ANALYSIS CONTENT OF SECTIONS OF PROGRAMMING LANGUAGES IN TEXTBOOKS OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY IN SECONDARY SCHOOLS

Uktamjon Jumankuziyev Utkirjon ogli
Lecturer at the Department of Informatics, Kokand DPI
jumanqoziyev@mail.ru, +998972157013

ABCTRACT

The state analyzes the content of programming language sections in computer science and information technology textbooks taught in general education schools.

Key words: Paradigm programming, Advanced language programming, PC, Classification of typical tasks.

ANNOTATSIYA:

Maqolada umumiy o’rta ta’lim maktablarida o’qitilayotgan Informatika va axborot texnologiyalari darsliklarida dasturlash tillari bo‘limlarining mazmuni tahlil qilingan.

Kalit so’zlar: Dasturlash paradigmalari , Yuqori darajali dasturlash tili, shahsiy kompyuter, Tipik masalalarni sinflashtirish.

АННОТАЦИЯ:

В статье анализируется содержание разделов языка программирования в учебниках информатики и информационных технологий, преподаваемых в общеобразовательных школах..

Клчевые слова: Парадигмы программирования, Продвинутый язык программирования, ПК, Классификация типовых задач.

INTRODUCTION:

The architectonics of studying the topic of programming is presented in a rather generalized form in the form of a methodological scheme, the use of which is permissible at any level of mastering by students of program material. Traditionally, it is generally accepted that programming is, from the point of view of the educational process, a difficult topic for the mastering of the computer science course by schoolchildren in general. Particularly difficult is the solution of problems in the context of mastering algorithmicization and programming, during which students develop an algorithmic style of thinking, and as a result, the formation of general educational skills.

The tasks of the investigated section of informatics include the development of educational material on the operations of developing software for electronic computers. If we more concretize the tasks of the topic, then we should turn to the comprehension of the very word "programming", by which it is customary to understand the process during which the duration is carried out, the result of which is the program.

The process of developing system software tools and programming systems is traditionally called system programming, which includes the creation of application programs (application programming). Currently, various programming paradigms are used, which in turn determines the difference in their teaching, characterized by a number of specific features. The main programming paradigms are those noted in Figure 1.

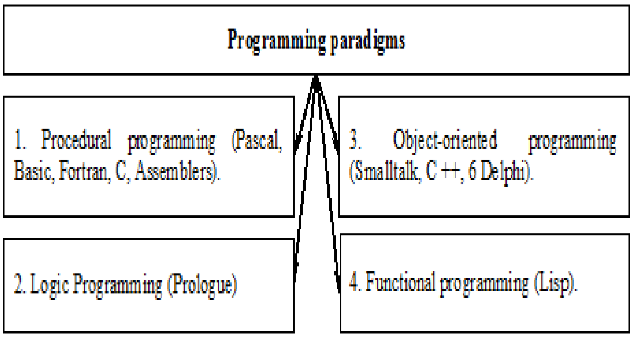

Programming paradigms
1. Procedural programming (Pascal, Basic, Fortran, C, Assemblers).
3. Object-oriented programming (Smalltalk, C ++, 6 Delphi).
2. Logic Programming (Prologue)
4. Functional programming (Lisp).

Figure 1 – Basic programming paradigms

Among the paradigms presented in Figure 1, the most used in programming practice is the procedural paradigm, which includes a sufficient layer of the program, which determines its application in educational institutions in order to solve the main problems of learning within the framework of existing curricula. Pascal and BASIC are often mastered within the framework of the named paradigm, which in what follows we will call the word "programming".

The process of mastering programming, presented in the educational program, traditionally includes three modules:

1 Module is aimed at mastering the methods by which the development of computational algorithms is carried out;

2 Module includes mastering the program itself;

Module 3 combines the actions of mastering and practical application of a specific programming system.

