Journal of Modern Education Review, ISSN 2155-7993, USA
May 2015, Volume 5, No. 5, pp. 470–480
Doi: 10.15341/jmer(2155-7993)/05.05.2015/005

http://www.academicstar.us

A Didactic Model for Developmental Training in Computer Science

Sava Grozdev[1], Todorka Terzieva[2]

(1. Institute of Mathematics and Informatics, Bulgarian Academy of Sciences, Bulgaria;

2. Department of Software Technologies, Faculty of Mathematics and Informatics, Plovdiv University Paisii Hilendarski, Bulgaria)

Abstract: The paper presents a didactic model based on Bloom's expanded taxonomy of learning and developed for a formation of algorithmic thinking and implementation of developmental training in Computer Science for first-year students. The fundamental elements of this model of the learning process are the actions which occur during the course. An educational environment and teaching technology for formation of algorithmic thinking was created through a system of learning tasks. Emphasis was placed on the formation and development of skills for understanding and implementation of algorithms, skills for modeling and skills for analyzing algorithms. Criteria and indicators for the diagnostics of the experiment results were developed. Didactic tests were created and probed. The pedagogical experiment was conducted with underground majors in Informatics in the Faculty of Mathematics and Informatics of Plovdiv University, Bulgaria. The following conclusion was derived from the results of the experiment — programming is a specific type of human activity whose successful implementation requires not only practical application of knowledge and skills but also a specific type of thinking.

Key words: computer science education, programming, algorithmic thinking, developmental training

1. Introduction

The dynamic development of information technologies has led to the creation and modification overtime of a significant number of programming languages with different areas of application and use.

The study of programming languages is traditionally associated with the specification of its syntax and semantics, illustrated with examples. Usually in training we use a particular version of language and programming tools.

The practical orientation of the training results in rapid absorption of skills is valid only in a specific context. The lack of a general approach to learning programming languages makes the process of understanding in learning more difficult, which prevents a wider use of already adopted languages — for example, in developing skills for implementing a new programming language or a new version of an already studied one. On the other hand, practical examples confirm the fact that after mastering two or three programming languages comprehension and understanding of a new programming language is easier. According to researchers, as a result

Sava Grozdev, Ph.D. in Mathematics and DSc in Pedagogical Sciences, Professor, Institute of Mathematics and Informatics, Bulgarian Academy of Sciences; research areas/interests: education, didactics, mathematical analysis, operational calculus, synergetic. E-mail: sava.grozdev@gmail.com.

Todorka Terzieva, Ph.D. in Pedagogy of Education in Informatics and Information Technology, Head Assistant Professor, Department of Software Technologies, Faculty of Mathematics and Informatics, Plovdiv University Paisii Hilendarski; research areas/interests: teaching methodology informatics and information technology, algorithms and data structures, programming languages, distance learning. E-mail: terzieva.dora@gmail.com.

of experience and skills, one creates a common pattern in his or her mind, which in cognitive science is referred to as conceptual model, thus facilitating the acquisition of new programming languages.

2. Algorithmic Thinking — A Key Competence in Computer Science

In the transition towards information society, within the conditions of constant interaction with computer systems, the algorithmic style of thinking is a necessary basis for the performance of every modern personality. The problem solving is inherent to every scientific field and academic discipline. Moreover, each scientific field is defined by the specifics of the problems it addresses, as well as by the methodology it uses for the solving process itself.

As a result of a conducted research (Grozdev, 2011), observation and study of the scientific literature on methodology (Cormen, 1990; Gazeykina, 2004; Milne, 2002; Kolczyk, 2008; Snyder, 2006; Wing, 2006; Vorontsova, 2010) and cognitive psychologies (Piaget, 1983), we came to the following conclusions: programming is a specific type of human activity and its successful realization requires not only practical application of knowledge and skills but also a specific type of thinking. On the other hand the new and rapidly changing content of informatics teaching requires the development of methods which can ensure not only reproduction of a great amount of knowledge but most of all forming and developing of student competences, which allow them to master the knowledge actively, and also building of skills for independent acquisition of new knowledge and its critical rationalization.

