Teaching Computer Science in a Web-Based Environment

Sanja Maravić Ćisar*, Robert Pinter*, Petar Čistar** and Dragica Radosav ***

* Subotica Tech-College of Applied Sciences, Subotica, Serbia
** Academy of Criminalistic and Police Studies, Belgrade-Zemun, Serbia
*** University of Novi Sad, Technical faculty "Mihajlo Pupin", Zrenjanin, Serbia

sanjam@vts.su.ac.rs, probi@vts.su.ac.rs, petar.cisar@kpa.edu.rs, radosav@tf.zr.ac.rs

Abstract—Programming is a major subject in Computer Science (CS) departments. However, students often face difficulties on the basic programming courses due to several factors that cause these difficulties. Maybe the most important reason is the lack of problem solving abilities that many students show.

This paper describes the web-based environment Pex4Fun from Microsoft Research for teaching computer science. Pex4Fun can be used to teach and learn computer programming at many levels, from high school all the way through graduate courses. Pex4Fun was experimentally used for learning Visual Basic with college students, the results of which are presented in this paper.

I. INTRODUCTION

Programming is a complex mental activity that is defined as an abstract process. Understanding and visualizing abstract processes poses a considerable problem for students when learning programming, as well as other fields with similar characteristics.

Linn and Dalbey define an ideal chain of the learning process of learning programming and suggest it as a standard for comparing programming teaching methods [1, 2]. The three links of the chain are:

- characteristics of the programming language – in order to log the programming solution of the problem with the given language, the student needs to understand the syntax, semantics, and expressive possibilities of the language.
- the skill of forming the program is the knowledge to use a bundle of techniques which, applied and combined, are used to solve the given problem. The skill is based on the knowledge of stereotypical code samples which combine different characteristics of the language. The models implement complex functions, such as sorting, finding the lowest common denominator of two numbers, counting words in a given text, etc. Programmers design the language and model characteristics to be combined, problems to be decomposed into parts, solving every part independently, then linking the partial solutions into a unique unit – the program. At the end of the program writing process its correctness is checked by testing.
- the general problem solving skills come to light during learning new formal systems and this is set as the goal to be achieved by studying programming. The same models and procedural skills are common to many, or even all formal systems. Therefore, this approach is used to learn models of logging with one formal system and the rules of transfer to the new one, the subject of the learning process, so it results in mental learning and activation of pre-existing knowledge.

Lemut et al. [3] explain the difficulty of learning programming by the need for implementing complex activities that have to be mastered even by beginners simultaneously. For example, the program is tested by executing it, using carefully selected input marginal values that will result in checking all program paths. The choice of the marginal values requires the knowledge of semantic program instructions. Opposed to this, the beginner programmer learns instructions, so they find it very difficult to choose such inputs on their own. Du Boulay [4] finds that the sources of difficulties are the following:

(1) Orientation – the general idea of students about programming and the program.
(2) Abstract engine – understanding the computing model that defines the program language.
(3) Notation - syntax and semantics of the language.
(4) Structures – knowledge of programming constructions as a composition of instructions with which certain program requirements are met.
(5) Pragmatics – skills implemented in creating the correct program (planning, decomposition, coding, testing, detecting and fixing errors).

The organization of the paper is as follows. Section II and III briefly describes Pex as a unit testing tool. In Section IV the opinion of our students is presented and also experimentally results of using Pex4Fun for a programming course. In the conclusion, the outline the future work in this direction and expected results are given.

II. PEX: UNIT TESTING TOOL FOR .NET

Pex is an automatic white-box test generation tool for .NET, based on dynamic symbolic execution [5,6]. This tool is integrated into Microsoft Visual Studio in the form of an add-in. It can generate test inputs which are combined with different unit testing frameworks [5,6].

One of the most important methodologies that Pex supports is called parameterized unit testing [5,6]. There are useful characteristics that Pex offers to support for testing. Primarily, there is the option of exploring code and suggesting the tests that should be done. Secondly, assuming that it is a parameterized test, Pex can determine the combination of parameters that has to be tested so as
to provide all feasible versions. Lastly, once Code Contracts is being used, Pex uses that information to fine-tune the unit tests that are offered or generated for the user [7].

