DOCTOR OF PHILOSOPHY

Serious games in teaching computer programming

Usage and evaluation

Abdellatif, Abdelbaset

Award date:
2020

Awarding institution:
Queen's University Belfast

QUEEN'S UNIVERSITY BELFAST

Serious Games in Teaching Computer Programming: Usage and Evaluation

by

Abdelbaset Jamal Naim Abdellatif

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

in the

Electronics and Computer Engineering
School of Electronics, Electrical Engineering and Computer Science

February 2020

Declaration of Authorship

I, Abdelbaset Abdellatif, declare that this thesis titled, 'Serious Games in Teaching Computer Programming: Usage and Evaluation' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

QUEEN'S UNIVERSITY BELFAST

Abstract

Electronics and Computer Engineering
School of Electronics, Electrical Engineering and Computer Science

Doctor of Philosophy

by Abdelbaset Jamal Naim Abdellatif

Several problems occur in higher education computing schools, such as the high attrition and failure rates and the difficulties in teaching computer programming modules. Numerous solutions were proposed to minimise and\or overcome these problems. Yet, the problems weren't solved. The focus was on the programming module as it was the module that contributed the most towards attrition and had the lowest pass rates among other modules.

The use of serious games or game-based learning for teaching computer programming has been used frequently to help overcome the difficulties. But there were no standards or guidelines to be followed to apply this method of teaching and most importantly the lack of appropriate evaluation frameworks limited the benefits of the serious games.

This research investigates the computing faculties problems in developed countries and in developing countries that have had less attention in the literature. The research provides data that has been collected from case study universities in the UK and Jordan. The results showed that the attrition rate is increasing in the universities in the developing countries. Further, the failure rate for the computer programming module is high in the universities in case studies of developed and developing countries. A total of 26 interviews were conducted with university teachers of programming modules in the UK and Jordan to highlight the main reasons behind the problems. The results showed that students struggle more with problem solving rather than coding. Also, the results showed that serious games haven't been used for teaching computer programming before apart from few attempts. The interviewed teachers showed willingness to use serious games in teaching computer programming. The results of the interviews marked loops and arrays as the hardest topics in the first programming module.

Since the introductory computer programming module was highlighted as the main cause of the attrition. The use of serious games was proposed to enhance students' understanding of programming concepts and enrich their experiences. The research aims to improve the use of serious games for teaching computer programming by exploring several success factors that are crucial for implementing the use of serious games effectively. Thus, a framework was developed and implemented through experiments using a serious game with year 1 students in case study universities in the UK and Jordan. Jordan was selected as a case study to check for the feasibility and the possibility of using serious games in teaching in developing countries. The UK was selected as a case study of developed countries for results comparison. The framework involved tests and questionnaires. A total of 82 students participated in the UK experiment and 123 students in Jordan experiment. No results were drawn from the UK experiment because most of the students dropped the experiment. The results of the experiment in Jordan showed that the best approach for using serious games for teaching computer programming is through tutorials.

Further, this research emphasises different quality characteristics that have been used in evaluating serious games and proposes a framework to evaluate several dimensions of serious games by choosing and combining appropriate quality characteristics. Pre-evaluation was applied using

the developed evaluation framework to a serious game through questionnaires with 16 university and 17 school students in the UK. The results showed that the understandability characteristic achieved the lowest rating. Thus, changes and amendments were added to the serious game, which was a series of tutorials for new users. Post-evaluation was applied using the same evaluation framework to the edited version of the serious game with 24 school students. The results showed an improvement for all the measured quality characteristics. The maximum increase occurred for the understandably characteristic. When the results were compared with the results of the pre-evaluation with school students. A significant improvement was found for 6 out of the 15 tested factors.

Acknowledgements

Since the first meeting at the foyer of the old Computer Science Building, a strong and solid relationship was established with one which I consider has had a big impact on me; its the one and only Dr Barry McCollum my supervisor. He wasn't my supervisor as much as he was my friend and he was always there for me to help and guide, and he even went further and assisted me in a very complicated personal situation. I owe him a great deal of gratitude.

Dr Paul McMullan my second supervisor offered me a great deal of help and without my two great supervisors, the delivery of this thesis would not have been possible.

I must thank my parents for their unconditional love and support. The two people who brought me to this life were nothing but perfect mentors for me through my whole life.

Also, I want to thank my brother, my sisters and my whole family especially my two wonderful nephews Amir and Qaysar, just thank you all for being in my life.

Finally, I want to thank all my friends in Jordan and in the UK and I owe a great deal of gratitude to Kelvin Old Boys the amateur league football club which I joined since I came here to Belfast and they made my time here inclusive and enjoyable.

Contents

List of Figures

List of Tables

List of publications

Parts of this work described in this thesis has been presented in the following publications:

Abdellatif, A. J., McCollum, B. and McMullan, P., (2018). "Serious Games Quality Characteristics Evaluation: The Case Study of Optimizing Robocode". *In Proc of the 20th International Symposium on Computers in Education (SIIE 2018).*

Abdellatif, A. J., McCollum, B. and McMullan, P., (2018). "Serious games: Quality characteristics evaluation framework and case study". *In Proc of the Integrated STEM Education Conference (ISEC 2018), pp. 112-119.*

Abdellatif, A. J. and McCollum, B., (2016). "A Proposed Framework for Simulation Based Learning of Inheritance". *In Proc of the 18th International Conference on Computer Modelling and Simulation (UKSim 2016), pp. 75-78.*

Dedicated to the person who literally influenced and changed my whole life, he is the ultimate definition of hope and infinite support, he is my friend, my brother, my supervisor and on top of that my father.