The development of thinking in the learning process means forming and perfecting of all types, forms and operations of thinking, development of skills and habits of applying the laws of thought in the cognitive and learning activities, as well as habits to transfer the intellectual activity methods from one area of knowledge to another (Andreev, 1996). Most generally, the schematic and the intellectual development of the student may be described and understood through the categories of knowledge — thinking — ability and motivation of the mental self-development. Knowledge is a necessary condition for correct and sufficient thinking processes — comparison, analysis and synthesis, generalization and concretization. The correct management of these processes contributes to perfecting and enrichment of the knowledge. Therefore, thinking may develop when there exists a certain amount of acquired knowledge.

We can define algorithmic thinking as a *way of thinking, which provides a solution for a specific task through a succession of elementary actions* (Grozdev, 2011). Algorithmic thinking consists of a wide range of abilities and is affected by many other cognitive factors. The initial course on informatics must introduce students to the technology of design, developing and application of computer programs, creating habits, which may be applied and developed while studying other informatics disciplines. At the same time, the introductory courses must present the basic intellectual aspects of the computer science to students. The algorithmic thinking components are: *analyzing* — determining the initial condition, target, hypothesis and limitations; *decomposition* — dividing the problem to sub-problems and determining the basic solution operations; *formalization in order to create a model* — reformulating the problem with computer science terms, creating an algorithm and defining the relation between the subtasks; *comprehension* and *applying formal ways* for recording the algorithms; execution of a certain algorithm through formal and precise execution of the main activities; *algorithm analysis* in order to determine the optimal solution; *modification* of already familiar algorithms for their *application in new conditions*; *creation of a new* (unknown) *algorithm*.

An important component of the level of algorithmic thinking is the **ability to create an algorithm** or the application (use) of the algorithm in a **new situation**.

Algorithmic thinking has general and specific properties compared to other styles of thinking. Among the general properties of the algorithmic thinking are integrity and performance, helping to see to a problem comprehensively and involves the preliminary mental image of the decision. The specific properties relate to discretion, abstraction, generalization, consisting of solving all the problems of a class skill for formalization - dividing a difficult and complex problem into sub problems. These properties suggest a step process algorithm, enable abstracting from specific input, and transition to a decision of a task in general form and presentation of the algorithm using a formal language.

Algorithmic thinking consists of a wide range of abilities and is influenced by many other cognitive factors. The initial course on computer science education must introduce the students to the technology of design, developing and application of a computer program, must create habits which may be applied and developed while learning other informatics-related disciplines. At the same time, the introductory courses must present the students to the basic intellectual aspects of the computer science.

The fundamental elements of the model of the learning process are the actions which occur in the course. In this sense, we share the view that a lot in training does not follow the complete volume of knowledge and training materials that are available, but rather the activities of students in solving problems and formalization in order to create a model — reformulating the problem with computer science terms, creating an algorithm and defining the relation between the subtasks; comprehension and applying formal ways for recording the algorithms; execution of a certain algorithm through formal and precise execution of the main activities; algorithm analysis in order to determine the optimal solution; modification of the known algorithms for their application in new situations; creation of a new (unknown) algorithm (Terzieva, 2011).

3. Structuring of Learning Objectives in Computer Science

The task of constructing a scheme for structuring the educational objectives was undertaken for the first time in the USA. In 1956 Benjamin Bloom published his taxonomy of the educational objectives for cognitive activities, which proved to be extremely valuable for the diagnostics of the results from educational work (Bloom, 1956). This theory bears the idea that the objectives and the outcomes of education are not the same. For example, the memorizing of the scientific facts, regardless of their importance, is at a lower level than the skills for their analyzing and evaluation. Bloom offers six levels: knowledge, comprehension, application, analysis, synthesis, evaluation. Many cognitive psychologists work on the development of more precise and adequate taxonomy for the basic cognitive conceptions and level of thinking.

The educational taxonomies, especially Bloom's, for cognitive activity has a significant effect on the development of instruction design in the last 60 years. Their application and use, however, creates a number of difficulties. The classification of the learning outcomes and the tests outcomes depends on their context. A task, which makes difficult the application of analysis and synthesis by a beginner in the field of educations, becomes routine in the application of knowledge by more advanced trainees (Fuller, 2007). In the same way, a student, who is trained how to solve problems, which are extremely similar to the given tests, will demonstrate skills, which are at a lower level in the hierarchical taxonomy, than those demonstrated by a student, who has been solving problems based on principles.