Next, let us turn to a more detailed consideration of the methodological scheme for studying the programming process, which for greater clarity is displayed by us in Figure 2.

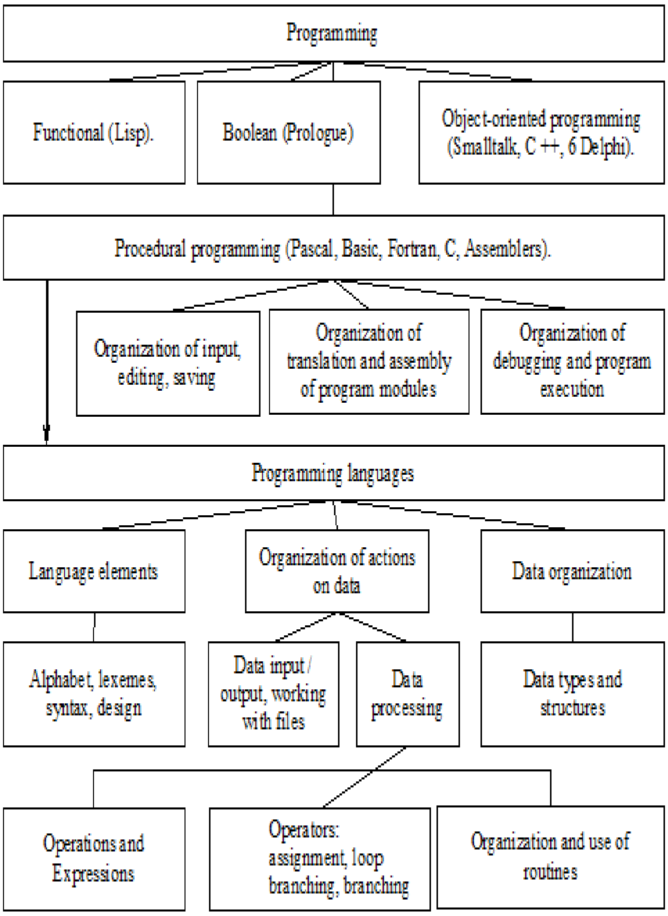

Programming
Functional (Lisp).
Boolean (Prologue)
Object-oriented programming (Smalltalk, C ++, 6 Delphi).
Procedural programming (Pascal, Basic, Fortran, C, Assemblers).
Organization of input, editing, saving
Organization of translation and assembly of program modules
Organization of debugging and program execution
Programming languages
Language elements
Organization of actions on data
Data organization
Alphabet, lexemes, syntax, design
Data input / output, working with files
Data processing
Data types and structures
Operations and Expressions
Operators: assignment, loop branching, branching
Organization and use of routines

Figure 2 Methodological algorithm for mastering the programming process

It should be noted that today the program is represented by two groups: Group I - machine-oriented (Autocodes, Assemblers); Group II - high-level programming languages. Whatever programming languages are used, the algorithm for solving a specific problem can be considered as a set of commands, the effectiveness and application of which is demonstrated to the student on personal computers. A distinctive feature of high-level programming languages is that during the execution of one command, a complex of operations is performed by the processor at once, which determines the name of the command "operator", the most significant of which is the assignment operator.

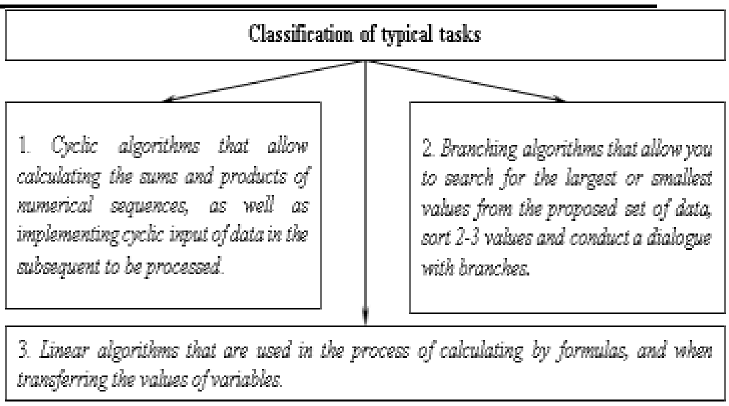
It should be noted that in high-level programming languages the named operator is written identically to the same command used in an algorithmic language, representing algorithmic structures: (branch, loop). Structural languages are commonly understood as programming languages that have structured operators (for example, Pascal and C). When choosing linking methods during the development of algorithms for working with quantities and programming languages, as a rule, two options are used:

Option I provides for the consideration of various algorithms for the purpose of the most detailed description of which block-diagrams and AN are used. After that, the rules of programming languages and ways of translating algorithms into a program by means of this language are determined;

Option II involves the simultaneous mastering of algorithms and programming languages.

It should be emphasized that the development by students of the processes of algorithmization and programming, outside of practical activities in the course of which the direct application of the learned theoretical material is carried out, is ineffective for the educational process pursuing specific learning goals. The foregoing determines the need to saturate the learning process of students with practical exercises, during which the developed algorithms are tested. To implement such practical lessons, the student must first master the programming languages, i.e. to bring to automatism the most optimal methods of work carried out by him in the programming system.

From our point of view, mastering the basics of programming should be carried out using typical tasks, in compliance with the principle of gradualness, which provides for a systematic complication of the structure of the algorithms used. All tasks used in teaching practice can be classified based on the selected features of the algorithmic structure (see Figure 3).



1. Cyclic algorithms that allow calculating the sums and products of numerical sequences, as well as implementing cyclic input of data in the subsequent to be processed.

2. Branching algorithms that allow you to search for the largest or smallest values from the proposed set of data, sort 2-3 values and conduct a dialogue with branches.

3. Linear algorithms that are used in the process of calculating by formulas, and when transferring the values of variables.

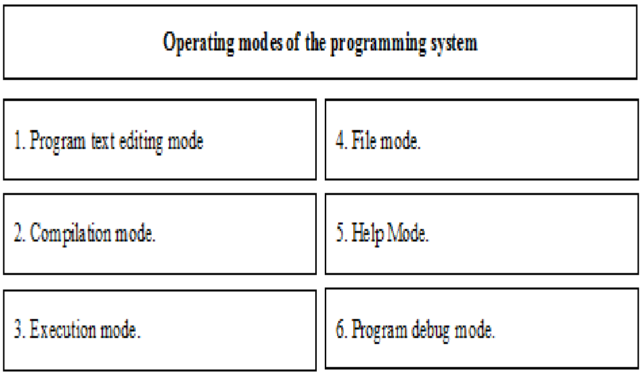
Figure 3 Classification of typical tasks based on algorithmic structure

It is necessary to pay special attention to the sequence of training, which, in our opinion, is especially important. So, in the course of teaching the repetition command, the first stage should be the introduction of a cycle with a postcondition, since such a sequence allows the student to initially think about the commands planned for implementation included in the general cycle, and then proceed to formulate the conditions for its repetition. In the event that a cycle with a precondition is initially introduced, then the student will need to perform the named actions at the same time, which will negatively affect the degree of effectiveness of the learning process.

Thus, the implementation of a cycle with a postcondition can be considered a preparatory stage, during which the basis for the "inclusion" of perception by learners is formed. This approach makes it possible to most effectively achieve the specified parameters of educational goals during the translation of knowledge and the subsequent transition to another type of repetition command (analogy). As part of the actions carried out in the process of pedagogical influences, special attention should be paid, highlighting them and emphasizing for students, the specific features of the types of the types of the cycle involved in the operation of checking the condition, and when returning to the repetition of previously performed actions. It should be especially noted the importance of only one repetition with a postcondition, the body of the loop can be executed once, while emphasizing that it may not be found at all in the repetition command with a precondition.

