Practice makes perfect – Gamification of a competitive learning experience

Antonio D. Kheirkhahzadeh[1], Christian S. Sauer[1], Panagiotis Fotaris[2]
[1]University of West London, London, UK
[2]University of East London, London, UK
kheiant@uwl.ac.uk
sauechr@uwl.ac.uk
p.fotaris@uel.ac.uk

Abstract: The ability to provide and implement software solutions is a fundamental component of a computer scientist curriculum. Commonly referred to as the ability to program, this task involves the development of programs to address everyday problems. Over the last decade teaching practices have evolved alongside programming languages to facilitate the learning process. While abstracting the level of understanding has helped students with the fundamentals of software development, issues related to students' engagement and motivation are still not adequately addressed. With motivation being a vital component of the students' life cycle and at the basis of their engagement, the concept of software engineering introduced in the class needs to be revised and become more engaging so as to be practised thoroughly by the students.

To address these challenges, educators have devised numerous frameworks to allow students to hone their programming skills. The idea of embedding gaming aspects into the learning cycle has led to the development of techniques such as serious games and game-based learning, while more recent techniques have been unified under the term gamification. Several researchers have incorporated the gamification concept into computer science classes in order to improve students' engagement with the teaching material, with early evaluations confirming the effectiveness of this approach. The present study focuses on the use of a gamification platform to create stimulating content and increase motivation. Students were presented with a new gamification system designed to attract and hold their attention through a number of programming challenges in the form of a contest. The results of the experiment demonstrate the students' behavioural changes towards a deeper cognitive engagement. The paper then further discusses the challenges that have arisen in this new learning environment, such as demotivation of students with low contest rankings.

Teaching how to write good software has been part of an ongoing debate for the last decade. With student motivation being a central component, this paper discusses the use of a gamification environment to engage students with the teaching material and reinforce the concepts of software engineering introduced in class.

Keywords: Gamification; Motivation; Engagement; Programming Education;

1. Introduction

The ability to develop software using a variety of programming languages is a fundamental part of computer scientists' curricula. Often shortened as the ability to *program*, these tasks require scientists to provide software-based solutions to real life problems and everyday challenges. From a high-level point of view there are different ways of providing a solution to a given problem. What usually differentiates a computer scientist from an amateur programmer is the ability to provide the most efficient and strategic solution while considering the environment where the software will be deployed. This requires the developer to have a good understanding of both the hardware and software, which is achieved after a successful completion of a computer science degree.

Nevertheless, teaching and assessment of computer programming is considered to be difficult and frequently ineffective, particularly for weaker students, as computer programs and algorithms are abstract and complex entities that involve concepts and processes which are often found hard to teach and learn (Olsson et al., 2015; Lahtinen et al., 2005). This sometimes results in undesirable outcomes such as disengagement, cheating, learned helplessness, and dropping out (Robins et al., 2003; Winslow, 1996). To address these issues, Higher education institutions (HEI) are beginning to adopt a set of techniques collectively called "gamification", which involve the use of game design elements in a non-game context (Deterding et al., 2011), to engage students with programming challenges in a more entertaining manner. Following successful examples from areas such as sales, marketing, finance, and health, the basic idea is to increase users' activity and retention while also improving their motivation and engagement by incorporating video game design elements and mechanics over an additional service layer (Dominguez et al., 2013). The latter can now be found in most popular e-learning

environment and from third-party content management systems. It often includes reputation systems, badges, levels, and leaderboards where students can compete when providing solutions to a given problem.

2. Rationale

Over the last decade there have been numerous studies addressing the difficulties of teaching and learning computer programming (Milne and Rowe, 2002). Several issues have been highlighted, resulting in the development of new teaching techniques and programming languages designed to facilitate learning (Vihavainen et al., 2011, Moore et al., 2013). While these solutions have been proven successful by abstracting the level of understanding, issues related to student engagement and motivation are still outstanding. The role of the educator is essential to ensure a correct delivery and understanding of programming concepts and designs. Even after a successful completion of the module assessment, students may still not have the programming skills expected in the learning outcomes (McCracken et al. 2001), with the main issue behind this being the lack of computing fundamentals and practical sessions. In order to progress from novice to expert students are required to apply the basic skills introduced in the class (Robins et al., 2003), as software engineering concepts need to be reinforced through practice. However, the ever-recurring issues of poor student engagement and motivation are usually the main culprits behind a lack of interest, which in turn leads to a lack of practice (Jenkins, 2001).