One of the hallmarks of psychological and educational theory and research on learning since the original taxonomy was published so far emphasize on helping students become more knowledgeable of and responsible for their own cognition and thinking. This change cuts across all the different theoretical approaches to learning and development-from Piaget models to cognitive science and information processing models, to Vygotsky and cultural or situated learning models. Regardless of their theoretical perspective, researchers agree that with development students become more aware of their own thinking as well as more knowledgeable about cognition in general (Pintrich, 2002).

These problems are common to all fields of education, but a number educators (Mikova, 2006; Snyder, 2006; Rahnev, 2010) note that there also appear specific difficulties in the teaching of computer sciences, They have established that the classical taxonomy is not suitable for evaluation of practical skills or for determining the relevant difficulty of the cognitive tasks in the field of computer sciences. A significant number of researchers believe that it is easier to apply the knowledge for solving simple problems than to describe this knowledge (Grozdev, 2007). Moreover, they have established that computer science lecturers do not find the terms "synthesis" and "evaluation" to be the most important in describing learning outcomes and in evaluation of the tasks in programming courses, especially at the basic level of education. Instead, they see the application of knowledge as the highest skill, which the trainees should develop.

In Bulgarian education, a special place is occupied by Bloom's taxonomy associated to the cognitive domain, which becomes the basis of the establishment of standards in the curriculum.

In 2001 Anderson and Krathwohl (Anderson, 2001) specify and develop the taxonomy suggested by Bloom, emphasizing more on the creative paradigm, in which the intellectual development is studied as a change of the thinking pattern of the trainees. The major differences lie in the more useful and comprehensive additions of how the taxonomy intersects and acts upon different types and levels of knowledge — factual, conceptual, procedural and metacognitive. This melding can be charted to see how one is teaching at both knowledge and cognitive process levels.

3.1 Levels of Knowledge in Bloom's Taxonomy Revised

The first three of these levels were identified in the original work, but rarely discussed or introduced when initially discussing uses for the taxonomy. **Metacognition** was added in the revised version.

- **Factual Knowledge** — knowledge that is basic to specific disciplines. This dimension refers to essential facts, terminology, details or elements students must know or be familiar with in order to understand a discipline or solve a problem in it.
- **Conceptual Knowledge** — knowledge of classifications, principles, generalizations, theories, models, or structures pertinent to a particular disciplinary area. The interrelationships among the basic elements within a larger structure that enable them to function together.
- **Procedural Knowledge** — refers to information or knowledge that helps students to do something specific to a discipline, subject, or area of study. It also refers to methods of inquiry and criteria for using skills, algorithms, techniques, methods and particular methodologies.
- **Metacognitive Knowledge** — knowledge of cognition in general, as well as awareness and knowledge of one's own cognition. It is strategic or reflective knowledge about how to go about solving problems, cognitive tasks, to include contextual and conditional knowledge and knowledge of self (Anderson, 2001).

The new taxonomy makes distinction between knowledge on *what* "contains the cognitive activity" and knowledge on how, i.e., the procedures used for solving problems. The skill to combine elements in order to obtain something new suggests creative activity for the creation of new schemes and structures. In the words of one of the creators of the extended taxonomy, "You may be able to think critically — to support your position, to draw conclusions etc., without having creative skills, but creative activity — to prove or reject ideas, to create new ideas, often requires critical thinking" (Halpern, 1996).

4. Two-dimensional Framework of Study Goals in Extended Bloom's Taxonomy

Although the taxonomy of Anderson and Krathwohl is not the only possible way to classify the levels of thinking, it has a clear structure, facilitates the organization process of the intellectual development education, starting with the initial stage of mastering techniques for thinking activity, transition towards intellectual operations at a higher level and adopting habits for highly organized thinking. The cognitive objectives of the extended taxonomy have universal nature and could be applied in programming teaching.

The schematic and the intellectual development of the student may be described and understood through the categories of knowledge — thinking — ability and motivation of the mental self-development (Krathwohl, 2002). The volume of the knowledge defines the horizon, the parameters, and the limits, on which the thoughts and the fantasies of man spread. The knowledge is a necessary condition for correct and sufficient thinking processes — comparison, analysis and synthesis, generalization and concretization. The correct management of these processes contributes for perfecting and enrichment of the knowledge. Therefore, the thinking may develop when there is a certain amount of acquired knowledge.

