

*X World Conference on Computers in Education
July 2-5, 2013; Toruń, Poland*

On-line Contests Hosting Service as a Tool to Teach Computer Science Students Programming

Marek Nowicki, *faramir@mat.umk.pl*

N. Copernicus University, Faculty of Mathematics and Computer Science,
Chopina 12/18, 87-100 Toruń, Poland

Łukasz Mikulski, *frodo@mat.umk.pl*

N. Copernicus University, Faculty of Mathematics and Computer Science,
Chopina 12/18, 87-100 Toruń, Poland

Marcin Piątkowski, *martinp@mat.umk.pl*

N. Copernicus University, Faculty of Mathematics and Computer Science,
Chopina 12/18, 87-100 Toruń, Poland

Rafał Kluszczyński, *klusi@mat.umk.pl*

N. Copernicus University, Faculty of Mathematics and Computer Science,
Chopina 12/18, 87-100 Toruń, Poland

Piotr Bała, *bala@mat.umk.pl*

N. Copernicus University, Faculty of Mathematics and Computer Science,
Chopina 12/18, 87-100 Toruń, Poland
University of Warsaw, ICM, Prosta 69, 00-878 Warszawa, Poland

Abstract

Computer science education is composed of two main approaches. One is to get students familiar with information technology where students learn how to use modern applications for information processing. Second approach is teaching students how to solve problems with the help of computers. This is always related with programming skills and abstract thinking. In this paper we describe developed by us service called *ZawodyWeb*. It checks on-line computer programs sent by students and allows the creation of assignments, to submit, compile and run solutions. We present also validation of the services based on experience gathered during teaching of computer science students.

Keywords

distance learning, programming, computer science, higher education

INTRODUCTION

Computer science education is composed of two main approaches. One is to get students familiar with information technology where students learn how to use modern applications for information processing. Second approach is teaching how to solve problems with the help of computers. This is always related with the programming skills and abstract thinking. First, students have to create a model of real-life problem. Next, with the help of computer they have to design application which solves it correctly and, what is also important, efficiently.

Over the years, different kinds of programming contests became very popular. The most popular are: ACM International Collegiate Programming Contest (<http://cm2prod.baylor.edu>), Imagine Cup (<http://imaginecup.com>) or Top Coder (<http://www.topcoder.com>). For that purpose many contests hosting services emerged. They enable remote validation of implemented solutions. Some of them are more oriented on algorithms and check if the solutions uses optimal one. Such systems are used in contests like:

- Polish Olympiad in Informatics (<http://www.oi.edu.pl>),
- The ACM International Collegiate Programming Contest (<http://cm2prod.baylor.edu>),
- Polish Academic Championship in Team Programming (<http://amppz.cs.put.poznan.pl>),
- Poznań Open Team Programming Championship (<http://www.mwpz.poznan.pl>).

Those systems are used in Poland to work with gifted IT students. A result is Polish students being amongst the best in the world in various programming contests (Diks & Madey 2008). Commercial companies also start to use automatic code validation platforms, especially at the first stage of recruitment process. This way they sort out candidates who cannot write simple programs correctly. A good example is Codility (Codility 2013) which specializes in the recruitment of programmers for international companies.

In this paper, we describe a service developed by us called *ZawodyWeb* that allows the checking of on-line computer programs sent by students. Presented service allows us to create assignments, to submit, compile, run solutions and check their validity. We present also validation of the services based on the experience gathered during teaching computer science students.

ZAWODYWEB SYSTEM

At the beginning of traditional learning loop, a teacher prepares suitable learning material. Then, during the lesson, he passes the knowledge to the students. Then the students play the role. They can acquire knowledge in full, in part or not at all. Some of them need extra time to master the material. All players would like to know whether the knowledge is well acquired. There is simple way to verify this: tests and exams. Unfortunately, this solution requires from students to be familiar with the material and checks only their knowledge. Tests and exams do not motivate students for further collection of knowledge - the score is final.

The other possibility to measure students' skills is through assignments to be solved at home. Unfortunately, this requires significant effort to check assignments, especially in the computer science, where one task can be solved in many different ways. To solve this problem students and teachers at the Faculty of Mathematics and Computer Science N. Copernicus University developed *ZawodyWeb* (WebContest) system for the automatic validation of the computer programs.

Overview

The *ZawodyWeb* system is fully web based and all activities both for teacher and student are performed on-line using web browser. A teacher can create new problem by entering its name, description, programming languages that can be used to solve the problem, test data (the inputs and outputs) and information about the accessibility.