III. PEX4FUN

Pex for fun on the web is a fundamentally simplified form of the fully featured Pex Power Tool for Visual Studio. There is no need for any installation; since it is handled in the cloud [10]. Code can either be written in C#, Visual Basic, or F#. Figure 1 shows the user interface of the Pex4Fun web [8].

A. Solve Puzzles

Pex4Fun has a given set of particular code examples, which are called puzzles; these are displayed in the working area for the players. Every puzzle is focused on a major method named Puzzle. When a puzzle is loaded in the working area, the user will click the “Ask Pex!” button so as to compile and run it. The compilation and execution takes place on the Pex4Fun server; only the testing results are displayed. The main Puzzle method can take parameters and return values. If one wants to run one of these Puzzle methods, argument values have to be provided. Pex automatically detects interesting argument values as it analyzes the code. A table of input and output values then shows the generated input argument values and produced return values under the working area. The player can click every row of the table for further details, e.g. console output or stack traces [5,6,9].

B. Solve coding duels

A coding duel is an interactive puzzle. In a coding duel, the idea is to apply the Puzzle method to recreate the same behavior as another secret Puzzle method (e.g., the teacher’s specification) [5,6,9].

Figure 1. The user interface of the Pex4Fun web [8]

Figure 2. A coding duel [10]
As far as learning and teaching are concerned, such coding duels serve the purpose of helping them to train different skill sets of players. These include the following, among others:

- Abstraction skills. The shown list of generated input argument values is there to exhibit various behaviors and identical behaviors, respectively, though these are just exemplary argument values, which means that these are not a complete set of argument values for exhibiting different or same behaviors. Before realizing how to alter the player’s implementation to move closer to the secret implementation, the player is forced to generalize from the seen exemplary values and the same or different [5,6,9].

- Problem solving or debugging skills. In order to solve a coding duel the player needs to run iterations of trials and errors. The player has to decompose the problem on the basis of the observed exemplary argument values and behaviors: grouping exemplary arguments that may show the same category of different behaviors, e.g., because of lacking a branch with the conditional of if (i>0). As a following step the player has to think of a hypothesized missing or corrected piece of code to cause failing tests (different-behavior-exposing tests) to pass as well as passing tests (same-behavior-exposing tests) to still pass. Following this, the player has to do a test to validate the hypothesis by clicking “Ask Pex!” Thus, solving a non-trivial coding duel may require exercising different problem solving skills [5,6,9].

- Program comprehension and programming skills. Assuming that the dummy implementation at the beginning is not that “simplest”, including non-trivial code, the player has to first comprehend what actions the dummy implementation is performing. This makes it clear that the players must have solid programming skills in order to do well on a non-trivial coding duel [5,6,9].

C. Create and teach a course

The purposes of Pex4Fun are manifold: it can be used to make classes on mathematics, algorithms, programming languages, or problem solving in general seem more captivating [5,6,9]. Teachers have at their disposal an embedded wiki to create class materials based on puzzles and coding duels. More specifically, this enables the teacher to integrate existing pages into the course. The author of these pages could either be the given teacher or the dummy implementation is performing. This makes it clear that the players must have solid programming skills in order to do well on a non-trivial coding duel [5,6,9].

D. Creating and publishing coding duels

There are five steps necessary to create and publish coding duels [5,6,9]. The first step is to sign in, so as for Pex4Fun to maintain coding duels for you. The second step is to write a specification setting out from a puzzle template where the specification is written as a Puzzle method which transforms inputs into output. The third step is creating the coding duel by clicking the button “Turn This Puzzle Into a Coding Duel” (which appears after clicking “Ask Pex!”). The fourth step is editing the visible program text by clicking the coding duel Permalink URL, which leads to the coding duel. You fill in a somewhat more useful outline of the implementation (as well as adding optional comments) which somebody else will at some point complete [5,6,9].

The fifth step is to publish once you have finished the editing process of the visible Puzzle method text, then you click “Publish”.

