Training the IT students skills in the digital educational environment

I A Andrianov¹, S U Rzheutskaya¹, A A Sukonschikov¹, E N Davydo¹va and A A Zaslavskiy²

¹ Vologda State University, 15, Lenin str., Vologda, 160000, Russian Federation
² Moscow City University, 4 Second Selskohoziajstvenny pass., Moscow, Russian Federation

E-mail: igand@mail.ru

Abstract. The article discusses a distant trainer for students’ IT skills, developed at Vologda State University. The history of the resource development, its functionality, architecture and implementation features are considered. An example of using the resource for teaching the course "Mathematical logic and theory of algorithms" is given.

1. Introduction

In the process of teaching IT students, much attention is paid to the development of their professional skills. Skills in software development are of particular importance, including the design of data structures and algorithms, the implementation and testing of program code, working with databases, etc. Such skills are developed gradually by completing a large number of training tasks while studying various specialized disciplines of the IT direction. In this regard, an important problem arises of supporting the continuous process of developing professional skills in software development in the digital educational environment.

The difficulty of this problem is as follows. For teaching programming and related disciplines, there are not enough opportunities provided by standard distance learning systems (for example, MOODLE, which is widely used in Russian universities). Of course, to master programming, students need to study a lot of theoretical material, and it is also advisable to take tests for its assimilation. However, the main way to get software developer skills is to independently develop (under the supervision of a teacher) educational software projects, starting with the simplest programming tasks and gradually increasing the complexity. This statement is recognized by the authors of all analyzed works devoted to teaching programming. It is substantiated in detail in [1].

Independent work of students, allowing them to acquire software development skills, needs substantial support from the digital educational environment. Currently, there are a number of software products specifically designed for teaching programming, for example, Alice (USA) [2], Codewitz-Minerva (Finland) [3], WAPE (Russia) [4]. Web resources for training database query development skills are also very popular, for example, sql-ex (www.sql-ex.ru). There are a number of massive open online courses platforms (MOOCs) that host programming courses with good computer support for skills training. However, each of the listed resources can be recommended to students only as an additional tool for the study of a specific discipline. To support the continuous development of the skills
of IT students, a universal interdisciplinary educational resource is required that corresponds to the educational process at the university and educational standards [5].

The article presents a distant trainer of IT skills developed at Vologda State University. This system is used in the process of teaching disciplines that mutually develop students' skills in software development, working with databases, etc. This educational resource, available at atpp.vstu.edu.ru/acm, is an integral part of the digital educational environment for teaching students of several IT areas. It has been successfully used in the educational process for over 15 years and is in continuous development.

2. History of creation and development of the resource

The idea of creating this simulator arose in the process of preparing students for programming competitions. By that time, systems for automatic online checking of problem solutions were already successfully used all over the world during programming competitions. A striking example is the International Collegiate Programming Contest (ICPC). Teams of Russian universities are the undisputed leaders of the final round of this competition, which is partly due to the good level of content and implementation of Russian automated systems for preparing for programming competitions. It is noteworthy that web resources hosting such systems, for example, Timus, Codeforces, ACMP, etc., have been steadily popular for many years.

When testing solutions, the following concept is applied. The solution to the problem (the source text in one of the programming languages) is considered as a black box. The automated testing system contains a set of tests for each task. It runs the solution with different test inputs and compares the result to the correct test output (or performs more complex checking steps). The solution is considered correct if all tests are passed.

After analyzing several resources for training and programming competitions, the authors came to the conclusion that the principle of on-line testing can and should be applied in regular practical programming lessons, as well as in organizing students' independent work. However, it turned out that the existing systems for on-line training and competition are not fully suitable for use in programming training courses for the following reasons:

- such systems are intended mainly for checking of solutions, while the learning elements are not very developed (the system does not help the student to find an error in solving the problem, but only states the presence of an error);
- the systematization of problems is not very developed, there are no built-in possibilities of creating learning scenarios, which is not suitable for the “step by step” teaching method used;
- sets of problems include problems of high complexity, offered at programming contests;
- the functions of accounting and control that accompany the educational process at the university are not sufficiently developed.

Taking these points into account, teachers usually recommend such resources for training and programming competitions only to the most motivated and advanced students.

The authors’ desire to improve the efficiency of the training process led to the formulation and solution of the problem of developing a training web resource, in which the feature of automatic testing of solutions is subordinate to the feature of education. The function of automatic testing of solutions allows us to use the resource as a trainer of software development skills. First of all, the following educational features were implemented:

- help for students in solving the problem in the form of tips that facilitate the process of localization and correction of errors in the wrong solution;
- support of training scenarios developed by teachers;
- selection of the best solutions for each problem according to the criterion of performance and the required hardware resources (students who have passed the problem have access to the source code of its best solution). Teachers can add their own solutions that are available to
students who have solved the problem. Such solutions may not be the most efficient in terms of demand for hardware resources, but contain interesting ideas or can be considered as examples of high-quality program code.