The aim of this study is to investigate whether applying the concept of gamification in a programming course can improve student motivation and engagement. Motivation is a crucial factor which influences the student behaviour hence the desire to practice beyond class activities. The idea behind gamification is to trigger the intrinsic and extrinsic motivation of students in order to practice more and extend their knowledge beyond the class. As discussed, issues related to the difficulties of teaching programming are mostly related to motivation. Students are required to practice and apply the concepts of good software engineering introduced in class. Therefore, teaching techniques that enable students to continue applying their knowledge and skills after the class are essential to enhance student performance and interest in the topic. Intrinsic motivation is driven by a genuine interest of an activity for its inherent satisfactions (Ryan and Deci, 2000). Conversely, individuals driven by an extrinsic motivation seek to comply with an external requirement for its instrumental value, which ranges from grades to leaderboards (Ryan and Deci, 2000). Recent research has divided the concept of engagement into three different categories listed below (Fredricks et al., 2004, Appleton et al., 2008, Astin, 1984).

- Cognitive engagement. Students invest additional time to achieve a deep understanding of the topic introduced in the class. This involves the desire to learn beyond the materials covered during taught sessions and to enjoy challenges (Fredricks et al., 2004).

- Behavioural engagement. This behaviour is related to an overall positive conduct and the absence of disruptive behaviours. Students express attention and persistence while participating to classroom activities (Fredricks et al., 2004, Appleton et al., 2008).

- Emotional engagement. Refers to the willingness of being involved in the classroom activities and a positive interest in learning (Appleton et al., 2008).

Considering that higher levels of learning outcomes have been associated with a cognitive engagement with courseware (Kong et al., 2003), the proposed method offers programming challenges to students over a gamified environment, thus allowing them to learn beyond the material covered in class, change their attitude towards programming, and become self-regulated learners. This cognitive engagement will result in an increase in practice, hence providing a better understanding of programming and software development methodologies (Robins et al., 2003).

3. Literature review

One of the main issues highlighted by research over the last decade is the challenging nature of teaching software development (Robins et al., 2003, Milne and Rowe, 2002, Gonzalez and Mora Carreno, 2014). Since mastering programming needs a high level of abstraction without direct connection with real world tasks, many students find it difficult to study under such circumstances and are not always prepared to spend a lot of time on reading documentation, writing code, and experimenting (Shabalina et al., 2013). To facilitate learning rather than deliver teaching while making the education process easier and more appealing, new teaching practices have been devised, including gamification (Vihavainen et al., 2011, Moore et al., 2013).

3.1 Gamification applied

The concept of gamification is being increasingly adopted into the teaching practices of a number of universities (Bouki et al., 2014, Wei-Qing et al. 2014, Gonzalez and Mora Carreno, 2014). Influenced by this growing trend, educational platforms and learning management systems such as Moodle and Blackboard are also evolving; new plug-ins have been developed which allow lecturers to use content-rich materials with game design elements and mechanics, thus gamifying the learning experience (Bouki et al., 2014). Traditional teaching techniques have therefore been revised and adapted to provide a more personalised teaching approach through the adopted course materials.

In this scenario the role of the educator is subject to a proactive change (Bellotti and Dagnino, 2013), as s/he must follow the students through the gamified environment and actively monitor their performance. While traditional teaching relies on the preparation of teaching material, its delivery, and the assessment of student work, gamification requires of the educator to prepare reflective material, guide students through the learning process, and monitor their performance via the learning environment (Tretinjak et al., 2014). Similarly, the role of the learner is also subject to a proactive change. Students are expected to practice their skills and verify their understanding of the material introduced during lectures. The necessary knowledge and skills are gained by completing the interactive material provided by the gamified environment. More importantly though, students are able to continue working with the gamified environment after the class, which provides the basis for developing a cognitive engagement. Applying the knowledge and basic skills introduced in class is essential to enhance their performance and improve their programming ability (Gonzalez and Mora Carreno, 2014, Uskov and Sekar, 2014).

Gamified frameworks are under the attention and constant scrutiny of both researchers and educators. Although qualitative and quantitative results are promising, achieving the desired effect is not trivial. Early implementations of a gamified teaching strategy identified various issues, e.g., the gamified environment needs to be designed, implemented and maintained on the effort of the teacher, while competing with top students on a leaderboard can result in a lack of interest (Dominguez et al. 2013). In the computing field, challenges such as the lack of immediate feedback (Kapp, 2012) have been resolved with the introduction of more complex systems that can automatically validate student work after submission. Alternatively, third-party virtual learning environments can be used to provide the basics for a gamified experience.

