Programming teaching methodological foundation improvement

Programming teaching methodological foundation improvement

D A Gramakov, M V Shevchuk, V G Shevchenko[1] and E M Chukalovskaya

Moscow Region State University, 141014, Vera Voloshinoy str, 24, Mytishchi, Moscow Region
Corresponding author: vg.shevchenko@mgou.ru

Annotation. In the paper an analysis of various programming languages previously used for educational process are carried out. Criteria are proposed to be followed for choosing a language in programming courses that will result in the improvements in the study. The basic methodological approaches to teaching programming in the course "Languages and programming methods" are described. The requirements for training tasks are developed, which will gain the interest in programming. Ways of further improvements to the teaching process are discussed.

1. Introduction

The aim of each teacher of any subject is that his (her) students acquire knowledges. To this end he (she) may use several methodologies relevant for the subject he teaches, and probably use these methodologies for some years. At the same time an active teacher should always question the methodologies used and reflect on how to improve learning in a continuous learning cycle. The subject associated with learning programming have features defined by some criteria.

1. the choice of a programming language for the taught subjects. The choice if language should be based on where the students can benefit in the future from the experience gained by studying this language.

2. the programming paradigm that will be used at the basis of the taught subject. The most common paradigms for programming teaching are an imperative, object-oriented, and functional paradigm.

3. the availability of tools of application development that are available for a particular programming language. The application development tool must be an open source program or a program with a license that entitles you to use it for free. The programming environment should be able to use various templates with basic structure of this type of application.

4. the availability of teaching materials on the used programming language in the taught discipline. It must be admitted that there is a significant lag between the dynamics of the development of the programming language and the emergence of appropriate teaching materials.

5. it is necessary to determine what types of applications will be used in teaching. Development of applications with a graphical interface arouses interest in programming and contributes to a better learning of the material of the subject.

And the last but not the least, learning of programming enables abilities to solve problems not only in programming itself but also of a wide spread due to the current development of digital economics.

In any subject correlated to programming a methodology of computational thinking should be used, the idea of which was formulated in the work [1]. All of the above criteria should be considered when planning a teaching methodology for program.

2 Methods

The subject "Languages and programming methods" is taught at the Physics and Mathematics Faculty of Moscow Region State University for many years in the departments: "Mathematics and

Computer Science", "Physics and Computer Science" and "Computer Science". This subject is the basis for the formation of key concepts related to the programming.

Different programming languages such as Pascal, Delphi, Visual Basic, Java, and Visual Basic.Net were basic for teaching in various periods of time. They played a role in the formation of methodologies used in the subject "Languages and programming methods." The Pascal was used when Borland company was one of the leaders in developing of programming environments and its Turbo Pascal language went through certain stages. However, now Pascal is a morally obsolete and does not satisfy modern requirements. The structure of the program code plays an important role in this. In Pascal-like languages, all variables are global and described at the beginning of the program code in the field of data description. The use of global variables for the whole code can lead to certain collisions, when different parts of the program have access to them. Such an approach is not welcomed in most modern programming languages. Authors of Pascal language, as well as Borland itself, have stopped developing it. Nicholas Wirth took a path of creating other Pascal-like programming languages, using the same syntax and semantics, and created sequentially the languages: Modula, Modula2, Modula3. Other extensions of the Pascal are the Oberon and Zonnon languages, but they are not widespread currently. This is partly due to the fact that they were created in the academic environment as a result of theoretical research and mostly for educational purposes. At the same time, these languages had a certain impact on the development of modern programming languages. Currently, they have some support in the .Net Framework of Microsoft, and even compilers were implemented in this environment for their usage but this did not add them popularity. The Pascal language was more fortunate - on its basis the ABCPascal language was developed as well as ABCPascal programming environment. This language and its environment have certain advantages, determined by the fact that one can use almost all the features of the .Net Framework.

However, based on the global trend in the development and application of programming languages, C-like programming languages now prevail. The most common difference between C-like languages is the use of curly braces instead of the words “begin” and “end”. Examples of such languages: Java and its derivatives, executed in JVM, C #, Rust, Go and many others. In addition, the advantage of C-like languages is the presence of the main function, which is the starting point of program execution. The execution process occurs through the launch of those program constructs that are involved in the code of this function, which makes it easier to organize debugging and testing of programs, as well as the development process.

Attempts to use Java language in the same role encountered a problem associated with the lack of a sufficient amount of educational materials and documentation from the developers for non-english speaking countries.

Visual Basic and Visual Basic.Net have been also used as basic programming languages for some time. But at the end of 2010th, it was decided to switch on to teaching programming on the basis of C# language. This choice was influenced by non-conventional syntax of the Visual Basic language and its verbosity for writing ordinary code constructions. Therefore, it could be more difficult for a teacher to switch from either Visual Basic or Pascal languages to C-like languages, which application in the world permanently increases. The C# language meets all the requirements to programming languages formulated in the introduction. In addition, the availability of modern integrated environment for developing applications, as well as documentation translated into Russian, containing a lot of educational materials and code examples in addition to the description of the language itself became the key factors determining the choice of C# as basic. In addition to Russification, the programming environment is available in various versions released at different times and intended for use on different versions of the Windows operating system. This factor allows its use by students, whose computers were purchased in a certain time interval.

