Characteristics Of Teaching Programming Based On Different Principles

Juraev Muzaffarjon Mansurjonovich
Dotsent, Department of Informatics, Kokan State Pedagogical Institute, Doctor of Philosophical Pedagogical Sciences (PhD)
Phone: +998 97 557 10 13.
Email address: juraevmuzaffar.kspi@gmail.com

Muzaffar Mansurovich Botirov,
Kokan State Pedagogical Institute, Informatics Department teacher

ABSTRACT

The article discusses the features of teaching students various principles of programming, the formation of students' ability to model and design a subject area with different styles based on procedural, object-oriented, functional, logical programming paradigms.

Keywords: programming paradigms, procedural, object-oriented, functional, logical, objects, programming.

In the current era of increasing competition, the requirements for training graduates of higher education institutions are also changing. One of the unique features of specialists is their creative approach to their activities, their readiness to change the type of their profession or complete retraining depending on the change of the working environment and the level of employment. There was a direct connection between the quality of the specialist's training and his ability to get a job and further growth. Therefore, in the creation of a model for the training of specialists, it is envisaged to use methods that ensure the effective development of students' abilities and skills of independent work, systematic thinking, and the ability to adapt to the rapidly changing era. When active methods of teaching are used, it will be possible to solve the above-mentioned problems at least partially.

The features of teaching students various principles of programming allow students to form the ability to model and design a subject area with different styles based on procedural, object-oriented, functional, logical programming paradigms. Objects are used in modeling, design and programming in various fields. The concept of object is common in every paradigm. However, the mechanisms that record the structure of the object and actions with it are implemented differently in different paradigms. Each style of programming requires a different approach to problem solving. It is best to use procedural and object-oriented programming principles to solve computational problems. Logical and functional programming principles are used in the design of knowledge bases and expert systems. When teaching

programming, it is necessary to use a system of specially selected tasks, while solving it, students learn the features of using various principles and methods of information processing. Programming occupies an important place in the system of training teachers of mathematics and informatics. The programming teaching system is based on the integration of different principles and different programming paradigms. There are the following programming paradigms:

- procedural;
- object oriented;
- functional;
- makes sense.

Each of the programming paradigms introduces the concept of an object. Objects are used in modeling, design and programming in various fields. The concept of object is common in every paradigm. However, the mechanisms describing the structure of the object and actions with it are implemented differently in different paradigms. For example, in object-oriented programming, a program can be viewed as a collection of interacting objects, while in logic programming, logical relationships between objects are modeled using a set of statements in a formal logic language.

Each style of programming requires a different approach to problem solving. It is best to use procedural and object-oriented programming principles to solve computational problems. Logical and functional programming principles are used in the design of knowledge bases and expert systems.

In object-oriented programming, a program can be viewed as a collection of interacting objects whose behavior is implemented using methods represented as procedures. Procedures are the heart of procedural programming. It is in the process of learning procedural programming that the principle of dividing the task into subtasks, determining their hierarchy, the mechanisms of transferring parameters to procedures is introduced, and the structure of the modules in which the procedures are placed is considered. Understanding the structure of modules and their implementation is important when learning object-oriented programming, because modules house class and object definitions.

The object-oriented approach is used in the development of a wide range of applications. Modeling real objects with classes in an object-oriented programming language is often more efficient and natural than procedural programming. Objects can be used to represent systems of a very different nature. Object-oriented approach is used in the design of user interface, databases, computer architecture.

The most important components of object approach are abstraction, encapsulation, modularity and hierarchy [2, 5]. Abstraction is the process of extracting abstractions on the subject being studied. Encapsulation is the merging of all properties that make up the state and behavior of an object into a single abstraction and restricting access to the implementation of these properties. Modularity is the principle of program development, which implies its implementation in the form of separate parts (modules). A hierarchy is a system of ordered or ranked abstractions.

