Hardware and software tools for teaching the basics of quantum informatics to students of specialized (high) schools

Liudmyla V. Lehka[1], Svitlana V. Shokaliuk[1] and Viacheslav V. Osadchyi[2]

[1]Kryvyi Rih State Pedagogical University, 54 Gagarin Ave., Kryvyi Rih, 50086, Ukraine
[2]Bogdan Khmelnitsky Melitopol State Pedagogical University, 20 Hetmanska Str., Melitopol, 72300, Ukraine

Abstract

The article defines the criteria for choosing a cloud-based platform for mastering the basics of quantum informatics by students of a specialized (high) schools: cross-browser; intuitive interface; the possibility of free access; access without registration and simplified registration; the presence of a systematized reference system with examples; support for the development of the environment by the developer; support for working in a personal educational environment; support for working with quantum algorithms in graphical mode; automatic conversion of quantum algorithms from graphic format to program code text; support for the Ukrainian-language localization; availability of a mobile application; responsive design. The possibilities of platforms for implementing quantum algorithms from the following companies are analyzed: Microsoft, QuTech, Amazon Braket, IBM. The choice of the IBM Quantum cloud-based platform is justified. Work at IBM Quantum Composer and IBM Quantum Lab is described. Information about quantum operations and gates is presented: their designation in IBM Quantum Composer and IBM Quantum Lab, the gate matrix, and the purpose of the gate. An example of implementing quantum teleportation in the form of a circuit and program is given.

Keywords

quantum calculations, quantum computer, quantum circuit, quantum algorithm, IBM Quantum Lab, IBM Quantum Composer, Python, Jupyter Notebook

1. Introduction

The study of the subject content lines of the school computer science course, namely, "Information Technologies for creating and processing information objects", "Modeling, algorithmization and programming", "Telecommunication technologies", is carried out with the support of appropriate hardware and software (in particular, Internet services).

With the experimental introduction of quantum informatics issues into the school course [1, 2] we can say about the problem of appropriate and pedagogically balanced expert selection of hardware and software tools to support studying, taking into account many criteria.

CTE 2021: 9th Workshop on Cloud Technologies in Education, December 17, 2021, Kryvyi Rih, Ukraine
asp-18-lehka@kdpu.edu.ua (L. V. Lehka); shokalyuk@kdpu.edu.ua (S. V. Shokaliuk); osadchyi@mdpu.org.ua (V. V. Osadchyi)
https://kdpu.edu.ua/personal/svshokaliuk.html (S. V. Shokaliuk); http://osadchyi.mdpu.org.ua (V. V. Osadchyi)
0000-0001-5768-5475 (L. V. Lehka); 0000-0003-3774-1729 (S. V. Shokaliuk); 0000-0001-5659-4774 (V. V. Osadchyi)

2. Criteria for selecting hardware and software tools to support the study of informatics and telecommunication technologies in secondary education schools

The teacher's use of various types of hardware and software in the informatics curriculum [3] is not limited, provided that it complies with the requirements of current legislation [4, 5, 6, 7]. Also, the program does not specify universal criteria for their selection. Nevertheless, there are the author's systems of criteria for selecting hardware and software tools to support the study of informatics.

Yatsenko and Yatsenko [8, p. 107] propose the following criteria for selecting software:

- criteria related to the capabilities of the programming language:
  - support for writing mathematical expressions in mathematical form;
  - calculation model used (data flow / control flow);
  - support for algorithmic constructs;
- criteria related to the possibility of using the environment at the initial stage of learning programming languages:
  - simplicity, modernity and visual appeal of the interface;
  - availability of methodological manuals;
  - Ukrainian-language interface;
  - cost (free / paid);
- criteria related to technological aspects of the environment:
  - cross-platform;
  - supporting popular robotic constructors;
  - license (proprietary or open);
  - supporting and developing the environment.

Shevchuk [9, p. 31] considers the programming environment as a learning tool to be important characteristics: prevalence, availability, interface features, implementation method, system requirements, methodological support, and user-friendly and intuitive interface.

Bazurin [10, p. 15] notes that the choice of a programming environment for use in the process of learning a programming language is influenced by the following conditions:

- computer specifications and system requirements of the programming environment;
- availability of operating systems and additional software required for the operation of the software environment;
- software environment functionality;
- software environment interface;
- availability of documentation for the software environment;
- availability of educational and methodological support;
- level of competence of an informatics teacher.

Vakaliuk [11, p. 156] outlines the following characteristics that should meet the cloud-based environment: accessibility and mobility; openness; integrity and continuity; efficiency; regularity; consistency and structure; innovation; integration with cloud-based resources; clarity; functionality; collectivity; ensuring project activities; scientificity; reliability; communication; flexibility and adaptability; individualization; fullness; convenience; expediency.

Vorozhbyt [12, p. 29] identifies the following criteria for the use of web-based technologies to create learning content:

- the cost of developing;
- flexibility of use;
- feedback from students;
- clarity of presentation of educational material;
- pedagogical control of knowledge, motivation to study;
- the ability to use multimedia dynamic content;
- educational activities of students;
- cooperation of teachers and students, students with each other.

When choosing a cloud-based platform for mastering the basics of quantum informatics by students of a specialized (high) schools, we took into account the following criteria:

- cross-browser capability;
- intuitive interface;
- possibility of free (unpaid) access;
- simplified registration;
- availability of a systematic help system with examples;
- support for the development of the environment by the developer;
- support for working in a personal educational environment;
- support for working with quantum algorithms in graphical mode;
- automatic conversion of quantum algorithms from graphic format to program code text;
- support for Ukrainian localization;
- availability of the mobile application;
- responsive design.

3. Cloud-based platforms for implementing quantum algorithms

The choice of a cloud-based platform to support the study of the basics of quantum informatics by students of specialized (high) schools was preceded by an analysis of possible platforms for implementing quantum algorithms from Microsoft, QuTech, Amazon Braket, IBM, and other (Alibaba, Google, Intel, D-Wave Systems, Quantum Circuits, IonQ, Honeywell, Xanadu, and Rigetti).

