Embedded Self Organizing Systems (Vol 9. No 4. 2022) (pp.48-51)

Embedded Selforganizing Systems

Special Issue Topic: “Learner Centered Learning“

Description Of The Task Generator For Testing Students' Knowledge

Alkhamov Radik
Faculty of Applied Mathematics and Informatics
Tashkent Branch of Moscow State University
Tashkent,Uzbekistan
radikrr60@gmail.com

***Abstract* - Automation of knowledge testing is one of the important tasks. There are special systems that allow you to test knowledge using a task generator. These systems use a special language similar to programming languages. An interpreter has been developed for the described language, which allows to translate the source text into an internal representation. This representation is transferred to the block in which the question is displayed, as well as the analysis of the student's answers. All tasks can contain random parameters, which makes the tasks unique. Each task can contain up to 16 types of answers. This article describes the system and language developed for knowledge testing. This system allows the student to display the algorithm for solving the problem step by step. The system also allows you to collect statistics of student responses.**

Keywords— automatic generation of tasks, knowledge check, task generator, random parameters.

I. Introduction

Nowadays, the use of computer technology in various fields is becoming crucially important. The best example is the education area. The use of computer technologies for testing student knowledge began with testing programs when the student must choose the correct answer from a given list of possible answers. However, this approach has several disadvantages. One of these disadvantages is the ability to "guess" the answers. The tool based on a task generator helps to reduce the impact of this disadvantage.

Theoretical bases for testing methods with random parameters are given in [1]-[4]. Description of such systems is given in [5]-[8].

DrFrost [9] and KutaSoftware [10] are the most used services. However, these services don`t allow third-party users to add new tasks. Similar systems have already been considered [11]-[14]. However, most of these systems do not have the ability to display the progress of the task

II. Methods for constructing a task generator

A task generator is a tool that allows the teacher to create a bank of tasks using random parameters.

A special language and interpreter have been developed for the task generator to make adding tasks to the system convenient for the teacher. The interpreter translates text from this language into an internal representation. Then the resulting program is transferred to the block that generates questions and the output block of the algorithm description for solving problems.

The proposed language has a block structure, i.e. all commands are written as blocks. Each instruction is written on a separate line. Each block starts with a line like "#<block name>" and ends with a line like "%<block name>". The following blocks are defined.

The “#K” block contains the values of the main constants, for example: DEL=':' UMN=CHR(183).

In program, for example: DEL=':' UMN=CHR(183).

The block “#V” contains the description of all variables. The following types of variables are allowed: INTEGER - integer variable in the range (from -65535 to 65536), REAL - real number in the interval (from 1.5 10^-18 to 3.2 10^18), STRING - string variable, up to 65535 characters, FRACTION – common fraction is a set of three integers, INTARRAY[N] array of integers, consisting of N elements, RLARRAY[N] array of real numbers, consisting of N elements, STRINGARRAY[N] array of string data, consisting of N elements.

Block "#C" contains the text of the program, which consists of instructions (operators).

The assignment operator has the form <variable>:=<arithmetic expression>. The type of the variable and the arithmetic expression must be the same.

The conditional operator looks like:

IF <conditional expression> <statements 1> ELSE <statements 2> ENDIF.

In addition, the user can use 2 types of loop operator.

The first loop operator looks like:

FOR <variable>:=<start value> TO <end value>

<operators>

ENDFOR

The second loop operator is DO, looks like: DO <statements> WHILE <conditional expression>

Block "#F" defines a variable containing the text of the task.

The "#P" block defines variables containing hints for the task.

The "#O" block defines the parameters of the task. This block consists of the type of answer and the value of the correct answer, which is defined as a variable.

The system supports the following answer types: one integer, one fraction, one real number, one character string, two integers in any order, two integers in any order, two fractions in any order, two fractions in this order, two real numbers in any order, two real numbers in the given order. Moreover, the number of answers can be increased to 4, and fractions can be presented both in reduced and non-reducible form.

The file “Tem.sqt” is required for the system to work, which contains the main parameters of the system, a description of the topics that the system uses, as well as the names of the files in which the description of the tasks is located.

The system works according to the following algorithm:

The student selects the topic and the type of problem. According to the type of the selected topic, a file containing a description of tasks on a given topic is read, and then the interpreter translates the task from a text form to an internal representation, while selecting random parameters and substituting them into the task conditions.

The generated task is transferred to the main testing block, which displays the task, and requests the student's answer, by the type of answer. Depending on the parameters of the task, the student may choose the answer several times. If the answer type is one number or a string of characters, then the student's answer is compared with the correct answer. If the answer type specifies several numbers, and their order is unimportant, then before comparing the answers, they are sorted in ascending order.

If student has entered a predefined number of incorrect answers, then he is suggested to look at the solution of the problem in expanded form.

