Annales UMCS Informatica AI IX, 1 (2009)
213–224; DOI: 10.2478/v10065-009-0017-9

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI
http://www.annales.umcs.lublin.pl/

Contests Hosting Service as a tool to teach programming

Rafał Kluszczyński, Łukasz Mikulski, Marek Nowicki and Piotr Bała*

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

Abstract

Computer science would not exist without the concept of algorithm. Therefore design of algorithms plays an important role in education while implementation is usually considered to be straightforward. Increasing variety of programming languages, wealth of possible constructions, programming environments and tools makes programming difficult for the beginners.

Apart from the idea of problem solution, it is important to teach programming skills. Size of classes of 10-20 pupils and a limited number of lessons and their short time are the major problem. The teacher has to check solution of every pupil, compile it and run tests. This is definitely a time-consuming process which makes teaching difficult. In this paper the authors present the use of problem solutions validation systems during classes. With the help of such a system called *ZawodyWEB*, the authors teach algorithms and programming for the secondary school students.

1. Introduction

One of the major goals of computer science education is to teach basic algorithm concepts with the skills to implement them in any programming language. Unfortunately, in Poland this is not an easy task to do. Lately reformed Polish

E-mail address: {klusi, frodo, faramir, bala}@mat.umk.pl

education system provides children with very few lessons devoted to problem solving with algorithms. The main effort is to get children familiar with the information technology. Pupils learn how to use computers and make computer applications. Lessons intended for teaching algorithms are sometimes treated as of secondary importance. One of the reasons may be the one reformed in paper [**1**] that teachers of computer science as a second specialization, are not fully knowledgeable in it. On the other hand, teachers with the computer science background sometimes present very abstract and theoretical approach.

Despite this, Polish pupils and students are the best in the world in various programming contests [**2**]. That is why local programming contests are becoming more and more popular. There is also an increasing number of web services enabling access to algorithmic problems with ability for remote validation of implemented solutions. Such systems are used in many contests like:

- Polish Olympiad in computer science [**3**],
- Nicolaus Copernicus University Programming Contests [**4**],
- Polish Academic Championship in Team Programming [**5**],
- Poznań Open Team Programming Championship [**6**],
- Polish Internet Championship in Programming [**7**].

Besides obvious applications of such tools like contests and championships, the authors want to point to the use of such systems for teaching programming at the level of senior secondary school (17-19 years old) and junior secondary school (children 14-16 years old). In particular, the authors have used successfully the contests hosting system in teaching algorithms as well as programming. This was performed during CS classes for children organize in the Faculty of Mathematics and Computer Science, Nicolaus Copernicus University. Pupils attended lectures and classes prepared by academic teachers. As one of the tools the contest service called ZawodyWEB has been used.

Teaching programming is very difficult. It is strongly related to teaching of abstract thinking. Pupils have to understand the problem with the included example and then design the program that will work on different data sets of not known size. Of course, there is almost always upper limit given, but they have no information about size of tests used during validation of programs. The teacher should have at least a few tests to check pupils' solutions against some algorithm special cases. If there are twenty pupils in a class, there is not enough time to check every solution carefully during the lessons. That is why teaching programming is a very hard job without help of additional tools. Pupils, who finished their work at home, not always have correct programs. They usually write them using the cases they know, not always foreseeing every possibility. That is why the authors propose the use of contests hosting service. The service

enables testing pupils' solutions against the earlier prepared tests. Teachers have access to their codes to see how the programs are written and possibly notice common mistakes.

2. ZawodyWEB Service

ZawodyWEB platform has been created in the Faculty of Mathematics and Computer Science at Nicolaus Copernicus University [**4, 8**] for Programming Contest organized here. Current version of the platform uses J2EE technology while the previous one used PHP language and Apache HTTP Server.

Platform enables to register users with different roles. Each role has different permissions. We can divide users into four groups: We have used Maven, which can automatically download dependencies (third party libraries) from a world-wide accessible repository. Apache/Tomcat 5.5, well known servlet container, is used for deploying and hosting platform. To store competitions, problems, solutions and other data, we use the MySQL database server in 5.0 version together with JDBC Driver. To dynamically generate web pages we have used JSP (Java Server Pages) technology. All user data are sent through the web browser to servlets which perform specified actions (e.g. process user submission of solution) and then show results. To change fragments of web page ZawodyWEB system use AJAX technology. Linux (Fedora Distribution) is a host operating system.

