TECHNOLOGY OF DEVELOPMENT OF ALGORITHMIC THINKING IN STUDENTS

[1]Aytimbetov Yusupbay
[1]Teacher of the Department of Methods of Teaching Computer Science
Nukus State Pedagogical Institute

ABSTRACT

Modern educational standards require that instilling algorithmic thinking in students today is considered a necessary component. This article discusses modern solutions to issues such as formation of algorithmic thinking in students, improvement of reflexive abilities, development of methodological competence.

Keywords: Algorithmic thinking, method, teaching, student, lesson, technology, component.

INTRODUCTION

The widespread introduction of computer technology, new information technologies and the transition to a market economy have necessitated the training of not just engineering personnel, but specialists capable of solving problems at the intersection of engineering and computer science.

The solution to professional engineering problems is based on the construction of various algorithms, their analysis, assessment and selection of the most effective solution options. Translation and implementation of software models of various information processes and systems related to the functioning of engineering objects is possible only with a fairly high level of development of cognitive abilities of specialists, in particular algorithmic thinking style (ASM). The presence of developed ASM among undergraduate students in technical areas of study not only facilitates the process of developing their skills and abilities to solve various types of professionally oriented tasks, but also creates confidence in graduates in their abilities and abilities, and increases their competitiveness in labor market.

MATERIALS AND METHODS

The beneficial influence of algorithmic activity on the formation of thought operations is scientifically substantiated in the works of M. V. Belyaeva, V. P. Bespalko, A. AND. Gazeikina, P. I. Galperina, D. E. Knut, A. N. Leontyeva, S. L. Rubinstein et al. [1–4]. Based on the works of numerical authors, it can be argued that training in programming and algorithmization shapes the methods of mental activity and develops ASM. Consequently, the methodology for teaching the programming of the laurels of technical fields should not only take this fact into account, but purposefully contribute to the development of students' cognitive abilities.

The scientific literature describes various approaches to training in algorithmization and programming. Thus, A.G. Gein and A.I. Senokosov propose to more actively apply the ideas of a structural approach; IN. N. Isakov and V. V. Isakov consider it expedient and effective to gradually increase the level of motivation of tasks; in the opinion of Ya. N. Zaidelmana, G. V. Lebedeva and L. E. Unauthorized, special attention should be paid to the constant mental work of students [5].

The relevance of the study is due to the need to use computer technologies in the process of forming algorithmic thinking of students majoring in pedagogical fields. However, this issue remains open and poorly studied scientifically due to insufficient scientific and pedagogical development of the possibilities of using computer technologies to form algorithmic thinking skills in future teachers, along with subject competence, beyond the core subjects.

The scientific novelty lies in the fact that the structural and functional method of using computer technologies in the process of forming algorithmic thinking of students majoring in pedagogical fields has been theoretically substantiated and tested.

The practical significance of the work lies in the possibility of using the research materials in the process of preparing materials for conducting pedagogical mastery classes, when writing teaching aids, as well as in further study of the topic.

Let us turn to the features of using computer technologies in the process of forming algorithmic thinking of students majoring in pedagogical fields and methods of forming algorithmic thinking in the professional training of students majoring in pedagogical fields.

RESULTS AND DISCUSSION

In pedagogical practice, when teaching algorithmization and programming, computer presentations are actively used to provide visualization of educational material; educational videos illustrating the execution of various algorithms. Recently, dynamic games, mental maps and diagrams have been increasingly used [5–9]. All these tools stimulate the development of ASM, increase the level of assimilation of educational material due to the fact that they involve both visual and auditory channels of information perception [10]. However, kinesthetic methods of information processing remain almost unclaimed, although, according to available statistics, they are the leading ones for approximately 40% of people [4]. Thus, there is a contradiction between the need to develop students' cognitive abilities when teaching algorithmization and programming and the insufficient development of tools and instruments to achieve this goal.

