Educational Matters
Научно-методически статии

FEATURES OF USING KODU GAME LAB IN TEACHING PROGRAMMING IN ELEMENTARY SCHOOL

[1)]Adel Kaplan,[2)]Dmitry Pavlov, [3)]Myrad Myradov
State Educational Institution "School № 2009" – Moscow (Russia)
[2)]Moscow State Pedagogical University – Moscow (Russia)
[3)]State Educational Institution "School № 1190" – Moscow (Russia)

Abstract. The article presents the research results on the development of methodological approaches to the propaedeutics programming in primary school using Kodu Game Lab environment. The authors have considered a variety of current trends in primary education programming, and in addition,connection between learning programming with the development of computer thinking, the formation of a "new literacy" and the achievement of the results of primary education. The authors identify seven methodological aspects of using Kodu Game Lab and give examples of their implementation. In addition, the article takes into account the lack of sufficient statistical support for the analysis of the effectiveness of the implementation of the developed approaches and formulates further ways of research. But also brings the arguments in favor of the use of the above methodological approaches.

Keywords: computer science; primary school; programming; propedeutics of programming; Kodu Game Lab

Introduction

Much attention is paid to teaching Informatics at the primary education level. In particular, at the 11th International conference ISSEP 2018 on teaching Informatics at school, held in St. Petersburg on October 10 – 12, 2018, a significant part of the presentations were devoted to the propaedeutic stage of teaching Informatics. Experts also discussed modern interpretations of Seymour Papert' s works and the use of the Logo language in the propaedeutics of teaching programming to primary students. It was noted that *"reading and writing skills enhanced by applications of mathematics are a crucial aspect of the development of logical thinking style"* (Loyo, 2018).

Also there were expressed other points of view. For example, some experts noted that traditional approaches and, in particular, the use of the Logo language is not optimal at the propaedeutic stage. To implement the propaedeutics of teaching

programming to primary students, it was proposed to*"identify primary cognitive operations in programming at the initial level and on the basis of the results to develop tasks of different levels of complexity that will support a smooth process of learning programming skills of primary school students, as well as the formation of computational thinking, with careful observance of the appropriateness of the development of relevant activities"* (Gujberova & Kalas,2013). These trends became the beginning of pedagogical research in early learning programming (Grozdev & Terzieva, 2011). They formed the basis of the didactic system of teaching computer science in different countries (Grozdev & Terzieva, 2015).

The term computational thinking today is one of the most discussed among specialists in the field of computer science teaching (Bosova & Kaplan, 2018). Today, both foreign and domestic experts pay attention to the study of this phenomenon. E. K. Henner in his works notes the term "ComputationalThinking", literally translated into Russian as «Вычислительноемышление», but this translation is a forced option, because *"the English word Computational, except for use in a purely mathematical sense (production of calculations), is currently used in parallel in a broader sense, related to the term "Computing" – the collective designation of the wholeness of computer sciences (Computer Science), information technologies and information systems, computer and software engineering"* (Henner, 2015). It is noted that computational thinking *"forms such qualities as patience, self-confidence, efficiency; develops the ability to solve previously unknown problems, cope with unknown problems; affects the ways of existence and thinking"* (Kalash, 2012). The development of computational thinking in informatics lessons today is increasingly becoming a defining task of the propaedeutic course of computer science both in Russia and abroad (Bosova, 2018).

But the term "computational thinking" is found today related to computer science not only in scientific and methodological literature. If JanetteWing and her followers today talk about computationalthinking as a self-sufficient phenomenon (Wing, 2006), then in the works of Lieberman D. A., Linn M., C. Barannikov, V. A. Dobryakova, M. A. Pinskaya, assumptions are made about the connection of elements of computational thinking with the comprehensive and harmonious development of modern personality in total. This thesis is confirmed in the conclusions of the international project "Key competencies and new literacy", in which computational literacy is one of the components of the "new literacy" (Frumin & al., 2018).