Dr Jamal Abdellatif

Chapter 1

Introduction

Technology has shaped and reformed almost every aspect of our life over the past decades. We are currently living in a digital world and Computer Science has a huge impact on this world. Thus, the demand on computer scientists is increasing to cope with the needs of this growing industry; however, computing faculties are facing major problems embodied by the high attrition rates and high failure rates (Dauski and Quaye, 2016; Seyal et al, 2015; Watson and Li, 2014; Corney et al, 2010; Lu and Fletcher, 2009; Gomes and Mendes, 2007; Kinnunen and Malmi, 2006; Beaubouef and Mason, 2005). Computer Science majors have been facing a sharp decline in enrolment (Ali, 2009; Benokraitis et al, 2009), where the Taulbee survey informed that about 50% fewer students entered Computer Science in 2007 compared to 2000 (Zweben, 2008). The Taulbee survey is a survey conducted yearly by the Computing Research Association to document trends in student enrolment and degree production in the United States and Canada universities that offers PhD in Computer Science, Computer Engineering or Information. The first increase in six years in enrolment for computing majors in the USA happened in 2008 with an increase of 6.2% compared to 2007 (Zweben, 2009). An increase for enrolments in computing majors has been reported each year

by the Taulbee survey with the last survey reporting an increase by 11.4% in 2017 comparing to the previous year (Zweben and Bizot, 2017). The survey reports a decrease of 12% in the awarding of BSc degrees in computing majors in 2009 compared to 2008 (Zweben, 2010). However, from 2010 to 2017 there was an increase in the awarding of the degrees. The increase of the awarding rate was expected because of the increase in the enrolments for the surveyed universities. But the survey does not report the attrition that occurs during the years. Moreover, the survey depends on the number of responses it receives. Each year, new universities are included in the survey and sometimes the previous universities can choose not to report. This makes the results inconsistent because some universities can simply not report to the survey when they have low enrolment or high failure rates. This can affect the results of the survey. In 2010/2011, Woodfield (2014) found that Computer Science was the discipline with the lowest continuation rate (91%), while student taking Computer Science formed 4.2% of the whole student body (67,847 people) included in the study. Similarly, University of West England reported an extremely low continuation rate in all computing programs. In the 2010/2011 academic year, the continuation rate was 78% and 82% in 2013/2014 (Green et al, 2016). Further, the study reported the retention or continuation rate for all the programs in the university between the years 2010 to 2015 and the continuation rate was more than 90% for all the years. Also, a study carried out by Talton and colleagues (2006) stated that about 25% of the total number of entering freshmen have dropped out of the Computer Science program by the end of their first year from 1998 to 2003. The failure and dropout rates are high, mostly in the introductory computer programming courses (Gomes and Mendes, 2007). Beaubouef and Mason (2005) identified that there is about 30%-40% attrition rate for

computing students and the vast majority of that occurs after the introductory to programming module. Also, McDowell et al (2006) stated that students change their majors after taking the first programming course in Computer Science. It has been shown that the problem occurs during the introductory to programming module, which leads to low continuation rate. Beaubouef and Mason (2005) connected the high attrition rate to poor advising, poorly planned labs, poor problem solving and maths skills and the lack of practice and feedback. Moreover, some studies have stated that the complexity of the programming languages used in the introductory courses might be one of the factors affecting the attrition rate (Manaris, 2007; Beaubouef and Mason, 2005). Furthermore, Gomes and Mendes (2007) have highlighted the need for an increased amount of practice time and students' engagement.

1.1 Difficulties in teaching computer programming

While investigating the high attrition rate in computing it has been found that computer programming attracted more interest over other Computer Science topics. Bergin and Reilly (2005) have confirmed the difficulties in learning computer programming and have concluded that this can result in high attrition and failure rates. According to Azad and Shubra (2010), the study estimated at least 25% of students drop the introductory to programming course because of the difficulty in learning computer programming. In the recent years, numerous studies have emerged to demonstrate that novice programmers have difficulties in learning Object Oriented Programming (OOP) concepts (Kunkle and Allen, 2016; Biju, 2012; Bennedsen and Caspersen, 2007; Goosen and Pieterse, 2005; Kelleher and Pausch 2005; Ragonis and Ben-Ari, 2005; Hanks et al,

2004; Robins et al, 2003; Jenkins, 2002). For example, students face several problems understanding classes, objects, recursion and inheritance (Yan, 2009). There are issues that emerge when teaching programming at an early stage, where students struggle with analysing and designing of the code (Papadopoulos and Tegos, 2012; Lopez et al, 2008; Cooper et al, 2000; Dann et al, 2000). Further, students face difficulties because of the rigid programming syntax and the large amount of time required to assemble a simple output (Sloan and Troy, 2008; Wilson, 2002).

1.2 Innovative ways of teaching programming

Lately, there has been criticism on the traditional methods of teaching computer programming implying that they don't allow students to apply their knowledge, which leads to a decrease in student retention. Several studies have stated that academic advising, involvement and engagement, well prepared teaching material, student support services and learning experiences are vital for student retention (Roberts and Styron, 2011; Barker et al, 2009; Pascarella and Terenzini, 2005). Thus, MacLean (2010) focused on the need for changing the introductory Computer Science courses teaching methods. Numerous proposed solutions were carried out. Some studies focused on students' attendance and marks, such as the one carried out by Green and colleagues (2016). They developed a system to identify and track students at risk of failing for the aim of preventing it. Some studies developed systems like the ones in the study (Galves et al, 2009). They developed an Object Oriented Programming System (OOPS), which is a problem solving environment, where students can solve OOP exercises and get instant on-demand feedback. Also, they used a web based assessment system called SIETTE as a testing system to form their own blended e-learning approach.

