УДК 378.1, 004.432.2
DOI: 10.25559/SITITO.15.201901.200-206

Two-Level Study of Object-Oriented Programming by University Students

T. A. Dmitrieva, A. V. Prutzkow*, A. N. Pylkin
Ryazan State Radio Engineering University, Ryazan, Russia
* mail@prutzkow.com

Abstract

The difficulties of learning the paradigm of object-oriented programming by university students are considered. Two approaches to learn object-oriented programming: Object-First (learning this paradigm at the start of programming learning) and Object-Later (prior learning of structured and procedural programming) are described. The expediency to use the second approach consisting of two levels is proved. The requirements to knowledge and skills being necessary for object-oriented programming are formulated. A review of programming languages used in both levels is presented. The conclusions are made; recommendations to study object-oriented programming are given.

Keywords: object-oriented programming study, Object-First approach, Object-Later approach, structured programming, procedural programming.

For citation: Dmitrieva T.A., Prutzkow A.V., Pylkin A.N. Two-Level Study of Object-Oriented Programming by University Students. *Sovremennye informacionnye tehnologii i IT-obrazovanie* = Modern Information Technologies and IT-Education. 2019; 15(1):200-206. DOI: 10.25559/SITITO.15.201901.200-206

Двухуровневое обучение студентов вузов объектно-ориентированному программированию

Т. А. Дмитриева, А. В. Пруцков, А.Н. Пылькин
Рязанский государственный радиотехнический университет, г. Рязань, Россия
* mail@prutzkow.com

Аннотация

Представлены трудности изучения концепции объектно-ориентированного программирования студентами. Описаны два подхода к изучению объектно-ориентированного программирования: Object-First (изучение этой концепции с началом изучения программирования) и Object-Later (предварительное изучение структурного и процедурного программирования). Обоснована целесообразность применения второго подхода, состоящего из двух уровней. Сформулированы требования к знаниям и умениям, необходимым для изучения объектно-ориентированного программирования. Представлен обзор языков программирования, используемых на первом и втором уровнях. Сделаны выводы, даны рекомендации по обучению концепции объектно-ориентированного программированию.

Ключевые слова: обучение объектно-ориентированному программированию, подход Object-First, подход Object-Later, структурное программирование, процедурное программирование.

Для цитирования: *Дмитриева Т. А., Пруцков А. В., Пылькин А. Н.* Двухуровневое обучение студентов вузов объектно-ориентированному программированию // Современные информационные технологии и ИТ-образование. 2019. Т. 15, № 1. С. 200-206. DOI: 10.25559/SITITO.15.201901.200-206

Introduction

Object-oriented programming has been dominating in the field of programming languages since 1990s. Modern industrial software development is made on the programming languages based exactly on this paradigm. Therefore, mastering object-oriented programming is necessary for university graduates.
The purpose of the study is to demonstrate modern approaches, their peculiarities as well as the problems of object-oriented programming study met by future programmers.
The authors of the paper teach the students of different directions and specialties connected with software engineering to learn programming in Ryazan State Radio Engineering University.

Programming Paradigms

In object-oriented programming a program is considered to be the interaction of objects. Each object is an instance of some class. An object is characterized by state (a set of instance variables) and functionality (a set of methods). Object-oriented programming is the development of two preceding paradigms: structured and procedural programming.
E. Dijkstra was the originator of structured programming in the end of 1960s [1, 2]. The fundamentals of structured programming are as follows: using only structured control flow constructs of concatenation (sequence), selection and repetition, development of programs using only top-down development. Structured programming allowed to decrease the time for program development and debugging as the correspondence between the step of computing and the fragment of program realizing it could always be determined. Mastering the technology of structured programming allows an average level skilled specialist to design programming code of increased complexity that will be abundant in this case.
Increasing complexity of software systems led to structured programming being transformed into procedural (procedure-modular) programming in late 1980s. Structured programming had no clearly defined criteria to divide programs into modules. Procedural programming introduced the criterion of modularity. This criterion was based on the concept of abstraction levels [3, 4]. Each routine solved one task on its own abstraction level directly using lower level routines.
The evolution of programming paradigms allowed to develop program systems of higher complexity in comparison with former one making development tools also more complex in mastering and usage. Therefore, the process of programming study has also become more complicated.