3.2 Other teaching strategies

Despite its recent popularity, gamification is not the only method incorporated into computer science courses in order to improve students' performance, engagement, and programming skills (Robins et al., 2003). Vihavainen et al (2011) introduced "extreme apprenticeship" as a teaching strategy where students learn under the constant supervision of an educator. This is an excellent way to teach programming since it is based on "learning by doing" and "continuous feedback" as the means to achieve learning. However, like other teaching techniques that have been able to achieve good results, it has a major drawback: it requires the constant presence of the educator to facilitate learning and guide students through the subject material. Gamification is able to solve this issue by providing constant monitoring via the learning environment and offer additional feedback when required.

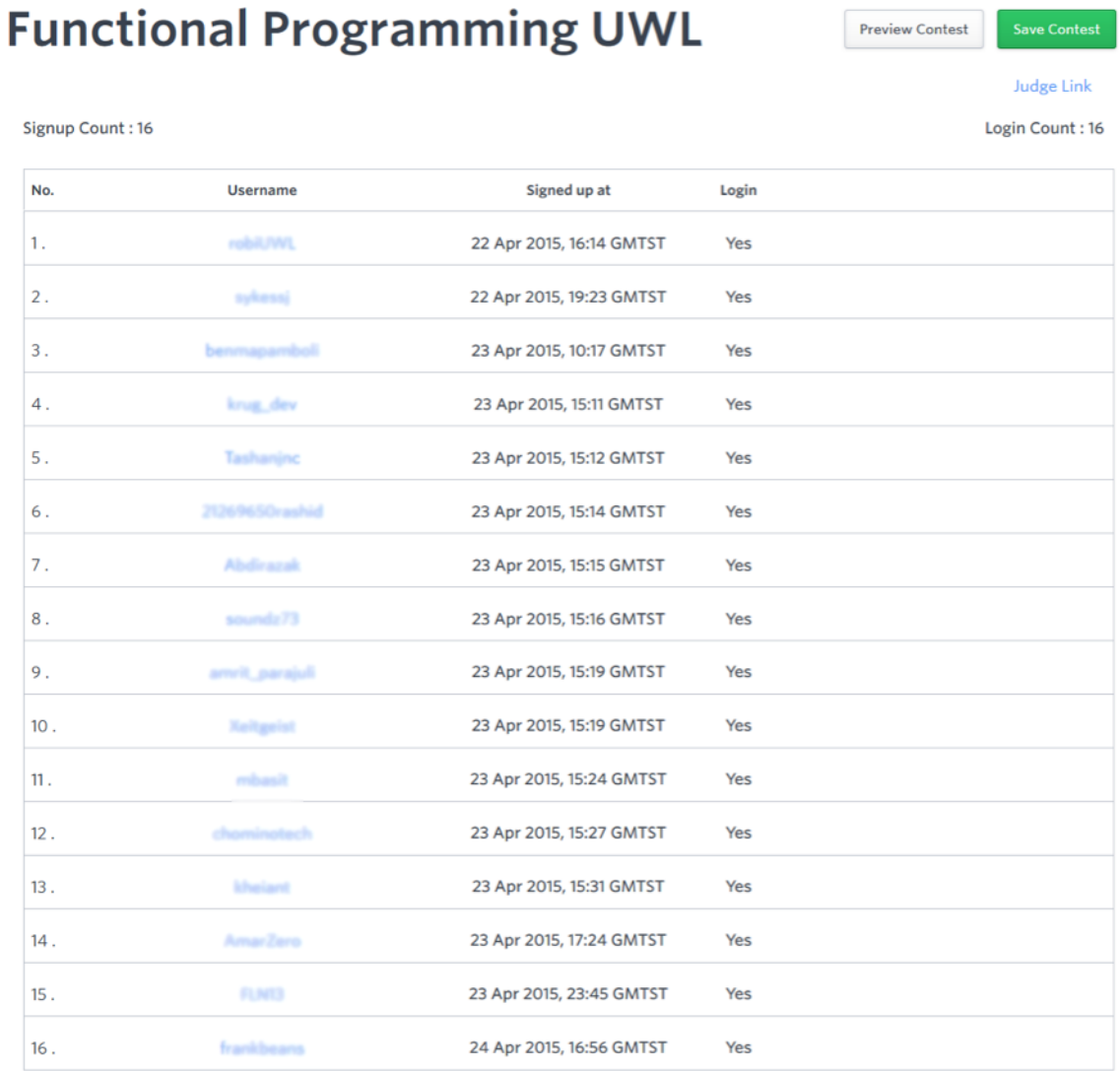
4. Methodology

This research explores how the application of a gamification framework in a computer programming course can affect the learning experience and the students' motivation and engagement. More specifically, the main aim of this experiment was to initially engage students with a programming contest, which provided the key elements that define a gamification environment, and subsequently monitor their performance within the said environment. Students were able to view the other contestants and their activities, as well as access the contest leaderboard. Performance and engagement were monitored throughout and after the experiment.

