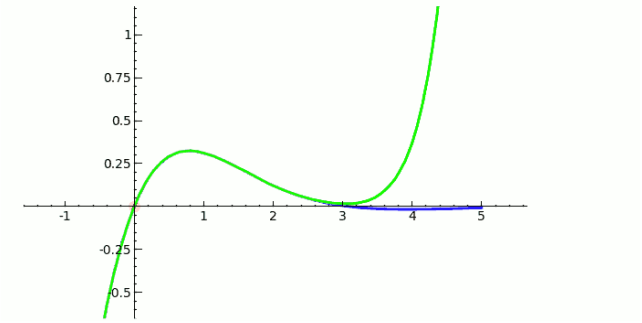


Розклад у ряд Тейлора

```
var('x')
x0  = 0
f   = sin(x)*e^(-x)
p   = plot(f,-1,5, thickness=2)
dot = point((x0,f(x0)),pointsize=80,rgbcolor=(1,0,0))
@interact
def _(order=(1..12)):
  ft = f.taylor(x,x0,order)
  pt = plot(ft,-1, 5, color='green', thickness=2)
  html('$f(x)\;=\;%s$'%latex(f))
  html('$\hat{f}(x;%s)\;=\;%s+\mathcal{O}(x^{%s})$'%(x0,latex(ft),
order+1))
 show(dot + p + pt, ymin = -.5, ymax = 1)
```

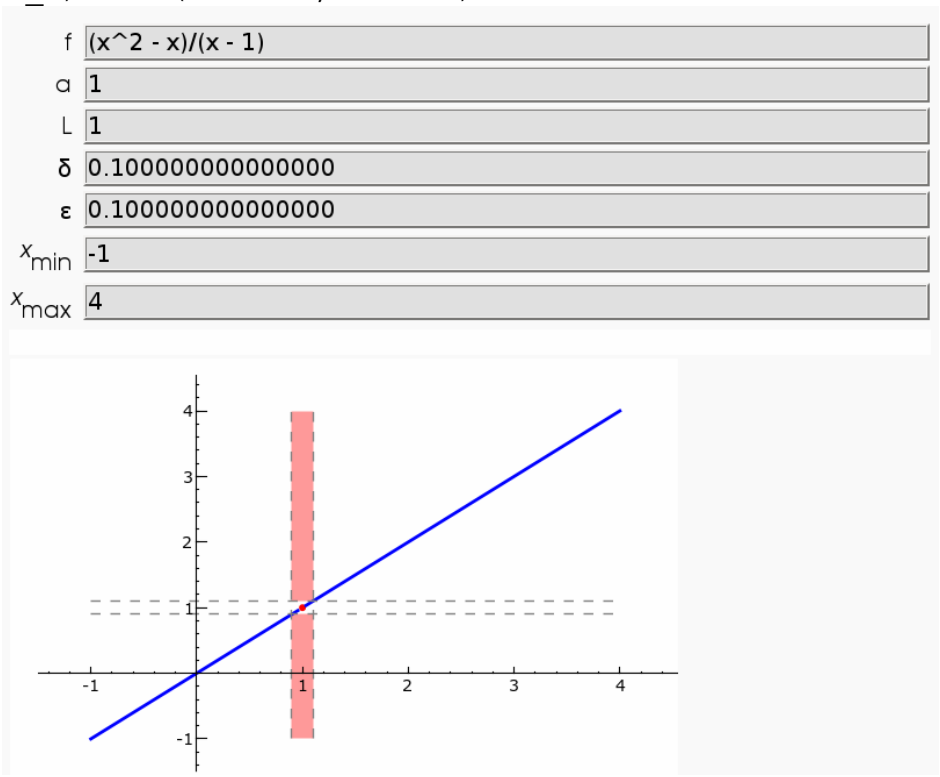
order

$$f(x) = e^{-x}\sin(x)$$
$$\hat{f}(x;0) = x - x^2 + \frac{x^3}{3} - \frac{x^5}{30} + \frac{x^6}{90} - \frac{x^7}{630} + \frac{x^9}{22680} - \frac{x^{10}}{113400} + \frac{x^{11}}{1247400} + \mathcal{O}(x^{13})$$



Означення границі

```
@interact
def delta_epsilon(f = input_box(default=(x^2-x)/(x-1)),
a=input_box(default=1), L = input_box(default=1),
delta=input_box(label="δ",default=0.1),
epsilon=input_box(label="ε",default=0.1),
xm=input_box(label="<i>x</i><sub>min</sub>",default=-1),
xM=input_box(label="<i>x</i><sub>max</sub>",default=4)):
    f_left_plot = plot(f,xm,a-delta/3,thickness=2)
    f_right_plot = plot(f,a+delta/3,xM,thickness=2)
    epsilon_line_1 = line([(xm,L-epsilon),(xM,L-epsilon)],
rgbcolor=(0.5,0.5,0.5),linestyle='--')
    epsilon_line_2 = line([(xm,L+epsilon),(xM,L+epsilon)],
rgbcolor=(0.5,0.5,0.5),linestyle='--')
    ym = min(f_right_plot.ymin(),f_left_plot.ymin())
    yM = max(f_right_plot.ymax(),f_left_plot.ymax())
    bad_region_1 = polygon([(a-delta,L+epsilon),(a-
delta,yM),(a+delta,yM),(a+delta,L+epsilon)], rgbcolor=(1,0.6,0.6))
    bad_region_2 = polygon([(a-delta,L-epsilon),(a-
delta,ym),(a+delta,ym),(a+delta,L-epsilon)], rgbcolor=(1,0.6,0.6))
    aL_point = point((a,L),rgbcolor=(1,0,0),pointsize=20)
    delta_line_1 = line([(a-delta,ym),(a-
delta,yM)],rgbcolor=(0.5,0.5,0.5),linestyle='--')
    delta_line_2 =
line([(a+delta,ym),(a+delta,yM)],rgbcolor=(0.5,0.5,0.5),linestyle='--')
    (f_left_plot +f_right_plot +epsilon_line_1 +epsilon_line_2
+delta_line_1 +delta_line_2 +aL_point +bad_region_1
+bad_region_2).show(xmin=xm,xmax=xM)
```



Знаходження границі відношення sin(x)/x при x→0

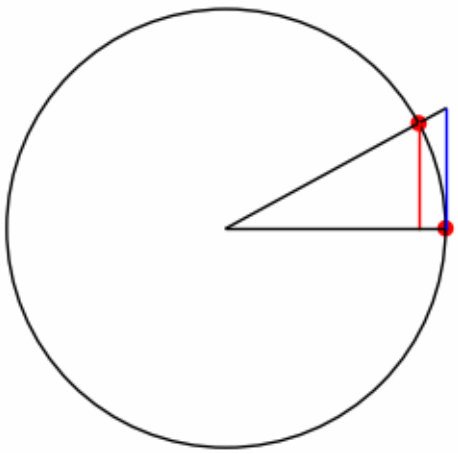
```
x=var('x')
```

```
@interact
def _(x = slider(-7/10,7/10,1/20,1/2)):
    html('<h3>Графічна ілюстрація $\lim_{x -> 0} \sin(x)/x =1$</h3>')
    html('На зображеному колі довжина <font color=red>червоної
лінії</font> відповідає |sin(x)|,')
    html('довжина <font color=blue>синьої лінії</font> відповідає
|tg(x)|, де x - довжина дуги кола.')
    html('З рисунка видно, що |sin(x)| $\le$ |x| $\le$ |tg(x)|.')
    html('Це випливає з того, що cos(x) $\le$ sin(x)/x $\le$ 1, коли x
прямує до 0.')
    html('З того, що $\lim_{x ->0} \cos(x) =1$, робимо висновок, що
$\lim_{x -> 0} \sin(x)/x =1$.')
    if not (x == 0):
        pretty_print("sin(x)/x = "+str(sin(float(x))/float(x)))
    elif x == 0:
        pretty_print("The limit of sin(x)/x as x tends to 0 is 1.")
    C=circle((0,0),1, rgbcolor='black')
    mvp = (cos(x),sin(x));tpt = (1, tan(x))
    p1 = point(mvp, pointsize=30, rgbcolor='red'); p2 = point((1,0),
pointsize=30, rgbcolor='red')
    line1 = line([(0,0),tpt], rgbcolor='black'); line2 =
line([(cos(x),0),mvp], rgbcolor='red')
    line3 = line([(0,0),(1,0)], rgbcolor='black'); line4 =
line([(1,0),tpt], rgbcolor='blue')
    result = C+p1+p2+line1+line2+line3+line4
    result.show(aspect_ratio=1, figsize=[3,3], axes=False)
```

x 1/2

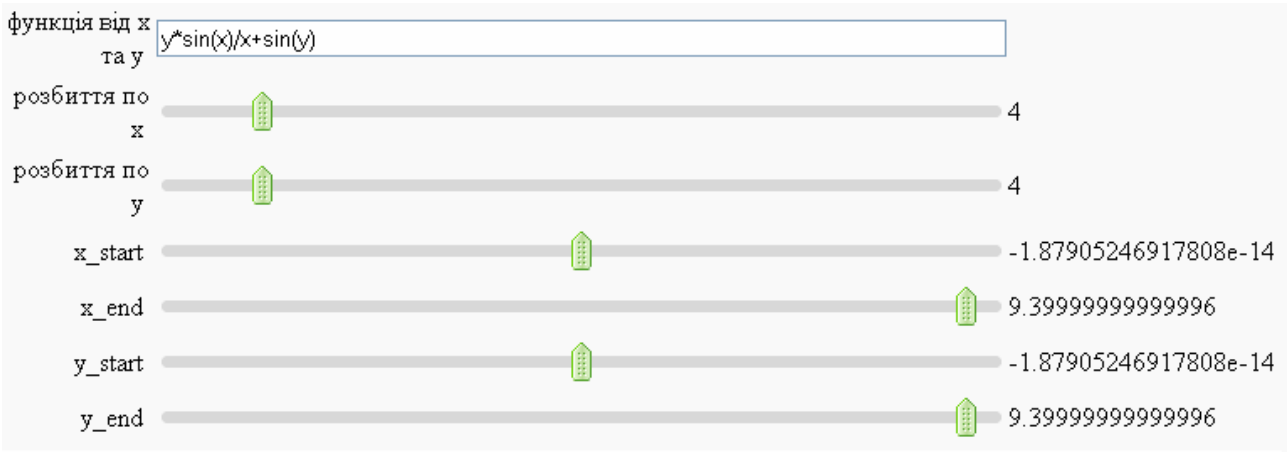
Графічна ілюстрація $\lim_{x->0} \sin(x)/x = 1$

На зображеному колі довжина червоної лінії відповідає |sin(x)|,
довжина синьої лінії відповідає |tg(x)|, де x - довжина дуги кола.
З рисунка видно, що |sin(x)| $\le$ |x| $\le$ |tg(x)|.
Це випливає з того, що cos(x) $\le$ sin(x)/x $\le$ 1, коли x прямує до 0.
З того, що $\lim_{x->0} \cos(x) = 1$, робимо висновок, що $\lim_{x->0} \sin(x)/x = 1$.
sin(x)/x = 0.958851077208



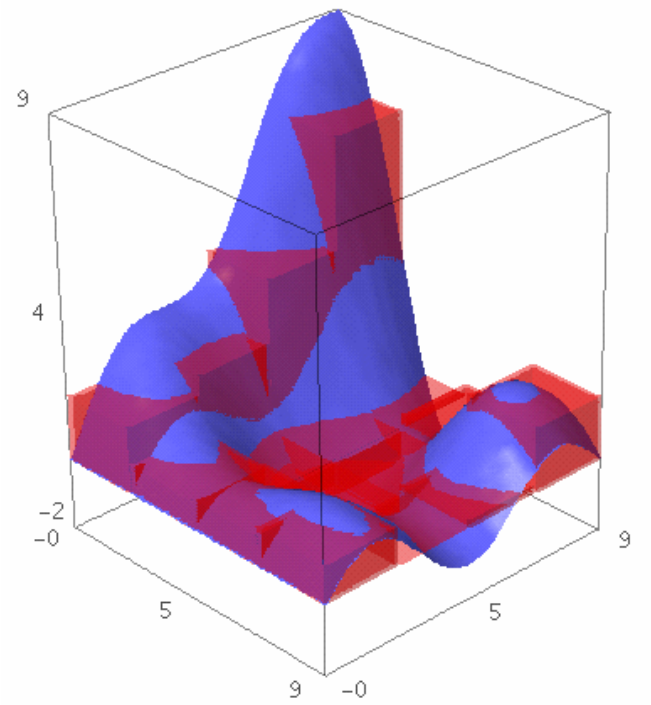
Чисельне інтегрування за методом середніх прямокутників функції двох змінних

```
from sage.plot.plot3d.platonic import index_face_set
def cuboid(v1,v2,**kwds):
    ptlist = []
    for vi in (v1,v2):
        for vj in (v1,v2):
            for vk in (v1,v2):
                ptlist.append([vi[0],vj[1],vk[2]])
    f_incs = [[0, 2, 6, 4], [0, 1, 3, 2], [0, 1, 5, 4], [1, 3, 7, 5],
[2, 3, 7, 6], [4, 5, 7, 6]]

    if 'aspect_ratio' not in kwds:
        kwds['aspect_ratio'] = [1,1,1]
    return index_face_set(f_incs,ptlist,enclosed = True, **kwds)
var('x,y')
R16 = RealField(16)
npi = RDF(pi)
sin,cos = math.sin,math.cos
@interact
def midpoint2d(func =
input_box('y*sin(x)/x+sin(y)',type=str,label='функція від x та y'), nx
= slider(2,20,1,3,label='розбиття по x'), ny =
slider(2,20,1,3,label='розбиття по y'), x_start = slider(-10,10,.1,0),
x_end = slider(-10,10,.1,3*npi), y_start= slider(-10,10,.1,0), y_end=
slider(-10,10,.1,3*npi)):
    f = sage_eval('lambda x,y: ' + func)
    delx = (x_end - x_start)/nx
    dely = (y_end - y_start)/nx
    xvals = [RDF(x_start + (i+1.0/2)*delx) for i in range(nx)]
    yvals = [RDF(y_start + (i+1.0/2)*dely) for i in range(ny)]
    num_approx = 0
    cubs = []
    darea = delx*dely
    for xv in xvals:
        for yv in yvals:
            num_approx += f(xv,yv)*darea
            cubs.append(cuboid([xv-delx/2,yv-
dely/2,0],[xv+delx/2,yv+dely/2,f(xv,yv)], opacity = .5, rgbcolor =
(1,0,0)))
    html("$$\int_{"+str(R16(y_start))+"}^{"+str(R16(y_end))+"} "+
"\int_{"+str(R16(x_start))+"}^{"+str(R16(x_end))+"} "+func+"\ dx \
dy$$")
    html('<p style="text-align: center;">Чисельне наближення: ' +
str(num_approx)+'</p>')
    p1 = plot3d(f,(x,x_start,x_end),(y,y_start,y_end))
    show(p1+sum(cubs))
```



$$\int_{-1.879e-14}^{9.400}\int_{-1.879e-14}^{9.400} y * sin(x)/x + sin(y)\, dx\, dy$$

```
Чисельне наближення: 136.143869777
```

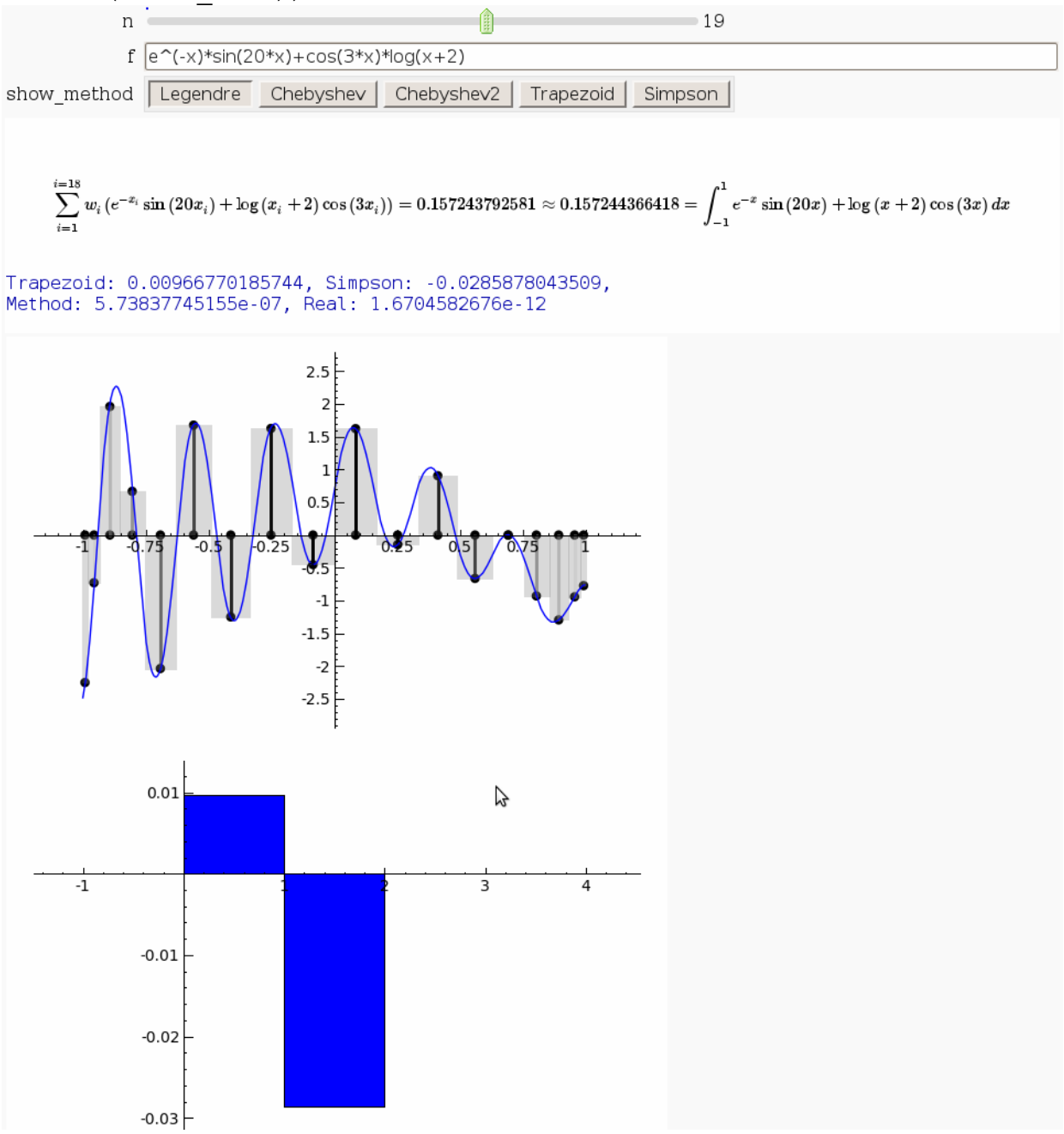


Квадратури Гауса (Лежандра)

```
from scipy.special.orthogonal import p_roots, t_roots, u_roots
from scipy.integrate import quad, trapz, simps
from sage.ext.fast_eval import fast_float
from numpy import linspace
show_weight_graph=False
methods = {'Лежандра': {'w': 1, 'xmin': -1, 'xmax': 1, 'func': p_roots},
'Чебишева': {'w': 1/sqrt(1-x**2), 'xmin': -1, 'xmax': 1, 'func':
t_roots}, 'Чебишева(2)': {'w': sqrt(1-x**2), 'xmin': -1, 'xmax': 1,
'func': u_roots}, 'трапецій': {'w': 1, 'xmin': -1, 'xmax': 1, 'func':
lambda n: (linspace(-1r,1,n), numpy.array([1.0r]+[2.0r]*(n-
2)+[1.0r])*1.0r/n)}, 'Сімпсона': {'w': 1, 'xmin': -1, 'xmax': 1, 'func':
```

```
lambda n: (linspace(-1r,1,n), numpy.array([1.0r]+[4.0r,2.0r]*int((n-
3.0r)/2.0r)+[4.0r,1.0r])*2.0r/(3.0r*n))}}
var("x")
def box(center, height, area,**kwds):
    width2 = 1.0*area/height/2.0
    return polygon([(center-
width2,0),(center+width2,0),(center+width2,height),(center-
width2,height)],**kwds)

@interact
def
weights(n=slider(1,30,1,default=10),f=input_box(default=3*x+cos(10*x)),
show_method=["Лежандра", "Чебишева", "Чебишева(2)", "трапецій",
"Сімпсона"]):
    ff = fast_float(f,'x')
    method = methods[show_method]
    xcoords,w = (method['func'])(int(n))
    xmin = method['xmin']
    xmax = method['xmax']
    plot_min = max(xmin, -10)
    plot_max = min(xmax, 10)
    scaled_func = f*method['w']
    scaled_ff = fast_float(scaled_func)
    coords = zip(xcoords,w)
    max_weight = max(w)
    coords_scaled = zip(xcoords,w/max_weight)
    f_graph = plot(scaled_func,plot_min,plot_max)
    boxes = sum(box(x,ff(x),w*ff(x),rgbcolor=(0.5,0.5,0.5),alpha=0.3)
for x,w in coords)
    stems = sum(line([(x,0),(x,scaled_ff(x))],rgbcolor=(1-y,1-y,1-
y),thickness=2,markersize=6,alpha=y) for x,y in coords_scaled)
    points =
sum([point([(x,0),(x,scaled_ff(x))],rgbcolor='black',pointsize=30) for
x,_ in coords])
    graph = stems+points+f_graph+boxes
    if show_weight_graph:
        graph += line([(x,y) for x,y in coords_scaled],
rgbcolor='green',alpha=0.4)
    show(graph,xmin=plot_min,xmax=plot_max)
    approximation = sum([w*ff(x) for x,w in coords])
    integral,integral_error = scipy.integrate.quad(scaled_ff, xmin,
xmax)
    x_val = linspace(min(xcoords), max(xcoords),n)
    y_val = map(scaled_ff,x_val)
    trapezoid = integral-trapz(y_val, x_val)
    simpson = integral-simps(y_val, x_val)
    html("$$\sum_{i=1}^{i=%s}w_i\left(%s\\right)= %s\\approx %s
=\int_{-1}^{1}%s \,dx$$"%(n,latex(f.subs(x="x_i")), approximation,
integral, latex(scaled_func)))
    error_data = [trapezoid, simpson, integral-
approximation,integral_error]
    print "Результат за методом трапецій: %s, Сімпсона: %s, \nза
обраним методом: %s, обчислений Sage: %s"%tuple(error_data)
```

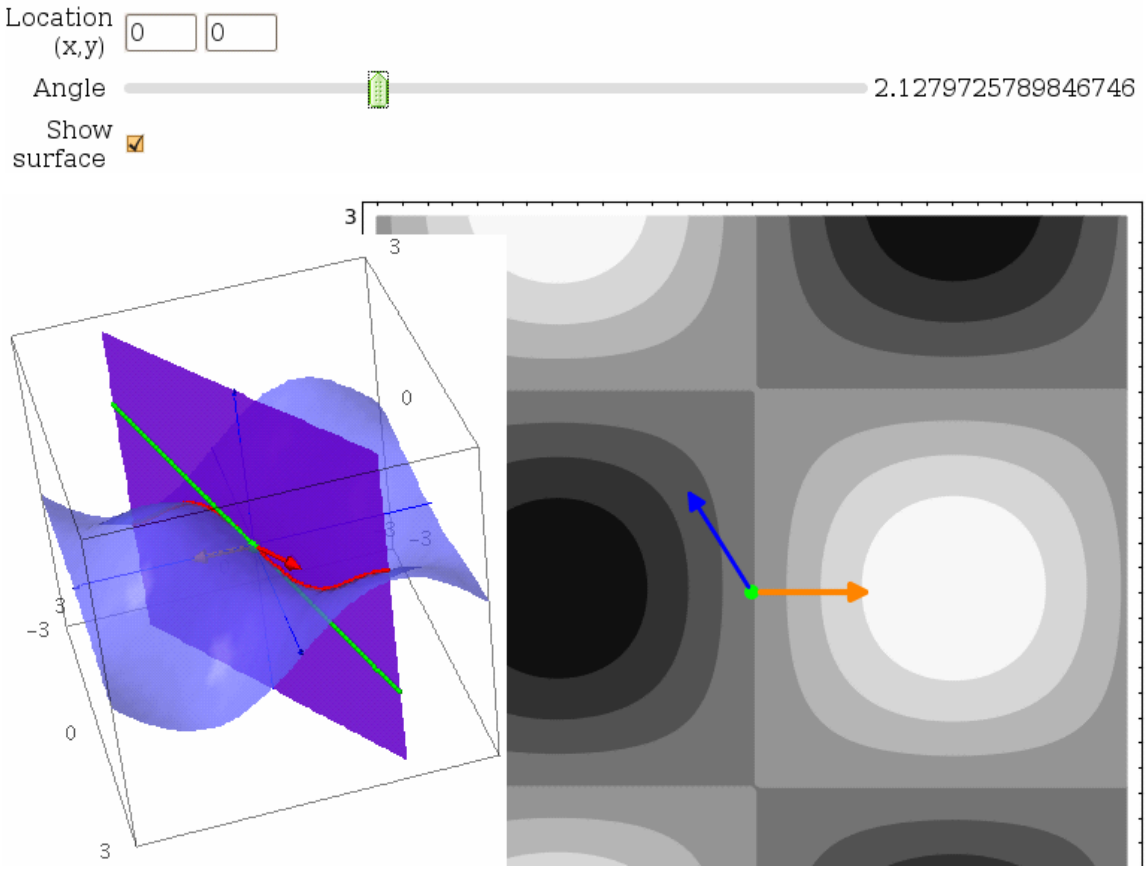
```
    show(bar_chart(error_data,width=1),ymin=min(error_data),
ymax=max(error_data))
```



Похідна за напрямом

```
var('x,y,t,z')
f(x,y)=sin(x)*cos(y)
pif = float(pi)
line_thickness=3
@interact
def myfun(location=input_grid(1, 2, default=[0,0], label = "Координати
(x,y)", width=2), angle=slider(0, 2*pif, label = "Кут"),
show_surface=("Показувати поверхню", True)):
    location3d = vector(location[0]+[0])
    location = location3d[0:2]
    direction3d = vector(RDF, [cos(angle), sin(angle), 0])
    direction=direction3d[0:2]
    cos_angle = math.cos(angle)
```

```
    sin_angle = math.sin(angle)
    df = f.gradient()
    direction_vector=line3d([location3d, location3d+direction3d],
arrow_head=True, rgbcolor='red', thickness=line_thickness)
    curve_point = (location+t*direction).list()
    curve = parametric_plot(curve_point+[f(*curve_point)], (t,-
3,3),color='red',thickness=line_thickness)
    plane =
parametric_plot((cos_angle*x+location[0],sin_angle*x+location[1],t), (x,
-3,3), (t,-3,3),opacity=0.8, color='purple')
    pt = point3d(location3d.list(),color='green', size=10)
    tangent_line = parametric_plot((location[0]+t*cos_angle,
location[1]+t*sin_angle, f(*location)+t*df(*location)*(direction)), (t,
-3,3), thickness=line_thickness, color='green')
    picture3d = direction_vector+curve+plane+pt+tangent_line
    picture2d = contour_plot(f(x,y), (x,-3,3),(y,-3,3), plot_points=100)
    picture2d += arrow(location.list(), (location+direction).list())
    picture2d += point(location.list(),rgbcolor='green',pointsize=40)
    if show_surface:
        picture3d += plot3d(f, (x,-3,3),(y,-3,3),opacity=0.7)
    dff = df(location[0], location[1])
    dff3d = vector(RDF,dff.list()+[0])
    picture3d += line3d([location3d, location3d+dff3d], arrow_head=True,
rgbcolor='orange', thickness=line_thickness)
    picture2d += arrow(location.list(), (location+dff).list(),
rgbcolor='orange', width=line_thickness)
    show(picture3d,aspect=[1,1,1], axes=True)
    show(picture2d, aspect_ratio=1)
```



Тривимірні графіки точок та кривих

```
x,y, t, u, v =var('x y t u v')
INI_func='x^2-2*x+y^2-2*y'
INI_box='-1,3.2,-1,3.2'
INI_points='(1,1,\'green\'),(3/2,3/2),(0,1),(1,0),(0,0,\'black\'),(3,0,
\'black\'),(0,3,\'black\')'
INI_curves='(t,0,0,3,\'red\'),(0,t,0,3,\'green\'),(t,3-t,0,3)'
@interact
def _(func=input_box(INI_func,label="f(x,y)=",type=str),\
  bounds=input_box(INI_box,label="xmin,xmax,ymin,ymax",type=str),\
  st_points=input_box(INI_points,\
  label="точки <br><small><small>(пари у дужках, розділені комами,
можливо вказання кольору)</small></small>", type=str),\
  bnd_curves=input_box(INI_curves,label="криві на границях<br>
<small><small><i>(x(t),y(t),tmin,tmax,'opt_color')</i></small></small>",
type=str),\
 show_planes=("Показати нульові площини", False),
show_axes=("Зобразити вісі", True), show_table=("Показати таблицю",
True)):
 f=sage_eval('lambda x,y: ' + func)
 html(r'Функція $ f(x,y)=%s$ '%latex(f(x,y)))
 xmin,xmax,ymin,ymax=sage_eval('('+bounds+')')
 A=plot3d(f(x,y),(x,xmin,xmax),(y,ymin,ymax),opacity=0.5)
 if not(bool(st_points=='')):
     st_p=sage_eval('('+st_points+',)')
     html(r'<table border=1>')
     for current in range(len(st_p)):
         point_color='red'
         if bool(len(st_p[current])==3):
              point_color=st_p[current][2]
         x0=st_p[current][0]
         y0=st_p[current][1]
         z0=f(x0,y0)
         if show_table:
              html(r'<tr><td>$\quad f(%s,%s)\quad $</td><td>$\quad
%s$</td>\
              </tr>'%(latex(x0),latex(y0),z0.n()))
         A=A+point3d((x0,y0,z0),size=9,rgbcolor=point_color)
     html(r'</table>')
 if not(bool(bnd_curves=='')):
     bnd_cc=sage_eval('('+bnd_curves+',)',locals={'t':t})
     for current in range(len(bnd_cc)):
         bnd_c=bnd_cc[current]+('black',)

A=A+parametric_plot3d((bnd_c[0],bnd_c[1],f(bnd_c[0],bnd_c[1])),\
             (t,bnd_c[2],bnd_c[3]),thickness=3,rgbcolor=bnd_c[4])
 if show_planes:

A=A+plot3d(0,(x,xmin,xmax),(y,ymin,ymax),opacity=0.3,rgbcolor='gray')
     zmax=A.bounding_box()[1][2]
     zmin=A.bounding_box()[0][2]
```

```
     A=A+parametric_plot3d((u,0,v),(u,xmin,xmax),(v,zmin,zmax),
opacity=0.3,rgbcolor='gray')
     A=A+parametric_plot3d((0,u,v),(u,ymin,ymax),(v,zmin,zmax),
opacity=0.3,rgbcolor='gray')
 if show_axes:
     zmax=A.bounding_box()[1][2]
     zmin=A.bounding_box()[0][2]
     A=A+line3d([(xmin,0,0), (xmax,0,0)],
arrow_head=True,rgbcolor='black')
     A=A+line3d([(0,ymin,0), (0,ymax,0)],
arrow_head=True,rgbcolor='black')
     A=A+line3d([(0,0,zmin), (0,0,zmax)],
arrow_head=True,rgbcolor='black')
 show(A)
```

f(x,y)= x^2-2*x+y^2-2*y

xmin,xmax,ymin,ymax -1,3.2,-1,3.2

точки (пари у дужках, розділені комами, можливо вказання кольору) (1,1,'green'),(3/2,3/2),(0,1),(1,0),(0,0,'black'),(3,0,'black'),(0,3,'black')

криві на границях (x(t),y(t),tmin,tmax,'opt_color') (t,0,0.3,'red'),(0,t,0.3,'green'),(t,3-t,0.3)

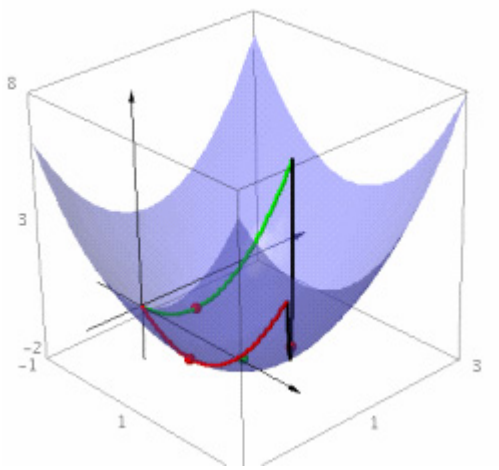
Показати нульові площини ☐

Зобразити вісі ☑

Показати таблицю ☑

Функція $f(x,y)=x^2+y^2-2x-2y$

$f(1,1)$	-2.00000000000000
$f(\frac{3}{2},\frac{3}{2})$	-1.50000000000000
$f(0,1)$	-1.00000000000000
$f(1,0)$	-1.00000000000000



Апроксимація функції двох змінних за диференціалом

```
x,y=var('x y')
@interact
def _(func=input_box('sqrt(x^3+y^3)',label="f(x,y)=",type=str), x0=1,
y0=2, deltax=slider(-1,1,0.01,0.2),\
  deltay=slider(-1,1,0.01,-0.4), xmin=0, xmax=2, ymin=0, ymax=3):
  f=sage_eval('lambda x,y: ' + func)
  derx(x,y)=diff(f(x,y),x)
  dery(x,y)=diff(f(x,y),y)
  tangent(x,y)=f(x0,y0)+derx(x0,y0)*(x-x0)+dery(x0,y0)*(y-y0)
  A=plot3d(f(x,y),(x,xmin,xmax),(y,ymin,ymax),opacity=0.5)
```

```
  B=plot3d(tangent(x,y),(x,xmin,xmax),(y,ymin,ymax),color='red',
opacity=0.5)
  C=point3d((x0,y0,f(x0,y0)),rgbcolor='blue',size=9)
  CC=point3d((x0+deltax,y0+deltay,f(x0+deltax,y0+deltay)),
rgbcolor='blue',size=9)
  D=point3d((x0+deltax,y0+deltay,tangent(x0+deltax,y0+deltay)),
rgbcolor='red',size=9)
  exact_value_ori=f(x0,y0).n(digits=10)
  exact_value=f(x0+deltax,y0+deltay)
  approx_value=tangent(x0+deltax,y0+deltay).n(digits=10)
  abs_error=(abs(exact_value-approx_value))
  html(r'Функція $ f(x,y)=%s \approx %s $
'%(latex(f(x,y)),latex(tangent(x,y))))
  html(r' $f %s = %s$'%(latex((x0,y0)),latex(exact_value_ori)))
  html(r'Зсунута точка $%s$'%latex(((x0+deltax),(y0+deltay))))
  html(r'Значення функції у зсунутій точці $%s$'%f(x0+deltax,y0+deltay))
  html(r'Значення дотичної площини у зсунутій точці
$%s$'%latex(approx_value))
  html(r'Похибка $%s$'%latex(abs_error))
  show(A+B+C+CC+D)
```

f(x,y)= sqrt(x^3+y^3)
x0 1
y0 2
deltax 0.370000000000001
deltay -0.399999999999999
xmin 0
xmax 2
ymin 0
ymax 3

Функція $f(x,y)=\sqrt{x^3+y^3}\approx \frac{1}{2}x+2y-\frac{3}{2}$

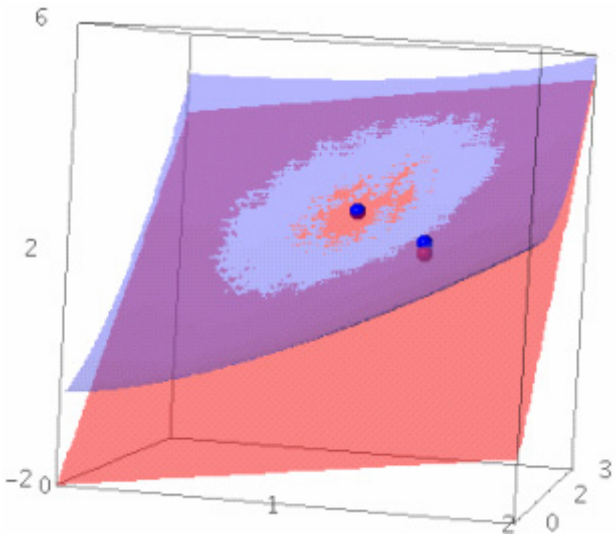
$f\left(1,2\right)=3.000000000$

Зсунута точка $\left(1.37000000000000,1.60000000000000\right)$

Значення функції у зсунутій точці 2.58212180192957

Значення дотичної площини у зсунутій точці 2.385000000

Похибка 0.1971218019



Апроксимація функції двох змінних рядом Тейлора

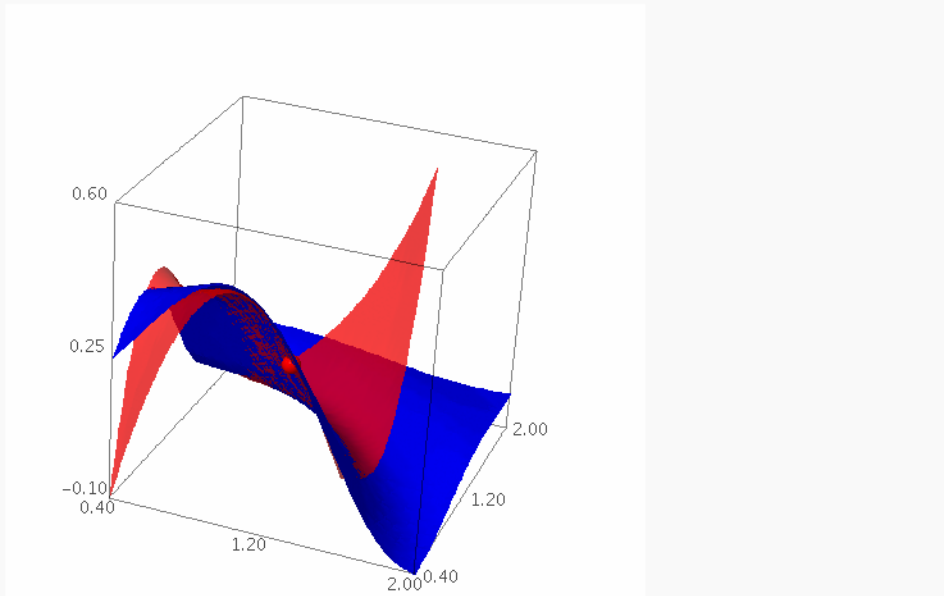
```
var('x y')
var('xx yy', ns=1)
G = sin(xx^2 + yy^2) * cos(yy) * exp(-0.5*(xx^2+yy^2))
def F(x,y):
    return G.subs(xx=x).subs(yy=y)
plotF = plot3d(F, (0.4, 2), (0.4, 2), adaptive=True, color='blue')
@interact
def _(x0=(0.5,1.5), y0=(0.5, 1.5),
      order=(1..10)):
    F0 = float(G.subs(xx=x0).subs(yy=y0))
    P = (x0, y0, F0)
    dot = point3d(P, size=15, color='red')
    plot = dot + plotF
    approx = F0
    for n in range(1, order+1):
        for i in range(n+1):
            if i == 0:
                deriv = G.diff(yy, n)
            elif i == n:
                deriv = G.diff(xx, n)
            else:
                deriv = G.diff(xx, i).diff(yy, n-i)
            deriv = float(deriv.subs(xx=x0).subs(yy=y0))
            coeff = binomial(n, i)/factorial(n)
            approx += coeff * deriv * (x-x0)^i * (y-y0)^(n-i)
    plot += plot3d(approx, (x, 0.4, 1.6),
             (y, 0.4, 1.6), color='red', opacity=0.7)
    html('$F(x,y) = e^{-(x^2+y^2)/2} \\cos(y) \\sin(x^2+y^2)$')
    show(plot)
```

x0 1.17935871743486

y0 0.788577154308618

order 4

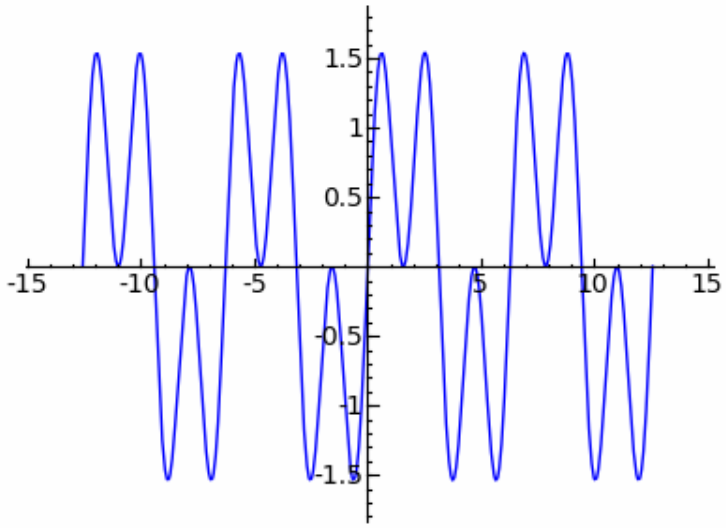
$F(x,y) = e^{-(x^2+y^2)/2}\cos(y)\sin(x^2+y^2)$

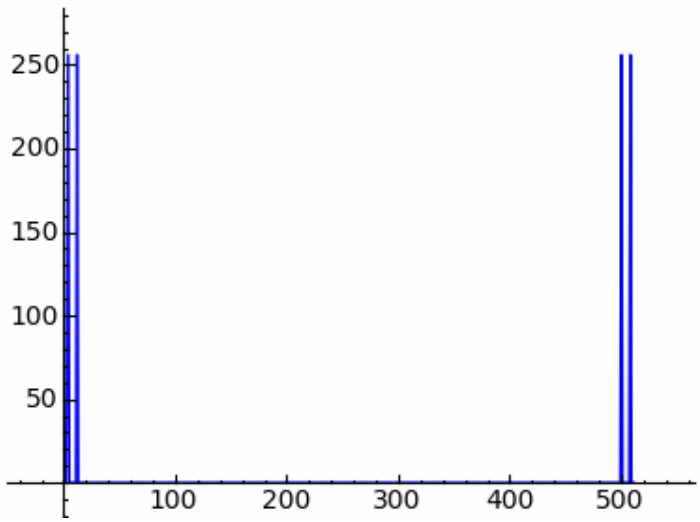


Дискретне перетворення Фур'є

```
import scipy.fftpack as Fourier
@interact
def discrete_fourier(f = input_box(default=sum([sin(k*x) for k in
range(1,5,2)]),label='f='), scale = slider(.1,20,.1,5,label='масштаб')):
    var('x')
    pbegin = -float(pi)*scale
    pend = float(pi)*scale
    html("<h3>Графік функції та її дискретне перетворення Фур'є </h3>")
    show(plot(f, pbegin, pend, plot_points = 512), figsize = [4,3])
    f_vals = [f(ind) for ind in srange(pbegin, pend,(pend-
pbegin)/512.0)]
    my_fft = Fourier.fft(f_vals)
    show(list_plot([abs(x) for x in my_fft], plotjoined=True), figsize
= [4,3])
```