Approaches to the study of object-oriented programming

Object-First approach
Object-First approach (sometimes – Object-Early) presupposes the study of object-oriented programming without any prior study of other paradigms. The basis for this approach is the fact that structured programming and object-oriented programming are mostly nothing else but the way of thinking. In the first case the task is described as an algorithm, in the second one it is seen as object interaction and interconnection. If you start learning structured programming first followed by object-oriented one, it'll be harder to switch from one way of thinking to the other.
Learning by structured and procedural programming
The approach called also as Object-Later (or Imperative-First), presupposes prior study of structured and procedural programming by students. After this step only they start learning object-oriented programming.
Comparison of the approaches
The papers that compare these approaches make ambiguous conclusions. The article [5] considers Object-First approach to be efficient. The discussion given in [6] represents the advantages and drawbacks of both approaches. The opinion presented in [7] shows the approaches being different only by topic sequence.
Our experience in teaching students programming using these approaches allows us to make the conclusion about higher efficiency of Object-Later approach compared to Object-First one. The whole material given below describing this approach while teaching the students object-oriented programming is the result of 20 year experience of our work in the university.

Requirements to knowledge and skills necessary to study object-oriented programming

The experience with different approaches to teaching the students programming allowed us to formulate the requirements to knowledge and skills being necessary to learn object-oriented programming.
To start the study of object-oriented programming a student should know the foundations of programming and be able to represent the solution to the task as a program. The text of a program should correspond to the paradigms of structured and procedural programming.
Further basic knowledge and skills necessary for a student to master object-oriented programming are listed.
Programming foundations:
– data types;
– operations (including mathematical ones);
– control structures;
– work with complex data types;
– data input and output;
– program debugging;
– program text design according to programming languages conventions.
Structured programming:
– writing or transforming program text into the form corresponding to the paradigm of structured programming;
– structured top-down development.
Procedural programming:
– principles to divide programs into routines;
– ways to exchange data between main program and routines;
– local and global variables;
– formal and factual parameters;
– creation of user module libraries.
First-year students usually have no skills and knowledge mentioned above. Consequently, the study of object-oriented programming is realized on two levels.

Two-level study of object-oriented programming

The first level is the level when the foundations of programming, structured and procedural programming are studied. The knowledge and skills of students achieve the required level. During the second level on the basis of skills received before, the study of concepts as well as skills in object-oriented programming takes place. Prior procedural programming experience is a predictor of success for the object-oriented language, but it is not necessarily a predictor of success for object-oriented programming [8]. Therefore in both levels students learn programming technologies but not programming languages. Programming language is considered to be the means of programming technology realization. The task of deep mastering the language itself is also achieved.

Object-oriented programming is intended to develop complex programming systems. Therefore the knowledge of structured and procedural programming is of vital importance for students to get positive results. This fact proves the necessity of two-level study.

First level of study

Tasks for the first level of study

The task on the first level of study is to learn students to program in such a way for their skills by the end of this first level to correspond necessary requirements mentioned above.

In this level of study the students learn paradigms of structured and procedural programming. Different programming languages can be used here. Below we shall review programming languages as well as their peculiarities used on this level.

Programming languages for the first level of study

Currently a great number of different programming languages can be observed, both classic and modern ones. Here the question arises on what programming language should be chosen to receive the knowledge and skills listed above.

Basic. It was created in 1964 as an interactive language to be learned by students-nonprogrammers enabling them to create programs independently in a short period of time [9]. Basic belongs to the family of high-level programming languages as it has a large number of different dialects oriented to interpret program code (leading to the decrease of program runs performance). Nowadays from simple, even primitive language Basic transformed into independent language called Visual Basic being used to develop applied programs for Windows operating system.

The disadvantages of Basic can be listed as follows.

1. Basic variations of the language don't allow to teach students structured programming [10]. The reason for this is the originality of control structures and routines being substantially different from the syntax of other programming languages.