Microsoft via the Azure Quantum cloud-based platform for quantum computing (figure 1) allows visitors to learn how to use the Quantum Development Kit to create applications for

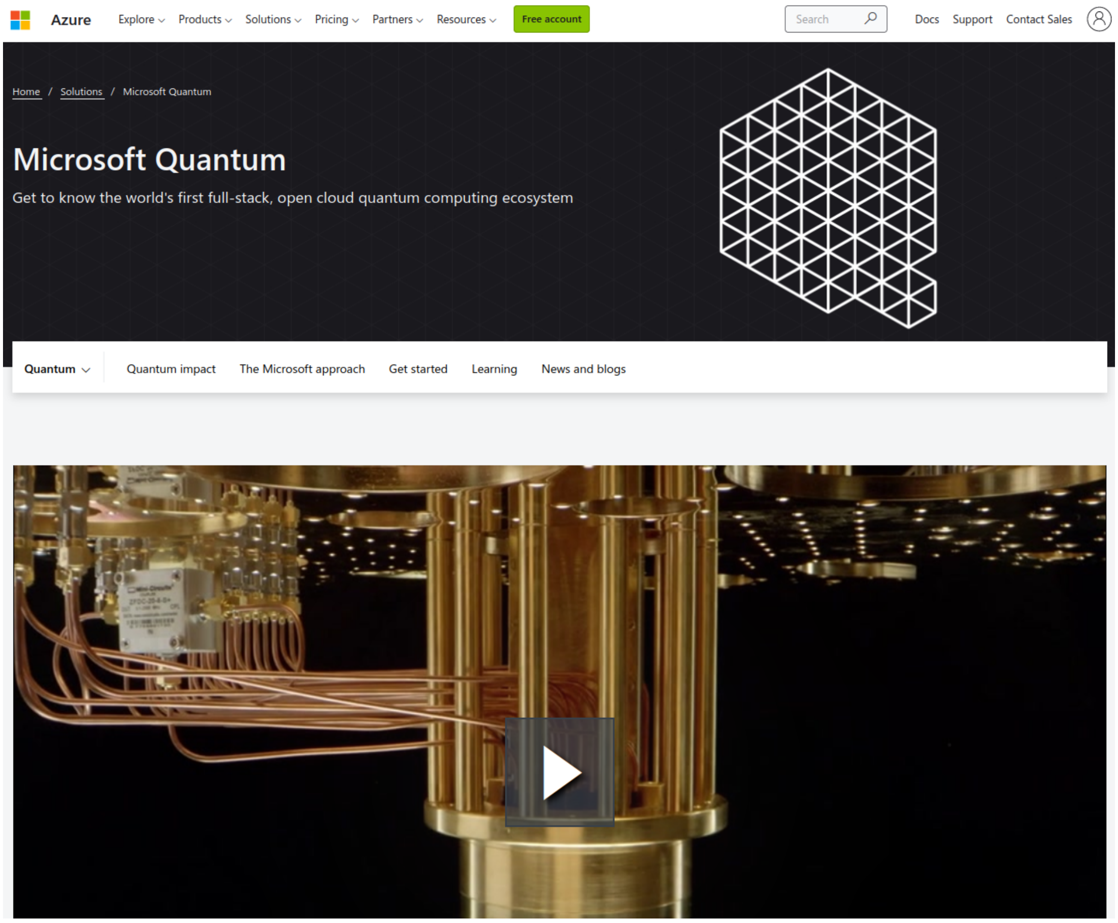

Figure 1: Home page of a cloud-based platform for Azure Quantum computing [13].

quantum equipment in the Q# language. Microsoft does not have its own quantum computer but provides access to Honeywell Quantum Solutions, IonQ, and 1QBit quantum equipment.

The Dutch company QuTech, through a cloud-based platform for Quantum Computing Quantum Inspire, provides free access without registration to educational materials and a quantum simulator, and for registered users – to quantum chips (figure 2).

Online quantum computing service Amazon Braket (figure 3) provides access to quantum equipment for companies D-Wave, IonQ and Rigetti.

IBM was the first to provide cloud access to its own quantum equipment (2016), and now, in our opinion, Quantum Composer and Quantum Lab from IBM provide the greatest opportunities for free use of quantum computers [16].

IIBM quantum simulators run on computers of classical architecture and allow you to simulate the execution of quantum algorithms and calculations. Quantum simulators work faster, so it is recommended that you first test your quantum algorithm on the simulator [18]. At the time of accessing the IBM quantum simulator resource, available for research were simulators from 32 to 5000 qubits (see Table 1).

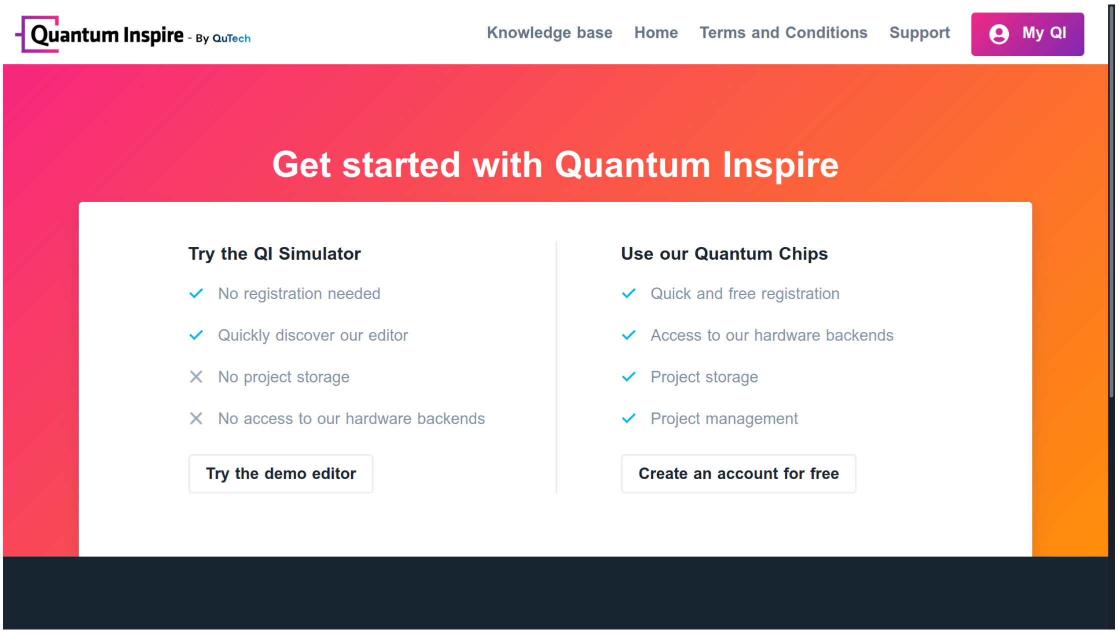

Figure 2: Fragment of the page of a cloud-based platform for quantum computing Quantum Inspire [14].

