World of Computer Science and Information Technology Journal (WCSIT)
ISSN: 2221-0741
Vol. 3, No. 9, 144-149, 2013

Software Support For Programming Language Tutorials

Dr. Vladimir M. Bondarev
Computer Software Department
University Of Radi Electronics
Kharkov, Ukraine.

Dr. Afif J. Almghawish, Dr. Alexandre F. Ossyka, Dr Issa S. Outoom
Computer Science Department, Al-Zaytoonah University Of Jordan
Amman, Jordan.

Abstract— Tutorials are indispensible for many university courses, such as computer programming. The paper contributes to the shift from the tutorial organization with one student solving a problem and others watching to the setting when each student solves his/her individual problems assisted by a computer and the instructor. The research resulted in creation of a computer-assisted tutorial system for the course of programming languages. The system uses a database of programming problems. It aids an instructor in preparing and delivering tutorials. It provides each student with an environment for problem selection, its solution and solution verification. The system has some special features. A problem requires writing a programming code. A student has to write only a fragment of the complete program. The context of the fragment is added by the system automatically. The system operates in a local network which connects personal computers of students and the instructor's portable computer. The first version of this computer system was successfully tested at the Kharkov National University of Radio Electronics.

Keywords- computer programming; software support; portable network server; automatic solution verification; system structure.

I. Introduction

Computer Based Instruction is an important area of scientific research and IT applications. Some university subjects, such as computer programming, cannot be learned by just attending lectures and reading books. They require tutorials, practice. Typically, programming tutorials are delivered by a human tutor in a group of 15 – 20 students, with one student solving a problem at a blackboard, others watching. Along with evident advantages, such tutorials have some limitations. For students below the group's average level, they are not really understandable, almost useless. For students above the average level, they may be boring, not effective. If some problem aims advanced students then most of the students miss the point, etc. In an ideal case, every student should solve problems that raise his/her individual knowledge level. One tutor is unable to effectively help each student with his /her problem, and to assess each solution. Tutorial computer programs are supposed to address problems of group and individual practical instruction [1], [2].

For several decades researchers and application programmers have devoted their efforts to development of software that teaches people how to program computers. Diversity of study contents and teaching methods resulted in adopting various approaches to delivering computer-based tutorials and programming training. Computer-based programming tutorials may consist of different types of files: text, audio, video, animation, program code. Tutorials may be used by students online or offline, in synchronous or asynchronous mode, individually or at a classroom setting, by considering code examples or by writing program codes themselves, and so on [3],[4].

Here are some examples of programming tutorial systems. "C++ Programming Bundle Training Video" is a CD with a set of videos each covering a particular programming topic. Videos show the process of writing examples of C++ codes with voice comments for each line. Some topics are supplied with exercises which are also done and commented by the same teacher. Here training is following the examples [5]. "Practice-It" is a web application that helps to practice in solving Java programming problems online. A student chooses a problem from a given list, types a solution, and submits it to the server. The system tests the solution and informs if it is correct or not [6]. "Online Python Tutor" is a program which creates pictures visualizing code execution. Programs may be written by a teacher or by a student, executed and traced [7]. An interesting idea of programming tutorial design consists in parsing an example source code with instructor comments into an intelligent learner environment. Students are guided step by step to develop the program solution. Explanations are auto-generated for each line of the code. So source code examples are used as self-contained tutorials [8].

There exist some software products for remote compilation and execution of a computer program code [9],[10],[11] and sets of problems that can be solved in such a mode[12],[13]. Their detailed analysis leads to the conclusion that they do not always provide a necessary speed and flexibility of adapting programming language tutorials to the individual student's knowledge level in a given study group.

Despite the existence of a large number of programming language tutorials delivered via computers, such tutorials are not always used even when they could be beneficial. In our opinion, some reasons for such a situation are as follows:

- There is a lack of information about existing tutoring programs that are suitable for presenting a particular learning content;
- Extra cost of hardware and software;
- Lack of trust to the qualification of a computerized tutor;
- It is not clear how a computer-based tutorial relates to course instruction;
- The interface may not support a sufficient interaction between a student and a tutor program;
- For an effective practice, absence of a qualified human 0 tutor is not helpful.