2. Visual Basic language is not intended for primary step of learning but for the solution of other tasks, particularly, for office programming.

Pascal. This language is purposefully created by N. Wirth in the end of 1960s – the beginning of 1970s to teach students structured programming [11]. Pascal is necessarily characterized by the division of a program into features (sections describing variables, types, constants, procedures, functions, etc.), as well as by strict data types and unambiguous interpretation of any command [10].

The same as Basic, the language has different dialects with introduced changes. The most famous realization of Pascal language providing language distribution and development is considered to be Turbo Pascal created by Borland [12]. Further development of Pascal is Object Pascal language implemented by the means of object-oriented programming. Object Pascal is used in Borland Delphi programming environment where considerable language extensions were introduced [13].

At present in order to learn programming with the help of Pascal in academic process we use open source development environment Pascal ABC possessing the whole potential for study. This environment functions on .NET platform. Programming language of Pascal ABC.NET environment includes not only classic Pascal language but also the majority of possibilities offered by Object Pascal and all modern programming language features such as classes, interfaces, lambda expressions, garbage collection and many others. These features allow to separate the study into necessary stages mentioned in the article given, i.e. to receive basic knowledge in the sphere of programming first, and deepen the knowledge received with the skills in object-oriented programming later.

The shortcoming of Pascal in study is the fact that in practice it is applied only in creating simple programs as it is quite difficult to use it for complex program development.

Python. This programming language suits both study and complex projects programming. Python is powerful high-level object-oriented programming language [14], developed by G. Rossum in the end of 1980s.

This is a portable language with sequential syntax, modules and simple scaling leading to easily read source codes of programs written in Python. Developed programs contain less number of code strings in comparison with other programming languages.

Python supports different programming paradigms including structured, procedural, object-oriented and functional programming. This language has interactive console where the command can be entered and the result can be immediately seen. Besides, Python is an interpreted programming language, i.e. any word processor will work with it. For convenient work an open source integrated development environment PyCharm Community Edition can be used.

Python is the highest growing programming language; according to the IEEE Spectrum programming language rating made in 2018 it ranks the first [15]. According to other ratings this language also features the leading ranks.

The main disadvantage of Python usage for programming basics is the lack of formalization; it has dynamic typing when the type of variable is determined only in the course of its execution. This fact can lead to the number of errors difficult to be found by a novice programmer. The other disadvantage is caused by a short period of its learning in our country leading to insufficient usage of this language here.

Nevertheless, Python can be definitely recommended to be used on the first level of study to receive initial programming skills, as well as the second level of study while learning object-oriented programming technology.

C. C programming language was developed by D. Ritchie in the end of 1960s – the beginning of 1970s [19]. It is a powerful, fast, protable general-purpose programming language. C is the first language used to write an operating system.

C is the language used for developing wide spectrum of applications from operating systems such as Windows and iOS to applied software for different devices and embedded systems.

This language can be considered as the language for programmer beginners. It is simple, has various features to write complex programs and is widely used throughout the world. Mastering C language helps to learn other high-level languages originating from it (C++, Java, C#).

From the viewpoint of learning the main drawback of C is the lack of full realization of procedural programming technology (has no procedure concept, a peculiar definition of function, etc.)

Ruby. It is dynamic object-oriented high-level programing language developed by Yu. Matsumoto in the beginning of 1990s. It is also recommended as the language to be taught at the start of learning programming. Ruby is characterized by dynamic typing and automatic memory control. All data here are objects unlike many other languages which feature primitive types, and each function is considered to be a method.
Ruby is used in web-development within open source web-framework Rails. According to ratings and official site of Ruby, this language enters the dozen of most popular programming languages. The growth of popularity for this language is caused by the popularity of software written in Ruby on Rails.
Ruby advantages are as follows: open source, portable, very high-level programming language having high abstraction level and objective approach in implementation of algorithms, implements conceptually pure object-oriented paradigm, provides advanced methods to operate with strings and text. The programs written in this language have good scaling and are easily maintained. Simple syntax of this language considerably facilitates the first steps of learning this language by programmers[1]. Ruby is an interpreted language, the file can be written in any word processor and run as a script using interpreter.
Drawbacks of Ruby are the following: further, more advanced learning of this language can be difficult due to insufficiency of the resources devoted to Ruby, it has lower performance compared to other languages used in web-development, it is developed rather slowly and not so popular in our country.
The experience of our teaching the discipline of programming in the first and second year of study for the specialization of 09.00.00 related to Computer Science and Computer Technology and has shown that the best languages to be taught for beginners in programming technologies are Pascal and Python.