A student has access to the problem description and after reading it he tries to solve it on the local machine, using the one of the listed programming languages. The student can check solution by himself preparing adequate input data, and calculating, mostly using pen and paper, output data, and checking if program he wrote generates the same output. Next, the student can submit his solution to the *ZawodyWeb* system, which automatically validates it. Validation is performed

through compilation, execution and then checking of the program results on the given input data sets. For the correct solution, the student receives points depends on the difficulty of the task and the correctness of the solution. He is convinced that solution is good enough when it passes all tests. When the output of one or more tests differs from those created by the teacher or program execution exceeds the time limit, the student gets that information and can improve his solution. Then he can check correctness of the new version. The student can also abandon solving the problem and select another one if it exists.

History

The history of the system used to the remote evaluation of solutions of programming problems goes at the Faculty of Mathematics and Computer Science ahead to 2002 (Bała et al 2003). It is connected with the Computer Science Contest (Konkurs Informatyczny), organized by the Faculty. In the early years of the Contest, the participants sent a solution first on floppy disks by the traditional mail and then by e-mail. Then solutions were evaluated manually.

In the 2002, a web system for registration of participants of the Computer Science Contest and collection of submitted solutions was created. In the coming years, the system was increasing its functionality and allowed for evaluation of solutions and generation of ranking after the competition finished. The problems of the Contest were solved using PHP or Java applets. These sometimes caused problems when launching solutions in the environment used by a Contest Commission therefore the participants were allowed to check the behaviour of their solutions in an environment similar to that used by the Commission (Nowicki, Kluszczyński & Bała 2008).

In the 2006 a system for Programming Mini Contest of the Toruń Festival of Science and Arts was created. It could automatically check the solution developed in PHP and Java. Its web interface was written in PHP. The server side was based on the Judge daemon written in Python which periodically checked if there is a solution to validate. If so, then the correctness was checked. In the case of multiple solutions simultaneously sent by the participants the queue was used and all submissions were put on hold until previous ones were processed.

One year later, new system called *ZawodyWeb* was created. It allowed for automated testing of solutions developed in the programming languages like Pascal, C, C++, Java and PHP. A multiple Judge daemons were introduced and *JudgeManager* was added to architecture of the system to determine whether there is a solution to check. If there is one it sends information to free Judge, which checks the correctness of a solution. As a result, it was possible to check simultaneously many solutions. The whole system (web interface, *JudgeManager* and *Judge*) was written in Java.

The current version of *ZawodyWeb* was created in the 2009 (Bała et al 2011). Principles of the system remained the same but newer technologies were used and the user interface has been modernized. The system allows for automatic validation of solutions written in the multiple programming languages: Pascal, C, C++, Java, Python as well as manual evaluation of the submissions. The system architecture composes of one instance of *JudgeManager* and multiple instances of *Judges*. Thanks to new technology it is possible to maintain system easily and extend its functionality with relatively small effort.

Technicalities

The *ZawodyWeb* system is created in the Java language using the Spring Framework and JavaServer Faces libraries (Facelets, Richfaces, Restfaces, etc.). The system is hosted on the Apache/Tomcat web server. The PostgreSQL database is used to store data and Hibernate is used for mapping Java objects to the data.

The *JudgeManager* and *Judges* are written in Java. Architecture of the system is presented in the Figure 1.