The skills of a software developer are interdisciplinary. Therefore, when forming the bank of tasks, it was decided not to use the systematization of tasks in accordance with the disciplines of the curriculum. Instead, the trainer uses a more general concept of a topic. Each topic allows to develop some specific skills, and it can relate to a specific discipline or its section, or it can be interdisciplinary. Topics are organized in a hierarchical structure, in which the number of levels is not limited. Each task relates to one or more different topics.

Filling the bank of tasks turned out to be the most time-consuming process when creating a resource, since in order to perform automated checking of solutions, it is necessary to prepare a complete set of tests for each task. Some tasks require the development of an individual checking program (so called «checker»). The first version of the trainer contained just over 200 tasks. At present there are more than 2000. Students take an active part in replenishing the bank of tasks. The quality of the tests they prepare is monitored by teachers and group mates.

All detailed information about the work of each student is recorded in the trainer's database. There is a possibility of generating reports for each lesson, each student or group. This allows teachers to fully control the learning process.

Throughout the long-term operation of the simulator, its concept and the basic functionality described above have remained unchanged. However, a number of additional features have been added. We will briefly list the most useful ones:

- Checking students' solutions for plagiarism. Unfortunately, the experience of using the trainer in the educational process has shown that there are cases of students using other people's solutions from time to time. To combat this negative phenomenon, during the checking of the solution, it is also compared with other solutions of the same problem.
- Support for subtasks and the ability to give partial points for the solution. This function is very useful for teaching, as it allows to evaluate not only absolutely correct and effective solutions, but also solutions that work only on a certain subset of valid input data.
- Support of programming contests, including virtual ones (in which participants start at different times). We use this opportunity to hold various competitions, including the municipal stage of the All-Russian school Olympiad in informatics. In contests, different methods of calculating the rating table are supported. According to the ICPC rules, the rating is determined based on the number of fully solved problems and the time spent. According to the rules of Russian school Olympiads, the rating is built according to the sum of the points scored.
- Support for interactive tasks. In a traditional problem, the input data are immediately available to a student's program. The program calculates the answer, then this answer is checked for correctness. In the interactive task, a student program communicates with a jury program during its execution. This allows learn in practice an important class of problems in which input data are not provided in advance, but are gradually formed in response to our actions.
- Support for problems with a plain text answer instead of source code. This feature allows to use the system not only for programming courses, but for almost any academic subjects. It can also be useful for problems with long-term computation of the answer (for example, in machine learning problems, the answer is sometimes computed for several hours). In this case, it is advisable to load the answer into the system, but not the program that calculates it. A similar approach is used, for example, in the well-known web resource Kaggle.
- The ability to assign individual educational trajectories to students and monitor learning progress. The system automatically calculates indicators of the difficulty of problems, which allows teachers to select problems that correspond to the student's capabilities.
Next, we will consider some key points of the resource implementation.

3. Architecture and implementation features of the resource

The system architecture shown in figure 1 is designed in accordance the classic client-server concept. Users interact with a web browser, which sends requests to a web server. The web server contains fast-cgi scripts that process user requests and generate html pages based on templates and database content.

![Figure 1. General architecture of the electronic resource.](image)

The processing of the submitted solutions is performed by the checking server. Due to the support of distributed work, several checking servers can be launched within the network or on one multiprocessor computer. To compile solutions, every checking server uses a set of compilers and interpreters that are installed and configured by the system administrator. Let's take a closer look at the main functions and capabilities of some system modules.

Authentication module performs user authentication. The standard “Digest” authentication scheme is used, which excludes the possibility of intercepting the password over the network. The function of recovering a forgotten password by email has also been implemented.

The problems viewer displays problem available for a given user, systematized on the topics. This topics are displayed in the form of a hierarchical structure (tree).

The problem editing module allows teachers to create new problems, as well as edit existing ones. The ability to manually create tests and load them from the archive is supported. The teacher can choose one of the standard checking programs, or add a new checker for the particular task.

The module for displaying the check results provides users with reports about the checking of their solutions. Depending on the settings of the particular problem and the results of the check, this module can provide additional hints to help students find errors in their solutions.
The module for generating statistics allows to build various reports on users, groups and problems. It supports displaying the "best" solutions for each problem (according to some criteria). The module also maintains the overall rating of the participants, thereby introducing a competitive element into the learning process.

The module of solutions checking implements the execution of the students' programs on test input data. During execution, it monitors compliance with restrictions on the use of processor time, memory, prevents access to prohibited resources (for example, to the network).