Regarding this, the authors of the article made an assumption about the need to introduce the basics of programming in the curriculum of primary school students. While sharing the view that Logois not the optimal language for propaedeutics of programming, meanwhile the authors of the article were not ready to consider the Emil environment, due to the fact that it requires payment (Bosova & Kaplan, 2018), and after analyzing the results of the Scratch Maths research project, decided to abandon the start of training using the Scratch environment (Kalas & Benton, 2017).

Among the promising and popular approaches in the teaching of informatics currently stands out gamification technology, which, according to experts, is not only *"designed to create such an information and learning environment that would contribute to the independent and active desire of students to obtain knowledge, skills and abilities, such as critical thinking, decision-making, teamwork"* (Varenina, 2014), but also proved its applicability in teaching programming (Pavlov, Butarev & Balashova, 2018). To implement their idea the authors chose KoduGameLab environment, however, faced with the lack of developed methodological approaches to teaching the basics of programming using this environment. Regarding this, the authors decided to develop methodological approaches to the implementation of propaedeutic programming course in primary school using Kodu Game Lab.

Methods

During the work authors have used the following methods of theoretical studies: analysis of scientific and methodical literature, abstraction and the method of incomplete induction. In the experimental and practical part, the authors conducted experiments, as well as observations of students, measurement and comparison of learning outcomes.

In the analysis of educational and methodical literature, the concept of "computational thinking"was clarified. In particular, in the interpretation of one of the authors of this term Seymour Papert: *"Computational thinking is a way of solving problems by people, and not an attempt to liken human thinking to computers. Computers are boring and tideous, and people are smart and imaginative. We - humans make computers efficient. Equipped with computing devices, we use our minds to solve problems that we could not solve before the computer era and create systems with functionality limited only by our imagination"*. A little later, one of the followers of Papert-Jeannette Wing formulated the following version of the definition: *"Computational thinking is the thought processes involved in the formulation of problems and their solutions in such a way that the solutions are presented in a form that can be effectively implemented with the help of information processing tools"* (National Research Council, 2010).

Working on the analysis of the term "computational thinking" E. K. Henner made the following conclusion: *"the formation of computational thinking can be considered in conjunction with the formation of information and communication competence and information culture. In process terms, they can be inseparable, but as a result of education, computational thinking retains relative independence in this triad: a person with computational thinking must be focused on solving problems with the help of infocommunication technologies, habitually think in appropriate categories. Perhaps this is the main feature of computational thinking; its presence should be an important personal and meta-objective result of school (and not only school) education, as well as its tool."* (Henner, 2016).

Also, the results of the international project "Key competencies and new literacy" conducted by representatives of the Institute of education of the National Research University "Higher school of Economics", Moscow city pedagogical University (Russia), Institute of Education, University College London (UK), Ontario Institute for Studies in Education, University of Toronto (Canada), Graduate School of Education, Peking University (China), College of Education, Seoul National University (South Korea), Evidence Institute Warsaw university (Poland), Lynch School of Education, Boston College (USA); Faculty for Educational Sciences, University of Helsinki (Finland) and aimed at conceptual clarification of the ongoing transformation of school education.

One of the results of the project was the clarification of the concept of literacy, or more precisely the formulation of the concept of "new literacy", which is a development of the traditional concept of "literacy" and consists of five components, namely three groups of competencies and two types of literacy. The competencies include:

– Competence of thinking;
– Competence of interaction with people;
– Competence of interaction with yourself;

The components of literacy include:

– Basic instrumental literacy;
– Basic special modern knowledge and skills.

The authors carefully considered the concept of "basic instrumental literacy". It is an integral component of the "new literacy" and "is based on the use of modern communication tools based on sign systems, implies the transformation in modern technological conditions of the usual literacy" read + write + count" adjusted for the formats of interaction and methods of information transmission, including in the mode of "human — human" and "human-machine". (Frumin & al., 2018).