Other studies focused on improving the teachers' methods of delivering lectures\lessons such as the proposed teaching workshops by Porter et al (2017), which aimed to show how to be an effective teacher. Furthermore, the use of pair programming in introductory courses was implemented to contribute to a greater perseverance in computing majors (Porter et al, 2013; McDowell et al, 2006). Pair programming targets the problem of low retention rates that is caused by the low percentage of students' participation in activities. Sprint and Cook (2015), promoted group programming in a competition using decks of cards for questions.

1.3 Serious games in teaching programming

Researchers are requesting for refinement and amendment in computing education, and the use of games to inspire and motivate students (Bayliss, 2007; Wolz et al, 2006). The use of serious games in teaching computer programming was an answer to the researchers request to change the traditional way of teaching computer programming. Rosenberg and Klling (1997) associated all the challenges and problems in teaching OOP to the lack of visualisations. Thus, in order to attract more students and increase the retention, computing faculties promoted their departments with teaching in different styles and techniques. The computing faculties are using modern technology in the teaching process and presenting the students with exciting research projects such as collaborative learning, teamwork and group stimulation (Betancur et al, 2011; Hanzu-Paraza and Barsan, 2010; Phuong et al, 2008; Marin-Garcia and Mauri, 2007). Game-based learning has become a research trend in the field of educational technology (Hwang and Wu, 2011). As a result, the Computer

Science teachers are using serious games to stimulate students in the assignments and research (Corral et al, 2014; Malliarakis et al, 2014; Eagle and Barnes, 2009; Barnes et al, 2007).

1.4 Serious games in developed and developing countries

Several studies proved the effectiveness of using serious games in teaching computer programming. The vast majority of the studies were conducted in developed countries such as the USA (Zhang et al, 2013; Baker et al, 2012; Hillyard et al, 2010; Eagle and Barnes, 2008; Ross, 2002). Developed countries don't have computing infrastructure or technology budget problems unlike developing countries. Few studies considered developing countries to test the use of simulation software and serious games in teaching (AlAmmary, 2012; Al-Linjawi and Al-Nuaim, 2010). The studies found that serious games have a positive effect, which is reflected on the overall students' marks, enhanced understanding and increased interest and motivation levels. While using simulation software and serious games in teaching computer programming in developed countries, several barriers appeared such as the fragile computing infrastructure, lack of technology budget (Talebian et al, 2014; AlAmmary, 2012) and cultural issues (Farid et al, 2015).

1.5 Evaluating serious games

Evaluating simulation software and serious games plays a crucial role in learning. Before an innovative technology can impact learning, students must be able to use it. Therefore, failure to develop usable game interfaces can restrict the goal of teaching for users and can have a negative effect on the quality and benefits of a game (Pinelle et al, 2008). Further,

Emmerich and Bockholt (2016) stated that the overall goal of evaluation is to prove the game's effectiveness and suitability with respect to its designated purpose and application context. Therefore, it is vital to evaluate the dimensions of learning effectiveness, engagement and the appropriateness of the design (De Freitas and Oliver, 2006). Moreover, El Borji and Khaldi (2014) stated that the evaluation of the educational, technical and playful aspects of serious games is a prerequisite before being used and introduced in education. Rondon, Sassi and Andrade (2013) expressed the need for research aimed at assessing the educational value of computer games in students' learning and knowledge retention.

Mayer et al (2014) stated that there is a lack of comprehensive, multipurpose frameworks for comparative evaluation. Since limited games were used for educational purposes, the evaluation of serious games has depended on studies of video games (Gee, 2003; Squire, 2002). However, evaluating serious games is different from leisure games. There are several frameworks to evaluate leisure games. Yet, De Freitas (2004) stated that there is a shortage of useful frameworks for evaluating serious games. This resulted in a significant barrier for using serious games in teaching. Thus, the studies (De Freitas, 2005; De Freitas, 2004), specified the need for new assessment frameworks and methods. Before using serious games for teaching, an appropriate evaluation must take place. Otherwise, the effectiveness of the used serious game will be limited. Moreover, the suitable use of evaluation frameworks can provide guidelines for the teachers and improvements to achieve the maximum benefits of serious games.

1.6 Research aims and goals

The research aims to identify the problem of attrition and failure rates in computing majors. By exploring published studies and collecting data from universities in case study countries. Also, the research aims to target the attrition problem by identifying the main causes of it and by exploring possible solutions. The research takes into consideration the teachers opinions and point of views. The research goal is to validate the previous results about the positive impact of using serious games in teaching. Further, the study aims to highlight the best approach for using serious games in teaching computer programming, by combining and comparing several key factors that have been proved to be influential on the success of using serious games in teaching. As far as we are aware, this research would be considered the first attempt to compare different serious games success factors for the aim of finding the best approach and best practices for using serious games in Computer Science teaching. The research will also investigate the differences in using simulation software and serious games in teaching programming between developed and developing countries. The research also aims to improve serious games by developing an evaluation framework from the used quality characteristics in measuring serious games. The framework will be used to assess several dimensions of serious games by choosing and combining appropriate quality characteristics. The developed framework will be applied on an example serious game and will be used to identify possible improvements for it.

1.7 Thesis structure

The structure of the thesis is as follow:

1.7.1 Chapter 2: Literature Review

This chapter will explain what is a serious game, its definition and what serious games are used for. Then, it will explain the pedagogy of serious games. After that, the chapter will outline the usage of serious games in teaching different disciplines with a focus on Computer Science modules, especially computer programming. It will explore and compare different approaches for using serious games, several types of serious games and different age groups for the participants of the experiments. Then, the chapter will highlight the quality characteristics of serious games and presents several evaluation frameworks. It will describe and compare several attempts for evaluating serious games for various purposes, such as identifying the appropriateness and suitability of the serious game, evaluating the effectiveness or assessing the quality of it.

