STANDARDIZATION OF THE DEVELOPMENT OF EDUCATIONAL DIALOGUE COMPUTER PROGRAMS

Philip Petrov Petrov[*], Alexander Doychinov Popov[**] & Ridvan Mustafov Itsufov [***]

[*] Sofia, jk. Lozenetz,Tzanko Tzerkovski №1 srt, fl.3, ap.6

[**] Sofia, Veliko Tarnovo str № 9

[***] Sofia, jk. Studentski grad, bl. 57, ent. B, fl. 5, ap. 506

[*]philip@abv.bg, [**] alexandur_popov@abv.bg, [***] ridvan@fmi.uni-sofia.bg

ABSTRACT

The present announcement is an attempt to further improve the methodology for creation of computer educational programs the purpose of which is to develop the heuristic abilities of students. We describe two software products that lay the foundations for the creation of a universal framework, aimed at the creation of different development environments for computer dialog-educational programs.

Keywords: *scenario, basic help, additional help, a system for multiple variants of solutions (answers), error diagnosis, automatic scoring.*

The main purpose of the educational dialogue computer programs is to ensure, when possible, a better imitation of the individual tutoring of a professional teacher and a student. That is why the basis for the methodological principles of the use of computers in mathematical education is to use the achievements of modern pedagogy and of modern methodology for education without computers. Among the most important of these achievements are the didactic principles and especially the individual approach principle and the activity principle. The other guiding idea is the understanding that a student solves problems in the course of his or her mathematical education not simply because these problems must be solved but because he or she must learn how to solve them on his or her own or with assistance. That is why when problem solving is the purpose and not an auxiliary means in education, a computer mustn't be used as the tool to solve the problem for the student [2].

The logical scheme of the educational dialogue computer programs was created in a textual version and as a graph scheme in 1984 by Prof. Ivan Ganchev. Based on this scheme, the development of applied educational software was started using Basic for 8-bit computers. In 1986-87 Prof. Ivan Ganchev created the didactic

flowchart, which included an additional component for error diagnosis, shown on Figure 1. In this paper, we present a new version of the didactic flowchart, named "logical-didactic flowchart", shown in Figure 2 by the third author of the message.