As mentioned above, the known methods of developing ASM are mainly aimed at activating its model and conceptual components, while the sensory component, based on kinesthetic channels of information perception, is used little or is ignored altogether. At the same time, according to the provisions of the bodily approach, which is a relatively new direction in psychology, our sensations play an important role in the formation of thinking in general and algorithmic thinking in particular [15–19]. To ensure the full development of the sensory component of ASM, special kinesthetic simulators are required, by which we mean means of developing and consolidating ASM that promote the emergence of mental images and mental schemes and take into account the relationship of bodily sensations with other channels of information perception. Based on the provisions of the information approach to learning and the bodily approach to the psychology of perception, as well as on the identified structural features of ASM, we designed a model for teaching algorithmization and programming to bachelors of technical fields, stimulating the development of their ASM [5]. This model was tested at the our University in the 2020–2023 academic years in the process of teaching the disciplines "Computer Science" and "Programming".

Taking into account that traditional methods of developing ASM practically do not address the sensory component of thinking, we will dwell on the issue of forming ASM at the kinesthetic level in more detail.

The main task of teaching programming at the first stage of developing ASM in students at the level of interest to us is the creation of an algorithmic image in the form of a mental scheme. For this purpose, we propose using kinesthetic simulators. We will show the possibilities of their application using the example of studying composite data types (array, records, sets), array sorting algorithms using the bubble method by future engineers, as well as mastering dynamic data types (stack and queue).

At the initial stage, students often confuse the value of the array element index with the value of the array element itself. Many students do not understand that all elements of an array must be of the same type, that the array must be processed element by element using a cycle; they do not understand how to access the required element of the array, prevent it from going beyond its limits, etc. Kinesthetic simulators will allow students to "touch" the array, "hold" it in their hands, and independently perform actions that students will later, as operators of a programming language, prescribe to the machine.
An array is a set of identical components, and to emphasize this, data of the same type is applied to chips of the same shape: for example, numbers are written on round chips, symbols on square chips, etc. (Fig. 1).

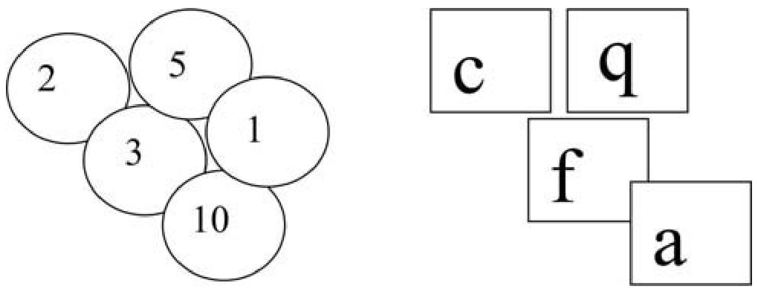

Fig. 1. The tops of date array
The training task consists of filling the created array with elements. The solution to such a task requires the use of a cycle, for the implementation of which the student must open cells 1 through N in order and put data into them. The teacher draws the students' attention to the fact that the array can be filled in two ways - from the keyboard and using a random number generator. In the first case, a chip with the desired number is taken and dropped into the box, in the second, a random chip is selected at random and the operation of the random number generator is demonstrated at the same time. Since the student has to perform the same actions several times, the meaning and necessity of using a cycle to fill the array is learned quite quickly and to the required extent. When designing such a simulator, various options for entering array elements can be taken into account. For example, you can make only one box open to simulate one element of the array, which should close automatically: or you can set the condition of extracting the contents from the box only with tweezers to show that the access to the elements of the array is carried out element by element and it is impossible to simultaneously get and hold two or more elements of the array in your hands, etc.

Practice shows that students have certain difficulties when studying array sorting algorithms. Even such a simple algorithm, which has an obvious natural association, as the bubble method, after its description at the model and conceptual levels is adequately perceived by far from all students.