Графік функції та її дискретне перетворення Фур'є





2. Алгебра

Базис Грьобнера

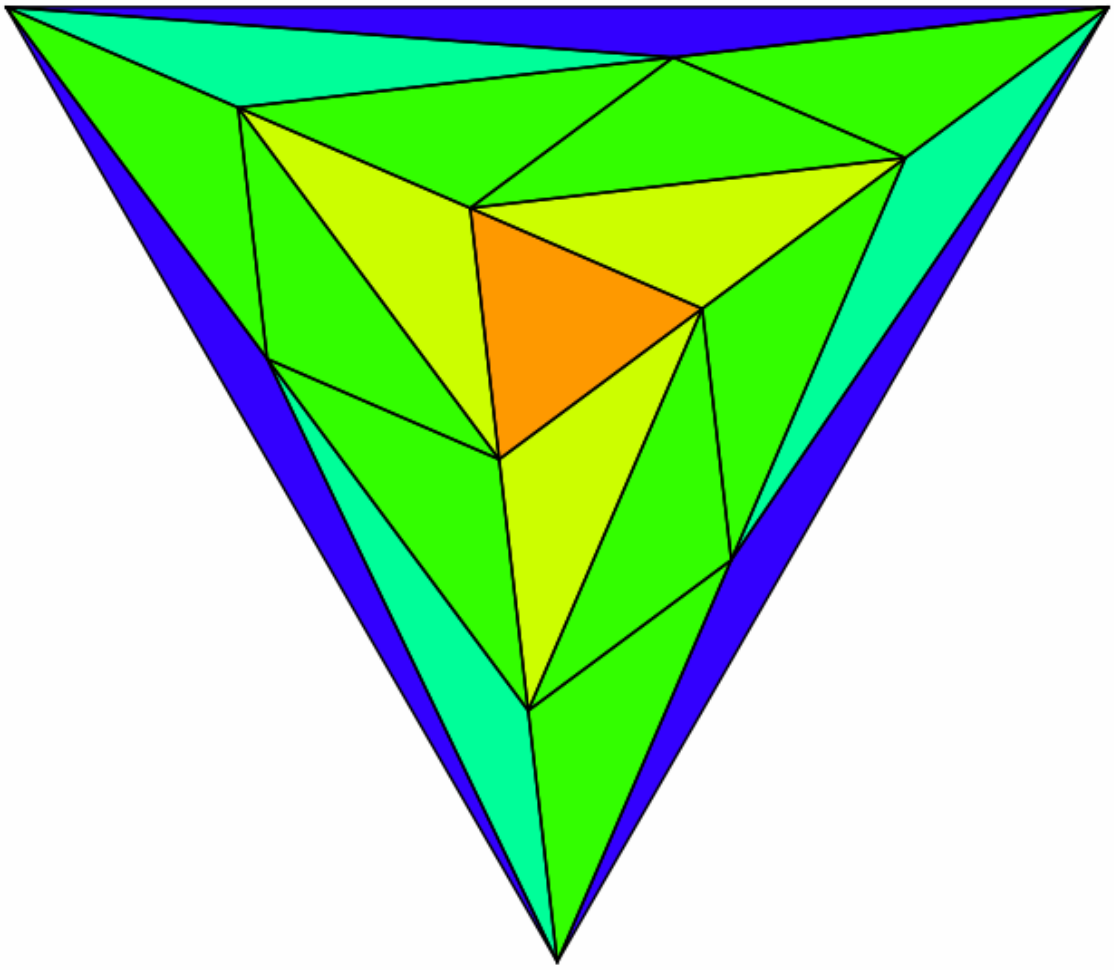
```
@interact
def gfan_browse(p1 = input_box('x^3+y^2',type = str, label='поліном 1:
'), p2 = input_box('y^3+z^2',type = str, label='поліном 2: '), p3 =
input_box('z^3+x^2',type = str, label='поліном 3: ')):
    R.<x,y,z> = PolynomialRing(QQ,3)
    i1 = ideal(R(p1),R(p2),R(p3))
    gf1 = i1.groebner_fan()
    testr = gf1.render()
    html('Базис Грьобнера ідеала алгебри многочленів K[' + str(p1) + ',
' + str(p2) + ', ' + str(p3))+ ']'
    show(testr, axes = False, figsize=[8,8*(3^(.5))/2])
```

поліном 1: x^3+y^2

поліном 2: y^3+z^2

поліном 3: z^3+x^2

Базис Грьобнера ідеала алгебри многочленів K[x^3+y^2, y^3+z^2, z^3+x^2]



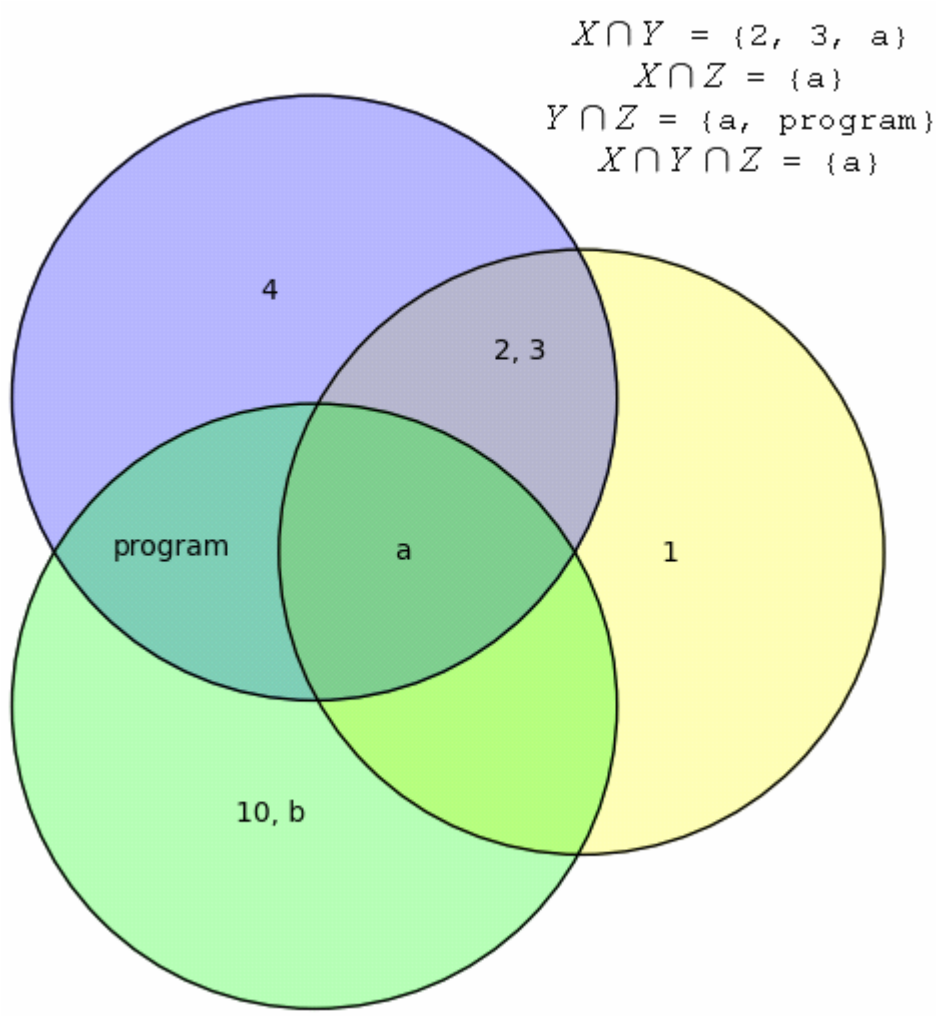
Діаграма Венна

```
def f(s, braces=True):
    t = ', '.join(sorted(list(s)))
    if braces: return '{' + t + '}'
    return t
def g(s): return set(str(s).replace(',',' ').split())

@interact
def _(X='1,2,3,a', Y='2,a,3,4,program', Z='a,b,10,program'):
    S = [g(X), g(Y), g(Z)]
    X,Y,Z = S
    XY = X.intersection(Y)
    XZ = X.intersection(Z)
    YZ = Y.intersection(Z)
    XYZ = XY.intersection(Z)
    html('<center>')
    html("$X \cap Y$ = %s"%f(XY))
    html("$X \cap Z$ = %s"%f(XZ))
    html("$Y \cap Z$ = %s"%f(YZ))
    html("$X \cap Y \cap Z$ = %s"%f(XYZ))
    html('</center>')
    centers = [(cos(n*2*pi/3), sin(n*2*pi/3)) for n in [0,1,2]]
    scale = 1.7
    clr = ['yellow', 'blue', 'green']
    G = Graphics()
    for i in range(len(S)):
        G += circle(centers[i], scale, rgbcolor=clr[i],
             fill=True, alpha=0.3)
    for i in range(len(S)):
        G += circle(centers[i], scale, rgbcolor='black')
    for i in range(len(S)):
        Z = set(S[i])
        for j in range(1,len(S)):
            Z = Z.difference(S[(i+j)%3])
        G += text(f(Z,braces=False),
(1.5*centers[i][0],1.7*centers[i][1]), rgbcolor='black')
    for i in range(len(S)):
        Z = set(S[i]).intersection(S[(i+1)%3]).difference(set(XYZ))
        C = (1.3*cos(i*2*pi/3 + pi/3), 1.3*sin(i*2*pi/3 + pi/3))
        G += text(f(Z,braces=False), C, rgbcolor='black')
    G += text(f(XYZ,braces=False), (0,0), rgbcolor='black')
    G.show(aspect_ratio=1, axes=False)
```

X 1,2,3,a

Y 2,a,3,4,program

Z a,b,10,program

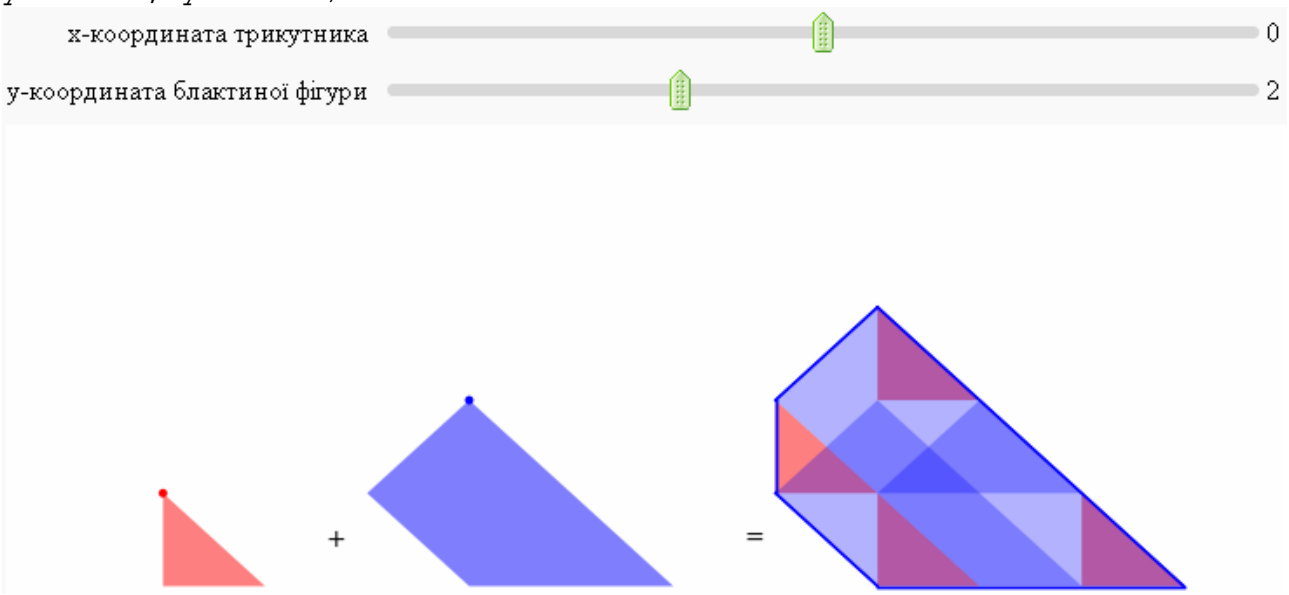


3. Геометрія

Сума Мінковського

```
def minkdemo(list1,list2):
    output = []
    for stuff1 in list1:
        for stuff2 in list2:
            temp = [stuff1[i] + stuff2[i] for i in range(len(stuff1))]
            output.append(temp)
    return output
@interact
def minksumvis(x1tri = slider(-1,1,1/10,0, label = 'x-координата
трикутника'), yb = slider(1,4,1/10,2, label = 'y-координата блактиної
фігури')):
    t_list = [[1,0],[x1tri,1],[0,0]]
    kite_list = [[3, 0], [1, 0], [0, 1], [1, yb]]
    triangle = polygon([[q[0]-6,q[1]] for q in t_list], alpha = .5,
rgbcolor = (1,0,0))
    t_vert = point([x1tri-6,1], rgbcolor = (1,0,0))
    b_vert = point([kite_list[3][0]-4,yb], rgbcolor = (0,0,1))
    kite = polygon([[q[0]-4,q[1]] for q in kite_list], alpha
= .5,rgbcolor = (0,0,1))
    p12 = minkdemo(t_list, kite_list)
    p12 = [[q[0],q[1]] for q in p12]
```

```
    p12poly = Polyhedron(p12)
    edge_lines = Graphics()
    verts = p12poly.vertices()
    for an_edge in p12poly.vertex_adjacencies():
        edge_lines = edge_lines + line([verts[an_edge[0]],
verts[an_edge[1][0]]])
        edge_lines = edge_lines + line([verts[an_edge[0]],
verts[an_edge[1][1]]])
    triangle_sum = Graphics()
    for vert in kite_list:
        temp_list = []
        for q in t_list:
            temp_list.append([q[i] + vert[i] for i in
range(len(t_list[0]))])
        triangle_sum = triangle_sum + polygon(temp_list, alpha = .5,
rgbcolor = (1,0,0))
    kite_sum = Graphics()
    for vert in t_list:
        temp_list = []
        for q in kite_list:
            temp_list.append([q[i] + vert[i] for i in
range(len(t_list[0]))])
        kite_sum = kite_sum + polygon(temp_list, alpha = .3,rgbcolor =
(0,0,1))
    labels = text('+', (-4.3,.5), rgbcolor = (0,0,0))
    labels = labels + text('=', (-.2,.5), rgbcolor = (0,0,0))
    show(labels + t_vert + b_vert+ triangle + kite + triangle_sum +
kite_sum + edge_lines, axes=False, figsize = [11.0*.7, 4*.7], xmin = -6,
ymin = 0, ymax = 4)
```

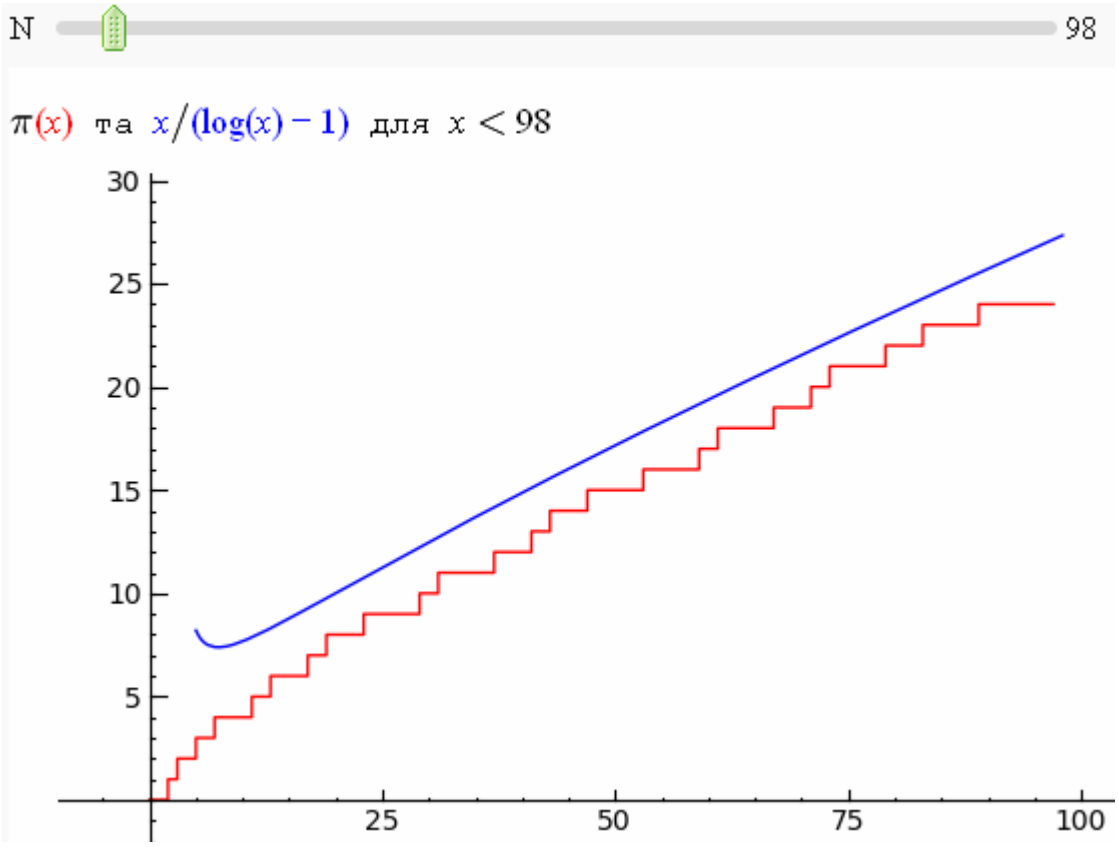


4. Теорія чисел

Розподіл простих чисел

```
@interact
def _(N=(100,(2..2000))):
    html("<font color='red'>$\pi(x)$</font> та <font
color='blue'>$x/(\log(x)-1)$</font> для $x < %s$"%N)
```

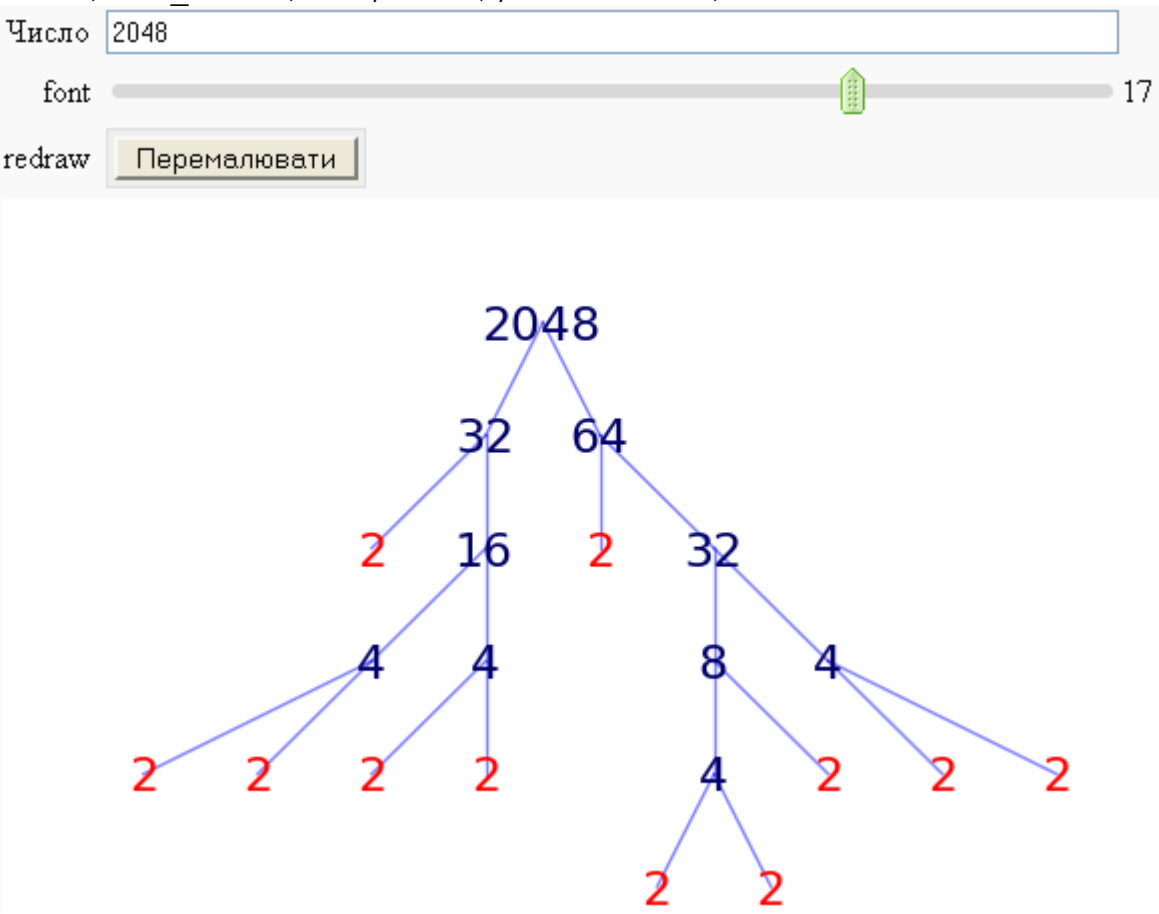
```
    show(plot(prime_pi, 0, N, rgbcolor='red') + plot(x/(log(x)-1), 5, N,
rgbcolor='blue'))
```



Факторизація цілих чисел

```
import random
def ftree(rows, v, i, F):
    if len(v) > 0:
        rows.append(v)
    w = []
    for i in range(len(v)):
        k, _, _ = v[i]
        if k is None or is_prime(k):
            w.append((None,None,None))
        else:
            d = random.choice(divisors(k)[1:-1])
            w.append((d,k,i))
            e = k//d
            if e == 1:
                w.append((None,None))
            else:
                w.append((e,k,i))
    if len(w) > len(v):
        ftree(rows, w, i+1, F)
def draw_ftree(rows,font):
    g = Graphics()
    for i in range(len(rows)):
        cur = rows[i]
        for j in range(len(cur)):
            e, f, k = cur[j]
            if not e is None:
```

```
                if is_prime(e):
                    c = (1,0,0)
                else:
                    c = (0,0,.4)
                g += text(str(e), (j*2-len(cur),-i), fontsize=font,
rgbcolor=c)
                if not k is None and not f is None:
                    g += line([(j*2-len(cur),-i), ((k*2)-len(rows[i-
1]),-i+1)], alpha=0.5)
    return g
@interact
def factor_tree(n=input_box(default=100, label="Число"), font=(10,
(8..20)), redraw=['Перемалювати']):
    n = Integer(n)
    rows = []
    v = [(n,None,0)]
    ftree(rows, v, 0, factor(n))
    show(draw_ftree(rows, font), axes=False)
```



Спіраль простих чисел (у полярній формі)

```
@interact
def polar_prime_spiral(start=input_box(default=1, label="Початкове
значення"), end=input_box(default=2000, label="Кінцеве значення"),
show_factors = checkbox(default=false, label="Показати співмножники"),
highlight_primes = checkbox(default=false, label="Виділяти прості"),
show_curves= checkbox(default=true, label="Показати криві"), n = 0):
    if start < 1 or end <=start: print "некоректне початкове або
кінцеве значення"
```

```
    if n > end: print "ПОПЕРЕДЖЕННЯ: n більше за кінцеве значення"
    def f(n):
        return (sqrt(n)*cos(2*pi*sqrt(n)), sqrt(n)*sin(2*pi*sqrt(n)))
    list =[]
    list2=[]
    if show_factors == false:
        for i in [start..end]:
            if i.is_pseudoprime(): list.append(f(i-start+1))
            else: list2.append(f(i-start+1))
        P = points(list)
        R = points(list2, alpha = .1) #Faded Composites
    else:
        for i in [start..end]:
            list.append(disk((f(i-start+1)),0.05*pow(2,len(factor(i))-
1), (0,2*pi)))
            if i.is_pseudoprime() and highlight_primes:
list2.append(f(i-start+1))
        P = plot(list)
        p_size = 5
        if not highlight_primes: list2 = [(f(n-start+1))]
        R=points(list2, hue = .1, pointsize = p_size)
    if n > 0:
        print 'n =', factor(n)
        p = 1
        W1 = disk((f(n-start+1)), p, (pi/6, 2*pi/6))
        W2 = disk((f(n-start+1)), p, (4*pi/6, 5*pi/6))
        W3 = disk((f(n-start+1)), p, (7*pi/6, 8*pi/6))
        W4 = disk((f(n-start+1)), p, (10*pi/6, 11*pi/6))
        Q = plot(W1+W2+W3+W4, alpha = .1)
        n=n-start+1
        if show_curves:
            begin_curve = 0
            t = var('t')
            a=1
            b=0
            if n > (floor(sqrt(n)))^2 and n <= (floor(sqrt(n)))^2 +
floor(sqrt(n)):
                c = -((floor(sqrt(n)))^2 - n)
                c2= -((floor(sqrt(n)))^2 + floor(sqrt(n)) - n)
            else:
                c = -((ceil(sqrt(n)))^2 - n)
                c2= -((floor(sqrt(n)))^2 + floor(sqrt(n)) - n)
            print 'Рожева крива:  n^2 +', c
            print 'Зелена крива: n^2 + n +', c2
            def g(m): return (a*m^2+b*m+c);
            def r(m) : return sqrt(g(m))
            def theta(m) : return r(m)- m*sqrt(a)
            S1 = parametric_plot(((r(t))*cos(2*pi*(theta(t))),
(r(t))*sin(2*pi*(theta(t)))), begin_curve, ceil(sqrt(end-start)),
rgbcolor=hue(0.8), thickness = .6)
            b=1
            c= c2;
```

```
            S2 = parametric_plot(((r(t))*cos(2*pi*(theta(t))),
(r(t))*sin(2*pi*(theta(t)))), begin_curve, ceil(sqrt(end-start)),
rgbcolor=hue(0.6), thickness = .6)
            show(R+P+S1+S2+Q, aspect_ratio = 1, axes = false)
        else: show(R+P+Q, aspect_ratio = 1, axes = false)
    else: show(R+P, aspect_ratio = 1, axes = false)
```

Початкове значення 1

Кінцеве значення 2000

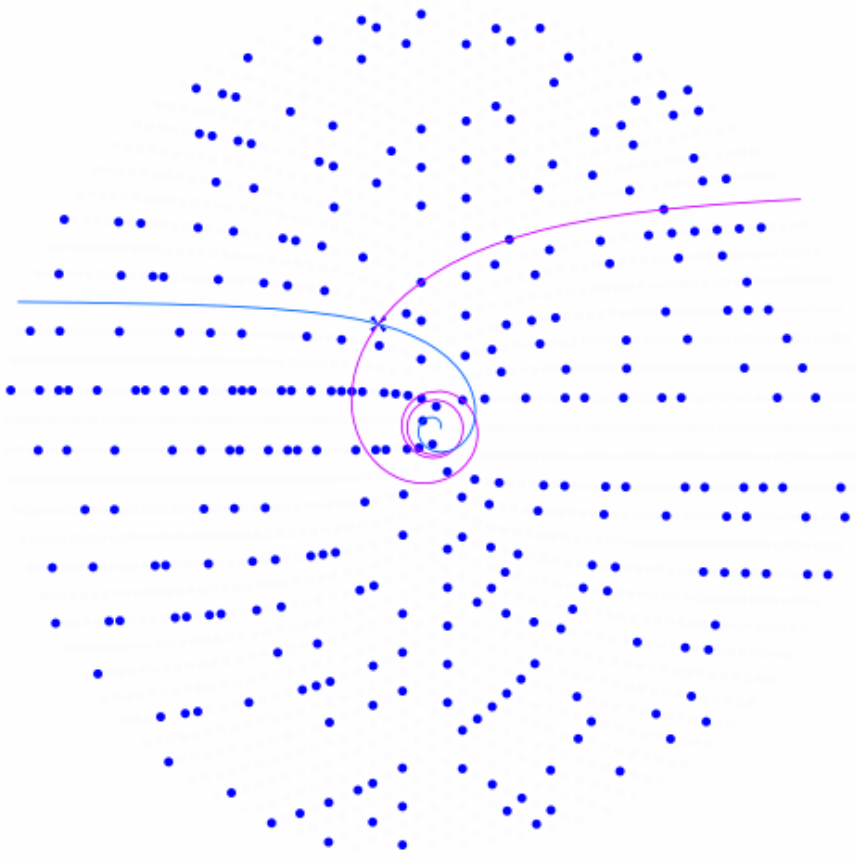
Показати співмножники ☐

Виділяти прості ☐

Показати криві ☑

n 152

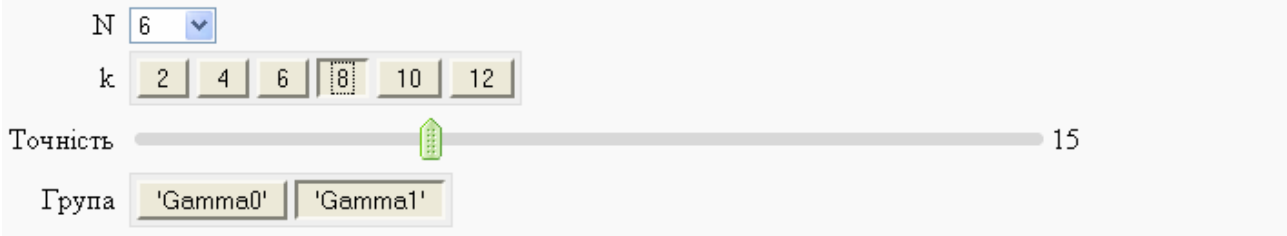
```
n = 2^3 * 19
Рожева крива:  n^2 + 8
Зелена крива: n^2 + n + -4
```



Обчислення модульних форм

```
j = 0
@interact
def _(N=[1..100], k=selector([2,4,..,12],nrows=1), prec=slider(3,40,
step_size=1,label='Точність'), group=selector(values=[(Gamma0,
'Gamma0'), (Gamma1, 'Gamma1')]), label='Група'):
```

```
    M = CuspForms(group(N),k)
    print j; global j; j += 1
    print M; print '\n'*3
    print "Обчислюємо базис...\n\n"
    if M.dimension() == 0:
         print "Простір має розмірність 0"
    else:
        prec = max(prec, M.dimension()+1)
        for f in M.basis():
             view(f.q_expansion(prec))
    print "\n\n\nОбчислення базису завершено."
```




```
4
Cuspidal subspace of dimension 5 of Modular Forms space of dimension 9
for Congruence Subgroup Gamma1(6) of weight 8 over Rational Field



Обчислюємо базис...
```

$$q + 204q^6 - 538q^7 + 96q^8 - 45q^9 - 800q^{10} + 3570q^{11} + 1728q^{12} - 2902q^{13} - 7968q^{14} + O(q^{15})$$
$$q^2 + 63q^6 - 252q^7 + 160q^8 + 216q^9 - 450q^{10} + 684q^{11} - 432q^{13} - 568q^{14} + O(q^{15})$$
$$q^3 - 8q^6 + 12q^9 + 64q^{12} + O(q^{15})$$
$$q^4 - 15q^6 + 42q^7 - 42q^8 - 36q^9 + 140q^{10} - 114q^{11} - 27q^{12} + 72q^{13} + 84q^{14} + O(q^{15})$$
$$q^5 - 6q^6 + 15q^7 - 16q^8 - 9q^9 + 48q^{10} - 49q^{11} + 18q^{13} + 48q^{14} + O(q^{15})$$

```
Обчислення базису завершено.
```

Характеристичний поліном та граф оператора Гекке

```
@interact
def f(N = prime_range(11,400),
      p = selector(prime_range(2,12),nrows=1),
      three_d = ("Тривимірний", False)):
    html("<h3> Характеристичний поліном та граф оператора Гекке: Рівень
%s, T_%s</h3>"%(N,p))
    S = SupersingularModule(N)
    T = S.hecke_matrix(p)
    G = Graph(T, multiedges=True, loops=not three_d)
    show(T.charpoly().factor())
    if three_d:
        show(G.plot3d(), aspect_ratio=[1,1,1])
    else:
        show(G.plot(),figsize=7)
```

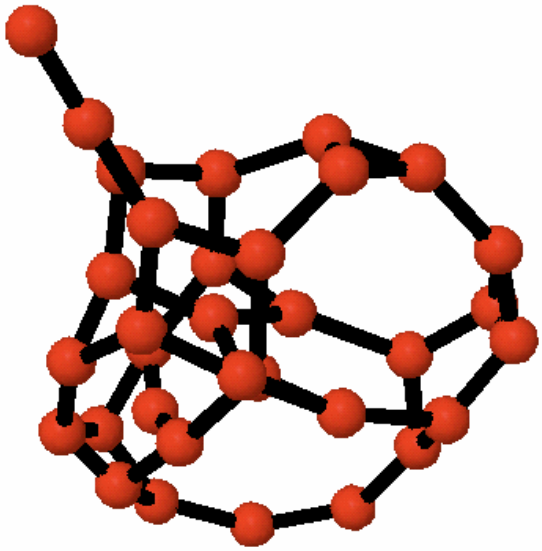
N 389

p 2 3 5 7 11

Тривимірний

Характеристичний поліном та граф оператора Гекке: Рівень 389, T_2

$$(x-3)\cdot(x+2)\cdot(x^2-2)\cdot(x^3-4x-2)\cdot(x^6+3x^5-2x^4-8x^3+2x^2+4x-1)\cdot(x^{20}-3x^{19}-29x^{18}+9$$



Таблиця квадратичних лишків

```
from numpy import array as narray
@interact
def quad_res_plot(
first_n_odd_primes=slider(2,200, step_size=1,label='Перші N непарних
простих'), display_size=selector([7..15], label='Розмір'):
    r = int(first_n_odd_primes)
    np = [nth_prime(i+2) for i in range(r)]
    leg = [[legendre_symbol(np[i], np[j]) for i in range(r)] for j in
range(r)]
    na = narray(leg)
    for i in range(r):
        for j in range(r):
            if na[i][j] == 1 and Mod((np[i]-1)*(np[j]-1)//4,2) == 0:
                na[i][j] = 2
    m = matrix(na)
    MP = matrix_plot(m, cmap='Oranges')
    for i in range(r):
        if np[-1] < 100:
            MP += text('%d'%nth_prime(i+2),(-.75,r-i-.5),
rgbcolor='black')
            MP += text('%d'%nth_prime(i+2), (i+.5,r+.5),
rgbcolor='black')
        if len(np) < 75:
```

```
            MP += line([(0,i),(r,i)], rgbcolor='black')
            MP += line([(i,0),(i,r)], rgbcolor='black')
    if np[-1] < 100:
        for i in range(r):
            for j in range(r):
                if m[j][i] == 0:
                    MP += text('0',(i+.5,r-j-.5),rgbcolor='black')
                elif m[j][i] == -1:
                    MP += text('N',(i+.5,r-j-.5),rgbcolor='black')
                elif m[j][i] == 1:
                    MP += text('A',(i+.5,r-j-.5),rgbcolor='black')
                elif m[j][i] == 2:
                    MP += text('S',(i+.5,r-j-.5),rgbcolor='black')
    MP += line([(0,r),(r,r)], rgbcolor='black')
    MP += line([(r,0),(r,r)], rgbcolor='black')
    MP += line([(0,0),(r,0)], rgbcolor='black')
    MP += line([(0,0),(0,r)], rgbcolor='black')
    if len(np) < 75:
        MP += text('q',(r/2,r+2), rgbcolor='black', fontsize=15)
        MP += text('p',(-2.5,r/2), rgbcolor='black', fontsize=15)
    html('Симетрія простих квадратичних лишків для перших %d непарних
простих.'%r)
    MP.show(axes=False, ymax=r, figsize=[display_size,display_size])
```

Перші N непарних простих 15

Розмір 7

Симетрія простих квадратичних лишків для перших 15 непарних простих.

q

p \ q	3	5	7	11	13	17	19	23	29	31	37	41	43	47	53
3	0	N	A	N	S	N	A	N	N	A	S	N	A	N	N
5	N	0	N	S	N	N	S	N	S	S	N	S	N	N	N
7	N	N	0	A	N	N	N	A	S	N	S	N	A	N	S
11	A	S	N	0	N	N	N	A	N	A	S	N	N	A	S
13	S	N	N	N	0	S	N	S	S	N	N	N	S	N	S
17	N	N	N	N	S	0	S	N	N	N	N	N	S	S	S
19	N	S	A	A	N	S	0	A	N	N	N	N	A	A	N
23	A	N	N	N	S	N	N	0	S	A	N	S	N	A	N
29	N	S	S	N	S	N	N	S	0	N	N	N	N	N	S
31	N	S	A	N	N	N	A	N	N	0	N	S	N	A	N
37	S	N	S	S	N	N	N	N	N	N	0	S	N	S	S
41	N	S	N	N	N	N	N	S	N	S	S	0	S	N	N
43	N	N	N	A	S	S	N	A	N	A	N	S	0	A	S
47	A	N	A	N	N	S	N	N	N	N	S	N	N	0	S
53	N	N	S	S	S	S	N	N	S	N	S	N	S	S	0

Таблиця кубічних лишків

```
def power_residue_symbol(alpha, p, m):
    if p.divides(alpha): return 0
    if not p.is_prime():
        return prod(power_residue_symbol(alpha,ell,m)^e
                for ell, e in p.factor())
    F = p.residue_field()
    N = p.norm()
    r = F(alpha)^((N-1)/m)
    k = p.number_field()
    for kr in k.roots_of_unity():
        if r == F(kr):
            return kr

def cubic_is_primary(n):
    g = n.gens_reduced()[0]
    a,b = g.polynomial().coefficients()
    if Mod(a,3)!=0 and Mod(b,3)==0:
        return True
    else:
        return False

from numpy import array as narray
@interact
def cubic_sym(n=(10..35), display_size=selector([7..15],
label='Розмір'):
    r = n
    m=3
    D.<w> = NumberField(x^2+x+1)
    it = D.primes_of_degree_one_iter()
    pp = []
    while len(pp) < r:
        k = it.next()
        if cubic_is_primary(k):
            pp.append(k)
    n = narray([ [ power_residue_symbol(pp[i].gens_reduced()[0], pp[j],
m) for i in range(r) ] for j in range(r) ])
    for i in range(r):
        for j in range(r):
            if n[i][j] == w:
                n[i][j] = int(-1)
            elif n[i][j] == w^2:
                n[i][j] = int(-2)
            elif n[i][j] == 1:
                n[i][j] = int(1)
    m = matrix(n)
    MP = matrix_plot(m,cmap="Blues")
    for i in range(r):
        MP += line([(0,i),(r,i)], rgbcolor='black')
        MP += line([(i,0),(i,r)], rgbcolor='black')
        for j in range(r):
            if m[i][j] == -2:
```

```
                MP += text('$\omega^2$',(i+.5,r-j-.5),rgbcolor='black')
            if m[i][j] == -1:
                MP += text('$\omega $',(i+.5,r-j-.5),rgbcolor='black')
            if m[i][j] == 0:
                MP += text('0',(i+.5,r-j-.5),rgbcolor='black')
            if m[i][j] == 1:
                MP += text('R',(i+.5,r-j-.5),rgbcolor='white')
    MP += line([(0,r),(r,r)], rgbcolor='black')
    MP += line([(r,0),(r,r)], rgbcolor='black')
    MP += line([(0,0),(r,0)], rgbcolor='black')
    MP += line([(0,0),(0,r)], rgbcolor='black')
    MP += text('$ \pi_1$',(r/2,r+2), rgbcolor='black', fontsize=25)
    MP += text('$ \pi_2$',(-2.5,r/2), rgbcolor='black', fontsize=25)
    html(' Симетрія простих кубічних лишків для перших ' \
          + '%d простих у $ \mathbf Z[\omega]$.'%r)
    MP.show(axes=False, figsize=[display_size,display_size])
```

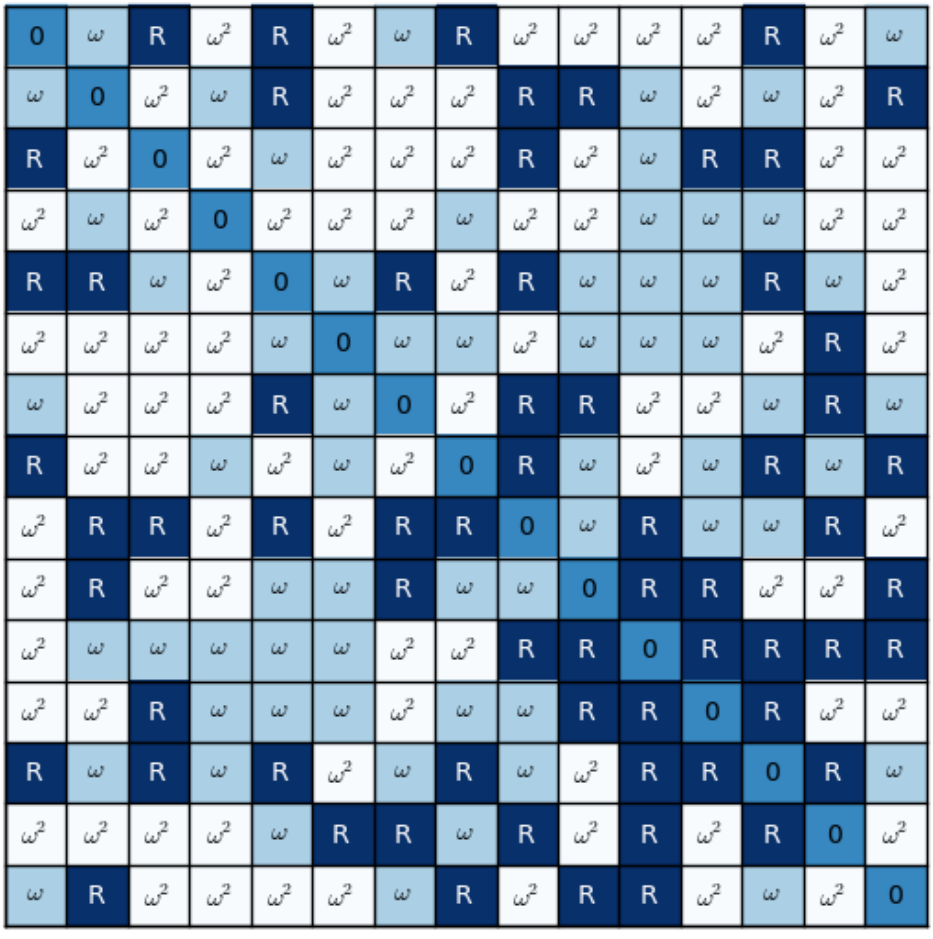
n 15

Розмір 7

Симетрія простих кубічних лишків для перших 15 простих у $\mathbf{Z}[\omega]$.