Solving problems based on the object-oriented paradigm can be divided into the following steps:

setting a problem, selecting objects based on class analysis (object-oriented analysis);

definition and implementation of relations between objects, evolution of interactions and inheritance of classes, modification of classes and their distribution by modules (object-oriented design);

implementation in an object-oriented language (object-oriented programming).

When teaching students the object-oriented method of programming, it is necessary to develop the ability to formalize the task, to distinguish abstractions and objects of a certain field of science, to structure them and to implement them using object-oriented programming technology. For example, the description of the TStudent class of objects of type student in Delphi might look like this:

```
Type TStudent=class
fam:string;
gr:integer;
procedureSetfam(x:string);
functionGetfam:string;
```

procedureSetgr(y:integer);
functionGetgr:integer;
end;

Thus, objects of type TStudent will have two data fields: last name and group number, as well as read and write methods for each of the fields.

When choosing educational tasks, it is necessary to take into account the development of object-oriented programming technology, which shows the difference in approaches to solving the same problem. It is necessary to teach students to apply knowledge in real situations, to expand the range of possible applications of OOP. For this, it is recommended to solve problems with objects whose prototypes are real-life mathematical objects and structures. The basic concepts of linear algebra and analytic geometry are vector and matrix. Their model in algorithmic languages is arrays. To design an analytic geometry problem solver, you can create a Tmas class for objects of type "array" with descriptions of the fields, methods, and properties of arrays needed to solve problems.

typeTMas=class
protected
Origin: pointer; jMin, jMax:integer;
functionElemP(j:integer):RealP; {specifies the address of element j}
public
constructor Create(jMin_,jMax_:integer);
destructor Destroy; cancellation;
procedure Clearance; {null array creation method}
procedureAdd(x:TMas); {method to add array elements}
procedureSub(x:TMas); {method to subtract element}
procedureMul(x:TMas); {method to multiply element}
procedureMulx(x:real); {method to multiply an element by a number}
procedureDivx(x:real); { method of dividing an element by a number }
Function Sum:real; {method to add array elements}
FunctionPMn(x:TMas;y:TMas):real; {area calculation method
directed polygon}

swearThus, the capabilities of OOP can be effectively used in the implementation of computational algorithms.

Students' "objective" thinking ability is formed when developing visual applications using standard objects (components) of object-oriented programming systems such as Delphi and C++ systems [2,3]. Students are attracted to the possibility of creating a graphical application interface from the visual components of these systems. For example, in an object-oriented programming technology course, students are asked to create a project to solve analytic geometry problems, among many other projects. The project interface is created using a menu placed on the form. The dialog menu allows you to solve tasks.

Educational systems built on the basis of functional and logical paradigms have not been sufficiently studied compared to educational systems based on procedural and object-oriented programming paradigms [1, 244].

Logic programming belongs to the field of artificial intelligence. Logical algebra is the basis of logical approach. The appearance of strange logic increased the expressiveness of the logical approach. Odd logic is widely used in modeling [6,526].

The Prolog language is not intended for calculation, but for solving logical problems, modeling the process of human logical thinking [1,128]. Facts and rules of the Prolog program - a description of relations and connections between objects of a certain subject area, i.e. recording the conditions of a particular logical problem to be solved. Defined relationships and connections are considered statically. This approach to programming is called declarative. All this allows us to classify Prolog as a very high-level language. Prolog is particularly well-suited for solving problems based on objects and relationships between them [1,115]. Let's see this with an example program:

yoqtiradi (Lola, o‘qish).
yoqtiradi (Nargiza, badminton).
yoqtiradi (Lola, badminton).
yoqtiradi (Madina, suzish).
yoqtiradi (Madina, o‘qish).
?- yoqtiradi (X, o‘qish), yoqtiradi (X, suzish).

The question is: Are there people who like both reading and swimming? Prolog first looks for a fact that matches the first part of the question:yoqtiradi (X, o‘qish). Dasturning birinchi fakti mos keladi

Likes (Lola, reading).

