CREATING COMPUTER PROGRAMS USING SPECIFIC PROGRAMMING LANGUAGES

Toshpulatov Rakhimjon Ismailovich
Kokand State Pedagogical Institute
Senior teacher of the Department of Informatics.

Abstract. This article devoted to imrove the process and art of creating computer programs using specific programming languages. It is also stated that a particular programming language is based on some guiding idea that significantly influences the style of the corresponding programs.

Key words: computer science, information technology, programming languages, algorithms, Coding and compilation, programmers.

For several decades, computer science and information technology have been associated with a stationary personal computer, local software installed on it and the presence of a network that allows data to be exchanged between the nodes of this network. However, the development of computing technology and the spread of the global network made it possible to use cloud computing, which implies the transition of computing resources from discrete devices to common centralized clusters connected via the Internet[1].

The term programming means the process and art of creating computer programs using specific programming languages. In the general sense of the word, programming is the formalization of a predetermined state, in response to an event, implemented by means of mathematics or the natural sciences. In the narrow sense of the word, programming is seen as the coding of algorithms in a given programming language. In a broader sense, programming is the process of creating programs, that is, developing software.

Programming includes:

- Analysis
- Design - development of a complex of algorithms
- Coding and compilation - writing the source code of the program and converting it into executable code using a compiler
- Testing and debugging - identifying and eliminating errors in programs
- Testing and delivery of programs
- Escort

Different programming languages support different programming styles (called "programming paradigms"). In part, the art of programming is to choose one of the languages that best suits the task at hand. Different languages require the programmer to have different levels of attention to detail when implementing an algorithm, often resulting in a trade-off between simplicity and performance (or between programmer time and user time).

The only language directly executed by the processor is machine language (also called "machine code"). As already mentioned, initially, all programmers worked out every little thing in machine code, but now this difficult work is no longer being done. Instead, programmers write source code, and the computer (using a compiler, interpreter, or assembler, which we'll talk about a little later) translates it, in one or more steps, fine-tuning all the details, into machine code ready to run on the target processor. However, in some languages, instead of machine code, an interpreted "virtual

ISSN 2277-3630 (online), Published by International journal of Social Sciences & Interdisciplinary Research., under Volume: 11 Issue: 11 in November-2022
https://www.gejournal.net/index.php/IJSSIR

machine" binary code, also called byte-code, is generated. This approach is used in Forth, Lisp, Java (Chapter 3 of the abstract is devoted to this language).

Now that we know a little about the concept of "programming", we can move on to the material part of the process of creating programs. These, of course, are technical (hardware) means of programming - a set of electrical, electronic and mechanical components of automated systems constitutes their technical support (as opposed to software, which is software of automated systems). For example, an electronic computer (computer) or computer is a set of hardware and software tools based on the use of electronics and designed for automatic or automated data processing in the process of solving computational and information problems.

Programming paradigms

A particular programming language is based on some guiding idea that has a significant impact on the style of the corresponding programs. Depending on the purpose and / or method of writing programs, programming paradigms (also known as approaches or technologies) are distinguished:

- Structured programming is a programming methodology based on a

systematic approach to the analysis, design and implementation of software. This methodology was born in the early 70s and proved to be so viable that it is still the main one in a large number of projects. The basis of this technology is the following provisions[2]:

- A complex task is broken down into smaller, functionally better

manageable tasks. Each task has one input and one output. In this case, the control flow of the program consists of a set of elementary subtasks with a clear functional purpose.

- The simplicity of the control structures used in the task. This provision

means that logically the task should consist of a minimal, functionally complete set of fairly simple control structures. An example of such a system is the algebra of logic, in which each function can be expressed through a functionally complete system: disjunction, conjunction, and negation.

• The development of the program should be carried out in stages. At each stage, a limited number of clearly defined tasks should be solved with a clear understanding of their meaning and role in the context of the entire task. If such an understanding is not achieved, this indicates that this stage is too large and should be divided into more elementary steps.

The concept of modular programming. As well as for the structural programming technology, the concept of modular programming can be formulated in the form of several concepts and provisions:

• Functional decomposition of a task - splitting a large task into a number of smaller, functionally independent subtasks - modules. Modules are interconnected only by input and output data.