IBM provides open (free) access to real quantum computers from 1 to 32 qubits (see Table 2). IBM quantum computers with more qubits are available to users on additional terms. The larger the "quantum volume", the larger the circuit can be implemented on its qubits.

4. IBM Quantum as a leading cloud-based platform for quantum computing

IBM Quantum provides the ability to create quantum circuits in the IBM Quantum Composer and to write quantum computer programs in QASM and Python in the IBM Quantum Lab.

4.1. IBM Quantum Composer

IBM Quantum Composer is the simplest set of IBM Quantum tools for creating and graphically visualizing quantum algorithms and then running them on quantum simulators or real IBM quantum computers. In figure 5 shows:

1 – sidebar provides access to your own files, tasks, or documentation. You can open or close the sidebar by clicking the icon on the tab;

2 – menu bar is used to create a new circuit, manage and save circuits, customize the workspace, get help, and more;

3 – account login area and parameter settings for running the quantum circuit;

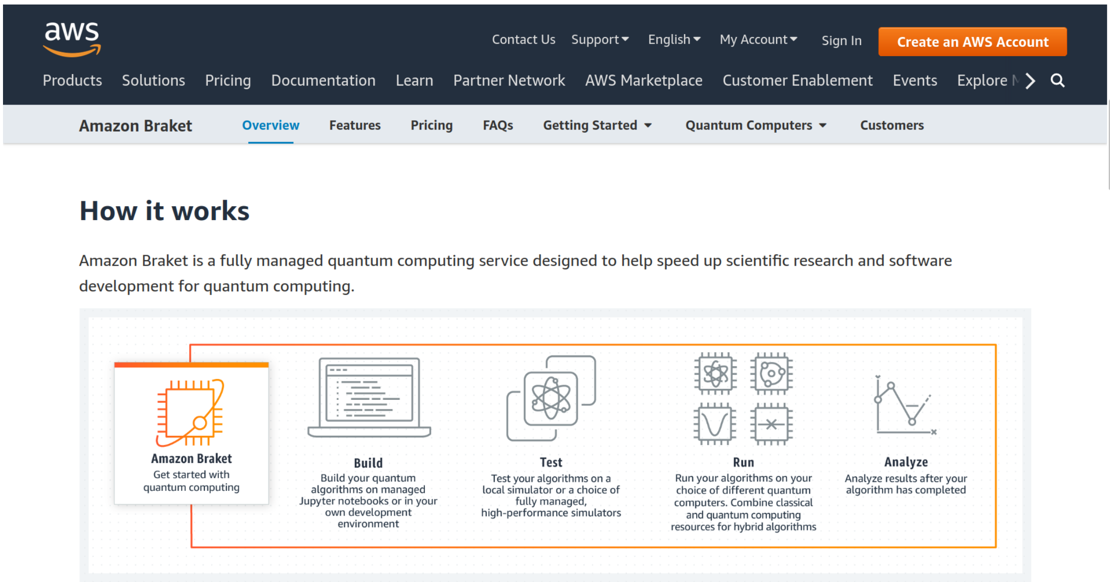

Figure 3: Fragment of the page of a cloud-based platform for quantum computing Amazon Braket [15].

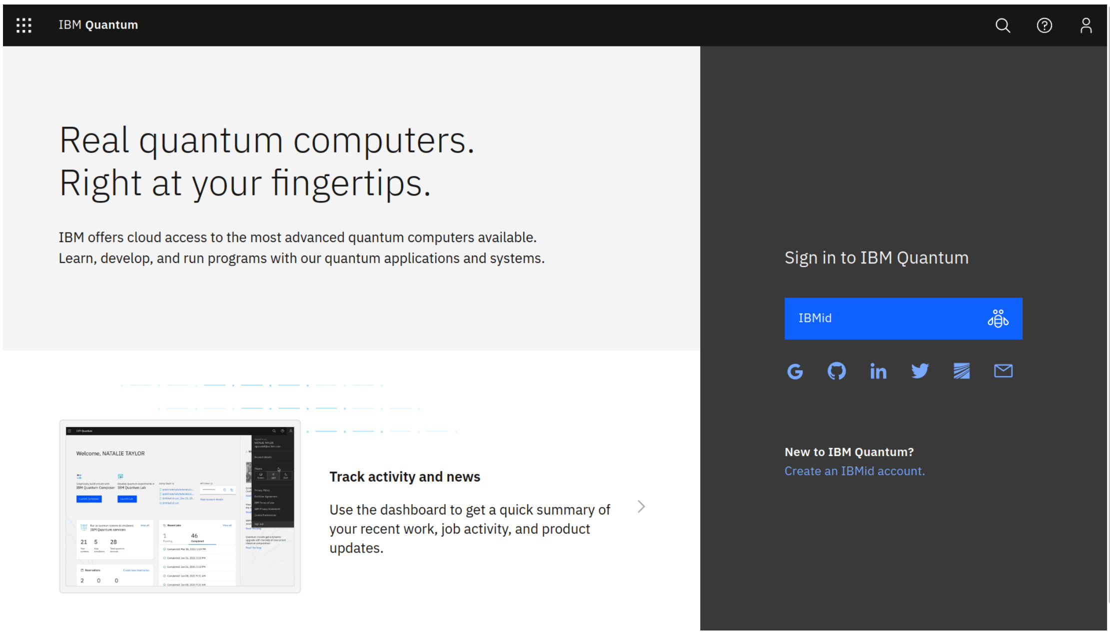

Figure 4: The main page of cloud-based platform for quantum calculation IBM Quantum [17].

4 – quantum gates and the operation panel are the building blocks of quantum circuits. Different types of gates are grouped by color: classic gates are dark blue, phase gates are light blue,

Table 1
Key features of IBM quantum simulators.