II. Research targets

The task of this research is to create a computer nework tutorial for a programming languages course that addresses some of the above problems:

- The system should be integrated into classroom tutorials combining advantages of computer-based and human tutoring with no questions about tutor qualification or relevance of the study content;
- It should intensify the process of problem solving by each student;
- Problem solving should require writing code (as opposed to just watching the process or multiple-choice solutions);
- Problems for solution should be adopted to the individual knowledge level of each student;
- The program should be interactive;
- It is supposed to be a low-cost solution;

In this project, verification of students' solutions and some other tasks are performed by a client-server computer program. The program operates in a wireless local area network consisting of a portable instructor's computer (server) and laptops or tablets of students which are connected by a Wi-Fi router. An instructor can create such a network by several clicks of a mouse immediately before the start of a tutorial.

III. Application scenarios

The suggested program is applied by two kinds of users: students and a tutorial instructor. The instructor activities are as follows.

Before the start of a tutorial, the instructor enters a list of student names and the problems to be solved into the program database. Evidently the name list and the problems can be used in several tutorials.

At the beginning of a tutorial, the instructor, via the program, assigns each student one or more problems to solve. By means of the program, the instructor is able to monitor how each student is solving the assigned problem and to interfere into the solution process when necessary or at the student's request. The solution correctness is verified by the program automatically without participation of the teacher.

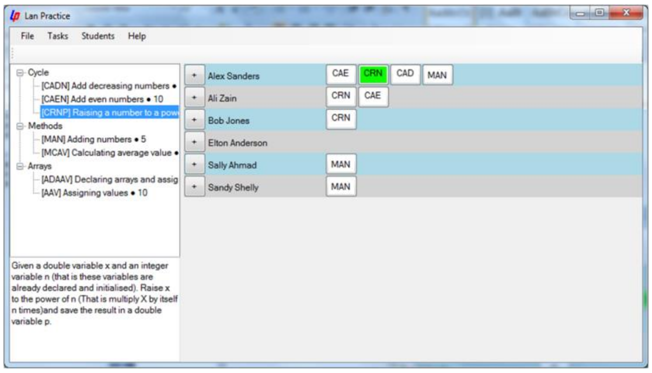
The teacher can add new problems to the initial problem set or replace some of them if the initial choice was unsatisfactory.

A student opens the web-browser window to see the statements of the problems to be solved. The student picks up one of the assigned problems, enters the solution code into input lines and sends the code to the server for automatic verification. The server inserts the solution code into the problem context code. The combined code is compiled. If the compilation is successful the program is executed. Depending on the test result, the student receives a message about a compilation error or a message that the solution passed or failed the correctness check. Code correction and solution verification can be iterated by a loop as many times as necessary.

A student can get automatic prompts in the process of a problem solution if such prompts were provided by the instructor as a supplement to the problem statement. In order to participate in such a tutorial, a student needs only a standard web-browser and a Wi-Fi receiver in a laptop, tablet, or a mobile phone. Naturally, a computer with a real keyboard is much more convenient.

3.1. Instructor interface

The instructor user interface consists of two windows: the main window and the problem status window (See Figure 1). The main window is divided into two parts. The left part contains the problem panel, the right part displays the list of students. The problem panel shows the set of problems for solution presented as a tree similar to the file system tree. An instructor can select, add, delete, and update problems that is to perform the standard operations for manipulating data.