Didactical flowchart
created in 1986-87 by Prof. Ivan Ganchev

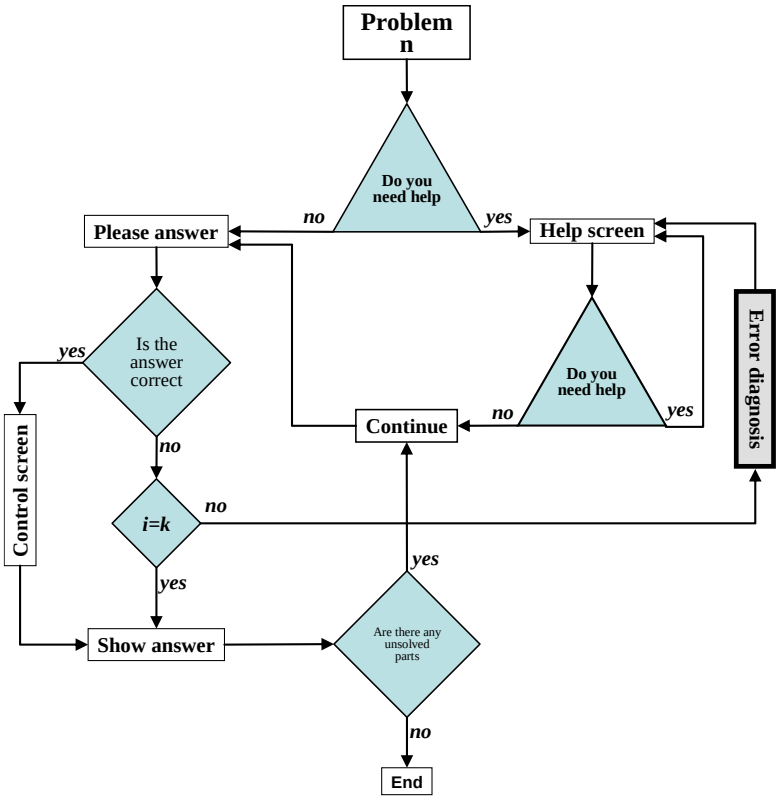


Fig. 1

Prior to 1999, every dialogue-educational program was implemented as a standalone applied educational software and each program required a separate set of code. At that moment, the third author of this announcement developed a solution which allowed to enter the contents and the links between the frames of the Microsoft Power Point software product. In 2003, the Prof. Ganchev's logical scheme of was redesigned into a newer, better structured version.

In 2008, the project continued its development and enhancement through the creation of a specialized "Internet-based environment for the development of educational dialogue computer programs". The product in question has been the topic of a graduate thesis for the Master's Degree at the Faculty of Mathematics and Infor-

matics of Sofia University St. Kliment Ohridski, Department of Mathematics and Informatics Education.

In 2009, the second author created “An environment for the development and use of lessons as exercise”. This environment introduced the terms “predictable” and “unpredictable” error. Another novelty was the creation of a unified file format, thanks to which educational dialogue computer programs became portable.

Logical-didactic flowchart of EDCP

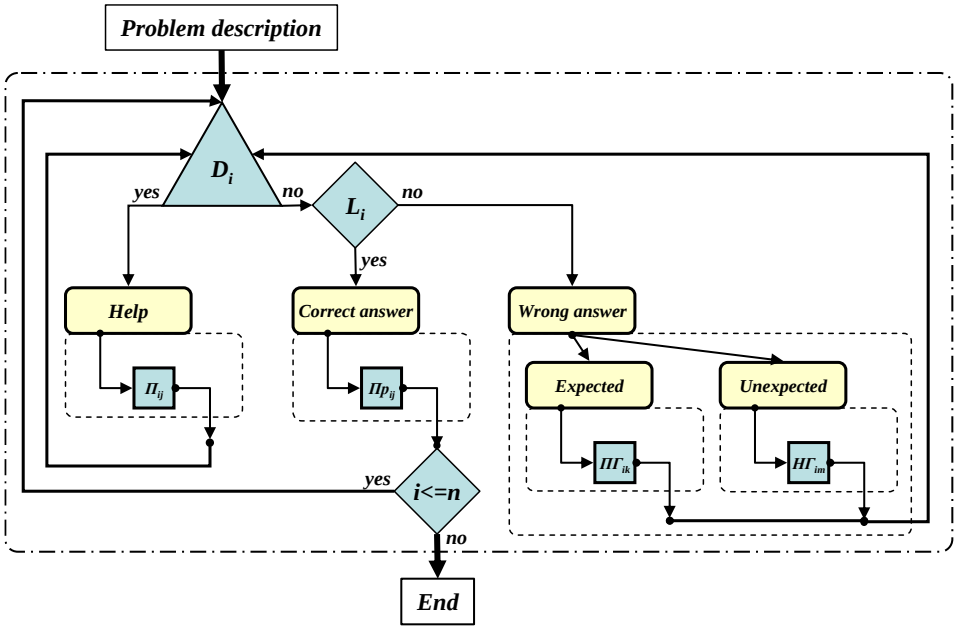


Fig. 2

LEGEND

△ – Didactical block, ◇ – Logical block, □ – Scenario;

▭ - Scenario (Help, Correct answer and Wrong answer);

⬚ - Actions, ⬚ - Activities;

n – amount of activities, i – activities i=1...n,

D_i – Didactical process (Help, Answer),

Help – ability to use help,

Answer – availability to provide answer.

L_i – Logical process (correct or wrong answer);

j, k , m – amounts of scenarios in the activity,

Scenario of activity:

$П_{ij}$ –**Help**, $Пр_{ij}$ – **Correct,**

$ПГ_{ik}$ – **Expected wrong answer**,

HT_{im} – **Unexpected wrong answer**,
⟶ - Progress to scenario;
⟶ - Progress to didactical block;
⟹ - Entrance and exit from activity.