π_1

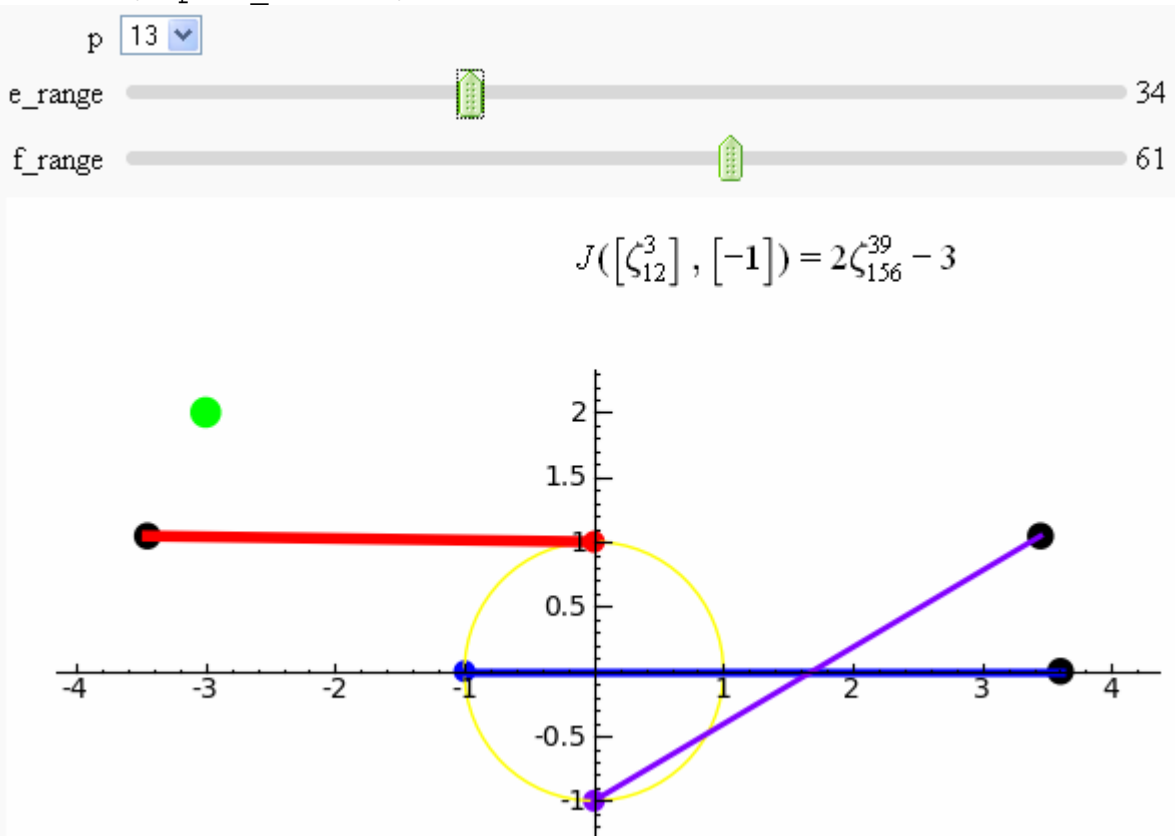
π_2



Суми Гауса та Якобі у комплексній площині

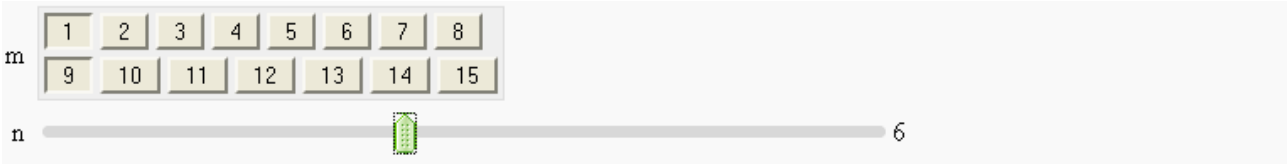
```
def jacobi_sum(e,f):
    if e.is_trivial() and f.is_trivial():
        return (e.parent()).order() + 1
    g = e*f
    if g.is_trivial():
        return -e(-1)
    elif not e.is_trivial() and not f.is_trivial():
        return e.gauss_sum()*f.gauss_sum()/g.gauss_sum()
    else:
        return 0

def latex2(e):
    return latex(list(e.values_on_gens()))

def jacobi_plot(p, e_index, f_index, with_text=True):
    G = DirichletGroup(p)
    e = G[e_index]
    f = G[f_index]
    ef = e*f
    js = jacobi_sum(e,f)
    e_gs = e.gauss_sum()
    f_gs = f.gauss_sum()
    ef_gs = (e*f).gauss_sum()

    f_pt = list(f.values_on_gens()[0].complex_embedding())
    e_pt = list(e.values_on_gens()[0].complex_embedding())
    ef_pt = list(ef.values_on_gens()[0].complex_embedding())
    f_gs_pt = list(f_gs.complex_embedding())
    e_gs_pt = list(e_gs.complex_embedding())
    ef_gs_pt = list(ef_gs.complex_embedding())
    try:
        js = int(js)
        js_pt = [CC(js)]
    except:
        js_pt = list(js.complex_embedding())
    S = circle((0,0),1,rgbcolor='yellow')  \
    + line([e_pt,e_gs_pt], rgbcolor='red', thickness=4) \
    + line([f_pt,f_gs_pt], rgbcolor='blue', thickness=3) \
    + line([ef_pt,ef_gs_pt], rgbcolor='purple',thickness=2) \
    + point(e_pt,pointsize=50, rgbcolor='red')  \
    + point(f_pt,pointsize=50, rgbcolor='blue') \
    + point(ef_pt,pointsize=50,rgbcolor='purple') \
    + point(f_gs_pt,pointsize=75, rgbcolor='black') \
    + point(e_gs_pt,pointsize=75, rgbcolor='black') \
    + point(ef_gs_pt,pointsize=75, rgbcolor='black') \
    + point(js_pt,pointsize=100,rgbcolor='green')
    if with_text:
        S += text('$J(%s,%s) = %s$'%(latex2(e),latex2(f),latex(js)), \
            (3,2.5),fontsize=15, rgbcolor='black')
    else:
        html('$$J(%s,%s) = %s$$'%(latex2(e),latex2(f),latex(js)))
```

```
    return S

@interact
def single_jacobi_plot(p=prime_range(3,100), e_range=(0..100),
f_range=(0..100)):
    e_index = floor((p-2)*e_range/100)
    f_index = floor((p-2)*f_range/100)
    S = jacobi_plot(p,e_index,f_index,with_text=False)
    S.show(aspect_ratio=1)
```



Узагальнені числа Бернуллі

```
@interact
def _(m=selector([1..15],nrows=2), n=(7,(3..10))):
    G = DirichletGroup(m)
    s = "<h3>Перші n=%s чисел Бернуллі та пов'язані з ними символи за
модулем m=%s</h3>"%(n,m)
    s += '<table border=1>'
    s += '<tr bgcolor="#edcc9c"><td
align=center>$\\chi$</td><td>Найменший модуль</td>' + \
          ''.join('<td>$B_{%s,\chi}$</td>'%k for k in [1..n])+ '</tr>'
    for eps in G.list():
        v = ''.join(['<td align=center
bgcolor="#efe5cd">$%s$</td>'%latex(eps.bernoulli(k)) for k in [1..n]])
        s += '<tr><td bgcolor="#edcc9c">%s</td><td bgcolor="#efe5cd"
align=center>%s</td>%s</tr>\n'%(eps, eps.conductor(), v)
    s += '</table>'
    html(s)
```



Перші n=6 чисел Бернуллі та пов'язані з ними символи за модулем m=9

χ	Найменший модуль	$B_{1,\chi}$	$B_{2,\chi}$	$B_{3,\chi}$	$B_{4,\chi}$	$B_{5,\chi}$	$B_{6,\chi}$
[1]	1	0	$-\frac{1}{3}$	0	$\frac{13}{15}$	0	$-\frac{121}{21}$
[zeta6]	9	$-\frac{2}{3}\zeta_6-\frac{2}{3}$	0	$10\zeta_6+4$	0	$-\frac{1270}{3}\zeta_6-\frac{340}{3}$	0
[zeta6 - 1]	9	0	$\frac{4}{3}\zeta_6+\frac{4}{3}$	0	$-\frac{136}{3}\zeta_6-\frac{88}{3}$	0	$3004\zeta_6+1684$
[-1]	3	$-\frac{1}{3}$	0	$\frac{2}{3}$	0	$-\frac{10}{3}$	0
[-zeta6]	9	0	$-\frac{4}{3}\zeta_6+\frac{8}{3}$	0	$\frac{136}{3}\zeta_6-\frac{224}{3}$	0	$-3004\zeta_6+4688$
[-zeta6 + 1]	9	$\frac{2}{3}\zeta_6-\frac{4}{3}$	0	$-10\zeta_6+14$	0	$\frac{1270}{3}\zeta_6-\frac{1610}{3}$	0

Алгоритм обміну ключами Діффі-Геллмана

```
@interact
def diffie_hellman(bits=slider(8, 513, 4, 8, 'Кількість бітів', False),
    button=selector(["Показати новий приклад"],label='',buttons=True)):
    maxp = 2 ^ bits
    p = random_prime(maxp)
    k = GF(p)
    if bits > 100:
        g = k(2)
    else:
        g = k.multiplicative_generator()
    a = ZZ.random_element(10, maxp)
    b = ZZ.random_element(10, maxp)
    print """
<html>
<style>
.gamodp, .gbmodp {
color:#000;
padding:5px
}
.gamodp {
background:#846FD8
}
.gbmodp {
background:#FFFC73
}
.dhsame {
color:#000;
font-weight:bold
}
</style>
<h2 style="color:#000;font-family:Arial, Helvetica, sans-serif">%s-
бітний алгоритм обміну ключами Діффі-Геллмана </h2>
<ol style="color:#000;font-family:Arial, Helvetica, sans-serif">
```

```
<li>Аліса та Боб домовляються використовувати просте число p = %s та
основу g = %s.</li>
<li>Аліса вибирає секретне ціле число a = %s та відсилає Бобу (<span
class="gamodp">g<sup>a</sup> mod p</span>):<br/>%s<sup>%s</sup> mod %s
= <span class="gamodp">%s</span>.</li>
<li>Боб вибирає секретне ціле число b=%s та відсилає Алісі (<span
class="gbmodp">g<sup>b</sup> mod p</span>):<br/>%s<sup>%s</sup> mod %s
= <span class="gbmodp">%s</span>.</li>
<li>Аліса обчислює (<span class="gbmodp">g<sup>b</sup> mod
p</span>)<sup>a</sup> mod p:<br/>%s<sup>%s</sup> mod %s = <span
class="dhsame">%s</span>.</li>
<li>Боб обчислює (<span class="gamodp">g<sup>a</sup> mod
p</span>)<sup>b</sup> mod p:<br/>%s<sup>%s</sup> mod %s = <span
class="dhsame">%s</span>.</li>
</ol></html>
    """ % (bits, p, g, a, g, a, p, (g^a), b, g, b, p, (g^b), (g^b), a,
p, (g^ b)^a, g^a, b, p, (g^a)^b)
```

Кількість бітів

Показати новий приклад

64-бітний алгоритм обміну ключами Діффі-Геллмана

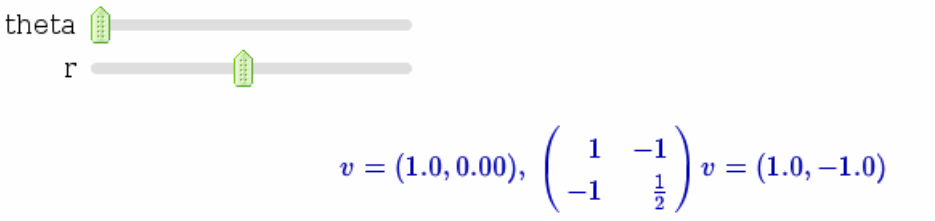
1. Аліса та Боб домовляються використовувати просте число p = 610895326348522849 та основу g = 23.
2. Аліса вибирає секретне ціле число a = 15822307641407439906 та відсилає Бобу ($g^a \bmod p$):
   $23^{15822307641407439906} \bmod 610895326348522849 =$ **138390022856187633**.
3. Боб вибирає секретне ціле число b=3485684563149628115 та відсилає Алісі ($g^b \bmod p$):
   $23^{3485684563149628115} \bmod 610895326348522849 =$ **549161498172059778**.
4. Аліса обчислює ($g^b \bmod p$)$^a \bmod p$:
   $549161498172059778^{15822307641407439906} \bmod 610895326348522849 =$ **168927368256737844**.
5. Боб обчислює ($g^a \bmod p$)$^b \bmod p$:
   $138390022856187633^{3485684563149628115} \bmod 610895326348522849 =$ **168927368256737844**.

5. Лінійна алгебра

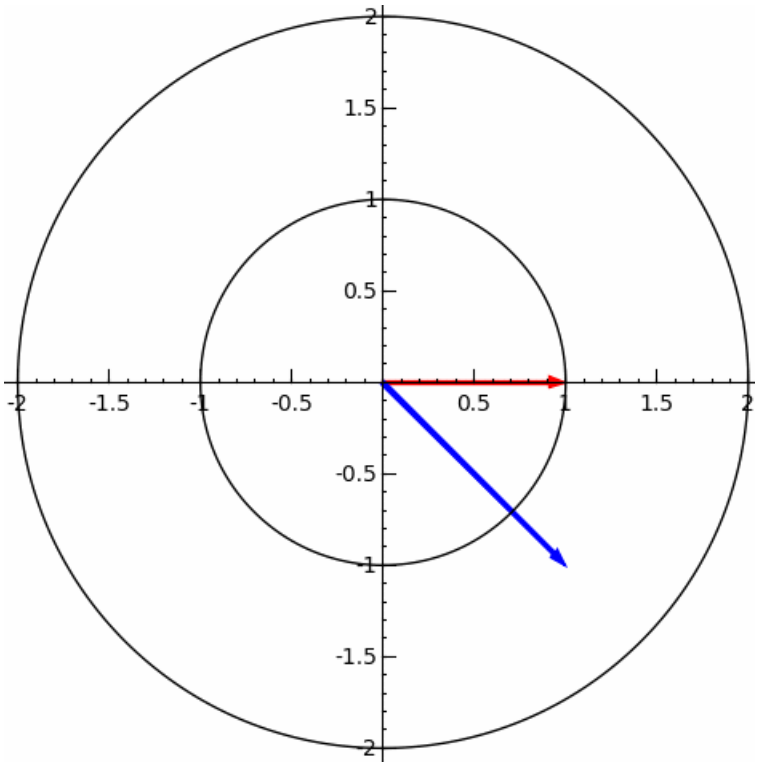
Лінійне перетворення

```
@interact
def linear_transformation(theta=slider(0, 2*pi, .1), r=slider(0.1,
2, .1, default=1)):
    A=matrix([[1,-1],[-1,1/2]])
    v=vector([r*cos(theta), r*sin(theta)])
    w = A*v
    circles = sum([circle((0,0), radius=i, rgbcolor=(0,0,0)) for i in
```

```
[1..2]])
    print jsmath("v = %s,\; %s v=%s"%(v.n(4),latex(A),w.n(4)))
    show(v.plot(rgbcolor=(1,0,0))+w.plot(rgbcolor=(0,0,1))+circles,
aspect_ratio=1)
```



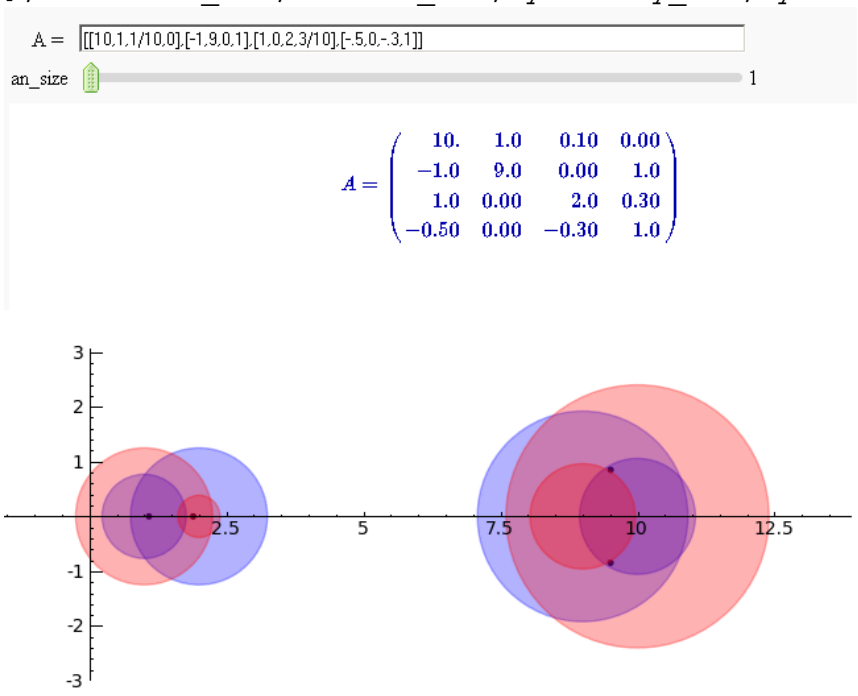
$$v = (1.0, 0.00),\ \begin{pmatrix} 1 & -1 \\ -1 & \frac{1}{2} \end{pmatrix} v = (1.0, -1.0)$$



Теорема Гершгоріна

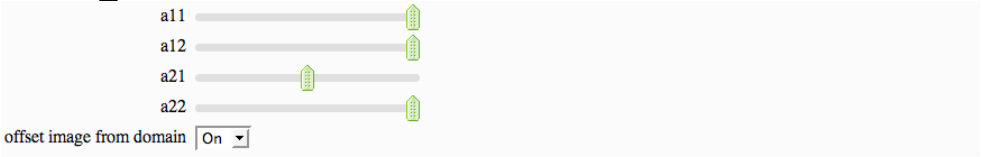
```
from scipy import linalg
@interact
def Gerschgorin(Ain = input_box(default='[[10,1,1/10,0],[-
1,9,0,1],[1,0,2,3/10],[-.5,0,-.3,1]]', type = str, label = 'A = '),
an_size = slider(1,100,1,1.0)):
    A = sage_eval(Ain)
    size = len(A)
    print jsmath('A = ' + latex(matrix(RealField(10),A)))
    A = matrix(RealField(10),A)
    B = [[0 for i in range(size)] for j in range(size)]
    for i in range(size):
        B[i][i] = A[i][i]
    B = matrix(B)
    frames = []
    centers = [(real(q),imag(q)) for q in [A[i][i] for i in
range(size)]]
    radii_row = [sum([abs(A[i][j]) for j in range(i)+range(i+1,size)])
for i in range(size)]
    radii_col = [sum([abs(A[j][i]) for j in range(i)+range(i+1,size)])
```

```
for i in range(size)]
    x_min = min([centers[i][0]-radii_row[i] for i in
range(size)]+[centers[i][0]-radii_col[i] for i in range(size)])
    x_max = max([centers[i][0]+radii_row[i] for i in
range(size)]+[centers[i][0]+radii_col[i] for i in range(size)])
    y_min = min([centers[i][1]-radii_row[i] for i in
range(size)]+[centers[i][1]-radii_col[i] for i in range(size)])
    y_max = max([centers[i][1]+radii_row[i] for i in
range(size)]+[centers[i][1]+radii_col[i] for i in range(size)])
    if an_size > 1:
        t_range= srange(0,1+1/an_size,1/an_size)
    else:
        t_range = [1]
    for t in t_range:
        C = t*A + (1-t)*B
        eigs = [CDF(x) for x in linalg.eigvals(C.numpy())]
        eigpoints = points([(real(q),imag(q)) for q in eigs],pointsize
= 10, rgbcolor = (0,0,0))
        centers = [(real(q),imag(q)) for q in [A[i][i] for i in
range(size)]]
        radii_row = [sum([abs(C[i][j]) for j in
range(i)+range(i+1,size)]) for i in range(size)]
        radii_col = [sum([abs(C[j][i]) for j in
range(i)+range(i+1,size)]) for i in range(size)]
        scale = max([(x_max-x_min),(y_max-y_min)])
        scale = 7/scale
        row_circles = sum([circle(centers[i],radii_row[i],fill=True,
alpha = .3) for i in range(size)])
        col_circles = sum([circle(centers[i],radii_col[i],fill=True,
rgbcolor = (1,0,0), alpha = .3) for i in range(size)])
        ft = eigpoints+row_circles+col_circles
        frames.append(ft)
    show(animate(frames,figsize = [(x_max-x_min)*scale,(y_max-
y_min)*scale], xmin = x_min, xmax=x_max, ymin = y_min, ymax = y_max))
```



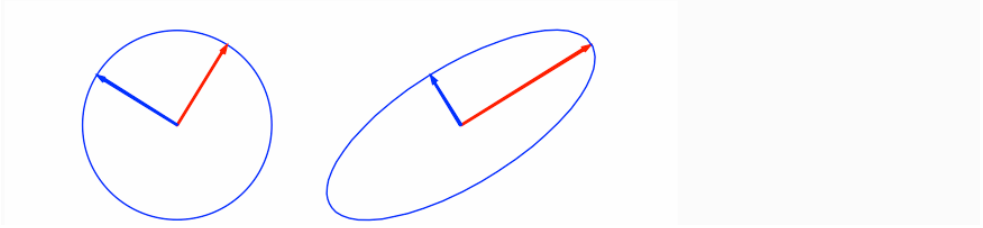
Сингулярний розклад матриці

```
import scipy.linalg as lin
var('t')
def rotell(sig,umat,t,offset=0):
    temp = matrix(umat)*matrix(2,1,[sig[0]*cos(t),sig[1]*sin(t)])
    return [offset+temp[0][0],temp[1][0]]
@interact
def svd_vis(a11=slider(-1,1,.05,1),a12=slider(-1,1,.05,1),a21=slider(-
1,1,.05,0),a22=slider(-1,1,.05,1),ofs=
selector(['Off','On'],label='offset image from domain')):
    rf_low = RealField(12)
    my_mat = matrix(rf_low,2,2,[a11,a12,a21,a22])
    u,s,vh = lin.svd(my_mat.numpy())
    if ofs == 'On':
        offset = 3
        fsize = 6
        colors = [(1,0,0),(0,0,1),(1,0,0),(0,0,1)]
    else:
        offset = 0
        fsize = 5
        colors = [(1,0,0),(0,0,1),(.7,.2,0),(0,.3,.7)]
    vvects = sum([arrow([0,0],matrix(vh).row(i),rgbcolor = colors[i])
for i in (0,1)])
    uvects = Graphics()
    for i in (0,1):
        if s[i] != 0: uvects +=
arrow([offset,0],vector([offset,0])+matrix(s*u).column(i),rgbcolor =
colors[i+2])
    print jsmath("A = %s  = %s %s %s"%(latex(my_mat),
latex(matrix(rf_low,u.tolist())),
latex(matrix(rf_low,2,2,[s[0],0,0,s[1]])),
latex(matrix(rf_low,vh.tolist()))))
    image_ell = parametric_plot(rotell(s,u,t, offset),0,2*pi)
    graph_stuff=circle((0,0),1)+image_ell+vvects+uvects
    graph_stuff.set_aspect_ratio(1)
    show(graph_stuff,frame = False,axes=False,figsize=[fsize,fsize])
```



Singular value decomposition: image of the unit circle and the singular vectors

$$A = \begin{pmatrix} 1.00 & 1.00 \\ 0.000 & 1.00 \end{pmatrix} = \begin{pmatrix} 0.851 & -0.526 \\ 0.526 & 0.851 \end{pmatrix} \begin{pmatrix} 1.62 & 0.000 \\ 0.000 & 0.618 \end{pmatrix} \begin{pmatrix} 0.526 & 0.851 \\ -0.851 & 0.526 \end{pmatrix}$$



```
def GS_classic(a_list):
    if type(a_list) != list:
        cols = a_list.cols()
        a_list = [x for x in cols]
    indices = range(len(a_list))
    q = []
    r = [[0 for i in indices] for j in indices]
    v = [a_list[i].copy() for i in indices]
    for i in indices:
        for j in range(0,i):
            r[j][i] = q[j].inner_product(a_list[i])
            v[i] = v[i] - r[j][i]*q[j]
        r[i][i] = (v[i]*v[i])^(1/2)
        q.append(v[i]/r[i][i])
    q = matrix([q[i] for i in indices]).transpose()
    return q, matrix(r)
def GS_modern(a_list):
    if type(a_list) != list:
        cols = a_list.cols()
        a_list = [x for x in cols]
    indices = range(len(a_list))
    q = []
    r = [[0 for i in indices] for j in indices]
    v = [a_list[i].copy() for i in indices]
    for i in indices:
        r[i][i] = v[i].norm(2)
        q.append(v[i]/r[i][i])
        for j in range(i+1, len(indices)):
            r[i][j] = q[i].inner_product(v[j])
            v[j] = v[j] - r[i][j]*q[i]
    q = matrix([q[i] for i in indices]).transpose()
    return q, matrix(r)
@interact
def gstest(precision = slider(range(3,53), default = 10,
label='точність'), a1 = input_box([1,1/1000,1/1000]), a2 =
input_box([1,1/1000,0]), a3 = input_box([1,0,1/1000])):
    myR = RealField(precision)
    displayR = RealField(5)
    html('точність (у бітах): ' + str(precision) + '<br>')
    A = matrix([a1,a2,a3])
    A = [vector(myR,x) for x in A]
    qn, rn = GS_classic(A)
    qb, rb = GS_modern(A)
    html('Класичний алгоритм Грама-Шмідта:')
    show(matrix(displayR,qn))
    html('Стабільний алгоритм Грама-Шмідта:')
    show(matrix(displayR,qb))
```


```
точність (у бітах): 10

Класичний алгоритм Грама-Шмідта:
```

$$\begin{pmatrix} 1.0 & 0.00 & 0.00 \\ 0.00098 & 0.00 & -0.72 \\ 0.00098 & -1.0 & -0.72 \end{pmatrix}$$

```
Стабільний алгоритм Грама-Шмідта:
```

$$\begin{pmatrix} 1.0 & 0.00 & 0.00 \\ 0.00098 & 0.00 & -1.0 \\ 0.00098 & -1.0 & 0.00 \end{pmatrix}$$

6. Теорія графів

Побудова графів

```
grs = ['BalancedTree', 'BullGraph', 'ChvatalGraph', 'CirculantGraph',
'CircularLadderGraph', 'ClawGraph', 'CompleteBipartiteGraph',
'CompleteGraph', 'CubeGraph', 'CycleGraph', 'DegreeSequence',
'DegreeSequenceConfigurationModel', 'DegreeSequenceExpected',
'DegreeSequenceTree', 'DesarguesGraph', 'DiamondGraph',
'DodecahedralGraph', 'DorogovtsevGoltsevMendesGraph', 'EmptyGraph',
'FlowerSnark', 'FruchtGraph', 'Grid2dGraph', 'GridGraph',
'HeawoodGraph', 'HexahedralGraph', 'HoffmanSingletonGraph',
'HouseGraph', 'HouseXGraph', 'IcosahedralGraph', 'KrackhardtKiteGraph',
'LCFGraph', 'LadderGraph', 'LollipopGraph', 'MoebiusKantorGraph',
'OctahedralGraph', 'PappusGraph', 'PathGraph', 'PetersenGraph',
'RandomBarabasiAlbert', 'RandomGNM', 'RandomGNP', 'RandomHolmeKim',
'RandomLobster', 'RandomNewmanWattsStrogatz', 'RandomRegular',
'RandomTreePowerlaw', 'StarGraph', 'TetrahedralGraph', 'ThomsenGraph',
'WheelGraph']
examples = {}
for g in grs:
    docs = eval('graphs.' + g + '.__doc__')
    for docline in docs.split('\n'):
        ex_loc = docline.find('graphs.' + g)
        if ex_loc != -1:
            end_paren_loc = docline[ex_loc:].find(')')
            ex_str = docline[ex_loc:end_paren_loc+ex_loc+1]
            ex_str = ex_str.replace('i+','2+')
            ex_str = ex_str.replace('(i','(4')
            break
```

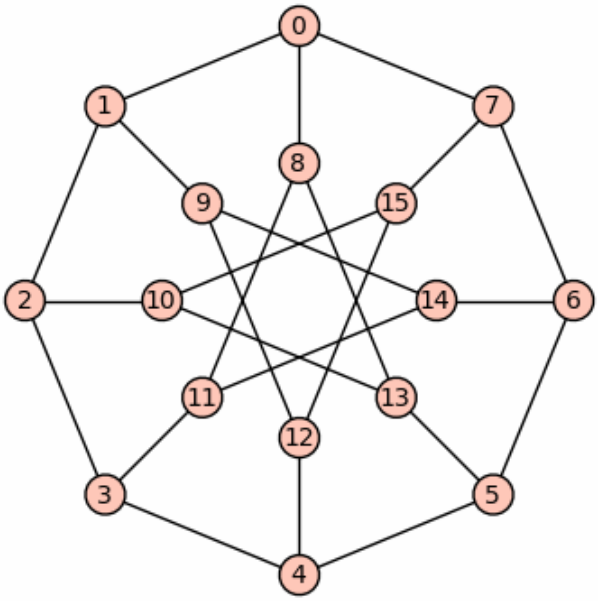
```
    try:
        gt2 = eval(ex_str)
        examples[g] = ex_str
    except:
        grs.remove(g)
@interact
def graph_browser(graph_name = selector(grs, label = "Тип графа:"),
newargs = input_box('',type=str, label='кортежи аргументів'),
output_type = selector(['2D','3D'], default='2D',
label='візуалізація')):
    base_g_str = 'graphs.' + graph_name
    docs = eval(base_g_str + '.__doc__')
    doc_ex_loc = docs.find('EXAMPLE')
    if docs.find('PLOTTING') != -1:
        doc_ex_loc = min(doc_ex_loc, docs.find('PLOTTING'))
    print docs[0:doc_ex_loc].replace('\n        ','\n')
    if newargs != '':
        try:
            t_graph = eval(base_g_str + newargs)
        except:
            print "Невірний аргумент, використовуємо за замовчанням"
            t_graph = eval(examples[graph_name])
    else:
        t_graph = eval(examples[graph_name])
    if output_type == '2D': show(t_graph)
    if output_type == '3D': t_graph.show3d()
```

Тип графу: MoebiusKantorGraph

кортежи аргументів

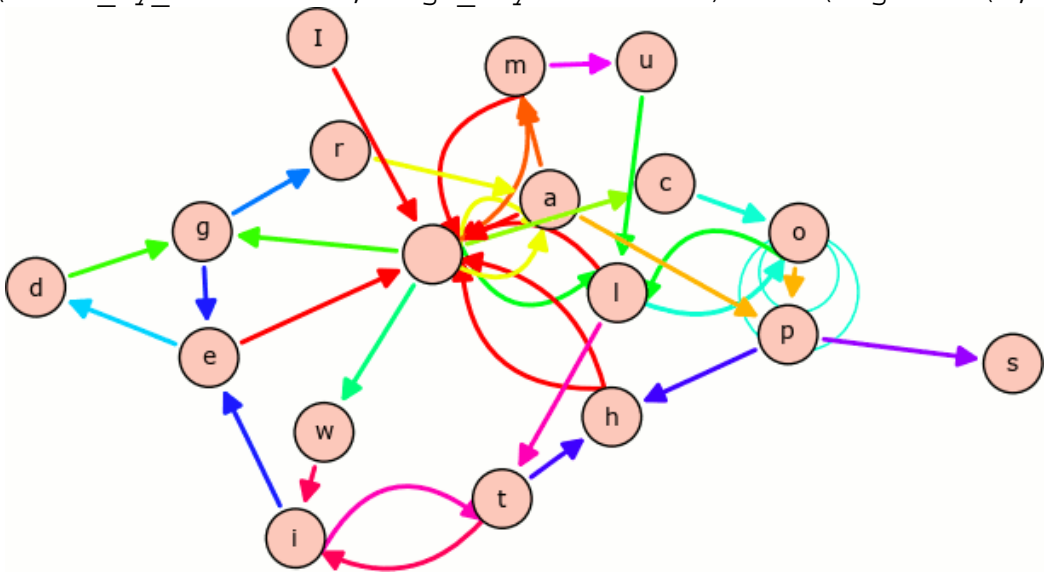
візуалізація 2D

```
Returns a Moebius-Kantor Graph.

A Moebius-Kantor graph is a cubic symmetric graph. (See also the
Heawood graph). It has 16 nodes and 24 edges. It is nonplanar and
Hamiltonian. It has diameter = 4, girth = 6, and chromatic number =
2. It is identical to the Generalized Petersen graph, P[8,3].
```



Граф з циклами

```
stnc = 'I am a cool multiedge graph with loops'
g = DiGraph({}, loops=True, multiedges=True)
for a,b in [(stnc[i], stnc[i+1]) for i in xrange(len(stnc)-1)]:
   g.add_edge(a, b, b)
g.plot(color_by_label=True, edge_style='solid').show(figsize=(8,8))
```



Автоморфізм графів

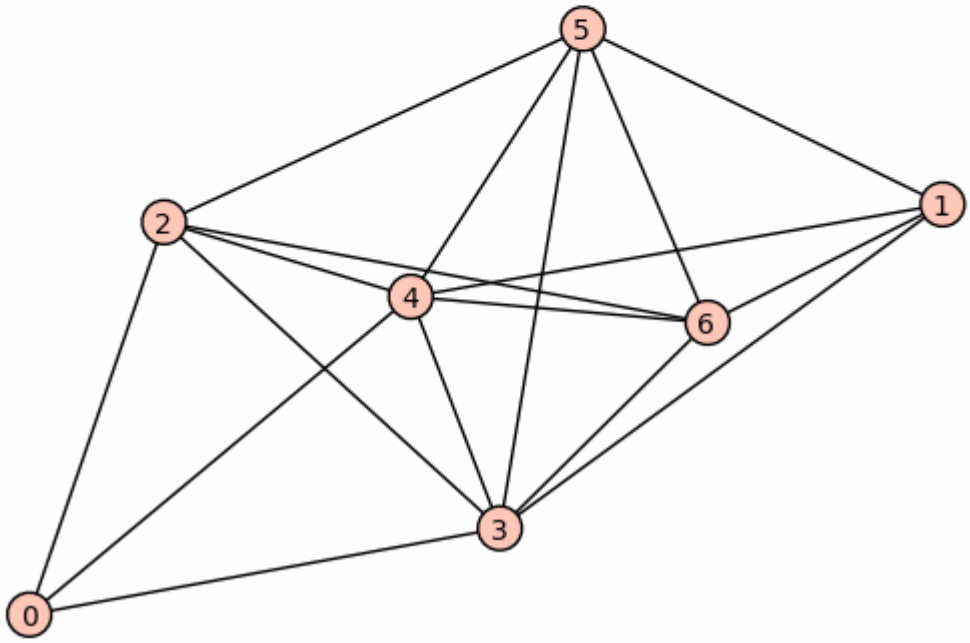
```
@interact
def _(graph=selector(['CycleGraph', 'CubeGraph', 'RandomGNP'],
label='граф'), n=selector([1..10], nrows=1), p=selector([10,20,..,100],
nrows=1)):
    print graph
    if graph == 'CycleGraph':
       print "n = %s (кількість вершин)"%n
       G = graphs.CycleGraph(n)
    elif graph == 'CubeGraph':
       if n > 8:
           print "n зменшене до 8"
           n = 8
       print "n = %s (розмірність)"%n
       G = graphs.CubeGraph(n)
    elif graph == 'RandomGNP':
       print "n = %s (кількість вершин)"%n
       print "p = %s%% (ймовірність наявності ребер в мережі)"%p
       G = graphs.RandomGNP(n, p/100.0)
    print G.automorphism_group()
    show(plot(G))
```

граф CycleGraph | CubeGraph | RandomGNP

n 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10

p 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100

```
RandomGNP
n = 7 (кількість вершин)
p = 70% (ймовірність наявності ребер в
мережі)
Permutation Group with generators [(5,6), (3,4)]
```

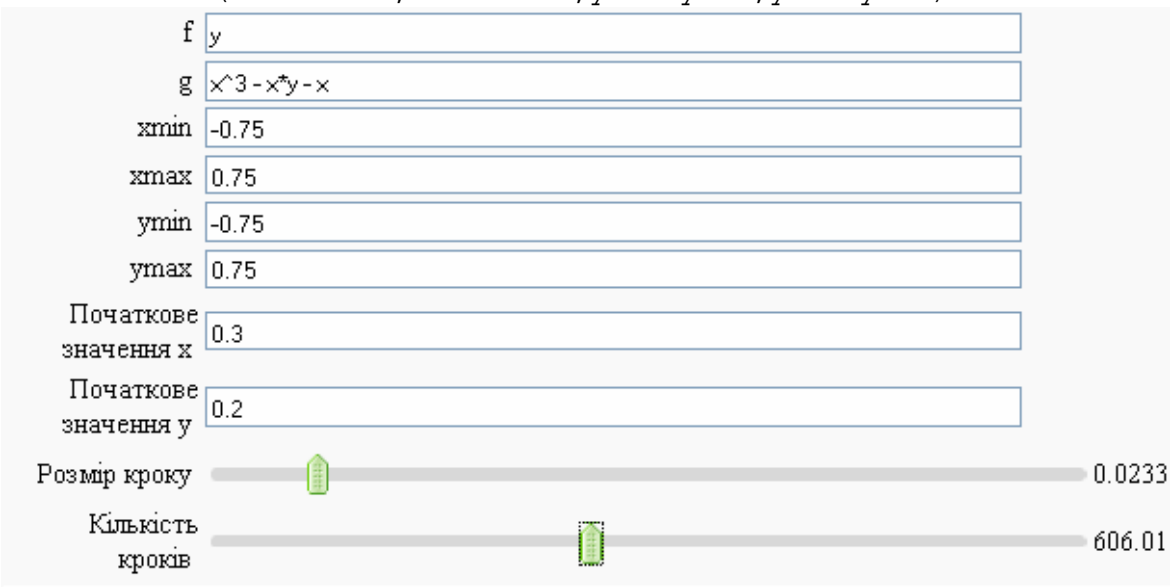


7. Диференціальні рівняння

Поле напрямів для методу Ейлера

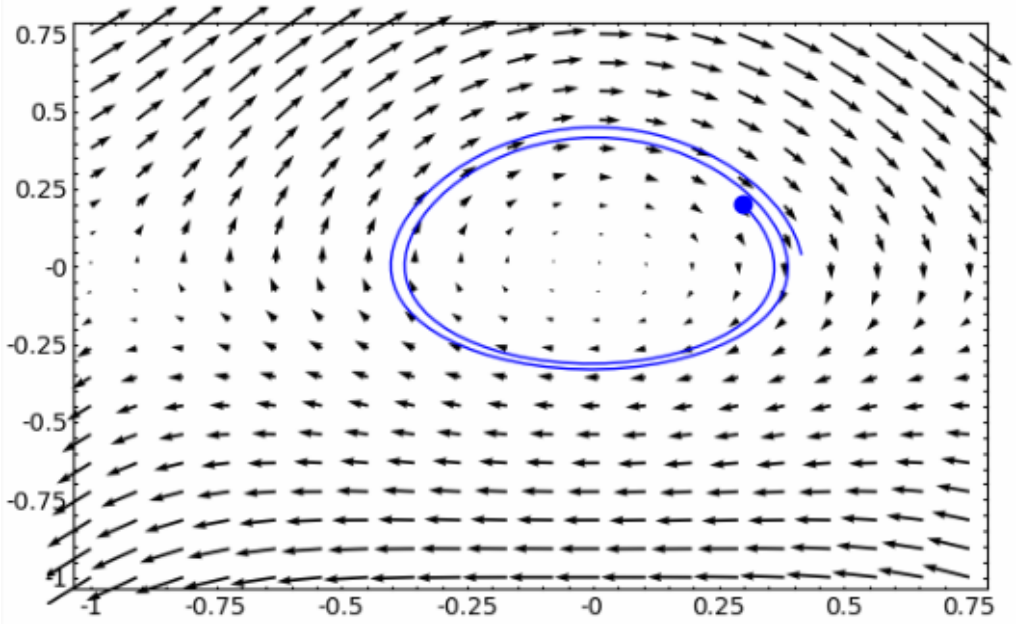
```
x,y = var('x,y')
@interact
def _(f = input_box(default=y), g=input_box(default=-x*y+x^3-x),
      xmin=input_box(default=-1), xmax=input_box(default=1),
      ymin=input_box(default=-1), ymax=input_box(default=1),
      start_x=input_box(default=0.5,label='Початкове значення x'),
      start_y=input_box(default=0.5,label='Початкове значення y'),
      step_size=slider(0.001, 0.2 ,default=0.01,label='Розмір кроку'),
      steps=slider(0, 1400, default=600,label='Кількість кроків') ):
    old_f = f
    f = f.function(x,y)
    old_g = g
    g = g.function(x,y)
    steps = int(steps)
    points = [ (start_x, start_y) ]
    for i in range(steps):
        xx, yy = points[-1]
        try:
```

```
            points.append( (xx+step_size*f(xx,yy),
yy+step_size*g(xx,yy)) )
        except (ValueError, ArithmeticError, TypeError):
            break
    starting_point = point(points[0], pointsize=50)
    solution = line(points)
    vector_field=plot_vector_field((f,g),(x,xmin,xmax),(y,ymin,ymax))
    result = vector_field + starting_point + solution
    html(r"<h2>$ \frac{dx}{dt} = %s$  $ \frac{dy}{dt} =
%s$</h2>"%(latex(old_f),latex(old_g)))
    print "Розмір кроку: %s"%step_size
    print "Кількість кроків: %s"%steps
    result.show(xmin=xmin,xmax=xmax,ymin=ymin,ymax=ymax)
```



$$\frac{dx}{dt} = y \qquad \frac{dy}{dt} = x^3 - xy - x$$

```
Розмір кроку: 0.0229338677354710
Кількість кроків: 606
```



Поле напрямів для методу Рунге-Кутта-Фельберга

```
var('x y')
@interact
def _(
fin = input_box(default=y+exp(x/10)-1/3*((x-1/2)^2+y^3)*x-
x*y^3,label='f='),
gin=input_box(default=x^3-x+1/100*exp(y*x^2+x*y^2)-0.7*x,label='g='),
xmin=input_box(default=-1), xmax=input_box(default=1.8),
ymin=input_box(default=-1.3),
ymax=input_box(default=1.5),
x_start=slider(-2, 2, default=-1, label='Початкове значення x'),
y_start=slider(-2, 2, default=0, label='Початкове значення y'),
error=slider(0,1,default=0.5, label='Похибка'),
t_length=slider(0, 100,default=23) ,
num_of_points = slider(5,2000, default=1500,label='Кількість точок'),
algorithm = selector([
  ("rkf45" , "Runge-Kutta-Fehlberg (4, 5)"),
  ("rk2" , "Embedded Runge-Kutta (2, 3)"),
  ("rk4" , "4th order (classical) Runge-Kutta"),
  ("rk8pd" , 'Embedded Runge-Kutta Prince-Dormand (8,9)'),
  ("rk2imp" , "Implicit 2nd order Runge-Kutta at Gaussian points"),
  ("rk4imp" , "Implicit 4th order Runge-Kutta at Gaussian points"),
  ("bsimp" , "Implicit Bulirsch-Stoer (requires the Jacobian)"),
  ("gear1" , "M=1 implicit Gear"),
  ("gear2" , "M=1 implicit Gear")
], label='Алгоритм')):
    f(x,y)=fin
    g(x,y)=gin

    ff = f._fast_float_(*f.args())
    gg = g._fast_float_(*g.args())

    path = []
    err = error
    xerr = 0
    for yerr in [-err, 0, +err]:
      T=ode_solver()
      T.algorithm=algorithm
      T.function = lambda t, yp: [ff(yp[0],yp[1]), gg(yp[0],yp[1])]
      T.jacobian = lambda t, yp: [[diff(fun,dval)(yp[0],yp[1]) for dval
in [x,y]] for fun in [f,g]]
      T.ode_solve(y_0=[x_start + xerr, y_start +
yerr],t_span=[0,t_length],num_points=num_of_points)
      path.append(line([p[1] for p in T.solution]))

    vector_field = plot_vector_field( (f,g), (x,xmin,xmax),
(y,ymin,ymax) )
    starting_point = point([x_start, y_start], pointsize=50)
    show(vector_field + starting_point + sum(path), aspect_ratio=1,
figsize=[8,9])
```

f=	-x*y^3 - 1/12*((2*x - 1)^2 + 4*y^3)*x + y + e^(1/10*x)
g=	x^3 - 1.70000000000000*x + 1/100*e^(x^2*y + x*y^2)
xmin	-1
xmax	1.80000000000000
ymin	-1.30000000000000
ymax	1.50000000000000

Початкове значення x -0.997995

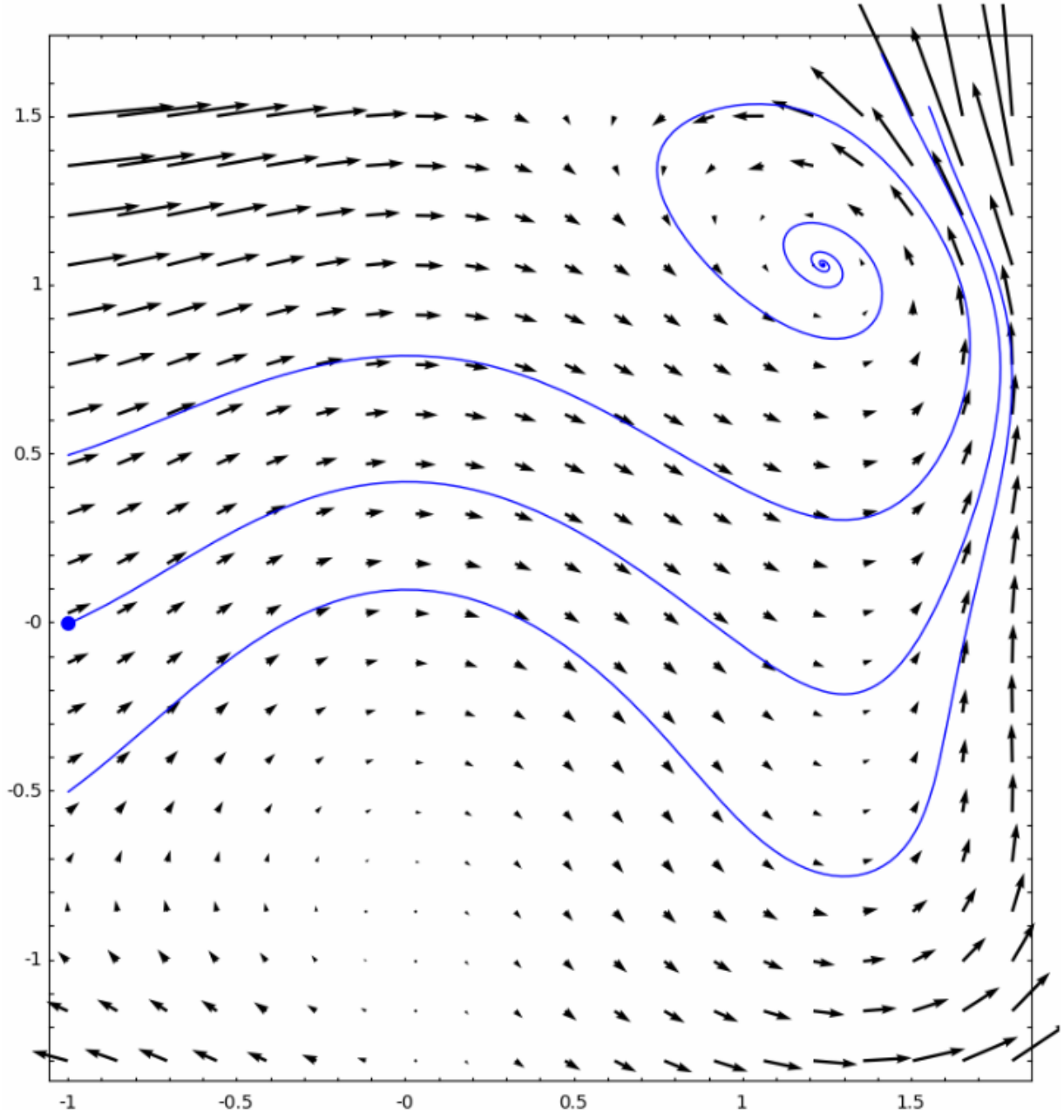
Початкове значення y -0.004008

Похибка 0.498997

t_length 22.84569

Кількість точок 1500.250

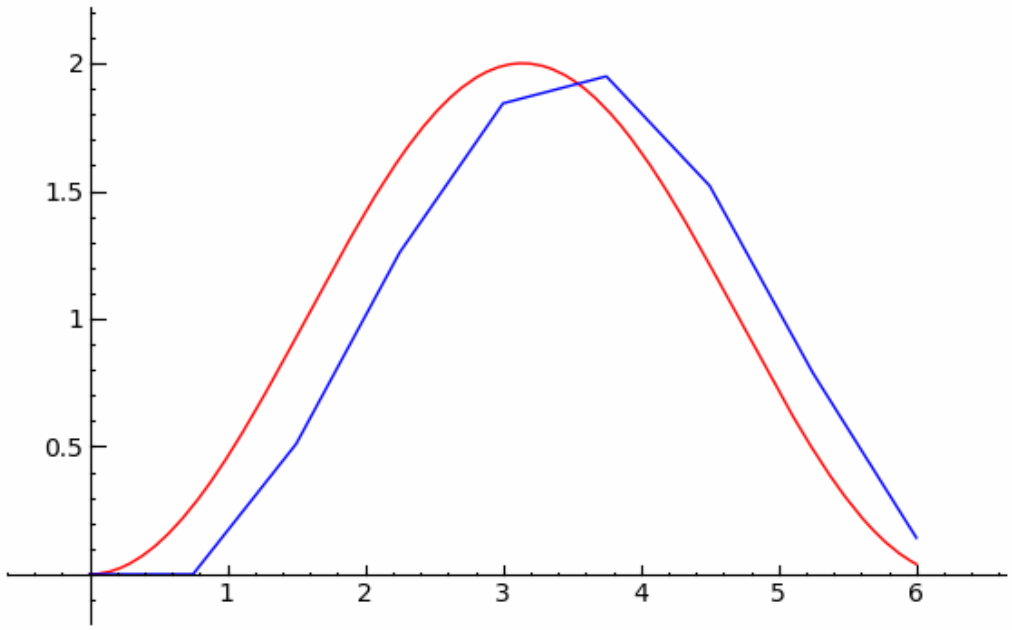
Алгоритм 'Runge-Kutta-Fehlberg (4, 5)'