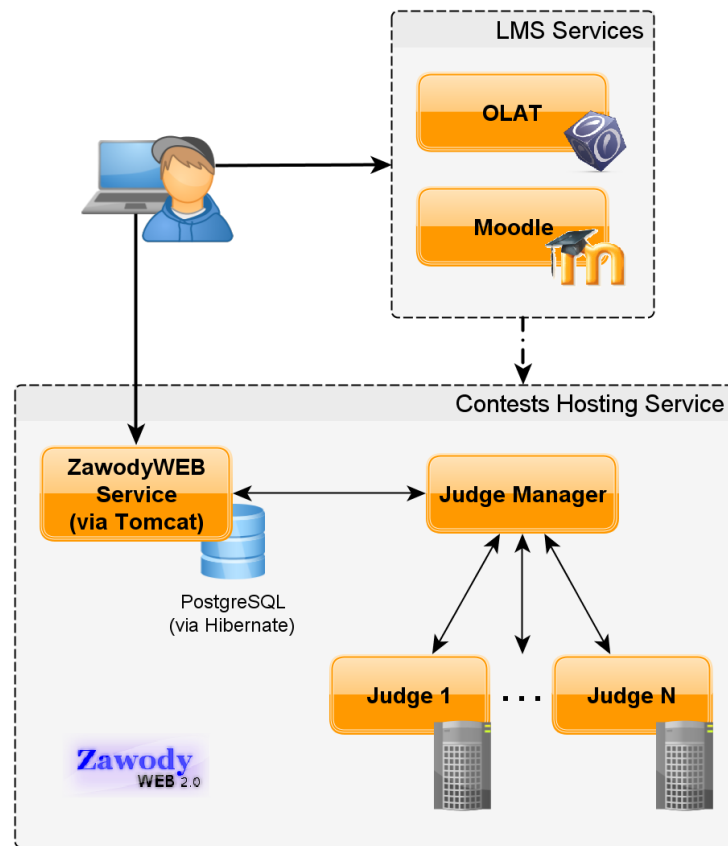


Figure 1: Architecture of *ZawodyWEB* system

Functionality

Most of the functionality of the *ZawodyWeb* system is related to the organization of programming contests, but it can be also used to support on-line learning. System allows to define contests, programming tasks, scores and rankings. System offers different roles for students, teachers (task and contest creators) and administrators. All activities are performed using web browsers.

In particular, teacher can use the system for automatic check of the correctness of the solutions of home assignments or exercises performed in the classroom. Each student's submission can be viewed on-line by the teacher, so he can respond quickly to a variety of problems arising. If the problem description requires advanced texts or drawings, which cannot be done using HTML, it is possible to display the contents of a PDF file attached to the problem. Teacher can use this way, to give a mathematical definition of the function to be implemented in a programming language.

ZawodyWeb system provides section where students can ask the teacher questions related to the problems. Questions are sent to the teacher via e-mail and he can answer directly or post comments visible by all students in the proper section of the service. The teacher can block visibility of the student's ranking therefore students do not see how many problems their colleagues have solved. There is also possibility to block the rankings for last few minutes, to motivate students to compete and try to solve more problems than their colleagues.

System *ZawodyWeb* contains mechanism that allows the teacher to download the students' solutions that are visible in the ranking. This helps to catch solutions that are copied from other sources or dependent.

Among useful features there is an ability to block submissions or hide the problems for computers outside defined IP range, e.g. from outside the computer lab. This feature is especially used during tests or exams.

Within the system there is a dedicated class that allows performing operations that are not permitted in programming competitions. Among operations that are not tested are: parsing parameters passed to the program, error codes for the program, custom headers, opening and reading or writing files.

The *ZawodyWeb* system has ability to provide additional parameters to compiler which is not widely used in other systems used to run programming contests. Setting up additional parameters the teacher can force the compiler to check source code compatibility with the selected language standard.

There is also an easy way to add new languages and new functionalities. In order to do this one has to create a Java class that implements appropriate interface.

ON-LINE TESTING METHODOLOGY

To test student's knowledge, an appropriate problem set has to be prepared. For instance, when we present depth-first search (DFS) algorithm it is good to start with a task checking how to use stack data type. The implementation of DFS algorithm can be subject of next assignment. For more complex problems we can design whole chain of programming tasks leading to final one.

There are two main types of ranking used for automated contest systems:

- *Normal ranking* where points are summed up only for tests that were passed. This method will reward students that were not able to figure out optimal solution.
- *All or nothing* where only solutions that pass all tests defined for the task are rewarded.

While teaching introduction courses in programming it is better to use first type of ranking, because students are usually getting some points and they are not depressed when they provide solution which is correct but not optimal. A variant of this ranking is used by Polish Olympiad in Computer Science. On the other hand, *all or nothing* score method is popular at university contests such as ACM International Collegiate Programming Contest.

In order to define problem in the *ZawodyWeb* system teacher has to enter various information such as:

- task name and description of the problem,
- memory size limit for particular task, which means that solution submitted by student can use only up to this value of memory,
- source file size limit meaning that only source files with size less or equals to this limit can be submitted,
- what programming languages are available for the task,
- output comparator, which defines a way of comparing results.