The logical didactical flowchart of educational dialogue computer programs can be seen in Appendix 1. Each problem is divided into a given number of activities. The textual explanation of the implementation of an activity is as follows:

1. The student enters the didactic block of the "i" activity, where in an information frame he or she reads the condition of the problem and is given a field to input the answer. In case the student can't give an answer, he or she is given the opportunity to use help.
   - If the student enters an answer, he or she is taken to point 2.
   - If the student demands help, he or she is taken to point 3.

2. This step is a transition to a logical block, where the computer analyzes the answer passed by the student.
   - If the answer is correct, the next step is point 4.
   - If the answer is wrong, the error is analyzed,
     - If the error is predictable (i.e. it is a frequent error and the teacher has predicted it), the next step is point 2.1.
     - If the error is unpredictable, the next step is point 2.2.

2.1. Specific information about the particular error is shown in the information frame. The system increments the b_{ij} negative coefficient, which influences the final score. The student goes back to point 1., where he or she is given one more chance to solve the problem.

2.2 A standard message, notifying the user that he or she has made a mistake, is shown in the information frame. The system increments the b_{im} (m-1 being the number of predictable errors) negative coefficient, which influences the final score. The student goes back to point 1., where he or she is given one more chance to solve the problem.

3. The system counts the number of times help was previous requested (if help had already been requested) and the information frame shows the next in line. This is accumulated in the a_{ij} coefficient (j is the index of the occurrence of help request), which is negative for the final score. When the last help option (with index k) is reached, the solution of the problem is displayed and the student goes to point 4. Otherwise, the student goes back to point 1.

4. The end of the activity is reached (step "i" of the problem). A positive for the final score coefficient c_i is accumulated. If this is the last activity in the problem, the problem ends and the student see his or her score. If there is another activity, the student goes to it.