Name of the IBM quantum simulator	Qubits	Basis gates
simulator_stabilizer	5000	ID, X, Y, Z, H, S, SDG, SX, SWAP, CX, CY, CZ, DELAY
simulator_mps	100	U1, U2, U3, U, P, CP, CX, CZ, ID, X, Y, Z, H, S, SDG, SX, T, TDG, SWAP, CCX, UNITARY, ROERROR, DELAY
simulator_extended_stabilizer	63	ID, X, Y, Z, H, S, SDG, SX, SWAP, CX, CZ,DELAY, P, CCX, U1, CCZ, T, TDG
simulator_statevector	32	U1, U2, U3, U, P, R, RX, RY, RZ, ID, X, Y, Z, H, S, SDG, SX, T, TDG, SWAP, CX, CY, CZ, CSX, CP, CU1, CU2, CU3, RXX, RYY, RZZ, RZX, CCX, CSWAP, MCX, MCY, MCZ, MCSX, MCP, MCU1, MCU2, MCU3, MCRX, MCRY, MCRZ, MCR, MCSWAP, UNITARY, DIAGOAL, MULTIPLEXER, INITIALIZE, KRAUS, ROERROR, DELAY
ibmq_qasm_simulator	32	U1, U2, U3, U, P, R, RX, RY, RZ, ID, X, Y, Z, H, S, SDG, SX, T, TDG, SWAP, CX, CY, CZ, CSX, CP, CU1, CU2, CU3, RXX, RYY, RZZ, RZX, CCX, CSWAP, MCX, MCY, MCZ, MCSX, MCP, MCU1, MCU2, MCU3, MCRX, MCRY, MCRZ, MCR, MCSWAP, UNITARY, DIAGONAL, MULTIPLEXER, INITIALIZE, KRAUS, ROERROR, DELAY

and non-unitary operations are gray. The button with three dots allows you to open the directory of quantum operations and gates, get help with the use of hotkeys, minimize the panel of quantum operations to one row, save the created quantum circuit as a file in different formats (pdf, svg, png);

5 – graphical circuit editor. Adding operations to be performed on cubes is performed by simply dragging the gate to the area of the graphical quantum circuit editor;

6 – code editor allows you to view and copy automatically generated OpenQASM or Qiskit code for use in other applications;

7 – phase disks represent the phase vector of the qubit state on the complex plane defined by a radial line that rotates counter clockwise;

8 – visualizations of the state of qubits modelling the created circuit in the process of construction.

Elements are presented in Table 3:

- gate H, or Hadamard gate, required to transfer the qubit to the state of superposition;
- Pauli X gate is equivalent to a bitwise negation;
- CNOT gate, also known as the controlled negation gate (CX), acts on a pair of qubits, one of which acts as a control and the other as a target. It executes an negation on the target

Table 2
The main characteristics of IBM quantum computers.

IBM quantum computer	Qubits	Quantum volume	Basis gates	Free access
ibmq_16_melbourne	15	8	CX, ID, RZ, SX, X	Yes
ibmq_5_yorktown	5	8	CX, ID, RZ, SX, X	Yes
ibmq_armonk	1	1	ID, RZ, SX, X	Yes
ibmq_athens	5	32	CX, ID, RZ, SX, X	Yes
ibmq_belem	5	16	CX, ID, RZ, SX, X	Yes
ibmq_bogota	5	32	CX, ID, RZ, SX, X	Yes
ibmq_brooklyn	65	32	CX, ID, RZ, SX, X	No
ibmq_cairo	27	64	CX, ID, RZ, SX, X	No
ibmq_casablanca	7	32	CX, ID, RZ, SX, X	No
ibmq_dublin	27	64	CX, ID, RZ, SX, X	No
ibmq_guadalupe	16	32	CX, ID, RZ, SX, X	No
ibmq_hanoi	27	64	CX, ID, RZ, SX, X	No
ibmq_jakarta	7	16	CX, ID, RZ, SX, X	No
ibmq_kolkata	27	128	CX, ID, RZ, SX, X	No
ibmq_lagos	7	32	CX, ID, RZ, SX, X	No
ibmq_lima	5	8	CX, ID, RZ, SX, X	Yes
ibmq_manhattan	65	32	CX, ID, RZ, SX, X	No
ibmq_manila	5	32	CX, ID, RZ, SX, X	Yes
ibmq_montreal	27	128	CX, ID, RZ, SX, X	No
ibmq_mumbai	27	128	CX, ID, RZ, SX, X	No
ibmq_nairobi	7	32	CX, ID, RZ, SX, X	No
ibmq_paris	27	32	CX, ID, RZ, SX, X	No
ibmq_peekskill	27	–	CX, ID, RZ, SX, X	No
ibmq_quito	5	16	CX, ID, RZ, SX, X	Yes
ibmq_rome	5	32	CX, ID, RZ, SX, X	No
ibmq_santiago	5	32	CX, ID, RZ, SX, X	Yes
ibmq_sydney	27	32	CX, ID, RZ, SX, X	No
ibmq_toronto	27	32	CX, ID, RZ, SX, X	No

qubit each time the control is in a state $|1\rangle$. If the control qubit is in superposition, this gate creates entanglement;
- gate Z changes the sign of the qubit;
- qubit measurement is an irreversible operation that changes the state of the qubit. The measurement result is a traditional bit;
- barrier is useful for visualizing quantum circuits.

IBM Quantum Composer allows you to visualize the circuit of quantum operations and the result in the form of probabilities, the output statevector, graphically on Q-sphere, as well as view the description of the created circuit on the QASM 2.0 or Qiskit quantum assembler with the ability to open the code of the IBM Quantum Lab. Viewing the created circuit in inspector mode allows simultaneous step-by-step viewing of the state of changes in qubits both on the graphical diagram and in the QASM quantum assembler window. IBM Quantum Composer allows you to choose a cloud-based quantum device on which the constructed quantum circuit

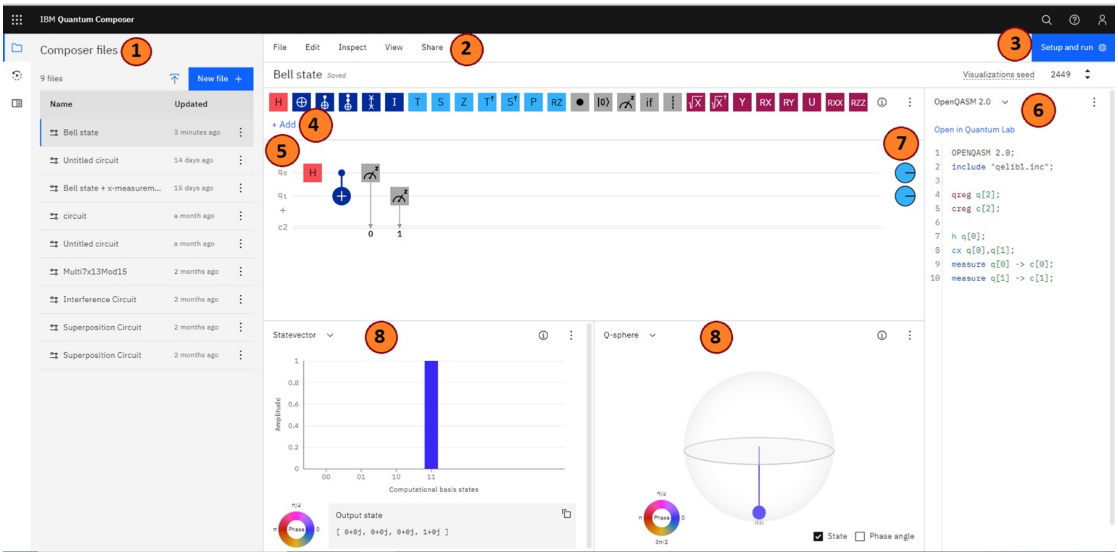

Figure 5: IBM Quantum Composer.