1.7.2 Chapter 3: Research Questions and Design

This chapter will analyse the obtained results further to form the research questions that this study aims to answer. The chapter will focus on the need for collecting secondary data, conducting interviews and experiments to fill in the gaps that were highlighted in the literature review chapter. This chapter will outline the serious game that will be used and will present the research questions for this study.

1.7.3 Chapter 4: Methodology

This chapter will explain the used methods for data collection and how it will be analysed. Firstly, secondary data will be explained. In this

research, this refers to data from universities in regards to students' numbers in computing majors in case study countries (UK and Jordan). Secondly, different types of interview techniques will be mentioned with a detailed focus on the chosen interview type for this research. Thirdly, the best approaches for using serious games will be explained, which are experiments to be conducted to find the best approach for using serious games in teaching computer programming. This section will describe key factors of using serious games, define the student sample and explain the building of the framework to be used highlighting the goals, the objectives and the analysis procedure of the data. Finally, the evaluation questionnaire section will describe the development of the evaluation framework, the targeted student sample with the tasks allocated for the participants and the analysis procedure for the data. At the end, there will be a section highlighting the limitations of this research.

1.7.4 Chapter 5: Results and Discussion

This chapter will explore the results obtained from collecting the data from the case study universities in the UK and Jordan. Then, will present the results obtained from conducting interviews with teachers of programming modules in the same universities. The chapter will explain the process of conducting the experiments in universities in the UK and Jordan to identify the best approach for using serious games in teaching computer programming. Also, will present the analysis of the data collected from running the experiments. Finally, this chapter will demonstrate how the proposed evaluation framework will be applied to the case serious game and will identify the changes and improvements recommended after running the framework. This chapter will also highlight the changes that have been made to the used serious game and will present

the results from running the evaluation framework on the edited version of the game to validate the applied changes.

1.7.5 Chapter 6: Conclusion

This chapter will provide a summary for this research. Then, the contribution of this research will be highlighted. This chapter will present the recommendation of this study and will list some potential future research paths.

Chapter 2

Literature Review

This chapter presents different definitions for serious games and then proposes a definition based on previous research contributions. Then, the pedagogy of serious games is explained. The chapter highlights the use of serious games in various disciplines and provides examples. Then, defines the usage of serious games in teaching various computer science topics and provides examples. The chapter focuses on the use of serious games in teaching computer programming. This section explores different approaches in using serious games, several types of serious games and different age groups for the participants of the experiments. Finally, the chapter highlights the serious games evaluation frameworks, describing several attempts for evaluating serious games for various purposes.

2.1 Serious games definition

Serious games have similar terms in the literature. Smith (2013) listed some of the popular terms, which are Educational Games, Edutainment, Simulation, Virtual Reality, Digital Game-Based Learning, Immersive

Learning Simulations, Tactical Decision-making Simulation and less popular terms, which are Alternative Purpose Games, Impact Games, Persuasive Games, Games for Change, Games for Good, Synthetic Learning Environments and Game-Based "X".

The serious games studies are diverse in the literature and serious games have various definitions. Most of them describe and demonstrate serious games as interactive, entertaining, goal-focused and competitive (Tobias and Fletcher, 2007; Vogel et al, 2006; Driskell and Dwyer, 1984). Abt (1970) provided one of the earliest definitions and descriptions of serious games, where serious games were described in these terms: "these games have an explicit and carefully thought-out educational purpose and are not intended to be played primarily for amusement. This doesn't mean that serious games are not, or should not be, entertaining". Alternative definitions for serious games are available in the literature such as Michael and Chen (2006) where they defined serious games as "a game in which education (in its various forms) is the primary goal, rather than entertainment". Instead, they offer educational content to users in an enjoyable way by simulating scenarios which promote learning. Accordingly, Egenfeldt-Nielsen (2006) stated that learning through games is the prime goal instead of pure entertainment in serious games. Similarly, (Greitzer et al, 2007; Gee, 2003) stated that serious games are designed to have an impact on the target audience, which is beyond the pure entertainment aspect.

Sitzmann (2011) defined computer-based simulation games as "instruction delivered via personal computer that immerses trainees in a decision-making exercise in an artificial environment in order to learn the consequences of their decisions". For Zyda (2005) a serious game is "a mental contest, played with a computer in accordance with specific rules, that

uses entertainment to further government or corporate training, education, health, public policy, and strategic communication objectives".

Michael and Chen (2006) had another definition for serious games, which is "games that do not have entertainment, enjoyment or fun as their primary purpose" and they also stated that "serious games are games that use the artistic medium of games to deliver a message, teach a lesson or provide an experience". Yet, Michael and Chen (2006) highlighted the importance of fun and enjoyment in serious games where 80% of respondents for a survey of serious games developers, educators and researchers felt the element of fun which can be referred to enjoyment too as important or very important. Moreover, Stokes (2005) defined serious games as "games that are designed to entertain players as they educate, train, or change behaviour".

Another definition of serious games was provided by Bergeron (2006) as "interactive computer application, with or without significant hardware component, that has a challenging goal, is fun to play and engaging, incorporates some scoring mechanism, and supplies the user with skills, knowledge, or attitudes useful in reality". Further, the study by (Laamarti et al, 2014) defined serious games as "an application with three components: experience, entertainment, and multimedia". Figure 2.1 describes their definition. The figure also demonstrates the differences between serious games and several terms such as training simulation, computer game, and sports. Game Based Learning which is another popular term of serious games was defined by (Livovsky and Poruban, 2014) as "a type of game play that has defined learning outcomes".