4.1 Participants

The present study was performed with undergraduate computer science students who were in the second semester of their studies. A prerequisite for participating in the contest was the successful completion of the first semester programming modules. This requirement ensured that all contestants had a basic understanding of software engineering and programming concepts. The gamification experiment was conducted during a laboratory module aimed at supporting students with their programming tasks and providing additional material to improve their problem-solving skills.

A total of 33 students were enrolled on the module for the duration of the semester, with 15 to 20 of them attending the class seminar on a weekly basis. The contest was announced via the institutional VLE (Blackboard) 2 weeks prior to its launch and registered a total of 16 active participants. The majority of them registered during the workshop class on the 23rd of April 2015, while some students registered one day before or after the contest launch.



Functional Programming UWL

Preview Contest | Save Contest

Judge Link

Signup Count : 16 | Login Count : 16

No.	Username	Signed up at	Login
1.	robiUWL	22 Apr 2015, 16:14 GMTST	Yes
2.	sykessj	22 Apr 2015, 19:23 GMTST	Yes
3.	benmapamboli	23 Apr 2015, 10:17 GMTST	Yes
4.	krug_dev	23 Apr 2015, 15:11 GMTST	Yes
5.	Tasharsjnc	23 Apr 2015, 15:12 GMTST	Yes
6.	21269650rashid	23 Apr 2015, 15:14 GMTST	Yes
7.	Abdirazak	23 Apr 2015, 15:15 GMTST	Yes
8.	sounds73	23 Apr 2015, 15:16 GMTST	Yes
9.	amrit_parajuli	23 Apr 2015, 15:19 GMTST	Yes
10.	Keltgeist	23 Apr 2015, 15:19 GMTST	Yes
11.	mbasit	23 Apr 2015, 15:24 GMTST	Yes
12.	chominotech	23 Apr 2015, 15:27 GMTST	Yes
13.	kheiant	23 Apr 2015, 15:31 GMTST	Yes
14.	AmarZaro	23 Apr 2015, 17:24 GMTST	Yes
15.	FLNTJ	23 Apr 2015, 23:45 GMTST	Yes
16.	frankbeans	24 Apr 2015, 16:56 GMTST	Yes

Figure 1: List of participants

4.2 The gamified environment

The virtual learning environment adopted for the gamification framework was provided by HackerRank, a company that focuses on competitive programming challenges for both consumers and businesses and has an online community of over one million computer programmers (Kosner, 2014). HackerRank's programming challenges can be solved in a variety of programming languages (including Python, Java, C++, PHP, SQL etc.) and span multiple computer science domains. Users are able to solve challenges individually or sign up for multi-users contests. For educational institutions, HackerRank also offers the ability to develop new challenges and contests aimed at students.