The kinesthetic trainer we developed for studying sorting algorithms looks like a set of billiard balls placed on a stand. Each ball, simulating an element of the array, has a number written on it – the index of the element of the array, and this number can be erased and rewritten, since during the execution of the algorithm the student will have to swap the elements. The balls have different weights, their number is equal to N (for example, 10). The goal of the educational task is to arrange the balls in descending order of weight. To reinforce visual associations, the balls of the array are colored in such a way that when the array is sorted, the colors will be arranged in a certain sequence – in the form of a rainbow spectrum or a gradual increase/decrease in tone [6].

Since the algorithm allows comparing only two adjacent elements of the array, the student is asked to take two adjacent balls and determine which of them is heavier and, if necessary, swap the balls by rewriting the indices on them. A similar procedure is repeated with subsequent balls up to the last pair. This clearly demonstrates the need to use a cycle during which the number of repetitions N – 1 is determined. As a result of executing the algorithm, the lightest ball ("bubble") becomes the last one ("floats to the surface"), but the other balls remain unordered. Therefore, the procedure is repeated from the very beginning, but the comparison of the weight of each pair is now performed not 9, but 8 (N – 2) times: since the lightest element is already in its place, there is no need to weigh and compare the last pair of balls. When the second lightest ball takes its place, it becomes the penultimate one; the next "bubble" "floats". Having clearly seen that the process of comparing pairs of balls has to be performed as many times as there are elements in the array (= N), the student realizes the need to use a structure of two nested loops to implement the sorting algorithm. Moreover, since at each subsequent stage of the algorithm execution it is necessary to compare an ever smaller number of pairs, it becomes clear to the student why the index of the nested loop (let's say, j) changes not to N, but to N – i, where i is the index of the outer loop, which, in turn, changes from 1 to N – as many times as it is necessary to start the process over again.

A similar simulator can also be used to clearly illustrate more complex sorting algorithms – Hoare sort, bit or digit sort, which very often remain unclear after the first verbal explanation even for strong students.

The principles of working with dynamic data types are also difficult for most students. Part of the difficulty is due to stereotypes that students have when working with static data types. Some, for example, believe that any element can be accessed directly by index, but these actions are unacceptable with respect to dynamic data types such as stacks and queues. A kinesthetic trainer for studying the latter looks like a tube in which tennis balls are stored. A stack is a unidirectional list in which all insertions and deletions are performed from the end, i.e. it works on the LIFO principle (last-in, first-out). A tube for balls that opens only from one end can serve as its model. It would be good if the balls were of different colors and the students could clearly see that if the red ball was dropped into the tube first, followed by the yellow, green, etc., then in order to get the red one (for example, in order to read what was written on it), you would need to roll out all the balls that were placed in the stack later than the required object [7].

A queue is a one-way list in which elements are removed from the beginning and new elements are added to the list at the end. In this case, the FIFO (first-in, first-out) principle works. A queue model can be a tube for balls that opens from two ends, but each end must have anti-valves that work in only one direction.

Since students acquire ASM during the course of programming by constructing an image of the action algorithm and becoming familiar with the combinations of these images at each level (sensory, model and conceptual), after using kinesthetic trainers, one should move on to forming the model and conceptual components of the algorithmic thinking style.

At the second stage of training – the development of ASM at the model level – various forms of recording the problem solving algorithm (verbal description, block diagram, etc.) are presented on the basis of the constructed mental algorithmic scheme. When solving a problem, the student himself chooses the most understandable, in his opinion, type of recording the algorithm.

The third stage of mastering programming, which corresponds to the conceptual level of the formation of the ASM, is devoted to studying the syntax of formal languages of computer programs and recording the algorithm in the form of a program in one of them.

At the model and conceptual levels of the formation and development of the ASM, from our point of view, mental maps and multi-level tasks are very effective as basic teaching tools [9].

Algorithmic images can be built in different ways; for this purpose, images of one or several memory areas and their combinations in various combinations can be used. The more images from the upper levels of the ASM are involved, the more developed the person's style of thinking in question will become.