Lan Practice
File Tasks Students Help
Cycle
[CADN] Add decreasing numbers •
[CAEN] Add even numbers • 10
[CRNP] Raising a number to a pow
Methods
[MAN] Adding numbers • 5
[MCAV] Calculating average value •
Arrays
[ADAAV] Declaring arrays and assig
[AAV] Assigning values • 10

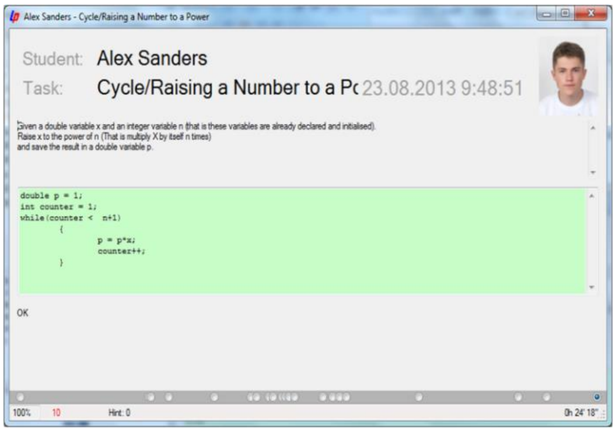
+	Alex Sanders	CAE	CRN	CAD	MAN
+	Ali Zain	CRN	CAE		
+	Bob Jones	CRN			
+	Elton Anderson				
+	Sally Ahmad	MAN			
+	Sandy Shelly	MAN			

Given a double variable x and an integer variable n (that is these variables are already declared and initialised). Raise x to the power of n (That is multiply X by itself n times)and save the result in a double variable p.

Figure 1. Program main window

The list of students is a sequence of data bars. The left-most position of each bar contains a student name. All problems assigned to a student for solution are shown as buttons to the right of the student name. The unsolved problems are represented by uncolored buttons, and the solved problems by green buttons. In order to assign a new problem to a student, it is necessary to select it in the problem pane of the main window, and click the button "+" near the student name. So if necessary, the same problem can be assigned to several students.

Clicking a button opens the problem status window (See Fig. 2). In this window one can see the student personal data, name of the problem and its statement, the chronology of the problem solution process. The chronology is a horizontal bar below the window with points on it. Points represent the time moments when the student sent his/her solution for verification. By selecting a point in the chronology bar, the instructor can see the code which was sent by the student to the server for verification at the chosen moment of time. The result of verification is also visible. So the teacher can trace the whole process of the problem solution and understand the student's knowledge gaps that hindered the successful problem solution.



Student: Alex Sanders
Task: Cycle/Raising a Number to a Pc 23.08.2013 9:48:51

Given a double variable x and an integer variable n (that is these variables are already declared and initialised).
Raise x to the power of n (That is multiply X by itself n times)
and save the result in a double variable p.

```
double p = 1;
int counter = 1;
while(counter <  n+1)
        {
                p = p*x;
                counter++;
        }
```

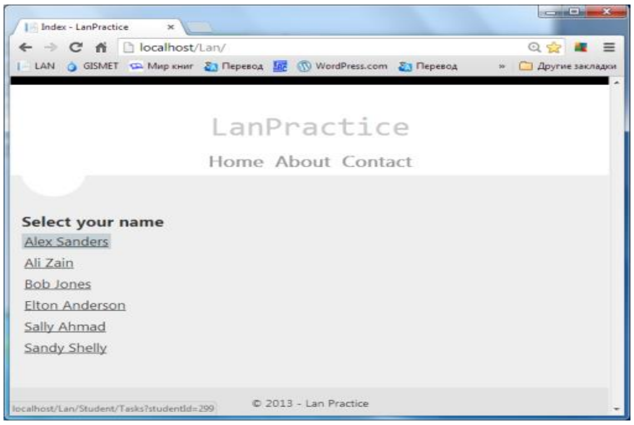
OK

100% 10 Hint: 0 0h 24' 18"

Figure 2. Problem status window

3.2. *Student interface*

To see the group name list, a student connects to the local network and selects the application URL in the browser (see Fig. 3). The student clicks on his/her name in the list to open the problem solution window (see Fig. 4).