Table 3
Some notations used in IBM Quantum Composer.

Circuit element	Notation in Quantum Composer	Example of use in IBM Quantum Lab	Matrix representation
H gate	H	circuit.h(qreg)	$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$
X gate	⊕	circuit.x(qreg)	$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
CNOT gate		circuit.cx(qreg)	$CX = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$
Z gate	Z	circuit.z(qreg)	$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$
Measurement		circuit.measure(qreg, creg)	
Barrier operation		circuit.barrier	

will be calculated. The quantum circuit is read from left to right.

Working with IBM Quantum Composer with access without registration requires using only the simulator.

To take advantage of the great features of IBM Quantum Composer, to be able to choose a simulator, run the circuit on a real quantum hardware, you need to log in with one of your accounts (Google, GitHub, Twitter, LinkedIn, Fraunhofer or email) or get an IBM account to access trial versions, demos, starter kits, services and APIs. Organizations within the IBM Quantum Network can access the latest quantum computing systems and development tools after submitting and reviewing an electronic application [19].

4.2. IBM Quantum Lab

IBM Quantum Lab is available in any standard browser for viewing quantum circuits, textual explanations to them, and visualizing them, provided that they are authorized (for example, via a Google account). In IBM Quantum Lab, you can create a new program for quantum hardware or open a circuit that was previously built in IBM Quantum Composer through the code editor. The IBM Quantum Lab interface will be familiar to those users who have experience with Python Notebook or Jupyter Notebook (figure 6).

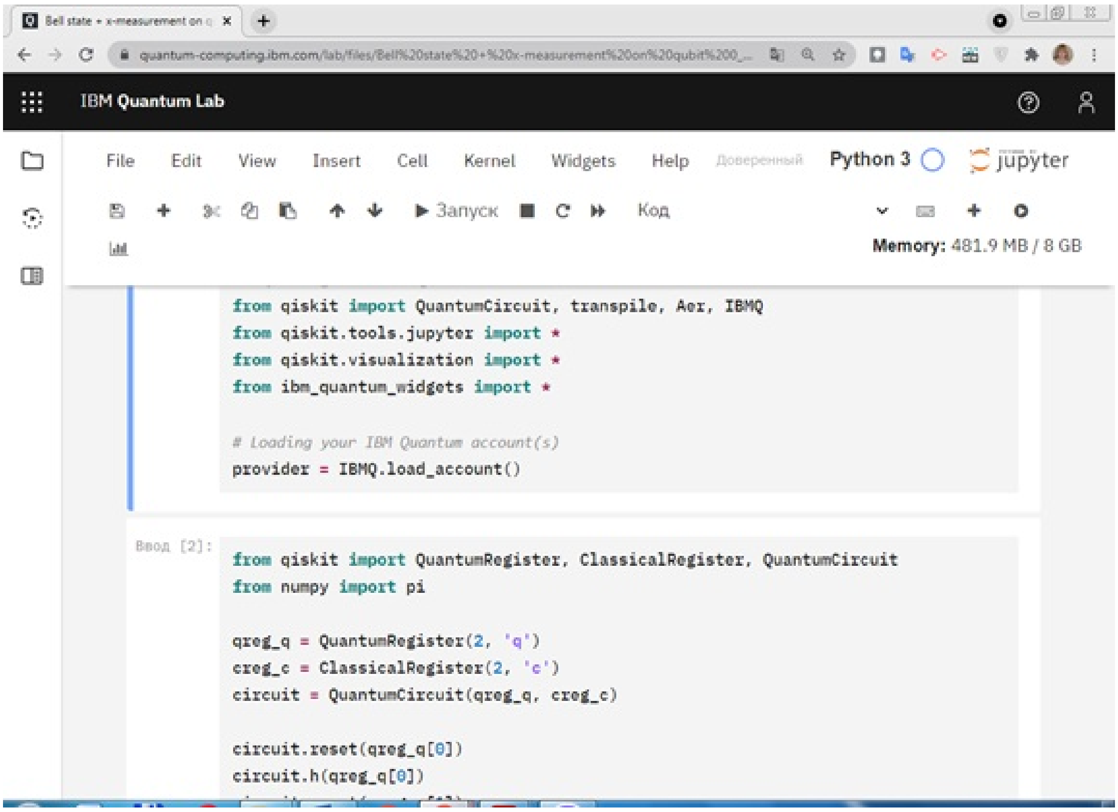

Figure 6: IBM Quantum Lab interface.

A command line is launched using the usual Jupyter Notebook keys: the button "Run" or the key combination Shift + Enter. While constructing a program for quantum equipment, you have to specify the required number of qubits and classical bits (by default, each qubit is set to zero initial state). Then you should add gates (operations) to manipulate them and output the result, or add any way to visualize the constructed quantum circuit.

We will demonstrate the possibilities of implementing quantum algorithms on the platform using the example of the quantum teleportation algorithm. Quantum teleportation is the transfer of quantum states from one qubit to another. Quantum teleportation is not the transport or any physical movement of a qubit from one location to another. In quantum mechanics, the clone (copy) negation theorem applies [20, p. 89]. When copying while working on quantum equipment, an implicit measurement occurs that destroys the current quantum state. To solve this problem, we use quantum entanglement. For qubits to be entangled, they have to interact with each other. Measuring the state of one entangled qubit results in an instantaneous transition to the corresponding state of another entangled qubit.