Second level of study

Stages of object-oriented programming study
As it was mentioned above, object-oriented programming has made development tools more complicated. Therefore, the process of learning object-oriented programming includes two stages.
1. Introduction to basic concepts (class, object) and principles (inheritance, incapsulation and polymorphism including dynamic binding).
2. Study of different types of interconnections between classes and objects, requirements to class design.
Object-oriented programming introduced a large number of new interconnections between the parts of a program not found in structured and procedural programming, such as:
– interactions between data classes and utility classes;
– design patterns [16, 17];
– refactoring [18].
Requirements to class design have also been introduced [17].
Interconnections and requirements are aimed to simplify the process of program change and to expand its functionality.
The first step is studied both theoretically and in practice. The second step is mainly aimed at practical study with a little use of theoretical material due to the fact that simple example can't illustrate the essence of interconnections and requirements to design. Therefore, the second step of study is the analysis of the exercises with explaining correct and false choices of interconnections and class structures.
The second level of study also realizes learning concepts and principles of object-oriented programming, but not the programming language itself, so the choice of language is determined by the presence of required concepts and principles in the language. The languages chosen for the second step of study are C++, Java, C#, as well as abovementioned languages Python, Ruby and other object-oriented programming languages.
Software development technologies
Object-oriented programming is closely connected with software development technologies. These technologies are based on the usage of special libraries and frameworks that reduce programming time. Modern technologies replace the old ones meaning that academic process lags behind the process of industrial development. It is practically impossible to reduce this lag as the period between practical application of technologies and their study at the university equals 4-6 years. Besides, learning new technologies requires time for methodical study and preparation of new textbooks.
Nevertheless, it is necessary to introduce the students into the world of new technologies. In order to do this, free collaborate classes of university teachers and leading specialists of partner-companies are organized for students providing them with most popular and promising methods to solve the tasks given as well as forming professional competence of future specialists.

Conclusion

The following conclusions should be made.
1. Mastering the paradigm of object-oriented programming is compulsory for a student who learns programming. A number of difficulties can arise while learning this paradigm.
2. To study object-oriented programming a student should have knowledge and skills of structured and procedural programming.
3. To teach object-oriented programming two approaches are used: Object-First and Object-Later. The first approach concerns the study of object-oriented programming without prior studying the foundations of structured and procedural programming. According to the second approach, the foundations of structured and procedural programming are studied first; the skills received on the first step are made use of in the process of studying object-oriented programming.
4. While teaching the students object-oriented programming we have used both approaches. Object-Later approach has shown the best results judging by our experience.
5. The review of programming languages used on the first and second levels is made. According to our experience, the best language taught on the first level are Pascal and Python, on the second level – C++, Java, C#.
6. Learning on the second level should be two-staged: 1) introduction to basic concepts and principles of object-oriented programming; 2) studying the types of interconnections between classes and objects, requirements to class design.
Programming paradigms considered represent the main direction of programming languages evolution. The usage of functional and logic programming paradigms for the first steps in programming study is considered in [20].

[1] Eggleston L. Your First Language: Ruby vs Python [Electronic recourse]. Course Report. March 2, 2018. Available at: https://www.coursereport.com/blog/ruby-vs-python-choosing-your-first-programming-language (accessed 25.01.2019). (In Eng.)