Метод Ейлера для функції однієї змінної

```
def tab_list(y, headers = None):
    s = '<table border = 1>'
    if headers:
        for q in headers:
            s = s + '<th>' + str(q) + '</th>'
    for x in y:
        s = s + '<tr>'
        for q in x:
            s = s + '<td>' + str(q) + '</td>'
        s = s + '</tr>'
    s = s + '</table>'
    return s
var('x y')
@interact
def euler_method(y_exact_in = input_box('-cos(x)+1.0', type = str,
label = 'Точний розв'язок = '),
y_prime_in = input_box('sin(x)', type = str, label = "y' = "),
start = input_box(0.0, label = 'Початкове значення x: '),
stop = input_box(6.0, label = 'Кінцеве значення x: '),
startval = input_box(0.0, label = 'Початкове значення y: '),
nsteps = slider([2^m for m in range(0,10)], default = 10, label =
'Кількість кроків: '),
show_steps = slider([2^m for m in range(0,10)], default = 8, label =
'Кількість кроків у таблиці: ')):
    y_exact = lambda x: eval(y_exact_in)
    y_prime = lambda x,y: eval(y_prime_in)
    stepsize = float((stop-start)/nsteps)
    steps_shown = max(nsteps,show_steps)
    sol = [startval]
    xvals = [start]
    for step in range(nsteps):
        sol.append(sol[-1] + stepsize*y_prime(xvals[-1],sol[-1]))
        xvals.append(xvals[-1] + stepsize)
    sol_max = max(sol +
[find_maximum_on_interval(y_exact,start,stop)[0]])
    sol_min = min(sol +
[find_minimum_on_interval(y_exact,start,stop)[0]])
    show(plot(y_exact(x), start, stop, rgbcolor=(1,0,0))+
line([[xvals[index],sol[index]] for index in range(len(sol))]),
xmin=start,xmax = stop, ymax = sol_max, ymin = sol_min)
    if nsteps < steps_shown:
        table_range = range(len(sol))
    else:
        table_range = range(0,floor(steps_shown/2)) + range(len(sol)-
floor(steps_shown/2),len(sol))
    html(tab_list([[i,xvals[i],sol[i]] for i in table_range], headers =
['крок','x','y']))
```

Точний розв'язок =	-cos(x)+1.0
y' =	sin(x)
Початкове значення x:	0.0
Кінцеве значення x:	6.0
Початкове значення y:	0.0
Кількість кроків:	8
Кількість кроків у таблиці:	8

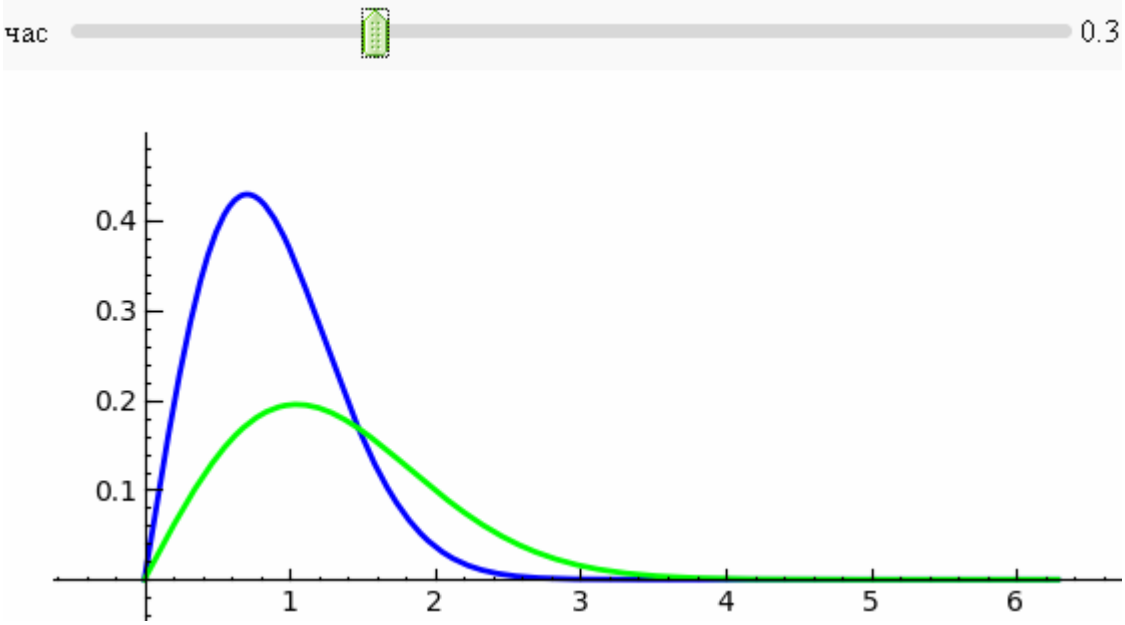


крок	x	y
0	0.000000000000000	0.000000000000000
1	0.750000000000000	0.000000000000000
2	1.50000000000000	0.511229070017501
3	2.25000000000000	1.25935030997054
5	3.75000000000000	1.94874521368138
6	4.50000000000000	1.52007422462462
7	5.25000000000000	0.786926636375802

Рівняння теплопровідності

```
var('x')
x0  = 0
k=1
f   = x*exp(-x^2)
p   = plot(f,0,2*pi, thickness=2)
c   = 1/pi
orden=10
alpha=[(n,c*numerical_integral(f(x)*sin(x*n/2),0,2*pi)[0] ) for n in
range(1,orden)]
@interact
def _(tiempo = slider([0.1*j for j in (0..10)],label='час') ):
```

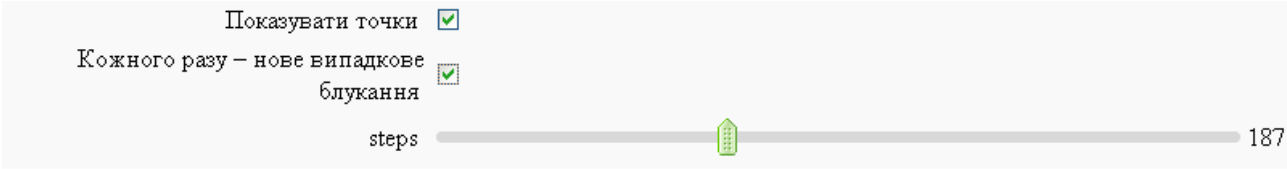
```
ft = sum( a*sin(x*n/2)*exp(-k*(n/2)^2*tiempo) for n,a in alpha)
pt = plot(ft, 0, 2*pi, color='green', thickness=2)
show( p + pt, ymin = -.2)
```



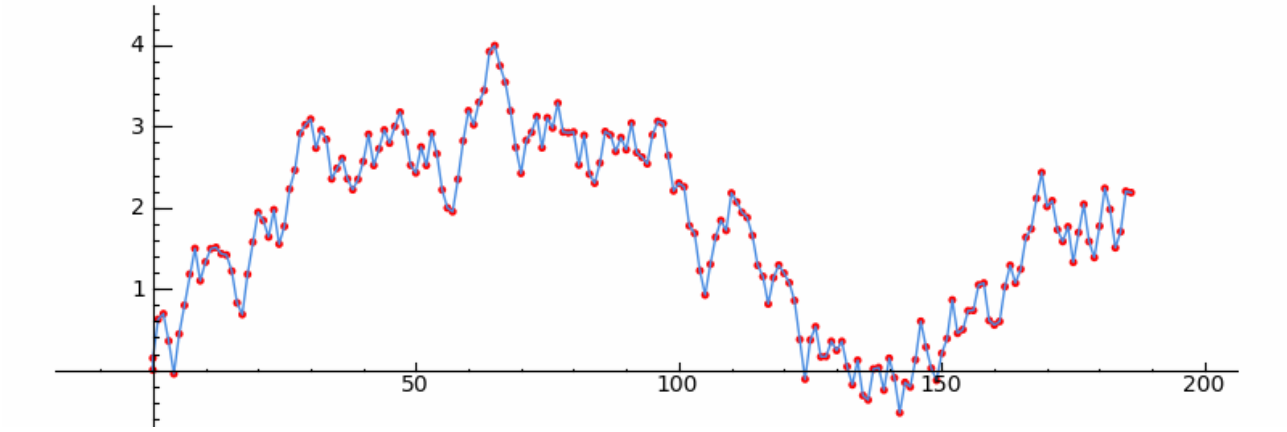
8. Динамічні системи

Випадкове блукання

```
vv = []; nn = 0
@interact
def foo(pts = checkbox(True, "Показувати точки"),
        refresh = checkbox(False, "Кожного разу - нове випадкове
блукання"),
        steps = (50,(10..500))):

    html("<h2>%s кроків</h2>"%steps)
    global vv
    if refresh or len(vv) == 0:
        s = 0; v = [(0,0)]
        for i in range(steps):
             s += random() - 0.5
             v.append((i, s))
        vv = v
    elif len(vv) != steps:
        s = vv[-1][1]; j = len(vv)
        for i in range(steps - len(vv)):
            s += random() - 0.5
            vv.append((i+j,s))
        v = vv[:steps]
    else:
        v = vv
    L = line(v, rgbcolor='#4a8de2')

    if pts: L += points(v, pointsize=10, rgbcolor='red')
    show(L, xmin=0, figsize=[8,3])
```

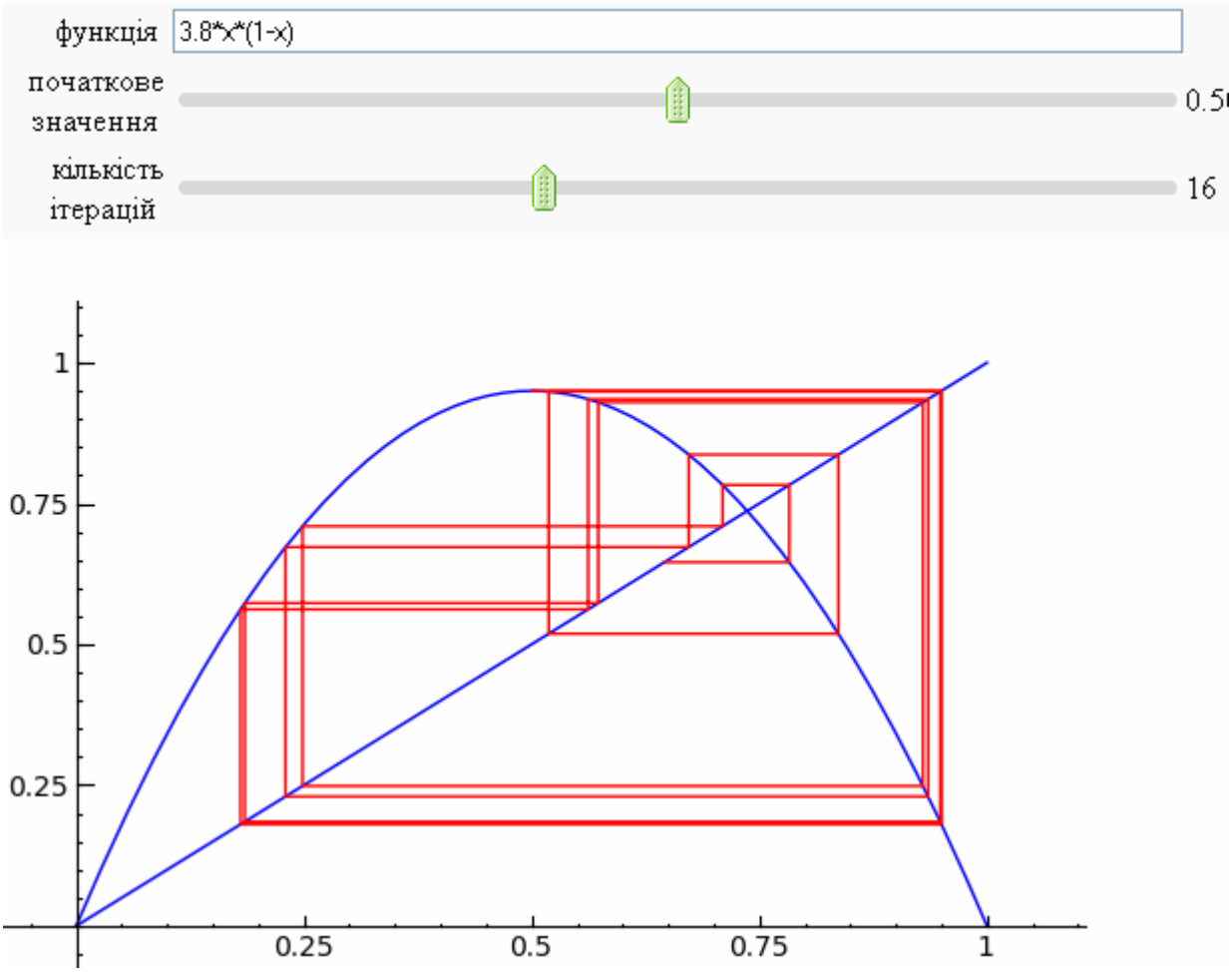


187 кроків



Діаграма Ферхюльста на [0; 1]

```
def cobweb(a_function, start, mask = 0, iterations = 20, xmin = 0, xmax
= 1):
    basic_plot = plot(a_function, xmin = xmin, xmax = xmax)
    id_plot = plot(lambda x: x, xmin = xmin, xmax = xmax)
    iter_list = []
    current = start
    for i in range(mask):
        current = a_function(current)
    for i in range(iterations):
        iter_list.append([current,a_function(current)])
        current = a_function(current)
        iter_list.append([current,current])
    cobweb = line(iter_list, rgbcolor = (1,0,0))
    return basic_plot + id_plot + cobweb
var('x')
@interact
def cobwebber(
f_text = input_box(default = "3.8*x*(1-x)",label = "функція", type=str),
start_val = slider(0,1,.01,.5,label = 'початкове значення'),
its = slider([2^i for i in range(0,12)],default = 16, label="кількість
ітерацій")):
    def f(x):
        return eval(f_text)
    show(cobweb(f, start_val, iterations = its))
```



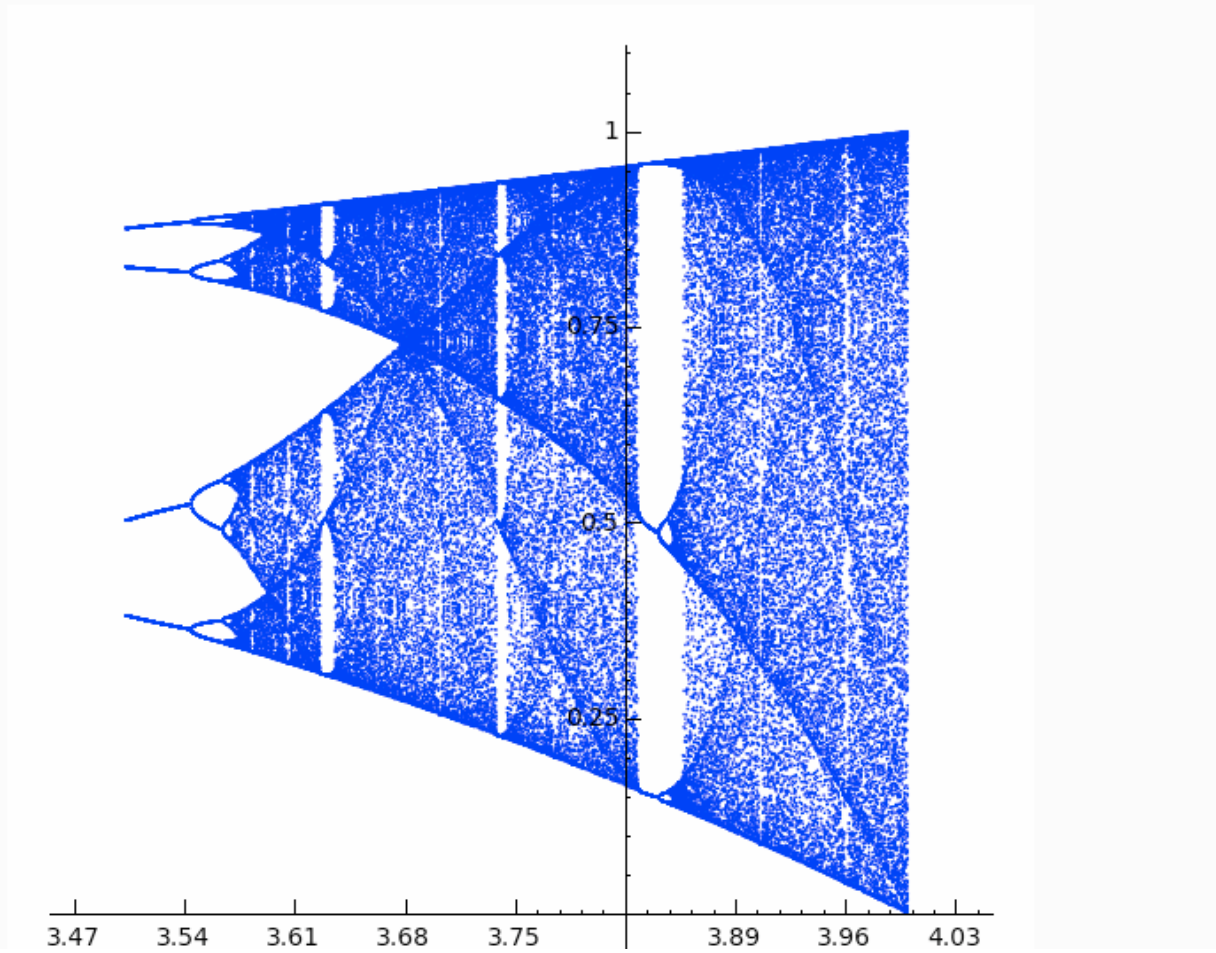
Логістичне відображення

```
%cython
cpdef double logorb(double k,long N,double x0):
    cdef double x = x0
    cdef long i
    for i from 1 <= i <= N:
        x = k*x*(1-x)
    return x

cpdef logtraj(double k,long N, double x0):
    cdef double x = x0
    xvals = []
    cdef long i
    for i from 1 <= i <= N:
        x = k*x*(1-x)
        xvals.append(x)
    return xvals

@interact
def logistic_bifs(k_min = slider(0.0,4.0,.001,3.5), k_max =
slider(0.0,4.0,.001,4.0)):
    tkmax = max(k_min, k_max)
    tkmin = min(k_min, k_max)
    dk = (tkmax - tkmin)/1000.0
    xpts = []
```

```
    x = .5
    for k in srange(tkmin,tkmax,dk):
        x = logorb(k,100,x)
        ks = logtraj(k,12,x)
        if max(ks)-min(ks) < .001:
            xpts.append([k,x])
        else:
            x = logorb(k,1000,x)
            ks = logtraj(k,100,x)
            xpts = xpts + [[k,q] for q in ks]
    show(points(xpts, pointsize = 1), figsize = [6,6])
```

k_min 3.50299999999973

k_max 4.00000000000000

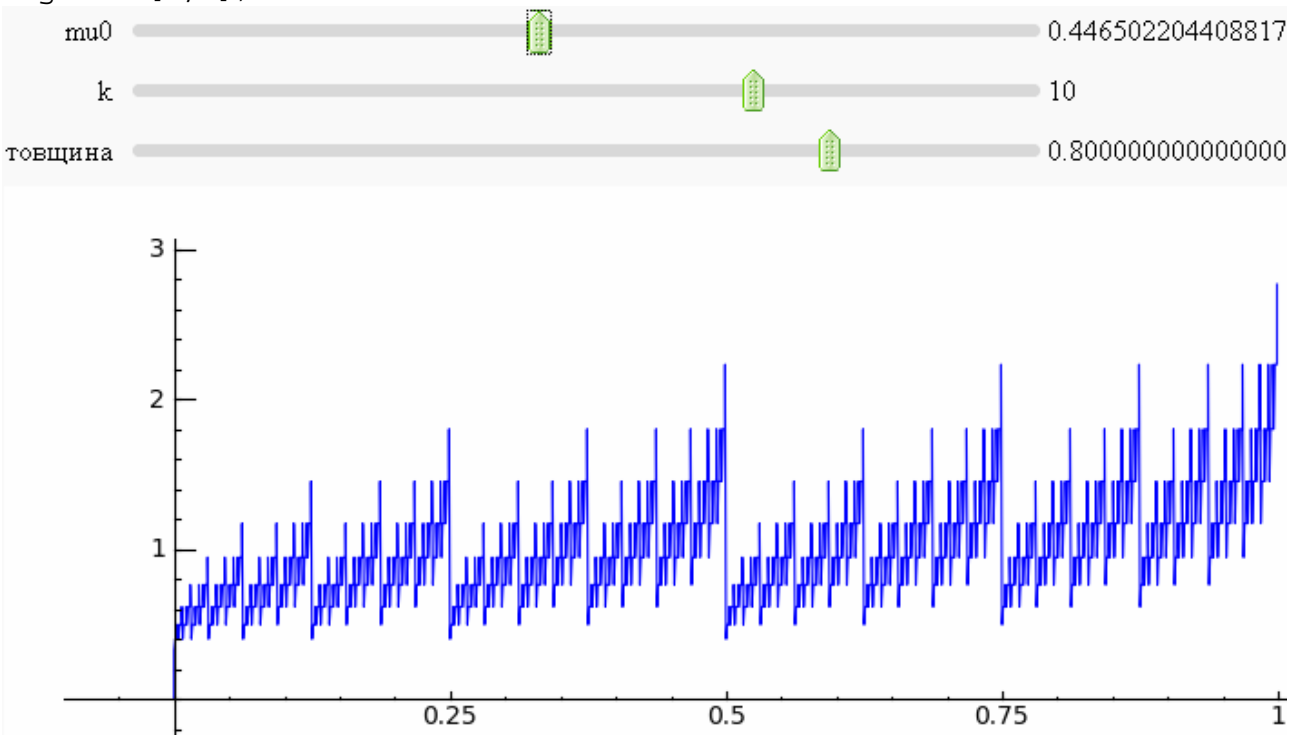


9. Фрактали

Біноміальний фрактальний розподіл Мандельброта

```
def muk_plot(m0,k):
    k = int(k)
    m0 = float(m0)
    m1 = float(1 - m0)
    assert m0 > 0 and m1 > 0, "обидва мають бути додатними"
    v = [(0,0)]
    t = 0
```

```
    two = int(2)
    delta = float(1/2^k)
    multiplier = float(2^k)
    for i in [0..2^k-1]:
        t = i * delta
        phi1 = i.str(two).count("1")
        phi0 = k - phi1
        y = m0^(phi0)*m1^(phi1)*multiplier
        v.append((t,y))
        v.append((t+delta,y))
    return v

@interact
def _(mu0=(0.3,(0.0001,0.999)), k=(3,(1..14)),
thickness=sloder([0.1,0.2,..,1.0], default=1.0, label='товщина')):
    v = muk_plot(mu0,k)
    line(v,thickness=thickness).show(xmin=0, xmax=1, ymin=0,
figsize=[8,3])
```



Дракон Хартера-Хейтуея

```
A = matrix([[1,1],[-1,1]])
D = [vector([0,0]), vector([1,0])]

@interact
def f(A = matrix([[1,1],[-1,1]]), D = '[[0,0],[1,0]]', k=(3..17)):
    print "Визначник = ", A.det()
    D = matrix(eval(D)).rows()
    def Dn(k):
        ans = []
        for d in Tuples(D, k):
            s = sum(A^n*d[n] for n in range(k))
            ans.append(s)
        return ans
```

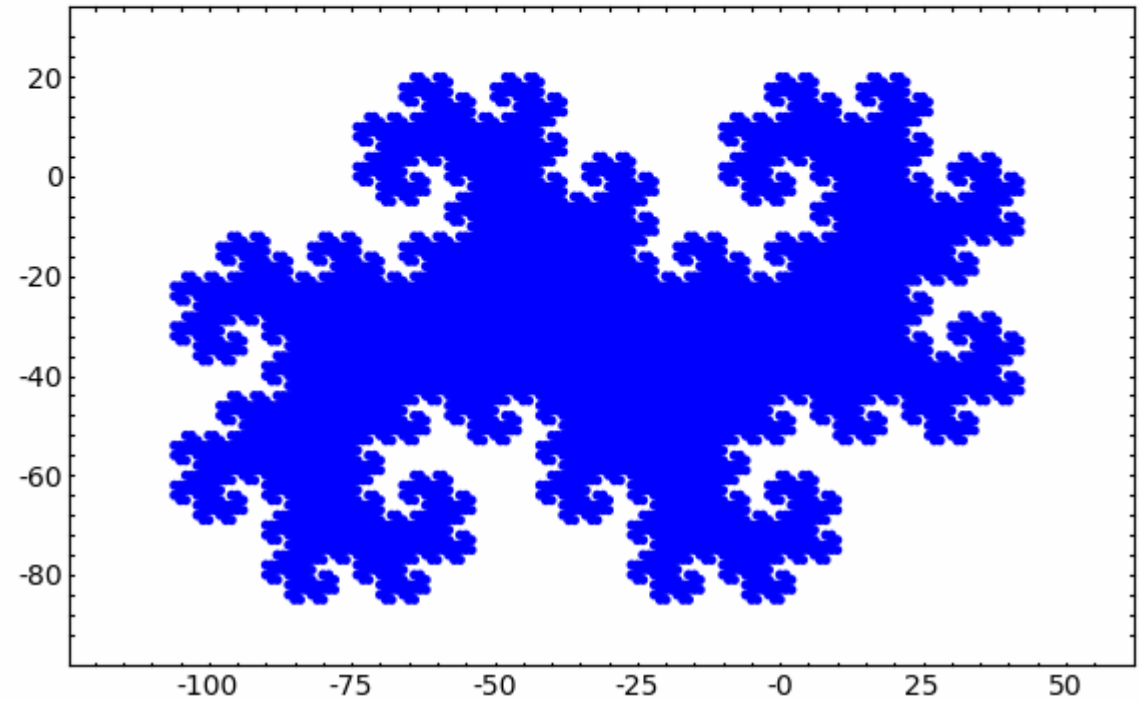
```
G = points([v.list() for v in Dn(k)])
show(G, frame=True, axes=False)
```

A [1] [1]
[-1] [1]

D [[0,0],[1,0]]

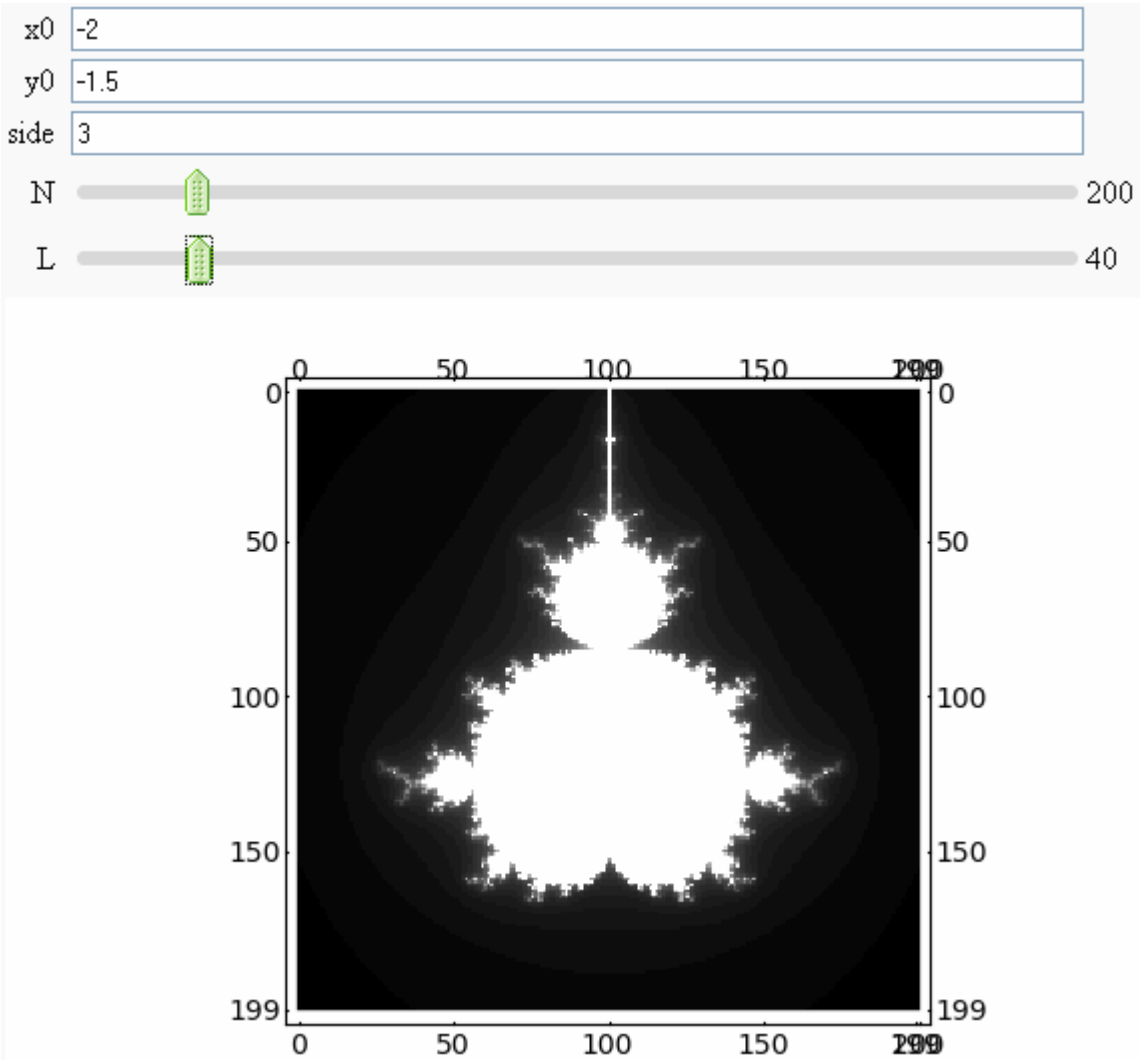
k 14

```
Визначник =  2
```



Множина Мандельброта

```
import numpy as np
def mandelbrot(x0,x1,y0,y1,N=200, L=50, R=3):
    m= np.zeros([N,N], dtype=np.int)
    for i in range(N):
        for k in range(N):
            c=complex(x0+i*(x1-x0)/N, y0+k*(y1-y0)/N)
            z=complex(0,0)
            h=0
            while (h<L) and (abs(z)<R):
                z=z*z+c
                h+=1
            m[i,k]=h
    return m

@interact
def showme_mandelbrot(x0=-2, y0=-1.5, side=3.0,N=(100*i for i in
range(1,11)), L=(20*i for i in range(1,11)) ):
    m=mandelbrot(x0 ,x0 + side ,y0 ,y0 + side , N, L )
    show(matrix_plot(m))
```


x0
-2
y0
-1.5
side
3
N
200
L
40
0
50
100
150
199

Додаток Г

Навчальна програма з дисципліни «Системне програмування»

ЗАГАЛЬНІ ВІДОМОСТІ

1. Мета курсу – дати теоретичну підготовку в галузі методів проектування та реалізації основних компонентів системного програмного забезпечення в середовищі POSIX-сумісних операційних систем.

Задачі курсу – забезпечити студентів поняттями, необхідними для:

1) ідентифікації абстрактних послуг, спільних для всіх операційних систем;
2) визначення системних компонент основних операцій, що підтримують машинно-незалежні абстракції в конкретній цільовій архітектурі;
3) розуміння принципів побудови цілісної операційної системи і взаємодії її окремих модулів;
4) розуміння засобів, за допомогою яких можуть бути проаналізовані фундаментальні проблеми в операційних системах;
5) розуміння принципів побудови системного програмного забезпечення.

2. Необхідною основою для вивчення курсу є попереднє засвоєння курсу «Програмування».

3. З дисципліни передбачено лекційні та лабораторні заняття.

4. В результаті вивчення курсу студент повинен:

знати:

– що таке операційна система;
– чотири генерації операційних систем;
– концепції операційних системи;
– режими виконання процесів в Unix;
– структуру операційної системи;
– організацію введення-виведення;
– мультіпрограмування;
– поняття процесу;
– підсистему управління процесами в Unix;
– міжпроцесну взаємодію;
– проблему критичних секцій;
– апаратні засоби синхронізації;
– семафори;
– розв’язання проблеми “Виробник–Споживач” за допомогою семафорів;
– лічильники подій;
– високорівневі методи синхронізації;
– схеми синхронізації, побудовані на повідомленнях;
– порти та поштові скриньки;
– віддалені виклики процедур;
– ядро операційної системи;
– структури даних для процесів та ресурсів;
– базові операції над процесами та ресурсами;

– організацію планувальників процесів;
– планування процесів чи центрального процесора;
– алгоритми планування, що базуються на часі виконання;
– FIFO- та LIFO-планування;
– планування наступним найкоротшого та найдовшого завдання;
– поняття взаємного блокування;
– системну модель взаємного блокування;
– необхідні умови взаємних блокувань;
– алгоритми виявлення взаємних блокувань;
– алгоритми банкіра та безпечного шляху;
– етапи підготовки програми до виконання;
– найпростіші схеми управління пам’яттю;
– принципи віртуальної пам’яті;
– планування процесів та захист пам’яті в сторінкових системах;
– реалізацію віртуальної пам’яті;
– алгоритми заміни сторінок;
– розподіл фреймів;
– керування пам’яттю в MS-DOS;
– структуру та типи файлів;
– файли, відображені у пам’ять;
– модель володіння в UNIX;
– організацію та структури даних для каталогів;
– операції над каталогами;
– структуру файлової системи;
– розподілені файлові системи;
– вторинне зберігання та будову диску;
– планування диску;
– ієрархічну модель файлової та дискової підсистем.

вміти:

– відкривати та створювати файл за допомогою системного виклику open;
– створювати файл за допомогою системного виклику open;
– створювати файл за допомогою системного виклику creat;
– закривати файл за допомогою системного виклику close;
– читати з файлу за допомогою системного виклику read;
– записувати у файл за допомогою системного виклику write;
– копіювати файли;
– довільно пересуватися по файлу за допомогою системного виклику lseek;
– дописувати дані у кінець файлу;
– знищувати файли;
– використовувати системний виклик fcntl;
– користуватися стандартним вводом, стандартним виводом і стандартним виводом діагностики;
– пов’язувати системні виклики, змінну errno та підпрограму perror;
– визначати власників, права доступу та режими файлів;
– застосовувати маску створення файлу і системний виклик umask;

- задавати права доступу до файлу за допомогою системного виклику open;
- визначати доступ до файлу за допомогою виклику access;
- змінювати права доступу за допомогою виклику chmod;
- змінювати власника за допомогою виклику chown;
- для файлів з декількома іменами застосовувати системні виклики link, unlink, rename, символьні посилання;
- одержувати інформацію про файл за допомогою викликів stat і fstat;
- застосовувати системні виклики link і unlink до каталогів;
- змінювати права доступу до каталогів;
- створювати і видаляти каталоги;
- відкривати і закривати каталоги;
- читати каталоги за допомогою викликів readdir і rewinddir;
- змінювати поточний робочий каталог за допомогою виклику chdir;
- визначати ім’я поточного робочого каталогу;
- обходити дерево каталогів;
- кешувати об’єкти файлових систем UNIX за допомогою викликів sync і fsync;
- працювати з файлами блокових і символьних пристроїв;
- використовувати структуру stat;
- отримувати інформацію про файлову систему;
- змінювати обмеження файлової системи за допомогою процедур pathconf і fpathconf;
- створювати процеси за допомогою системного виклику fork;
- запускати нові програми за допомогою виклику exec;
- спільно використовувати виклики exec і fork;
- успадковувати дані і дескриптори файлів;
- завершувати процеси за допомогою системного виклику exit;
- синхронізувати процеси за допомогою системного виклику wait;
- очікувати завершення визначеного нащадка за допомогою виклику waitpid;
- перешкоджати породженню зомбі-процесів при передчасному завершенні програми;
- визначати ідентифікатор процесу;
- оперувати з групами процесів і ідентифікаторами групи процесів;
- змінювати групи процесу;
- визначати ідентифікатор сеансу;
- отримувати та встановлювати змінні програмного оточення;
- змінювати поточний робочий каталог;
- змінювати поточний кореневий каталог;
- визначати ідентифікатори користувача і групи;
- змінювати обмеження на розмір файлу за допомогою виклику ulimit;
- змінювати пріоритети процесів за допомогою виклику nice;
- задавати обробники;
- блокувати сигнали;
- посилати сигнали;
- створювати програмні канали;
- використовувати канали у програмі;

- визначати розмір каналу;
- записувати та читати канали без блокування;
- використовувати системний виклик select для роботи с кількома каналами;
- створювати іменовані канали;
- блокувати записи за допомогою виклику fcntl;
- програмувати черги повідомлень, семафори, поділювану пам'ять;
- програмувати термінал та змінювати його властивості;
- програмувати в режимі TCP-з'єднання: створювати сокет, зв'язуватися та вмикати прийом TCP-з'єднань;
- програмувати в режимі пересилань UDP-дейтаграм;
- керувати динамічним розподілом пам'яті;
- програмувати введення/виведення з відображенням у пам'ять;
- одержувати параметри локальної системи.

5. Форми та засоби проведення поточного контролю – захист лабораторної роботи, співбесіда за матеріалом контрольної роботи.

Тематика курсу лекцій

№	Найменування теми (модуля). Основні питання лекції і її зміст
1.	**Вступ до системного програмування.** Поняття операційної системи. Багатослойна архітектура комп'ютерної системи. Ранні системи (1945-1955). Друге покоління систем (1956-1965). Третє покоління систем (1965-1980). Четверте покоління та подальший розвиток операційних систем. Історія Windows 2000. Історія традиційних UNIX-систем. Сучасні UNIX-системи: SVR4, Solaris 2.x, 4.4BSD, Linux. Історія та модульна архітектура Linux. Концепції операційної системи: програма, процеси, потоки, файли, системні виклики, оболонки. Концепції операційної системи: ядро. Ядро в UNIX та Windows NT. Концепції операційної системи: пам'ять, управління пам'яттю. Режими виконання процесів в UNIX. Структура операційної системи: мінімальні ОС, багатослойні системи, віртуальні машини, модель клієнт-сервер. Засоби введення-виведення. Мультипрограмування. Мультипрограмні ОС та їх задачі. Операційні системи як віртуальні машини. Однокористувацька багатозадачність, архітектура, процеси користувача, модель клієнт-сервер, потоки, об'єкти у Windows 2000.
2.	**Взаємодія процесів.** Поняття процесу. Модель процесу. Конкуренція процесів. Ієрархія процесів. Стани процесу. Реалізація процесів: таблиця процесів та вектор переривання. Створення процесів. Процеси в UNIX. Процеси в MS-DOS. Завершення процесу. cobegin/coend. Примітиви fork, join та quit. Підсистема управляння процесами в UNIX. Взаємодія процесів. Процеси та потоки, багатопоточність, стани потоків, підтримка потоків в підсистемах ОС, симетрична багатопроцесність у Windows 2000. Стани,

№	Найменування теми (модуля). Основні питання лекції і її зміст
	опис та управління процесами в SVR4. Багатопоточна архітектура, структура процесу, виконання потоків, переривання як потоки у Solaris. Процеси та потоки в Linux. Проблема критичної секції: умови розробки протоколу взаємодії процесів, використання змінних у спільної пам'яті та прапорів. Механізми конкуренції у Windows 2000. Механізми конкуренції в UNIX: канали, повідомлення, спільна пам'ять, семафори та сігнали. Примітиви синхронізації в Solaris: блокування взаємного виключення, семафори, умовні змінні, блокування читання/запису. Проблема критичної секції: алгоритм Петерсона, алгоритм трьох станів, алгоритм Бейкері. Апаратні засоби синхронізації. Семафори. Бінарні та узагальнені семафори. Проблема зайнятості-очікування. Протокол блокування-пробудження. Розв'язання проблеми "Виробник–Споживач" за допомогою семафорів. Лічильники подій. Високорівневі методи синхронізації: монітори. Розв'язання проблеми обідаючих філософів за допомогою моніторів. Реалізація моніторів. Високорівневі методи синхронізації: умовні критичні регіони. Схеми синхронізації, що базуються на повідомленнях. Примітиви send та receive. Проблема "Виробник–Споживач" у термінах повідомлень. Проблеми, які виникають при реалізації схем синхронізації, що базуються на повідомленнях. Порти та поштові скриньки. Віддалені виклики процедур.
3.	**Планування процесів.** Ядро операційної системи. Структури даних для процесів та ресурсів: блок управління процесом. Структури даних для процесів та ресурсів: дескриптори ресурсів. Базові операції над процесами та ресурсами. Організація планувальників процесів. Пріоритети процесів і потоків та багатопроцесорне планування у Windows 2000. Планування в Linux та SVR4. Вимоги до алгоритму планувальника. Типи планувальників. Універсальний планувальник. Алгоритми планування, що базуються на часі виконання. FIFO-планування. LIFO-планування. Планування наступним найкоротшого завдання. Планування першим найдовшого завдання. Планування за найменшим очікуваним часом. Планувальник Раунд-Робіна. Планування за багаторівневим зворотним зв'язком. Планування процесора за полісною функцією.
4.	**Керування ресурсами та тупики.** Поняття взаємного блокування. Приклади взаємних блокувань. Системна модель взаємного блокування. Необхідні умови взаємних блокувань. Взаємні блокування в Unix. Граф розподілу ресурсів. Виявлення взаємних блокувань. Алгоритми виявлення взаємних блокувань. Відновлення після взаємних блокувань. Попередження взаємних блокувань. Уникнення взаємних блокувань. Алгоритми банкіра та безпечного шляху.