The revision of the original Bloom's taxonomy is a two-dimensional framework: Knowledge and Cognitive Processes. Therefore, every type of educational activities for the formation and development of algorithm thinking can be introduced by using the two-dimensional framework of the extended Bloom's taxonomy represented as a combination of the type of knowledge and corresponding level of cognitive process (Table 1).

Table 1 Learning Activities for The Development of Algorithmic Thinking

Type of know ledge	LEVELS OF COGNITIVE PROCESS					
	Remembering	**Under-standing**	**Applying**	**Analysis**	**Evaluation**	**Creation**
Factual Knowledge	• lists fundamental concepts • recognizes simple primitive data structures • knows the basic operations	• gives examples of the basic concepts • under-stands the values of program objects • explains the relationship between the type of concepts	• implements simple data structures • applies examples of basic concepts • reads values and results of basic operations • understands the action of basic operations	• discovers similarities and differences between concepts • compares two objects (standard data types or elementary operations) • discovers similarities and differences between standard data structures	• indicates which data types of are used most often • states which are the most important concepts and why • locates syntax errors	• locates criteria for the application of standard structures • corrects syntax errors • defines a set of input data for the decision of the task

(Table 1 to be continued)

(Table 1 continued)

Conceptual Knowledge	• lists simple and abstract data structures • specifies algorithm properties • recognizes abstract data structures	• reads the results of different actions • classifies concepts • indicates the implications of facts and limitations • groups objects • under-stands basic concepts	• understands the results of different actions • uses standard control structures • gives examples of concepts and structures • discovers properties of algorithms • reads and explains fragments of a simple program	• compares types of data structures • explains differences, lists results of various actions • compares alternative representation of data structures • groups the main objects of a problem (algorithm) • identifies the type of prototype problem	• compares advantages and disadvantages of static and dynamic implementation of data structures • determines basic actions to solve a problem • identifies similarities with similar problems	• formulates objective, hypotheses and boundary conditions for the solution of the problem • selects the right data structure for decision tasks • implements abstract data structures
Procedural Knowledge	• describes basic operations • lists the types of control structures • reads programming objects • describes the standard algorithms	• under-stands basic operations and values of program objects • under-stands results using control structures • separates major and minor elements (facts) for the solution of a problem • deter-mines restrictions for input and or output data	• implements operations: arithmetic, logic, etc. • explains the values of program objects • implements appropriate management structures • changes the input data (after an error) • structures information • plans the strategy and analysts of the results • performs formal algorithms • checks the results through formal and proper execution of actions	• divides concept (operation) of the basic components • compares alternative representations of data structures from the viewpoint of performance. • Categorizes problem. • Compares different solutions to a problem. • Analyzes the results obtained. • Matches the solution model (algorithm) with a problem (task). • Provides a decision and a number of separate steps. • Analyzes the correctness of the algorithm	• tests and evaluates errors in an algorithm • understands and explains the cases in which a suitable algorithm can be applied (iteration, recursion) • determines the sequence of execution of elementary actions to solve a problem • critiques ready programs • evaluates the effectiveness of the algorithm (time and memory usage) • evaluates alternative (different) solutions to a problem • specifies properties inherent to the best algorithm • formulates difficulties in the implement-ratio n of the decision • working group (team), assesses (accepts or rejects) the ideas of team members	• creates standard algorithms • modifies the algorithm • sets the algorithm • within the system of elementary actions, builds an algorithm • implements programs using abstract data structures • adapts standard algorithms to a real task. • recognizes and extracts the prototype of similar problems in different contexts problem • reformulates the problem in terms of IT

(Table 1 to be continued)

(Table 1 continued)

Metacognitive Knowledge	• recognizes the main object of a real problem. • indicates the type of concept • recognizes program objects • knows the standard algorithms • identifies strategies for retaining information • detects the type of problem • describes strategies • specifies alternative concepts and objects • understands the algorithm • uses the result of an action for proof or rejection of hypothesis	• detects the type of problem • describes strategies • specifies alternative concepts and objects • understands the algorithm. • uses the result of an action for proof or rejection of hypothesis	• explains the results of different actions • determines contraction algorithm • describes the typical applications of data structures • describes and applies strategies for debugging • explains the algorithm action	• describes the essential and secondary connections between concepts • finds the relation between subtasks • classifies and analyze information • structures the facts • justifies the choice of action to address the problem • designs effective tests for the analysis of algorithms	• detects errors (semantic, logical) in the algorithm • tests the results and adjusts the input, if there is a discrepancy of these results with expected results • assesses in which cases a method or an algorithm is suitable • establishes cause and effect relationships • formulates a hypothesis by accepting or rejecting the opinion (in the team) • formulates conclusions • develops some criteria for evaluations of the project	• defines a problem that is new (unknown) or applicable in a new situation • formulates a new problem in terms of informatics • formulates alternative hypotheses • offers improved project decision based on defined criteria • develops a research plan of the (new) unknown problem • implements the idea for the research of a new problem • formulates difficulties in the solution of an unknown problem