To be sure that the service will check solutions by testing correctness and efficiency instructor has to prepare a set of tests for each problem. He should carefully analyze the problem and any possible mistakes students may make. Then tests have to be prepared in such a way, that only correct and efficient solutions will pass all of them. A good practice is that first tests will check basic correctness using small size data. Next tests should have increasing size ending up with a largest test which usually checks if the solution is the optimal one and finishes in given time. Tests should also check boundary conditions of the problem. In addition tests should be difficult to guess.

Every test should be composed of:

- input data treated as a standard input for submitted solutions,
- output data used by comparison mechanisms to check results,

- time limit setting maximal period of time program can run. After that time program is being killed and treated as failed,
- number of points added after particular test is passed correctly.

Tests preparation is the most time-consuming part of setting assignments in on-line system and has to be done carefully to avoid errors.

For example, if input data contains only one integer number of defined range (namely between 0 and 20), students can avoid solving the problem correctly. If input value does not match the value coded in solution, program will be stopped and will get wrong answer (WA) result. When the input value matches, the program will start an infinite loop. Solution will be terminated by the system and result will be TLE (Time Limit Exceeded). Using described method students may easily detect input data, calculate results using brute-force algorithms and implement a program which will print a hardcoded answer.

Another example is where answer has only few values (e.g. yes or no). In such case students can easily implement a program printing always the same result. With few attempts they determine value which gives highest score. To avoid this, the same number of tests for every possible answer has to be prepared.

One way of overcoming above problems is preparation of collective tests. They contain cases with different data which have to be read and calculated one by one. Such test will pass only if there is correct answer for all cases.

RESULTS

ZawodyWeb in regular curriculum

Although *ZawodyWEB* system was designed to support competitive programming, it is successfully used in the teaching process for computer science students. Most of the programming courses at Faculty of Mathematics and Computer Science N. Copernicus University utilize it as an important component of curriculum.

In the courses *Introduction to Programming* (1st semester), *C/C++ programming* (2nd semester) and *Java programming* (3rd semester) the teacher can prepare tasks covering particular features of language and restrict the language of submitted solutions.

Algorithms and Data Structures is another course where *ZawodyWeb* are used. It has focus on the algorithm efficiency and time used for running programs. Thus students have to implement efficient algorithms to avoid the Time Limit Exceeded error (TLE).

A good example of usage of on-line tools is *C/C++ programming* course for part time (extramural) students. Starting from the school year 2011/12 the course is divided into two parts: (i) e-learning and (ii) lectures and laboratories. The first five lectures on the C/C++ programming basics such as elementary data types, formatted input and output, conditional instructions, loops, etc. are organised in the form of e-learning. Students use materials hosted on the Moodle lecture management system getting theoretical information followed by a short quiz. Then the knowledge is used in practice by solving the tasks in *ZawodyWEB* system. During the e-learning stage students can interact with the teacher using the question interface available in the system, the forum available through Moodle platform or by e-mail.

After the e-learning phase students are obliged to pass the on-line evaluation test (quiz) which consists of several questions taken randomly from the set prepared by the teacher. The test is created in the Moodle system which covers random choice of questions and the evaluation of the students' answers. Students are allowed to take the evaluation test multiple times. Positive pass is necessary condition to attend the lectures and laboratory classes.

The laboratory classes focus on practical programming. For each meeting there are two sets of assignments available in *ZawodyWeb* system. The tasks from the first set are solved in the classroom under the guidance of the teacher. They allow to practice material introduced during the lecture. The second set is left as a home

assignment to consolidate the material learned in the classroom. In addition, there are several tasks with higher difficulty and short availability time, which allow to gain some bonus points for the practical part of the final exam.

To positively complete the laboratory class students have to solve most of the tasks published in the *ZawodyWEB* system, pass the practical programming test and implement some object-oriented software using C++ language.

The test consists of several tasks placed in *ZawodyWeb* system that have to be solved in the classroom under instructor supervision within limited time. The difficulty of the tasks is similar to those left as homework. The final exam is composed of two parts: theoretical and practical one affecting the final mark with the balance 40% and 60% respectively.

The theoretical part checks students' knowledge of C and C++ programming and is organized in similar way as the mid-semester evaluation test. The practical part consists of several tasks to be solved using *ZawodyWeb* system. The biggest advantage of such exam structure is that results are available immediately after exam. Students do not have to wait until the teacher manually evaluates the solutions.