• Module - the basis of the concept of modular programming. Each module in the functional decomposition is a "black box" with one input and one output. The modular approach allows you to painlessly upgrade the program during its operation and facilitates its maintenance. Additionally, the modular approach allows you to develop parts of the programs of one project in different programming languages, and then use the assembly tools to combine them into a single boot module.

• Implemented solutions should be simple and clear. If the purpose of the module is not clear, then this indicates that the decomposition of the initial or intermediate task was not done well enough. In this case, it is necessary to analyze the task again and, possibly, to carry out an additional division into subtasks. If there are difficult places in the project, they need to be documented in more detail using a well-thought-out comment system. This process should be continued until you really achieve a clear understanding of the purpose of all the modules of the problem and their optimal combination.

• The purpose of all module variables should be described using comments as they are defined.

ISSN 2277-3630 (online), Published by International journal of Social Sciences & Interdisciplinary Research., under Volume: 11 Issue: 11 in November-2022
https://www.gejournal.net/index.php/IJSSIR

Object-Oriented Programming (OOP). The idea of OOP is to link data with procedures that process this data into a single whole - an object. OOP is based on three essential principles that give objects new properties. These principles are encapsulation, inheritance, and polymorphism.

• Encapsulation - combining data and algorithms for processing this data into a single whole. Within OOP, data are called fields of an object, and algorithms are called object methods.

• Inheritance - property of objects to generate their descendants. A child object automatically inherits all fields and methods from its parents, can supplement objects with new fields and replace (override) parent methods or supplement them.

• Polymorphism is a property of related objects (ie objects that have the same common parent) to solve similar problems in different ways.

There are other programming technologies, which should also be said a little.

Application programming - development and debugging of programs for end users, such as accounting, word processing, etc.

System programming - development of general software tools, including operating systems, auxiliary programs, general system software packages, for example: automated control systems, database management systems, etc.

Declarative (logical, production) programming is a programming method designed to solve artificial intelligence problems. In this context, the program describes the logical structure of the solution to the problem, indicating mainly what needs to be done, without going into details of how it is done. Programming languages such as Prolog are used[3].

Parallel programming is the development of programs that provide simultaneous (parallel) execution of operations related to data processing.

Procedural (procedure-oriented) programming is a programming method in which programs are written as lists of sequentially executed instructions. It uses procedurally oriented programming languages.

Functional programming is a programming method based on dividing an algorithm for solving a problem into separate functional modules, as well as describing their relationships and the nature of interaction. For functional programming, the most widely used languages are HOPE and ML. Elements of functional programming are also implemented in other languages, such as C.

Heuristic programming is a programming method based on modeling human mental activity. It is used to solve problems that do not have a strictly formalized algorithm or are associated with incomplete initial data.

Toolkit of programming technology

And finally, in order to fully understand the principle of the programming system, we will consider the tools of programming technology, i.e. a set of programs that provide technology for the development, debugging and implementation of software products.

Currently, the direction associated with the technology of creating software products is rapidly developing. This is due to the transition to industrial technology for the production of programs, the desire to reduce the time, labor and material costs for the production and operation of programs, to ensure a guaranteed level of their quality[4].

Within these areas, the following groups of software products have been formed:

1. tools for creating applications (a set of programming languages and systems, as well as various software systems for debugging and supporting programs being created.), including:

• local tools that ensure the performance of individual work on the creation of programs; include programming languages and systems, as well as the user's tool environment;

• integrated software developer environments that provide a set of interrelated work to create programs that increase the productivity of programmers[5];

ISSN 2277-3630 (online), Published by International journal of Social Sciences & Interdisciplinary Research., under Volume: 11 Issue: 11 in November-2022
https://www.gejournal.net/index.php/IJSSIR

2. CASE-technology (Computer - AidedSystem Engineering), representing methods of analysis, design and creation of software systems and designed to automate the development and implementation of information systems.

I would like to dwell on CASE - technologies in more detail, because the idea of them is connected in our minds with something that has nothing to do with ordinary programming.