№	Найменування теми (модуля). Основні питання лекції і її зміст
5.	**Керування пам'яттю.** Підготовка програми до виконання: створення програми, зв'язування та переміщення адресів, компонування. Найпростіші схеми управління пам'яттю: управління пам'яттю в одному та двох блоках. Багатоблочний поділ пам'яті. Проблема фрагментації. Оверлії. Принципи віртуальної пам'яті: обмін, базові регістри, сторінки. Планування процесів в сторінкових системах. Поділювані сторінки. Захист пам'яті в сторінкових системах. Сегментація. Сторінкова сегментація. Реалізація віртуальної пам'яті: обмін за вимогою та його продуктивність, промахи, розрахунок ефективного часу доступу. Заміна сторінок, брудний біт. Алгоритми заміни: FIFO, оптимальний, LRU, LFU, MFU. Аномалія Біледі. Розподіл фреймів. Порушення розподілу. Керування пам'яттю в MS-DOS.
6.	**Файлові системи.** Файли. Структура файлу. Типи файлів: звичайні, каталоги, спеціальні символьні та блочні, хардлінки та сімлінки, іменовані канали, сокети. Звичайні файли. Доступ до файлів. Атрибути файлів. Операції над файлами. Файли, відображені у пам'ять. Модель володіння в UNIX. Ієрархічна система каталогів. Організація та структури даних для каталогів. Шляхи імен. Операції над каталогами. Файлові системи. Коренева файлова система. Доступ до файлів в UNIX. Індексні дескриптори. Структура файлової системи. Розподілені файлові системи. Файлові системи Windows NT. Active Directory в Windows 2000.
7.	**Керування введенням/виведенням.** Вторинне зберігання. Будова диску. Управління вільним місцем. Методи виділення: суцільне, зв'язане, індексне. Планування диску. Вибір алгоритму планування. Ієрархічна модель файлової та дискової підсистем. Дискові файлові системи. Захист файлів.

Перелік тем лабораторних занять

№	Найменування теми (модуля). Основні питання лабораторного заняття
1.	**Лабораторна робота №1. Файли та каталоги** (завдання 1–43). 1. Файл: каталоги і шляхи, власник файлу і права доступу, узагальнення концепції файлу. 2. Процес, міжпроцесна взаємодія. 3. Системні виклики і бібліотечні підпрограми. 4. Примітиви доступу до файлів у системі UNIX: системний виклик open,

№	Найменування теми (модуля). Основні питання лабораторного заняття
	створення файлу за допомогою виклику open, системний виклик creat, системний виклик close, системний виклик read, системний виклик write, виклик lseek і довільний доступ, дописування даних у кінець файлу, видалення файлу, системний виклик fcntl. 5. Стандартний ввід, стандартне вивід і стандартний вивід діагностики. 6. Стандартна бібліотека введення/виведення. 7. Системні виклики і змінна errno, підпрограма perror. 8. Файли в багатокористувацькому середовищі: користувачі і права доступу, права доступу і режими файлів, додаткові права доступу для файлів, що виконуються, маска створення файлу і системний виклик umask, виклик open і права доступу до файлу, визначення доступності файлу за допомогою виклику access, зміна прав доступу за допомогою виклику chmod, зміна власника за допомогою виклику chown. 9. Файли з декількома іменами: системний виклик link, системний виклик unlink, системний виклик rename, символьні посилання. 10. Одержання інформації про файл: виклики stat і fstat, докладніше про виклик chmod. 11. Каталоги з погляду користувача 12. Реалізація каталогів: знову про системні виклики link і unlink, крапка і подвійна крапка, права доступу до каталогів. 13. Використання каталогів при програмуванні: створення і видалення каталогів, відкриття і закриття каталогів, читання каталогів: виклики readdir і rewinddir, поточний робочий каталог, зміна робочого каталогу за допомогою виклику chdir, визначення імені поточного робочого каталогу, обхід дерева каталогів. 14. Файлові системи UNIX, кешування: виклики sync і fsync. 15. Імена пристроїв UNIX: файли блокових і символьних пристроїв, структура stat, інформація про файлову систему, обмеження файлової системи: процедури pathconf і fpathconf.
2.	**Лабораторна робота №2.** Процеси та сигнали (завдання 1–20). 1. Поняття процесу. 2. Створення процесів: системний виклик fork. 3. Запуск нових програм за допомогою виклику exec: сімейство викликів exec, доступ до аргументів, переданих при виклику exec. 4. Спільне використання викликів exec і fork. 5. Успадкування даних і дескрипторів файлів: виклик fork, файли і дані, виклик exec і відкриті файли. 6. Завершення процесів за допомогою системного виклику exit. 7. Синхронізація процесів: системний виклик wait, очікування завершення визначеного нащадка: виклик waitpid. 8. Зомбі-процеси і передчасне завершення програми. 9. Атрибути процесу: ідентифікатор процесу, групи процесів і ідентифікатори групи процесів, зміна групи процесу, сеанси й ідентифікатор сеан-

№	Найменування теми (модуля). Основні питання лабораторного заняття
	су, змінні програмного оточення, поточний робочий каталог, поточний кореневий каталог, ідентифікатори користувача і групи, обмеження на розмір файлу: виклик ulimit, пріоритети процесів: виклик nice. 10. Поняття сигналу: імена сигналів, нормальне та аварійне завершення. 11. Обробка сигналів: набори сигналів, задання обробника сигналів, сигнали та системні виклики, процедури sigsetjmp і siglongjmp. 12. Блокування сигналів. Посилка сигналів: посилка сигналів іншим процесам, посилка сигналів самому процесу, системний виклик pause.
3.	**Лабораторна робота №3. Взаємодія процесів** (завдання 1–20). 1. Програмні канали: канали на рівні команд, використання каналів у програмі, розмір каналу, закриття каналів, запис та читання без блокування. 2. Використання системного виклику select для роботи с кількома каналами, канали та системний виклик exec. 3. Іменовані канали. 4. Блокування записів за допомогою виклику fcntl. 5. Черги повідомлень, семафори, поділювана пам'ять.
4.	**Лабораторна робота №4. Керування терміналом** (завдання 1–12). 1. Термінал UNIX: керуючий термінал, передача даних, луна-відображення символів, що вводяться, випереджальне уведення з клавіатури, канонічний режим терміналу, редагування рядка і спеціальні символи. 2. Програмування терміналу: системні виклики open, read, write, функції ttyname і isatty, зміна властивостей термінала: структура termios. 3. Псевдотермінали.
5.	**Лабораторна робота №5. Інтерфейс сокетів** (завдання 1–4). 1. Сокети: типи з'єднання, адресація Internet, порти, інтерфейс сокетів. 2. Програмування в режимі TCP-з'єднання: створення сокету, зв'язування, вмикання прийому TCP-з'єднань. 3. Програмування в режимі TCP-з'єднання: прийом запиту на встановлення TCP-з'єднання, підключення клієнта, пересилання даних, закриття TCP-з'єднання. 4. Програмування в режимі пересилань UDP-дейтаграм.

Перелік навчально-методичної літератури

№	Назва, автори, рік видання
	Основна
1.	Хэвиленд К., Грэй Д., Салама Б. Системное программирование в UNIX. Руководство программиста по разработке ПО: Пер. с англ. – М.: ДМК Пресс, 2000. – 368 с.
2.	Робачевский А.М. Операционная система UNIX. – СПб.: БХВ-Петербург, 2000.
3.	Керниган Б.В., Пайк Р. UNIX – универсальная среда программирования. –

№	Назва, автори, рік видання
	М.: Финансы и статистика, 1992.
4.	Бах М. Архитектура операционной системы UNIX. – М.: Финансы и статистика, 1994.
5.	Цикритзис Д., Бернстайн Ф. Операционные системы. – М.: Мир, 1977.
6.	A.D. Marshall. Programming in C. UNIX System Calls and Subroutines using C. (http://www.cm.cf.ac.uk/Dave/C/CE.html)
7.	Столлингс В. Операционные системы. Внутренне устройство и принципы проектирования. – М.: Вильямс, 2002. – 848 с.
	Допоміжна
8.	Симоненко В.П. Организация вычислительных процессов в ЭВМ, комплексах, сетях и системах. – К.: ТОО «Век+», 1997.
9.	Олифер Н.А., Олифер В.Г. Сетевые операционные системы. – СПб.: Питер, 2001.
10.	Уолтон Шон. Создание сетевых приложений в среде Linux. Руководство разработчика. – М.: Вильямс, 2001. – 464 с.
11.	Стивенс Уильям. UNIX: взаимодействие процессов. – СПб.: Питер, 2002. – 576 с.
12.	Богатырев А. Хрестоматия по программированию на Си в Unix.
13.	Голдт С., Ван дер Меер С., Буркетт С., Уэлш М. Руководство программиста для Linux.
14.	Стивенс У.Р. UNIX: разработка сетевых приложений.
15.	Таненбаум Э.С., Вудхалл А.С. Операционные системы. Разработка и реализация, 2005. – 576 с.
16.	Дегтярев Е.К. Введение в UNIX. – М., 1991.
17.	Соломон Д., Руссинович М. Внутреннее устройство Microsoft Windows 2000 (Мастер-класс)
18.	Фролов А.В., Фролов Г.В. Программирование для Windows NT. Часть 1, 2. – М.: Диалог-МИФИ, 1996. – 272 с.
19.	Ядро ОС Linux. Руководство программиста.
20.	Рихтер Дж. Windows для профессионалов: создание эффективных WIN32-приложений с учетом специфики 64-разрядной версии Windows.
21.	Дюпуа Б., Габасси М. UNIX и TCP/IP: программирование на языке С.

Додаток Д

Завдання олімпіади з системного програмування (КДПУ, 2006 р.)

Олімпіада складається із двох частин – теоретичної (тести) і практичної (завдання). Кожна частина оцінюється в 50 балів. Вся робота оцінюється з максимуму 100 балів.

У **теоретичній** частині потрібно дати відповіді на будь-які 25 питань (1 питання дає 2 бали). При відповіді більш ніж на 25 питань ураховуються тільки перші 25. Відповіді записуються в ***аркуш відповідей***, що здається по завершенні роботи.

У практичній частині потрібно розв'язати будь-які 5 завдань на вибір (кожне завдання дає 10 балів).

При оцінюванні завдань ураховуються:

1. Ступінь працездатності й повнота виконання поставленого завдання в створеному ПЗ.

2. Опис розробленого алгоритму (текстовий та/або графічний; **записується в аркуш відповідей**).

3. Оптимальність реалізованого алгоритму.

4. Опис реалізованої технології одержання ПЗ (коротко; текстовий/графічний; **записується в аркуш відповідей**).

5. Рівень застосування сучасних технологій створення ПЗ.

6. Оцінка розміру файлів у створеному ПЗ користувача.

7. Наявність діалогової взаємодії ПЗ з користувачем (наочність, простота, підказки, …)...

8. Ступінь захищеності створеного ПЗ від помилок користувача.

9. Текст інструкції для експлуатації створеного ПЗ (короткий, найбільш важливе; **записується в аркуш відповідей**).

10. Читабельність написаної програми (структурованість, наявність пояснюючих коментарів).

По закінченні роботи необхідно зберегти всі вихідні файли в папці із прізвищем учасника. Наприклад: **C:\Work\Olimpiada2\Ivanov**

Тести

При відповідях на тестові питання потрібно записувати в аркуш відповіді лише номера правильних відповідей. Наприклад: 1.6, 2.9 і т.д.

1. Фрагмент програми: const n=10; int a[n],i,s=0; srand(time(NULL)); for(i=0;i<n;i++) s+=(!((a[i]=rand()%10)%2)); обчислює:

1.1) Суму елементів масиву, кратних 20.

1.2) Кількість елементів масиву, кратних 20.

1.3) Суму парних елементів масиву.

1.4) Кількість парних елементів масиву.

1.5) Суму непарних елементів масиву.

2. Шаблон змінної cod має вигляд: union tcod{char sim;struct{int x:5;int y:3;}hh;}; Код латинської букви 'A' дорівнює 65. Фрагмент програми: tcod cod; cod.hh.x=4; cod.hh.y=2; cout<<"\n"<<' '<<sizeof(cod)<<' '<<cod.sim;
виведе на екран:

2.1) 5 D

2.2) 2 C

2.3) 4 D

2.4) 4 C

2.5) 1 D

3. Чи може бути розіменований вказівник типу void?

3.1) так;

3.2) немає;

3.3) не завжди;

3.4) залежно від його використання;

3.5) так, якщо він описує тип результату функції, що повертається.

4. Нижче за допомогою термінальних і нетермінальних символів, знака порожньо ($), операції АБО (|) і породження (→) записані різні граматики. Яка із цих граматик містить зайві символи?

4.1) A→BC	4.2) I→AB	4.3) I→AB	4.4) I→AB	4.5) I→a
B→a	B→ab | a	C→a | a	B→ab | $	B→ab | a
B→$	A→b | B	A→b	A→b	C→b

5. У системі Intel в форматі double компілятора мови С у восьми байтах пам’яті зберігається наступне значення:

00000000 00000000 00000000 00000000 00000000 00000000 00100001 01000000

Примітка: адреси байтів збільшуються в напряму ліворуч – праворуч.

Визначити, яке число відповідає даному значенню.

5.1) 1.0625;

5.2) 2.125;

5.3) 4.25;

5.4) 8.5;

5.5) 17.

6. М-арне дерево – це дерево, у якому:

6.1) М вузлів;

6.2) М ребер;

6.3) М ярусів;

6.4) кількість ребер, вхідних у вузол дерева, менше або дорівнює М;

6.5) кількість ребер, вихідних з вузла дерева, менше або дорівнює М.

7. Укажіть мінімальну кількість логічних елементів типу «2І-НІ» для апаратної реалізації функції «4АБО»:

7.1) 7 елементів;

7.2) 9 елементів

7.3) 11 елементів;

7.4) це неможливо;

7.5) 12 елементів.

8. В універсальну мову програмування паралельних процесів CC++ (Compositional C++) уведений новий розділ, що включає функції із ключовим

словом **atomic**. Який механізм при взаємодії паралельних процесів реалізує це поняття?

8.1) можливість виконувати одночасне перетворення спільних даних;

8.2) блокування обмінних процесів;

8.3) взаємне виключення при доступі до спільних даних;

8.4) синхронізацію процесів, які передають спільні дані;

8.5) буферизацію набору спільних переданих даних.

9. При виконанні якої команди процесора Intel уміст прапора CF процесора включається в інформаційну частину числа?

9.1) rol;

9.2) rcl;

9.3) shl;

9.4) shr;

9.5) sar.

10. Яка мінімальна кількість контрольних розрядів необхідно для коду Хемінга, що коригує одиночні й виявляє подвійні помилки, для слів, що містять 10 інформаційних розрядів?

10.1) 4;

10.2) 5;

10.3) 6;

10.4) 7;

10.5) 8.

11. У чому особливість вказівників на функції?

11.1) не можна створювати масиви вказівників на функції;

11.2) не можна надавати їхнього значення вказівнику типу void*;

11.3) не можна без розіменування викликати функцію, на яку вказує вказівник;

11.4) не можна використовувати адресну арифметику для вказівників на функції;

11.5) не можна задавати синоніми типу вказівника на функцію за допомогою typedef.

12. За допомогою директиви using можна зробити доступними:

12.1) імена з безіменного простору імен;

12.2) імена, оголошені в просторі імен, що включається;

12.3) тільки імена із простору імен std;

12.4) імена локальних змінних;

12.5) імена вкладених класів.

13. Простори імен можуть знаходитися:

13.1) усередині функцій;

13.2) усередині класів;

13.3) усередині безіменних просторів імен;

13.4) усередині блоків;

13.5) усередині об’єднань.

14. Який з описів функції синтаксично правильний в мові C/C++?

14.1) void f(int x) { int x = 0; }

14.2) void f(int x) { { int x = 0; } }

14.3) void f(int x) { x = 0; return x; }

14.4) void f(int x) { X = 0; }

14.5) void f(int x) { typedef int x; }

15. Яке з наведених тверджень завжди правильне в мові C/C++?

15.1) статична локальна змінна доступна ззовні функції;

15.2) статична локальна змінна ініціалізується щоразу при виклику функції;

15.3) статична локальна змінна може бути жодного разу не проініціалізована;

15.4) статична локальна змінна може бути тільки цілою;

15.5) статичні локальні змінні можуть бути тільки в статичних функціях-елементах.

16. Який з перерахованих прототипів синтаксично правильний у мові C/C++?

16.1) void f(int x, int y = 5, int);

16.2) void f(int, int, int, int y = 5);

16.3) void f(int x; int y = 5);

16.4) void f(int x, y = 5);

16.5) void f(int x, int y = x).

17. Як називається мінімальна одиниця розбиття файлу на твердому магнітному диску з файловою системою FAT?

17.1) біт;

17.2) байт;

17.3) сектор;

17.4) кластер;

17.5) сегмент.

18. Для двох вказівників, що вказують на елементи одного й того самого масиву в мові C, завжди можливе використання результату наступної операції над ними:

18.1) додавання;

18.2) множення;

18.3) ділення;

18.4) віднімання;

18.5) не доступні ніякі операції.

19. Визначити кількість розрядів сигнатури n-розрядної послідовності, отриманої за допомогою сигнатурного аналізатора з утворюючим поліномом ступеня m (n>m):

19.1) m;

19.2) n;

19.3) m+n;

19.4) m*n;

19.5) n-m.

20. Засобами мови C# описана логічна змінна b. Який з наведених виразів синтаксично правильний?

20.1) if(b) then b=true;

20.2) if(b==true) b=false;

20.3) if(!b+1) b=true;

20.4) if!(b) b=true;

20.5) if(b && b) else b++;

21. Який з наведених фрагментів програми дозволяє поміняти місцями значення цілочисельних змінних a і b без застосування додаткових змінних?

21.1)	21.2)	21.3)	21.4)	21.5)
a:=a-b;	a:=a+b;	a:=a+b;	a:=a-b;	a:=a+b;
b:=a+b;	b:=a+b;	b:=a-b;	b:=a+b;	b:=a-b;
a:=a-b;	a:=a-b;	a:=a+b;	a:=a+b;	a:=a-b;

22. Яка із числових послідовностей являє собою послідовність чисел Фібоначчі?

22.1) 1,1,4,10,28,76;

22.2) 1,1,2,3,5,8,13;

22.3) 1,1,4,7,13,22;

22.4) 1,1,4,7,10,13;

22.5) 1,1,5,9,12,15.

23. Укажіть послідовність команд для встановлення прикладної програми в ОС Unix з вільно поширюваних дистрибутивів:

23.1) make, install, clean, configure;

23.2) configure, install, clean, make;

23.3) configure, make, install, clean;

23.4) make, configure, clean, install;

23.5) make, install, configure, clean.

24. У результаті незакінченого переналаштування сегменту однорангової локальної мережі на частині машин була встановлена мережна маска класу B (далі “група B”), на машинах, що залишилися, маска класу C (далі “група C”). У мережі встановлений шлюз для з’єднання з іншими сегментами. Які наслідки це буде мати для локальної мережі?

24.1) Машини груп B и C не будуть мати доступ одна до одної за іменами та адресами, але будуть мати доступ до інших сегментів за адресами.

24.2) Машини групи B будуть мати доступ до машин групи C тільки за адресами, машини групи C не будуть мати доступ до машин групи B. До-

ступ до інших сегментів буде мати група, чия маска збігається з маскою, встановленою на шлюзі.

24.3) Машини групи B будуть мати доступ до групи C за адресами, машини групи C будуть мати доступ до групи B за іменами та адресами, доступ до інших сегментів буде залежати від кількості машин у групі C.

24.4) Групи B и C будуть мати доступ друг до друга тільки за адресами через шлюз, доступ до інших сегментів буде залежати від кількості машин у групах B и C.

24.5) Машини групи B будуть мати доступ до групи C за адресами, машини групи C будуть мати доступ до групи B за іменами та адресами, доступ до інших сегментів порушений не буде.

25. Адміністраторові сервера бази даних було поставлене завдання забезпечити шифрування даних, переданих по відкритому каналі. Адміністратор сформував 1 самопідписаний сертифікат SSL і встановив його на серверній і клієнтській стороні для шифрування даних стороннім програмним забезпеченням. Оцініте підхід системного адміністратора.

25.1) SSL-сертифікат використовується тільки для аутентифікації клієнтської або серверної сторони й не може бути використаний для шифрування даних.

25.2) Використання однакового сертифіката на клієнтській і серверній стороні не дозволить виконати ані аутентифікацію, ані шифрування, тому що шифрування при ініціювання з’єднання однаковими сертифікатами з обох сторін не буде асиметричним.

25.3) SSL-з’єднання буде встановлено, однак відсутність підпису сертифікаційного центра на SSL-сертифікаті не дозволить пройти аутентифікацію для встановлення тунелю незалежно від використовуваного програмного забезпечення.

25.4) Створений сертифікат дозволить виконати аутентифікацію й сервера, й клієнта з наступним шифруванням переданих даних незалежно від фак-

ту наявності підпису сертифікаційного центра та ідентичності сертифікатів.

25.5) Адміністраторові необхідно створити ще один сертифікат і завірити його в сертифікаційному центрі, тільки тоді буде виконуватися як аутентифікація, так і шифрування.

26. Під формальною мовою, що використовують при побудові компіляторів, розуміють:

26.1) скінченну множину букв;

26.2) скінченну множину символів;

26.3) скінченну множину ланцюжків;

26.4) множину букв;

26.5) множину символів.

27. Що характеризує канальна матриця (матриця перехідних ймовірностей)?

27.1) швидкість передавання даних;

27.2) пропускну здатність каналу зв’язку;

27.3) втрати даних в каналі зв’язку;

27.4) ймовірності правильного й помилкового прийому (передавання) даних;

27.5) ентропію повідомлень.

28. Назвіть найбільш ефективний за швидкістю спосіб шифрування, використовуваний у мережах передавання даних:

28.1) гамування;

28.2) системи з відкритим ключем;

28.3) блокове шифрування;

28.4) переставні шифри.

29. Який графічний простір бажано (виходячи з економічної доцільності) використати при розробці програми виводу на кольоровому принтері? Укажіть номер правильної відповіді.

29.1) RGB;

29.2) YIQ;

29.3) HSB;

29.4) Lab;

29.5) YCC;

29.6) CMYK;

29.7) HLS.

30. При якому законі розподілу безперервного, випадкового сигналу з обмеженою середньою потужністю швидкість його передавання по безперервному каналі зв’язку буде максимальною?

30.1) Гауса;

30.2) рівномірної щільності;

30.3) Пуассона;

30.4) Ерланга;

30.5) Лапласа.

31. Що характеризує кодова відстань?

31.1) продуктивність джерела повідомлення;

31.2) довжину кодової комбінації;

31.3) завадостійкість;

31.4) обсяг коду;

31.5) швидкість передавання даних.

32. Які способи задання схем граматики найчастіше використовується при побудові компіляторів:

32.1) символічна;

32.2) форма Наура-Бекуса;

32.3) ітераційна форма;

32.4) синтаксичні діаграми.

33. Необхідно приєднати внутрішню мережу служби, що працює з інформацією, що вимагає підвищеної захищеності, до великої локальної корпоративної мережі. Як зробити це, виключивши можливість виявлення присутності нових машин членами корпоративної мережі й знизити ймовірність злому?

33.1) встановити ще один свіч (селектуючий концентратор) між службою та корпоративною мережею й зробити каскадне підключення;

33.2) встановити роутер між службою та корпоративною мережею й налаштувати на ньому DNS;

33.3) встановити хаб (неселектуючий концентратор) між службою та корпоративною мережею й настроїти всі машини на IP-адреси різних сегментів;

33.4) встановити роутер з проксі-сервером і файрволом між службою та корпоративною мережею;

33.5) налаштувати роутинг на одній з машин, уже підключених до корпоративної мережі фірми й підключити мережа служби через цей роутер.

Завдання

1. Написати програму, що розміщує в області SysTray панелі завдань іконку додатків, що обираються мишею. Здійснити приховування додатка, що відповідає іконці, по натисканню правої клавіші миші, показ по натисканню лівої клавіші миші.
2. Напишіть програму отримання у вікні X Window наступної печатки: у вигляді окружності текстом з підкресленням 'Це моя печатка олімпіаді програмістів 2006', шрифтом – Courier висотою 30 пунктів. У центрі окружності горизонтально шрифтом Arial Italic висотою 80 пунктів написати з підкресленням '2006'. У заголовку вікна написати 'Завдання олімпіади програмістів 2006'.
3. Написати програму, що дозволяє в ОС Windows визначити кількість логічних дисків (і їхні імена A, B, C і т.д.), типи файлових систем (FAT, FAT32, NTFS, EXT2 і т.д.), типи дискових накопичувачів (НГМД, НЖМД, CD-ROM, і т.д.), мітки дисків (томів).
4. Написати програму, що у зображенні, заданому файлом *.xpm, дозволяє користувачеві за допомогою миші виділити фрагмент у вигляді прямокутника, збільшити виділений фрагмент у 2 рази по кожній з координат, змінити контраст у виділеному зображенні.

5. Розробити програму, що дозволяє в середовищі POSIX отримати інформацію про привілеї поточного користувача.
6. Напишіть програму виводу на екран при кожному натисканні на ліву клавішу миші наступної інформації: 1) кількість функціональних клавіш (Fxx) у використовуваній клавіатурі; 2) число натискань кожної з Fxx (парна або непарна кількість разів вона була натиснута з моменту останнього запуску операційної системи).
7. Створити Web-сторінку з текстом "Абырвалг" і з посиланням червоним кольором шрифтом 60 пунктів: ОЛІМПІАДА 2006. При вході в Web-сторінку посилання не повинно бути видимим на екрані. Текст посилання повинен з’являтися тільки після натискання клавіші F12 (за рахунок використання JavaScript). Забезпечити можливість переміщення тексту посилання, що з’явиться, в довільне місце екрана за допомогою миші.
8. Написати програму, що має 2 потоки. В одному із них виконується уведення рядків із клавіатури в буфер, в іншому – зчитування рядків із цього буфера зі зміною регістру символів і виводом їх на екран. Розмір буфера = 5 рядків. Активізація обох потоків виконується в довільні моменти часу.
9. Вивести інформацію про всі мережні ресурси локальної мережі, доступних з даного комп’ютера.
10. Написати програму, що: за допомогою генератора випадкових чисел створює файл на жорсткому диску розміром 100 Кб, що містить довільний набір символів латинського алфавіту; оптимально за швидкодією виконує переведення у цьому файлі малих символів у заголовні з визначенням витраченого процесорного часу.
11. Написати програму визначення й відображення у вікні виведення значення мінімального інтервалу між повідомленнями від програмного таймера (у використовуваній ОС та ПК).
12. Одержати номер версії операційної системи.

13. Написати програму виведення стандартного вікна із кнопкою. Натискання на кнопку мінімізує всі інші відкриті вікна запущених у ПК додатків, крім вікна даної програми.
14. Створіть програму, що відслідковує вміст Clipboard і виводить інформацію з появою заданого рядка символів.
15. Розробити програму переведення чисел з однієї позиційної системи числення в іншу. Основи використовуваних систем числення наступні: 2, 3, 4, 5, 6, 7, 8, 9, 10. Вхідне число, його система числення й система числення вихідного числа задаються користувачем.
16. Створити додаток, що містить одну кнопку. При натисканні на цю кнопку виводиться список всіх динамічних бібліотек, які використовує Windows Explorer.
17. Розробити програму, що одержує й виводить у вікно (на екран) і файл список всіх каталогів на обраному диску у використовуваній операційній системі.
18. В обраній операційній системі складіть програму, що виводить список всіх процесів в операційній системі, з максимумом інформації про кожен з них (ідентифікатор, власник, використовувана пам’ять і т.п.).

Додаток Е

Кваліфікаційні роботи, захищені з окремих аспектів дослідження

– методика навчання мережних технологій та дистанційне навчання:

1. Застосування сучасних мережних технологій в освітній діяльності (О.М. Коваленко, 1999 р.)

2. Використання мережних технологій в навчальному процесі середньої школи (О.О. Тинок, 2000 р.)

3. Методичні основи дистанційного тестування знань засобами FTN-технологій (В.А. Денисюк, 2002 р.)

4. Використання Web-технологій для підтримки курсу алгебри та початків аналізу (Т.А. Скляр, 2002 р.)

5. Мережеві технології в ОС Linux (В.А. Онічкін, 2003 р.)

6. Програмно-апаратне забезпечення мережі навчального закладу (О.О. Осипенко, 2004 р.)

7. Розробка програмного комплексу для дистанційного тестування знань (В.В. Кравченко, 2005 р.)

8. Розробка навчальних курсів в системі дистанційного навчання Moodle (О.А. Гаврилович, 2006 р.)

9. Розробка Інтернет-магазину навчального відео (О.В. Козаченко, 2006 р.)

10. Засоби подання математичної інформації в Інтернет (О.В. Козаченко, 2007 р.)

– методика навчання інтелектуальних систем:

1. Побудова інтелектуальних систем в курсі інформатики середньої школи засобами мови Лісп (О.Г. Жовнуватий, 2004 р.)

2. Вивчення експертних систем в курсі інформатики середньої школи засобами мови Пролог (Л.П. Кузьміна, 2004 р.)

3. Організація даних та знань в інтелектуальних системах, побудованих за принципом «клієнт-сервер» (В.А. Онічкін, 2004 р.)

4. Розробка мінімальної оболонки експертних систем (О.М. Моргун, 2005 р.)

5. Аналіз шахових партій як одна з проблем штучного інтелекту (Д.Г. Медведєв, 2006 р.)

6. Автоматичне розпізнавання друкованих та стилізованих рукописних письмових знаків (А.С. Бойко, 2007 р.)

7. Оптимізація переборних стратегій в логічних іграх (Д.Г. Медведєв, 2007 р.)

8. Розробка експертних систем в середовищі CLIPS (О.О. Немич, 2007 р.)

9. Моделювання мовленнєвої поведінки людини у програмах-співрозмовниках (Т.С. Пазинич, 2007 р.)

10. Розробка Інтернет-співрозмовника (Я.О. Онищенко, 2008 р.)

– *методика навчання операційних систем:*

1. Методична система навчання інформатики у середовищі операційної системи Linux (К.В. Гавриш, 2002 р.)

2. Активізація пізнавальної діяльності студентів педвузів в процесі навчання операційних систем (Є.С. Панкратов, 2002 р.)

3. Розробка програмного комплексу віртуальної лабораторії з дисципліни «Теорія операційних систем» (М.С. Стрюк, 2004 р.).

– *методика навчання системного програмування:*

1. Тестова система з курсу системного програмування в UNIX (А.А. Собещикова, 2003 р.)

2. Програмна підтримка системного програмування в UNIX засобами Free Pascal (Н.О. Єрьома, 2005 р.)

3. Портування файлового менеджеру Midnight Commander на платформу Free Pascal (П.В. Бригінець, 2005 р.)

4. Системне програмування в UNIX засобами Free Pascal: взаємодія процесів (О.В. Кадченко, 2006 р.)

5. Програмна підтримка системного програмування засобами мови Python (О.О. Лещенко, 2006 р.)

– методика навчання системного програмного забезпечення:

1. Розробка інтерпретатора мови LOGO (О.І. Сергєєва, 2005 р.)

2. Побудова інтерпретатора LISP (Т.С. Пазинич, 2006 р.)

– методика навчання об'єктно-орієнтованого програмування:

1. Вивчення об'єктно-орієнтованого програмування в середовищі Squeak (М.М. Лізуненко, 2005 р.)

– методика навчання подіє-орієнтованого програмування:

1. Програмно-методичний комплекс «Програмування в систем X Window мовою Паскаль» (Є.М. Ракова, 2003 р.)

2. Створення мережевих графічних програм в системі X Window (О.В. Яцишина, 2003 р.)

– ІКТ в навчанні фізико-математичних дисциплін:

1. Комп'ютерне забезпечення курсу алгебри і числових систем (А.В. Погорілецька, 1999 р.)

2. Розвиток просторової уяви учнів 10–11 класів на уроках стереометрії засобами комп'ютерно-орієнтованих систем навчання (І.А. Мороз, 2003 р.)

3. Розвиток пізнавальної активності учнів 10–11 класів в процесі навчання алгебри і початків аналізу засобами комп'ютерно-орієнтованих систем навчання (Н.О. Сирота, 2003 р.)

4. Комп'ютерна підтримка курсу астрономії (Г.Б. Захарова, 2003 р.)

5. Використання вітчизняного програмного засобу GRAN1–3D для комп'ютерної підтримки шкільного курсу фізики (А.П. Рядно, 2005 р.)

6. Розробка програмного комплексу для підтримки курсу астрономії у вищій школі (О.В. Кадченко, 2007 р.)

7. Програмна підтримка динамічної геометрії в середовищі Python (І.С. Амброзяк, 2007 р.)

– комп'ютерне моделювання:

1. Побудова регресійних математичних моделей методом структурної мінімізації (О.В. Чорноок, 1999 р.)

2. Розробка демонстраційних стохастичних комп'ютерних моделей для підтримки курсу «Елементи комп'ютерного моделювання» (А.Л. Коробєйніков, 2005 р.)

3. Розробка програми згладжування і прогнозування числових послідовностей на базисі ортогональних поліномів Чебишева (О.М. Кириленко, 2006 р.)

4. Параметрична ідентифікація динамічних систем методом послідовного симплекс-пошуку (В.І. Таранін, 2006 р.)

5. Побудова 3D-моделей в середовищі Python (Н.Ю. Саглаєва, 2006 р.)

6. Розробка компонентно-орієнтованого середовища для моделювання (О.Ю. Горячко, 2008 р.)

7. Об'єктно-орієнтоване моделювання рухів тіл в полі сил тяжіння (Т.М. Лисенко, 2008 р.)

– паралельні та розподілені обчислення:

1. Застосування метакомп'ютингу для розв'язання теоретико-числових проблем (О.В. Макаренко, 2003 р.)

2. Побудова розподіленої системи комп'ютерної алгебри (О.О. Синенко, 2004 р.)

3. Побудова розподіленої системи для розв'язання теоретико-числових проблем (Д.О. Чумак, 2007 р.)

4. Розробка розподіленої системи тестування знань учнів (О.В. Сидорович, 2008 р.)

– системи комп'ютерної математики:

1. Розробка гіпертекстового довідника з системи Maxima (В.В. Вакуленко, 2005 р.)

2. Розробка класу поліномів для математичного комплексу Matlab (К.В. Баканова, 2006 р.)

3. Розробка графічного інтерфейсу до системи комп'ютерної алгебри (С.В. Іванов, 2006 р.)

4. Створення інтерфейсу користувача для системи комп'ютерної математики (І.В. Кривошей, 2006 р.)

5. Інтеграція СКМ Maxima в систему дистанційного навчання (А.І. Антоненко, 2007 р.)

6. Розробка локалізованої версії системи комп'ютерної математики Maxima (В.Е. Керницький, 2007 р.)

7. Розробка графічного інтерфейсу до СКМ Maxima в середовищі Python (А.М. Павленко, 2007 р.)

– *мобільні технології:*

1. Програмно-апаратні засоби підтримки дистанційного навчання (А.С. Бойко, 2008 р.)

2. Системне програмування в середовищі Windows CE.NET (А.І. Антоненко, 2008 р.)

3. Розробка додатків в середовищі Wolf Linux для LBook V3 (М.І. Мороз, 2008 р.)

4. Розробка ALM-модулів для LBook V8 (О.Є. Кокоріна, 2008 р.)

5. Програмування мобільних додатків засобами eVB (Ю.Л. Коломоєць, 2008 р.)

6. Програмування мобільних додатків в середовищі Symbian (М.В. Петрова, 2008 р.)

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Абрамова Н. Т. Границы фундаменталистского идеала и новый образ науки / Абрамова Н. Т. // Философские науки. – 1989. – №11. – С. 39–50.
2. Аданников А. А. Фундаментализация физико-математической подготовки в профессиональном образовании студентов технических вузов : автореф. дис. на соискание ученой степени канд. пед. наук : спец. 13.00.08 «Теория и методика профессионального образования» / Аданников Андрей Анатольевич ; Самарский гос. пед. ун-т – Тольятти, 2000. – 21 с.
3. Амамия М. Архитектура ЭВМ и искусственный интеллект / Амамия М., Танака Ю. – М. : Мир, 1993. – 400 с.
4. Андреев А. А. Дистанционное обучение : сущность, технология, организация / Андреев А. А., Солдаткин В. И. – М. : Изд-во МЭСИ, 1999. – 196 с.
5. Андриевский Б. Р. Элементы математического моделирования в программных средах MATLAB 5 и Scilab / Андриевский Б. Р., Фрадков А. Л. – СПб. : Наука, 2001. – 286 с.
6. Анри Ф. Заочное обучение и коммуникация с помощью ЭВМ / Анри Ф. // Перспективы. Вопросы образования. – 1989. – №1. – С. 82–87.
7. Архангельский С. И. Учебный процесс в высшей школе, его закономерные основы и методы / Архангельский С. И. – М. : Высшая школа, 1980. – 368 с.
8. Асманова И. Ю. Развитие системного мышления студента как условие фундаментализации и профессионализации усваиваемый знаний : дис. ... канд. пед. наук : 13.00.08 «Теория и методика профессионального образования» / Асманова И. Ю. ; Ставропольский гос. ун-т – Ставрополь, 2004. – 178 с.
9. Афанасов В. А. Пути совершенствования молодыми учителями своего педагогического мастерства : автореф. дис. на соискание ученой степени канд. пед. наук / Афанасов В. А. – М., 1961.
10. Афанасьев В. В. Современные проблемы и концепции математического

образования учителя физики / Афанасьев В. В., Смирнов Е. И. // Ярославский педагогический вестник. – 2002. – №1.

11. Бабанский Ю. К. Методы обучения в современной общеобразовательной школе / Бабанский Ю. К. – М. : Просвещение, 1985. – 208 с.
12. Балахонов А. В. Фундаментализация высшего медицинского образования на основе системного естественнонаучного знания : автореф. дис. на соискание ученой степени доктора пед. наук : спец. 13.00.08 «Теория и методика профессионального образования» / Балахонов Алексей Викторович ; Ленингр. гос. обл. ун-т им. А.С. Пушкина – Санкт-Петербург, 2007. – 52 с.
13. Баляева С. А. Теоретические основы фундаментализации общенаучной подготовки в системе высшего технического образования : дис. ... доктора пед. наук : 13.00.01 «Общая педагогика» / Баляева Светлана Анатольевна. – М., 1999. – 458 с.
14. Баранова Е. В. Теория и практика объектно-ориентированного проектирования содержания обучения средствами информационных технологий : автореф. дис. на соискание ученой степени доктора пед. наук : спец. 13.00.02 «Теория и методика обучения информатике» / Баранова Е. В. – СПб., 2000. – 36 с.
15. Бардачева О. П. Восстановление зависимостей по выборкам ограниченного объема / Бардачева О. П., Семериков С. А. // Молодий науковець XXI століття : матеріали Міжнародної науково-практичної конференції (Кривий Ріг, 17–18 листопада 2008 р.). – Кривий Ріг : Видавничий центр КТУ, 2008. – С. 209–211.
16. Бауэр Ф. Л. Информатика : В 2 ч. / Ф. Л. Бауэр, Г. Гооз ; Перевод с нем. М. К. Валиева и др.; Под ред. А. П. Ершова. – 2-е, полностью перераб. и расшир. изд. – М. : Мир, 1990. – Ч. 1. – 324 с. : ил.
17. Бауэр Ф. Л. Информатика : В 2 ч. / Ф. Л. Бауэр, Г. Гооз ; Перевод с нем. М. К. Валиева и др.; Под ред. А. П. Ершова. – 2-е полностью перераб. и расшир. изд. – М. : Мир, 1990. – Ч. 2. – 742 с.
18. Бевз В. Г. Історія математики у фаховій підготовці майбутніх учителів /

Бевз В. Г. ; Національний педагогічний ун-т ім. М.П. Драгоманова. – К. : НПУ імені М.П. Драгоманова, 2005. – 360 с. – Бібліогр. : с. 328-359.

19. Бек К. Экстремальное программирование. Библиотека программиста / Бек К. – СПб. : Питер, 2002. – 224 с.
20. Белошапка В. К. О языках, моделях и информатике / Белошапка В. К. // Информатика и образование. – 1987. – №6. – С. 12–16.
21. Беспалько В. П. Образование и обучение с участием компьютеров (педагогика третьего тысячелетия) / Беспалько В. П. – М. : Московский психолого-социальный институт; Воронеж : МОДЭК, 2002. – 352 с.
22. Бешенков С. А. Развитие содержания обучения информатике в школе на основе понятий и методов формализации : дис. ... доктора пед. наук : 13.00.02 «Теория и методика обучения информатике» / Бешенков С. А. – М., 1994. – 250 с.
23. Бизли Д. М. Язык программирования Python. Справочник / Бизли Д. М. – К. : ДиаСофт, 2000. – 336 с.
24. Блюменау Д. И. Информационный анализ/синтез для формирования вторичного потока документов : учебно-практическое пособие / Блюменау Д. И. – Санкт-Петербург : Профессия, 2002. – 235 с.
25. Болонський процес і кредитно-модульна система організації навчального процесу (методичні рекомендації для викладачів та студентів) / Євдокімов В. І., Микитюк О. М., Харченко Л. П., Луценко В. В. – Харків : ХНУРЕ, 2004. – 40 с.
26. Болонський процес у фактах і документах (Сорбонна – Болонья – Саламанка – Прага – Берлін) / [упорядники : Степко М. Ф., Болюбаш Я. Я., Шинкарук В. Д., Грубіянко В. В., Бабин І. І.] – Тернопіль : Вид-во ТДПУ ім. В. Гнатюка, 2003. – 52 с.
27. Бондаренко З. В. Курс вищої математики з комп’ютерною підтримкою. Диференціальні рівняння : [навч. посіб. з вищ. математики для студ. усіх спец.] / Бондаренко З. В., Клочко В. І. – Вінниця : Вінниц. нац. техн. ун-т, 2004. – 130 с.

28. Бороненко Т. А. Основные направления фундаментализации содержания обучения школьного курса информатики / Бороненко Т. А., Рыжова Н. И. // Материалы восьмой конференция представителей региональных научно-образовательных сетей RELARN-2001. – М., 2001.
29. Бороненко Т. А. Теоретическая модель методической системы подготовки учителя информатики : автореф. дис. на соискание ученой степени доктора пед. наук : спец. 13.00.02 «Теория и методика обучения информатике» / Бороненко Т. А. – М., 1998. – 34 с.
30. Бороненко Т. А. Теоретическая модель системы методической подготовки учителя информатики : дис. ... доктора пед. наук : 13.00.02 / Бороненко Татьяна Алексеевна – СПб., 1997. – 335 с.
31. Бочкин А. И. Методика преподавания информатики / Бочкин А. И. – Минск : Вышэйшая школа, 1998. – 431 с.
32. Брунер Дж. Культура образования / Брунер Джером ; [пер. Л. В. Трубицыной, А. В. Соловьева]. – М. : Просвещение, 2006. – 213 с. – (Серия «Образование : мировой бестселлер» / Московская высш. шк. социальных и экономических наук).
33. Брыксина О. Ф. Основные содержательные линии базового курса информатики [Электронный ресурс] / Брыксина Ольга Федоровна, Овчинникова Ольга Александровна // Материалы Международной научно-практической конференции «Информационные технологии в образовании» («ИТО-Поволжье-2006»). 27–28 апреля 2006 г. – Самара : Самарский филиал МГПУ, 2006. – Режим доступа : http://ito.edu.ru/2006/Samara/I/I-0-4.html
34. Булатов И. С. Теоретические, содержательные и методические основы курса истории информатики в подготовке учителя в педагогическом вузе : автореф. дис. на соискание ученой степени канд. пед. наук : спец. 13.00.02 «Теория и методика обучения информатике» / Булатов И. С. ; Ростовский гос. пед. ун-т – М., 2000. – 22 с.
35. Буракова Г. Ю. О проблеме профессионализации в обучении математике студентов педвузов / Буракова Г. Ю. // Ярославский педагогический вест-

ник. – 2002. – №3 (32).