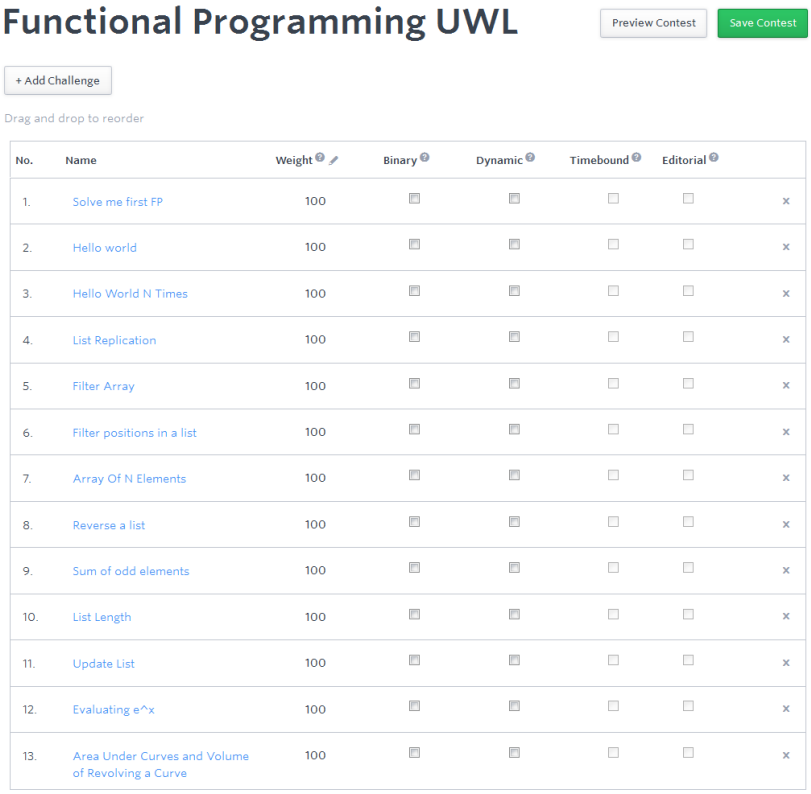


Figure 2: Contest's challenges

The contest domain was based on functional programming, a paradigm introduced during the first semester of the course. As shown in figure 2, a total of 13 challenges were used to trigger the student problem-solving skills. These tasks ranged from simple to more complex challenges, which are usually offered to first year computer scientists. Each challenge had a problem statement, an input/output format, and execution samples. Students were allowed to use various functional languages to provide solutions to tasks, as well as compare their program output against a test case before entering their final submission. The latter was then compared against more test cases, which checked the robustness of the developed program. The scoring system was based on the solutions to the programming challenges (i.e., the percentage of test cases the submitted code passed), with a correct and optimal solution passing all the test cases. Additional system features included the ability to monitor each submission and provide immediate feedback when needed. Figure 3 illustrates the list of student submissions, the accepted solutions, and the achieved score.

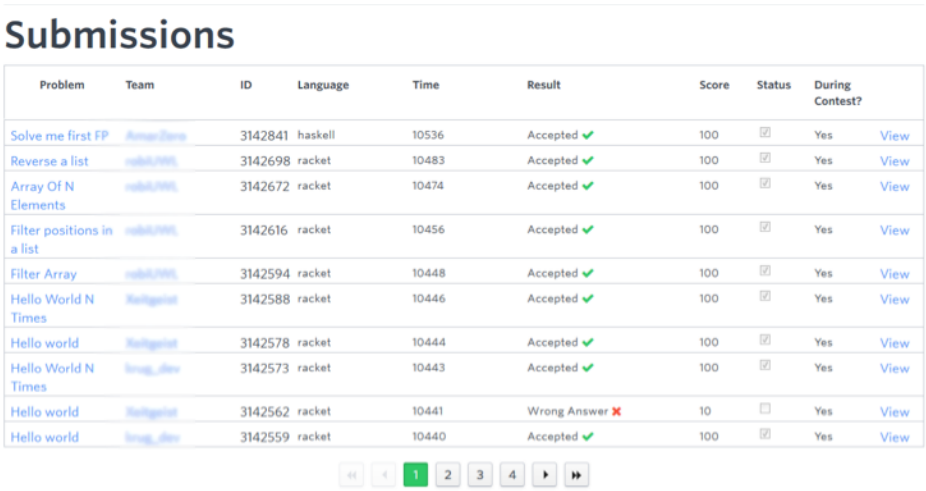


Figure 3: Contest's student submissions

4.3 Implementation

The gamification experiment was based on a series of exercises discussed in the module specification. As part of the module, students were encouraged to register to the contest using their university email address. The contest was performed over a period of one week. Prior to this, students were introduced to the gamification environment and encouraged to solve other programming challenges provided by HackerRank. The contest was formally announced during the class and was concluded after two weeks. Although the majority of attending students participated in the contest during the class, there was also a small number of non-attending contestants.

With submissions being monitored throughout the contests, most challenges were solved in-class during the workshops. Although the majority of students managed to find the correct solutions to the challenges on their own, facilitators were always available to offer additional support when needed. However, their intervention was limited and it mostly addressed usability issues of the gamified environment. Finally, student performance was also monitored outside the class in order to understand whether there was further engagement with the contest challenges.

4.4 Ethical considerations

No confidential data was released by the students during the process of the gamification experiment. The third-party system provided the standard security measurements to ensure privacy and anonymity. Students were reassured that the data collected from the gamification experiment would be anonymised and would not be used for grading purposes.

5. Results

The aim of this study was to evaluate student performance and engagement with a number of programming challenges. The results of the experiment can be divided into two parts: the contest and the subsequent interaction with the gamified environment.

The results of the gamification experiment were collected at the end of the contest. Figure 4 illustrates the percentage of challenges completed by each one of the 16 students. The legend provides the amount of points collected based on the number of challenges solved. 38% of the students collected a total of 100 points (the minimum requirement), 6% got 110 points, 13% received 300 points, and 6% were awarded with the maximum of 800 points after solving all available challenges. The remaining 37% failed to collect any points at all. According to these findings there appears to be an equal amount of students who were able to achieve 100 points and of students who were unable to provide a correct solution to the contest challenges, respectively. This can be attributed to issues related to usability and disengagement, as some of the challenges already provided a sample solution to the problem. A point range over 100 was reached by 24% of the student cohort and it demonstrates their interest to solve optional challenges during the contest in order to collect more points.