The problems and peculiarities of studying object-oriented programming presented in the article can be different in different universities and for other specializations which opens the field for discussion on the topic.
It is reasonable to master some part of the material at the start of studying programming languages in secondary school.
It is also rational to start learning programming in secondary school allowing teachers to provide university students with additional programming technologies. Currently, the Unified state exam in computer science covers such programming languages as Basic, Pascal, C++ and Python, which means studying these languages at school.
At present software development companies experience quite serious lack of qualified specialists in this field. The President of the Russian Federation in his messages determines digitalization as a priority direction in the development of Russian economy which means the application of high-tech breakthrough technologies. Consequently, to prepare qualified programmers knowing modern technologies is a cutting-edge task nowadays.

References

[1] Dijkstra E.W. Structured programming. In: Buxton J.N., Randell B. (eds.) Software Engineering Techniques: Report on a Conference Sponsored by the NATO Science Committee. Rome, Italy, 27th to 31st October 1969, pp. 84-88. NATO Scientific Affairs Division, Brussels, Belgium, 1969. (In Eng.)

[2] Dahl O.-J., Dijkstra E.W., Hoare C.A.R. Structured programming. No. 8. London-New York: Academic Press. VIII, 1972. (In Eng.)

[3] Dijkstra E.W. The structure of the "THE"-multiprogramming system. *Commun. ACM*. 1968; 11(5):341-346. (In Eng.) DOI: 10.1145/363095.363143

[4] Liskov B.H. A Design Methodology for Reliable Software Systems. Proceedings of Fall Joint Computer Conference (AFIPS'72), AFIPS, 1972. p. 191-199. (In Eng.)

[5] Bennedsen J., Caspersen M.E. Teaching Object-Oriented Programming – Towards Teaching a Systematic Programming Process. Proceedings of the 8th Workshop on Pedagogies and Tools for the Teaching and Learning of Object-Oriented Concepts, 18th European Conference on Object-Oriented Programming (ECOOP 2004). Oslo, Norway, 2004. Available at: http://cs.au.dk/~mec/publications/workshop/11--ecoop2004.pdf (accessed 25.01.2019). (In Eng.)

[6] Lister R., Berglund A., Clear T., Bergin J., Garvin-Doxas K., Hanks B., Hitchner L., Luxton-Reilly A., Sanders K., Schulte C., Whalley J.L. Research perspectives on the objects-early debate. Working group reports on ITiCSE on Innovation and technology in computer science education (ITiCSE-WGR '06). ACM, New York, NY, USA, 2006. p. 146-165. DOI: (In Eng.) 10.1145/1189215.1189183

[7] Ehlert A., Schulte C. Empirical comparison of objects-first and objects-later. Proceedings of the fifth international workshop on Computing education research workshop (ICER '09). ACM, New York, NY, USA, 2009. p. 15-26. (In Eng.) DOI: 10.1145/1584322.1584326

[8] Govender I. From Procedural to Object-Oriented Programming (OOP) - Performance in OOP: An empirical study. *South African Computer Journal*. 2010; (46). (In Eng.) DOI: 10.18489/sacj.v46i0.13

[9] Basic. A Manual for BASIC, the elementary algebraic language designed for use with the Darthmouth Time Sharing System. Dartmouth College, 1964. Available at: http://www.bitsavers.org/pdf/dartmouth/BASIC_Oct64.pdf (accessed 25.01.2019). (In Eng.)

[10] Bogardus Cortez M. History of Basic Computer Programming Languages: C Language, BASIC and Beyond [Electronic recourse]. *EdTech Magazine*. June 2017. Available at: https://edtechmagazine.com/higher/article/2017/06/history-programming-languages-c-language-basic-and-beyond (accessed 25.01.2019). (In Eng.)

[11] Wirth N. The Programming Language Pascal. *Acta Informatica*. 1971; 1(1):35-63. (In Eng.) DOI: 10.1007/BF00264291

[12] Chapman M.R. In Search of Stupidity: Over 20 Years of High-Tech Marketing Disasters. 2nd ed. Apress, 2006. Available at: https://www.r-5.org/files/books/ethology/experience/Merrill_R_Chapman-In_Search_of_Stupidity-EN.pdf (accessed 25.01.2019). (In Eng.)

[13] Gibson S. Borland and Microsoft Enter the Object-Oriented Pascal Ring. *Infoworld*. 1989; 11(9):28. (In Eng.)