Logical-didactic flowchart
with elements of automatic rating

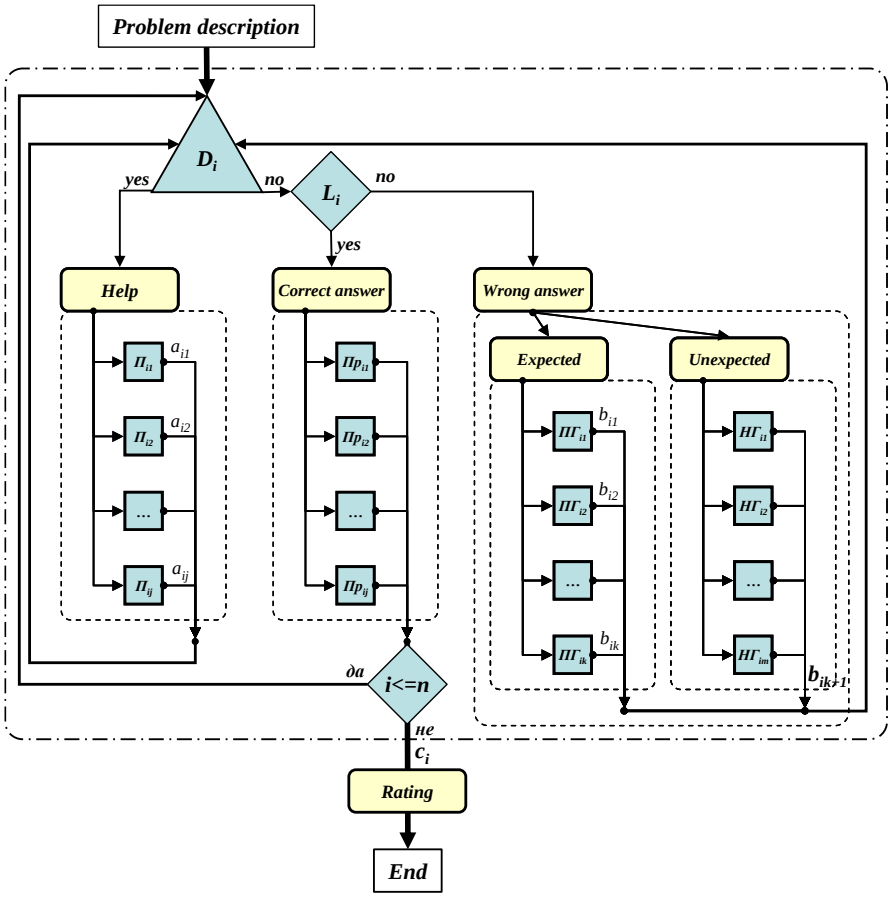


Fig. 3

LEGEND

– Scenario (rating);
n – amount of activities, ***i=1...n***;
i – index of the specific activity;
j – index for used help;
a_{ij} – coefficient for specific help;
b_{im} – coefficient for specific expected wrong answer;
b_{im+1} – coefficient for unexpected answer;
c_i – positive coefficient for specific activity;

The final score is calculated based on the following formula:

$$P = \sum_{i=1}^{n} p_i$$

$$p_i = c_i - h_i - w_i$$

$$h_i = \sum_{j=1}^{k} a_{ij}$$

$$w_i = \sum_{q=1}^{m+1} b_{iq} r_{iq}$$

P – the total score for the whole problem;
p_i – the score for a particular activity. If $p_i<0$, then we assume it is 0;
n – the number of available activities in the problem;
i – an index number for the consecutive activity in the problem;
c_i – a positive coefficient for a particular activity;
h_i – the accumulated negative coefficients for used help for a particular activity;
a_{ij} – a coefficient for used help;
j – an index for used help;
k – the total number of used helps for a particular activity;
w_i – the sum of accumulated errors for a particular activity;
b_{iq} – a coefficient for a particular predictable error. The last coefficient b_{m+1} is for an unpredictable error;
r_{iq} – an integer, which shows how many times a particular mistake has been made;
q – an index for a consecutive predictable or unpredictable error;
m – the number of available predicted wrong answers for a particular activity.

The dependency in this formula is as follows:

$$c_i = \sum_{j=1}^{k} a_{ij}$$

In other words, if the student uses all available helps for a given activity, he or she will get 0 scores for that. The author of the problem decides on the relative weights of the error coefficients.
Let's assume that Q is the maximum score for all the activities in the problem:

$$Q = \sum_{i=1}^{n} c_i$$

In order to calculate the final score O, normalized in the [2, 6] range, we use the following formula:

$$O = 2 + \frac{4P}{Q}$$

As you can see, the score criterion depends largely on the b_{ij} error coefficients, which are controlled by the teacher, who creates the problem.

In the Internet-based environment for the development of educational dialogue computer programs, created by the first author, the scheme used is similar to the basic scheme described here.

The main technical means for the implementation of the project are the PHP programming language, the MySQL database and the LateX module for mathematical formulas.

As far as the lessons for new topics and the lessons for exercises are concerned, the guiding principles are two – on one hand, the product shouldn't demand the installation of specialized software and still be accessible to teachers and students, who do not have special technical knowledge, and on the other hand, the product must give limitless freedom to the creator of the problem to insert dynamic objects and even to add programming logic to the product, provided that he or she has the necessary knowledge to do it.

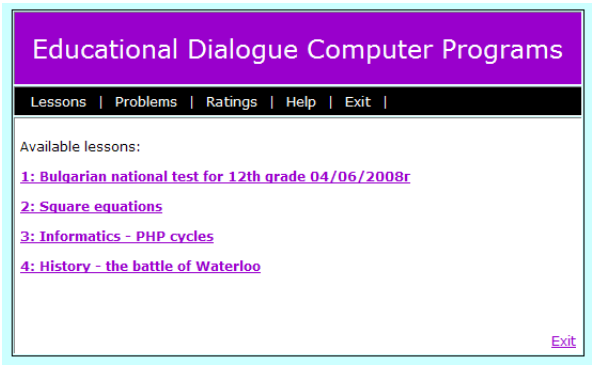
This product uses for the first time the **automatic student scoring** functionality. This automatic scoring is done depending on the help requested and the mistakes made. Such a program is useful for control tests and term tests in schools. Students can see their score right after the end of the test and their score will not be a subjective one. The probability that a problem is not solved to the end (for instance, the student has given up in the middle of the problem) is also included. In this case the student gets score but the program calculates that all unsolved activities equal 0 points and the problem is marked as "not completely solved".