5. Criteria for Diagnostics of Learning Results

One of the major problems of both the theory and practice of the didactic testing is the determination of the objectives and the tasks of the educational work, the achievement of which is diagnosed by tests (Bizhkov, 2007). The defining of the objectives is an important stage of the overall planning, conducting and evaluation of the education.

According to the definition adopted by the European Qualification Framework (EQF), the learning outcome is defined as an indicator of what the trainee knows, understands and is able to do in completing the learning process. Therefore, the emphasis is on the learning results, which are specified in three categories — knowledge, skills and competence. Within the context of EQF, competence means a proved ability to use knowledge, skills and personal, social and/or methodological abilities in the work to study situations and to achieve professional and personal development. The term competence is broader and typically refers to the ability of a person to face new situations successfully using and applying knowledge and skills in an independent and self-directed way (ESCO, 2013).

The initial teaching of informatics and information technologies must form not only the basic concepts, skills and habits to work with computer, but also to provide development of certain style of thinking.

For the obtaining of objective information regarding the accessibility of the suggested educational content and the efficiency of the developed educational methodology aimed at the development of algorithmic thinking, criteria and indicators for the evaluation of the learning outcomes are necessary. The traditional structure of conducting pedagogical experiments includes three stages: a preliminary experimentation, a forming experiment and a concluding experiment. The objective is to follow the development of the results by applying the elaborated methodology. Since the suggested methodology includes the content of a course on “Basics of the Computer

Science" and "Programming", it is very difficult to devise criteria and indicators for preliminary evaluation of the trainees which could be used in both experiments — the preliminary and the control experiment. The reason is the fact that in the last two stages of the experiment, concepts and algorithms are observed, which cannot be known to the trainees previously and the degree of their mastering cannot be followed at the preliminary stage. That is why most of the indicators used for the evaluation of the outcomes are with changeable formulation for the preliminary and the concluding experiment (Terzieva, 2011). For the operationalization of the objectives, the extended Bloom's taxonomy is used.

We made a survey of the opinion of teachers in Computer Science at Plovdiv University Paisii Hilendarski, Bulgaria, regarding the degree of significance of the named skills and objectives for the basic training of the students in the Informatics Bachelor program. The results obtained showed that most important were considered the skills for problem analysis and algorithm analysis, followed in significance by the skills for formalization, abstracting from the specific input data and proceeding to the solution of the task in general aspect, as well as the use of a general algorithm for solving a specific problem. The lecturers considered the creation of a new (unknown for the students) algorithm difficult and less significant activity in the teaching of computer science and accentuate on the analyzing and formalizing skills (Terzieva, 2011).

We used the following criteria to evaluate the results:

Criterion I: Knowledge and skills related to problem solving.

1) Ability to analyze, define problems and identify appropriate data types.
2) Ability to decompose a problem into subtasks which can be differentiated into subroutines.
3) Ability to define and to use abstract data structures.
4) Ability to implement basic algorithms in abstract data structures.

Criterion II: Knowledge and skills related to understanding and implementing the algorithm.

1) Understanding and monitoring the implementation of a program.
2) Understanding and modifying the algorithm of the context.
3) Ability to define an appropriate data structure and algorithmic performance.
4) Ability to test and to adjust a program and to correct the errors in the algorithm.

Criterion III: Knowledge and skills related to the analysis of algorithms.

1) Analysis of the correctness of the algorithm.
2) Evaluation of the effectiveness of the algorithm.
3) Comparison and analysis of different solutions of a problem.
4) Ability to experiment, analyze the obtained results and correct the input data if necessary.

The main questions which must be answered are related to whether the objectives are achieved, to the efficiency of the learning work, how well is developed the educational environment and the technology of teaching, etc.

6. Analysis of the Test Results

Tools used to assess the results obtained by training include didactic tests, tasks and assignments that require writing a complete program.