IV. EXPERIMENT ON USING PEX4FUN

A survey was carried out among the undergraduate students during the Visual Programming course in the 2012/13 academic year. The total of 83 students were included in the survey, out of which 79.52% were male and 20.48% were female students. The course lasts for 15 weeks. The students attended two different groups based on the language of instruction, which was Hungarian or Serbian. The experimental group consisted of 30 students and the control group of 53. The aim of this survey was to establish to what degree Pex4Fun makes studying of Visual Basic easier for students.

Traditional lectures were held for both groups, and all students had the lab practices in the PC laboratory. The students of both groups used Microsoft Visual Studio for coding and solving practical problems, however, the students of the experimental group also used Pex4Fun during the lab exercises.

Coding duels from Pex4Fun were used for mastering and developing the students’ problem-solving and program-understanding skills. In a coding duel, students are required to write a missing code in the Puzzle method so as to obtain the exact same behavior as another secret Puzzle method, which is not revealed to the player [5,6,9]. Finding the solution for a coding duel needs the student to decompose the problem, to put hypothesis (to correct some part of code or to write down missing part of code), and finally, to run an experiment by pressing “Ask Pex!”. The student is given feedback from Pex4Fun in the form of a table, marked by red and green circles which serves as indication as to whether the student had or had not passed the test for corresponding input value [9].

![Figure 3. Creating and playing a coding duel [9]](image-url)
For the purpose of examining the advantages and disadvantages of learning with Pex4Fun the five exercises were designed. The students’ task was to figure out what the program does and to create appropriate sample values. The MCQ test was given to the students of the both groups. The total number of students that participated in the test was 67 out of 83 (16 students did not take the test). The results of the test are presented in Table I.

### Table I.
<table>
<thead>
<tr>
<th>Group</th>
<th>Number of students</th>
<th>Pass</th>
<th>Failed</th>
<th>Passing rate [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>control</td>
<td>42</td>
<td>18</td>
<td>24</td>
<td>42.85</td>
</tr>
<tr>
<td>experimental</td>
<td>25</td>
<td>12</td>
<td>13</td>
<td>48</td>
</tr>
</tbody>
</table>

As seen from the table, the passing rate of the test for the control group is 42.85% and for the experimental group it is 48%.

Questionnaires are common research practice and measure the opinion and attitude about the subject of the research. The questionnaire consisted of 5 statements about Pex4Fun. The Likert scale was used, with the answers ranging from 1 meaning “I totally disagree with the statement”, to 5 meaning “I totally agree with the statement”. Table II shows the students’ opinion from the experimental group about Pex4Fun. Mean values, standard deviation and variance values are shown.

### Table II.
<table>
<thead>
<tr>
<th>statement</th>
<th>mean</th>
<th>stdev</th>
<th>var</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pex4Fun was easy to use.</td>
<td>3.44</td>
<td>0.82</td>
<td>0.67</td>
</tr>
<tr>
<td>Pex4Fun was interesting for usage.</td>
<td>3</td>
<td>0.99</td>
<td>0.92</td>
</tr>
<tr>
<td>Pex4Fun motivating me to work.</td>
<td>3.68</td>
<td>0.85</td>
<td>0.73</td>
</tr>
<tr>
<td>Pex4Fun was a positive experience for me.</td>
<td>3.88</td>
<td>0.78</td>
<td>0.61</td>
</tr>
<tr>
<td>Pex4Fun is suitable for learning programming.</td>
<td>3.24</td>
<td>1.052</td>
<td>1.11</td>
</tr>
</tbody>
</table>

As can be seen from the table, students have a positive attitude about the Pex4Fun tool. In total 4 values are above 3.5, only one value is 3 which represents the state “undecided”, making us believe that in general opinion of and attitude to Pex4Fun is positive. It is interesting to see that the two highest grades were given to statements expressing motivation to work and the positive experience for students.

V. CONCLUSION

Computer programming has been usually introduced using programming languages that students find hard to use, and the proposed activities rarely resonated with learners or their life activities and experiences. It is the lecturer’s task to detect and introduce students to novel programming tools and activities, so that students will find computer programming more straight forward. Pex4Fun was experimentally used in learning Visual Basic and presented in this paper. The initial positive results are an encouraging introduction to the use of this program, leading to future experiments regarding Pex4Fun.

REFERENCES