8th International Conference on Higher Education Advances (HEAd'22)
Universitat Politècnica de València, València, 2022
DOI: http://dx.doi.org/10.4995/HEAd22.2022.14660

Use of Assessment and Feedback Systems for Introductory Computer Programming Modules of Higher Education: A Comparative Study

Jagadeeswaran Thangaraj[1], Monica Ward[1] , Fiona O'Riordan[2]
[1]School of Computing, Dublin City University, Ireland, [2]Teaching Enhancement Unit, Dublin City University, Ireland.

Abstract

Teaching introductory programming modules in higher education is highly challenging. In particular, it is hard to motivate novices if there is a lack of tutoring support and it is also difficult to assess their progress when there is a large class. Automated assessment and feedback systems help in this scenario. The use of automated assessment and feedback systems support teaching programming and motivate the novices in their learning process. This paper discusses these assessment and feedback systems which enable academics to assess, grade and support students learning programming. This study mainly concentrates on whether these systems enable them to understand basic programming concepts and help to improve their skills. The result of this study is that they are very effective in scaffolding the teaching of programming and assessing programming assignments. This study concludes that these systems still need some development to be more effective to motivate their learning process.

Keywords: *Automatic assessment; computing education; digital learning environment; formative assessment; higher education; interactive feedback.*

1. Introduction

Programming modules are essential for any software development related courses at higher education institutions all over the world. In Ireland, there are a number of programming modules with different names (Becker 2019), although many of them have similar content. The objective of these courses is to give the basic knowledge of programming languages by introducing syntax and semantics. Therefore, these modules play an important role to make them comfortable in continuing their education in computing. There are a number of activities introduced to motivate the novice students in programming modules: e.g. Intelligent tutoring, real-time problem solving, competence based (Najar A *et al*., 2012) and computational thinking (Lockwood *et al*., 2018). Assessment and feedback are also important to help and motivate their programming skills (Wang *et al*., 2017). Once assessment motivates their confidence, then their interest in programming will increase and the dropout rates will decrease (O'Brien *et al.,* 2016). Most of the studies found that the traditional feedback system does not notify students what the errors are exactly, what type of mistakes they made in programming assignments. It just assesses what they know, what they can achieve and not where they failed. It will enhance their ability if it points out exact information.

There are a number of solutions that have been found in order to make assessment easy and feedback to students where they made mistakes or found difficulties. Some of them are automated assessments and rubric styled evaluation. Automated assessment system (Qian *et al*., 2019) evaluates the student's program submissions and gives the feedback immediately. There are a variety of automatic assessment systems available to evaluate student programming assignments. They support automatic evaluation of student submissions based on different approaches such as natural language processing, machine learning, image recognition, targeted feedback, and gamified web systems. The objective of this study is to compare these automated assessment and feedback systems and approaches in order to motivate and build up the confidence of novice students in higher education.

2. Related Systems

This paper reviews the following assessment systems with regards to assessment and feedback techniques for first year programming modules. The reason for choosing these systems is that they are currently practising in some academic institutions.

1. Leed's automated assessment and feedback platform (Evans *et al*., 2020).
2. CodeRunner – Quiz based programming assessment system (Lobb *et al*., 2016).
3. VPL, the Virtual Programming lab for Moodle (Rodríguez-del-Pino *et al*., 2012)
4. Browser based pedagogical coding environment (Culligan *et al*., 2018, Azcona *et al.*, 2018).
5. 2TSW: Gamified Web Based System (Polito *et al*., 2019).

2.1. Leed's automated assessment and feedback platform

This system (Evans *et al*., 2020) enables students to receive automatically generated, instant and personalised feedback. It provides formative feedback to students after each learning exercise, with feedback usually provided following formal summative assessment. It is designed for multiple programming languages: C, C++, Python and simulation packages. It allows electronic submission and students receive the automatic grading with textual feedback on their submission. It follows a student-centred approach. The drawbacks of the system are unsuitability for learning management systems as it is a stand-alone system, and it fails to notify exactly where errors are.

2.2. CodeRunner

CodeRunner (Lobb *et al*., 2016) is another feedback system which was developed at the University of Canterbury. It follows the Moodle question type. Therefore, it is suitable for learning management systems (LMS). It runs millions of student quiz question submissions in Python, C, JavaScript, PHP, Octave and Matlab. CodeRunner is a Moodle quiz question type that allows teachers to run a program in order to grade a student's answer. By far the most common use of CodeRunner is in programming courses where students are asked to write program code to solve programming problems and that code is then graded by running it in a series of tests. CodeRunner enables the provision of different possible correct answers to assess the correctness of student's programs. These questions can be used in different computer science areas as well. Adaptive mode is available in Coderunner which allows students to write the programs to get immediate test case results. If they find any errors, they can resubmit with correct answers with a small penalty. It is a student-centred approach with immediate feedback to resubmit for further grading.