Used didactic means: **Tests** — the main experimental diagnostic tool. Tests are conducted at each of the stages. Specially created **control exercises** — include creating a model of a subject area, a description of the

stages of the development of an algorithm written in code with a given specification, detection and correction of syntactic and semantic errors in a program task to find the optimal solution with the used algorithm and data structure, etc. **Tasks**, which require writing a comprehensive program, suitable for evaluation of the applications and practical skills of the students.

The suggested criteria for evaluation of the algorithmic thinking formation are approbated during the lectures with first-year students in the Informatics major in the Faculty of Mathematics and Informatics at the Plovdiv University for a period of three years.

The traditional structure of conducting pedagogical experiments includes three stages (Bizhkov, 2007): preliminary (notes) experiment, procedure (formative) and final test. The aim of the preliminary experiment was found to establish a baseline of the object of study. The formative experiment was proposed after 4-5 weeks of the process of the experimental learning. The final test was conducted at the end of the period. The aim was to trace the development of the results of applying the elaborated methodology.

At each of the three stages of the experiment, students completed a test of 12 questions and solved an additional task in C++. The relevant questions significantly differed in numbers and results between different measurements could not be compared. The results obtained by the students for each of the questions were read personally, but for the sake of clarity, the data were presented in the form of summary — the total number of points obtained after collection of individual item test results of each student in both groups (EG and KG) of students in the Informatics Bachelor program.

The teaching methodology used in the experimental group (EG) achieved significant results. The main indicator for this was the statistical significance of the interaction effect between the factors of the measurement stage and the belonging to a control or experimental group (Terzieva, 2012).

The classical analysis of the reliability was found by calculating the Cronbach's alpha coefficient with maximum value 1, and for each of the item individual indices of discrimination with a maximum value 1. (Bizhkov, 2007). To compare the results of test 3, we put into practice the method of Kolmogorov-Smirnov (Figure 1) and also examined the mathematical expectation, applying T-statistics (t-Test: Two-Sample Assuming Equal Variances).

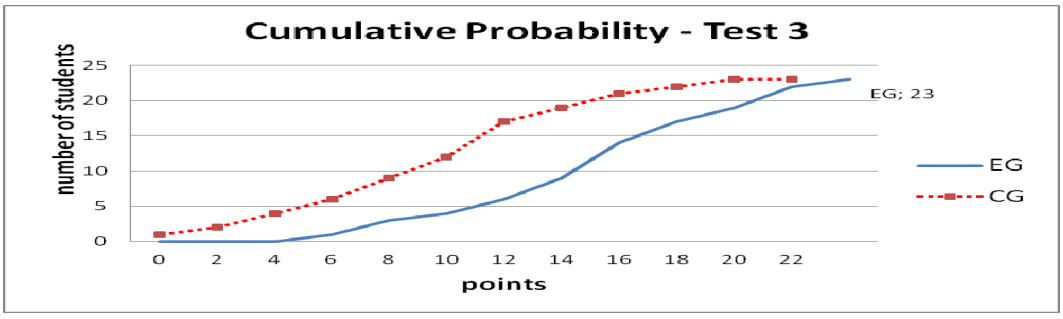


Figure 1 Results of Kolmogorov-Smirnov Test

The classical analysis of the reliability is found by calculating the Cronbach's alpha coefficient with maximum value 1, and for each of the item individual indices of discrimination with a maximum value 1. To compare the results of the final test (test 3) (Figure 1) the method of Kolmogorov-Smirnov is put into practice and the mathematical expectation is also examine, applying T-statistics (t-Test: Two-Sample Assuming Equal Variances) (Figure 2).