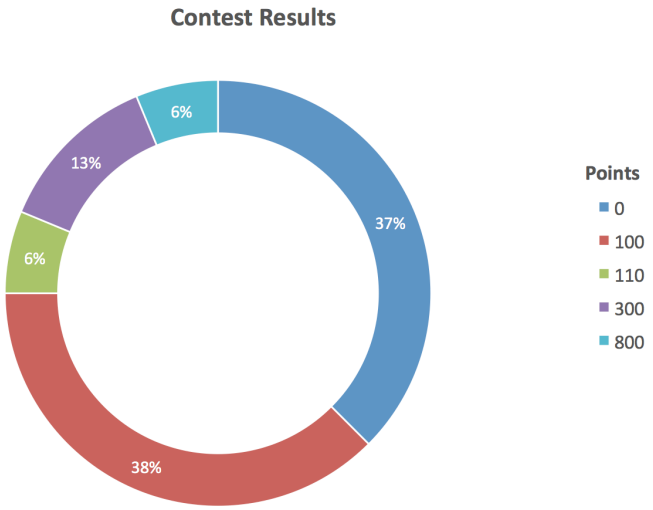


Figure 4: Contest results

The second set of results provides additional information in regards to the students' engagement with the gamified environment. As depicted in Figure 5, most students engaged with the contest in-class (94%) when the workshop took place. A total of 5 students (31%) practised off-site one to two days before the workshop and only 1 student (6%) continued practising with extra challenges provided by HackerRank after the end of the contest.

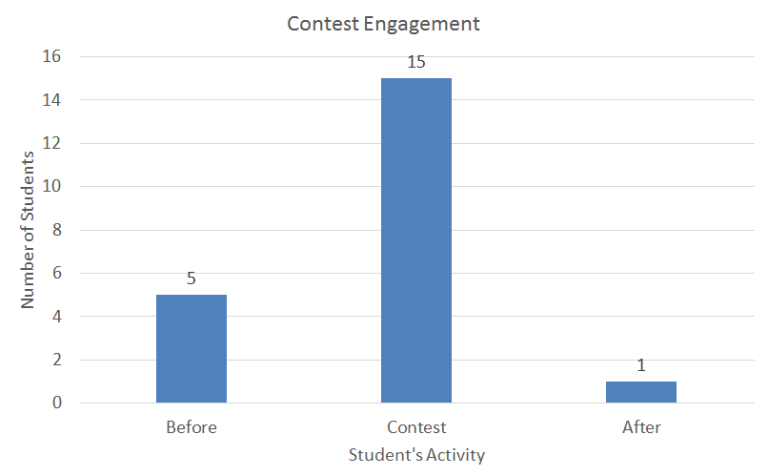


Figure 5: Contest engagement

Overall, these results demonstrate a good interaction and engagement with the contest in-class, with almost 2/3 of the cohort collecting between 100-800 points. However, despite having an adequate number of students engaging with the contest prior to the workshop class, it was rather disappointing to see only one student undertaking additional challenges after the end of the contest.

6. Discussion

A qualitative analysis of the gamification experiment can provide additional information on the student engagement with the module material. Although several studies have already evaluated the benefits of a gamified environment, some issues are still outstanding (Hamari et al., 2014). The rationale for this study was to trigger a cognitive engagement with the module material in order to achieve a deep understanding of the subject. This requires a desire to go beyond the module requirements and seek additional learning material. Most of the studies done to date do not discuss whether students express a cognitive engagement, which is partially related to the limitation of a gamified environment and the difficulties of tracking student work outside the class. Dicheva et al. (2014) argue the need for a more solid evidence base and suggest the use of tools that can assist in implementing gamification effectively, since leaderboards and points alone are deemed insufficient. A gamified learning environment that aspires to facilitate the transition from behavioural to cognitive engagement needs to be capable of providing additional learning materials.

6.1 The educator as a guide

In this new gamified environment the role of the educator is often neglected. Recent studies focus on the use of gamified learning with little or no participation of an educator. Research has shown how some of the best teaching strategies involve better interaction between learners and educators (Vihavainen et al., 2011, Robins et al., 2003, Milne and Rowe, 2002). With particular attention being paid to teaching programming for beginners Vihavainen et al. (2011) have shown how techniques such as extreme apprenticeship are extremely efficient at teaching skills that require building routines. Bellotti and Dagnino (2013) describe the role of the educator as key to demonstrating that gamification cannot be reduced to game elements and design. The gamification experiment conducted in this work has shown how the role of the educator is indeed necessary. However, compared to the extreme apprenticeship technique, this gamified environment mostly requires the educator to assist students during the first few interactions with the system. Assistance and guidance are constantly needed during the practical sessions of the contest to reinforce the concepts of good software

engineering introduced in the lecture. Furthermore, the educator can assist students with usability issues that commonly arise during the first interactions with a gamified environment.