III. EXAMPLES OF PROGRAM INTERFACE

Below is the text of the program in the language that is processed by the system.

```
#Z 0402 Quadratic equation with not reduced
#K constant
STRING PLS='+'
STRING DEL=':'
STRING UMN=CHR(183)
STRING MNS='-'
STRING PRB=' '
%K
#V Variable
A,B,C,X1,X2,X3,X4,D,E:INTEGER
D1,D2:FRACTION
S01,S02,S03,S04,S05,S06,S07,S08,S09:STRING
%V
#C -main program
X1:=Gener(-20,20,'<>0')
X2:=Gener(-20,20,'<>0')
X3:=Gener(2,10,'<>0')
X4:=Gener(2,10,'<>0')
A:=X3*X4
B:=X1*X4+X2*X3
B:=(-1)*B
C:=X1*X2
D:=B*B-4*A*C
S01:=VIR2(A,'x',B,C)+' = 0'
S02:='D='+STRSQUARE('B')+'-4'+UMN+'A'+UMN+ 'C='
S02:=S02+INTTOSTR(B*B)+'-4'+UMN+
INTTOSTSKOB(A)+UMN+INTTOSTSKOB(C)
S02:=S02+'='+INTTOSTR(D)
E:=ISQRT(D)
S03:=STRSQRT('D')+' = '+INTTOSTR(E)
S08:='-B+'+STRSQRT('D')
S09:='2'+UMN+'A'
S04:='x'+STRSUB('1')+'='+STRTOFRAC(S08,S09)
S08:='-'+INTTOSTSKOB(B)+'+'+INTTOSTR(E)
S09:='2'+UMN+INTTOSTR(A)
S04:=S04+'='+STRTOFRAC(S08,S09)
D1:=INT3TODRB(0,X1,X3)
PRIVED(D1)
S04:=S04+'='+DRBTOSTR(D1)
S08:='-B-'+STRSQRT('D')
S09:='2'+UMN+'A'
S05:='x'+STRSUB('2')+'='+STRTOFRAC(S08,S09)
S08:='-'+INTTOSTSKOB(B)+'-'+INTTOSTR(E)
S09:='2'+UMN+INTTOSTR(A)
S05:=S05+'='+STRTOFRAC(S08,S09)
D2:=INT3TODRB(0,X2,X4)
PRIVED(D2)
S05:=S05+'='+DRBTOSTR(D2)
%C
#F task formation
ZAD01:=S01
PST01:='Solve the equation'
%F
#P hints
PDS01:=S01
PDS02:=S02
PDS03:=S03
PDS04:=S04
PDS05:=S05
%P
#O form task
```

```
Kolstrk:=1
Kolx:=2
TipOtv1:=10
TipOtv2:=10
Kolp:=0
X01:=D1
X02:=D2
HZ1:=34
HZ2:=20
HZ3:=20
HZ5:=10
NPS01:='x'+STRSUB('1')+'='
NPS02:='x'+STRSUB('2')+'='
%O
%Z
```

Let's look at an example of how the program works. The interface consists of the following dialog windows. In “Fig. 1” shows the main window contains a list of topics and tasks on this topic.

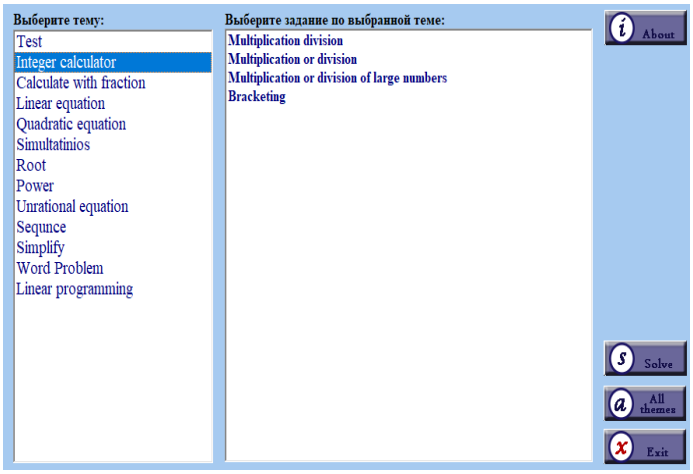

Fig. 1. Main window contains a list of topics and tasks.

After selecting a task, in the window appears a task for an independent solution. In “Fig. 2” shows a window for selecting a task.

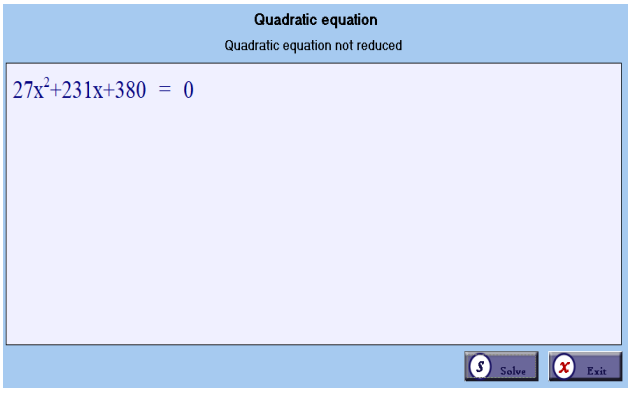

Fig. 2. Selecting a task.