[14] Srinath K.R. Python – The Fastest Growing Programming Language. *International Research Journal of Engineering and Technology (IRJET)*. 2017; 4(12):354-357. Available at: https://www.irjet.net/archives/V4/i12/IRJET-V4I1266.pdf (accessed 25.01.2019). (In Eng.)

[15] Cass S. The 2018 Top Programming Languages. Python stays on top, and Assembly enters the Top Ten [Electronic recourse]. *IEEE Spectrum*. July 2018. Available at: https://spectrum.ieee.org/at-work/innovation/the-2018-top-programming-languages (accessed 25.01.2019). (In Eng.)

[16] Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns Elements of Reusable Object-Oriented Software. Addison-Wesley; 1995. (In Eng.)

[17] Martin R.C. Agile Software Development: Principles, Patterns, and Practices. Prentice Hall; 2003. (In Eng.)

[18] Fowler M., Beck K., Brant J., Opdyke W., Roberts D. Refactoring: Improving the Design of Existing Code. Addison-Wesley, Upper Saddle River, NJ; 1999. (In Eng.)

[19] Ritchie D.M. The development of the C language. The second ACM SIGPLAN conference on History of programming languages(HOPL-II). ACM, New York, NY, USA, 1993. p. 201-208. (In Eng.) DOI: 10.1145/154766.155580

[20] Vujošević-Janičić, M., Tošić, D. The Role of Programming Paradigms in the First Programming Courses. *The Teaching of Mathematics*. 2008; 11(2):63-83. Available at: http://elib.mi.sanu.ac.rs/files/journals/tm/21/tm1122.pdf (accessed 25.01.2019). (In Eng.)

Submitted 25.01.2019; revised 20.02.2019;
published online 19.04.2019.

Поступила 25.01.2019; принята к публикации 20.02.2019;
опубликована онлайн 19.04.2019.

About the authors:
Tatiana A. Dmitrieva, Associate Professor, Associate Professor of the Department of Computational and Applied Mathematics, Ryazan State Radio Engineering University (59/1 Gagarina St., Ryazan 390005, Russia), Ph.D. (Engineering), ORCID: http://orcid.org/0000-0002-3560-0482, dmitrieva.tatiana.al@gmail.com

Alexander V. Prutzkow, Associate Professor, Professor of the Department of Computational and Applied Mathematics, Ryazan State Radio Engineering University (59/1 Gagarina St., Ryazan 390005, Russia), Dr.Sci. (Engineering), ORCID: http://orcid.org/0000-0002-4110-5269, mail@prutzkow.com
Alexander N. Pylkin, Professor, Head of the Department of Computational and Applied Mathematics, Ryazan State Radio Engineering University (59/1 Gagarina St., Ryazan 390005, Russia), Dr.Sci. (Engineering), ORCID: http://orcid.org/0000-0002-9112-4716, pylkin.a.n@rsreu.ru

All authors have read and approved the final manuscript.

Об авторах:

Дмитриева Татьяна Александровна, доцент, кафедра вычислительной и прикладной математики, Рязанский государственный радиотехнический университет (390005, Россия, г. Рязань, ул. Гагарина, д. 59/1), кандидат технических наук, ORCID: http://orcid.org/0000-0002-3560-0482, dmitrieva.tatiana.al@gmail.com
Пруцков Александр Викторович, доцент, профессор кафедры вычислительной и прикладной математики, Рязанский государственный радиотехнический университет (390005, Россия, г. Рязань, ул. Гагарина, д. 59/1), доктор технических наук, ORCID: http://orcid.org/0000-0002-4110-5269, mail@prutzkow.com
Пылькин Александр Николаевич, профессор, заведующий кафедрой вычислительной и прикладной математики, Рязанский государственный радиотехнический университет (390005, Россия, г. Рязань, ул. Гагарина, д. 59/1), доктор технических наук, ORCID: http://orcid.org/0000-0002-9112-4716, pylkin.a.n@rsreu.ru

Все авторы прочитали и одобрили окончательный вариант рукописи.