The system can be easily translated into different languages. The language settings are contained in a file, where system texts are stored. A beta version of the program can be found at:

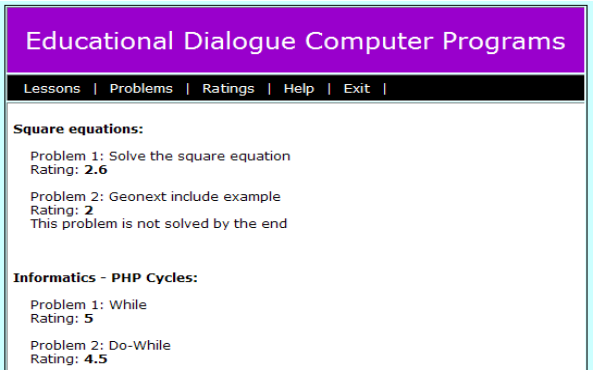
http://www.ekai.org/login.php

User (student) accounts can be registered as many times as needed. If you want to register an administrative (teacher) account, please write to philip@abv.bg.

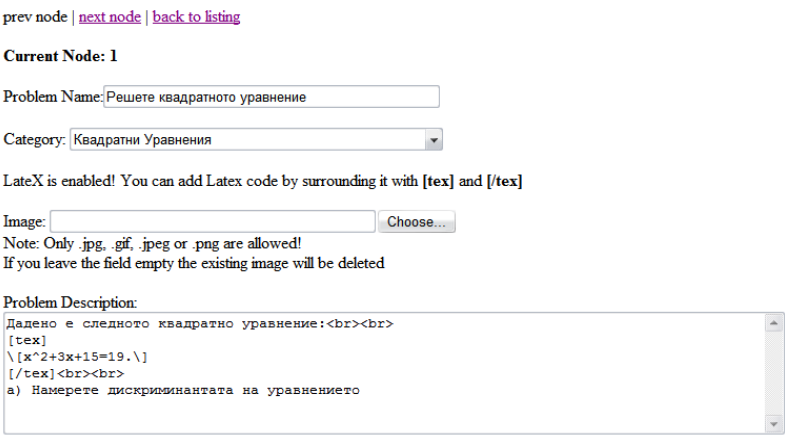
A part of the software can be seen on the following figures:



List of problems



Ratings over solved problems



Helps#: 4
Note: Number between 1 and 9. Incorrect entries will be converted to 1

Fill the text areas from "Help 1" to "Help Helps#"

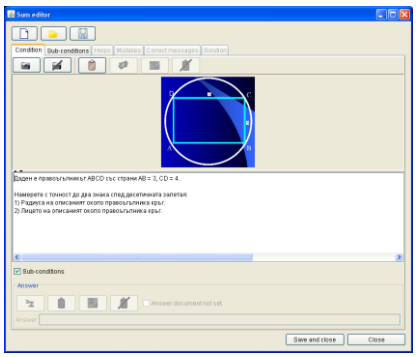
Help 1	Help 2	Help 3
Приведете уравнението в нормален вид: [tex] \[ax^2+bx+c=0\] [/tex]	Нормалния вид на уравнението е: [tex] \[x^2+3x-4=0\] [/tex]	Намерете дискриминантата по формулата: [tex] \[D=b^2-4ac\] [/tex]
Help 4	**Help 5**	**Help 6**
Дискриминантата на уравнението е: [tex] \[D=3^2-4(-4)=9+16=25\] [/tex]		
Help 7	**Help 8**	**Help 9**