Next, the student should write the answer in the special field of interactive interface of the task. In “Fig. 3” shows an example of a task.

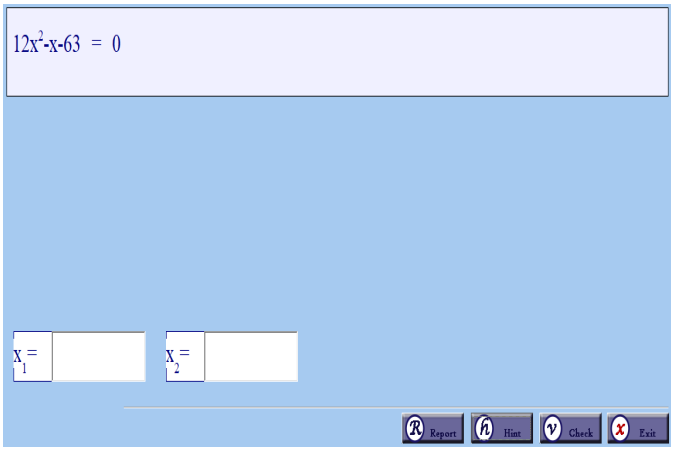

Fig. 3. Example of a task.

There is a button with the ability to see the progress of the solution in the window with progress of solving the problems. In “Fig. 4” shows an example of the progress of solving the problem.

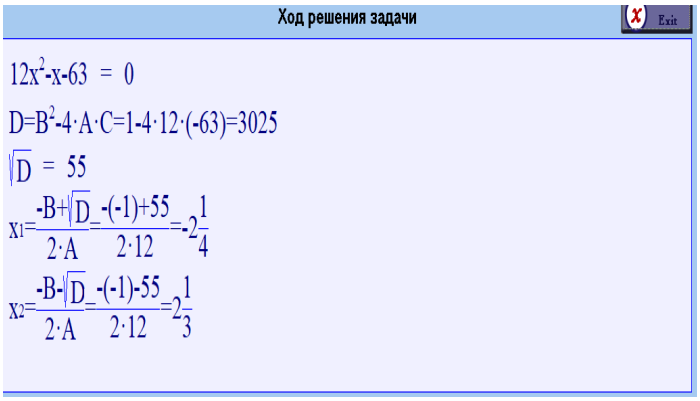

Fig.4. Progress of solving the problem.

IV. Conclusion

The described tool can be used not only as a checking knowledge tool but as a learning tool. The system can be improved by providing theoretical material. The considered system in the course of experiments showed that the use of this system increases the assimilation of mathematical knowledge among students. The main advantage of the system is the ability of displaying step-by-step solution of tasks.

References

[1] P. Cristea, R. Tuduce, Automatic generation of exercises for self-testing in adaptive elearning environments, Workshop on Authoring of Adaptive and Adaptable Educational Hypermedia (Part of WBE), 1126 (WSEAS Publishing, 2011).

[2] Mark Kellogg, Steve Rauch, Ryan Leathers, Mary Ann Simpson, David Lines, Lisa Bickel, and Jeff Elmore, Construction of a Dynamic Item Generator for K-12, Mathematics Annual Meeting of the National Council on Measurement in Education Chicago, IL April 15-19, 2015.

[3] Sinharay, S., and Johnson, M.. Statistical modeling of automatically generated items. Automatic item generation: Theory and practice, 2013, (pp. 183-195).

[4] M. Gangur, Mikulas.. Matlab Implementation of the Automatic Generator of the Parameterized Tasks. 2013, from https://www. researchgate.net/publication/259901567.

[6] M. Gangur, Automatic generation of cloze questions. Proceedings of 3rd International Conference on Computer Supported Education, 2011, CSEDU'11.

[7] Todeush Alexander, Sherman Mychailo, Methods of generating tasks for knowledge assessment Kherson State University DOI, from https://doi.org /10.32839/2304-5809/2020-10-86-42.

[6] Vysotsky V.Yu., Goguneky V.T. Tests with random parameters for automated learning modeling in applied research - Proceedings of the National Polytechnic University. X1X2011 Odessa 2009.

[7] Overview of some technologies for designing a task generator in higher mathematics for distance learning systems S.A. Mukhanov, A.A. Mukhanova, and V.V. Britvina1, SHS Web of Conferences 141, 03009, MTDE 2022.

[8] Ganna Kaplun, Automatic test generation: approaches and tools, from https://dou.ua/lenta/articles/automatic-test-generation/.

[9] Kuta Software LLC. from https://www.kutasoftware.com/ index.html.

[10] Dr Frost, from https: //www.drfrostmaths.com/index.php.

[11] Math Goodies, from https://www.mathgoodies.com.

[12] SunRav TestOfficePro Program for creating tests, from https://sunrav.ru/testofficepro.html.

[13] Overview and possibilities of the "MOODLE" system: a course for teachers, from ttps://prepod.nspu.ru/course/ view.php?id=1378.

[14] Using Item Response Theory in adaptive testing, from http://www.wikiznanie.ru/ru-wz/index.php.