The means of CASE-technologies are divided into two groups:

• implementations built into the system - all design and implementation decisions are tied to the selected database management system (DBMS);

• system-independent implementation - all design solutions are focused on the unification of the initial stages of the life cycle and the means of their documentation, provide greater flexibility in the choice of means of implementation.

The main advantage of CASE-technology is the support of team work on the project due to the possibility of working in the local network of developers, export/import of any fragments of the project, organizational project management.

Within the framework of CASE-technologies, the project is accompanied in its entirety, and not only its program codes. Project materials prepared in CASE technology serve as a task for programmers, and programming itself is rather reduced to coding - translation of data structures and methods for their processing into a certain language, if automatic code generation is not provided[6].

Most CASE technologies also use the "prototype" method to quickly create programs in the early stages of development. Code generation of programs is carried out automatically - up to 90% of object codes and texts are in high-level languages, and Ada, C, Cobol are most often used as languages.

Today, the world's leading CASE system is Rational Rose by Rational Software Corporation. Rational Rose aims to create modules using the Unified Modeling Language (UML). The latest version of the company's CASE system is already being used to create commercial software and supports popular programming languages such as Java, C++, Smalltalk, Ada, Visual Basic and Forte.

Using such technologies, it is possible to interactively develop the architecture of the application being created, generate its source texts and, in parallel, work on documenting the system being developed.

The external structure of the Constitution describes its relationship with other sources of law, the totality of relations, its place and role in the legal system and its significance in the system of social and normative regulation in society.

The article presents the role of family, forming system of upbringing, traditional-educational system and traditions in Uzbekistan.

In an article consistently revealing the principles of the Bologna process for measuring the quality of education, the dynamics of internationalization and the logic of integration in European higher education and in Eurasia.

Finally, having considered almost all aspects of the programming process, we move on to the most significant component - programming languages. “Natural languages are not natural for machines,” once said the American programmer Alan J. Perlis. We will prove the correctness of his statement in the next chapter.

List of used literature

1. Muydinovich, R. I. (2022). METHODOLOGY OF USING THE GOOGLE CLASSROOM MOBILE APPLICATION IN TEACHING INFORMATICS AND INFORMATION TECHNOLOGIES FOR SECONDARY SCHOOL STUDENTS. European Journal of Interdisciplinary Research and Development, 3, 158-162.

ISSN 2277-3630 (online), Published by International journal of Social Sciences & Interdisciplinary Research., under Volume: 11 Issue: 11 in November-2022
https://www.gejournal.net/index.php/IJSSIR

2. Bishop D. Effective work: Java 2. - St. Petersburg: Peter; K.: BHV Publishing Group, 2002. - 592s.
3. Zaretskaya I.T., Kolodyazhny B.G., Gurzhiy A.N., Sokolov A.Yu. Informatics 10-11 class. - K .: "Forum", 2001 - 494s.
4. Lyakhovich V.F. Fundamentals of informatics. - Rostov-on-Don: Phoenix, 1996. - 699s.
5. Informatics: Basic course / S.V. Simonovich and others - St. Petersburg: Peter, 1999 - 640s.
6. Website materials: www.sun.ru
7. Kamolov.A. Methods Of Teaching Computer Science In Higher Education. International Journal of Advanced Science and Technology 7 (29), 2221-2224.
8. Tolibjonovich, M. T. (2021). The Constitution is a Legal Guarantee for the Development of the Country and the Well-Being of Society. *International Journal of Human Computing Studies*, *3*(2), 105-109.
9. Abdullaev, A. N. (2017). THE ROLE OF THE NATIONAL TRADITIONS AND RITES IN FAMILY UPBRINGING. *Modern Science*, (4-2), 6-8.
10. Jamoliddinovic, U. B. (2022). Origins, Dynamics and Logics Bologna Process. *European Multidisciplinary Journal of Modern Science*, *5*, 239-245.

ISSN 2277-3630 (online), Published by International journal of Social Sciences & Interdisciplinary Research., under Volume: 11 Issue: 11 in November-2022
https://www.gejournal.net/index.php/IJSSIR