There are various definitions for serious games and its different terms, but they all agree on the importance of the delivered educational impact.

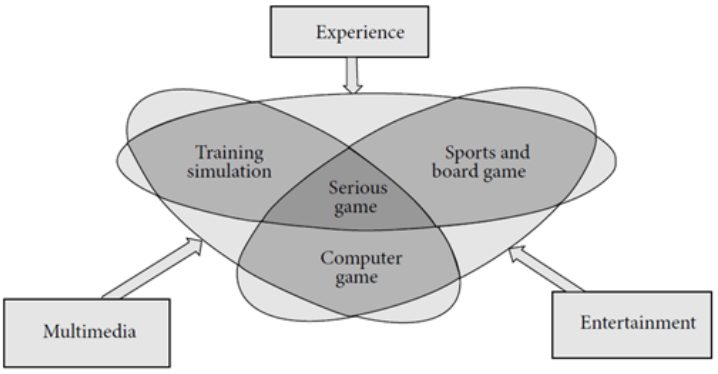


FIGURE 2.1: (Laamarti et al, 2014) Definition of serious games.

Moreover, the definitions highlight the importance of different characteristics that must be present in the serious games to engage, motivate and immerse the users such as entertainment and enjoyment. If a serious game doesn't engage or motivate the user in an interactive and entertaining way as a video game does, the user will not be immersed and focused while using the serious game. Thus, the serious game will fail to deliver its educational content to the user or the benefits of playing the game will be minimised. To conclude, we can define serious games as the use of non-software, software or hardware or a mix in an engaging way that motivates and engages the players to use it to deliver an educational content for learning, raising awareness and/or training purposes in an entertaining and interactive way that has less priority than the educational content itself.

2.2 Serious games pedagogy

Bloom (1956) identified three domains of learning, which are cognitive, affective and psychomotor. The cognitive domain involves knowledge and the development of the intellectual skills and it consists of six major categories, which are knowledge, comprehension, application, analysis, synthesis and evaluation. The categories are listed in order in terms of its

difficulty. The cognitive domain has been edited by the study (Anderson et al, 2001), where the names of the categories have been changed from noun to verb forms with a slight change on their order as shown in the figure 2.2.

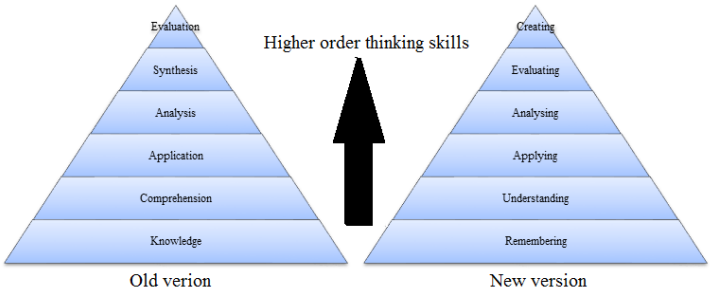


FIGURE 2.2: The two versions of the cognitive domain in Bloom's Taxonomy.

Tang, Hanneghan and El Rhalibi (2007a) focussed on defining the pedagogical elements and components that can be constructed using components in digital games to produce serious games that can be used for teaching purposes. The pedagogy elements of the serious games are:

1. The properties and behaviour of in-game components.

   This refers to the representation of the objects and the actors of the serious game, where the user can learn by observing the in-game components. The amount of knowledge gained depends on the details provided by each serious game.

2. Relationship between in-game components.

   This refers to the relationships between the objects and actors, in which the user can learn by interacting with in-game components and develop knowledge.

3. Tasks and problems in a given scenario.

   This refers to the tasks that the user is asked to complete. To be able to do that, the user must have knowledge in the properties and the relationship of the in-game components of the serious game.

Figure 2.3 shows the three pedagogy elements of serious games on Bloom's Taxonomy.

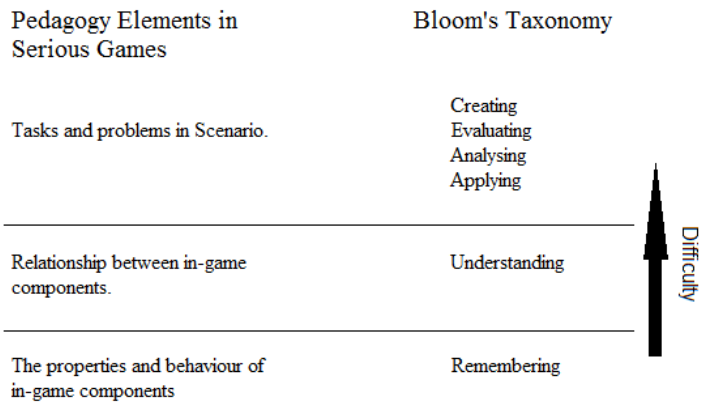


FIGURE 2.3: (Tang et al, 2007a) pedagogy elements of serious games on Bloom's Taxonomy.

According to (Tang et al, 2007a), the components of pedagogy in serious games are game screens, cut-scenes, game tutorials and game levels. Game screens are used to display menus, game objectives and statistics. It is represented by text or graphical content. Game screens are used pedagogically for informing the user of the learning objectives. Cut-scenes is a storytelling technique that can be used in serious games to provide the user with information visually using animation. Cut-senses are used to immerse the user in the game. Game tutorials guide the user to complete a simple task for the goal of informing the user of the options that can be used and to familiarise the user of the game. However, it is not mandatory for serious games to have tutorials. Game levels refer to the different levels or tasks that the user complete, in which every time a level is completed, the next level will be more difficult, more challenging and the scope of the level will be greater.