The plagiarism detection module allows to identify suspiciously similar solutions. In the implementation of this module, an original approach is used, in which the comparison is performed not of the source codes, but of the compiled program modules (object codes). This allows to successfully deal with such intentional code modifications as renaming identifiers, inserting deliberately non-executable code, unused data, etc. For more details about the implementation of the plagiarism detection subsystem, see our article [6].

4. An example of using the resource in the course "Mathematical Logic and Theory of Algorithms"

The successful experience of using the trainer for courses on programming and databases made the teachers want to use it for other academic disciplines. Next, we will briefly describe the experience of using the resource for organizing practical work on the course "Mathematical logic and theory of algorithms".

In this course students carry out a number of practical works, for which it is quite possible to automate the checking of solutions. In this article, we will consider the implementation of automatic verification of practical tasks on the following three topics: "Digital logic circuits", "Turing machine", "Normal Markov algorithms".

One of the topics of practical work is the development of digital logic circuits. A typical assignment from this topic is to design a circuit that contains a given number of inputs and outputs and works as described in the assignment. There should be no cycles in the constructed circuit, that is, the output of any element should not return again to its input [7]. Task example: build a circuit with three inputs and two outputs so that the binary number at the output is equal to the number of ones at the inputs. One of the possible solutions to this task is shown in figure 2.

![Figure 2. An example of solution the problem of developing a logical circuit.](image)

Our students use the Atanua program [8] to design the logical circuits. This program has a user-friendly interface and contains all the necessary functionality. An important feature of this program is that the resulting files are saved in XML format. This made it easy to implement their parsing in the checking program.

The developed checker works as follows. First, the XML file is parsed, and a graph corresponding to the schema is built in memory. Next, using the depth-first traversal algorithm, the graph is checked for the presence of cycles. If a cycle is detected, the solution is considered incorrect. Next, the values of
the logical functions are calculated corresponding to each of the outputs of the circuit. The resulting values are compared with the correct answer. If the solution turned out to be correct, then the length of the longest path from the inputs of this circuit to its outputs is additionally calculated. This value is used to determine the final score for the solution.

To carry out practical work on the topics "Turing Machine" and "Normal Markov Algorithms", the corresponding simulators from the site [9] are used. These simulators do not provide the ability to automatically check solutions. Fortunately, they support export the created models to text files. Taking this opportunity into account, we have developed our own interpreters that work on the command line and accept such text files as input data. The developed command-line interpreters are easy to integrate into the automatic system as checking modules.

5. Conclusion
The experience of using a distant trainer of IT skills convincingly proves the advantages of digital education in the process of training specialists in the field of software development. The effectiveness of the learning process is significantly increased, since the teacher in the classroom is relieved of the routine work of checking students' solutions and keeping records on paper. Automation improves the quality of both these jobs at the same time. The freed up time of teachers can be used for individual work with each of the students, which is especially important when teaching specialized IT disciplines that are sometimes not easy to understand.

Most students willingly work with the trainer on their own, quickly master its capabilities, and notice the benefits of this way of developing professional skills. From the very first steps of training, the students get used to the neat design and thorough testing of their solutions. The element of competition, which is necessarily present in this form of education, increases interest in studying IT courses. All of these factors ultimately contribute to the successful development of IT students' skills in the digital educational environment.

References
[1] Kelleher C and Pausch R 2008 Lowering the barriers to programming: A taxonomy of programming environment and languages for novice programmers ACM Computing Surveys 37(2) 83-137
[2] Dann W, Copper S and Pausch R 2006 Learning to program with Alice (NJ: Prentice Hall) p 384
[3] Kirsti A-M 2008 Problems in learning and teaching programming - a literature study for developing visualizations in the CodeWiz-Minerva project (Institute of Software Systems, Tampere University of Technology, Finland) p 13
[4] Kasyanov V N and Kasyanova E V 2008 WAPE – a system for distance learning of programming IFIP International Federation for Information Processing 261 355-6
[5] Rzheutskaya S Y and Kharina M V 2017 Interdisciplinary cooperation in the integrated information learning environment of technical university Open Education 2 21-8
[6] Andrianov I, Rzheutskaya S, Sukonschikov A, Kochkin D, Shvetsov A and Sorokin A 2020 Duplicate and Plagiarism Search in Program Code Using Suffix Trees Over Compiled Code Proc. of the 26th Conference of Open Innovations Association FRUCT 16-22
[7] Andrianov I A 2019 Automation of logical circuits checking Proc. of 10 Int. Sci. and Tech. Conf. Intelligent Information Technologies and Intelligent Business 114-7
[8] Atanua Real Time Logic Simulator Retrieved from: https://sol.gfxile.net/atanua
[9] Website of Konstantin Polyakov Retrieved from: http://kpolyakov.spb.ru