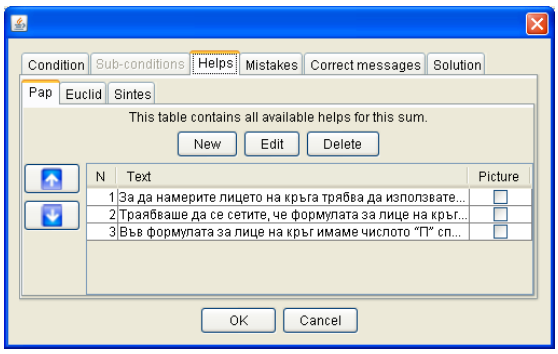
In the desktop environment for the development of educational dialogue computer programs created by the second author, the idea for the creation of a standardized file format was used for the first time. The standardized file format contains all the necessary information for the description of the dialog-educational program. The creation of such a format allows programs to be stored and ported. In addition to that, implementation of the same educational dialogue computer programs in different environments is also planned. The development and implementation of the file format is the guiding idea for the development of the environment. This is why it is divided into two parts. One part is for the teacher and thanks to it he or she can create scenarios for educational dialogue computer programs. The second part is for the student, who runs them. This approach allows parallel development.

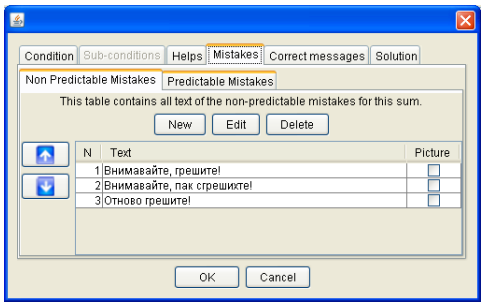
The technical means used for the development of the environment are the Java programming language and the XML format for data storage.

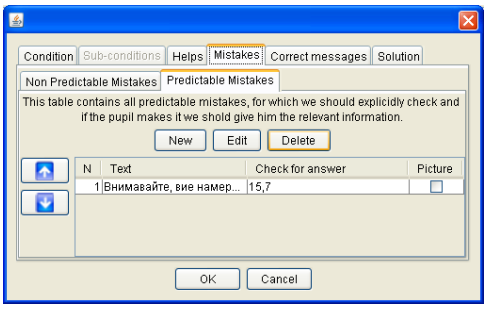
Another innovation in the presented idea is that when help is requested, different models for problem solving are used – the models of Pap, Euclid and Synthes. In this way the possible scenarios for solving problems are extended further.

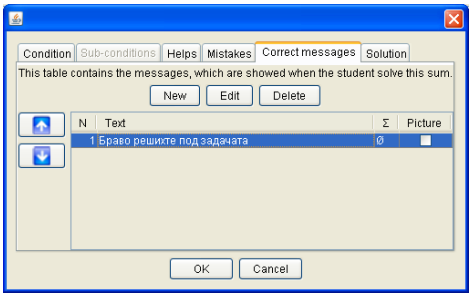
A part of the software can be seen on the following figures:

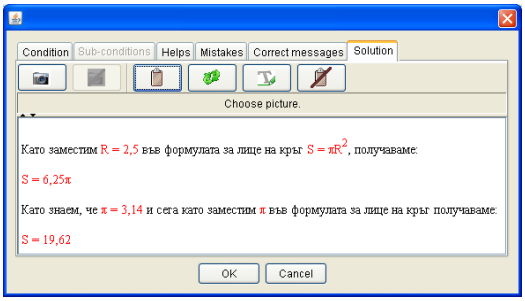


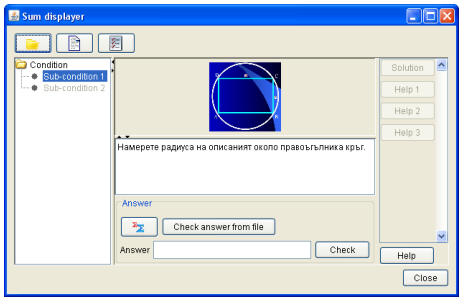


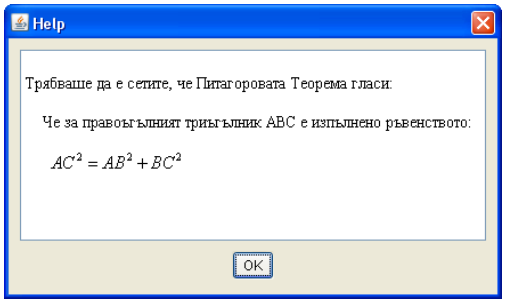


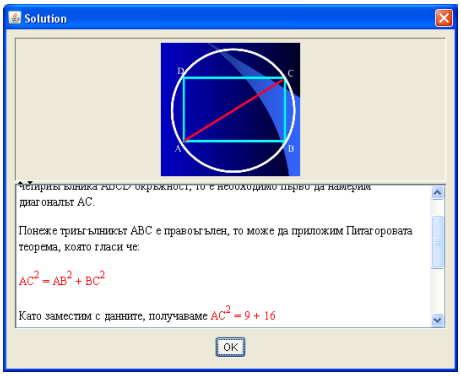












The described development environments and the announced scheme allow for the first time educational dialogue computer programs to be developed on the basis of unified standards, without requiring programming knowledge from the creator of the problems. The plans for future development of the projects include the creation of a framework – a set of libraries, which can be used for the creation of different environments for the development of educational dialogue computer programs. One of the priorities is the development of a system for the analysis of input data, sent by the student to the logical block.

REFERENCES

[1] **Vygotsky, L. S.,** - The Problem of Learning and Mental Development at School Age, in "Selected Psychological Investigations", Moscow, **1956;** (in Russian)

[2] **Ganchev, I., Kuchinov, Y.,** Dialogue Educational Software - overview, guiding ideas, principles, and stages of their development, Mathematics Education, **2, (1987), 1-8;** (in Bulgarian)

[3] **Ganchev, I., Kuchinov, Y.,** The Personal Computer as a Means of Enhancing Efficiency in Teaching Mathematics in the Secondary School, Mathematics Education, **3, (1987), 9-12.** (in Bulgarian)

[4] **Ganchev, I.,** Essential activities in a mathematics lesson, Modul-96, Sofia, 1999. (in Bulgarian)
[5] **Isufov, M. R**., On a Way of Rational Use of Computers in Teaching Mathematics , Proceedings, XXXIII Spring Conference, UMB, pp. 343-348, Borovets, 1-4 April 2004; (in Bulgarian)
[6] **Isufov, M. R**., Computer Dialog Educational Programs, thesis paper, "St.St. Cyril and Methodius"Veliko Tarnovo University, 1999; (in Bulgarian)
[7] **Isufov, M. R**., Educational Program on Formatting a Cell in a Spreadsheet**,** pp., 502-507, Scientific and Applied Science Conference "Challenges of Information Society Before Statistics and Mathematics - XXI century", Svishtov,
16-18 October 2003; (in Bulgarian)
[8] **Kostov, K.,** Computer Didactics, Blagoevgrad, 1991; (in Bulgarian)
[9] **Kuchinov, Y., Ganchev, I.,** Educating with Computers – needs, capacity, and State of Play, "Mathematics Education ", **4, (1986), 5-10;** (in Bulgarian)
[10] **Кучинов, Й., Ganchev, I.,** Educating with Computers – Issues, Trends and Prospects, "Mathematics Education", **4, (1987), 1-6;** (in Bulgarian)
[11] **Petrov, P. F.,** Development Environment for Dialog Educational Software, thesis paper, **"St. Kliment Ohridski" Sofia University - FMI, 2008.** (in Bulgarian)

РЕЗЮМЕ

В настоящото съобщение е направен опит за по-нататъшно усъвършенстване на начина за създаване на компютърни програми за обучение, осигуряващи развитие на евристични способности на учениците. Представени ви са два софтуерни продукта, които полагат основите за създаване на универсален framework, чрез който да се създават за различни среди разработка на компютърни диалогово-обучаващи програми.

Ключови думи: *сценарий, помощ, диагностика на грешка, действие, дейност, автоматично оценяване.*