- competitors - default role,
- testers - users who can rank solutions,
- problem authors,
- administrators.

Competitors can submit solutions and watch the results. Testers can manually judge competitors' solutions. The authors are able to create new problems and add tests to them. Administrators can create competitions, series, languages and do all the things that authors, testers and competitors can do.

Competition consists of a set of series, which are the set of problems. Additionally, series store information about start and end dates, date of freezing and unfreezing ranking. First of them is a date, when the problems belonging to the series, start to be visible for competitors. End date is the date, when competitors can no longer send their solutions for problems belonging to the series. During the time between freezing and unfreezing date, ranking is no longer updated. Competitors can view points for their solutions but it is not included in the ranking.

Each problem has the information about name, abbreviation, memory limit, possible languages to use in solution (presented to contestants, see Fig. 1) and

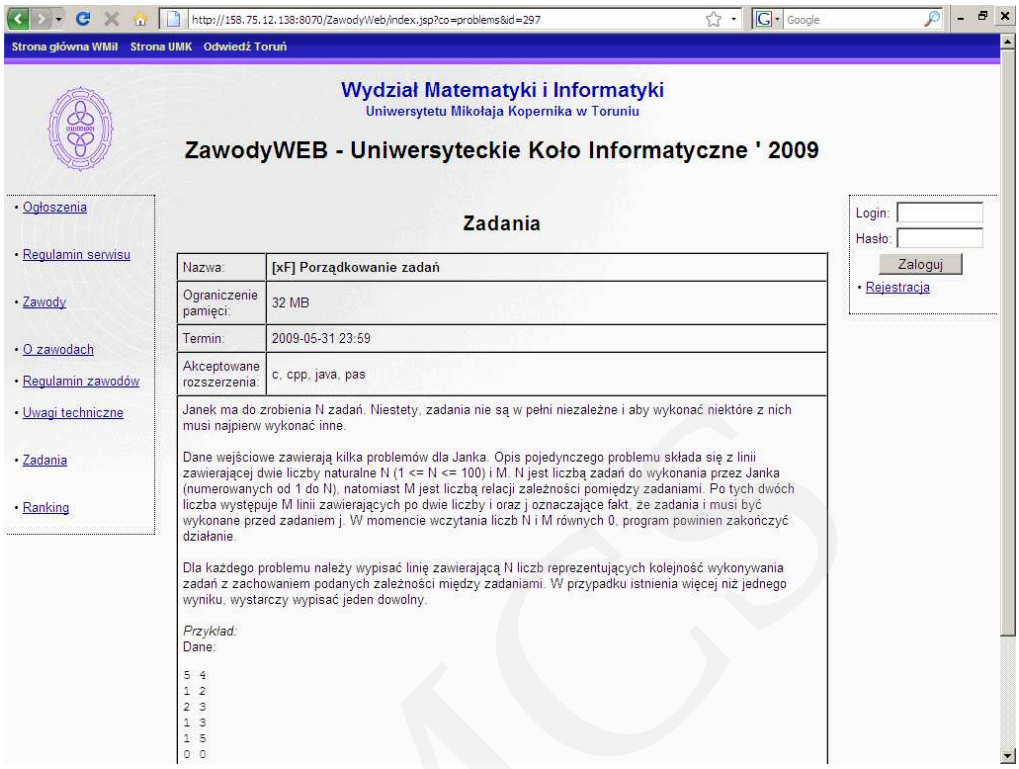

Fig. 1. Accessing ZawodyWEB service through web browser. Problem description (in Polish) with defined memory limit, end date of submissions and programming languages is presented.

"differer", which compares the solution output with that stored in the database. Tests are assigned to the problem and consist of input and output data, time limit and a number of points that solution can achieve for solving the test (Fig. 2). Language is recognized by the extension of solution filename. Different languages require specific compilation and run methods (e.g. to run solution written in the Java language we must use Java Virtual Machine but the solution written in the C++ language compiles to machine code and can be run like any other operating system command). There can exist more than one language with the same extension (e.g. with different compilation flags). However, a problem cannot have more than one language assigned with the same extension.

As mentioned earlier the user solutions can be checked manually or automatically. Manual check needs human interaction to rate the user solution. Automatic check is faster and easier to use, but more complex in architecture.