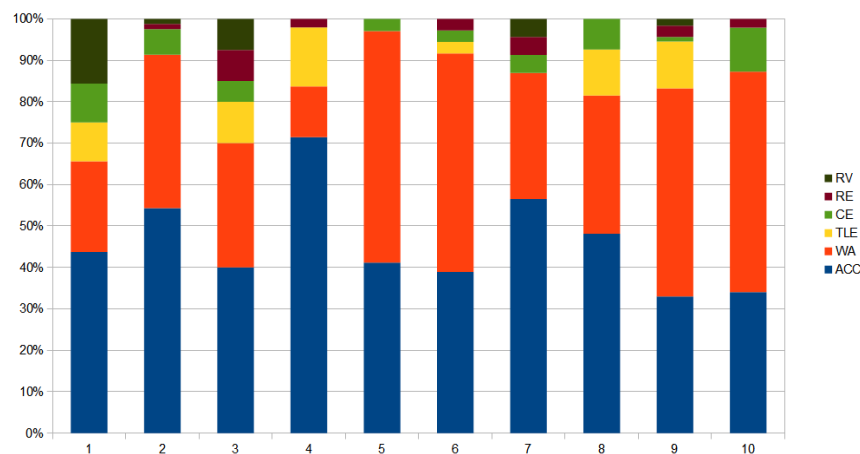


Figure 2: Statistics of submissions results during 1st semester of programming course. Average acceptance of submitted solutions is 41%.

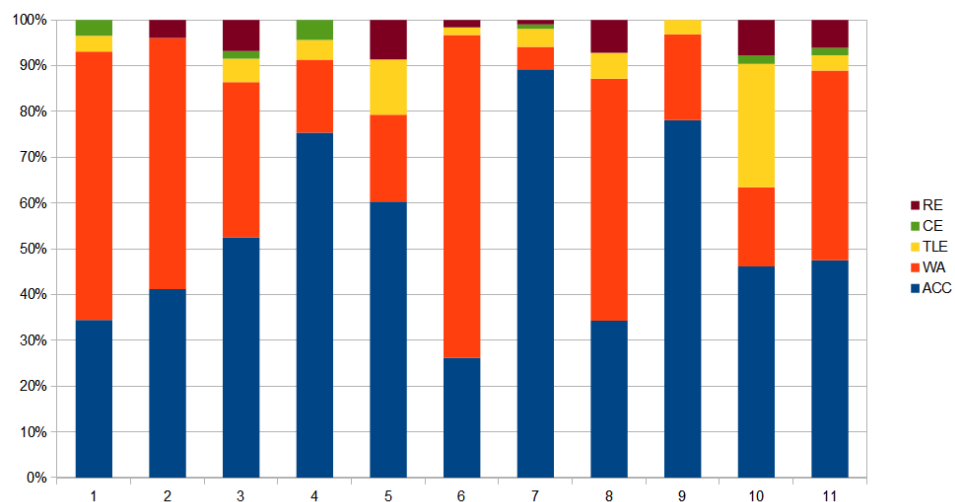


Figure 3: Statistics of submissions results during 2nd semester of programming course. Average acceptance of submitted solutions is 54%.

Working for 3 semesters with one group of students, we observed increasing ratio of correct solutions every next semester. Students learned to analyze their programs before submission to the system which lowered number of tries before their

solution was passing all tests. Figure 2 presents results of submissions for 10 series of tasks. Every colour means different submission result which may be one of following:

- accepted (ACC) meaning that solution passed all tests,
- wrong answer (WA) meaning that program generated wrong output data,
- time limit exceeded (TLE) - program was working too long,
- compile error (CE) - source file could not be compiled,
- runtime error (RE) - program unexpectedly interrupted its execution,
- rule violation (RV) meaning that program tries to use not allowed function.

In the first semester of introduction to programming course, average number of accepted submissions was at level of 41%. In the second semester this value increased to 54% (Figure 3) and finally in third one reached value of 67% (Figure 4).