Basic instrumental literacy is a composite concept consisting of:

– Reader component;
– Mathematical component;
– Computational (algorithmic) component;

The analysis of the works of the project participants showed that the computational component of basic instrumental literacy in fact is the development of computational thinking, which confirms the relevance of the work of the authors.

As for Kodu Game Lab environment, only few materials are presented on the use of this environment in the Russian scientific and methodological literature. Some authors consider the use of Kodu Game Lab as a tool for the development of algorithmic thinking style and note that "the Algorithmic thinking is

applicable not only in computer disciplines, but also in general education subjects" (Cheburina, 2017). We also consider a promising combination of Kodu – Scratch as two consecutive stages of mastering the basics of programming in primary school. This approach has already been implemented into the course "Informatics for all" issued by the publishing house "BINOM. Knowledge laboratory"under the editorship of A. V. Goryachev.

Some experts suggested using Kodu Game Lab as a propaedeutics of robotics training (Belenov & Samsonova, 2015), and also noted that this environment is the best suited for the use of gamification technology in programming training, as well as that the use of Kodu Game Lab environment *"allows you to update the content of initial Informatics and adapt it to the planned results, which are set by the Federal State educational standard Primary General education (GEF NOO)"* (Kaplan, 2018).

However, despite some attention to using Kodu Game Lab, the authors did not find relevant methodological developments, which also confirmed the relevance of the work.

The experimental part of the study was conducted in schools of Moscow and the Moscow region in 3 classes. The substantive basis of the experimental part was a course of training in the basics of programming for the 3rd grade, implemented in the Educational and Methodical Complex (UMK) "Informatics for all". The authors observed the development of the material, analyzed the involvement, performance, general trends and differences in the results shown by the students. Various pedagogical techniques and technologies were used. The results were generalized and systematized.

Results

As a result, the following features techniqueswere highlighted:

– Quick entry of students into the topic;

– Mastering the method of acting, instead of memorizing a sequence of actions;

– The development of the mode of action instead of listing;

– Differentiation in the implementation of the results of formalized tasks;

– The implementation of the project activities within the lesson;

– Conducting research on the laws of the virtual environment;

– Development of skills of group work on a digital project;

Let us consider in detail each result obtained.

Rapid entry of students into the topic

The figure № 1 shows taken screen shots in two different programs. The left shot is Kodu Game Lab environment, the right one is Minecraft game. The experiment showed that all students from 1st to 11th grade see significant similarities between these two software products.

Figure 1. Comparison by children of the user interface of two programs

Moreover, seeing Kodu Game Lab on the screen for the first time, primary school students exclaim "O, Minecraft!" and in addition, discuss that during the lesson they will play.

Exploiting this not-too-obvious adult resemblance is a powerful motivating tool in the hands of the teacher. Students should not be dissuaded that they are waiting for the game! The primary level of interest is so high, and the Kodu environment Toolkit is so diverse, that by the time students begin to realize that Kodu Game Lab and Minecraft are different things, their interest in the game has already been replaced by an interest in creating the game.

Mastering the method of action instead of memorizing the sequence of actions

One of the main problems in teaching students to use a particular environment is the algorithmic approach to learning. Based on those developed in the 70s of the twentieth century, this approach has long been considered promising. In particular, it was noted that *"In the course of work with the use of algorithmic method in solving problems, students establish links between the concepts learned during the study of the topic, which leads to a meaningful application of the gained knowledge, there is confidence in their powers and abilities. The main characteristic features of the algorithmic method are determinism, mass character and efficiency"* (Landa, 1966). However, the practice of teaching shows that during the development of the environment, algorithmization of actions when performing operational tasks does not allow students to memorize the correct order of actions that is connected with uniqueness of thinking of today's students.