Index - LanPractice
localhost/Lan/
LAN GISMET Мир книг Перевод WordPress.com Перевод Другие закладки
LanPractice
Home About Contact

Select your name
Alex Sanders
Ali Zain
Bob Jones
Elton Anderson
Sally Ahmad
Sandy Shelly

localhost/Lan/Student/Tasks?studentId=299
© 2013 - Lan Practice

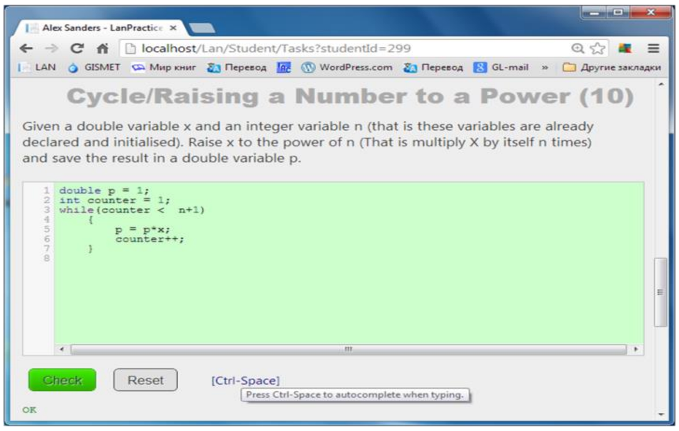
Figure3. Group name list

The problem solution window contains all problems assigned to the student by the instructor. Each problem has its name, its statement, and its area for the program code input. The student picks any unsolved problem in the window and enters the solution code into its input area. To verify the solution by the server, the student clicks the button "Check". In a short time the server's answer is returned to the student. The button "Reset" displays the problem input area in its initial state.

A problem may contain hints to the solution provided by a teacher. In such a case besides buttons 'check" and "Reset", there are buttons for hints. Each button is labeled by the word "Hint" and the number specifying the final problem grade after using the hint.

IV. PROBLEMS

The discussed system has a special feature. A problem solution does not require a complete


Alex Sanders - LanPractice
localhost/Lan/Student/Tasks?studentId=299
LAN GISMET Мир книг Перевод WordPress.com Перевод GL-mail Другие закладки
Check
Reset
[Ctrl-Space]

Cycle/Raising a Number to a Power (10)

Given a double variable x and an integer variable n (that is these variables are already declared and initialised). Raise x to the power of n (That is multiply X by itself n times) and save the result in a double variable p.

```
double p = 1;
int counter = 1;
while(counter <  n+1)
     {
          p = p*x;
          counter++;
     }
```

Press Ctrl-Space to autocomplete when typing.

OK

Figure 4. Problem solution window

program code with data input, output, etc. It is necessary to design only a code fragment which works correctly in the given context. The description of the code fragment context and the code requirements constitute the problem statement. Such an approach to programming problem solving has some advantages:

- It is possible to state very different problems.
- A student is relieved from writing standard parts of any program code. It saves the time of a tutorial.
- A student can concentrate mainly on the algorithmic side of the program, not on its routine features.

Here are some examples of programming problems to be solved in C# at t utorials.

4.1. *Problem statement examples*

Assignment. Declare an integer variable a, initialize it by 5. Declare an integer variable b initialized by 1. Add 1 to both variables. Add the values of both variables and save the sum in an integer variable c.

Exchanging the values of two integer variables. There are two integer variables a and b (that is they have been declared and initialized by some values). Write the code that exchanges the values of a and b. (See Fig. 5)

Triangle sides. Given three float variables a, b, c with some values in them. It is necessary to determine if these values can be side lengths of some triangle. If they are possible length values assign true to a logical variable t. Otherwise, assign false to t.

Factorial. Declare a static method Fill() which receives a non-negative integer number and returns the factorial of this number. Note: 0! = 1.

Filling an array by element values. Declare a static method Fill() which receives an integer number n and returns an array of n elements filled by values 0, 1, 2, … , n-1.

A circle with constructors. Declare a class Circle with double type data members X and Y (center coordinates) and R (radius). The class should contain a constructor with three parameters (coordinate x, coordinate y, and radius r). Also it must contain a default constructor that assigns zero values to all data members of the class.

Zero items. Declare a class MyList which inherits the library class List<T> and has a data member int ZeroCount. This data member contains the number of zero items in a MyList object. Zero item is a list item with default (T) value.