Here is a verbal description of the quantum teleportation algorithm, the graphical description of which is carried out using the IBM Quantum Composer service and is shown in figure 7:

1) using the Not operation, we will convert the zero qubit to state 1, and leave the first and second qubits in the primary zero states. (Note. This action should be in the example so that we don't pass a NULL value, in fact, the null qubit will contain the value that needs to be teleported);
2) let's convert the first qubit to a superposition by H gate;
3) let's entanglement the first and second qubits with a CNOT gate (where the first is the control one, and the second is the target one. If the control (first) qubit is in state 1, then the target (second) is inverted by the CNOT gate);
4) similarly we entanglement the zero and first qubits;
5) convert a zero qubit to a superposition (using H gate);
6) we measure the states of the zero and first qubits (Measurement operation). The measurement results are stored in two classical bits transmitted by a classical connection;
7) on the side where the state of the zero qubit is passed, there is a second qubit, to which we apply the gates X and Z (in the sequence X or Z, it does not matter what will be the first), as a result, we get the value of the zero qubit in the second qubit;
8) we measure the value of the second qubit.

For step-by-step tracking of the execution of the compiled circuit, you can use the Inspect command (figure 8).

The teleportation result can be seen on the phase disks and the lower part of the Quantum Composer window in state vector, probabilities and Q-sphere.

Let's analyze the appearance of Phase disks after constructing and running a quantum circuit (figure 7):

- for a zero qubit, the phase disk is unpainted, so it is in the state $|0\rangle$;
- for the first and second qubits, the phase disk is completely painted over, so it is in the $|1\rangle$ state.

Consider visualizing the result in Statevector and Q-sphere modes, reading from right to left – 011. Three positions of the resulting binary number indicate that the last qubit has a value of 1. Q-sphere, which can be rotated, relates the calculated value of each qubit of the quantum circuit to a point on the surface of the sphere.

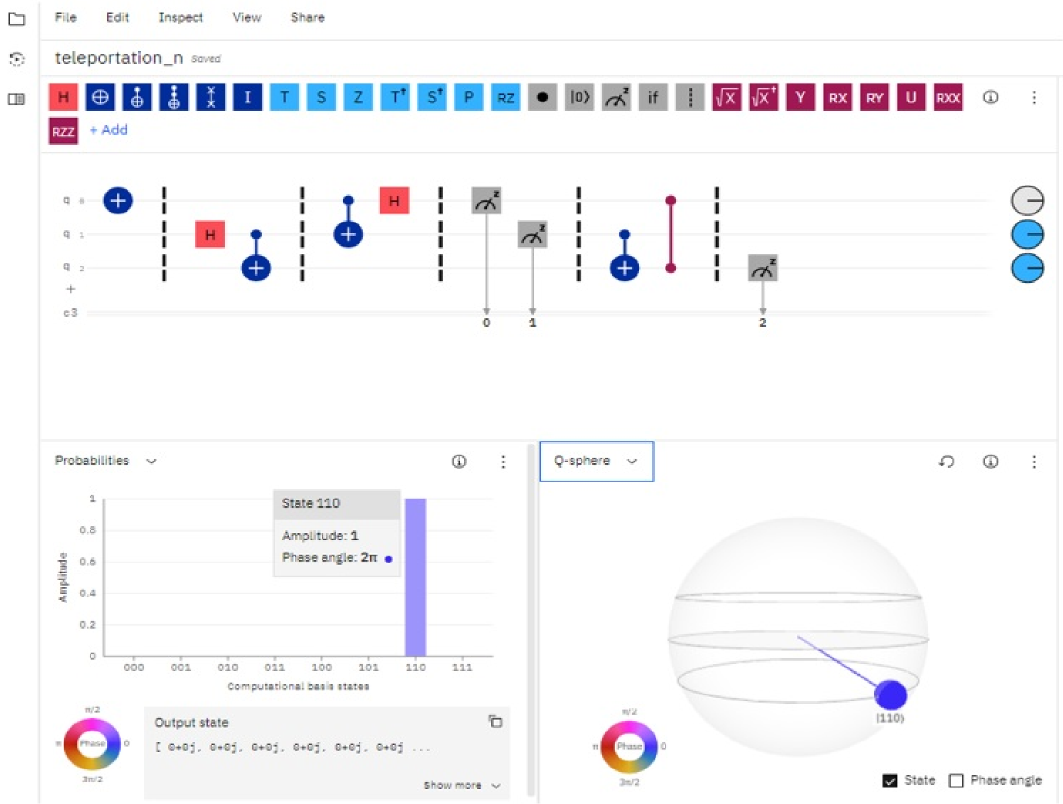

Figure 7: The circuit of the quantum teleportation algorithm for IBM Quantum Composer.

To visualize completed qubit operations in Qiskit Jupyter Notebook, you can use code `%matplotlib inline` and use draw – `QantumCircuite.draw()` – and as a circuit `QantumCircuite.draw(output='mpl')`.

To complete the circuit, configure the simulator. `Qasm_simulator` is an element of Aer in Qiskit – `simulator = Aer.get_backend('qasm_simulator')`.

The results of the circuit performed on the simulator are stored in the corresponding variable. Then they can be displayed as a histogram:

```
from qiskit.visualization import plot_histogram
plot_histogram(result.get_counts(circuit))
```

Executing the circuit on a quantum computer requires using IBM account. To do this run the following commands:

```
IBMQ.load_account();
provider = IBM.get_provider(hub=$'ibm-q'$)
```

Next, you need to choose a quantum computer from those that are available at this time, for example, qcomp `= provider.get_backend('ibmq_16_melbourn')`.

Using IBM Quantum Composer we will open a description of the created circuit in Python:

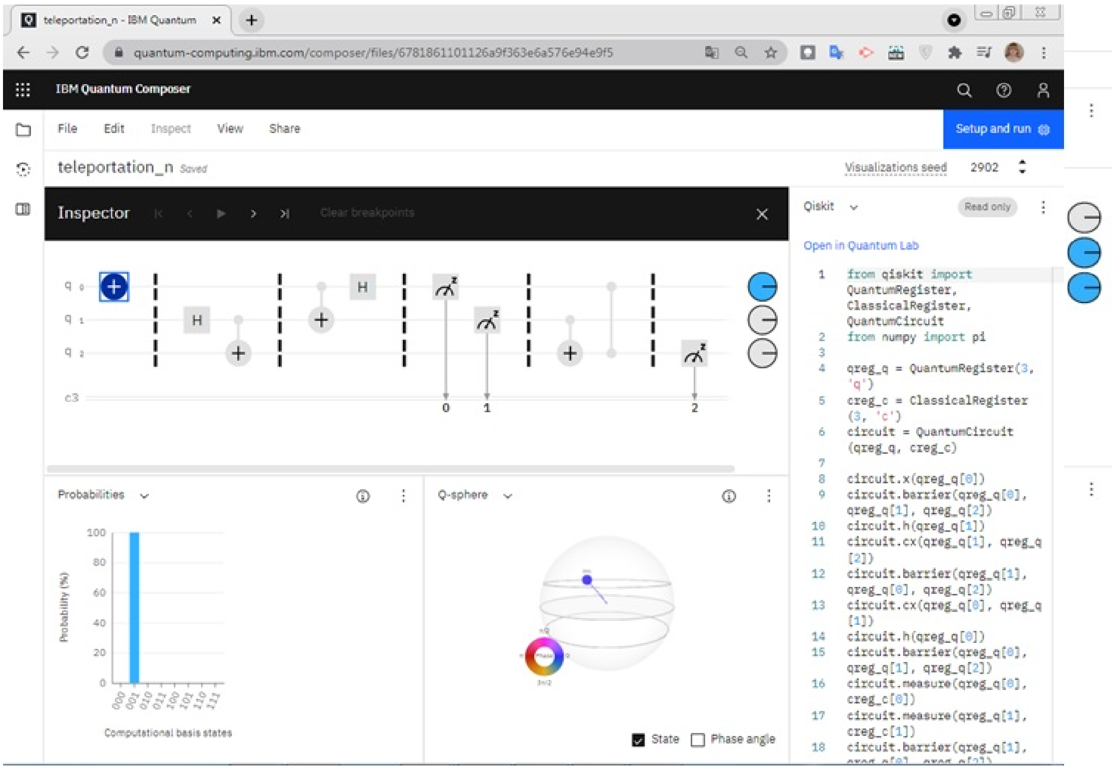

Figure 8: Illustration of step-by-step execution of the diagram via Inspect.

```
import numpy as np
# Importing standard Qiskit libraries
from qiskit import QuantumCircuit, transpile, Aer, IBMQ
from qiskit.tools.jupyter import *
from qiskit.visualization import *
from ibm_quantum_widgets import *

# Loading your IBM Quantum account(s)
provider = IBMQ.load_account()
from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
from numpy import pi

qreg_q = QuantumRegister(3, 'q')
creg_c = ClassicalRegister(3, 'c')
circuit = QuantumCircuit(qreg_q, creg_c)

circuit.x(qreg_q[0])
circuit.barrier(qreg_q[0], qreg_q[1], qreg_q[2])
circuit.h(qreg_q[1])
```

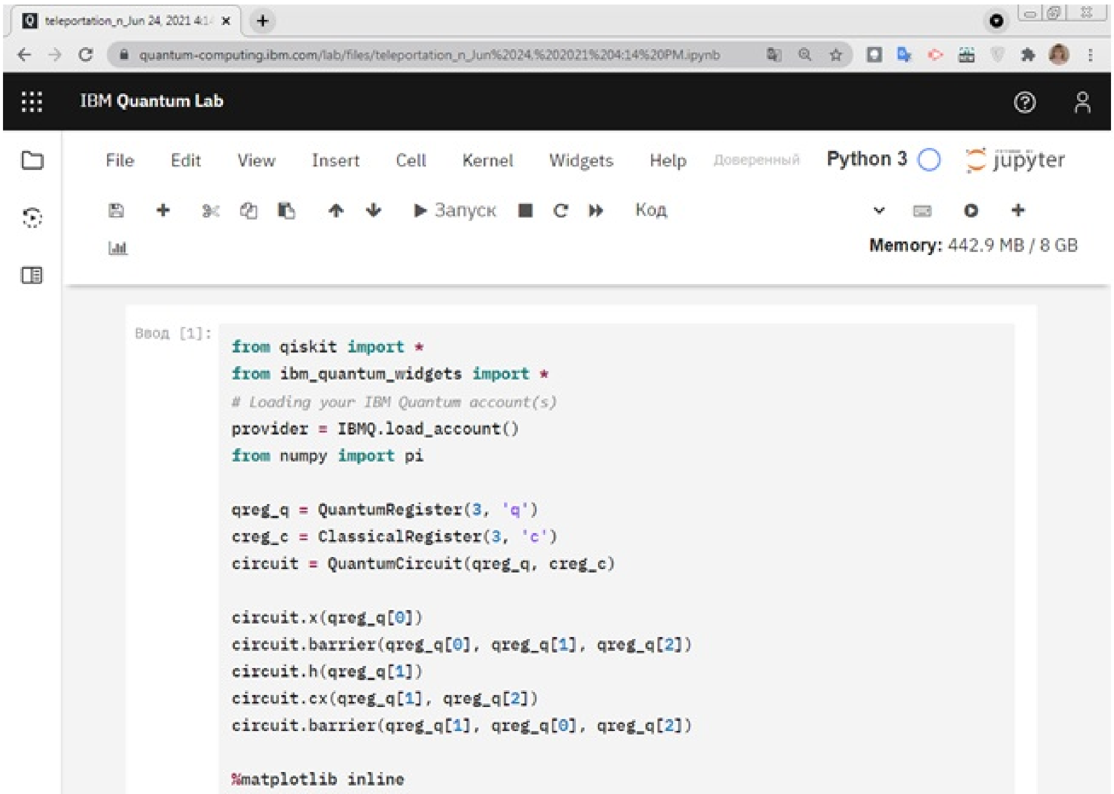

Figure 9: Illustration of performing completed operations with qubits in Qiskit Jupyter Notebook.

```
circuit.cx(qreg_q[1], qreg_q[2])
circuit.barrier(qreg_q[1], qreg_q[0], qreg_q[2])
circuit.cx(qreg_q[0], qreg_q[1])
circuit.h(qreg_q[0])
circuit.barrier(qreg_q[0], qreg_q[1], qreg_q[2])
circuit.measure(qreg_q[0], creg_c[0])
circuit.measure(qreg_q[1], creg_c[1])
circuit.barrier(qreg_q[1], qreg_q[0], qreg_q[2])
circuit.cx(qreg_q[1], qreg_q[2])
circuit.cz(qreg_q[0], qreg_q[2])
circuit.barrier(qreg_q[1], qreg_q[0], qreg_q[2])
circuit.measure(qreg_q[2], creg_c[2])

editor = CircuitComposer(circuit=circuit)
editor
```

Changes can be made to the above code, and a diagram can be added at intermediate stages, for example, after each gate barrier:

```
editor = CircuitComposer(circuit=circuit)
editor
```

Taking into account the formulated and illustrated capabilities of the IBM Quantum cloud-based quantum computing platform, in particular, the IBM Quantum Composer and IBM Quantum Lab services, as well as the results of comparing the capabilities of analog platforms according to the constructed system of criteria (Table 4), it was found that this platform is the most acceptable in support of mastering the basics of quantum informatics by students of specialized (high) schools. First of all, because this platform has free access, is constantly being improved by the developer, and provides the ability to implement quantum algorithms graphically with synchronous conversion to program code (in QASM mode – quantum assembler or Qiskit with support for Python 3.6 and higher).