The solution of this problem was proposed in the framework of the system of developing training D. B. Elkonin-V. V. Davydov, where it is proposed: *" action,*

and not by memorizing the partials". In addition, primary school teachers often note that modern students are much less attentive to the words spoken by classmates, which often leads to the fact that within a few minutes of the lesson, a number of students can ask the same question, which the teacher has already answered.

Kodu Game Lab contains excellent tools to avoid these difficult moments and teach children not to ask for help, which often hides a desire to shift the solution of the problem on the shoulders of the teacher, and to master the way to find the answer to the question "How to perform a specific action" in the Kodu Game Lab environment.

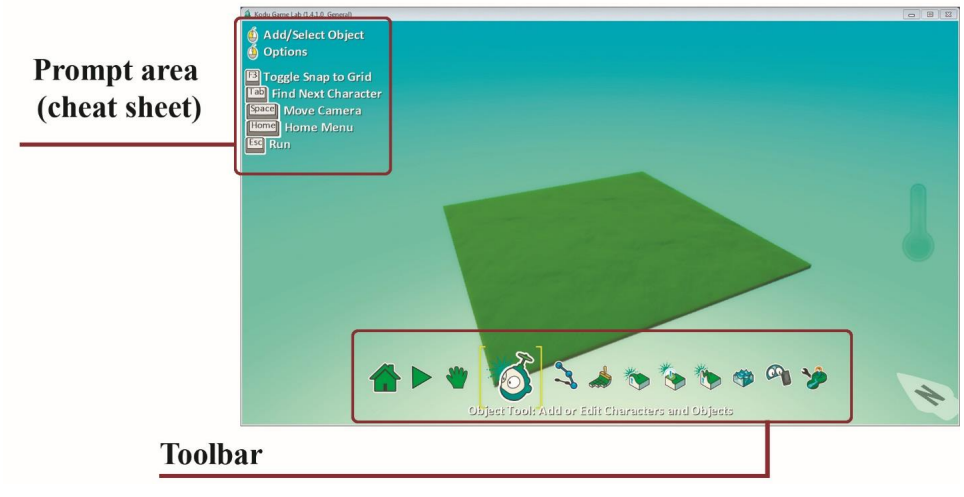

Figure 2. The main elements of Kodu Game Lab interface

For qualitative reclamation of the features of this environment, students need to learn two ways of action. The first is to check the active tool. This will help to avoid numerous questions like "Why do I click on an object and nothing happens". From the first lesson, students should be told that if something goes wrong on the map in edit mode, they should check which tool they have chosen. That is, the student does not use a ready-made algorithm, but creates it by answering questions:

Problem: when you click on an object, it isn't highlighted!

Search area: toolbar at the bottom of the screen.

Search result: a Yellow frame stands on the "Hand" tool.

Correction method: select the "Object" tool.

Telling students this method of action already in the first lesson, the teacher should reflect on this at the end of the class and do it early next 2 – 4 lessons at the stage of actualization. If during the lesson the teacher receives a question, the

essence of which is an incorrectly chosen tool, the teacher needs to attract the attention of children and together with them, without telling the way of correction, remember where to look for the problem.

The second method of action is designed to minimize the questions: "How to do..." concerning the Kodu environment Toolkit. Figure 2 shows the tooltip area (similar to the context menu), the content of which varies depending on the selected tool. In fact, it is a cheat sheet, and teaching children to use it-one of the primary tasks, if the teacher does not want the whole lesson to answer the same type of questions. Students will not remember what and how to do if the teacher will constantly think for them and the questions "how to change the size of the brush", "how to change the color of the object" and so on will be the main ones in each lesson, distracting students from the new material.

That's why the teacher needs to introduce students to the "cheat sheet" at the first lesson, analyze the change in content depending on the selected tool and report the following method of action: "If you have forgotten how to use a particular tool, look for the answer in the upper left corner of the screen". This method of action, as well as the previous one, is reflected at the appropriate stage of the first lesson and at the stages of actualization of 2 – 4 lessons, as well as situationally, when the question "How to do..." appears.