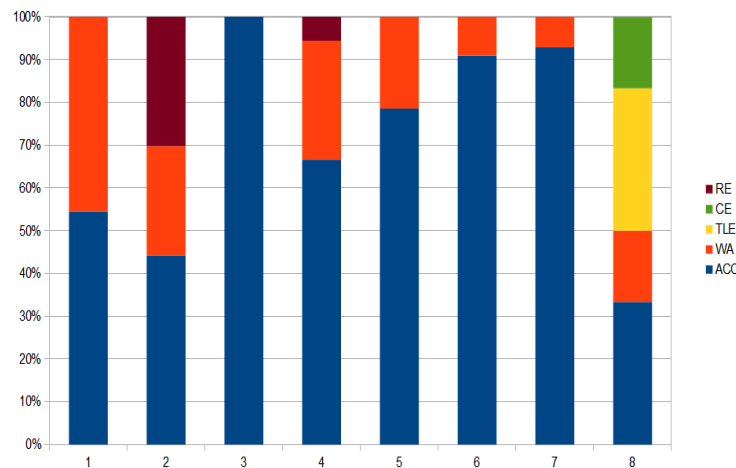


Figure 4: Statistics of submissions results during 3rd semester of programming course. Average acceptance of submitted solutions is 57%.

ZawodyWeb in student preparation for programming contests

As presented before, there are many individual and collegiate programming competitions available. The examples of national competition are: *Akademickie Mistrzostwa Polski w Programowaniu Zespołowym* (collegiate) and *Potyczki Algorytmiczne* (individual). To be able to compete in such competitions students have to be taught and trained in very special way.

To help students with this task, we provide them with two facultative (non-obligatory) courses: *Preparation for the competitions* (Przygotowanie do olimpiad) and *Seminar on competitive algorithmic* (Konwersatorium algorytmiki turniejowej). The first one is devoted to fundamentals of competitive programming. It consists of 60 meetings in computer laboratory. The course starts with the systematization and consolidation of the knowledge acquired by the students during other courses. Later on, the more advanced and complex problems are presented. During the meetings in a classroom, after the instructor presentation, students solve the set of tasks available in *ZawodyWeb* system. Each meeting has the related set of homework tasks with the increasing difficulty. Some meetings are organised as mini contests, where students have to solve within the limited time as many task as the can. Such contests allow to observe the progress of students' competitive programming skills.

Students can compare their results in the online ranking presented in *ZawodyWeb* system. The position in ranking is affected by the number of task solved and the total time used. The course ranking and contests introduce some competition among students which is the positive motivation to work and learn more. Although students work and solve problems individually, the seeds of future teams are formed.

The *Seminar on competitive algorithmic* is a natural continuation of the *Preparation for the competitions*. It helps to select the representation to national and international contests. The course consists of 30 lectures and 30 laboratory classes. Lectures include introduction to competitive strategy and to analysis of the methods to solve algorithmic problems. Laboratory classes are dedicated to solve problems appearing in the contests currently taking place. In this case mini contests are organized but students participate in them as teams. In addition to the lectures and laboratory classes, students (namely teams) organise additional meetings on their own.

The key to success of a team is proper selection of its members. Besides great algorithmic and programming skills the interpersonal relationships in the team are very important. Prospective candidates are offered special laboratory groups, where the difficulty of material presented and problems solved is higher than in other groups. Students write programs solving sets of tasks prepared in *ZawodyWeb* system. Moreover, at the end of the semester a mini contest is organized which allows to gain some bonus points included in the final exam.

From 14 students that started extended course, 10 was attending the second semester and 7 the third. Moreover, 11 of them took an optional course *Preparation to the competitions* and 9 have chosen *Seminar on competitive algorithmic* as their lecture of choice. Most of them were successfully representing the university in the international programming competitions. Figure 5 presents the results achieved by individual students: we draw the solution performance in subsequent semesters measured by total number of submissions that lead to positive results divided by the number of solved problems. The number of problems was 17, 34.5 and 16, in subsequent semesters respectively. We observe rising trend confirming increase of programming skills. In four cases the performance increased 1.78 - 1.88 times. In two other cases the performance improved by a factor of 1.46 and 1.11. Only in one case the performance fell down.