36. Бурда М. І. Методичні основи диференційованого формування геометричних умінь учнів основної школи : дис. … доктора пед. наук : 13.00.02 «Теорія та методика навчання математики» / Бурда М. І. ; АПН України, Інститут педагогіки. – К., 1994. – 347 с.
37. Бурдье П. Университетская докса и творчество : против схоластических делений / Бурдье П. – Мн. : БГУ, 2006. – (Universitas).
38. Вахович І. М. Функціонування ВНЗ приватної форми власності в умовах становлення соціально орієнтованої ринкової економіки вищої школи / Вахович І. М., Волинчук Ю. В. // Економічні науки. Серія «Економіка та менеджмент». Збірник наукових праць. Луцький державний технічний університет. Випуск 5 (18). – Ч. 1. – Редкол .: відп. ред. д.е.н., професор Герасимчук З. В. – Луцьк, 2008. – 376 с.
39. Велика Хартія Університетів [Електронний ресурс] – Болонья, 18 вересня 1988. – Режим доступу : http://www.magna-charta.org/magna.html
40. Великий тлумачний словник сучасної української мови: 250 000 / Вячеслав Тимофійович Бусел (уклад. і голов. ред.). – К.; Ірпінь : Перун, 2007. – 1736 с.
41. Вербицкий А. А. Активное обучение в высшей школе : контекстный подход / Вербицкий Андрей Александрович. – М. : Высшая школа, 1991. – 204 с.
42. Вища освіта України і Болонський процес : [навчальний посібник] / Степко М. Ф., Болюбаш Я. Я., Шинкарук В. Д., Грубінко В. В., Бабин І. І. ; за редакцією В.Г Кременя. – Тернопіль : Навчальна книга–Богдан, 2004. – 384 с.
43. Відкриття геометрії через комп’ютерні експерименти в пакеті DG : Посібник для вчителів математики / Раков С. А., Горох В. П., Осенков К. О., Думчикова О. В., Костіна О. В, Ларін О. Р., Лисиця В. І., Олійник Т. О., Пікалова В. В. – Харків : Вікторія, 2002. – 136 с.
44. Вовк А. І. Архітектура порталу мобільного навчання / Вовк А. І., Гір-

ник А. В., Неминуща А. Ф., Хоменко О. І., Шокалюк С. В., Теплицький О. І. // Теорія та методика навчання математики, фізики, інформатики : збірник наукових праць : випуск VII : В 3-х томах. – Кривий Ріг : Видавничий відділ НМетАУ, 2008. – Т. 3 : Теорія та методика навчання інформатики. – С. 20–24.

45. Вопросы развития самостоятельности учащихся в процессе воспитания и обучения / Ленингр. гос. пед. ин-т им. А. И. Герцена ; [редкол. : Е. Я. Голант (отв. ред.) [и др.]. – Л., 1965. – 303 с. – (Ученые записки Ленинградского государственного педагогического института им. А.И. Герцена; т. 246).
46. Воронин Ю. А. Компьютеризированные технологии в процессе подготовки учителя / Воронин Ю. А. // Педагогика. – 2003. – №8. – С. 53–59.
47. Габрусєв В. Ю. Вивчаємо комп'ютерні мережі / Габрусєв В. Ю. – К. : Видавничий дім «Шкільний світ», 2005. – 128 с. – (Бібліотека «Шкільного світу»).
48. Габрусєв В. Ю. Зміст і методика вивчення шкільного курсу інформатики на основі вільно поширюваної операційної системи LINUX : дис. ... канд. пед. наук : 13.00.02 «Теорія та методика навчання інформатики» / Габрусєв В. Ю. ; Національний педагогічний ун-т ім. М. П. Драгоманова. – К., 2003. – 221 с.
49. Галузеві стандарти вищої освіти. Напрям підготовки 0101 Педагогічна освіта. Спеціальність 6.010100 Педагогіка і методика середньої освіти. Математика. – К. : Видавництво НПУ імені М. П. Драгоманова, 2003. – 148 с.
50. Гальперин П. Я. Развитие исследований по формированию умственных действий / Гальперин П. Я. // Психологическая наука в СССР. – М. : Изд-во АПН РСФСР, 1959. – Т. 1. – 599 с.
51. Гладун А. Д. Роль фундаментального естественнонаучного образования в становлении специалиста / Гладун А. Д. // Высшее образование в России. – 1994. – №4. – С. 21–23.

52. Глушков В. М. Кибернетика. Вопросы теории и практики / В. М. Глушков – М. : Наука, 1986. – 488 с.
53. Глушков В. М. Основы безбумажной информатики / В. М. Глушков. – 2-е изд., испр. – М. : Наука, 1987. – 552 с.
54. Гончаренко С. У. Український педагогічний словник / Гончаренко С. – К. : Либідь, 1997. – 376 с.
55. Гончаренко С. У. Фундаменталізація освіти як дидактичний принцип / Гончаренко С. // Шлях освіти. – 2008. – №1. – С. 2-6.
56. Гончаренко С. У. Фундаменталізація професійної освіти як дидактичний принцип / Гончаренко С. У. // Теорія і практика управління соціальними системами : філософія, психологія, педагогіка, соціологія : щоквартальний науково-практичний журнал. – 2008. – №2. – С. 87–91.
57. Гончарова О. М. Теоретико-методичні основи особистісно-орієнтованої системи формування інформатичних компетентностей студентів економічних спеціальностей : автореф. дис. на здобуття наук. ступеня доктора пед. наук : спец. 13.00.02 «Теорія та методика навчання інформатики» / Гончарова О. М. ; Національний педагогічний ун-т ім. М. П. Драгоманова. – К., 2007. – 40 с.
58. Горбунова Н. Ю. Университетский комплекс как фактор развития региональной системы непрерывного технического образования : автореф. дис. на соискание ученой степени канд. пед. наук : спец. 13.00.01 «Общая педагогика, история педагогики и образования» / Горбунова Н. Ю. ; Бурятский гос. ун-т – Улан-Удэ, 2006. – 25 с.
59. Горошко Ю. В. Вплив нової інформаційної технології на практичну значимість результатів навчання математики в старших класах середньої школи : дис. … канд. пед. наук : 13.00.02 / Горошко Ю. В. – К., 1993. – 103 с.
60. Горячев А. В. Информатика фундаментальная и прикладная / Горячев Александр // Информатика и образование. – 1998. – №6. – С. 27–30.
61. Грабарь М. И. Применение математической статистики в педагогических

исследованиях. Непараметрические методы / Грабарь М. И., Краснянская К. Я. – М. : Педагогика, 1977. – 136 с.

62. Григорьева М. А. Программа учебного курса «Применение мобильных образовательных систем» / Григорьева М. А. // Вестник МГПУ. Серия «Информатика и информатизация образования». – М. : МГПУ, 2004. – № 1 (2). – С. 28-29.
63. Гриншкун В. В. Языки и методы системного программирования: Типовая программа / Гриншкун В. В. // Типовые программы по информатике и прикладной математике (Для студентов и преподавателей педагогических университетов) / Под редакцией проф. С. Г. Григорьева. – М. : МГПУ, 2005. – С. 71–77.
64. Гриффитс А. GCC. Настольная книга пользователей, программистов и системных администраторов / Гриффитс А. – К. : Диасофт, 2004. – 624 с.
65. Гришко Л. В. Концептуальні підходи до навчання основ програмування у вищій школі / Гришко Л. В. // Науковий часопис НПУ імені М. П. Драгоманова. – Серія 2. – Комп'ютерно-орієнтовані системи навчання : зб. наук. праць. – К. : НПУ ім. М. П. Драгоманова. – №1 (8). – 2004. – С. 134–147.
66. Грэхем Р. Конкретная математика. Основание информатики / Грэхем Р., Кнут Д., Паташник О. ; пер. с англ. – М. : Мир, 1998. – 703 с., ил.
67. Гумбольдт В. фон. О внутренней и внешней организации высших научных заведений в Берлине / Вильгельм фон Гумбольдт // Университетское управление : практика и анализ. – 1998. – №3.
68. Гуменюк А. П. Разработка Интернет-собеседника / Гуменюк А. П., Семериков С. А. // Молодий науковець XXI століття : матеріали Міжнародної науково-практичної конференції (Кривий Ріг, 17–18 листопада 2008 р.). – Кривий Ріг : Видавничий центр КТУ, 2008. – С. 218–221.
69. Державна національна програма «Освіта. Україна XXI століття». – К. : Райдуга, 1994. – 61 с.
70. Державний стандарт освітньої галузі "Технології" (проект) для загальноосвітньої середньої школи / Биков В. Ю., Жалдак М. І., Морзе Н. В., Мос-

тіпан О. І., Рамський Ю. С. // Освіта України. – 2003. – № 3-4. – 10 с.

71. Дидактика средней школы. Некоторые проблемы современной дидактики / Под ред. М. А. Данилова и М. Н. Скаткина. – М. : Просвещение, 1975. – 304 с.
72. Дичківська І. М. Інноваційні педагогічні технології : [навчальний посібник] / Дичківська І. М. – К. : Академвидав, 2004. – 352 с.
73. Доля смартфонов на украинском рынке возросла на 3,5% [Електронний ресурс] – Режим доступу : http://itua.info/news/communications/21061.html
74. Дуда Г. Введение к меморандуму Вильгельма фон Гумбольдта / Дуда Г. // Университетское управление : практика и анализ. – 1998. – №3.
75. Дутка Г. Я. Проблема фундаменталізації економічної освіти в педагогічній науці : навч.-метод. посібник / Дутка Г. Я. – Львів : ЛБІ НБУ, 2004. – 63 с.
76. Ёлгина Л. С. Фундаментализация образования в контексте устойчивого развития общества : Сущность, концептуальные основания : дис. ... канд. филос. наук : 09.00.11 / Ёлгина Л. С. – Улан-Удэ, 2000.
77. Ершов А. П. Избранные труды / Ершов А. П. – Новосибирск : Наука, 1994. – 416 с.
78. Ершов А. П. Компьютеризация школы и математическое образование / Ершов А. П. // Информатика и образование. – 1992. – №5–6. – С. 3–12.
79. Есипов Б. П. Пути совершенствования методов обучения / Есипов Б. П. // Советская педагогика. – 1963. – №12.
80. Євець О. В. Фундаменталізація вищої педагогічної освіти як об’єкт наукового дослідження [Електронний ресурс] / Євець О. В. // Вибрані матеріали Всеукраїнської науково-практичної конференції «Художньо-освітній простір України в контексті новітньої історії». – Київ, 22-23 листопада 2007 року. – Режим доступу : http://www.culturalstudies.in.ua/kns2_7.php
81. Євтєєв В. М. Досвід вивчення інтерактивних Web-технологій в середній школі та педагогічному ВНЗ / Євтєєв В. М., Семеріков С. О., Теплицький І. О. // Рідна школа. – 2004. – №2. – С. 46–47.
82. Євтєєв В. М. Локалізація експертної оболонки CLIPS / Євтєєв В. М., Кра-

вченко В. В., Лінник О. П., Семеріков С. О., Теплицький О. І. // Проблеми підготовки та перепідготовки фахівців у сфері інформаційних технологій : матеріали IV Міжнародної науково-технічної конференції “Комп’ютерні технології в будівництві”. – Київ–Севастополь, 18-21 вересня 2006 р. – Кривий Ріг, 2006. – С. 19–20.

83. Жалдак М. І. Теорія ймовірностей і математична статистика з елементами інформаційної технології / Жалдак М. І., Кузьміна Н. М., Берлінська С. Ю. – К. : Вища школа, 1995. – 352 с.
84. Жалдак М. И. Система подготовки учителя к использованию информационной технологии в учебном процессе : дис. ... в форме науч. доклада доктора пед. наук : 13.00.02 / Жалдак М. И. ; АПН СССР; НИИ содержания и методов обучения. – М., 1989. – 48 с.
85. Жалдак М. І. Елементи стохастики з комп’ютерною підтримкою : посіб. для вчителів / Жалдак М. І., Михалін Г. О. – К. : ДІНІТ, 2001. – 70 с.
86. Жалдак М. І. Інформатика – 7 : експериментальний навчальний посібник для учнів 7 класу загальноосвітньої школи / Жалдак М. І., Морзе Н. В. – К. : ДіаСофт, 2000. – 207 с.
87. Жалдак М. І. Інформатика : навч. посібник / Жалдак М. І., Рамський Ю. С. ; за ред. М. І. Шкіля. – К. : Вища шк., 1991. – 319 с.
88. Жалдак М. І. Комп’ютер на уроках геометрії : посібник для вчителів / Жалдак М. І., Вітюк О. В. – К. : ДІНІТ, 2003. – 168 с.
89. Жалдак М. І. Комп’ютер на уроках математики / Жалдак М. І. – К. : Техніка, 1997. – 303 с.
90. Жалдак М. І. Математика (тригонометрія, геометрія, елементи стохастики) з комп’ютерною підтримкою : навч. посіб. / Жалдак М. І., Грохольська А. В., Жильцов О. Б. – К. : Міжрегіон. акад. упр. персоналом, 2004. – 456 с.
91. Жалдак М. І. Математика з комп’ютером / Жалдак М. І., Горошко Ю. В., Вінниченко Є. Ф. – К. : ДІНІТ, 2004. – 168 с.
92. Жалдак М. І. Методика вивчення основ інформатики та обчислювальної

техніки в педагогічному вузі : учбовий посібник / Жалдак М. І. ; КДПІ ім. О. М. Горького. – К., 1986. – 74 с.

93. Жалдак М. І. Обчислювальна математика : спец. курс факультативних занять у 9-х і 10-х кл. / Жалдак М. І., Ковбасенко Б. С., Рамський Ю. С. – К. : Рад. школа, 1973. – 184 с.
94. Жалдак М. І. Основи теорії і методів оптимізації : навч. посіб. для студ. мат. спец. вищ. навч. закл. / Жалдак М. І., Триус Ю. В. – Черкаси : Брама-Україна, 2005. – 607 с.
95. Жалдак М. І. Педагогічний потенціал комп'ютерно-орієнтованих систем навчання математики / Жалдак М. І. ; Редкол. // Комп'ютерно-орієнтовані системи навчання : зб. наук. праць. – К. : НПУ імені М. П. Драгоманова. – Вип. 7. – 2003. – С. 3–16.
96. Жалдак М. І. Програма курсу з основ інформатики та обчислювальної техніки / Жалдак М. І., Морзе Н. В., Науменко Г. Г. – К. : Перун, 1996. – 23 с.
97. Жалдак М. І. Програма шкільного курсу «Інформатика» для базової школи (7-9 класи) / Жалдак М. І., Морзе Н. В., Науменко Г. Г. // Інформатика. – 2003. – 26 с.
98. Жалдак М. І. Формування інформаційної культури вчителя [Електронний ресурс] / Жалдак М. І., Хомік О. А. – [30 листопада 1998]. – Режим доступу : http://www.icfcst.kiev.ua/SYMPOSIUM/Proceedings/Galdak.doc
99. Жалдак М. І. Чисельні методи математики : посібник для самоосвіти вчителів / Жалдак М. І., Рамський Ю. С. – К. : Рад. школа, 1984. – 206 с.
100. Загальні відомості про вищу освіту в Україні [Електронний ресурс]. – К., 2006. – Режим доступу : http://www.mon.gov.ua/education/higher/higher
101. Закон України «Про вищу освіту» / Верховна Рада України. Інститут законодавства. – К., 2002. – 96 с.
102. Закон України «Про інноваційну діяльність» // Відомості Верховної Ради. – 2002. – №36. – С. 226.
103. Згуровський М. З. Дослідницькі університети : шанс для Європи / Згуров-

ський М. // Дзеркало тижня. – 2006. – №39 (618), 14–20 жовтня.

104. Зубрицька М. О. Вища освіта України буде такою, якою ми її хочемо бачити і якою її зробимо [Електронний ресурс] / Зубрицька Марія // Академічна спільнота. – 17 лютого 2005. – Режим доступу : http://rpl.org.ua/ukr/article;2/
105. Иващенко В. П. Некоторые особенности реализации беспроводного Internet на базе технологии Wi-Fi / Иващенко В. П., Швачич Г. Г. // Проблеми підготовки та перепідготовки фахівців у сфері інформаційних технологій : матеріали V Міжнародної науково-технічної конференції «Комп'ютерні технології в будівництві». – Київ–Севастополь, 18-21 вересня 2007 р. – Кривий Ріг, 2008. – С. 41–43.
106. Ильин В. В. Теория познания. Эпистемология / Ильин В. В. – М. : Изд-во Моск. ун-та, 1994. – 136 с.
107. Исхакова Д. Д. Преемственность непрерывной химической подготовки специалистов в технологическом университете : автореф. дис. на соискание ученой степени канд. пед. наук : спец. 13.00.08 «Теория и методика профессионального образования» / Исхакова Д. Д. ; Казанский гос. технолог. ун-т – Казань, 2003. – 21 с.
108. Ігнатенко М. Я. Активізація навчально-пізнавальної діяльності учнів старших класів при вивченні математики : монографія / Ігнатенко М. Я. – К. : Тираж, 1997. – 300 с.
109. Ігнатенко М. Я. Активізація навчально-пізнавальної діяльності учнів старших класів при вивченні математики : дис. ... доктора пед. наук: 13.00.02 / Ігнатенко М. Я. – К. : 1997. – 355 с.
110. Інформатика. 10-11 класи. Програми для загальноосвітніх навчальних закладів / Жалдак М. І., Морзе Н. В., Мостіпан О. І., Науменко Г. Г. – Кам'янець-Подільський : Абетка–НОВА, 2002. – 80 с.
111. Інформаційні технології в аналітичній геометрії / Раков С. А., Горох В. П., Олійник Т. О., Гармашова Н. М., Якуба М. Р. – Харків : РЦНИТ, 2000. – 192 с.

112. Казанцев С. Я. Дидактические основы и закономерности фундаментализации обучения студентов в современной высшей школе : дис. ... доктора пед. наук : 13.00.01 / Казанцев С. Я. – Казань, 2000. – 295 с.
113. Казанцев С. Я. Дидактические основы фундаментализации обучения в системе высшего образования : монография / Казанцев С. Я. – Казань : Издательство Казанского гос. ун-та, 2000. – 138 с.
114. Калверт Ч. Borland Kylix. Руководство разработчика / Калверт Ч., Калверт М., Кастер Дж., Свот Б. – М. : Вильямс, 2002. – 880 с.
115. Калмыкова И. Р. Портфолио как средство самоорганизации и саморазвития личности / Калмыкова И. Р. // Образование в современной школе. – 2002. – №5. – С. 23–25.
116. Каракозов С. Д. Развитие содержания обучения в области информационно-образовательных систем : подготовка учителей информатики в контексте информатизации образования : [монография] / Каракозов С. Д. ; Под ред. Н. И. Рыжовой. – Барнаул : Изд-во БГПУ, 2005. – 300 с.
117. Касаткин В. Н. Информация, алгоритмы, ЭВМ / Касаткин В. Н. – М. : Просвещение, 1991. – 192 с.
118. Касьянов В. Н. Оптимизирующие преобразования программ. – М.: Наука, 1988. – 336 с.
119. Кинелев В. Г. Контуры системы образования XXI века / Кинелев В. Г. // Информатика и образование. – 2000. – № 5. – С. 2–7.
120. Кинелев В. Г. Об итогах работы высшей школы в 1994 году и основных направлениях ее деятельности в 1995 году / Кинелев В. // Высшее образование в России. – 1995. – №1. – С. 7–27.
121. Кинелев В. Г. Образование и цивилизация / Кинелев В. Г. // Высшее образование в России. – 1996. – №3. – С. 4–12.
122. Кинелев В. Г. Фундаментализация университетского образования / Кинелев В. Г. // Высшее образование в России. – 1994. – № 4. – С. 6-13.
123. Кирютенко Ю. А. Объектно-ориентированное программирование. Язык Smalltalk / Кирютенко Ю. А., Савельев В. А. – М. : Вузовская книга, 2007.

– 328 с.

124. Клепко С. Ф. Автономія університету і міждисциплінарна інтеграція [Електронний ресурс] / Клепко С. Ф. – [2006]. – Режим доступу : http://www.cfh.lviv.ua/seminar2/Klepko.htm
125. Клочко В. І. Використання технології «клієнт-сервер» для побудови навчальних систем / Клочко В. І., Жовтяк І. В. // Вісн. Вінниц. політехн. інту. – 2001. – №3. – С. 117-122.
126. Клочко В. І. Елементи теорії ймовірностей і математичної статистики : навч. посіб. / Клочко В. І., Литвинюк В. П. – Вінниця : Вінниц. нац. техн. ун-т, 2007. – 123 с.
127. Клочко В. І. Застосування новітніх інформаційних технологій при вивченні вищої математики у технічному вузі : навчально-методичний посібник / Клочко В. І. – Вінниця : ВДТУ, 1997. – 300 с.
128. Клочко В. І. Лінійна алгебра. Аналітична геометрія : навч. посіб. / Клочко В. І., Литвинюк В. П. – Вінниця : Вінниц. нац. техн. ун-т, 2006. – 120 с.
129. Клочко В. І. Нові інформаційні технології навчання математики в технічній вищій школі : дис. ... доктора пед. наук : 13.00.02 / Клочко В. І. ; Вінницький держ. технічний ун-т. – Вінниця, 1998. – 396 с.
130. Клочко В. І. Формування методологічної компетентності студентів технічних університетів / Клочко В. І., Клочко Н. О. // Теорія та методика навчання фундаментальних дисциплін у вищій школі : збірник наукових праць. – Випуск V. – Кривий Ріг : Видавничий відділ НМетАУ, 2008. – С. 151–158.
131. Кнорринг В. Г. Столетие Е. Г. Шрамкова и проблема фундаментализации образования / Кнорринг В. Г. // Приборы и системы управления. – 1995. – №6. – С. 1–4.
132. Кобильник Т. П. Компетентнісний підхід при вивченні «математичної інформатики» у педагогічному університеті [Електронний ресурс] / Кобильник Т. П. // Інформаційні технології і засоби навчання. – 2007. – Вип. 2. – Режим доступу до журн. :

http://www.nbuv.gov.ua/e-journals/ITZN/em2/emg.html

133. Кобильник Т. П. Методична система навчання математичної інформатики у педагогічному університеті : автореф. дис... канд. пед. наук : 13.00.02 – теорія та методика навчання (інформатика) / Кобильник Тарас Петрович ; Національний педагогічний ун-т ім. М.П.Драгоманова. – К., 2009. – 20с.
134. Кобильник Т. П. Системи комп'ютерної математики: Maple, Mathematica, Maxima / Тарас Петрович Кобильник. – Дрогобич : Редакційно-видавничий відділ ДДПУ імені Івана Франка, 2008. – 316 с.
135. Кобильник Т. П. Фундаментальність інформатичної освіти / Кобильник Т. П. ; Редкол. // Науковий часопис Національного педагогічного університету імені М.П. Драгоманова. Серія №2. Комп'ютерно-орієнтовані системи навчання : зб. наукових праць. – К. : НПУ імені М.П. Драгоманова. – №5 (12). – 2007. – С. 78–81.
136. Ковалев М. П. Электроника помогает считать. Пособие для учителей. / Ковалев М. П., Шварцбурд С. И. – М.: Просвещение, 1978. – 96 с.
137. Коган А. Г. Некоторые вопросы преподавания программирования в школе с углубленным изучением математики / Коган А. Г., Шень А. Х. // Изучение основ информатики и вычислительной техники в средней школе : опыт и перспективы / Сост. В. М. Монахов [и др.] – М. : Просвещение, 1987. – 192 с. : ил. – (Б-ка учителя математики)
138. Колин К. К. Фундаментальная информатика и качество образования : лекция-доклад / Колин К. К. – М. : Исследовательский центр проблем качества подготовки специалистов, 2001. – 29 с. – (Серия материалов Третьей Всероссийской школы-семинара «Информационные технологии в управлении качеством образования и развитии образовательного пространства»).
139. Комплекс нормативних документів для розроблення складових системи галузевих стандартів вищої освіти : за загальною редакцією В. Д. Шинкарука. – К. : МОН України; Інститут інноваційних технологій і змісту освіти, 2008. – 69 с.

140. Компьютерная технология обучения : словарь-справочник / Под редакцией Гриценко В.И., Довгялло А.М., Савельева А.Я. – К. : Наукова думка, 1992. – 650 с.
141. Кондратенко С. В. Maxima/MathML – новый интерфейс к системе компьютерной алгебры Maxima / Кондратенко С. В., Моисеенко Н. В., Семериков С. А., Теплицкий И. А. // Проблеми підготовки та перепідготовки фахівців у сфері інформаційних технологій : матеріали IV Міжнародної науково-технічної конференції «Комп'ютерні технології в будівництві». – Київ–Севастополь, 18-21 вересня 2006 р. – Кривий Ріг, 2006. – С. 33–34.
142. Кондратьев В. В. Фундаментализация профессионального образования специалиста в технологическом университете / Кондратьев В. В. – Казань : КГТУ, 2000. – 323 с.
143. Кондратьев В. В. Фундаментализация профессионального образования специалиста на основе непрерывной математической подготовки в условиях технологического университета : автореф. дис. на соискание ученой степени доктора пед. наук : спец. 13.00.08 «Теория и методика профессионального образования» / Кондратьев В. В. ; Казанский гос. технологич. ин-т. – Казань, 2000. – 40 с.
144. Кондратьев М. Ю. Социальная психология закрытых образовательных учреждений / Кондратьев М. Ю. – СПб. : Питер, 2005. – 304 с. – (Серия : Детскому психологу).
145. Концепция развития научно-инновационной деятельности в системе Министерства образования Республики Беларусь на 2007–2010 годы [Электронный ресурс] . – [2006]. – Режим доступа : http://www.minedu.unibel.by/modules.php?op=modload&name=UpDownload&file=index&req=viewdownload&cid=11&menuID=M70000000
146. Концепція Державної програми розвитку освіти на 2006-2010 роки // Збірник нормативно-правових документів з питань вищої освіти. – К., 2007. – 87 с.
147. Копаєв О. В. Фундаментальний аспект базового курсу інформатики /

Копаєв О. В., Триус Ю. В. // Сучасний стан і перспективи шкільних курсів математики та інформатики у зв'язку з реформуванням у галузі освіти (Дрогобич, 14–16 листопада 2000 р.) : [всеукраїнська науково-практична конференція] : тези доповідей. – Дрогобич : ДДПУ, 2000. – С. 138–140.

148. Корольский В. В. Информационная система высшего учебного заведения (концептуальный аспект) / Корольский В. В., Семериков С. А. // Інформоенергетичні технології адаптаційних процесів життєдіяльності на початку ІІІ тисячоліття : збірник наукових праць. – Київ–Кривий Ріг : КОЛО, 2001.– С. 253-258.
149. Корсак К. В. Освіта, суспільство, людина в ХХІ столітті : інтегрально-філософський аналіз / Корсак К. В. ; Інститут вищої освіти АПН України. – К.; Ніжин : Видавництво НДПУ ім. М. Гоголя, 2004. – 222 с. – Бібліогр. : с. 197–210.
150. Костикова М. Н. Инновационные процессы в развитии педагогического образования / Костикова М. Н. // Традиции и инновации в системе образования : Гуманитаризация образования : матер. науч.-практ. конф. – Ч. 1. – Чита : Изд-во ЗабГПУ, 1998. – С. 36–41.
151. Кремень В. Г. Вища освіта і наука – пріоритетні сфери розвитку суспільства у ХХІ столітті / Кремень В. Г. // Вища школа. – 2002. – №4–5. – С. 3–33.
152. Крилова Т. В. Наукові основи навчання математики студентів нематематичних спеціальностей : дис. ... доктора пед. наук: 13.00.02 / Крилова Т. В. – К., 1999. – 473 с.
153. Крилова Т. В. Проблеми навчання математики в технічному вузі : монографія / Крилова Т. В. – К. : Вища школа, 1998. – 438 с.
154. Крылова Т. В. Проблемы дистанционного обучения математике / Крылова Т. В., Гулеша Е. М. // Матеріали Всеукраїнської науково-методичної конференції «Проблеми математичної освіти» (ПМО–2005), м. Черкаси. – Черкаси : Вид. від. ЧНУ ім. Б. Хмельницького, 2005. – С. 258–259.

155. Крысько В. Г. Психология и педагогика : Схемы и комментарии / Крысько В. Г. – М. : Владос-Пресс, 2001. – 368 с.
156. Кузнецов А. А. Развитие методической системы обучения информатике в средней школе : автореф. дис. на соискание ученой степени доктора пед. наук : спец. 13.00.02 «Теория и методика обучения информатике» / Кузнецов А. А. ; НИИСиМО АПН СССР. – М., 1988. – 47 с.
157. Кузовлев В. П. Профессиональная подготовка студентов в педагогическом вузе : автореф. дис. на соискание ученой степени доктора пед. наук : спец. 13.00.08 «Теория и методика профессионального образования» / Кузовлев В. П. – М., 1999. – 49 с.
158. Кузьмина Н. В. Профессионализм личности преподавателя и мастера производственного обучения / Кузьмина Н. В. – М. : Высш. шк., 1990. – 119 с.
159. Кук Д. Компьютерная математика / Кук Д., Бейз Г. – М. : Наука, 1990. – 384 с.
160. Куклев В. А. Сущностные характеристики мобильного обучения как педагогической инновации / Куклев В.А. // Мир науки, культуры и образования. – 2008. – №5(12). – С. 204–207.
161. Курганский В. Виртуальные машины – концептуальная и методическая основа информатики / Курганский В. // Информатика и образование. – 1992. – №2. – С. 3–8.
162. Кухтенко А. И. Кибернетика и фундаментальные науки / Кухтенко Александр Иванович. – К. : Наукова думка, 1987. – 140 с.
163. Кыверялг А. А. Методы исследования в профессиональной педагогике / Кыверялг А. А. – Таллинн : Валгус, 1980. – 324 с.
164. Лаптев В. В. Концепция фундаментализации образования в области информатики и ее реализация в педагогическом вузе / Лаптев В. В., Рыжова Н. И. // Известия Российского государственного педагогического университета им. А.И. Герцена. Психолого-педагогические науки (психология, педагогика, теория и методика обучения). – СПб., 2002. – № 2 (3). – С. 124–135.

165. Лаптев В. В. Методическая система фундаментальной подготовки в области информатики : теория и практика многоуровневого педагогического университетского образования / Лаптев В. В., Швецкий М. В. ; РГПУ им. А. И. Герцена. – СПб. : Издательство СПбГУ, 2000. – 505 с.
166. Лаптев В. В. Методическая теория обучения информатике. Аспекты фундаментальной подготовки / Лаптев В. В., Рыжова Н. И., Швецкий М. В. – СПб. : Изд-во С.-Петерб. ун-та, 2003. – 352 с.
167. Лапчик М. П. Информатика и технология : компоненты педагогического образования / Лапчик М. П. // Информатика и образование. – 1992. – №1. – С. 3–6.
168. Лапчик М. П. Методика преподавания информатики : учеб. пособие для студ. пед. вузов / М. П. Лапчик, И. Г. Семакин, Е. К. Хеннер; Под общей ред. М. П. Лапчика. – М. : Академия, 2001. – 624 с.
169. Лапчик М. П. Структура и методическая система подготовки кадров информатизации школы в педагогических вузах : дис. на соискание ученой степени доктора пед. наук в форме научн. докл. : 13.00.02 / Лапчик М. П. – М., 1999. – 82 с.
170. Левченко И. В. Развитие системы методической подготовки учителей информатики в условиях фундаментализации образования вузах : дис. на соискание ученой степени доктора пед. наук : 13.00.02 – теория и методика обучения и воспитания (информатика) / Левченко Ирина Витальевна. – М., 2009. – 46 с.
171. Леднев В. С. Научное образования : развитие способностей к научному творчеству / Леднев В. С. – М. : МГАУ, 2002. – 120 с.
172. Леднев В. С. О теоретических основах содержания обучения информатике в общеобразовательной школе / Леднев В. С., Кузнецов А. А., Бешенков С. А. // Информатика и образование. – 2000. – №2. – С. 13–16.
173. Лейнингем ван И. Освой самостоятельно Python за 24 часа / Иван ван Лейнингем. – М. : Вильямс, 2001. – 448 с.
174. Леонова Н. А. До питання розробки та впровадження системи символьної

математики Maxima у ВНЗ України / Леонова Н. А., Теплицький І. О., Семеріков С. О. // Сборник трудов четвертого научно-методического семинара «Информационные технологии в учебном процессе». – Одесса : ЮГПУ им. К. Д. Ушинского, 2003. – С. 183–185.

175. Линькова В. П. Развитие методической системы обучения информатике на основе информационного и информационно-логического моделирования : автореф. дис. на соискание ученой степени доктора пед. наук : спец. 13.00.02 «Теория и методика обучения информатике» / Линькова В. П. – М., 2000. – 37 с.
176. Ліннік О. П. Об'єктно-орієнтоване моделювання у підготовці майбутніх учителів фізики / Ліннік О. П., Моісеєнко Н. В., Євтєєв В. М., Теплицький І. О., Семеріков С. О. // Збірник наукових праць Кам'янець-Подільського державного університету : серія педагогічна. – Випуск 12 : Проблеми дидактики фізики та шкільного підручника фізики в світлі сучасної освітньої парадигми. – Кам'янець-Подільський : Кам'янець-Подільський державний університет, інформаційно-видавничий відділ, 2006. – С. 127-130.
177. Ліннік О. П. Програмна підтримка комп'ютерного моделювання засобами мови Python / Ліннік О. П., Семеріков С. О., Теплицький І. О., Шокалюк С. В. // Інноваційні технології навчання в сучасній дидактиці вищої школи : Матеріали Другої всеукраїнської науково-практичної конференції 13–16 березня 2007 р. – Полтава, 2007. – С. 57–58.
178. Лутц М. Программирование на Python / Лутц М. – СПб. : Символ-Плюс, 2002. – 1136 с.
179. Любченко К. М. Елементи математичної логіки з комп'ютерною підтримкою : посіб. для вчителів / Любченко К. М., Триус Ю. В. – Черкаси : Черкас. нац. ун-т ім. Б. Хмельницького, 2004. – 87 с.
180. Мазурок И. Е. Использование мобильных коммуникационных устройств в образовательных целях / Мазурок И. Е., Мазурок Т. Л. // Теорія та методика навчання математики, фізики, інформатики : збірник наукових праць. – Випуск V : В 3-х томах. – Кривий Ріг : Видавничий відділ НМетАУ, 2005.

– Т. 3. – С. 175–179.

181. Макаренко Е. В. Использование метакомпьютинга для решения теоретико-числовых проблем / Макаренко Е. В., Семериков С. А. // Збірник праць III Всеукраїнської конференції «Сучасні технології в науці та освіті». – Кривий Ріг : Видавничий відділ КДПУ, 2003. – С. 24–27.
182. Маслоу А. Теория человеческой мотивации // Мотивация и личность. – СПб. : Евразия, 1999. – С. 77–105.
183. Машбиц Е. И. Введение в язык Лого : учеб. пособие для ПТУ / Е. И. Машбиц, В. Н. Каптелинин, Е. Д. Маргулис; Под общ. ред. А. А. Стогния [и др.] – К. : Выща шк., 1989. – 207 с.
184. Машбиц Е. И. Компьютеризация обучения : проблемы и перспективы / Машбиц Е. И. – М. : Знание, 1986. – 80 с.
185. Машбиц Е. И. Психологические основы управления учебной деятельностью / Машбиц Е. И. – К. : Выща школа, 1987. – 224 с.
186. Машбиц Е. И. Психолого-педагогические проблемы компьютеризации обучения / Машбиц Е. И. – М. : Педагогика, 1988. – 192 с.
187. Межвузовский центр маркетинга научно-технических разработок [Электронный ресурс]. – [2006]. – Режим доступа : http://www.icm.by/_private/RusInd.html
188. Методичні вказівки з виконання лабораторних робіт для студентів денної та заочної форми навчання з дисципліни «Архітектура комп'ютерних систем та мереж» (рос. мовою). // Укладач : к.пед.н., доц. С. О. Семеріков. – Кременчук : Інститут економіки та нових технологій, 2003. – 205 с.
189. Методичні вказівки з виконання лабораторних робіт для студентів денної та заочної форми навчання з дисципліни «Архітектура комп'ютерних систем та мереж». Вступ до Асемблера (рос. мовою). // Укладач : к.пед.н., доц. С. О. Семеріков. – Кременчук : Інститут економіки та нових технологій, 2003. – 47 с.
190. Методологические проблемы развития педагогической науки. – М. : Педагогика, 1985. – 240 с.

191. Микешина Л. А. Научная картина мира как мировоззренческая форма знания / Микешина Л. А. // Научная картина мира. – К., 1983. – С. 74–79.
192. Минькович Т. В. Обучение компьютерным технологиям с позиции решения сквозных проблем информатики / Минькович Т. В. // Материалы XI Международной конференции-выставки «Информационные технологии в образовании» («ИТО-2001»). – М., 2001.
193. Михайленко М. Індивідуалізація та фундаменталізація навчального процесу в умовах євроінтеграції (Із Всеукраїнської науково-практичної конференції, що відбулася в Переяславу-Хмельницькому на базі економічного факультету) [Текст] / М. Михайленко. – // Освіта України. – Київ : Міністерство освіти і науки України, академія пед. наук України, Профспілка працівників освіти і науки, ВАК України, Видавництво «Педагогічна преса», 2007. – №45 (15 черв.). – С. 10-11.
194. Михалін Г. О. Професійна підготовка вчителя математики у процесі навчання математичного аналізу / Михалін Г. О. – К. : ДІНІТ, 2003. – 320 с.
195. Михалін Г. О. Формування основ професійної культури вчителя математики у процесі навчання математичного аналізу : дис. ... доктора пед. наук : 13.00.04 / Михалін Г. О. ; Національний педагогічний ун-т ім. М.П. Драгоманова. – К., 2004. – 413 с.
196. Мінтій І. С., Семеріков С. О. Компетентнісний підхід : надбання та напрямки подальшої розробки // Молодий науковець XXI століття : Матеріали Міжнародної науково-практичної конференції (Кривий Ріг, 17–18 листопада 2008 р.). – Кривий Ріг : Видавничий центр КТУ, 2008. – С. 18–20.
197. Мозолин В. В. Информационная подготовка в профессиональном вузе. – М. : Образование и Информатика, 2005. – 128 с.
198. Моисеев Н. Н. Алгоритмы развития / Н. Н. Моисеев ; АН СССР. – М. : Наука, 1987. – 302 с.
199. Монахов В. М. Введение в школу приложений математики, связанных с использованием ЭВМ : автореф. дис. на соискание ученой степени доктора пед. наук. – М., 1973. – 63 с.

200. Монахов В. М. Концепция создания и внедрения новой информационной технологии обучения / Монахов В. М. // Проектирование новых информационных технологий обучения : Сб. – М., 1991. – С. 4–30.
201. Монахов В. М. Технологические основы проектирования и конструирования учебного процесса / В. М. Монахов; Волгогр. гос. пед. ун-т. – Волгоград : Перемена, 1995. – 152 с.
202. Монахов В. М. Что такое новая информационная технология обучения / Монахов В. М. // Математика в школе. – 1990. – № 2. – С. 47–52.
203. Морзе Н. В. Дистанційна технологія як основа сучасних інформаційних технологій у навчанні / Морзе Н. В. // Наук.-метод. центр вищої освіти, Наук.-метод. центр середньої освіти. – Вип. 27 – К., 2003. – С. 64-78.
204. Морзе Н. В. Лабораторний практикум з методики навчання інформатики / Морзе Н. В., Дубова Т. В. – К. : Курс, 2003. – 242 с.
205. Морзе Н. В. Методика навчання інформатики. Ч. 1. Загальна методика навчання інформатики / Морзе Н. В. – К. : Навчальна книга, 2003. – 254 с.
206. Морзе Н. В. Методика навчання інформатики. Ч. 2. Методика навчання інформаційних технологій / Морзе Н. В. – К. : Навчальна книга, 2003. – 288 с.
207. Морзе Н. В. Методика навчання інформатики. Ч. 3. Методика навчання основним послугам глобальної мережі Інтернет / Морзе Н. В. – К. : Навчальна книга, 2003. – 196 с.
208. Морзе Н. В. Методика навчання інформатики. Ч. 4. Методика навчання основам алгоритмізації і програмування / Морзе Н. В. – К. : Навчальна книга, 2003. – 250 с.
209. Морзе Н. В. Основи методичної підготовки вчителя інформатики : монографія / Морзе Н. В. – К. : Курс, 2003. – 372 с.
210. Морзе Н. В. Система методичної підготовки майбутніх вчителів інформатики в педагогічних університетах : дис. ... доктора пед. наук : 13.00.02 / Морзе Н. В. ; Національний педагогічний ун-т ім. М. П. Драгоманова. – К., 2003. – 605 с.

211. Морзе Н. В. Система методичної підготовки майбутніх вчителів інформатики в педагогічних університетах : автореф. дис. на здобуття наук. ступеня доктора пед. наук : спец. 13.00.02 «Теорія та методика навчання інформатики» / Морзе Н. В. ; Національний педагогічний ун-т ім. М.П. Драгоманова. – К., 2003. – 40 с.
212. Навчальні програми для профільного навчання профільного навчання. Інформатика / Жалдак М. І., Морзе Н. В., Мостіпан О. І. [і ін.] – К. : 2003. – 320 с.
213. Національна доктрина розвитку освіти України у XXI столітті // Освіта України. – 2002. – № 6. – С. 3–9.
214. Нікітін В. Спроможність до автономії [Електронний ресурс] / Нікітін В. – [2006]. – Режим доступу : http://www.cfh.lviv.ua/seminar2/Nikitin.htm
215. Ніколаєнко С. М. Вища освіта і наука – найважливіші сфери відповідальності громадянського суспільства та основа інноваційного розвитку [Електронний ресурс] / Ніколаєнко С. М. – 24 березня 2005. – Режим доступу : http://www.mon.gov.ua/education/higher/topic/rzvosv
216. Новиков А. М. Российское образование в новой эпохе : парадоксы наследия, векторы развития / Новиков Александр Михайлович. – М. : Эгвес, 2000. – 272 с.
217. Новые информационные технологии образования: концепция программно-методического обеспечения учебно-воспитательного процесса (ПМО УВП) / [Огородников Е. В., Гладилин А. В., Марусин В. В. и др.; Под ред. Разумовского В. Г., Бобко И. М.]; АПН СССР, НИИ информатики и вычисл. техники. – Новосибирск : Б. и., 1990. – 49 с.
218. Новые педагогические и информационные технологии в системе образования / Е. С. Полат, М. Ю. Бухаркина, М. В. Моисеева, А. Е. Петров. – М. : Академия, 2005. – 272 с.
219. Носырев С. В. О фундаментализации образования как условии целостного мировоззрения / Носырев С. В. // Проблемы учебного процесса в инновационных школах : Сб. науч. тр. – Иркутск : ИГУ, 2004. – Вып. 9. – С. 155-

161.

220. Ньюмен Дж. Г. Идея Университета / Ньюмен Дж. Г. – Мн. : БГУ, 2006. – 208 с. : ил. – (Universitas)
221. О Концепции развития научной и инновационной деятельности в системе Министерства образования на 2007-2010 гг. Пресс-бюллетень Министерства образования РБ. – Апрель 2007 г. – Минск, 2007.
222. Огородников И. Т. Педагогика : учеб. пособие для студентов пед. ин-тов / И. Т. Огородников. – М. : Просвещение, 1968. – 374 с.
223. Ожегов С. И. Словарь русского языка : Около 57 000 слов. [Изд. 10-е, стереотип.] / Ожегов С. И. ; под ред. доктора филолог. наук проф. Н. Ю. Шведовой. – М. : Сов. Энциклопедия, 1973. – 846 с.
224. Окулов С. М. Когнитивная информатика : монография / Окулов С. М. – Киров : Изд-во ВятГУ, 2003. – 224 с.
225. Ольнева А. Б. Вариативный подход к математическому образованию в техническом вузе : автореф. дис. на соискание ученой степени доктора пед. наук : спец. 13.00.08 «Теория и методика профессионального образования» / Ольнева Ангелина Борисовна ; Астраханский гос. тех. ун-т – Ярославль, 2006. – 19 с.
226. Ольнева А. Б. Формирование фундаментальных знаний в системе профессионального образования студентов технических вузов / Ольнева Ангелина Борисовна. – М. : МПГУ, 2003. – 181 с.
227. Ольнева А. Б. Фундаментализация профессионального образования / Ольнева А. Б., Марфин С. Г. – Саратов : Науч. кн., 2004. – 448 с.
228. Ортега-и-Гассет Х. Миссия университета / Хосе Ортега-и-Гассет. – Мн. : БГУ, 2005. – 104 с. : ил. – (Universitas).
229. Основи нових інформаційних технологій навчання : Посібник для вчителів / Гокунь О. О., Жалдак М. І., Машбиць Ю. І. [та ін.] – К. : Віпол, 1997. – 262 с.
230. Основы информатики и вычислительной техники в базовой школе : пособие для учителя / Л. А. Залогова, С. В. Русаков, И. Г. Семакин и др. ; под

ред. И. Г. Семакина ; Гл. упр. образования Перм. обл., Лаб. информатизации образования. – Пермь : Лаб. информатизации образования Перм. обл., 1995. – 282 с.