3 Results

Most of the students did not study programming at school. Thus, teaching starts from an explanation of computer architecture, a description of the compilation and interpretation processes, a brief historical overview of programming languages. It should be mentioned that a similar approach is used in many foreign programming books [2]. But there it is most often caused by the absence of the subject of computer science in schools. It is necessary to give a description of the programming environment, to talk about what is a usual order of action which result writing, compilation and execution of the program. The initial tasks should be aimed to perform actions that students are well aware of. After getting acquainted with the programming environment, students become familiar with the process of writing simple programs. Programs show how to perform arithmetic operations, comparison operations and logical operations. Solving such tasks that do not require a complex algorithm allows to remove a certain fear of the coding process and working with the programming environment. The next exercises may be to calculate algebraic expressions containing a certain set of variables and including elementary mathematical functions studied in high school. In the C # programming language, mathematical functions are implemented as a Math class. Here difficulties arise due to the fact that the concepts of class and object are practically unknown even for those who studied programming elements in high school. A typical computer science curriculum [3-4] implemented in line of the Federal State Educational Standards (the second generation) does not imply the active introduction of object-oriented programming in the process of teaching programming in high school. As well as the sample curriculum of the previous standard [5], programming training is reduced to familiarize with algorithmic structures, which, according to the authors of these documents, will allow you to teach programming. However, modern programming is not limited to the use of only algorithmic structures, it has gone further towards to using other structures. Most programming environments integrate with huge class libraries. Students acquire the first skills in using classes by solving programming tasks on using mathematical functions. Further training in the course "Languages and Programming methods" involves a more detailed acquaintance with the paradigm of object-oriented programming. It is necessary to form the concepts of inheritance, polymorphism and encapsulation, and also pay attention to abstraction, which serves as the basis for the formation of classes. Two approaches are possible. First, when creating applications is based on existing classes. This approach is used when creating GUI applications. Almost all programming languages have a set of controls or widgets that allow to build an application with a graphical interface. Each such control is a class. This class has a large set of properties that characterize its state and events that can occur with this control.

In this case programming comes down to placing a certain set of elements on a basic element called a form or window and establishing links between these elements. The necessary characteristics of the controls can be set by writing program code. The second approach is usually used for creating console applications (applications without a graphical interface). It uses a class that is defined in the current application and solves a certain type of problem. In the beginning of learning of object-oriented programming, this method created knowledge about the structure of the class and the methods of defining its members.

Only after creating simple classes in programs on the basis of ordinary concepts (transport, animals, books, professions, equations, etc.) one can start using classes from existing class libraries and develop applications with a graphical interface. Constructs with branch and cycle are introduced after students learn the basics of object-oriented programming, because they are most often used inside methods for developed classes. Among the cyclic constructions, special attention is paid to the foreach cycle, which is a new cyclic construction. The teaching methodology also includes acquaintance with method overloading, which allows implementing polymorphism, and acquaintance with a new concept such as a delegate. Arrays are considered to be traditional data structures that have a size formed by their description and whose data type is given by numerical types. Besides, some attention is paid to arrays whose data types can be structures and classes. The concept of a universal type is also considered.

4 Discussion

Programming tasks are the foundation of learning progress. It is a core for acquiring knowledges and skills necessary to form abilities to apply programming not only to specific tasks of programming but also for the wide field of tasks. Handbooks of programming were usually designed to solve problems based on imperative programming. Therefore, it is necessary to make the handbooks include new tasks that taking to account the features of imperative programming, as well as object-oriented programming. The considered methodology of teaching programming requires a certain correction due to increasing between-subjects and meta-subject integration at school.

Tasks should include not only traditional tasks related to mathematics or computer science, but also the implementation of tasks related to natural sciences (physics, chemistry, biology), as well as other disciplines of the school cycle. This will allow students to develop computational thinking and prepare them for work in a digital society. It is necessary to concentrate on the development of game type tasks, which usually cause a more active interest in programming. At the same time it is possible that a programmable game is built using existing class libraries to create game applications.

Two approaches to using existing libraries are possible here. First consists of offering basic blocks of code which based on a certain library and asking the students to program a game according to a given description. The students develop an algorithm of interaction between these blocks of code and program the algorithm by combining them. This option simplifies the use of the library, since it does not require a detailed acquaintance with it. Students should only understand the proposed basic blocks of code, their functional purpose and its use in applications. This option can be used in practical exercises for any students. The second approach offers students to get acquainted with the library after studying the main classes. Further, the students design the game scenario themselves or develop the proposed version of the scenario. In both cases they develop an application algorithm based on the library used. This option is more appropriate for individual work of students and the organization of student-centered learning. At the same time, teamwork on the project is possible if 2-3 students develop a joint game application.

5 Conclusion

Teaching programming is an ideal tool for developing computational thinking skills, as it includes solving problems using a computer. The learning process does not consist only in acquaintance with the basic concepts of the studied programming language, descriptions of what they are intended for. It also teaches you to think, analyze the situation, and apply the most appropriate solution methods. Solving these types of problems requires considerable efforts in abstraction, especially if it is built on the basis of object-oriented programming. It is important that the students understand that the ability to write a correct, effective, well-structured and properly documented programs is a prerequisite for the efficient use of computers in the era of the digital economics.

6 References:

[1] Wing J. 2006 Computational Thinking *Communications of the ACM*. (*Electronic Materials* Vol 49) chapter 3 pp 33–35.

[2] Tony Gaddis 2019 *Starting Out with Python* (Spb: BHV - Petersburg)

[3] Approximate basic educational program of basic general education (*Electronic Materials)* https://fgosreestr.ru/registry/primernaya- osnovnayaobrazovatelnaya-programma-osnovnogo-obshhego- obrazovaniya-3

[4] Approximate basic educational program of secondary general education (*Electronic Materials)* https://fgosreestr.ru/registry/primernaya-osnovnaya- obrazovatelnaya-programma-srednego-obshhego-obrazovaniya/

[5] Computer science and information technology: an approximate program of basic general education (*Electronic Materials)* http://window.edu.ru/resource/183/37183/