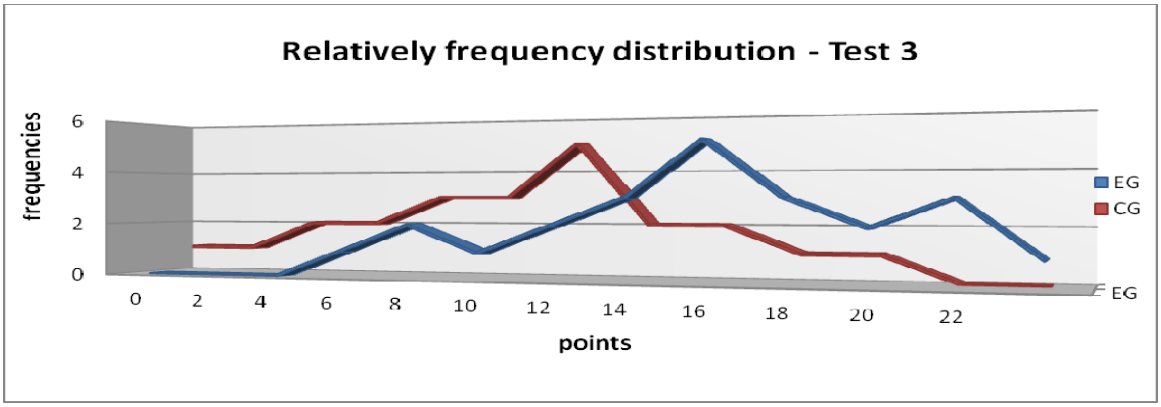
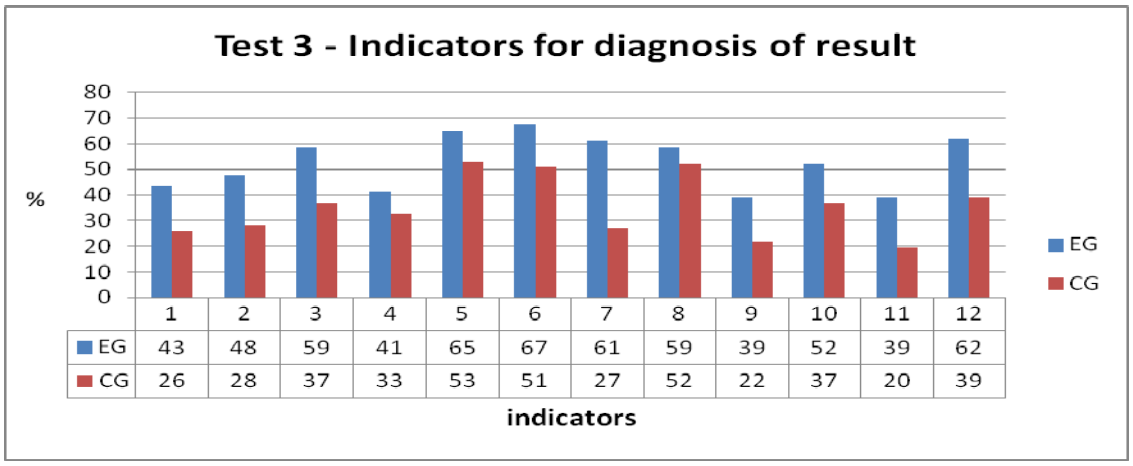


Figure 2 Results of T-statistics

From Figure 1 shows that the cumulative frequencies were substantially different, as the control group (CG) is located entirely above the line of the EG, and in some parts, the distance between them is considerable. If we examine the histograms of the two groups, we will notice the difference in the distribution of scores after the training (Figure 2). Students in the CG showed significantly lower results compared to the experimental group.

The survey data on the indicators (Figure 3) shows that the most significant difference in terms of the results of the seventh indicator is the ability to define an appropriate data structure, as well as the ability to analyze a problem, the skill to divide the problem into subtasks and the ability to compare and analyze different solutions. The ability to understand and implement the algorithm in the training process was involved significantly. Most significant was the difference in terms of the ability for modeling.



	1	2	3	4	5	6	7	8	9	10	11	12
EG	43	48	59	41	65	67	61	59	39	52	39	62
CG	26	28	37	33	53	51	27	52	22	37	20	39

Figure 3 Comparative Results on Indicators from Final Test

7. Conclusion

The main educational activities related to the formation of skills for problem analysis, algorithm comprehension and execution, as well as algorithm analysis were at a higher cognitive level. They were exclusively procedural and of metacognitive type of knowledge. The levels of the cognitive process were also of a higher level — analysis, synthesis, evaluation. Therefore, special efforts are needed to form and improve these skills.

The initial programming courses introduce to students the technology of design, developing and implementation of a computer program. At the same time, we must encourage the development of skills necessary for the application of conceptual knowledge to create habits that can be applied for studying and developing the next disciplines in computer science. Thus, indeed, programming is a specific type of human activity and its successful implementation requires not only practical application of the acquired knowledge and skills, but also a specific type of thinking.