Yes, on the one hand, it is not so difficult to report a ready answer. But then students will accumulate" undeveloped» technical skills, which will interfere with their creativity, as well as the creation of programs, and the lesson will turn into a series of similar answers and questions. While the development of the two ways of action presented by children will allow them to always be "in good shape" and will contribute to the achievement of meta subject results of primary education, in terms of the formation of regulatoryUniversal Learning Activities (OOD).

The development of the mode of action instead of listing

Another problem when teaching programming in school is listing – the presentation of the finished program instead of mastering the skills of constructing algorithms.

When using Kodu Game Lab in primary school, this problem can be solved by using cross-subject relationships, named syntactic work (Pavlov, 2018).

For example, if the first program that we learn with children, should be a program of movement of the object controlled by the keys, then before we understand how to create this program, we will invite students to speculate. The reasoning, for the convenience of its integration into the Kodu program, should begin with the word "when".

1. When will what happen?
2. When the **keys** are pressed!
3. When **what keys** will be pressed?
4. When the **arrow** keys are pressed!

5. What happens **when the arrow keys are pressed**?
6. The **object** will **move**!

When the questions are asked and arranged in the order "from condition to action", students can disassemble the technology of collecting the program, according to the formulated algorithm. You will get the program shown in the figure.

Figure 3. Example of the program for movement

After that, students can be given a number of tasks for independent development. For example, to press the spacebar, the performer jumps. The teacher should not show the finished program at this point, but it is necessary to make a text description of the future program together with the students, in the way of asking questions, as in the earlier example. And in the future it is necessary to teach children to ask questions, and not to wait for the finished program so, for example, asking the following sequence of questions:

1. When will what happen?
2. When I see a **tree**!
3. What **tree**?
4. **Green**!
5. What happens **when I see a green tree**?
6. The **object** will experience **emotion**!
7. What **emotion**?
8. **Joy**!

Figure 4. Example of a program to display emotions

Students will be able to make a program as shown in Figure 4 themselves.

In the future, the ratio of examples, which deals with the preparation of a specific program (listing), to examples, compiled by students through a chain of questions should be 1/5 or even 1/7. Listing - examples are valid in complex, new cases, such as the primary parsing of situations where one condition leads to 2 or more actions at a time, or when explaining how to work with variables.

As a result of the development of the proposed method of action, instead of the traditional listing, the pace of the lesson increases, and students gain the ability to independently design a program for a specific performer, which will be useful to them in the main school when mastering the relevant section of computer science.

Differentiation of results in the implementation of formalized tasks

Despite all the requirements associated with the individualization of the learning trajectory, the need for individualization of results and other modern psychological and pedagogical trends, we are often forced to work with children in an extremely formalized framework. So, at the basic school level, a program that calculates the roots of a quadratic equation or, for example, performs bubble sorting of an array will be the same for all students. Unless names and formatting of variables features will differ.