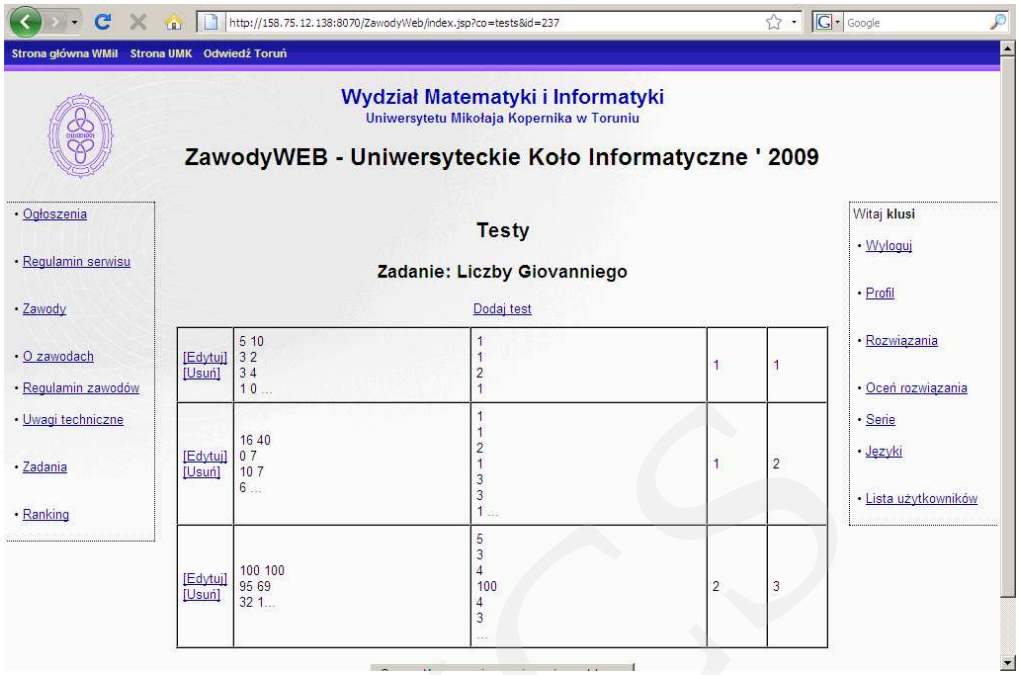

Fig. 2. ZaowdyWEB system allows to specify exact input and output data. One can see three tests with specified time limit (in seconds) and point values.

ZawodyWEB service has to catch compilation error, forces termination of solution when it reaches time limit or tries to allocate more memory than limit and prevents from leaking input data. Additionally, automatic testing should work fast and reliably, to show problem results to the user almost immediately. Process of automatic checking consists of phases:

- after upload - things that have to be done immediately after solution is sent,
- compilation of solution,
- running program on the set of various tests,
- check the output data of program.

If the solution is uploaded as a compressed archive the unpacking program is invoked. After that JudgeManager is informed about the submitted solution. JudgeManager is a process that has a list of all running Judge processes which perform solutions check. Manager checks the list and if there is Judge process available, it sends solution to check. It is clear, that in such a way, there can be used more then one Judge process, which is especially useful during short contests with many submissions. This solution allows to use efficiently multicore systems as well as distributed ones e.g. clusters. Judge gets the solution from

the database and tries to compile it. If there are no compilation errors, solution is checked against the tests stored in the database. Each input data is passed as a standard input to the program and the results printed to the standard output are saved. When the solution returns error code 0, "differer" compares the solution output with that stored in the database. If outputs are the same, solution gets points. Finally, the sum of points is saved in the database and Judge informs manager that he has finished and waits for another solution to check.

ZawodyWEB platform can handle two types of ranking:

- normal ranking - every test assigned to a problem is checked independently,
- ACM ranking based on the ICPC contest rule [**9**], meaning "all or nothing".

Solutions in the contests with normal ranking get points for every test. Different tests can have individual point values. Place in ranking depends on the sum of all points. In the ACM ranking solution can get 0 (incorrect solution, because of wrong answer or time limit exceeded etc.) or 1 (correct solution -– accepted) point. Place in the ranking depends on the number of correct solutions and in the case of equal point value, total time of solving solutions. Total time can be increased by defined penalty time for each incorrect solution, before solution gets accepted.