2.3. VPL: The virtual programming lab

The VPL (Rodríguez-del-Pino *et al*., 2012) is an extension to Moodle which enables academics to assess and grade programming assignments in many programming languages. It allows students to edit, run and test the computer programs in the browser. It also allows the academics to automatically evaluate and check plagiarism. It supports many programming languages: Java, Python, C#, PHP, JavaScript, Matlab and more. It allows the academics to set up different test cases in order to check student's submissions. Therefore, the students are immediately able to receive the feedback about the errors against the test cases. It helps to manage all the programming assignments on Moodle which makes the students' learning process easier.

2.4. Browser based pedagogical coding environment (MULE & Einstein)

There are a number of pedagogical coding environments available in use. MULE stands for Maynooth University Learning Environment. MULE (Culligan *et al*., 2018) is an on-line,

browser-based pedagogical coding environment, for delivering, marking and providing feedback on coding assignments. This system evaluates and provides feedback for 'Java' programs. The major advantage of this system is providing feedback immediately when learners submit their programming assignments. However, this tool needs to enhance how it provides error messages to support novice programmers. It follows the unit testing technique in order to grade.

Einstein (Azcona *et al.*, 2018) is another browser (docker) based learning system for programming modules. This is a stand alone virtual learning environment and it is in use at Dublin City University for a number of programming modules. This system lists out the programming exercises on its dashboard and allows the students to submit their solutions in 'Python' language. This system calculates the grades and provides the feedback immediately against predefined test cases. These both systems do not notify the student as to where the errors exactly are, how to improve them and what type of the error. However, they both provide immediate feedback and allow resubmission for improved grading. Also, they do not run on LMS as they are stand-alone applications.

2.5. 2TSW: Gamified Web Based System

2TSW is a web based assessment and feedback system to assess complex programming tasks (Polito *et al.*, 2019). It follows the gamified structure to motivate and engage students in programming tasks. This is a teacher-centred approach as teachers are able to post different tasks in the module space and students can submit their solutions. Instead of traditional grading, it provides batches to reward student's completions along with grade percentages. It motivates them to attempt different test cases. For failed results, it provides possible feedback to reattempt the questions. It supports peer feedback as well.

3. Review Criteria & Discussion

In order to investigate the use of assessment and feedback systems for introductory computer programming modules, the following criteria has been used for effective review of these systems.

3.1. Feedback types

This criterion defines the types of feedback the assessment systems provide.

- Text feedback
- Audio or visual feedback
- Peer feedback

3.2. Assessment approaches

Assessment approaches that the system follows are classified into three types based on the assessment escalation (Souza 2016).

- Teacher-centred: Teachers initiate the assessment process in these tools, so students do not get immediate feedback.
- Student-centred: Students get immediate feedback as they initiate the assessment process in these tools.
- Hybrid: Either student or teacher can initiate the assessment process and students get partial feedback.

Table 1. Comparison of Automated assessment and feedback systems

Tool	Technology	Features	List of Languages support	Approach	Feedback type
Leed's	Stand-alone application	Formative feedback	C, C++, Python	Teacher centred	Text
CodeRunner	Moodle plugin	Moodle quiz of programming assignments	Python, C, JavaScript, PHP, Octave, Matlab	Student centred	Text
VPL	Moodle plugin	Immediate feedback	Many programming languages including Python & Java	Student centred	Text
MULE	Stand-alone application	Immediate feedback	Java	Student centred	Text
Einstein	Stand-alone application	Immediate feedback	Python	Student centred	Text
2TSW	Stand-alone application	Summative feedback	C	Teacher centred	Text & Peer

3.3. Features of the tools

This criterion lists out different features that the assessment systems provide.

- Electronic submission
- Automatic assessment
- Automatic grading
- Immediate feedback

3.4. Assessment system interfaces

This criterion describes the working environment of the systems.

- Learning management system
- Stand-alone system
- Web user interface
- Integrated development environment

3.5. Discussion

This study revealed that Leed's, MULE, Einstein and 2TSW are stand-alone systems which do not support common learning management systems as Coderunner and VPL do as shown in Table.1. Every system supports different programming languages. Coderunner, Einstein and MULE are student-centred as students can start to run the system whenever they need contrast with others. In conclusion, most of the systems have established that their primary motivation is to provide the feedback immediately as a student-centred approach to help them understand where the code went wrong, what type of the errors are in the code and the option to make the changes and re-submit. The 2TSW allows peer assessment where students can get feedback from their fellow students. Therefore, these assessment systems help the students to reflect and enable them to learn from the mistakes.