Gagne (1965) defined Nine Events of Instructions as conditions for learning. Table 2.1 highlights the priority of the pedagogy components in relation to the nine learning events as reported by (Tang et al, 2007a), where H means high, M means medium and L means low. Similarly,

Becker (2005) clarified how games match with Gagne's theory of learning, where the study applied each of the nine events to games to show how game features are linked and match the nine events instructions. Further, the study (Mavromihales et al, 2018) conducted an experiment on Mechanical Engineering students taking Assembly module to evaluate the effectiveness of game-based learning. The results of the used questionnaires showed that the gaming activity had met with Gagne's nine elements of instructions.

TABLE 2.1: Components of pedagogy in relation to Gagns Nine Instructional Events.

Gagnes Nine Instructional Events	Game Screen	Cut-Scene	Game Tutorial	Game Level
Gaining Attention		H		
Informing Learners of the Objectives	H	M		
Stimulating Recall of Prior Learning	M	H	L	L
Presenting the Stimulus		H	M	L
Providing Learner Guidance		M	H	
Eliciting Performance			H	H
Providing Feedback	M		H	H
Assessing Performance			L	H
Enhancing Retention and Transfer				H

Several theories and models have been proposed that focus on the pedagogy of serious games. For example, the study (Garris et al, 2002) proposed an Input-Process-Output model highlighting the relation between game features, instructional content and the player from a motivational and engagement perspective. The input represents the instructional content and the game features and the output represents the learning outcome. The key component of this model is the process which represents the iterative game cycle that will keep the user motivated and engaged while completing the game objectives and acquiring the learning content. Kiili (2005) proposed an experiential gaming model focused on linking gameplay with experiential learning to facilitate flow experience.

The model is limited only to link the educational theory with game design. The study recognises, but does not consider storytelling, appropriate graphics and sounds and game balance in the proposed model.

Bellotti et al (2011) focused on the presentation and user interaction in relation with the pedagogy of serious games. The study showed that the pedagogy of serious games can be affected by the game interface, the method or the level of interaction, the game feedback and reward and by the type of evaluation of performance. Paras and Bizzocchi (2005) listed multiple characteristics that are important for serious games effective learning, such as motivation, flow, learning environment, play, immersion and reflection. The focus was on reflection which is the purposeful and fundamental thoughts that lead to an idea being developed. The study stated that without reflection the learning cycle is unable to generate new conclusions and actions effectively.

Lepper and Malone (1987) highlighted four key attributes that serious games must have, which are challenge, curiosity, control and fantasy. The essence of such attributes leads to more engagement in the gameplay and more gained knowledge. Furthermore, Tang et al (2007b) stated that games encourage the users to learn from their interaction with the game and frequently challenge them to employ their gained knowledge in solving problems. Students engagement is essential and vital for educational success (Shernoff et al, 2014; Fredricks et al, 2004). In which, engaged and motivated students inspect and explore their understanding rather than only receiving the educational material (Mestadi et al, 2018). Thus, serious games engage the users by using the fun or enjoyment factor to immerse the users in an active learning environment (Garris et al, 2002).

The pedagogy of serious game has been addressed by several studies and

it was shown how serious games features match with the well-established learning theories. The learning process when using serious games starts by observing the game features. But the deep learning and understanding of the educational content occurs while completing the given tasks in an iterative and cyclic manner. However, for the learner to complete the assigned iterations, the game must engage and motivate the learner. Several studies have been devoted to evaluate and measure different attributes of serious games such as motivation (Wangenheim et al, 2012; Miller et al, 2011), engagement (Dunwell et al, 2013; Pourabdollahiana et al, 2012) and the educational content (Manero et al, 2013; De Freitas and Oliver, 2006).

In the process of learning computer programming there are two critical and different parts, which are programming knowledge and programming strategies (Davies, 1993). For example, programming knowledge refers to being able to know how the for loop works. Programming strategies refers to being able to apply and use the for loop in a program suitably. Thus, students with good programming knowledge but poor programming strategies will face difficulties in developing a program using the different programming concepts.

Debabi and Champagnat (2017) identified two approaches for using serious games for teaching computer programming, which are learning programming through game development and learning programming by playing games. Another approach was identified by the study (Shabalina et al, 2017), which is learning programming by the development of games for learning programming. The study (Wallace et al, 2010) offered more details about the methods used for teaching using serious games and divided them into four approaches, which are learning by implementing games, learning by writing programs that implement a critical aspect of

the game, learning by writing programs that act as a player in an existing game and learning by playing educational games.

Laporte and Zaman (2018) stated that the research about serious games for programming focused mainly on what is being taught. Very little known about how serious games for teaching computer programming afford learning and playing by design. The pedagogy of serious games is missing in the literature. Most of the available research highlights the importance of different quality characteristics of the serious games on teaching computer programming, such as motivation (Yassine et al, 2017; Long, 2006; Garris et al, 2002), engagement (Yassine et al, 2017), flow (Seng and Yatim 2014; Zapusek and Rugelj, 2013), longevity (Gibson and Bell, 2013) and the competitive nature of the game (Zapusek and Rugelj, 2013; Hakulinen, 2011). According to Sizmann (2011), the games that teach actively rather than passively produce better results. Teaching actively through serious games means having the gameplay teach the content, where teaching passively means having the teaching in the game separate from the gameplay.