References

Anderson L. and Krathwohl D. R. (2001). *A Taxonomy for Learning, Teaching, and Assessing*, New York: Longman.

Andreev M. (1996). *The Process of Learning*, Didactics, University Publishing House "St. Kliment Ohridski", Sofia. (in Bulgarian)

Bizhkov G. and Kraevski S. (2007). *Methodology and Methods of Pedagogical Research*, Publishing House "St. University Ohridski", Sofia. (in Bulgarian)

Bloom B. (1956). *Taxonomy of Educational Objectives: The Classification of Educational Goals: Handbook I, Cognitive Domain*, New York: Longmans.

Brown W. (2002). *Introduction to Algorithms* (2nd ed.), MIT Press, pp. 61–66.

Cormen T., Leiserson C., Rivest R. and Stein C. (1990). *Introduction to Algorithms*, Cambridge, MA: MIT Press, p. 984.

Fuller U., Johnson C. G., Ahoniemi T., Cukierman D., Hern I. and Hernn-Losada I. (2007). *Developing A Computer Science-Specific Learning Taxonomy*, *ACM SIGCSE Bulletin*, Vol. 39, No. 4, pp. 152–170.

Gazeykina A. (2004). *Styles of Thinking in Teaching Programming to Students of Pedagogical Universities*, Ekaterinburg. (in Russian)

Grozdev S. (2007). *For High Achievements in Mathematics: The Bulgarian Experience — Theory and Practice*, ADE, Sofia.

Grozdev S. and Terzieva T. (2011). "Research of the concept of algorithmic thinking in teaching computer science", *The International Scientific-Practical Conference "Informatization of Education – 2011"*, Elec: EGU Bunin, 14–15 June, pp. T1, 112–119. (in Russian)

Halpern D. (1996). *Thought and Knowledge: An Introduction to Critical Thinking*, Hillsdale, New Jersy, Mahwah.

Kolczyk E. (2008). *Algorithm — Fundamental Concept in Preparing Informatics Teachers*, Springer Berlin/Heidelberg, Volume 5090, pp. 265–271.

Knuth D. (1985). "Algorithmic thinking and mathematical thinking", *The American Mathematical Monthly*.

Krathwohl D. (2002). "A revision of bloom's taxonomy: An overview", *Theory into Practice*, Vol. 41, No. 4, pp. 212–218.

Mikova E. (2005). "Developing of algorithmic thinking: The base of programming", *International Journal of Continuing Engineering Education and Life Long Learning*, Vol. 15, Number 3–6, pp. 135–147.

Piaget J. (1983). "Piaget's theory", in: P. Mussen (Ed.), *Handbook of Child Psychology* (4th ed.), Vol. 1, New York: Wiley.

Pintrich P. (2002). "The role of Metacognitive knowledge in learning, teaching, and assessing: Theory into practice", Copyright College of Education, the Ohio State University, Vol. 41, No. 4, pp. 219–225.

Rahnev A. (2010). "Intensification of teaching programming using information technology", *Habilitation Thesis for the Award of the Academic Title "Professor"*, Sofia. (In Bulgarian)

Schwank I. (1993). "On the analysis of cognitive structures in algorithmic thinking: Processing language in introduction to computer science honors", *Journal of Mathematical Behavior*, Vol. 12, pp. 209–231.

Snyder L. (2006). *Algorithmic Thinking: The Key for Understanding Computer Science*, Springer-Verlag Berlin Heidelberg, pp. 159-168.

Terzieva T. (2011). "Some criteria and indicators for diagnosis of the forming of algorithmic thinking in computer science", Scientific Works, Plovdiv University, Vol. 38, Book 3, Mathematics, Plovdiv, Bulgaria.

Terzieva T. (2012). "Experimental study on forming of knowledge and skills in programming", *Anniversary National Scientific International Conference "Traditions, Directions, Challenges"*, October 19–21, 2012, Smolyan, Bulgaria, pp. 183–189.

Vorontsova and Rusakov (2010). "On the formation of the notion of 'Algorithm' in the course of Informatics on high school", *Educational Informatics*, Vol. 3. (In Russian)

Wing J. (2006). "Computational thinking", *Communications of the ACM*, Vol. 49, No. 3, pp. 33–35.

ESCO (2013). "European skills, competences, qualifications and occupations", available online at: https://ec.europa.eu/esco/web/guest/escopedia/-/escopedia/Competence.