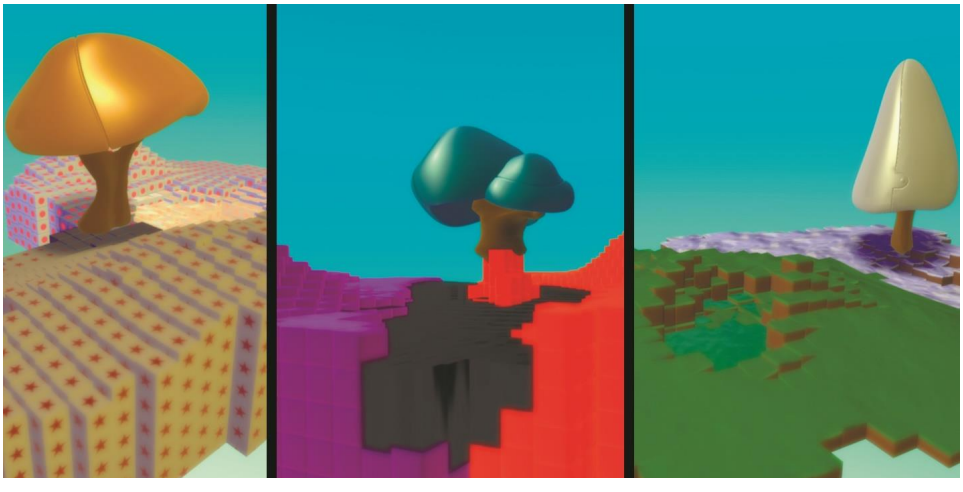

Figure 5. Examples of children's work

The figure 5 has three works of children, who tried to perform the following task: to "the Created world must have two types of surface, two hills, one tree and one pond". And despite the fact that the task is very much formalized (such tasks are often found at the initial stage of training), the works look VERY diverse and do not look like one another, although each of them meets the established criteria.

This effect increases the involvement of students in the learning process and at the same time does not require serious costs from the teacher, allowing you to concentrate on the content of training and creating unusual tasks.

Implementation of project activities in the framework of the lesson

Much attention is paid to project activities, particularly at the primary education level. It is noted that "design and research activities allow to reveal the individual

abilities of children of primary school age and gives them the opportunity to apply their knowledge, benefit and publicly show the achieved result" (Blednova, 2012).

Today, many school teachers are forced to treat the method of projects differently. Excessive formalization of the education system leads to the fact that often homework or certain types of traditional work in the classroom, in order to meet the formal criteria of reporting, are declared project work. Meanwhile, teachers often note that the implementation of the project within 45 minutes of the lesson is a task, if it's not impossible, then extremely difficult, requiring long preparation.

It is difficult to disagree with this, especially if we remember that the project *"is a complex of search, research, calculation, graphic and other types of work performed by students independently (in pairs, groups or individually) for the purpose of practical or theoretical solution of a significant problem"* (Pozdeeva & Kuznetsova, 2006).

However, after 7 – 8 classes in Kodu environment, in other words, after mastering the main methods of action and features of the environment, every second lesson begins to meet the formal criteria of project activity (Polat, 2002). The child more and more independently formulates the task, it has for the pupil suppose subjective, but novelty and significance, for its realization the pupil independently plans a course of works, masters new skills and receives at an output a digital product.

Thus, the use of Kodu Game Lab environment allows not formally, but fully implement the method of projects, harmoniously integrating it into the training program and not spending disproportionately serious resources on training, but only implementing the educational program of the initial course of Informatics.

Conducting research on the laws of the virtual environment

Unlike Logo, Emil or Scratch – Kodu Game Lab is a full-fledged virtual environment that lives according to certain laws. These laws are often akin to the familiar laws of physics, but they also have their own characteristics that are not described in advance anywhere.

Most often we do not notice this, telling children everything that "they need to know" in advance, and missing an essential pedagogical component. Namely-Kodu Game Lab environment is a wonderful research simulator! In this environment, students can learn the skills of systems analysis and research, really discovering something new.

Simple example: objects in a Kodu environment can move at different speeds. The command "move" you can add modifiers like "quick" or "slow". But there is no description of how much faster our performer will go if you tell him to "move+-fast". You can miss this moment, but you can invite children to create a long straight track, prepare a timer and detect how much normal movement differs from fast or slow, filling out the appropriate table.

Such a study is still akin to a simple observation, but it raises a number of additional questions:

– Is it possible to make two or more identical modifiers? Does it affect speed?
– Do modifiers work the same for different performers?
– "Move + fast + slow" is the same as just "move"?
– Do different surface types affect acceleration-deceleration?
– Does the slope affect at the acceleration-deceleration?

All these are questions for independent research, on the example of which students can work out the skills of planning and conducting research, creating models and presenting results.