CONCLUSION

The formation of the ASM should begin at an early age, but the most active period is considered to be the school period. However, the modern school is focused mainly on stimulating the development of the logical and auditory components of the thought process. According to the professor of psychology at Harvard Linguistic University G. Gardner, education in general education institutions is designed for children with logical-mathematical or linguistic types of thinking, which are not predominant among representatives of the new generation of youth [10]. It turns out that a significant part of today's schoolchildren and students remain "overboard" of quality education. Taking these circumstances into account, we assumed that the process of formation and development of the ASM will be more effective and productive if, in addition to the visual, auditory and abstract zones of the student's memory, his motor activity is also involved. In the course of our research:

- a three-level model of teaching programming to future engineers was designed, consisting of three components - sensory, model and conceptual, corresponding to the structure of the ASM;
- based on the basic provisions of the bodily and informational approaches to the organization of the educational process, the need to develop and use kinesthetic simulators for the development of ASM in students of technical fields of training when teaching them programming was theoretically substantiated;
- it was established that performing algorithmic operations manually with the help of kinesthetic simulators contributes to the construction of algorithmic images in the minds of students in the form of mental schemes, which significantly simplify the understanding of the educational material, allowing you to write down an algorithm for solving a problem in the form of a block diagram, and then - in the form of a program in one of the formal programming languages;
- diagnostics of the level of development of the ASM (carried out using the Amthauer test) and a test check of the quality of training of students of the Siberian Federal University confirmed the effectiveness of using kinesthetic means that activate the motor zone of memory

for mastering the required skills and abilities in the algorithmization of activities and for consolidating knowledge and competencies in the field of programming.

In the future, we plan to create a full range of kinesthetic simulators that provide didactic support for all topics of the university course "Programming" and the replication of this complex by means of a 3D printer.

We also see prospects for continuing research in this area in clarifying and detailing the information model for the development of the algorithmic style of thinking and its application in the mastering of the content of other disciplines by students, for example, mathematics.

In addition, the issue of diagnosing the level of algorithmic thinking currently remains open. Most existing options for monitoring knowledge assess their quality and the effectiveness of education as a whole, whereas to identify the state of algorithmic thinking itself, it is necessary to develop a set of specific diagnostic procedures.

REFERENCES

1. Gazeikina A. I. Thinking styles and teaching programming to students of a pedagogical university // Collection of reports from the ITO conference. Moscow, 2016. Pp. 102–103.
2. Knut D. E. Algorithmic thinking and mathematical thinking [Electronic resource]. Access mode: http://ai.obrazec.ru/ai_sense.htm (accessed: 10/30/2015).
3. Leontiev A. N. Activity. Consciousness. Personality. Moscow: Smysl, 2012. 352 p.
4. Rubinstein S. L. Fundamentals of General Psychology. St. Petersburg: Piter, 2017. 718 p.
5. Balan I. V. Using mental maps in training // Young scientist. 2015. No. 11 (1). P. 58–59.
6. Gimaletdinova K. R., Muleeva A. Yu. Application of mental maps in computer science lessons // Innovative trends in the development of the education system: materials of the VI International scientific and practical conference, February 19, 2017, Cheboksary. Cheboksary: Interactive plus, 2017. P. 59–61.
7. Zhemchuzhnikov D. G. Development of dynamic games as a means of teaching programming // Bulletin of the Moscow City Pedagogical University. Series: Computer Science and Informatization of Education. 2010. No. 2. P. 49–51.
8. Ivanov A. P., Fedoseeva A. P. Development of a set of mental maps for the section "algorithmization and programming" of the school course in computer science // Problems and prospects of physical, mathematical and technical education: collection of materials of the All-Russian scientific and practical conference. Tyumen: Tyumen State University, 2015. Pp. 91–95.
9. Tony Buzan, Jennifer Goddard. Brain Training for Kids // Proactive Press, 2012. 44 p.
10. Kholodnaya M. A. Cognitive styles: on the nature of the individual mind. Moscow: PER SE, 2012. 302 p.