In the event of a misunderstanding of the principle according to which the algorithm is translated into a formal language, it is possible to follow the following solution algorithm: mastering programming languages should start with the use of basic algorithmic constructions, in parallel with various forms of the algorithm presentation, which ultimately allows creating favorable conditions for the transition to the execution of a specific algorithm.

One of the concepts that students need to master is the environment of the programming system, which should be understood as the current situation deployed on the screen (system shell). In this case, the teacher should describe and demonstrate to the student exactly the PROGRAMMING SYSTEM with which actions should be carried out within the framework of a specific topic being mastered. As the characteristic modes of the PROGRAMMING SYSTEM, it is necessary to highlight those noted by us in Figure 4.


Operating modes of the programming system

1. Program text editing mode
2. Compilation mode.
3. Execution mode.
4. File mode.
5. Help Mode.
6. Program debug mode.

Figure 4 Modes of functioning of the programming system

It should be noted that the editing mode in most cases is set automatically during the initialization of the system, during which the text editor built into the system operates. The text information presented in the program can be prepared using the tools of another editor that allows you to generate text files. The next mode, the compilation mode, is included in those systems that are used in the group of compiled languages (Pascal, CI, Fortran, etc.), and the result of the compilation is a program executed from the programming system using the machine instruction language. Quite often, the result is obtained during program execution in two stages: Stage 1 includes the actual compilation; Stage 2 ⍰ editing the existing relationships.

The mode of the execution programming system assumes that the programming system executes the program in machine instructions at the end of its translation, and already through the interpreter, it is implemented into high-level programming languages.

The mode of working with files in which the text information of the program is saved, written in the original language. Further, the translation is carried out directly and the initial data and the results obtained at the end of the above actions are fixed.

In the file mode, traditional operations are performed by the programming system, i.e. information is saved, information is transferred to RAM, a file is named, etc. Also, within the framework of the named mode, the command for outputting the contents of the editor window for printing is carried out, i.e. output of information to a file that has a relationship with the printer.

In the help mode, a hint from the programming system is sent to the executor of the program by displaying it on the screen. Such a hint greatly facilitates the programmer's activities in the course of his operations carried out with the system and programming languages.

The debug mode is available in the programming system programming level that is sufficiently developed from the point of view of professionalism. The named mode allows you to implement tracing, step-by-step program execution, monitoring changes in specific values, setting the end point of program execution (place, conditions). All of the above becomes possible thanks to the available tools designed to search for algorithmic errors that may be in the program.

Thus, relying on the methodological algorithm previously provided by us (see Figure 2) and the modes of operation of the programming system, the teacher, when organizing the learning process, must inform students about the system of commands that are used in each of the above modes.

THE LIST OF USED LITERATURE:

1) Semakin, I. G. i dr. Informatika: uchebnik dlya 8 klassa / I.G. Semakin, L.A. Zalogova, S.V. Rusakov, L.V. Shestakova. – M.: BINOM. Laboratoriya znaniy, 2012. – 176 s.
2) Semakin, I. G. i dr. Informatika. Bazovыy uroven: uchebnik dlya 10 klassa. / I.G. Semakin, Ye.K. Xenner, T.Yu. Sheina – M.: Binom. Laboratoriya znaniy, 2015. – 264 s.
3) Semakin, I. G. i dr. Prepodavanie bazovogo kursa informatiki v sredney shkole: metod. posobie. / I.G. Semakin, T.Yu. Sheina – M.: BINOM. Laboratoriya znaniy, 2005. – 245 s.
4) Semakin, I.G. i dr. Informatika i IKT: uchebnik dlya 9 klassa / I.G. Semakin, L.A. Zalogov, S.V. Rusakov, L.V. Shestakova. – 5-ye izd. – M.: BINOM. Laboratoriya znaniy, 2012. – 341 s.