And the variable X is associated with the value "Lola". At the same time, Prolog fixes a pointer to the fact list that indicates the state of the search procedure. Next, Prolog tries to match the second part of the query under the condition X = Tulip, that is, it looks for the truth "likes (Tulip, swim)" from the beginning of the program. There is no such thing in the program. Prolog then returns to the first part of the query: likes(X, read) , "frees" the variable X, and continues searching for matching facts, starting at the pointer previously set in the list of facts. read)", the variable X is set to the value lena, then the second part of the question successfully matches the truth "likes (Medina, swim)". Prolog has performed a backward search in this example. Graphically, the program execution process is a binary tree - is expressed as a traversal of the output tree.

In addition to the procedural and object-oriented approaches, as well as the logical approach represented by the Prolog language, there is another direction - the functional direction. It appeared with the creation of the Lisp programming language. Most artificial intelligence programs are written in Lisp [4, 848]. Lists in Lisp are the primary means of representing knowledge, as are objects in Delphi. For example, nested lists can be used to represent a specific feature of a person: (xodim

(ismi Anvar)

(otasining ismi Abdullayevich)

(Farmonov familiyasi)

(ta’lim (o‘rta (1990 yildan 2008 yilgacha))

(oliy (QDPI (2008 yildan 2012 yilgacha))

Farg‘ona davlat universiteti, Moskva (2016 yildan 2022 yilgacha)) (mutaxassislik)

(texnik kibernetika)

(dasturlash)

(tajriba (2016 yildan 2022 yilgacha)

Lisp's functional approach to programming is based on the idea that processing data and obtaining the desired result can be expressed as internal and / or recursive function calls that perform some actions, so the value of one function can be used as an argument to another. . The value of this function becomes the argument of the next one and continues until the final result - the solution of the problem - is obtained. Programs are built from logically divided function definitions. Definitions consist of control structures that organize calculations and internal function calls. The main techniques of functional programming are composition and recursion. All this is the realization of the ideas of the theory of recursive functions.

Thus, when teaching programming, it is necessary to use a system of specially selected tasks, while solving it, students learn the features of using various principles and methods of information processing. It should be noted that programming is one of the subjects, in order to successfully master it, it is necessary not only to apply the acquired knowledge and skills, but also to have abstract and logical thinking and research skills. In turn, teaching programming based on the integration of programming paradigms helps to develop such skills.

Conclusion

Teaching students different principles of programming allows them to develop the ability to model and design the field of science with different methods based on different programming paradigms. When teaching programming, it is necessary to use a system of tasks that show the difference in approaches to solving them. When studying different principles of programming, students learn the characteristics of using different methods and principles of structuring and processing information.

References

1. Mansurjonovich, J. M. (2022). CURRENT STATUS OF THE SCIENCE OF INFORMATICS AND INFORMATION TECHNOLOGIES IN THE PROFESSIONAL EDUCATION SYSTEM, EXISTING PROBLEMS AND SOLUTIONS, PRINCIPLES AND CONTENT OF THE