3. University Computer Science Course for High School Pupils

In the academic year 2007/2008 Computer Science Course for Secondary School pupils has been organized in the Faculty of Mathematics and Computer Science. The main goal was to replace computer science school courses organized only in a few leading secondary schools as the addition to normal classes. There were usually a few pupils really interested in programming or computer science attending these extra courses. In such small groups it was very difficult to support young enthusiasts by the teacher. Usual classes made it possible to gather pupils interested in computer science from different schools. Additionally, learning from more experienced colleagues and competing with them is much more motivating than any teachers' actions.

Classes have been organized once a week from November 2007 to the end of April 2008. Every class consisted of two parts, 45 minutes each. The intention of the first one was to present various aspects of computer science in the form of lecture. The second part was dedicated to practical programming in C++ or Java. Academic teachers have used university computer laboratories to ensure, that every pupil tries to write a program that solves the presented problem by

himself. To organize the programming classes in a much more efficient way, based on the questionnaires completed by the pupils during the first classes, there have been formed two groups:

- teaching programming, where problems do not require too much abstract thinking,
- understanding and programming algorithms, where it is assumed that pupils have basic knowledge about programming and can write at least simple programs.

It is clear that the second group of pupils can create a challenge even for an experienced teacher. Many of them are talented and exercises have to be much more stimulating than standard classes. The authors decided to use the presented here contests hosting service ZawodyWEB. Apart from standard explanation how algorithm works based on couple of examples, there was organized a programming league. Pupils could implement algorithms at home and submit them on-line to the contest system. Such a solution enabled them to use new knowledge in practice. The form of competition motivated pupils to compete with each other, even from the same school. The classes were quite popular and there were almost 200 registered pupils.

The first edition of the Computer Science course was a big success. That is why in the academic year 2008/2009 we continued classes. For the second edition, based on the feedback, we decided to separate theoretical and practical part. In the first three months, from the beginning of October to December 2008, we organized lectures. Sometimes there were 2 separate topics (45 minutes each), but usually doubled time enabled us to present various computer science topics in much more detailed way. Starting with the year 2009 we began a practical part which took place in computer laboratories. Again, doubled time means new possibility for teaching programming.

In the beginners group, we explained the concept of variables, basic statements and language constructions. Then pupils had possibility to try their skills in a more practical form using the contests hosting service. The problems to solve should not only enable to practice basic statements, but also teach steps needed for solving problems with the help of computers. Pupils wrote their programs, tested them using the examples discussed in the class, and then sent solutions to the contests hosting service. ZawodyWEB system checked them using different sets of input data and showed the results. Pupils were informed if solutions were wrong, partially good or very good. This of course depends on the tests prepared by the teacher.

In the advanced group, we changed a little bit concept of teaching. The problems which were discussed in class were presented as a number of less

complex tasks. This allowed students to concentrate on parts of the algorithm. When simple examples were solved they could be used as a part of more complex algorithm. A breadth-first search (BFS) can be an example. The simple task was to implement the queue data structure. Once students can implement the data structure representing graph they have got all what is needed to solve any problem using the BFS algorithm.

4. Contests Hosting Service

Utilization of contests hosting system in teaching is connected with particular didactic problems. First of all, an arbitrary contests hosting system is a new tool for both pupils and teachers. Before we start to use such a system there has to be at least one person that is able to administrate the service. Besides, every teacher should have the role of problem author. To prepare classes he has to know the basics of service activities. For instance, he has to be able to add new tasks and upload the tests.

The most important users of our service are undoubtedly pupils. During the first classes teacher shows them how to register in the contest solving system, find tasks, upload their solutions and check the system responses. It is also important to make pupils aware of the rules of the system. Especially, the formats of inputs and outputs are important. In the development of the system ZawodyWEB, we put pressure on making everything as much intuitively as possible. Our experiences showed that this was achieved.

Understanding algorithms and concepts of programming is connected with the abstract thinking and it is not easy to achieve satisfactory effects. When we want to make use of contests hosting service to support the educational process the most important thing is to prepare suitable and carefully thought out sets of tasks. The concept of spiral teaching, presented in [**10**], proposes to divide the tasks to levels. One starts with the set of basic subjects and tasks. In the second series we refer to simple problems and introduce more complicated ones. During the whole education we get back to the same topics many times, each return is connected with repetition and refilling the specified part of knowledge.