6.2 A consumer culture

The results of the experiment suggest a positive engagement with the programming challenges. However, the percentage of student who did not participate in the content is relatively high. Almost one third of the class did not engage with the material available on the third party system. This can be attributed to the issue related to module learning outcomes and assessment criteria as the experiment was not part of a formal assessment. As highlighted by Molesworth et al. (2009), student behaviour towards education has changed over the last few years. In a consumer culture, students focus on 'having' a degree rather than 'being learners'. This perception of education has a negative impact on the students' learning as they seek the easiest way to obtain a qualification.

6.3 Behavioural changes

In regards to student behaviour, the results provide evidence of cognitive engagement. A total 24% of the student cohort showed interest in solving additional challenges in order to collect more points. This demonstrates a positive attitude and commitment towards learning. However, similar to the Ibanez et al. (2014) case study, there were several students who did not continue to work on additional challenges after achieving 100 points. As Domínguez et al. (2014) also suggest, this demonstrates failure in transitioning from emotional to cognitive engagement as the motivation was not the same for everyone. With the gamified environment being monitored after the completion of the contest on an individual basis, it was observed that engagement for students on a lower ranking system was less compared to those with higher points. This can also be associated to the issue highlighted by Domínguez et al. (2014) where students “did not find it fun” to compete for a rank on the leaderboard. Conversely, students on the top positions of the leaderboard were actively engaged with the contest and demonstrated a good level of competition.

7. Conclusions and Future work

Motivation and engagement are important issues that affect student learning. As suggested by the present study, diverse teaching strategy capable of entertaining students can trigger the student cognitive engagement and enhance their problem-solving and programming skills. However, some of the limitations with this work involve the gamified environment that was employed for the experiment. More specifically, a number of students experienced usability issues while dealing with the contest submissions, which led to frustration and disengagement. In hindsight, more time should have been spent to introducing the system to students in order to have avoided these issues. Another limitation involving the gamified environment is related to the monitoring of the students. While the system allows to review the contest submissions, it is not possible to monitor the amount of hours the student engaged with the material nor the amount of repeat sessions.

Future work plans to repeat the experiment across several semesters with a larger cohort of students and provide more insight and results with comparisons between students that engaged with the gamified environment and those who did not. This includes expanding this study with a larger student cohort; second and third year undergraduate computing students will also be invited to participate in a new department-wide contest. The participants’ engagement with the gamified environment and their motivation to solve challenges will be monitored throughout and after the contest. Finally, there are plans of integrating the gamified platform into Blackboard, as this would centralise the system and allow contest challenges to be part of formal assessments, which could possibly result in wider participation.

References

Appleton, J.J., Christenson, S.L., Furlong, M.J., (2008). *Student engagement with school: Critical conceptual and methodological issues of the construct.* Psychology in the Schools 45, 369–386.

Astin, A.W., (1984). *Student involvement: A developmental theory for higher education*. Journal of college student personnel 25, 297–308.

Bellotti, F., Berta, R., De Gloria, A., Lavagnino, E., Dagnino, F.M., Antonaci, A., Ott, M., (2013). *A Gamified Short Course for Promoting Entrepreneurship among ICT Engineering Students*, in: Advanced Learning Technologies (ICALT), 2013 IEEE 13th International Conference on. pp. 31–32. doi:10.1109/ICALT.2013.14

Bouki, V., Economou, D., Kathrani, P., (2014). *???Gamification??? and legal education: A game based application for teaching university law students*, in: Interactive Mobile Communication Technologies and Learning (IMCL), 2014 International Conference on. pp. 213–216. doi:10.1109/IMCTL.2014.7011134

Deterding, S., Dixon, D., Khaled, R., Nacke, L., (2011). *From game design elements to gamefulness: defining gamification*, in: Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments. ACM, pp. 9–15.

Dicheva, D., Irwin, K., Dichev, C., Talasila, S., (2014). *A course gamification platform supporting student motivation and engagement*, in: Web and Open Access to Learning (ICWOAL), 2014 International Conference on. IEEE, pp. 1–4.

Domínguez, A., Saenz-de-Navarrete, J., De-Marcos, L., Fernández-Sanz, L., Pagés, C., Martínez-Herráiz, J.-J., (2013). *Gamifying learning experiences: Practical implications and outcomes*. Computers & Education 63, 380–392.

