Y. Makhenko, Y. Stelmashenko
Research supervisor: O.M. Kryvonos
Candidate of Pedagogical Sciences, Associate Professor
Zhytomyr Ivan Franko State University
Language tutor: T. Biloshytska
Candidate of Pedagogy Sciences
Zhytomyr Ivan Franko State University

LEARNING OBJECT-ORIENTED PROGRAMMING AT SCHOOL

Introduction. In connection with the rapid development of the latest technologies in programming, it is increasingly possible to meet the object-oriented approach to programming, so it is clear why in the computer science school course they begin to study such topics as "Paradigms and technologies of programming", after studying which students should be able to design object-oriented architecture of software solutions, create object-oriented software solutions, etc. Since this course is just beginning to be introduced into the school curriculum, it is found only in the programs of specialized classes, where computer science is studied from elementary school. Therefore, although it is clear that education that includes the study of the topic will be of higher quality, teachers try to bypass this selective block due to the problem of selecting methods for learning object-oriented programming and the complexity of presenting the material. But taking into account all the advantages, such as: the development of object-oriented thinking and abstract ideas in students, it becomes clear why such a topic should be studied in standard-level classes as well.

Presenting main material. The term object-oriented programming is used to mean many things, the key term here being "object". Objects are entities that combine procedural and data properties as they perform computations and maintain local state. The uniform use of objects contrasts with the use of separate procedures and data in conventional programming. An object consists of two elements: data and

functions (called methods) that work with the data. For example, objects are strings, and data are letters that make up a string. This applies not only to strings, but also to integers, decimals, and even functions.

An object-oriented approach to programming is based on manipulating objects. This means that the development of program logic is achieved by defining classes of various objects and using the interaction of objects.

The first attempts to teach object-oriented programming at school proved that students who had previously studied programming at least at a basic level learn OOP worse, due to a completely different thought process, than those who first understood the basics of OOP and then began to develop knowledge of programming. This phenomenon was called a "paradigm shift" and began to appear in sources more and more often.

So it becomes clear that when a topic such as "Programming Paradigms" is introduced into the school curriculum in the school computer science course, it is necessary to introduce the study of object-oriented programming into the computer science course in advance before learning the basics of programming. This statement is held by certain teachers and scientists. Thus, scientists investigating this concept proved in the course of experiments that "interaction with objects from the very beginning helped students build a concrete understanding and provided appropriate conceptual models. Those who study object-oriented programming can easily focus on the concepts of class and object relationships instead of focusing on the facts of structured programming."[1]

However, there is also the reverse statement that the sequence of study is of no importance. Thus, as a result of experimental research by two scientists, it was announced: "The main result of the study says that there is no difference between learning object-oriented programming first, and then the basics of programming (OOP-first), or vice versa - first the basics of programming, and then OOP (OOP-later) - it does not affect the learning result".[2]

It is also worth mentioning experimental studies that show differences in the understanding of object-oriented style programming texts. As a result of this

experiment, a sharp contrast between the mental representations of imperative and object-oriented programs was revealed. While comprehension of imperative programs was generally better than that of object-oriented programs, perceptions of imperative programs focused on program-level knowledge. On the other hand, the mental representations of object-oriented programs focus more on domain-level knowledge.[3]

As a result, it becomes obvious that object-oriented programming must be studied in a mandatory module of the program, and it does not matter whether it will be preceded by the study of the basics of programming, because, analyzing the research conducted on this topic, it becomes clear that object-oriented -oriented programming expands the possibilities of thinking under any conditions of its study.

Literature

1. The Effects of Objects-First and Objects-Late Methods on Achievements of OOP Learners. URL: https://www.scirp.org/html/23962.html?pagespeed=noscript (date of application: 01.10.2022).
2. Empirical comparison of objects-first and objects-later. URL: https://dl.acm.org/doi/pdf/10.1145/1584322.1584326?casa_token=xL2ji8lMSTUAAAAA:zivCJixx7uP2Q3K3ECycHzMuGBZpjNicBpHBhtvMSrTB4YsK0H-0B607TzZbMp1_SyriZWDwYp-D (date of application: 01.10.2022).
3. An Empirical Study of Novice Program Comprehension in the Imperative and Object-Oriented Styles. URL: https://dl.acm.org/doi/pdf/10.1145/266399.266411 (date of application: 01.10.2022).