231. Остапенко А. А. Концентрированное обучение как педагогическая технология : дис. ... канд. пед. наук : 13.00.01 «Общая педагогика» / Остапенко Андрей Александрович. – Краснодар, 1998. – 200 с.
232. Открытые системы : концепция и реальность // Открытые системы. – 1993. – №4. – С. 53–58.
233. Паламарчук В. Ф. Тенденції розвитку інноваційних процесів у вищій освіті України у контексті Європейського вибору / Паламарчук В., Даниленко Л. // Післядипломна освіта в Україні. – 2004. – №1. – С. 39–43.
234. Педагогика / В. А. Сластенин [и др.] – М. : Школа-Пресс, 1998. – 512 с.
235. Пеньков А. В. Использование новой информационной технологии при преподавании математики в старших классах средней школы : Дис… канд. пед. наук : 13.00.02 / Пеньков А. В. – К., 1992. – 171 с.
236. Пермякова О. С. Застосування нейронних мереж у задачах прогнозування / Пермякова О. С., Семеріков С. О. // Молодий науковець XXI століття : матеріали Міжнародної науково-практичної конференції (Кривий Ріг, 17–18 листопада 2008 р.). – Кривий Ріг : Видавничий центр КТУ, 2008. – С. 237–239.
237. Плаксин М. А. «Информатика и системология» – «Пермская версия» сквозного курса информатики с 1-го по 11-ый класс / Плаксин М. А. // Материалы X Международной конференции-выставки «Информационные технологии в образовании» («ИТО-2000»). – М., 2000.
238. Подготовка учителя математики : Инновационные подходы : [учебное пособие] / Афанасьев В. В., Поваренков Ю. П., Смирнов Е. И., Шадриков В. Д. – М. : Гардарики, 2001. – 383 с.
239. Полат Е. С. Дистанционное обучение / Полат Е. С., Моисеева М. В. – М. : Владос, 1998. – 192 с.
240. Полищук А. П. Автоматика : учебное пособие / Полищук А. П., Семери-

ков С. А. – Кривой Рог : Издательский отдел КГПИ, 1999. – 277 с.

241. Полищук А. П. Концепция преподавания курса «Численные методы в объектной методологии» / Полищук А. П., Семериков С. А. // Информационные технологии и информационная безопасность в науке, технике и образовании «Инфотех–2002» : материалы международной научно-практической конференции, 30 сентября – 5 октября 2002 г. – Киев–Севастополь : НТО РЭС Украины, 2002. – С. 117-119.
242. Полищук А. П. Методы вычислений в классах языка С++ : учебное пособие / Полищук А. П., Семериков С. А. – Кривой Рог : Издательский отдел КГПИ, 1999. – 350 с.
243. Полищук А. П. Некоторые особенности программной реализации методов экспериментальной идентификации линейных процессов / Полищук А. П., Семериков С. А. // Комп’ютерне моделювання та інформаційні технології в науці, економіці та освіті : збірник наукових праць : в 2-х томах. – Кривий Ріг : Видавничий відділ КДПУ, 2001. – Т. 1. – С. 202-210.
244. Полищук А. П. О реализации практикума по программированию лексических и синтаксических анализаторов при создании языковых интерпретаторов / Полищук А. П., Семериков С. А. // Теорія та методика навчання математики, фізики, інформатики : збірник наукових праць. – Випуск 4 : в 3-х томах. – Кривий Ріг : Видавничий відділ НМетАУ, 2004. – Т. 3 : Теорія та методика навчання інформатики. – С. 250–259.
245. Полищук А. П. О составлении и реализации учебных планов по кибернетическим дисциплинам в высшей школе / Полищук А. П., Семериков С. А. // Теорія та методика навчання математики, фізики, інформатики : збірник наукових праць. – випуск 3, том 3. – Кривий Ріг : Видавничий відділ НМетАУ, 2003. – С. 273-279.
246. Полищук А. П. Применение Free Pascal для поддержки курса системного программирования в UNIX / Полищук А. П., Семериков С. А. // Проблеми підготовки та перепідготовки фахівців у сфері інформаційних технологій : матеріали IV Міжнародної науково-технічної конференції «Комп’ютерні

технології в будівництві». – Київ–Севастополь, 18-21 вересня 2006 р. – Кривий Ріг, 2006. – С. 48–49.

247. Полищук А. П. Программирование в X Window : учебное пособие / Полищук А. П., Семериков С. А. – Кривой Рог : Издательский отдел КГПУ, 2003. – 192 с.
248. Полищук А. П. Программирование в X Window средствами Free Pascal [Электронный ресурс] : учебное пособие / Полищук А. П., Семериков С. А. – Кривой Рог : Издательский отдел КГПУ, 2005. – 128 с. – Режим доступа : http://rus-linux.net/MyLDP/BOOKS/ProgrX/xwin-contents.shtml
249. Полищук А. П. Системное программирование в UNIX средствами Free Pascal [Электронный ресурс] / Полищук А. П., Семериков С. А. – Кривой Рог : Издательский отдел КГПУ, 2005. – 418 с. – Режим доступа : http://www.interface.ru/home.asp?artId=1617
250. Полищук А. П. Событийно-ориентированное программирование / Полищук А. П., Семериков С. А. – Кривой Рог : КГПУ, 2001. – 336 с.
251. Поліщук О. П. Web-СКМ SAGE у задачах теорії кодування / Поліщук О. П., Шокалюк С. В., Закарлюка І. С. // Комп'ютерні технології в будівництві : матеріали VI Міжнародної науково-технічної конференції «КОМТЕХБУД 2008». – Київ-Севастополь, 9-12 вересня 2008 р. – К. : Міністерство регіонального розвитку та будівництва України, 2008. – С. 101-104.
252. Поліщук О. П. Історія мобільного навчання / Поліщук О. П., Семеріков С. О., Теплицький І. О. // Теорія та методика навчання математики, фізики, інформатики : збірник наукових праць. – Випуск VII : В 3-х томах. – Кривий Ріг : Видавничий відділ НМетАУ, 2008. – Т. 3 : Теорія та методика навчання інформатики. – С. 20–24.
253. Поліщук О. П. Концепція курсу подіє-орієнтованого програмування в системі X Window / Поліщук О. П., Семеріков С. О. // Збірник наукових праць Кам'янець-Подільського державного педагогічного університету : серія педагогічна. – Випуск 8 : Дидактики дисциплін фізико-математичної та

технологічної освітніх галузей. – Кам’янець-Подільський : Кам’янець-Подільський державний педагогічний університет, інформаційно-видавничий відділ, 2002. – С. 164-167.

254. Поліщук О. П. Методичні та організаційні проблеми навчання комп’ютерного програмування у вищих навчальних закладах / Поліщук О. П., Семеріков С. О. // Теорія та методика навчання математики, фізики, інформатики : збірник наукових праць. Випуск VI : В 3-х томах. – Кривий Ріг : Видавничий відділ НМетАУ, 2006. – Т. 3 : Теорія та методика навчання інформатики. – С. 8–11.
255. Поліщук О. П. Побудова інтерфейсу користувача в системі X Window / Поліщук О. П., Семеріков С. О. // Комп’ютерне моделювання та інформаційні технології в науці, економіці та освіті : збірник наукових праць. – Черкаси : Брама ІСУЕП, 2003. – С. 114-116.
256. Поліщук О. П. Програмна підтримка системного програмування засобами мови Python / Поліщук О. П., Семеріков С. О., Теплицький І. О., Бойко А. С. // Інноваційні технології навчання в сучасній дидактиці вищої школи : матеріали Другої всеукраїнської науково-практичної конференції 13–16 березня 2007 р. – Полтава, 2007. – С. 71–72.
257. Поліщук О. П. Розподілені обчислення у Web-СКМ SAGE / Поліщук О. П., Шокалюк С. В., Середа С. В. // Комп’ютерні технології в будівництві : матеріали VI Міжнародної науково-технічної конференції «КОМТЕХБУД 2008». – Київ-Севастополь, 9-12 вересня 2008 р. – К. : Міністерство регіонального розвитку та будівництва України, 2008. – С. 91-92.
258. Поліщук О. П. Систематичне навчання моделюванню в підготовці майбутнього вчителя / Поліщук О. П., Теплицький І. О., Семеріков С. О. // Комп’ютерне моделювання в освіті : матеріали Всеукраїнського науково-методичного семінару. – Кривий Ріг, 26 квітня 2006 р. – Кривий Ріг : КДПУ, 2006. – С. 48-49.
259. Поліщук О. П. Спецкурс з сучасних методів і технологій розробки про-

грамних виробів компонентної архітектури / Поліщук О. П., Семеріков С. О. // Теорія та методика навчання математики, фізики, інформатики : збірник наукових праць. – Випуск VI : в 3-х томах. – Кривий Ріг : Видавничий відділ НМетАУ, 2006. – Т. 3 : Теорія та методика навчання інформатики. – С. 12–15.

260. Постанова Кабінету Міністрів України «Про затвердження Державної програми "Інформаційні та комунікаційні технології в освіті і науці" на 2006-2010 роки» // Офіційний вісник України. – 2005. – № 49. – С. 40.
261. Похлебаев С. М. Методологические и содержательные основы преемственности физики, химии, биологии при формировании фундаментальных естественно-научных понятий : автореф. дис. на соискание ученой степени доктора пед. наук : спец. 13.00.02 / Похлебаев С. М. ; Челяб. гос. пед. ун-т. – Челябинск, 2007. – 48 с.
262. Проблемы компьютерного обучения : [Сб. ст.] / Под общ. ред. В. Г. Разумовского. – М. : Знание, 1986. – 61 с. – (Вып. 2).
263. Програма для загальноосвітніх навчальних закладів «Основи інформатики та обчислювальної техніки» / Жалдак М. І., Морзе Н. В., Науменко Г. Г., Мостіпан О. І. – К. : Шкільний світ, 2001. – 63 с.
264. Програма продовження педагогічного експериментального дослідження всеукраїнського рівня за темою: «Методична система підготовки та підвищення кваліфікації вчителів щодо використання інформаційно-комунікаційних технологій у навчальному процесі за програмою Intel® «Навчання для майбутнього» [Електронний ресурс] / Додаток 1 до наказу МОН України від «24» березня 2009 р. №271. – Режим доступу : http://iteach.com.ua/files/dodatok1_nakaz271.pdf
265. Программа курса «Основы информатики и вычислительной техники» (X–XI классы) // Математика в школе. – 1986. – №3. – С. 49–53.
266. Программирование на языке R-Лисп / Крюков А. П., Радионов А. Я., Таранов А. Ю., Шаблыгин Е. М. – М. : Радио и связь, 1991. – 192 с.
267. Профессионализация предметной подготовки учителя математики в педа-

гогическом вузе : [монография] / Афанасьев В. В., Поваренков Ю. П., Смирнов Е. И., Шадриков В. Д. – Ярославль : Изд-во ЯГПУ, 2000. – 389 с.

268. Пупцев А. Е. Изучение информатики на базовом уровне в республике Беларусь / Пупцев Александр Евгеньевич // Материалы XVII Международной конференции «Применение новых технологий в образовании», 28–29 июня 2006 г. – Троицк : Центр новых педагогических технологий, Байтик, 2006. – С. 49–50.
269. Райхерт Т. Н. Обучение теории информации как средство фундаментализации предметной подготовки будущих учителей информатики : дис. ... канд. пед. наук : 13.00.02 / Райхерт Татьяна Николаевна ; Пермский гос. пед. ун-т – Пермь, 2001. – 168 с.
270. Ракитина Е. А. Теоретические основы построения концепции непрерывного курса информатики / Ракитина Елена Александровна. – М. : Информатика и образование, 2002. – 88 с.
271. Раков С. А. Дослідницький підхід у математичній освіті, пакети динамічної геометрії та динамічні опорні конспекти / Раков С. А. // Комп'ютер у школі і сім'ї», 2005. – № 5. – С. 17–21.
272. Раков С. А. Использование пакета Derive в курсе математики : учебное пособие / Раков С. А., Олейник Т. А., Скляр Е. В. – Харьков : РЦНИТ, 1996. – 160 с.
273. Раков С. А. Компьютерные эксперименты в геометрии / Раков С. А., Горох В. П. – Х. : РЦНИТ. – 1996. – 176 с.
274. Раков С. А. Математична освіта : компетентнісний підхід з використанням ІКТ : моногр. / Раков С. А. – Х. : Факт, 2005. – 360 с.
275. Раков С. А. Формування математичних компетентностей учителя математики на основі дослідницького підходу у навчанні з використанням інформаційних технологій : дис. ... доктора пед. наук : 13.00.02 / Раков С. А. ; Харківський національний педагогічний ун-т ім. Г.С. Сковороди. – Х., 2005. – 516 с.
276. Рамський Ю. С. Вивчення інформаційно-пошукових систем мережі Інтер-

нет : навч.-метод. посіб. / Рамський Ю. С., Рєзіна О. В. – К. : Нац. пед. ун-т ім. М. П. Драгоманова, 2004. – 60 с.

277. Рамський Ю. С. Логічні основи інформатики : навч. посіб. для студ. фіз.-мат. спец. вищих пед. навч. закл. / Рамський Ю. С. – К. : НПУ ім. М.П. Драгоманова, 2003. – 286 с.
278. Рамський Ю. С. Методичні основи вивчення експертних систем у школі / Рамський Ю. С., Балик Н. Р. – К. : Логос, 1997. – 128 с.
279. Рамський Ю. С. Основи програмування (мовою Паскаль) : Короткий курс лекцій. Лабораторн. практикум : [навч. посіб. для студ.] / Рамський Ю. С., Цибко Г. Ю. – К. : Нац. пед. ун-т ім. М. П.Драгоманова, 2004. – 141 с.
280. Рамський Ю. С. Формування інформаційної культури вчителя математики при вивченні методів обчислень у педагогічному вузі / Рамський Ю. С. // Комп'ютерно-орієнтовані системи навчання : збірник наукових праць. – Випуск 2. – К. : НПУ ім. М. П. Драгоманова, 2000. – С. 25-47.
281. Рекомендации по преподаванию информатики в университетах : Пер. с англ. – СПб., 2002. – 188 с.
282. Решетова З. А. Психологические основы профессионального обучения / Решетова З. А. – М. : МГУ, 1985. – 207 с.
283. Розпорядження Кабінету Міністрів України від 18 липня 2007 р. №548-р «Про схвалення Концепції Державної цільової програми «Наука в університетах» на 2008–2012 роки» // Офіційний вісник України від 03.08.2007 – 2007. – № 54. – С. 86.
284. Ростовская Е. Г. Дифференцированное обучение как условие подготовки конкурентоспособного специалиста в системе среднего профессионального образования : автореф. дис. на соискание ученой степени канд. пед. наук : спец. 13.00.08 «Теория и методика профессионального образования» / Ростовская Елена Геннадьевна ; Ставропольский гос. ун-т – Ставрополь, 2005. – 27 с.
285. Русаков С. В. Тестовые задания по базовому курсу информатики / Русаков С. В., Шестакова Л. В. – М. : Чистые пруды, 2006. – 29 с. – (Библио-

течка «Первого сентября». Серия «Информатика»; Вып. 6 (12)).

286. Рыжова Н. И. Основные составляющие содержания обучения, обеспечивающие фундаментальность образования по информатике / Рыжова Н. И. // Проблемы теории и практики управления образованием : материалы межрегиональной конференции. – Часть II. – Барнаул : Изд-во БГПУ, 2002. – С. 215-217.
287. Рыжова Н. И. Развитие методической системы фундаментальной подготовки будущих учителей информатики в предметной области : дис. ... доктора пед. наук : 13.00.02 / Рыжова Н. И. – СПб., 2000. – 429 с.
288. Рыжова Н. И. Развитие методической системы фундаментальной подготовки будущих учителей информатики в предметной области : автореф. дис. на соискание ученой степени доктора пед. наук : спец. 13.00.02 13.00.02 «Теория и методика обучения информатике» / Рыжова Н. И. – СПб., 2000. – 43 с.
289. Садовников Н. В. Теоретико-методологические основы методической подготовки учителя математики в педвузе в условиях фундаментализации образования : автореф. дис. на соискание ученой степени доктора пед. наук : спец. 13.00.02 «Теория и методика обучения математике» / Садовников Николай Владимирович ; Мордовский гос. пед. ин-т им. М. Е. Евсевьева. – Саранск, 2007. – 41 с.
290. Садовничий В. А. Роль университетов в формировании естественнонаучного образования / Садовничий В. А. // Высшее образование в России. – 1993. – № 1. – С. 38–44.
291. Сейдаметова З. С. Методическая система уровневой подготовки будущих инженеров-программистов по специальности «Информатика» : дис. ... доктора пед. наук : 13.00.02 / З. С. Сейдаметова ; НПУ им. М. П. Драгоманова. – К., 2007. – 559 с.
292. Семакин И. Г. Информатика в школе гуманитарного профиля / Семакин И. Г., Хеннер Е. К. // Материалы XI Международной конференции-выставки «Информационные технологии в образовании» («ИТО-2001»). –

М., 2001.

293. Семакин И. Г. Научно-методические основы построения базового курса информатики : автореф. дис. на соискание ученой степени доктора пед. наук : спец. 13.00.02 «Теория и методика обучения информатике» / Семакин И. Г. – Омск, 2002. – 42 с.
294. Семакин И. Г. Преподавание базового курса информатики в средней школе : методическое пособие / Семакин И. Г., Шеина Т. Ю. – М. : Лаборатория базовых знаний, 2000. – 496 с.
295. Семёнов А. Л. Математическая информатика в школе / Семёнов А. Л. // Информатика и образование. – 1995. – №5. – С. 54–58.
296. Семериков С. А. Разработка программного обеспечения для электронной книги lBook V8 / Семериков С. А., Теплицкий И. А., Линник Е. П., Корнилов Г. И. // Комп'ютерні технології в будівництві : матеріали VI Міжнародної науково-технічної конференції «КОМТЕХБУД 2008». – Київ–Севастополь, 9-12 вересня 2008 р. – К. : Міністерство регіонального розвитку та будівництва України, 2008. – С. 122–125.
297. Семеріков С. О. CLIPS : локалізована оболонка експертної системи для вітчизняної системи освіти / Семеріков С. О., Теплицький І. О. – Кривий Ріг, 2006. – 34 с.
298. Семеріков С. О. Maxima 5.13 : довідник користувача / Семеріков С. О. ; за ред. академіка АПН України М. І. Жалдака. – К. : Національний педагогічний ун-т ім. М.П. Драгоманова, 2007. – 48 с.
299. Семеріков С. О. Вільне програмне забезпечення як фактор стабілізації вузівських курсів інформатики / Семеріков С. О., Теплицький І. О. // Інформаційні технології в освіті : матеріали Всеукраїнської науково-практичної конференції (24–26 травня 2006 р.). – Мелітополь : МДПУ, 2006. – С. 55–56.
300. Семеріков С. О. Еволюція та сучасний стан курсу чисельних методів у вищій школі / Семеріков С. О. // Збірник наукових праць Кам'янець-Подільського державного педагогічного університету : Серія педагогічна.

– Випуск 8 : Дидактики дисциплін фізико-математичної та технологічної освітніх галузей. – Кам'янець-Подільський : Кам'янець-Подільський державний педагогічний університет, інформаційно-видавничий відділ, 2002. – С. 189-193.

301. Семеріков С. О. Застосування системи комп'ютерної алгебри Maxima для генерування математичних текстів в системі дистанційного навчання / Семеріков С. О., Теплицький І. О. // Актуальні проблеми психології : Психологічна теорія і технологія навчання. – К. : Міленіум, 2007. – Т. 8, вип. 3. – С. 85-95.
302. Семеріков С. О. Застосування системи комп'ютерної математики Maxima для генерування математичних текстів в системі дистанційного навчання / Семеріков С. О., Теплицький І. О. // Тези доповідей науково-практичної конференції "Нові технології навчання : психологічні аспекти" / За ред. С. Д. Максименка, М. Л. Смульсон. – Житомир : Вид-во ЖДУ ім. І. Франка, 2007. – С. 39–40.
303. Семеріков С. О. Інваріантність до операційної системи та мови програмування як засіб фундаменталізації курсів інформатики у ВНЗ / Семеріков С. О., Теплицький І. О. // Інформаційні технології в освіті, науці і техніці / Матеріали V Всеукраїнської конференції молодих науковців ІТОНТ–2006 : Черкаси, 3–5 травня 2006 р. – Черкаси : ЧНУ, 2006. – С. 140.
304. Семеріков С. О. Методичні аспекти вивчення теми «Основи компіляції» у підготовці майбутнього вчителя інформатики / Семеріков С. О., Теплицький І. О. // Рідна школа. – 2004. – №4. – С. 32–33.
305. Семеріков С. О. Методичні основи вивчення теми «Операційні системи» у підготовці майбутнього вчителя / Семеріков С. О. // Рідна школа. – 2003. – № 1. – С. 44-45.
306. Семеріков С. О. Мобільне навчання в методичній системі фундаментальної інформатичної освіти / Семеріков С. О. // Комп'ютерні технології в будівництві : Матеріали VI Міжнародної науково-технічної конференції «КОМТЕХБУД 2008». – Київ–Севастополь, 9-12 вересня 2008 р. – К. : Мі-

ністерство регіонального розвитку та будівництва України, 2008. – С. 53.

307. Семеріков С. О. Модель Гумбольдта як першоджерело Болонського процесу / Семеріков С. О. // Матеріали Всеукраїнської науково-практичної конференції «Проектування освітніх середовищ як методична проблема» / Укладач : Шарко В. Д. – Херсон : Видавництво ХДУ, 2008. – С. 70–72.
308. Семеріков С. О. Нові засоби дистанційного навчання інформаційних технологій математичного призначення / Семеріков С. О., Теплицький І. О., Шокалюк С. В. // Вісник. Тестування і моніторинг в освіті. – 2008. – №2. – С. 42–50.
309. Семеріков С. О. Оболонка CLIPS як засіб вивчення експертних систем / Семеріков С. О., Теплицький І. О. // Науковий часопис Національного педагогічного університету імені М. П. Драгоманова. – Серія №2. Комп'ютерно-орієнтовані системи навчання : зб. наукових праць / Редкол. – К. : НПУ імені М. П. Драгоманова. – №5 (12). – 2007. – С. 31–36.
310. Семеріков С. О. Огляд інтерфейсів системи комп'ютерної математики Maxima / Семеріков С. О., Теплицький І. О. // Модернізація освіти : пошуки, проблеми, перспективи : Матеріали міжнародної науково-практичної конференції (Київ–Переяслав-Хмельницький, 22–25 травня 2006 року). – Київ–Переяслав-Хмельницький, 2006. – С. 178–181.
311. Семеріков С. О. Основи комп'ютерного моделювання у школі та педагогічному ВНЗ / Семеріков С. О., Теплицький І. О. // Информационные технологии и информационная безопасность в науке, технике и образовании «Инфотех–2004» : материалы международной научно-практической конференция, 20–25 сентября 2004 г. – Киев–Севастополь : НТО РЭС Украины, 2004. – С. 197-207.
312. Семеріков С. О. Побудова найпростішого інтерпретатора в процесі вивчення теми «Основи компіляції» / Семеріков С. О. // Науковий часопис Національного педагогічного університету імені М. П. Драгоманова. – Серія №2 : Комп'ютерно-орієнтовані системи навчання : зб. наукових праць / Редкол. – К. : НПУ імені М. П. Драгоманова. – №4 (11). – 2006. – С. 119–

123.

313. Семеріков С. О. Побудова найпростішої системи тестового контролю знань на основі Web-технологій / Семеріков С. О., Теплицький І. О. // Науковий часопис Національного педагогічного університету імені М. П. Драгоманова. – Серія №2. Комп'ютерно-орієнтовані системи навчання : Зб. наукових праць / Редрада. – К. : НПУ імені М. П. Драгоманова, 2004. – №1 (8) – С. 106-116.
314. Семеріков С. О. Побудова системи дистанційного тестування знань засобами FTN-технологій / Семеріков С. О. // Науковий вісник Ізмаїльського державного гуманітарного університету. Педагогічні науки. – Ізмаїл, 2004. – Вип. 16. – С. 40-45.
315. Семеріков С. О. Побудова системи дистанційного тестування знань засобами FTN-технологій / Семеріков С. О. // Інформаційно-комунікаційні технології у середній і вищій школі : матеріали міжнародної науково-практичної конференції (м. Ізмаїл, 27-29 травня 2004 року). – Київ-Ізмаїл, 2004. – Ч. 2. – С. 137-138.
316. Семеріков С. О. Розробка гіпертекстового довідника з системи Maxima для підтримки факультативного курсу «Комп'ютерні технології в наукових дослідженнях» / Семеріков С. О., Теплицький І. О. // Матеріали міжнародної конференції "PDMU-2005 : проблеми прийняття рішень в умовах невизначеності". 12-17 вересня 2005 р. – Бердянськ, 2005. – С. 96-97.
317. Семеріков С. О. Розробка гіпертекстового довідника з системи Maxima для підтримки факультативного курсу «Комп'ютерні технології в наукових дослідженнях» / Семеріков С. О., Теплицький І. О. // Збірник наукових праць Бердянського державного педагогічного університету (педагогічні науки). – №3. – Бердянськ : БДПУ, 2005. – С. 51-55.
318. Семеріков С. О. Розробка системи символьної математики для системи вищої освіти України / Семеріков С. О. // Формування духовної культури особистості в процесі навчання математики в школі та вищому навчальному закладі : матеріали всеукраїнської науково-практичної конференції

22-24 травня 2003 року. – Луцьк : РВВ «Вежа» Волин. держ. ун-ту ім. Лесі Українки, 2003. – С. 46-47.

319. Семеріков С. О. Стабілізація курсів інформатики як засіб фундаменталізації інформатичних дисциплін / Семеріков С. О. // Рідна школа. – 2008. – №5. – С. 11–12.
320. Семеріков С. О. Фундаменталізація інформатичної освіти у вищій школі / Семеріков С. О. // Міжвузівська науково-практична конференція «Актуальні проблеми технічних, природничих та соціально-гуманітарних наук в забезпеченні цивільного захисту» (3 квітня 2008 року) : тези доповідей. – Черкаси : АПБ ім. Героїв Чорнобиля, 2008. – С. 51.
321. Семеріков С. О. Фундаменталізація навчання інформатичних дисциплін у вищій школі : монографія / Семеріков С. О. ; науковий редактор академік АПН України, д.пед.н., проф. М. І. Жалдак. – Кривий Ріг : Мінерал; К. : НПУ ім. М.П. Драгоманова, 2009. – 340 с.: іл. – Бібліогр.: с. 284–339.
322. Семеріков С. О. Фундіювання змісту навчання як основа фундаменталізації інформатичної освіти / Семеріков С. О. // Наукові записки Тернопільського національного педагогічного університету імені Володимира Гнатюка. Серія : Педагогіка. – 2008. – №8. – С. 71–75.
323. Семеріков С. О. Функціональне програмування в фундаментальній підготовці майбутнього вчителя / Семеріков С. О., Теплицький І. О., Мінтій І. С. // Комп'ютерні технології в будівництві : матеріали VI Міжнародної науково-технічної конференції «КОМТЕХБУД 2008». – Київ–Севастополь, 9-12 вересня 2008 р. – К. : Міністерство регіонального розвитку та будівництва України, 2008. – С. 54–55.
324. Семеріков С. О. Штучний інтелект в курсі інформатики педагогічного ВНЗ / Семеріков С. О., Теплицький І. О. // Інформаційні технології в освіті, науці і техніці : матеріали IV Всеукраїнської конференції молодих науковців ІТОНТ–2004 : Черкаси, 28–30 квітня 2004 р. – Черкаси : ЧНУ, 2004. – Ч. 2. – С. 180-183.
325. Семеріков С. О., Теплицький І. О., Шокалюк С. В. Maxima – система

комп'ютерної математики для вітчизняної системи освіти / Семеріков С. О., Теплицький І. О., Шокалюк С. В. // Науковий часопис Національного педагогічного університету імені М.П. Драгоманова. Серія №2. Комп'ютерно-орієнтовані системи навчання: Зб. наукових праць / Редкол. – К.: НПУ імені М.П. Драгоманова. – №6 (13). – 2008. – С. 32–39.

326. Сергєєв О. В. Фундаменталізація освіти у вищій школі / Сергєєв О. В. // Теорія та методика навчання фундаментальних дисциплін у вищій школі : збірник наукових праць. – Кривий Ріг : Видавничий відділ НМетАУ, 2005. – С. 4–7.
327. Сергієнко І. В. Становлення і розвиток досліджень з інформатики / Сергієнко І. В. – К. : Наукова думка, 1998. – 204 с.
328. Середа С. В. Разработка программного обеспечения для моделирования бот-сетей / Середа С. В., Семериков С. А. // Молодий науковець XXI століття : матеріали Міжнародної науково-практичної конференції (Кривий Ріг, 17–18 листопада 2008 р.). – Кривий Ріг : Видавничий центр КТУ, 2008. – С. 247–249.
329. Сидоренко В. Фундаменталізація професійної підготовки як один із пріоритетних напрямів розвитку вищої освіти в Україні / Сидоренко В., Білевич С. // Вища освіта України. – 2004. – №3. – С. 35–41.
330. Сидоренко Е. В. Методы математической обработки в психологии / Сидоренко Е. В. – СПб. : Речь, 2003. – 350 с.
331. Скоробогатова Н. В. Наглядное моделирование профессионально-ориентированных задач в обучении математике студентов инженерных направлений технических вузов : автореф. дис. на соискание ученой степени канд. пед. наук : спец. 13.00.02 «Теория и методика обучения математике» / Скоробогатова Н. В. ; Ярославский гос. пед. ун-т им. К. Д. Ушинского – Ярославль, 2006. – 25 с.
332. Сластенин В. А. Формирование личности учителя советской школы в процессе профессиональной подготовки / Сластенин В. А. – М. : Просвещение, 1976. – 160 с.

333. Смирнова-Трибульска Є. М. Дистанційне навчання з використанням системи MOODLE : навчально-методичний посібник для студентів вищих педагогічних навчальних закладів / Смирнова-Трибульска Є. М. ; науковий редактор д.пед.н., академік АПН України, проф. М. І. Жалдак. – Херсон : Айлант, 2007. – 492 с.
334. Смирнова-Трибульска Є. М. Інформаційно-комунікаційні технології в професійній діяльності вчителя : посібник для вчителів / Смирнова-Трибульска Є. М. ; науковий редактор д.пед.н., академік АПН України, проф. М. І. Жалдак. – Херсон : Айлант, 2007. – 560 с.
335. Смирнова-Трибульская Е. Н. Основы формирования информатических компетентностей учителей в области дистанционного обучения : монография / Смирнова-Трибульская Е. Н. ; научный редактор академик АПН Украины, д.пед.н., проф. М. И. Жалдак. – Херсон : Айлант, 2007. – 704 с.
336. Смолянинова О. Г. Подготовка бакалавров образования по профилю «Информатика в начальной школе» в классическом университете / Смолянинова О. Г. // Материалы XVII Международной конференции «Применение новых технологий в образовании», 28–29 июня 2006 г. – Троицк : ГОУ ДПО «Центр новых педагогических технологий» Московской области, МОО Фонд новых технологий в образовании «Байтик», 2006. – С. 426–427.
337. Совершенствование обучения младших школьников / [А. М. Пышкало, Л. К. Назарова, Г. А. Фомичева и др.]. – М. : Педагогика, 1984. – 128 с.
338. Создание среды электронного обучения «1 ученик : 1 компьютер» для 21 века. Информационное руководство Intel World Ahead Education. – Intel, 2008. – 32 с.
339. Соколов В. Е. Фундаментальные биологические и экологические исследования / Соколов В. Е. // Вестник РАН. – 1994. – Т. 64. – № 9. – С. 797–809.
340. Соколова Э. Р. Фундаментализация содержания дисциплины «Инженерная графика» в ССУЗ машиностроительного профиля : автореф. дис. на соискание ученой степени канд. пед. наук : спец. 13.00.02 «Теория и мето-

дика обучения и воспитания (общетехнические и специальные дисциплины в средних специальных учебных заведениях)» / Соколова Э. Р. ; Ин-т педагогики и психологии проф. образования РАО – Казань, 2007. – 22 с.

341. Соловйов В. М. Методи математичного моделювання : лабораторний практикум з курсу / Соловйов В. М., Теплицький І. О., Семеріков С. О. – Видання 3-тє, виправлене. – Кривий Ріг–Черкаси, 2003. – 104 с.
342. Соловьев В. Н. Особенности компьютерного моделирования в социально-гуманитарных науках / Соловьев В. Н., Семериков С. А., Теплицкий И. А. // Комп'ютерне моделювання та інформаційні технології в науці, економіці та освіті : збірник наукових праць. – В 2-х томах. – Кривий Ріг : Видавничий відділ КДПУ, 2001. – Т. 1. – С. 230–236.
343. Соловьев В. Н. Синергетический подход к компьютерному моделированию социально-экономических процессов / Соловьев В. Н., Семериков С. А., Теплицкий И. А. // Информационные технологии и информационная безопасность в науке, технике и образовании «Инфотех–2002» : материалы международной научно-практической конференция, 30 сентября – 5 октября 2002 г. – Киев–Севастополь : НТО РЭС Украины, 2002. – С. 61–62.
344. Спикльмайр С. Zope. Разработка Web-приложений и управление контентом / Спикльмайр С., Фридли К., Спикльмайр Д., Брэнд К. – М. : ДМК, 2003. – 464 с.
345. Співаковський О. В. Алгоритмізація та програмування. Енциклопедичне видання : навч.-метод. посіб / Співаковський О. В. – К. : Комп'ютер, 2007. – 128 с.
346. Співаковський О. В. Лінійна алгебра з використанням інформаційних технологій : навч. посіб. для студ. вищ. навч. закл. / Співаковський О. В. – Херсон : Айлант, 2003. – 190 с.
347. Співаковський О. В. Основні задачі проектування комп'ютерних систем підтримки практичної навчальної математичної діяльності / Співаковський О. В., Львов М. С., Гуржій Т. А. // Нові технології навчання : наук.-

метод. зб. – Вип. 33. – Київ, 2002. – С. 24–28.

348. Співаковський О. В. Програмно-педагогічний засіб «Світ лінійної алгебри» / Співаковський О. В. // Вестн. Херсон. гос. техн. ун-та. – 2003. – №3(19). – С. 402-405.
349. Співаковський О. В. Теоретико-методичні основи навчання вищої математики майбутніх вчителів математики з використанням інформаційних технологій : дис. ... доктора пед. наук : 13.00.02 / Співаковський О. В. ; Херсонський держ. ун-т. – К., 2003. – 534 с.
350. Співаковський О. В. Теорія і практика використання інформаційних технологій у процесі підготовки студентів математичних спеціальностей : моногр. / Співаковський О. В. – Херсон : Айлант, 2003. – 228 с.
351. Станіслав Ніколаєнко про використання мобільних телефонів в школі [Електронний ресурс] – 25 травня 2007. – Режим доступу : http://www.loga.gov.ua/oda/about/depart/guon/news/2007/05/25/news_262.html?template=33
352. Стафеев С. К. Разработка и реализация автономных устройств тестирования с централизованной поддержкой через Интернет / Стафеев С. К., Хлебников В. А., Волков С. А., Волков А. М., Мельничук А. П. // Труды Международной научно-методической конференции «Телематика-2001». – 18–21 июня 2001 г. – СПб.: Изд-во СПбГИТМО, ГНИИ ИТТ «Информика», 2001.
353. Степанов А. Г. Объектно-ориентированная модель информатики как предмета обучения [Электронный ресурс] / Степанов Александр Георгиевич // Материалы конференции ИТО-2006. – М., 2006. – Режим доступа : http://ito.edu.ru/2006/Moscow/I/1/I-1-6306.html
354. Степанов А. Г. Объектно-ориентированный подход к отбору содержания обучения информатике / Степанов Александр Георгиевич. – СПб. : Политехника, 2005. – 229 с.
355. Стефанова Н. Л. Теоретические основы развития системы методической подготовки учителя математики в педагогическом вузе : автореф. дис. на

соискание ученой степени доктора пед. наук : спец. 13.00.02 «Теория и методика обучения математике» / Стефанова Н. Л. – СПб., 1996.

356. Столлингс В. Операционные системы / Столлингс В. – 4-е изд. – М. : Вильямс, 2002. – 848 с.
357. Стронгин Р. Г. Инновационный университет : новый подход к управлению / Стронгин Р. Г., Грудзинский А. О. // Высшее профессиональное образование и кадровая политика в современной России. Аналитический вестник Совета Федерации ФС РФ. – 2006. – № 25 (313).
358. Суворова Т. Н. Совершенствование методики изучения информационных технологий в школьном курсе информатики : автореф. дис. на соискание ученой степени канд. пед. наук : спец. 13.00.02 «Теория и методика обучения информатике» / Суворова Татьяна Николаевна ; Вятский гос. гуманит. ун-т – М., 2007. – 22 с.
359. Сузи Р. А. Язык программирования Python : учебное пособие / Сузи Р. А. – М. : ИНТУИТ, БИНОМ. Лаборатория знаний, 2006. – 328 с.
360. Суханов А. П. Информация и прогресс / А. П. Суханов; Отв. ред. А. Л. Симанов; АН СССР, Сиб. отд-ние. – Новосибирск : Наука : Сиб. отд-ние, 1988. – 190 с. – (Сер. «Наука и техн. прогресс»).
361. Талызина Н. Ф. Деятельностный подход к построению модели специалиста / Талызина Н. Ф. // Вестник высшей школы. – 1986. – №3. – С. 10–14.
362. Талызина Н. Ф. Педагогическая психология : учеб. для студ. сред. пед. учеб. заведений / Талызина Н. Ф. – 3-е изд., стереотип. – М. : Академия, 2001. – 288 с.
363. Талызина Н. Ф. Теоретические основы разработки модели специалиста / Талызина Н. Ф. – М. : Знание, 1986. – 108 с.
364. Таненбаум Э. Операционные системы : разработка и реализация [+CD] / Таненбаум Э., Вудхалл А. – 3-е изд. – СПб. : Питер, 2007. – 704 с. : ил. – (Классика CS)
365. Тенденции в реформировании высшего образования, развитии стандартизации и образовательных стандартов высшей школы в странах СНГ : Мо-

нографический сборник научных статей. – М. : Исследовательский центр проблем качества подготовки специалистов, 2007. – 232 с.

366. Теплицкий И. А. Введение в программирование систем искусственного интеллекта на языке Лисп : Лабораторный практикум / Теплицкий И. А., Семериков С. А. – Кривой Рог : КГПУ, 2004. – 88 с.
367. Теплицкий И. А. Использование Web-технологий для организации массового психологического тестирования / Теплицкий И. А., Семериков С. А. // Інформаційні технології в освіті : матеріали Всеукраїнської науково-практичної конференції (24–26 травня 2006 р.). – Мелітополь : МДПУ, 2006. – С. 64.
368. Теплицкий И. А. Личность в информационном обществе / Теплицкий И. А., Евтеев В. Н., Семериков С. А. // Актуальні проблеми духовності : збірник наукових праць. – Випуск 5 (2). – Кривий Ріг : Видавництво СП «Міра», 2004. – С. 179–191.
369. Теплицкий И. А. Создание 3D-моделей физических процессов в среде Python / Теплицкий И. А., Семериков С. А. // Дні науки : зб. тез доповідей : В 3 т. / Гуманітарний університет «ЗІДМУ», 27-28 жовтня 2005; Ред. кол. В. М. Огаренко та ін. – Запоріжжя : ГУ «ЗІДМУ», 2005. – Т. 2. – С. 157-159.
370. Теплицький І. О. «Віртуальний фізичний лабораторний практикум» як актуальна проблема сучасної дидактики / Теплицький І. О., Семеріков С. О. // Теорія та методика навчання математики, фізики, інформатики : збірник наукових праць. – Випуск 4 : В 3-х томах. – Кривий Ріг : Видавничий відділ НМетАУ, 2004. – Т. 2 : Теорія та методика навчання фізики. – С. 414–421.
371. Теплицький І. О. Дослідження дидактичних можливостей мови Лісп як засобу побудови інтелектуальних систем у шкільному курсі інформатики / Теплицький І. О., Семеріков С. О. // Проблеми сучасного підручника : зб. наук. праць / Ред. кол. – К. : Педагогічна думка, 2004. – Вип. 5., Ч. ІІ. – С. 183–191.

372. Теплицький І. О. Дослідження дидактичних можливостей мови Лісп як засобу побудови інтелектуальних систем у шкільному курсі інформатики / Теплицький І. О., Семеріков С. О. // Інформатика та комп'ютерна підтримка навчальних дисциплін у середній і вищій школі : матеріали Всеукраїнської науково-практичної конференції (м. Бердянськ, 23-26 червня 2004 року). – Бердянськ : БДПУ, 2004. – С. 112–115.
373. Теплицький І. О. Елементи комп'ютерного моделювання / Теплицький І. О. – Кривий Ріг : КДПУ, 2005. – 208 с.
374. Теплицький І. О. З досвіду використання Вільного програмного забезпечення у підготовці майбутнього вчителя / Теплицький І. О., Семеріков С. О. // Рідна школа. – 2003. – № 5. – С. 40–41.
375. Теплицький І. О. Задача про політ паперового літачка / Теплицький І. О., Семеріков С. О. // Збірник наукових праць Кам'янець-Подільського державного університету : Серія педагогічна. – Випуск 11 : Дидактика фізики в контексті орієнтирів Болонського процесу. – Кам'янець-Подільський : Кам'янець-Подільський державний університет, інформаційно-видавничий відділ, 2005. – С. 264–272.
376. Теплицький І. О. Інформаційна безпека як нова складова інформаційної культури / Теплицький І. О., Семеріков С. О. // Рідна школа. – 2006. – №2. – С. 63–64.
377. Теплицький І. О. Інформаційне суспільство : гуманістичний аспект / Теплицький І. О., Семеріков С. О. // Науковий часопис Національного педагогічного університету імені М. П. Драгоманова. – Серія №2. Комп'ютерно-орієнтовані системи навчання : Зб. наукових праць / Редкол. – К. : НПУ імені М. П. Драгоманова. – №2 (9). – 2005. – С. 79–88.
378. Теплицький І. О. Комп'ютерна навчальна фізична гра «М'яка посадка» / Теплицький І. О., Семеріков С. О. // Наукові записки : збірник наукових статей Національного педагогічного університету імені М.П. Драгоманова. – К. : НПУ, 2003. – Випуск LIII (53) – С. 347–355.
379. Теплицький І. О. Комп'ютерне моделювання абсолютних та відносних ру-

хів планет Сонячної системи / Теплицький І. О., Семеріков С. О. // Збірник наукових праць Кам'янець-Подільського державного університету : Серія педагогічна : Дидактика фізики і підручники фізики (астрономії) в умовах формування європейського простору вищої освіти. – Кам'янець-Подільський : Кам'янець-Подільський державний університет, інформаційно-видавничий відділ, 2007. – Вип. 13. – С. 211–214.

380. Теплицький І. О. Комп'ютерне моделювання визначальних фізичних експериментів / Теплицький І. О., Семеріков С. О. // Комп'ютерне моделювання та інформаційні технології в науці, економіці та освіті : збірник наукових праць. – Відповід. ред. проф. В. М. Соловйов. – Кривий Ріг : КЕІ ДВНЗ «КНЕУ ім. В. Гетьмана», 2007. – С. 167–170.
381. Теплицький І. О. Комп'ютерне моделювання механічних рухів у середовищі електронних таблиць. Частина 1. Механічні коливання / Теплицький І. О., Семеріков С. О. // Фізика та астрономія в школі. – 2002. – № 5. – С. 40–46.
382. Теплицький І. О. Комп'ютерне моделювання рухів тіл в центральному полі зі змінним потенціалом / Теплицький І. О., Семеріков С. О. // Збірник наукових праць Кам'янець-Подільського державного університету : Серія педагогічна. – Випуск 12 : Проблеми дидактики фізики та шкільного підручника фізики в світлі сучасної освітньої парадигми. – Кам'янець-Подільський : Кам'янець-Подільський державний університет, інформаційно-видавничий відділ, 2006. – С. 313–316.
383. Теплицький І. О. Комп'ютерне моделювання рухів тіл в центральному полі зі змінним потенціалом / Теплицький І. О., Семеріков С. О. // Теорія та методика навчання математики, фізики, інформатики : Збірник наукових праць. – Випуск VI : В 3-х томах. – Кривий Ріг : Видавничий відділ НМетАУ, 2006. – Т. 2 : Теорія та методика навчання інформатики. – С. 69–78.
384. Теплицький І. О. Комп'ютерне моделювання рухів тіл під дією сили всесвітнього тяжіння / Теплицький І. О., Семеріков С. О. // Інформатика та інформаційні технології в навчальних закладах. – 2008. – №1. – С. 85–95.