However, the development of serious games for teaching computer programming is designed based on the well-established learning theories. For example, the described game design in the study (Yassine et al, 2017) for teaching pointers in C programming language. The design was based on Solo taxonomy which stands for Structure of Observed Learning Outcome. Solo taxonomy acts as a model of learning that defines the level of understanding and it is used by teachers to design the pedagogy based on the students level of understanding. Further, the study by Zapusek and Rugelj (2013) stated that serious games can be used to teach computer programming concepts after examining the mechanics behind the

concept to be taught and include it in a suitable game activity which resembles its logic. Similarly, AlgoGame designed by Debabi and Champagnat (2017) to introduce students to algorithmic concepts. The game was designed in which all the taught concepts were linked and related to one or more missions.

The pedagogy of serious games has been addressed by several studies. However, the pedagogy of serious games for teaching programming is missing in the literature. In the development phase of computer programming serious games, the studies depended on the established learning theories. Thus, the focus was on the characteristic of serious games, which will decide if the game will be used by the users and if they will benefit from it. The characteristics of serious games are explored and explained in section 2.7. The following section will address the usage of serious games in different areas.

2.3 Serious games usage

Serious games have been used in many areas, such as military (Djaouti et al, 2011), government and politics such as FloodSim, Serious Policy and Budget Hero (PlayGen, 2016), language (Alyaz et al, 2017), health-care (De Paolis, 2012; Sabri et al, 2010), business (Fotiadis and Sigala, 2015; Loon et al, 2015; Nguyen, 2015), children teaching for reading, maths, science and art (Age of Learning, 2018), archaeology (Anderson et al, 2009) and music (Maria et al, 2016; Bergomi et al, 2013). A market study showed that the worldwide serious games market is worth 1.5 billion in 2010, with a growth rate, over the last two years, nearly 100% per year (Alvarez et al, 2010).

The military has the longest history in using serious games for training. Battle and war games help the soldiers to increase their skills in the war strategies and techniques and to become better planners. The first serious game that was developed for military is Army Battlezone and it was designed by Atari in 1980. Many games have been developed for military such as Harpoon 3, Doom, WarCraft, TaCops and Arma 2. These simulation games supply training to take place in an appealing style without a high cost, long-time of training sessions and the side effects of the dangerous mistakes that would happen in the real world (Ulicsak 2010; Susi et al, 2007; Michael and Chen, 2006). A first-person shooter game named America's Army was developed by the US army and it was released for the public over the Internet in 2002. The game simulates military training exercises and combat assignments. The goal of the game was to promote the American army and to serve as a recruitment tool for young people between the ages of 16 and 24 (Djaouti et al, 2011).

Health-care domain in serious games is growing fast and it is very common (Sardi et al, 2017; Kapralos et al, 2015). These games define health and health-care including meditation, biofeedback, rehab, patients' cases and conditions, nursing and pain management. Serious games in health-care increase personal treatment for the clients and professional practice that will show in detail all the movements, pain pressure, all kind of diseases and its solution for medication. There are various examples on health-care games, such as Hungry Red Planet, Free-Dive, S.M.A.R.T Brain-Games, Brain-training by Nintendo and Triage Trainer (Smith, 2013; Ulicsak 2010; Ferriera, 2008; Susi et al, 2007; Michael and Chen, 2006).

An example of serious games in business area is The Small Business Game. It is a serious game that can be accessed online for free, where the user runs their own football retail store. The game is supported by

additional classroom resources such as handouts, presentations and templates. The game was designed for schools and universities teaching but it can be customised to be used by organisations, such as banks.

An example of governmental serious games is FloodSim. The game was developed by PlayGen Ltd and commissioned by Norwich Union to raise the awareness of issues about flooding policy and citizen engagement in the UK. The player represents the role of a flood policy strategist who needs to set a strategy over the period of three years related to the risk of future flooding. The game is explored further in (Rebolledo-Mendez et al, 2009) study.

Below are three examples with figures for the use of serious games in different areas.

FIGURE 2.4: Adventure German The Mystery of Nebra game is an example of serious games for teaching German language. The user will be solving a puzzling mystery in an adventurous way while learning the German language.

FIGURE 2.5: Rome in Danger is a serious game that aims to teach the history of ancient Rome.

FIGURE 2.6: Climate Challenge is a serious game to raise awareness of climate change and its terrible effects.

Serious games have been used in different domains and areas and few examples were provided in this section; the next section will explore the use of serious games in Computer Science domain.

2.4 Serious games in computer science

The use of serious games in teaching computer science subjects is diverse. According to the systematic review by Battistella and von Wangenheim (2016), software engineering has the most number of serious games as a knowledge area of computer science. The second knowledge area is programming fundamentals.

Studies used serious games for teaching computer science topics, such as software project management (Misfeldt, 2015), human computer interaction (Do and Lee, 2009), software engineering (Wangenheim et al, 2012; Mittermeir et al, 2003), security (Jordan et al, 2011; Sheng et al, 2007), operating systems (Hill et al, 2003), computer assembly (Hou and Li, 2014), computer networks (Melero et al, 2012), computer architecture (Melero et al, 2012), database (Connolly et al, 2006), artificial intelligence (Syberfeldt and Syberfeldt, 2010; Hartness, 2004), distributed systems (Wein et al, 2009), algorithms (Cesare, 2013; Rossiou and Papadakis, 2007) and computer programming. The following section describes briefly a few serious games that have been used for teaching different computer science topics.

Do and Lee (2009) developed a 3D Lego game, which is an augmented reality game inspired by the real LEGO game. The game has multiple levels and the players or the users are asked to reconstruct 3D models. The players will use their hands to control the physical makers. Another game was developed by (Mittermeir et al, 2003) called AMEISE, in which students represent the role of a technical project manager. Students can hire and fire employees, structure the project and assign tasks. Students are challenged to manage a project based on a model chosen by the instructor. The instructor chooses the number of trials (simulation runs) to solve given tasks. Students can analyse and learn from previous

simulation runs and then make changes to the used strategies. Students can measure their own success using the game self-assessment feature.

