Promoting success in beginner programming students with Virtual Programming Lab

Marílio Cardoso
ISEP - Polytechnic of Porto
(Portugal)
joc@isep.ipp.pt

Rui Marques
ISEP - Polytechnic of Porto
(Portugal)
rfm@isep.ipp.pt

António Vieira de Castro
ISEP - Polytechnic of Porto
(Portugal)
avc@isep.ipp.pt

Abstract – **Teaching of computer programming remains as one of the biggest challenges for any teacher, particularly for those who have novice students. Usually, for these students, learn to code is not only challenging, but also a very hard and complex task. Teaching these subjects implies close monitoring for correction and overall orientation, towards correct assessment. So, all the help that teachers and students can have to achieve their goals is welcome. A tool that can be useful to this purpose is VPL (Virtual Programming Lab), a Moodle plug-in that allows students to submit their code and get instantaneous feedback without teacher's intervention. In order to test this concept, an experiment is currently been carried out with several classes of beginner programming students of the course unit of Algorithms and Programming (APROG) from the course of Informatics Engineering of the Informatics Engineering Department of School of Engineering (ISEP), Polytechnic Institute of Porto (P.PORTO). The students were challenged to test their assignments in VPL with a set of test values previously defined by teachers.**

Keywords - Automatic assessment, Java, Teaching programming, Virtual Programming Lab.

INTRODUCTION

Software development is a very important activity for technological development in the era of the information and knowledge society. Although software production is not just programming, this task is crucial to the success of the process. The basis of the programming process is the algorithm, which consists of a finite and well-defined set of rules aiming at solving a problem in a finite time and should be effective and efficient. After specifying the algorithm, it will be encoded (written) in a programming language, which is a formal language with a specific set of well-defined instructions that allow you to develop software. There is a huge set of programming languages, each with specific fields of syntax and features and applications.

It is well known that, for many students, learning to program is notoriously difficult[1]. Programming is very a complex task that requires effort, a lot of practice and a special approach on the way it is taught and learned. To become a good programmer, student must practice a lot and have significant feedback about their work.

APROG's course unit is the first course unit that directly deals with programming and it takes place during the first year, first semester. It is organized into different types of classroom lessons: theoretical (T), theoretical-practical (TP) and practical-laboratory (PL) in a total of six hours weekly. In practical-laboratorial (PL) classes students must solve a set of exercises that are available in Moodle (Modular Object-Oriented Dynamic Learning Environment). Students must work in pairs with the EduScrum methodology, a variation of Scrum for education. They must codify the assignments in Java, using an Integrated Development Environment (IDE). Teachers suggest them to use NetBeans, but they can use another IDE like Eclipse or IntelliJ, if they prefer.

Due to the amount of exercises that students must solve to practice and improve their programming skills, and the number of students and the need for teachers to evaluate their assignments, class time is not enough to analyze all the tasks of all students in a deep way. So, the solution is evaluating by sampling, giving an incomplete and delayed feedback. However, the lack of feedback and mentoring was identified as one of the most important issues about programming learning [2], and a source of students' lack of motivation and commitment.

Therefore, we have tried to find a way to reduce teachers' evaluation work, giving them more time to clarify the students' doubts and to promote a solution that allows students a more autonomous work, less dependent on the teacher review to test their assignments. We looked for tools that could meet these requirements and we choose Virtual Programming Lab (VPL) [3], a Moodle plugin.

METHODOLOGY

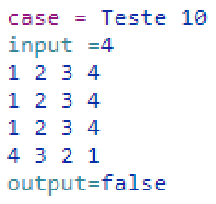
Virtual Programming Lab a plugin integrated into Moodle for programming area, and it has open source under the GNU / GPL license. VPL has been developed by the Department of Computer Science and Systems from University of Las Palmas of Gran Canarias, Spain[3].

This tool was developed with the objective of being open source and, int that way, receiving contributions that can promote its development. It should also be secure, independent of the programming language, easy to use, adequate to novice programming students and should allow automatic grading. It's functionalities includes to edit, compile, run, debug and evaluate code, with instantaneous feedback [4], allowing to track submissions, providing an historic results about compilation and implementation of each assignment[5].