Fredricks, J.A., Blumenfeld, P.C., Paris, A.H., (2004). *School engagement: Potential of the concept, state of the evidenc*e. Review of educational research 74, 59–109.

Gonzalez, C.S., Mora Carreno, A., (2014). *Methodological proposal for gamification in the computer engineering teaching*, in: Computers in Education (SIIE), 2014 International Symposium on. pp. 29–34. doi:10.1109/SIIE.2014.7017700

Hamari, J., Koivisto, J., Sarsa, H., (2014). *Does Gamification Work? – A Literature Review of Empirical Studies on Gamification*, in: System Sciences (HICSS), 2014 47th Hawaii International Conference on. pp. 3025–3034. doi:10.1109/HICSS.2014.377

Ibanez, M., Di Serio, A., & Delgado Kloos, C. (2014). *Gamification for Engaging Computer Science Students in Learning Activities: A Case Study*. Learning Technologies, IEEE Transactions on, PP, 1. doi:10.1109/TLT.2014.2329293

Jenkins, T., (2001). *The motivation of students of programming*, in: ACM SIGCSE Bulletin. ACM, pp. 53–56.

Kapp, K.M., (2012). *The Gamification of Learning and Instruction: Game-Based Methods and Strategies for Training and Education*, 1 edition. ed. John Wiley & Sons, San Francisco, CA.

Kong, Q.-P., Wong, N.-Y., Lam, C.-C., (2003). *Student engagement in mathematics: Development of instrument and validation of construct*. Mathematics Education Research Journal 15, 4–21.

Kosner, A. W. (2014). *HackerRank Solves Tech Hiring Crisis By Finding Programmers Where They Live*. Forbes, [Online], Available: http://www.forbes.com/sites/anthonykosner/2014/06/12/hackerrank-solves-tech-hiring-crisis-by-finding-programmers-where-they-live/#b7c922c1501e [1 May 2016].

McCracken, M. et al. (2001) *A multi-national, multi-institutional study of assessment of programming skills of first-year CS students*. ACM SIGCSE Bulletin. 33(4). p.125-180. ACM

Milne, I. and Rowe, G. (2002) *Difficulties in learning and teaching programming—views of students and tutors.* Education and Information technologies. 7(1). p. 55-66. Springer.

Molesworth, M., Nixon, E. Scullion, R. (2009). *Having, being and higher education: The marketisation of the university and the transformation of the student into consumer*. Teaching in Higher Education. 14 (3). p. 277-287.

Moore, J.P.T., Bagale, J.N., Kheirkhahzadeh, A.D. (2013). *Teaching Networking Fundamentals with Sound, in: Advanced Learning Technologies* (ICALT), IEEE 13th International Conference on. pp. 369–370. doi:10.1109/ICALT.2013.113

Ryan, R and Deci, E. L. (2000) *Intrinsic and Extrinsic Motivations: Classic Definitions and New Directions*, Contemporary Educational Psycology, vol. 25, no. 1, pp 54-67

Robins, A., Rountree, J., Rountree, N. (2003). *Learning and teaching programming: A review and discussion*. Computer Science Education. 13(2). P 137-172. Taylor & Francis.

Shabalina, O., Sadovnikova, N., Kravets, A. (2013) *Methodology of Teaching Software Engineering: Game-Based Learning Cycle*, in: Engineering of Computer Based Systems (ECBS-EERC), 2013 3rd Eastern European Regional Conference on the, Budapest, 2013, pp. 113-119. doi: 10.1109/ECBS-EERC.2013.22

Tretinjak, M.F., Bednjanec, A., Tretinjak, M., (2014). *Application of modern teaching techniques in the educational process*, in: Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014 37th International Convention on. pp. 628–632. doi:10.1109/MIPRO.2014.6859643

Uskov, V., Sekar, B., (2014). *Gamification of software engineering curriculum*, in: Frontiers in Education Conference (FIE), 2014 IEEE. pp. 1–8. doi:10.1109/FIE.2014.7044098

Vihavainen, A., Paksula, M., Luukkainen, M., (2011). *Extreme apprenticeship method in teaching programming for beginners*, in: Proceedings of the 42nd ACM Technical Symposium on Computer Science Education. ACM, pp. 93–98.

Wei-Qing, Q., Ming, W., Yan-Fei, Z., Bang-Quan, L., (2014). *Research on teaching gamification of software engineering*, in: Computer Science Education (ICCSE), 2014 9th International Conference on. pp. 855–860. doi:10.1109/ICCSE.2014.6926583