Table 4
Compliance with the criteria for the main characteristics of cloud-based platforms for the implementation of quantum computing.

Criterion (characteristic)	Microsoft	QuTech	Amazon Braket	IBM
cross-browser capability	+	+	+	+
intuitive interface	+	+	+	+
access without registration and simplified registration	identification by phone number or bank card	+	identification by phone number, address	+
possibility of free access	+	+	+	+
availability of a systematic help system with examples	-	+	-	+
support for the development of the environmen t by the developer	+	+	+	+
support for working in a personal educational environment	?	+	?	+
support for working with quantum algorithms in graphical mode	?	-	?	+
automatic conversion of quantum algorithms from graphic format to program code text	?	-	?	+
support for Ukrainian-language localization	-	-	-	-
availability of the mobile app	-	-	-	-
responsive design	+	+	+	-

Currently, the lack of Ukrainian localization of the IBM Quantum interface is not a significant problem for students in grades 10-11. With such an organization of training, it is natural to develop their key multilingual competence. Due to the rapid development of the IBM Quantum platform, we can hope for the emergence of responsive design and a mobile application.

5. Conclusions

So, based on the outlined criteria for selecting a cloud-based platform for mastering the basics of quantum informatics in general secondary education institutions, analysing the possibility of platforms for implementing quantum algorithms of leading companies, a cloud-based platform IBM Quantum that meets certain criteria in the best way for studying the basics of quantum informatics have been determined.

At the moment, the experimental implementation of educational and methodological materials on the basics of quantum programming in the educational process of a specialized (high) schools has been completed, and the results obtained are being processed.

References

[1] L. V. Lehka, S. V. Shokaliuk, E. Y. Bogunenko, Propaedeutics of studying quantum computer science in a specialized (high) school, Physical and Mathematical Education 28 (2021) 51–56. doi:10.31110/2413-1571-2021-028-2-009.

[2] L. V. Lehka, A. O. Bielinskyi, S. V. Shokaliuk, V. N. Soloviev, P. V. Merzlykin, Y. Y. Bohunenko, Prospects of quantum informatics and the study of its basics in the school course, in: S. Semerikov, V. Osadchyi, O. Kuzminska (Eds.), Proceedings of the Symposium on Advances in Educational Technology, AET 2020, University of Educational Management, SciTePress, Kyiv, 2022.

[3] Curricula for grades 10-11, 2021. URL: https://mon.gov.ua/ua/osvita/zagalna-serednya-osvita/navchalni-programi/navchalni-programi-dlya-10-11-klasiv.

[4] President of Ukraine, On the decision of the national security and Defense Council of Ukraine of April 28, 2017 "On the application of personal special economic and other restrictive measures (sanctions)", 2017. URL: https://zakon.rada.gov.ua/go/133/2017v.

[5] President of Ukraine, On the decision of the National Security and Defense Council of Ukraine of May 2, 2018 "On the application and cancellation of personal special economic and other restrictive measures (sanctions)", 2018. URL: https://zakon.rada.gov.ua/go/126/2018.

[6] Law of Ukraine "On copyright and related rights", 1993. URL: https://zakon.rada.gov.ua/go/3792-12.

[7] Cabinet of Ministers of Ukraine, On approval of regulations on the distribution of copies of audiovisual works, volumes, videograms, computer programs, databases, 2000. URL: https://zakon.rada.gov.ua/go/1555-2000-%D0%BF.

[8] O. I. Yatsenko, O. S. Yatsenko, Criteria and indicators for selecting the playing environment for the development of information and communication competence of future primary

school teachers, Actual issues of modern computer science 12 (2019) 105–108. URL: http://eprints.zu.edu.ua/31178/1/Zbirnuk_Zhytomyr_IKT_2019-2.pdf.
[9] P. Shevchuk, The problem of choosing a programming language and environment as a learning tool, 2010, pp. 30–31. URL: http://eprints.zu.edu.ua/id/eprint/13592.
[10] V. M. Bazurin, Programming environments as a means of teaching pupils to programming basics, Information Technologies and Learning Tools 59 (2017) 13–27. URL: https://journal.iitta.gov.ua/index.php/itlt/article/view/1601. doi:10.33407/itlt.v59i3.1601.
[11] T. A. Vakaliuk, Features cloud oriented learning environment for training bachelors information, Naukovyi chasopys NPU imeni M.P. Drahomanova. Seriia 2. Kompiuterno-oriientovani systemy navchannia (2019) 154–157. URL: https://sj.npu.edu.ua/index.php/kosn/article/view/28.
[12] A. V. Vorozhbyt, Using web-based technologies in learning Informatics in the establishments of general secondary education, Thesis for a candidate degree in pedagogical sciences, specialty 13.00.02 – theory and methods of teaching (computer science), National Pedagogical Dragomanov University, Kyiv, Ukraine, 2019. URL: https://old.npu.edu.ua/images/file/vidil_aspirant/dicer/%D0%94_26.053.03/Vorozhbyt.pdf.
[13] Microsoft, Microsoft Quantum, 2022. URL: https://azure.microsoft.com/solutions/quantum-computing/.
[14] QuTech, Quantum Inspire, 2022. URL: https://www.quantum-inspire.com/features/.
[15] Amazon Web Services, Amazon Braket: Accelerate quantum computing research, 2022. URL: https://aws.amazon.com/braket/?nc1=h_ls.
[16] Five years ago today, we put the first quantum computer on the cloud. Here's how we did it, 2021. URL: https://research.ibm.com/blog/quantum-five-years.
[17] IBM Quantum, 2021. URL: https://quantum-computing.ibm.com/.
[18] IBM Quantum Services – Simulators, 2021. URL: https://quantum-computing.ibm.com/services?services=simulators.
[19] Contact IBM Quantum, 2021. URL: https://www.ibm.com/it-infrastructure/us-en/resources/campaignmail/quantum-computing/contact/.
[20] V. Silva, Practical Quantum Computing for Developers: Programming Quantum Rigs in the Cloud using Python, Quantum Assembly Language and IBM QExperience, Apress, 2018.