In the academic year of 2017-2018 an experience was carried out in APROG, after solved some technical issues and assured all the installations and technical and logistical conditions [6]. Also, administrative conditions were fulfilled after obtained formal authorizations from responsible of the unit curricular of APROG, from the Director of the Informatics Engineering course and from School Board. There were enrolled in the process 4 teachers from 8 classes, in a potential universe of 161 students. However only 135 students attended classroom lessons on a regular basis, and less than 70 had made at least one submission in a process that was not mandatory.

APROG unit course had already several editions collecting a big set of exercises proposed to code by students to improve their programming skills. From this set of exercises, we selected six of them to test in VPL. The description of each exercise was adapted in order to make the statement more objective, with less possibility of diversity of solutions and to avoid some particularities that can cause malfunctions in VPL's submission.

It was necessary to prepare all the system, make students inscriptions, prepare the assignments, create the activities in Moodle, and prepare a set of test cases that can allow to evaluate each assignment in a complete way. Figure 1 shows an example of a test in an assignment that intents to verify is a matrix is a magic square:



```
case = Teste 10
input =4
1 2 3 4
1 2 3 4
1 2 3 4
4 3 2 1
output=false
```

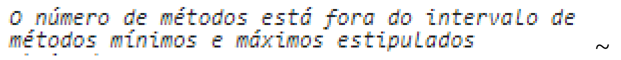
Figure 1. Example of a test case.

In the academic year of 2018-2019 is in progress another experience, with the same exercises that have been improved in same aspects. This year we tried to involve more students and in a deeper way, enrolling more than 200 students and 4 teachers. The assignments already submitted until now, had more than 180 average submissions for each assignment.

The process was already improved with additional verifications by an extended array of tools that can provide pre-compile, runtime tests, that suite each exercise specification. These changes allow to verify not only the results, but mainly the process, promoting coding best-practices, and help students to improve their code style. Some of the additional verification points includes:

- java class name and filename must match;
- forbidden use of Java Swing or Java FX;
- search for a specific method name that must be used;
- number of functions created for each assignment;
- number of constants;
- number of returns in a given method name;
- number of lines of a specific method name;

As example of this verification when the number of methods is out of the expected range the user get this message:



```
O número de métodos está fora do intervalo de
métodos mínimos e máximos estipulados
```

Figure 2. Example of a error message.

This year we also changed some evaluation restrictions trying to avoid that students try a huge number of times to evaluate their assignments in a try-error logic, without reflected properly about what is wrong in their assignment.

CONCLUSIONS

In this paper we presented an overview of an experience that intents to study the impact of VPL on the process of teach and learn programming. With the obtained preliminary results of this experience in a real unit course of a real course of Computer Engineering, we may consider that VPL is a useful tool that can provide to students an innovative pedagogical approach.

Overall, VPL reveals great potential to assist teachers to have a more efficient action and to improve students learning in a more effective and autonomous way, with less time and effort spent by teachers in task evaluation, reducing the human error of judgment.

REFERENCES

[1] J. E. Moström, A Study of Student Problems in Learning to Program (PhD dissertation), Department of Computing Science, Umeå University, Umeå, Sweden, 2011.

[2] T. Koulouri, S. Lauria and R. D. Macredie, "Teaching Introductory Programming: A Quantitative Evaluation of Different Approaches," ACM Transactions on Computing Education (TOCE), v.14 n.4, pp. 1-28, February 2015.

[3] "VPL - the Virtual Programming Lab for Moodle," [Online]. Available: http://vpl.dis.ulpgc.es. [Accessed 29 11 2018].

[4] D. Thiébaut, "Automatic evaluation of computer programs using Moodle's virtual programming lab (VPL) plug-in," *Journal of Computing Sciences in Colleges*, vol. 30(6), pp. 145-151, 2015.

[5] A. V. Wanhenheim, J. E. Martina, R. L. Cancian and J. C. Dovichi, Developing Programming Courses with Moodle and VPL - The Teacher's Guide to the Virtual Programming Lab, Bookess, 2015.

[6] M. Cardoso, A. Vieira de Castro, Á. Rocha, "Integration of Virtual Programming Lab in a process of teaching programming EduScrum based", 13th Iberian Conference on Information Systems and Technologies (CISTI), Cáceres, Spain, 2018.