385. Теплицький І. О. Комп’ютерне моделювання рухів тіл під дією сили всесвітнього тяжіння / Теплицький І. О., Семеріков С. О. // Науковий часопис Національного педагогічного університету імені М. П. Драгоманова. – Серія №2. Педагогічні науки : реалії та перспективи. Випуск 12 : збірник наукових праць / За ред. П. В. Дмитренка, В. Д. Сиротюка. – К. : Вид-во НПУ імені М.П. Драгоманова, 2008. – С. 319–328.
386. Теплицький І. О. Комп’ютерне моделювання руху тіл під дією сили всесвітнього тяжіння / Теплицький І. О., Семеріков С. О. // Збірник наукових праць Кам’янець-Подільського державного університету : Серія педагогічна. – Випуск 10 : Дидактики дисциплін фізико-математичної та технологічної освітніх галузей. – Кам’янець-Подільський : Кам’янець-Подільський державний університет, інформаційно-видавничий відділ, 2004. – С. 166–172.
387. Теплицький І. О. Комп’ютерне моделювання як інтеграційна основа навчання інформатичних дисциплін / Теплицький І. О. // Рідна школа. – 2009. – №2-3. – С. 26–27.
388. Теплицький І. О. Легалізація програмного забезпечення в галузі освіти / Теплицький І. О., Семеріков С. О., Ліннік О. П., Моісеєнко Н. В. // Рідна школа. – 2007. – №2. – С. 28–29.
389. Теплицький І. О. Методика ознайомлення школярів з поняттям фазового простору в курсі фізики / Теплицький І. О., Семеріков С. О. // Збірник наукових праць Кам’янець-Подільського державного університету : Серія педагогічна. Випуск 9 : Методологічні принципи формування фізичних знань учнів і професійних якостей майбутніх учителів фізики та астрономії. – Кам’янець-Подільський : Кам’янець-Подільський державний університет, інформаційно-видавничий відділ, 2003. – С. 163–165.
390. Теплицький І. О. Мобільне навчання : від ООП до OLPC / Теплицький І. О., Поліщук О. П., Семеріков С. О. // Комп’ютерне моделювання та інформаційні технології в науці, економіці та освіті : збірник наукових праць. – Відп. ред. д.ф.-м.н., проф. В. М. Соловйов. – Кривий Ріг : КЕІ

ДВНЗ «КНЕУ ім. В. Гетьмана», 2008. – С. 162–163.

391. Теплицький І. О. Модель мобільного навчання в середній та вищій школі / Теплицький І. О., Семеріков С. О., Поліщук О. П. // Комп'ютерне моделювання в освіті : матеріали III Всеукраїнського науково-методичного семінару. – Кривий Ріг, 24 квітня 2008 р. – Кривий Ріг : КДПУ, 2008. – С. 45–46.
392. Теплицький І. О. Необмежені можливості та можливі обмеження застосувань комп'ютера у фізичному лабораторному експерименті / Теплицький І. О., Семеріков С. О. // Фізика та астрономія в школі. – 2004. – №2. – С. 47–49.
393. Теплицький І. О. Новий технічний засіб навчання – електронна книга / Теплицький І. О., Семеріков С. О., Шокалюк С. В., Ліннік О. П. // Рідна школа. – 2007. – №7–8. – С. 53–54.
394. Теплицький І. О. Основні елементи технології мобільного навчання / Теплицький І. О., Семеріков С. О., Шокалюк С. В. // Інформаційні технології в освіті, науці і техніці : матеріали VI Всеукраїнської конференції молодих науковців ІТОНТ-2008. – Черкаси, 5-7 травня 2008 року. – Черкаси : Вид. від. ЧНУ імені Богдана Хмельницького, 2008. – С. 106–107.
395. Теплицький І. О. Психологічні умови ефективності творчої діяльності з комп'ютерного моделювання / Теплицький І. О., Семеріков С. О. // Проблеми підготовки та перепідготовки фахівців у сфері інформаційних технологій : матеріали V Міжнародної науково-технічної конференції «Комп'ютерні технології в будівництві». – Київ–Севастополь, 18-21 вересня 2007 р. – Кривий Ріг, 2008. – С. 85–86.
396. Теплицький І. О. Психологічні умови ефективності творчої діяльності учнів з комп'ютерного моделювання / Теплицький І. О., Семеріков С. О. // Актуальні проблеми психології : Психологічна теорія і технологія навчання / За ред. С. Д. Максименка, М. Л. Смульсон. – К. : Міленіум, 2007. – Т. 8, вип. 3. – С. 95–109.
397. Теплицький І. О. Розвиток пізнавальної активності учнів 10–11-х класів у

процесі навчання алгебри і початків аналізу засобами комп'ютерно орієнтованих систем навчання / Теплицький І. О., Віхрова О. В., Семеріков С. О. // Рідна школа. – 2004. – №6. – С. 48–49.

398. Теплицький І. О. Розвиток творчих здібностей засобами комп'ютерного моделювання : психолого-педагогічний аспект / Теплицький І. О., Семеріков С. О. // Актуальні проблеми психології : Психологічна теорія і технологія навчання / За ред. С. Д. Максименка, М. Л. Смульсон. – К. : Міленіум, 2005. – Т. 8, вип. 1. – С. 225–232.
399. Теплицький І. О. Системи керування базами даних : навчальний посібник / Теплицький І. О. – В 3-х ч. – Кривий Ріг : КДПУ, 2001.
400. Технология проектирования траектории профессионального становления будущего учителя : (Проектирование учеб. планов и программ для пед. вузов на основе гос. образоват. стандартов) : Учеб. пособие / Волгогр. гос. пед. ун-т и др.; [Монахов В. М. и др.]. – Волгоград; М. : Перемена, 1998. – 83 с.
401. Тищенко Е. А. Организация системы кредитного обучения в техническом вузе : автореф. дис. на соискание ученой степени канд. пед. наук : спец. 13.00.08 «Теория и методика профессионального образования» / Тищенко Евгения Анатольевна ; Южный федеральный ун-т – Ростов-на-Дону, 2007. – 23 с.
402. Товажнянський Л. Л. Болонський процес : цикли, ступені, кредити : [монографія] / Л. Л. Товажнянський, Є. І. Сокол, Б. В. Клименко. – Харків : НТУ «ХПІ», 2004. – 144 с.
403. Триус Ю. В. Віртуальне середовище для дистанційного навчання в Internet / Триус Ю. В., Мещеряков А. П., Коваль Н. О. // Комп'ютерне моделювання та інформаційні технології в науці, економіці та освіті : збірник наукових праць. – Черкаси : Брама ІСУЕП, 2003. – С. 161–165.
404. Триус Ю. В. Інформаційні технології в математичних дослідженнях / Триус Ю. В. // Матеріали тринадцятої наукової сесії Наукового Товариства ім. Шевченка у Черкасах. – Черкаси, 2002. – С. 50–54.

405. Триус Ю. В. Комп’ютерно-орієнтовані методичні системи навчання математичних дисциплін у вищих навчальних закладах : дис. ... доктора пед. наук : 13.00.02 / Триус Ю .В. ; Черкаський нац. ун-т ім. Б. Хмельницького. – Черкаси, 2005. – 649 с.

406. Триус Ю. В. Комп’ютерно-орієнтовані методичні системи навчання математичних дисциплін у вищих навчальних закладах : автореф. дис. на здобуття наук. ступеня доктора пед. наук : спец. 13.00.02 «Теорія та методика навчання інформатики» / Триус Ю. В. ; Національний педагогічний ун-т ім. М. П. Драгоманова. – К., 2005. – 48 с.

407. Триус Ю. В. Комп’ютерно-орієнтовані методичні системи навчання математики : моногр. / Триус Ю. В. – Черкаси : Брама-Україна, 2005. – 400 с.

408. Триус Ю. В. Особливості створення методичної системи навчання основам програмування для підготовки майбутніх інженерів-програмістів / Триус Ю. В., Богатирьов О. О., Гришко Л. В. // Вісник Черкаського університету. Серія Педагогічні науки. – Випуск 35. – Черкаси, 2002. – С. 133–141.

409. Триус Ю. В. Региональный образовательный портал как основной информационный ресурс поддержки непрерывного и открытого образования / Триус Ю. В., Соловьев В. Н., Сердюк А. А., Пискун А. В. // Управляющие системы и машины. – 2004. – №4. – С. 74–81.

410. Триус Ю. В. Система дистанційного навчання освітньо-наукового порталу університету / Триус Ю. В., Беседков С. В., Пустовіт В. А., Бодненко Д. М. // Науковий часопис НПУ імені М. П. Драгоманова. – Серія 2. – Комп’ютерно-орієнтовані системи навчання : зб. наук. праць. – К. : НПУ ім. М. П. Драгоманова. – №3(10). – 2005. – С. 250–266.

411. Указ Президента України від 04.06.05р. № 1013/2005 «Про невідкладні заходи щодо забезпечення функціонування та розвитку освіти в Україні» // Збірник нормативно-правових документів з вищої освіти. – К., 2007. – 87 с.

412. Україна : інтелект нації на межі століть : кол. монографія / Керівник авто-

рського колективу В. К. Врублевський. – К. : Інтелект, 2000. – 516 с.

413. Фокин Р. Р. Метамодель обучения информатике в высшей школе : автореф. дис. на соискание ученой степени доктора пед. наук : 13.00.02 «Теория и методика обучения информатике» / Фокин Р. Р. – СПб., 2000. – 32 с.
414. Формування культури користування мобільним зв'язком (методичні рекомендації) / За загальною редакцією Жебровського Б. М., Литовченко І. В. – К., 2007. – 26 с.
415. Франклин Д. Flash 4 : Анимация в Интернете : [пер. с англ.] / Дерек Франклин, Брукс Паттон. – СПб. : Символ, 2000. – 458 с.
416. Фундаментализация и гуманизация технических университетов : материалы 49-й научно-технической конф. студентов и аспирантов ЮРГТУ / Южно-Российский гос. технический ун-т (Новочеркасский политехнический ин-т) / Е. А. Ныркова (отв. ред.). – Новочеркасск : ЮРГТУ, 2000. – 271 с.
417. Фундаменталізація вищої технічної освіти – необхідна умова випуску конкурентоспроможних фахівців : матеріали міжнар. наук.-метод. конф., 11-13 квітня 2001 року / Національний технічний ун-т «Харківський політехнічний ін-т». – Х. : НТУ «ХПІ», 2001. – 354 с.
418. Хабермас Ю. Идея университета / Хабермас Ю. // Вестник высшей школы. – 1994. – №4. – С. 25.
419. Хантер Р. Основные концепции компиляторов / Хантер Р. – М. : Вильямс, 2002. – 256 с.
420. Харченко Л. Н. Теория и практика биологического образования в современном педагогическом вузе : дис. ... доктора пед. наук : 13.00.08 «Теория и методика профессионального образования» / Харченко Л. Н. – Ставрополь, 2002. – 399 с.
421. Хоменко Л. Г. История отечественной кибернетики и информатики : монография / Хоменко Л. Г. – К. : Ин-т кибернетики им. В.М. Глушкова НАН Украины, 1998. – 455 с.
422. Хуторской А. В. Современная дидактика / Хуторской А. В. – СПб. : Питер, 2001. – 544 с.

423. Частиков А. П. Разработка экспертных систем. Среда CLIPS / Частиков А. П., Гаврилова Т. А., Белов Д. Л. – СПб. : БХВ-Петербург, 2003. – 608 с. : ил.

424. Челак Е. Н. Развивающаяся информатика : методическое пособие / Челак Е. Н., Конопатова Н. К. – М. : Лаборатория Базовых Знаний, 2001. – 208 с.

425. Читалин Н. А. Многоуровневая фундаментализация содержания профессионального образования : автореф. дис. на соискание ученой степени доктора пед. наук : спец. 13.00.01 «Общая педагогика, история педагогики и образования» / Читалин Николай Александрович ; Ин-т пед. и психол. профес. образования РАО. – Казань, 2006. – 39 с.

426. Чумак Д. О. Розробка програмного комплексу для метакомп'ютерних обчислень / Чумак Д. О., Семеріков С. О. // Проблеми підготовки та перепідготовки фахівців у сфері інформаційних технологій : Матеріали V Міжнародної науково-технічної конференції «Комп'ютерні технології в будівництві». – Київ–Севастополь, 18-21 вересня 2007 р. – Кривий Ріг, 2008. – С. 102–103.

427. Швецкий М. В. Методическая система фундаментальной подготовки будущих учителей информатики в педагогическом вузе в условиях двухступенчатого образования : автореф. дис. на соискание ученой степени доктора пед. наук : спец. 13.00.02 – теория и методика обучения и воспитания (информатика) / Швецкий М. В. – СПб., 1994. – 36 с.

428. Швець В. О. Оновлення методичної системи навчання математики / Швець В. О. // Проблеми навчання математики в університеті й школі : тези доповідей науково-методичної конференції математичного факультету. – Донецьк : Донецький державний університет, 1994. – С. 3–6.

429. Шень А. Х. Программирование : теоремы и задачи / Шень А. – М. : МЦНМО, 1995. – 264 с.

430. Шилдт Г. Полный справочник по C : Содержит описание C99-нового стандарта яз. C, одобр. ком. ANSI/ISO : Подробности о яз. и б-ках функ-

ций : Рассмотрены все новейшие средства яз. С, включая применение restrict к указ., встроен. функции, массивы перем. длины, а также операции над комплекс. числами и функции комплекс. перем. : Сотни примеров и прил. / Герберт Шилдт ; [Пер. с англ. А.Г. Беляева и др.]. – 4. изд. – М. [и др.] : Вильямс, 2002. – 699 с.

431. Широких А. А. Методическая система подготовки учителя информатики по основам искусственного интеллекта : автореф. дис. на соискание ученой степени канд. пед. наук : спец. 13.00.02 «Теория и методика обучения информатике» / Широких Анна Александровна ; Пермский гос. пед. ун-т – Омск, 2007. – 23 с.
432. Шлеер С. Объектно-ориентированный анализ : моделирование мира в состояниях / Шлеер С., Меллор С. – К. : Диалектика, 1993. – 240 с.
433. Шнедельбах Г. Университет Гумбольдта / Шнедельбах Г. // Логос. – 2002. – №5–6 (95).
434. Шокалюк С. В. Дистанційне навчання інформаційних технологій математичного призначення у школі / Шокалюк С. В. // Матеріали Всеукраїнської науково-практичної конференції «Проектування освітніх середовищ як методична проблема» / Укладач : Шарко В. Д. – Херсон : Видавництво ХДУ, 2008. – С. 223-224.
435. Шокалюк С. В. Застосування СДН MOODLE для навчання комп’ютерної алгебри / Шокалюк С. В. // Міжвузівська науково-практична конференція «Актуальні проблеми технічних, природничих та соціально-гуманітарних наук в забезпеченні цивільного захисту» (3 квітня 2008 року) : тези доповідей. – Черкаси : АПБ ім. Героїв Чорнобиля, 2008. – С. 56.
436. Шокалюк С. В. Інноваційні інформаційно-комунікаційні технології у післядипломній педагогічній освіті / Шокалюк С. В., Семеріков С. О. // Молодий науковець XXI століття : матеріали Міжнародної науково-практичної конференції (Кривий Ріг, 17–18 листопада 2008 р.). – Кривий Ріг : Видавничий центр КТУ, 2008. – С. 50–52.
437. Шокалюк С. В. Інформаційні технології математичного призначення в ку-

рсі фізики середньої та вищої школи / Шокалюк С. В., Семеріков С. О. // Збірник наукових праць Кам'янець-Подільського національного університету: Серія педагогічна / [редкол.: П.С. Атаманчук (голова, наук. ред.) та ін.]. – Кам'янець-Подільський: Кам'янець-Подільський національний університет, 2008. – Вип. 14: Інновації в навчанні фізики та дисциплін технологічної освітньої галузі: міжнародний та вітчизняний досвід. – С. 108–113.

438. Шокалюк С. В. Інформаційні технології математичного призначення у навчальних та наукових дослідженнях / Шокалюк С. В. // Наукові записки Тернопільського національного педагогічного університету мені Володимира Гнатюка. Серія : Педагогіка. – 2008. – №7. – С. 37–42.
439. Шокалюк С. В. Мобільні технології дистанційного навчання у середній та вищій школі / Шокалюк С. В., Теплицький О. І. // Інформаційні технології в професійній діяльності : матеріали доповідей ІІ Всеукраїнської науково-практичної конференції студентів, аспірантів та науковців 27 березня 2008 року. – Рівне : РДГУ, 2008. – С. 26-27.
440. Шокалюк С. В. Основи роботи в SAGE / Шокалюк С. В. ; за ред. академіка АПН України М. І. Жалдака. – К. : НПУ імені М.П. Драгоманова, 2008. – 64 с.
441. Шокалюк С. В. Програмна підтримка навчальних математичних досліджень засобами систем дистанційного навчання / Шокалюк С. В. // Комп'ютерне моделювання та інформаційні технології в науці економіці та освіті : збірник наукових праць / Відповід. ред. проф. В. М. Соловйов. – Кривий Ріг : КЕІ ДВНЗ „КНЕУ ім. В. Гетьмана", 2007. – С. 208–210.
442. Шокалюк С. В. Разработка графического интерфейса к системе компьютерной математики Maxima в среде Python / Шокалюк С. В., Моисеенко Н. В., Семериков С. А., Теплицкий И. А. // Проблеми підготовки та перепідготовки фахівців у сфері інформаційних технологій : матеріали V Міжнародної науково-технічної конференції «Комп'ютерні технології в будівництві». – Київ–Севастополь, 18-21 вересня 2007 р. – Кривий Ріг,

2008. – С. 108–109.

443. Шокалюк С. В. Розширення можливостей Web-СКМ SAGE / Шокалюк С. В., Руденко Г. Ю. // Комп'ютерні технології в будівництві : матеріали VI Міжнародної науково-технічної конференції «КОМТЕХБУД 2008». – Київ-Севастополь, 9-12 вересня 2008 р. – К. : Міністерство регіонального розвитку та будівництва України, 2008. – С. 87–90.
444. Штырлина И. А. Зарубежный опыт изучения программирования и информатики в средней школе капиталистических стран / Штырлина И. А. // Изучение основ информатики и вычислительной техники в средней школе : опыт и перспективы / Сост. В. М. Монахов [и др.] – М. : Просвещение, 1987. – 192 с. : ил. – (Б-ка учителя математики)
445. Шульман Л.М. Реформа організації науки та процесу взаємодії наука-держава [Електронний ресурс] / Шульман Леонід – 10 травня 2005. – Режим доступу : http://rpl.org.ua/ukr/article;41/
446. Якунин В. А. Педагогическая психология : учеб. пособие / Якунин Валерий Александрович. – СПб. : Полиус, 1998. – 639 с.
447. Ярославский В. В. Разработка и реализация Java-комплекса с элементами триз для творческого навигационного поиска в сети : диссертация ... кандидата физико-математических наук : 05.13.11 / Ярославский Владимир Валерьевич. – Санкт-Петербург, 2001. – 98 с.
448. Ясперс К. Идея университета / Ясперс К. – Мн. : БГУ, 2006. – 159 с. : ил. – (Universitas).
449. Abernathy, D. Get Ready for M-Learning / Abernathy, D. // Training & Development. – 2001. – February. – P. 20-21.
450. ACM Curriculum Committee on Computer Science. Curriculum recommendations for the undergraduate program in computer science // SIGCSE Bulletin (ACM). – 1977. – №2 (June). – P. 1-16.
451. ACM Curriculum committee on computer science. Curriculum'68 : Recommendations for academic programs in computer science // Communications of the ACM. – 1968. – Vol. 11. – №3 (March). – P. 151-197.

452. ACM Curriculum committee on computer science. Curriculum'78 : Recommendations for academic programs in computer science // Communications of the ACM. – 1979. – Vol. 22. – №3 (March). – P. 147-166.
453. Alexander, P. A. College instruction and concomitant changes in students' knowledge, interest, and strategy use : A study of domain learning / Alexander, P. A., Murphy, P. K., Woods, B. S., Duhon, K. E. & Parker, D. // Contemporary Educational Psychology. – 1997. – Vol. 22. – P. 125-146.
454. Alexander, P. A. Learning from text : A multidimensional and developmental perspective / Alexander, P. A. & Jetton, T. L. // Handbook of reading research / Ed. M. L. Kamil, P. B. Mosenthal, P. D. Pearson & R. Barr. – Vol. 3. – Mahwah, NJ : Erlbaum, 2000. – P. 285–310.
455. Anastopoulou, S. Object Manipulation In Educational Multimodal Systems for Contextual Learning / Anastopoulou, S., Barber, C. [et al.] // Proceedings of the Europeain Workshop on Mobile and Contextual Learning, The University of Birmingham, England, 2002. – Birmingham, 2002.
456. Anteboth, M. Organizing Mobile Teaching / Anteboth, M., Tangermann, M. [et al.] // Proceedings of the Europeain Workshop on Mobile and Contextual Learning, The University of Birmingham, England, 2002. – Birmingham, 2002.
457. Attewell, J. Mobile Learning / Attewell, J. // Literacy Today. – 2003. – September, 14.
458. Attewell, J. Mobile technologies and learning : A technology update and m-Learning project summary / Attewell, J. – London : Learning and Skills Development Agency, 2005. – 25 p.
459. Baggaley, J. Portbale Applications in Mobile Education / Baggaley, J. // International Review of Research in Open and Distance Learning. – 2006. – Volume 7, Number 2.
460. Bates, A. W. Educational Multimedia in a Networked Society / Bates, A. W. // Educational Multimedia and Hypermedia. Proceedings of ED-MEDIA World Conference on Educational Multimedia and Hypermedia, 1994.
461. Beale, R. Evaluating m-Learning / Beale, R. // Proceedings of the Europeain

Workshop on Mobile and Contextual Learning, The University of Birmingham, England, 2002. – Birmingham, 2002.

462. Bentley, T. Learning Beyond the Classroom : Education for a Changing World. / Bentley, T. – London : Routledge, 1998.
463. Bershin, J. The Blended Book of Learning / Bershin, J. – San Francisco : Pfeiffer, 2004. – 345 p.
464. Beyer, H. Contextual Design : Defining Customer-Centred Systems / Beyer, H. and Holtzblatt, K. – San Francisco, California : Morgan Kaufmann Publishers Inc., 1998.
465. Big Issues in Mobile Learning : Report of a workshop by the Kaleidoscope Network of Excellence Mobile Learning Initiative / Edited by Mike Sharples. – Nottingham : Learning Sciences Research Institute, 2007. – 40 p.
466. Bijker, W. E. Of Bicycles, Bakerlites, and Bulbs : Toward a Theory of Sociotechnical Change / Bijker, W. E. – Cambridge, Massachusetts : The MIT Press, 1995. – 123 p.
467. Bloom, B.S. The 2 sigma problem : The search for methods of group instruction as effective as one-to-one tutoring / Bloom, B.S. // Educational Researcher. – 1984. – Vol. 13(6). – P. 4-16.
468. Botturi, L. A Framework for the Evaluation of Visual Languages for Instructional Design : the Case of E2ML / Botturi, L. // Journal of Interactive Learning Research. – 2005. – 16 (4). – P. 329–351.
469. Brockbank, A. Facilitating Reflective Learning in Higher Education / Brockbank, A. and McGill I. – Buckingham : Society for Research into Higher Education and Open University Press, 1998.
470. Brown, J. S. Situated cognition and the culture of learning / Brown, J. S., Collins, A. [et al.] // Educational Researcher. – 1989. – January-February 1989. – P. 32-42.
471. Brown, J. S. Sophie : A Step Towards a Reactive Learning Environment / Brown, J. S., Burton, R. R. [et al.] // International Journal of Man-Machine Studies. – 1975. – Vol. 7. – P. 675-696.

472. Bryant, S. Becoming Wikipedian : Transformation of participation in a collaborative online encyclopedia / Bryant, S., Forte, A. & Bruckman, A. // Proceedings of GROUP International Conference on Supporting Group Work. – 2005. – P. 1–10.
473. Buzan, T. Use Your Head / Buzan, T. – London : BBC Books, 1989.
474. Campbell, S. Modeling and Simulation in Scilab/Scicos / Campbell, S., Chancelier, J.-Ph., Nikoukhah, R. – New York : Springer Science, 2006. – XI, 313 p.
475. Canadian students put wireless learning to the test [Electronic resource]. – Mode of access : http://www.bce.ca/en/news/releases/bm/2002/02/26/6832.html
476. Canfield, B. Insider Views on Trends in Global Sourcing [Electronic resource] / Canfield, Bryce. // Outsourcing Journal. – 2005. – November. – Mode of access to the magazine : http://www.outsourcing-journal.com/nov2005-global.html
477. Caudill, J. G. The Growth of m-Learning and the Growth of Mobile Computing : Parallel developments / Caudill, J. G. // International Review of Research in Open and Distance Learning. – 2007. – June, Volume 8, Number 2.
478. Cheverst, K. Developing a context-aware electronic tourist guide : some issues and experiences / Cheverst, K., Davies, N. [et al.] // CHI'2000. – New York : ACM, 2000.
479. Communication from the Commission : E-Learning – Designing “Tejas at Niit” tomorrow’s education [Electronic resource]. – Brussels : European Commission, 2006. – Mode of access : http://ec.europa.eu/education/programmes/elearning/doc_en.html
480. Computing curricula 1991 // Communications of the ACM. – 1991. – Vol. 34. – №6. – P. 69-84.
481. Corlett, D. Innovating with OVID / Corlett, D. // Interactions. – 2000. – Vol. 7. – P. 19-26.
482. Dewey, J. Democracy And Education : An Introduction to the Philosophy of Education / Dewey, J. – New York : Free Press, 1997. – 384 p.
483. Diehl, S. Distributed virtual worlds : Found. a. implementation techn. using VRML, Java, a. CORBA / Stephan Diehl. – Berlin [etc.] : Springer, Cop. 2001.

– XII+166 p.

484. Downes, S. E-learning 2.0 [Electronic resource] / Downes, S. // eLearn Magazine. – 2005. – №10. – Mode of access to the magazine : http://www.elearnmag.org/subpage.cfm?section=articles&article=29-1
485. Draft Standard for Information Technology – Portable Operating System Interface (POSIX) Part 2 : Shell and Utilities (IEEE P1003.2 Draft 11.2 – September 1991)
486. Draper, S. W. Electronically enhanced classroom interaction [Electronic resource] / Draper, S. W. // Australian journal of educational technology. – 2002. – №18. – Mode of access to the magazine : http ://www.psy.gla.ac.uk/~steve/ilig/handsets.html#Abstract
487. Druin, A. KidPad : A Design Collaboration Between Children, Technologists, and Educators / Druin, A., Stewart, J. [et al.] // Proceedings of CHI'97. – Atlanta, Georgia : ACM/Addison-Wesley, 1997. – P. 463-470.
488. Engeström, Y. Activity theory as a framework for analyzing and redesigning work / Engeström, Y. // Ergonomics. – 2000. – Vol. 43, No. 7. – P. 960–974.
489. Engeström, Y. Learning by expanding : An activity-theoretical approach to developmental research / Engeström, Y. – Helsinki : Orienta-Konsultit, 1987.
490. Farias, A. Building software agents for training systems : a case study on radiotherapy treatment planning / Farias, A. and Arvanitis, T. N. // Knowledge-Based Systems. – 1997. – Vol. 10. – P. 161-168.
491. Feiner, S. A Touring machine : prototyping 3D mobile augmented reality systems for exploring the urban environment / Feiner, S., MacIntyre, B. [et al.] // Digest of Papers of the First International Symposium on Wearable Computers. – Los Alamitos, CA : IEEE Computer Society, 1993. – P. 74-81.
492. Fountain, A. Volume 6A : Motif Programming Manual for Motif 2.1, Open Source Edition / Fountain, A., Huxtable, J., Ferguson, P. and Heller, D. – O'Reilly & Associates, 2001. – 975 p.
493. Fozdar, B. I. Mobile Learning and Student Retention / Fozdar, B. I., Kumar, L. S. // International Review of Research in Open and Distance Learning. –

2007. – June, Volume 8, Number 2.

494. Freeman, L. C. The Development of Social Network Analysis : A Study in the Sociology of Science / Linton C. Freeman. – Vancouver : Booksurge Publishing, 2004. – 205 p.
495. Georgiev, T. M-learning – a New Stage of E-Learning / Georgiev, T., Georgieva, E., Smrikarov, A. // Proceedings of the 5th International Conference on Computer Systems and Technologies – CompSysTech'2004. – Rousse, 2004. – P. IV.28-1 – IV.28-5.
496. Georgiev, T. Transitioning from e-Learning to m-Learning : Present issues and future challenges / Georgiev, T., Georgieva, E., & Trajovski, G. // Proceedings of the Seventh ACIS International Conference on Software Engineering, Artifical Intelligence, Networking, and Parallel/ Distributed Computing (SNPD '06).
497. Getting Started Guide to Wireless Networking. – University of Birmingham, 2003. – 21 p.
498. Gomez, S. Scroll to 'E' for Education / Gomez, S. // The Times Higher Education Supplement. – 2007. – Vol. 1780. – P. 13.
499. Guzdial, M. Squeak : Object-Oriented Design with Multimedia Applications [Electronic resource] / Guzdial, M. – [1994]. – Mode of access to the book : http://guzdial.cc.gatech.edu/squeakbook/
500. Heath, B. P. Project Numina : Enhancing Student learning with Hanheld Computers / Barbara P. Heath, Russell L. Herman, Gabriel G. Lugo, James H. Reeves, Ronald J. Vetter, Charles R. Ward // Computer. – 2005. – June. – P. 46–53.
501. Holme, O. Implementing a Student Learning Organiser on the Pocket PC Platform / Holme, O. and Sharples, M. // Proceedings of European Workshop on Mobile and Contextual Learning. – Birmingham, UK, 2002. – P. 41-44.
502. Inkpen, K. M. Designing Handheld Technologies for Kids / Inkpen, K. M. // Personal Technologies. – 2000. – 3(1&2). – P. 81-89.
503. Jenkins, H. Convergence Culture : Where Old and New Media Collide / Jenkins, H. – New York : NYU Press, 2006.

504. Johnston, A. N. An assistant for crew performance assessment / Johnston, A. N., Rushby, N. [et al.] // International Journal of Aviation Psychology. – 2000. – Vol. 10(1). – P. 99-108.

505. Kay, A. Personal Dynamic Media / Kay, A., Goldberg A. // IEEE Computer. – 1977. – Vol. 10(3). – P. 31-41.

506. Keegan, D. M-learning : how far has it come [Electronic Resource] / Desmond Keegan // Mlearning – The Future of Mobile? Proc. Ericsson Education Conference, September 9th 2005. – Mode of access : http://learning.ericsson.net/mlearning2/files/conference/dkeegan.pdf

507. Keegan, D. The incorporation of mobile learning into mainstream education and training / Desmond Keegan // 4th World Conference on m-Learning (m-Learn 2005), 25-28 October 2005, Cape Town, South Africa.

508. Knut Lundby, O.S. Networked PDA's in a Community of Learners / Knut Lundby, O.S., Larsen, A., Fjuk, A. // Proceedings of CSL 2002. – Boulder, Colarado, USA, January 7–11, 2002. – P. 548, 550.

509. Kukulska-Hulme, A. Mobile Usability in Educational Contexts : What have we learnt? / Kukulska-Hulme, A. // International Review of Research in Open and Distance Learning. – 2007. – Volume 8, Number 2.

510. Kurbel, K. Virtuality on the Students' and on the Teachers' sides : A Multimedia and Internet based International Master Program / Kurbel, Karl // Proceedings on the 7th International Conference on Technology Supported Learning and Training; Berlin, Germany / ICEF Berlin GmbH (Eds.) – Online Educa. – 2001. – November. – P. 133–136.

511. Kuutti, K. Activity Theory as a framework for Human-Computer Interaction Research / Kuutti, K. // Context and Consciousness : Activity Theory and Human-Computer Interaction. – Cambridge : MIT Press, 1996.

512. Laurillard, D. Pedagogical forms of mobile learning : Framing research questions / Laurillard, D. ; N. Pachler (Ed.) // Mobile learning: Towards a Research Agenda. – 2007. – Vol. 1. – P. 33-54.

513. Liao, C. J. A Grid Service Oriented Platform for Mobile Learning in English /

Liao, C. J., Chi, Y. L., Ou Yang, F. C. // Proceedings of E-Learn 2003. – Phoenix, Nov. 2003. – P. 2265–2268.

514. Low, L. M-learning standards report. Background, discussion and recommendations for usable and accessible m-learning / Leonard Low. – Version 1.0, 8 January 2007. – 41 p.
515. Martin, J.-P. Lernen durch Lehren : Paradigmenwechsel in der Didaktik? / Martin, J.-P., Oebel, G. // Deutschunterricht in Japan. – 2007. – Vol. 12. – P. 4–21 (Zeitschrift des Japanischen Lehrerverbandes)
516. McCalla, G. I. A peer help system for workplace training / McCalla, G. I., Greer, J. E. [et al.] // Artificial Intelligence in Education / Ed. B. du Boulay and R. Mizoguchi. – Amsterdam : IOS Press, 1997. – P. 183–190.
517. McConatha, D. Mobile learning in higher education : an empirical assessment of a new educational tool / McConatha, D., Praul, M., Lynch, M. J. // The Turkish Online Journal of Educational Technology. – 2008. – July, volume 7, Issue 3. – Article 2
518. Microlearning : Emerging Concepts, Practices and Technologies after e-Learning / Hug, T., Lindner, M., Bruck, P. A. (eds.). – Innsbruck University Press, 2006. – 230 p.
519. Mobile Learning : a Handbook for Educators and Trainers / Edited by : Agnes Kukulska-Hulme, John Traxler. – Routledge, 2005. – 192 p.
520. Mobile phones : Child's play? [Electronic resource] // BBC News. – 2000. – January, 5. – Mode of access : http://news.bbc.co.uk/hi/english/uk/newsid_591000/591791.stm
521. Nardi, B. Context and Consciousness : Activity Theory and Human-Computer Interaction / Nardi, B. – Cambridge : MIT Press, 1996.
522. Norman, D. A. Cognitive Engineering / Norman, D. A. // User Centred System Design / Ed. D. A. Norman and S. W. Draper. – Hillsdale, New Jersey : Lawrence Erlbaum Associates, 1986. – P. 31-61.
523. Novak, J. D. The use of concept mapping and knowledge vee mapping with junior high school teachers / Novak, J. D., Gowin, D. B. [et al.] // Science Edu-

cation. – 1983. – Vol. 67. – P. 625–645.

524. Nye, A. Volume 1 : Xlib Programming Manual / Nye, A. – 3rd Edition. – O'Reilly & Associates, 1992. – 821 p.
525. Oosterholt, R. Interaction design and human factors support in the development of a personal communicator for children / Oosterholt, R., Kusano, M. [et al.] // Proceedings of CHI 96. – ACM/ Addison Wesley, 1996. – P. 450-457.
526. Ou Yang, F.-Ch. The Construction of a Service-Oriented Learning Grid : Thesis to the Degree of Doctor of Philosophy / Ou Yang, F.-Ch. – Taipei, 2004. – 50 p.
527. Palm™ Education Pioneers Program : Final Evaluation Report. SRI International, September 2002.
528. Papert, S., Harel, I. Constructionism (Cognition and Computing) / Seymour Papert and Idit Harel. – Norwood : Ablex Publishing Corporation, 1991. – 518 p.
529. Pascoe, J. Developing personal technology for the field / Pascoe, J., Morse, D. [et al.] // Personal Technologies. – 1998. – Vol. 2(1). – P. 28-36.
530. Pask, G. Minds and media in education and entertainment : some theoretical comments illustrated by the design and operation of a system for exteriorizing and manipulating individual theses / Pask, G. // Progress in Cybernetics and Systems Research / Ed. R. Trappl and G. Pask. – Washington and London : Hemisphere Publishing Corporation, 1975. – IV. – P. 38-50.
531. Perlin, K. Pad : An Alternative Approach to the Computer Interface / Perlin, K., Fox, D. // Proceedings of SIGGRAPH '93. – New York, 1993. – P. 57-64.
532. Peters, O. Learning and Teaching in Distance Education / Peters, O. – London : Kogan Page, 1998.
533. Radkevitch, U. Ukraine : Will the Orange Revolution Boost the IT Outsourcing Industry [Electronic resource] / Radkevitch, Ulad, Starkell, Natasha // Outsourcing Journal. – 2005. – October. – Mode of access to the magazine : http://www.outsourcing-journal.com/oct2005-ukraine.html
534. Ravenscroft, A. Designing argumentation for conceptual development / Ravenscroft, A. // Computers and Education. – 2000. – Vol. 34. – P. 241-255.
535. Rickel, J. Intelligent tutoring in virtual reality : A preliminary report / Rickel, J.,

Johnson, L. // Artificial Intelligence in Education / Ed. B. du Boulay and R. Mizoguchi. – Amsterdam : IOS Press, 1997. – P. 294-301.

536. Rismark, M. Using mobile phones to prepare for university lectures : student's experiences / Rismark, M., Sølvberg, A. M., Strømme, A., Hokstad, L. M. // The Turkish Online Journal of Educational Technology. – 2007. – October, Volume 6, Issue 4, Article 9
537. Rosental, B. E. Archstone Consulting/Duke University Offshoring Study : Respondents Reported 30 Percent Annual Savings [Electronic resource] / Beth Ellyn Rosental. – March, 2005. – Mode of access : http://www.outsourcing-information-technology.com/university.html
538. Rossum, G. van. Python Reference Manual / Rossum, G. van. – Release 2.4.4, 18 October 2006.
539. Roth, W.-M. Learning environments research, lifeworld analysis, and solidarity in practice / Roth, W.-M. // Learning Environments Research. – 2000. – Vol. 2(3). – P. 225-247.
540. Rushby, N. Editorial / Rushby, N. // British Journal of Educational Technology. – 2005. – Vol. 36 (5). – P. 709–710.
541. Ryan, N. Enhanced reality fieldwork : the context-aware archaeologist assistant / Ryan, N., Pascoe, J. [et al.] // Archaeology in the Age of the Internet : Computer Applications and Quantitative Methods in Archaeology 1997. / Ed. L. Dingwall, S. Exon, V. Gaffney, S. Laflin and M. van Leusen. – Oxford, Archaeopress, 1997. – P. 269-274.
542. Scaife, M. External cognition : how do graphical representations work? / Scaife, M., Rogers Y. // International Journal of Human-Computer Studies. – 1996. – Vol. 45. – P. 185-213.
543. Schindler, E. Gartner Explains Why Windows Is Broken [Electronic resource] / Esther Schindler. – 2008. – April, 09 – Mode of access : http://www.cio.com/article/331963/Gartner_Explains_Why_Windows_Is_Broken
544. Seppala, P. Mobile Learning in Teacher Training / Seppala, P., Alamaki, H. //

Journal of Computer Assisted Learning. – 2003. – Vol. 19. – P. 330-335.

545. Sharples, M. A Socio-Cognitive Engineering Approach to the Development of a Knowledge-based Training System for Neuroradiology / Sharples, M., Jeffery, N. [et al.] // World Conference on Artificial Intelligence in Education (AI-ED ’97). – Kobe, Japan, 1997. – P. 402–409.
546. Sharples, M. A Theory of Learning for the Mobile Age / Sharples, M., Taylor, J., Vavoula, G. // The Sage Handbook of E-learning Research / R. Andrews & C. Haythornthwaite (eds.). – London : Sage, 2007. – P. 21–47.
547. Sharples, M. Developing a Writer's Assistant / Sharples, M., Goodlet, J. [et al.] // Technology and Writing : Readings in the Psychology of Written Communication / Ed. J. Hartley. – London : Jessica Kingsley, 1992. – P. 209–220.
548. Sharples, M. Evaluation of a Mobile Learning Organiser and Concept Mapping Tools / Mike Sharples, Tony Chan, Paul Rudman, Susan Bull // Proceedings of MLEARN 2003 : Learning with Mobile Devices. – London : Learning and Skills Development Agency, 2003. – P. 60–61.
549. Sharples, M. Learning as conversation : Transforming education in the mobile age / Sharples, M. // Proceedings of Conference on Seeing, Understanding, Learning in the Mobile Age, Budapest, Hungary. – Budapest, 2005. – P. 147–152.
550. Sharples, M. Socio-Cognitive Engineering : A Methodology for the Design of Human-Centred Technology / Sharples, M., Jeffery, N. [et al.] // European Journal of Operational Research. – 2002. – Vol. 132(2). – P. 310-323.
551. Sharples, M. Structured Computer-based Training and Decision Support in the Interpretation of Neuroradiological Images / Sharples, M., Jeffery, N. [et al.] // International Journal of Medical Informatics. – 2000. – Vol. 60(30). – P. 263-28.
552. Sharples, M. The design and implementation of a mobile learning resource / Sharples, M., Corlett, D., Westmancott, O. // Personal and Ubiquitous Computing. – 2002. – Vol. 6. – P. 220–234.
553. Sharples, M. The design of personal mobile technologies for lifelong learning /

Mike Sharples // Computers and Education. – 2000. – Vol. 34. – P. 177–193.

554. Shaw, M. We Can Teach Software Better / Mary Shaw // Computing Research News. – September 1992. – 4(4) – P. 2–4, 12.

555. Shiratuddin, N. E-Book Technology and Its Potential Applications in Distance Education / Shiratuddin, N., Landoni M., Gibb, F., Hassan, S. // Journal of Digital Information. – 2003. – Volume 3, Issue 4. – February. – E-education : Design and Evaluation.

556. Siemens, G. Connectivism : A Learning Theory for the Digital Age / George Siemens // International Journal of Instructional Technology and Distance Learning. – 2005. – Vol. 2. – No. 1, Jan. – P. 3–10.

557. Skliarenko, E. Transitional economies : trends in new higher educational systems in Easter Europe – the example of Ukraine [Electronic resource] / Skliarenko, E. – Proceedings of ICED2004. – 2004, June 21–23. – Mode of access: http://www.uottawa.ca/services/tlss/iced2004/pages/doc/ski.doc

558. Soloway, E. Handheld Devices are Ready at Hand / Soloway, E., Norris, C. [et al.] // Communications of the ACM. – 2001. – Vol. 44(6). – P. 15-20.

559. Spender, D. Embracing e-Learning in Australian Schools / Dale Spender, Fiona Stewart. – Melbourne : Commonwealth Bank, 2002. – I+117 p.

560. Thornton, P. Using mobile phones in English education in Japan / Thornton, P., Houser, C. // Journal of Computer Assisted Learning. – 2005. – Vol. 21. – P. 217–228.

561. Traxler, J. Defining, Discussing, and Evaluating Mobile Learning : The moving finger writes and having writ… / Traxler, J. // International Review of Research in Open and Distance Learning. – 2007. – June, Volume 8, Number 2.

562. Vavoula, G. A Lifecycle approach to evaluating MyArtSpace / Vavoula, G., Meek, J., Sharples, M., Lonsdale, P., Rudman, P. // Fourth IEEE International Workshop on Wireless, Mobile and Ubiquitous Technology in Education, 2006. WMUTE '06. – 16-17 Nov. 2006. – Athens, 2006. – P. 18–22.

563. Vavoula, G.N. Studying the Learning Practice : Implications for the Design of a Lifelong Learning Support System / Vavoula, G.N., Sharples M. // Proceedings

of IEEE International Conference on Advanced Learning Technologies (ICALT 2001) / Ed. T. Okamoto, R. Hartley, Kinshuk and J.P. Klus. – Madison, USA : IEEE Computer Society, 2001. – P. 379-380.

564. Vetter, R. Numina II SRS Student Response System Home Page [Electronic resource] / Ron Vetter. – [2000]. – Mode of access: http://aa.uncw.edu/numina/srs/
565. Viljoen, J.-M. The case for using SMS technologies to support distance education students in South Africa : conversations / Viljoen, Jeanne-Marie, Du Preez, Carl, Cook, Antoinette // Perspectives in Education : Recearch on ICTs and Education in South Africa : Special Issue. – 2005. – Vol. 23, issue 4. – P. 115–122.
566. Wagner, E. Disconnected / Wagner, E., Wilson, P.// ASTD. – 2005. – December. – P. 40-43.
567. Wagner, E. Enabling Mobile Learning / Wagner, E. // Educause Review. – 2005. – Vol. 40(3). – P. 40-53.
568. Weber, Ch. M. Rapid Learning in High Velocity Environment : Dissertation to the Degree of Doctor of Philosophy In Management of Technological Innovation and Entrepreneurship / Weber, Ch. M. – Massachusetts Institute of Technology, 2003. – 569 p.
569. Wellman, B. Networks in the Global Village / Wellman, B. – Boulder : . Westview Press, 1999. – 345 p.
570. Whitsed, N. Learning and Teaching / Whitsed, N. // Health Information & Libraries Journal. –2004. – Dec., Vol. 21. – P. 273–275.
571. Wilensky, U. Learning through participatory simulations : Network-based design for systems Learning in Classrooms Computer Supported Collaborative Learning / Wilensky, U., Stroup, W. // Conference on Computer-Supported Collaborative Learning (CSCL '99), Stanford University, California, December 12-15, 1999.