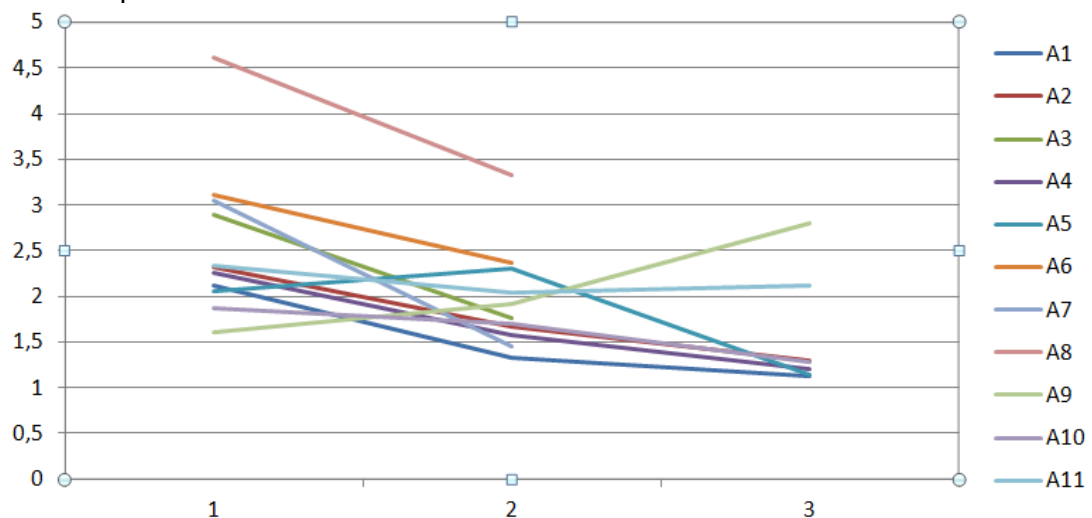


Figure 5: Statistics of individual submissions per assignment during three subsequent semesters. Lines present data for different students (A1-A11).

CONCLUSION

Benefits of using contests hosting service are obvious; however, such systems have to be used with a care. Students may check their solutions at any time. They can work at home correcting trivial errors based on the feedback from the service. In result the teacher will get code that works and can be run. He can focus on the student's solution instead of looking for trivial errors such as syntax or missing characters. The published rankings are good motivation for finalizing assignments on time and stimulate increase of programming skills.

On-line testing of programming tasks requires from the teacher special attention while assignments and tests are prepared to minimize probability of improper solutions.

Despite known disadvantages we observe, the careful use of automated on-line programming contests systems can be recommended in education of computer science students.

REFERENCES

- Bała P., Kwiatkowska A. B., Roguska J., and Wojdyło B. Toruńskie konkursy informatyczne. In *Informatyka w Szkole, XIX*. Oficyna Wydawnicza Zachodniopomorskiego Centrum Edukacyjnego, 2003.
- Codility Platform. <http://codility.com> (accessed 15 January 2013)
- Diks K, Madey J. From top coders to top IT professionals. In Mittermeir R. T. and Sysło M. M. (editors), *Informatics Education - Supporting Computational Thinking, Proceedings of ISSEP'08*, Lecture Notes in Computer Science 5090, pages 31-40. Springer, 2008.
- Nowicki M., Kluszczyński R., and Bała P. Internetowe narzędzia do sprawdzania zadań programistycznych i ich zastosowania. In *Informatyka w Edukacji, V*. N. Copernicus University, 2008.
- UVa Online Judge. <http://uva.onlinejudge.org> (accessed 15 January 2013)

Biographies



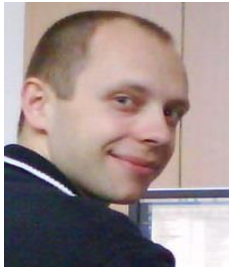
Marek Nowicki is a PhD student at the Nicolaus Copernicus University (Toruń, Poland) since 2010, majoring in Computer Science, especially in parallel and distributed calculations using Java language. He is main developer of ZawodyWeb system. Additionally, he takes part in translating Scratch, developed by the Lifelong Kindergarten Group at the MIT Media Lab, into a Polish language.



Łukasz Mikulski – computer scientist working in the areas of concurrency and algorithms. Academic teacher preparing students to participate in the national and international programming contests. He prepared on-line modules and programming workshop for secondary school students.



Marcin Płatkowski – computer scientist working in the area of formal languages and concurrency. Academic teacher preparing students to participate in the national and international programming contests. He prepared on-line modules and programming workshop for secondary school students.



Rafał Kluszczyński is a computer scientist at the Interdisciplinary Centre for Mathematical and Computational Modelling UW. He is currently involved in "PL-Grid Plus" project. Most of his teaching experience he has gained at N. Copernicus University where he taught students programming and trained them in programming. He prepared on-line modules and programming workshop for secondary school students.



Piotr Bała, physicist and computer scientist – academic teacher, instructor at courses for teachers and school students, organizer of the competitions and teaching activities in programming for children. Active researcher in the area of new technologies such as high performance computing and web technologies. Has extended experience in parallel and distributed computing. Recipient of numerous national and international research grants in the area of ICT.

Copyright

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/3.0/>