Queen iterator. Declare a static iterator Queen() which gets the queen's position on an empty chessboard and produces the list of all squares that are under immediate attack of the queen. The queen's square is not under attack. A chessboard position is defined by a two-symbol string, for example, "a1", "e2", "b8".

4.2. *Problem representation*

A problem to be solved at a tutorial consists of four parts: name, problem statement, hints and the context.

A name includes the path and the name itself. The path determines the place of the problem in the problem collection. The collection has a hierarchical structure. The name must be unique for all the problems with the same path.

A problem statement is an arbitrary text in a language understandable for a student.

A hint should help a student to solve a problem if the solution is not clear. There may be several hints for the same problem but not less than one. The first hint is given by default in the code input area when the problem is opened. The first hint may be just an empty line in the window.

A context is a compilable computer program which includes as its part the code of the problem solution entered by a student. In the context, the solution code is placed inside the "string-brackets": //BEGIN … //END for C++ and C# and #//BEGIN … #//END for Python languages. These delimiters are transparent for compilers which simplifies the design of the solution by a student.

In the system database, the problems are stored in xml format. The external representation of a problem may be the same as internal (in xml) or may have an alternative form, a so called plain format.

In the plain format, all parts of the problem are placed sequentially. They are separated by a line of three or more hyphens. A hyphen line may contain an information marker at its end. For example a hyphen line preceding the problem statement may be terminated a programming language marker: cs, cpp, or py. The delimiting line before a hint contains a number which is the grade given for the final correct solution in case of using this hint. With each hint such numbers are decreasing but not necessarily.

The experience of the system use shows that it takes from 5 to 20 minutes to prepare a problem for input into the system database. Once it is done the problem can be used many times.

It may be useful to have a common database of formatted problems for programming tutorials. All instructors of the subject can use the problems without preparation overheads.

V. PROJECT STRUCTURE

The project consists of two application programs: a network program named LanPractice and a desktop application LanTutor. These programs use two libraries: Compilers and Repository.

The web-application LanPractice supports the student activity. By means of this program, a student receives the problems for solution, writes the solution code, sends it for verification, and receives the verification results.

The desktop application program LanTutor is a tool of an instructor. It supports such activities: selection of problems for solution at a given tutorial, assignment of problems to students, manipulating list of students, and monitoring the process of a problem solution by any student in the class.

```
Assignment/Exchange of Variables' Values
--------------------------------------cs
Given two integer variables a and b. They have been already declared and got their values. Write a code which exchanges their  values.
-------------------------------------10
//int a = 3, b = 5; - Do not remove the comment slashes in this line.
-------------------------------------9
//int a = 3, b = 5; - Do not remove the comment slashes in this line.
//Use a third variable.
  int t = a;
------------------------------------------
using System;
public class Program
{            public static int Main()
             {            int a = 3, b = 5;
//BEGIN
                          int t = a;
                          a = b;
                          b= t;

//END
                          If(a == 5 && b == 3) return 0;

                          return 1;
```

Figure 5. An example of a problem presented in the plain format.

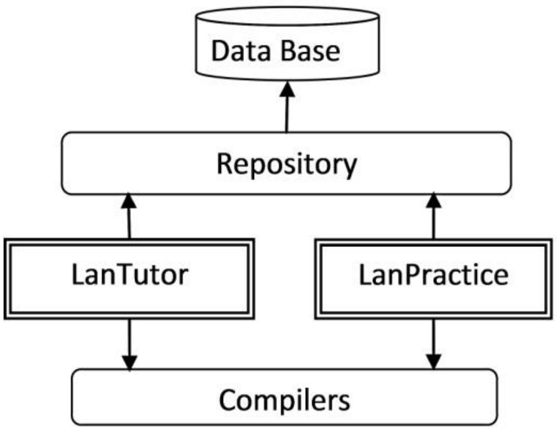


Figure 6. Project block diagram

Both programs are connected via a common database. The access to the database is provided by the library repository. Another library, Compilers, provides compilation of a student's code together with the problem context and execution of the code. This library is also used by the two programs.

The general structure of the project is presented on Figure 6. Arrows on the diagram denote an exchange of data and services between system components.