4. Conclusion

This study reviewed different assessment and feedback systems and looked at how they assess computer programming assignments, how they provide feedback and grading and how they help academics in online teaching. All of them are very useful in different ways and for different requirements. Therefore, these systems help the students to reflect and enable them to learn from their mistakes. However, this study found that there are some drawbacks with these systems. The primary one is that they assess the students with the same questions, with no differentiation for student ability. Adaptive assessment is the process for assessing the students with different abilities with different sets of questions (Chatzopoulou 2010). It assesses the students repeatedly until they attempt the correct answer in the level. Therefore, it confirms the students' knowledge in the level and notifies them the area where they lack

competence or have difficulties. This study suggests an adaptive assessment for these systems which fosters student commitment to complete the programming tasks. It would be better to implement the adaptive assessment in addition to the automated feedback to enhance the learning process.

References

Azcona, David and Hsiao, I-Han and Smeaton, Alan F. (2018), "Personalizing Computer Science Education by Leveraging Multimodal Learning Analytics," *2018 IEEE Frontiers in Education Conference (FIE)*, pp. 1-9, doi: 10.1109/FIE.2018.8658596.

Becker, Brett A. (2019). A Survey of Introductory Programming Courses in Ireland. *The 2019 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '19)*. Association for Computing Machinery, New York, NY, USA, 58–64. https://doi.org/10.1145/3304221.3319752

Chatzopoulou, Dimitra I. and Economides, Anastasios A. (2010). Adaptive assessment of student's knowledge in programming courses, *J. Comput. Assist. Learn.* 26 (2010): 258-269: 10.1111/j.1365-2729.2010.00363.x

Culligan, Natalie and Casey, Kevin (2018). Building an Authentic Novice Programming Lab Environment. *Irish Conference on Engaging Pedagogy*, 13-14 December 2018, Dublin City University.

Evans, Craig A. and Wilson, Sam (2020). Development of an automated assessment and feedback platform, *Project report*, Feb 2020, University of Leeds.

Lockwood, James and Mooney, Aidan. (2018). Developing a Computational Thinking Test using Bebras problems.

Lobb, Richard and Harlow, Jenny. (2016). Coderunner: a tool for assessing computer programming skills. *ACM Inroads* 7, 1 (March 2016), 47–51. DOI: https://doi.org/10.1145/2810041

Najar, Amir Shareghi and Mitrovic, Antonija (2012). Using Examples in Intelligent Tutoring Systems. In: Cerri S.A., Clancey W.J., Papadourakis G., Panourgia K. (eds) *Intelligent Tutoring Systems. ITS 2012. Lecture Notes in Computer Science, vol 7315*. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-30950-2_76

O'Brien, C., Humphreys, J., & Ide McAuliffe, N. (2016). Concern over drop-out rates in Computer Science courses. *Irish Times*. Available: http://www.irishtimes.com/news/education/concern-over-drop-out-rates-in-computer-science-courses-1.2491751

Polito, Giuseppina & Temperini, Marco and Sterbini, Andrea (2019). "2TSW: Automated Assessment of Computer Programming Assignments, in a Gamified Web Based System," *18th International Conference on Information Technology Based Higher Education and Training (ITHET)*, 2019, pp. 1-9, doi: 10.1109/ITHET46829.2019.8937377.

Qian, Yizhou and Lehman, James D. (2019). Using Targeted Feedback to Address Common Student Misconceptions in Introductory Programming: A Data-Driven Approach. *SAGE Open*. https://doi.org/10.1177/2158244019885136

Rodríguez-del-Pino, Juan Carlos and Royo, Enrique Rubio and Zenón José Hernández Figueroa. (2012). A Virtual Programming Lab for Moodle with automatic assessment and anti-plagiarism features. *International Conference on e-Learning, e-Business, Enterprise Information Systems, and e-Government.*

Souza, Draylson and Felizardo, Katia and Ellen Barbosa. (2016). A Systematic Literature Review of Assessment Tools for Programming Assignments. *2016 IEEE 29th International Conference on Software Engineering Education and Training (CSEET)*, 147-156. 10.1109/CSEET.2016.48.

Wang, Xiao-Ming, Gwo-Jen Hwang, Zi-Yun Liang, and Hsiu-Ying Wang. (2017). Enhancing Students' Computer Programming Performances, Critical Thinking Awareness and Attitudes towards Programming: An Online Peer-Assessment Attempt. *Journal of Educational Technology & Society*, 20(4), 58–68. http://www.jstor.org/stable/26229205