In addition to the mentioned example in our work interests raised questions:
– Which falls faster from a height – an apple or a ball?
– Which of the objects that fell into the water will float and which will drown?
– If two characters controlled by the computer must touch to eat each other, then which of them will eat the other first?

And many other questions that can distract students from the game and make them feel like researchers, without being distracted from the course of computer science.

Development of skills of group work on a digital project

Not every student can master all the wisdom of programming and just find himself in it. However, the modern digital product is not the result of an individual startup. This is a complex fruit of the work of entire teams, in which not all are programmers. Therefore, students who find programming difficult can still become part of the creative team creating a computer game. And then the project activity comes to our rescue again. If students have already mastered the functionality of Kodu Game Lab and are able to perform individual open tasks, some tasks should be done in groups, with the division of tasks. So the following roles are interested:
– Author-there may be several authors in the team. All members of the group can offer some ideas and participate in the development of the idea;
– Level designer-the person who designs the levels, making them beautiful and comfortable;
– Programmer-a person who "animates" the world, responsible for the work of performers in the created world;
– Tester – a person who tries out the game in a variety of modes, passing judgment on the quality of the game, suggesting improvements and pointing out errors. Such a specialist can be invited from another team;
– Promotion specialist-a person who will be able to talk beautifully about the work done and vividly and eloquently present the project;

Such an approach shouldn't become the main one in the implementation of the course, otherwise temporary difficulties may alienate gifted children from programming, but it can also allows to find new interests for students who have shown a tendency to programming, and did not find themselves in it.

Summary

The methodological approaches obtained in the course of theoretical search and practical research allowed the authors to prepare a list of methodological recommendations and developments that should help primary school teachers and informatics teachers working with younger students to effectively and organically implement the course of propaedeutics of programming using Kodu Game Lab. The effectiveness of the proposed approaches can be judged by the following indicators:

– At the average level of "outflow" of students from groups of extracurricular activities during the year 25 – 50%, outflow from groups on the basics of programming was on average no more than 1 – 2 students per group per year;

– Taking into account the maximum capacity of one class equipped with computer workstations, there was not only a high occupancy rate of groups, but also an increase in the number of propaedeutics groups opened in the school twice in three years;

– Within three years, the amount of knowledge and skills mastered by children in the course of 34 hours became 2/3 more than the original, the number of creative and project tasks increased by half, which indicates a higher density and efficiency of classes using the developed methodological approaches;

However, at the moment, despite the theoretical background, the authors do not undertake to judge the impact of the developed approaches on the formation of computational thinking and the impact on the results of primary general education. Not enough data has been collected for this and this aspect of the study needs further development.

In addition, the authors of the study to date have not carried out the processing of learning outcomes using methods of mathematical statistics. In the next 2 years, a significant expansion of the research base of schools is planned. The results obtained from 10 – 12 different schools in 3-4 regions will, according to the authors, be more relevant.

Thus, the suggested methodological approaches certainly need further testing, discussion and development, but at the same time already today can be useful for primary school teachers and teachers of Informatics working with younger students in the implementation of a propaedeutic programming course using Kodu Game Lab.

REFERENCES

Loyo, A. H. (2018, October). Effects on the School Performance of Teaching Programming in Elementary and Secondary Schools. *In International Conference on Informatics in Schools: Situation, Evolution, and Perspectives*, 30 – 41. Springer, Cham.

Gujberova, M. & Kalas, I. (2013, November). Designing productive gradations of tasks in primary programming education. *In Proceedings of the 8th Workshop in Primary and Secondary Computing Education*, 108 – 117. ACM.

Grozdev, S. & Terzieva, T. (2015). A didactic model for developmental training in computer science. *Journal of Modern Education Review*, 5(5), 470 – 480.