VI. CONCLUSION

Individual independent solution of programming problems is the only way to learn how to develop computer programs. The approach suggested in the paper increases considerably the number of problems solved by each student at a tutorial. It makes the learning process more intensive and fruitful.

At the present time, students can solve programming problems by means of the system in C#, C++, and Python. The system is open in respect of programming languages. The set of usable languages can be easily expanded.

An important part of the project is the set of specially formatted problems that are solved at tutorials supported by the system. All instructors interested in the system are supposed to contribute to creation of this problem set. Multiple sources of problem ideas increase the quantity, quality and variety of problems solved at tutorials. It also unifies the study material covered by different instructors in the same course.

The suggested hardware solution is a portable low-cost approach to introducing computer-based tutorials into the programming subject. Such tutorials do not require special lab rooms, additional university hardware or software. Everything is stored in the portable computer of an instructor. Evidently, with this system, an instructor has to devote more time and effort to the initial preparation of each tutorial. When advantages of this approach to delivering tutorials become generally recognized at a university it is possible to shift to a university centralized system. It will relieve an instructor from creating his/her own local network at the start of each tutorial.

REFERENCES

[1] R.C. Clar, R.E. Mayer, E-Learning and the Science of Instruction. Proven Guidelines for Consumers and Designers of Multimedia Learning. 3d edition, Pfeiffer, 2011, 528 p.

[2] O. Kozar, The use of synchronous online tools in private English language teaching in Russia, In Distance Education, 33(3), 2012, p 415-420.

[3] G. Salmon, E-moderating: The key to teaching and learning online. 2nd edition, Routledge Falmer, 2004, 256 p.

[4] A.S. Gibbons, P.G. Fairweather, Computer-Based Instruction: Design and Development. Education Technologies Publications, Inc., 1998, 530 p.

[5] M. McMillan, C++ Programming Bundle Training Video. A Practical C++ Training Course that Teaches Real World Sills. Date Released 2012-07/04. (http://www.infiniteskills.com/training)

[6] Practice-It! A Web-based Java Practice Problem Tool for Computer… (http://practiceit.cs.washington.edu/)

[7] Ph.J. Guo, Online Python Tutor: Embeddable Web-Based Program Visualization for CS Education. In Proceedings of the ACM Technical Symposium on Computer Science Education (SIGCSE), March 2013, p.p. 579-584.

[8] R.R. Gajraj, et. al., Transforming Source Code Examples into Programming Tutorials. ICCGI 2011: The Sixth International Multi-Conference on Computing in the Global Information Technology, 2011, p.p. 160-164.

[9] Pascal ABC WEB. Modern Programming in Pascal Language. (Russian language) (http://pascalabc.net/).

[10] TopCoder, a Corporation Arranging Competitions in Sport Programming. (Russian language) (http://www.topcoder.com/).

[11] Ejudge, a System for Arranging Various Actions where Automatic Program Verification is Required. (Russian language) (http://ejudge.ru/).

[12] M.E. Abramian, S.S. Mikhalkovich, Web-Environment for Program Development and Instruction. In Open Systems. Database Systems. No. 10, 2012, p.p. 56-59. (Russian language).

[13] Programming Taskbook. A Digital Set of Problems in Programming. (Russian language). (http://taskbook.com/ru/).

AUTHORS PROFILE

Dr. Vladimir M. Bondarev is with the Full professor of the Computer Software department, Kharkov National University of Radio Electronics, city of Kharkov, Ukraine.

Dr. Afif J. Almghawish is with the Associate professor of the Computer Science department, faculty of Science and Information Technology, Al-Zaytoonah University of Jordan, Amman, Jordan.

Dr. Alexandre F. Ossyka is with the Associate professor of the Computer Science department, faculty of Science and Information Technology, Al-Zaytoonah University of Jordan, Amman, Jordan.,

Dr. Issa S. Ottoum is with Assistant professor of the Computer Information System department, faculty of Science and Information Technology, Al-Zaytoonah University of Jordan, Amman, Jordan.