SCIENCE ORGANIZATION. Galaxy International Interdisciplinary Research Journal, 10(12), 327-331.
2. Mansurjonovich, JM, & Sattorovich, YS (2023). MAXSUS IZLAMALARDAN FOYDALANISH TA'LIM JARAYONINI TASHKIL ETISHNING MUHIM AVTOZYATLARI. Ochiq kirish ombori , 4 (3), 126-133.
3. Mansurjonovich, J. M. (2023). Designing an electronic didactic environment to ensure interdisciplinary integration in the teaching of" Informatics and information technologies" during professional education. Confrencea, 11(11), 78-82.
4. Mansurjonovich, J. M. (2022). Professional Educational Institutions Theoretical and Practical Basis of Development of the Content of Pedagogical Activity of Teachers of" Information and Information Technologies". Texas Journal of Engineering and Technology, 15, 49-53.
5. Davronovich, A. D., & Mansurjonovich, J. M. (2023). IMPORTANT ADVANTAGES OF ORGANIZING THE EDUCATIONAL PROCESS IN A DIGITAL TECHNOLOGY ENVIRONMENT. Galaxy International Interdisciplinary Research Journal, 11(2), 149-154.
6. Mansurjonovich, J. M., & Davronovich, A. D. (2023). INTERDISCIPLINARY INTEGRATION IS AN IMPORTANT PART OF DEVELOPING THE PROFESSIONAL TRAINING OF STUDENTS. Open Access Repository, 9(1), 93-101.
7. Melikyzievich, S. I., Turdalievich, M. I., Shukurovich, M. S., & Mansurovich, Z. M. (2022). THE METHOD OF REFERENCE TESTS FOR THE DIAGNOSIS OF DIGITAL DEVICES. International Journal of Early Childhood Special Education, 14(7).
8. Mansurjonovich, J. M. Description of the Methodological Basis for Ensuring Interdisciplinary Continuity of the Subject" Computer Science and Information TECHNOLOGY" in Vocational Education. JournalNX, 7(10), 223-225.
9. Mansurjonovich, J. M. (2021). Experience Of Cambridge Curricula In Ensuring The Continuity Of Curricula In The Field Of "Computer Science And Information Technology" In The System Of Professional Education. The American Journal of Interdisciplinary Innovations Research, 3(11), 26-32.
10. Juraev, M. M. (2021). PEDAGOGICAL CONDITIONS FOR THE DEVELOPMENT OF VOCATIONAL EDUCATION THROUGH INTERDISCIPLINARY INTEGRATION INTO THE VOCATIONAL EDUCATION SYSTEM. In НАУКА, ОБРАЗОВАНИЕ, ОБЩЕСТВО: АКТУАЛЬНЫЕ ВОПРОСЫ, ДОСТИЖЕНИЯ И ИННОВАЦИИ (pp. 110-112).
11. Juraev, M. M. ZY Xudoyberdiyev Theoretical analysis of the continuity model of computer science and information technology in the System of professional education. European Scholar Journal (ESJ)//ISSN (E), 2660-5562.
12. Juraev, M. M. (2022). Theoretical and practical principles of improving the content of the pedagogical activity of ICT teachers of professional educational institutions in the conditions of information of education.
13. Mansurjonovich, J. M. (2022). METHODOLOGICAL FOUNDATIONS FOR IMPROVING THE CONTENT OF TRAINING FUTURE ICT TEACHERS IN THE CONDITIONS OF DIGITAL TRANSFORMATION OF EDUCATION. АКТУАЛЬНЫЕ ВОПРОСЫ СОВРЕМЕННОЙ НАУКИ И ОБРАЗОВАНИЯ, 9.
14. Juraev, M. M. (2021). OA Qo 'ysinov Description of the methodological basis for ensuring interdisciplinary continuity of the subject "Computer Science and Information Technology" in vocational education. JournalNX-A Multidisciplinary Peer Reviewed, 7(10).
15. Juraev, M. M. (2022). The value of open mass competitions in the process of digitalization of extracurricular activities

of schoolchildren. Web of Scientist: International Scientific Research Journal, 3(10), 338-344.
16. Jo‘rayev, M. (2022). Professional ta’lim jarayonida fanlararo uzvilik va uzliksizlikni ta’minlash o ‘quvchilari kasbiy tayyorgarligining muhim omili sifatida. Zamonaviy dunyoda amaliy fanlar: Muammolar va yechimlar, 1(29), 43-46.
17. Juraev, M. M. (2022). Prospects for the development of professional training of students of professional educational institutions using electronic educational resources in the environment of digital transformation. Academicia Globe: Inderscience Research, 3(10), 158-162.