Such model needs years of lessons to be successfully implemented. Our activity is directed to the course that lasts one or two years. In this case, especially in the advanced group, it is better to compress the idea of spiral teaching, going closer to the concept used in Bebras Contest [**11**]. We can try to cover topics once, but make the tasks with a few levels of difficulty.

After proposing the good set of tasks one should put pressure on adequate choice of tests. This necessity concern every automatically reported contest [**2**]. Unlike the regular contests, our tests should not be binary. It is good,

when they foresee possible mistakes and give some scores for partially solved problems. Even poor pupils should be able to get some points. It can force them to improve the solution and gives them hope to achieve better score. On the other hand, the total score should be achieved only by good and efficient solutions. Adding the situations of good but not efficient solutions we see the need for difficulty gradation.

In the context of problem formulation we came across an unexpected problem. Some pupils are oriented to wards getting the score better than the others and try to cheat the system. Taking care of the normal security criteria are not enough for e-learning [**12**]. After each solution sending, pupils get full report of achieved scores, distributed to single test cases. When the expected result is from a small set, for instance is a single numeral or a true/false value, they write programs which print only one value, the one that achieves the best score. This method gives at least half of the total score for the true/false problems!

Sometimes situation is even worse. Pupils try to find a characterization of the test cases. By sending many solutions they find the dependencies between the input length and the expected result. Such approach can head for achieving the total score for the task.

Both cases can be easily detected by the teacher looking at the source code of solutions. Unfortunately this process is difficult to automate and has to be performed manually.

5. Semi–remote Teaching

The described system is a very nice tool that can be used to make interesting and efficient lessons. The first advantage is the organization of the classes. After short theoretical preparation pupils work with given tasks individually. It engages the whole group and gives the teacher time to look after single pupils. The system gives the teacher opportunity to read uploaded solutions just after pupils send them. He can verify the program using his own computer, without disturbing pupils. Depending on the kind of error, the teacher can give the whole group a hint or talk with the pupils face to face. This access gives opportunity to control individual and overall progress.

During this time, group works on the given tasks. Pupils get immediate responses to the quality of result. In the programming learning the immediately observed results of produced solution is very important aspect [**13**]. The proposed model goes even further, pupils test their solutions using some situations not described in the problem text and will not come to their mind. Such possibility encourages them to work on partially solved tasks. Program can work

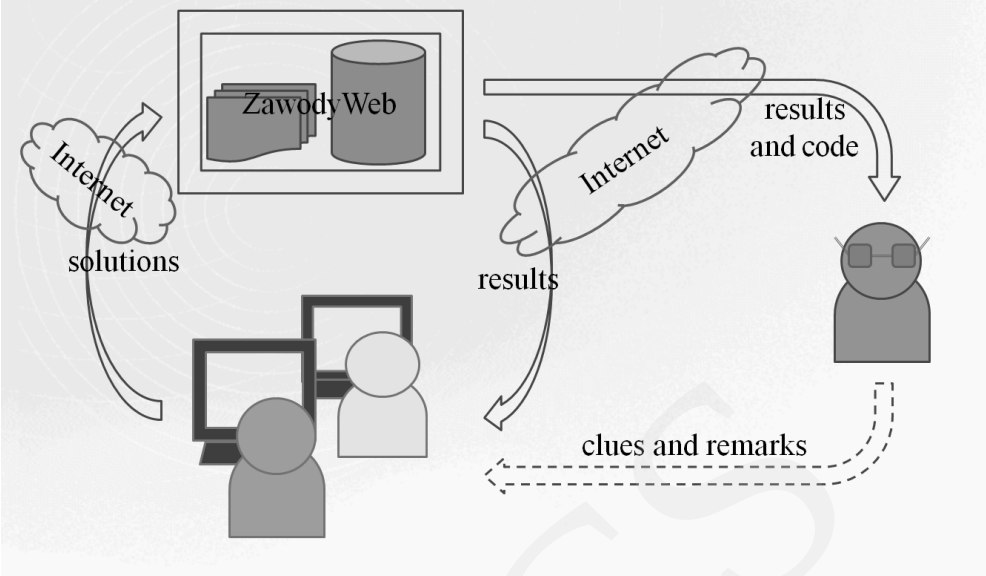

Fig. 3. Model of semi–remote teaching. The dashed line differentiates between two possible scenarios, lessons in a classroom and e-learning.

on simple tests but fails on more complicated ones. Therefore students are not satisfied with their first, sometimes not correct or not efficient, solutions.