Grozdev, S. & Terzieva, T. (2011). Research of the concept of algorithmic thinking in teaching computer science, *The International Scientific-Practical Conference "Informatization of Education – 2011", Elec: EGU Bunin, 14–15 June,* pp. T1, 112 – 119. (in Russian)

Bosova, L.L. & Kaplan, A.V. (2018). International Conference on School Computer Science ISSEP 2018. *Informatics at School*, 9 (142), 2 – 6. (in Russian)

Henner, E. K. (2015). The body of computer science knowledge and the content of the school subject. *Informatics and Education*, 7 (266), 24 – 32. (in Russian)

Kalash, I. (2012). Computational thinking and pre-school education. *Collection of materials of the Annual International Scientific and Practical Conference "Education and Training of Young Children"*, 1,50. (In Russian)

Bosova, L. L. (2018). School informatics in Russia and in the world. *Informatics at School*, 3 (39), 134 – 145. (In Russian)

Wing, J. M. (2006). *Computational thinking*. Communications of the ACM, 49 (3), 33 – 35.

Frumin, I. D. et al (2018). Universal competences and new literacy: what to learn today for success tomorrow. Preliminary findings of the international report on trends in the transformation of school education. *National Research University Higher School of Economics, Institute of Education.* HSE.2 (19) (in Russian)

Varenina, L. P. (2014). *Gamification in education*. ISOM, 6 (2) (In Russian)

Pavlov, D. I., Butarev, K. V. & Balashova, E. V. (2018). On the prospects for the use of gamification technologies for early learning in object-oriented programming. *Modern information technologies and IT education*, 4, 977 – 985 (in Russian)

National Research Council. (2010). *Report of a workshop on the scope and nature of computational thinking*. National Academies Press.

Henner, E. K. (2016). Computational thinking. *Education and science*, 2 (131), 18 – 33. (In Russian)

Cheburina, O. V. (2017). Formation of algorithmic thinking in learning game programming. *Science and Prospects*, 2, 1 – 5. (In Russian)

Belenov, N. V. & Samsonova, O. S. (2015). Robotics in extracurricular activities as a factor in the development of technical abilities in students. *International scientific review*, 4 (5). (In Russian)

Kaplan, A. V. (2018). The use of gamification technology in propedeutics of programming in primary school. *Informatics at School*, 6 (139), 65 – 67 (in Russian)

Landa, L. N. (1966). *Algorithmization in learning*. Enlightenment. (in Russian)

Pavlov, D. I. (2018.) On the connection of informatics in primary school with other disciplines. *Informatics at school*, 6, 63 – 64. (In Russian)

Blednova, E. V. (2018). Project activity as a condition for the development of the creative abilities of primary school students. *Municipal education: innovation and experiment*, 6, 28 – 31. (In Russian)

Pozdeeva, S. I., Kuznetsova, T. V. (2006). Design activities in the practice of anprimary school teacher. *Bulletin of TSPU*, 10, 65 – 68. (In Russian)

Polat, E. S. (2002). Method of projects: typology and structure. *Lyceum and gymnasium education*, 9, 9 – 17. (In Russian)

✉ **Ms. Adel Victorovna Kaplan**
ORCID: 0000-0002-8581-5086
Primary school teacher
Moscow State Pedagogical University – MSPU
School № 2009
16, bldg. 1, Admiral Rudneva St.
117042 Moscow, Russia
E-mail: adel.caplan@ya.ru

✉ **Mr. Dmitry Igorevich Pavlov**
ORCID: 0000-0002-0074-0899
Senior Lecturer
Institute of Mathematics and Informatics
Moscow State Pedagogical University – MSPU
14, Krasnoprudnaya St.
107140 Moscow, Russia
E-mail: di.pavlov@mpgu.su

✉ **Mr. Myrat Vepayevich Myradov**
ORCID: 0000-0002-8860-2561
Secondary school teacher
School № 1190
Moscow State Pedagogical University – MSPU, Moscow
21A, Pyatnitskoye Highway
125430 Moscow, Russia
E-mail: myratmyradov1997@gmail.com