The described scenario encourages us to propose a model of semi–remote teaching with use of contests hosting service. The schema of such process is presented in Fig. 3.

The main idea is to support traditional classes with the on-line tools used in the classroom and at home. The classes are used to present programming tasks and solve them partially in the classroom. Then the students are left with the problems to solve at home. The homework problems are either different versions of those discussed in the classroom or new ones. They can also return to the tasks they were solving in the classroom, download solutions they sent and continue working on the problem. For the teacher, it is an opportunity to use service to give the homework to pupils.

One should note that the guaranteed security (students do not see solutions sent by others) is enough to use the system as a platform for class tests.

The most significant disadvantage of the presented model is resignation from an interactivity in program writing. The solutions behave like black boxes. They only read a specific input and give back an expected output. Every encouragement like "Give an arbitrary positive number" written by the program is treated as an error. Pupils do not exercise the interaction with the user of their program. They do not care about giving precise and comprehensible

instruction. Furthermore, their programs work in the text environment. They do not give any possibilities of practising the graphical aspects of programming. The only topics covered by this model are algorithms and numerical tasks.

One should note that the contest system can be also used for the programs oriented towards the user interface implemented in web technology, namely PHP scripts, Java applets or JavaScript codes. Unfortunately, the evaluation of programs has to be done manually by the teacher and the contest system is used to store uploaded solutions and to organize grading process.

One should note that in all presented cases the teacher plays an important role. The contests hosting service is very useful tool, but only a tool. The code of final solution should be read and verified against dishonest solutions.

6. Conclusions

Based on the practical experience, the authors present the use of contests hosting services during computer science classes. The system helps then teacher grade fair pupils, simplifies the process and allows to individualize teaching. On the other hand, young people interested in programming may continue work at home. The proposed solution stimulates finding better solution when their idea does not get all possible points. With the help of such systems, classes become much more interesting. Additionally the student competition gives them more motivation to work.

References

[1] Kolczyk E., *Algorithm – Fundamental Concept in Preparing Informatics Teachers*, Informatics Education - Supporting Computational Thinking, Proceedings of ISSEP'08", Springer, Lecture Notes in Computer Science 5090 (2008), 265.

[2] Diks K and Madey J., *From Top Coders to Top IT Professionals*, Informatics Education - Supporting Computational Thinking, Proceedings of ISSEP'08, Springer,Lecture Notes in Computer Science, 5090 (2008), 31.

[3] Polish Olympiad in Informatics. http://www.oi.edu.pl.

[4] Nocolaus Copernicus University Programming Contest. http://www.mat.umk.pl/ki.

[5] Polish Academic Championship in Team Programming. http://amppz.cs.put.poznan.pl.

[6] Poznań Open Team Programming Championship. http://www.mwpz.poznan.pl.

[7] Polish Internet Championship in Programming. http://opss.assecobs.pl.

[8] Nowicki M., Kluszczyński R., Bała P., *Internetowe narzędzia do sprawdzania zadań programistycznych i ich zastosowania*, Informatyka w Edukacji, V, Nicolaus Copernicus University 2008.

[9] The ACM International Collegiate Programming Contest. http://cm2prod.baylor.edu.

[10] Yovcheva B. B., *Spiral Teaching of Programming to 10-11 Year-Old Pupils After Passed First Training (Based on the Language C++)*, Informatics Education - Supporting Computational Thinking, Proceedings of ISSEP'08, Springer, Lecture Notes in Computer Science 5090 (2008) 171.

[11] Dagiene V. and Futschek G., *Bebras International Contest on Informatics and Computer Literacy: Criteria for Good Tasks*, Informatics Education - Supporting Computational Thinking, Proceedings of ISSEP'08, Springer, Lecture Notes in Computer Science 5090 (2008), 19.

[12] Eibl Ch. J. and Schubert S. E., *Development of E-Learning Design Criteria with Secure Realization Concepts*, Learning to Live in the Knowledge Society, Proceedings, Springer, IFIP 281 (2008) 361.

[13] Salanci L., *Understanding Algorithms and Programming*, Informatics Education Contributing Across the Curriculum (2008) 10.