The study (Wangenheim et al, 2012) developed a board game named DELIVER! that aims to teach earned value management in monitoring and controlling the execution of a software project. The game is played in groups of four pairs of players on one game board. The duration of game-play is around 90 minutes divided into four steps and it involves explanation of activity, project planning, project execution & monitoring & control and debriefing.

The study (Sheng et al, 2007) developed a serious game called Anti-Phishing Phil. The game's main objective is to teach how to identify phishing URLs, where to look for signs for trustworthy or untrustworthy sites in web browsers and how to use search engines to find genuine sites. The user represents a young fish named Phil living in the Inter-web Bay. Phil wants to eat worms but must be cautious of phishers that try to trick him with fake worms which represent phishing attacks. Each worm is associated with a URL and Phil's job is to eat all the real worms, which have URLs that represent legitimate websites and reject all the bait, which have phishing URLs before running out of time. Another example of the usage of serious games in computer security topics is the study (Jordan et al, 2011). They developed a single player serious game called CounterMeasures, where the player is guided through several missions, in which each mission teaches a new aspect of security. CounterMeasures uses a real and interactive shell for input and targets a real server for exploits to provide an environment resembling security systems currently deployed. Players are given a fully functional shell that runs commands while working on a mission.

Another study (Melero et al, 2012) designed a mini puzzle solving game.

The game focussed on teaching how the data travel through the network, how data are sent from one computer to another and how to connect different devices to have ADSL in home. Routers, IPs, Computer ports, browsers or frames are examples of puzzle pieces in the game. The player has to solve several problems on different levels of difficulty. Hints and feedback are provided for the players to complete the objectives. Also, the study (Melero et al, 2012) designed another mini puzzle solving game. The game focussed on teaching the main elements of the motherboard and Boolean logic. The player has to solve several problems on different levels of difficulty. Hints and feedback are provided for the players to complete the objectives.

Connolly et al (2006) developed an online games-based learning approach for teaching database designing concept. The design consists of 3 levels that form the learning environment, which are the online learning units/topics that introduce the concepts to be explored, the visualisations that enhance learning by providing an animated walk-through for examples and the simulation game that provides a real-world simulated environment to apply skills and techniques.

Syberfeldt and Syberfeldt (2010) study describes a game for teaching the concept of production optimisation and how it can be improved using artificial intelligence techniques. The simulation shows a physical production process in the form of a miniature factory for producing bubble gums using Lego bricks and Mindstorms NXT. The player represents the production manager and his/her task is to find the best configuration for the production process, such as the production flow and buffer sizes to maximise profit. The player has to develop artificial intelligence algorithms to find the optimal configuration. The study (Wein et al, 2009) developed a game called PDConsole where a team of students can play the game and the players represent an administrator who is responsible

for maintaining the distributed system. The site displays mock advertisements when different pages are viewed and the game generates artificial traffic against the site. The game begins when the game administrator breaks parts of the system in some way. Players notice that the system's performance has degraded because of ads and revenue drop. Then, the players have to identify the problem and fix it.

Cesare (2013) developed a serious game where the idea of the game is based on World of Warcraft game and supports single and multiplayer modes. The game also allows for spectators, which can be a teacher. This game teaches sorting algorithms using bubble and merge algorithms. The users have to collect boxes and sort them based on their values. Another example of serious games for teaching algorithms is the work by Rossiou and Papadakis (2007). They created Algorithms Recursive game by adopting the generic Snakes and Ladders shell of Simulation Advanced Game Environment. The game has 44 questions with three levels of difficulty which covered several aspects of recursive algorithms. The game aims to revise the recursion algorithm for the students in which 2 students play the game at a time against each other.

Different serious games have been developed to be used in teaching various computer science subjects. The examples provided previously shows that different game genres have been employed such as role play, puzzle and simulation. Also some of the games supported multiplayer. Pre and post tests and questionnaires were used to evaluate the serious games. All the results stated that using the serious games was beneficial for the students to increase their understanding of the covered topics and increase their motivation. The following section will cover the use of serious games in teaching computer programming.

2.5 Serious games in teaching computer programming

The use of serious games and simulation software for teaching computer programming have been used in various ways for the goal of enhancing students' understanding of the OOP concepts and add to their overall experience. Some studies used non-software games (Sprint and Cook, 2015), while others used software and interactive games (Corral, 2014; Zhang, 2013). Several types of research used open-source projects (Al-Linjawi and Al-Nuaim, 2010; Liu, 2008), few developed their own projects and games (Baker et al, 2012) and others used developed games (Malliarakis et al, 2014; Eagle and Barnes, 2008) to apply on students.

An example of a non-software game is the study (Sprint and Cook, 2015). They presented a gamified approach to engage students in the classrooms, it consists of two decks of cards one for questions and the other for bonus. There are different levels of questions, where students are required to code the problems on the cards to get points and proceed in the leaderboard. They tested it on 15 students taking a C programming course, where the students were put into groups. They have conducted pre- and post- surveys on the students. The pre-survey revealed that 100% of the students are interested in trying an alternative to lecture-based learning. The post-survey showed that 100% of the students enjoyed the game and 83.33% agreed that they prefer the gamified approach over the traditional approach. They concluded that the game creates a high amount of learning activities and is well anticipated by the students. Also, Ross (2002) used puzzles as assignments for undergraduate students to cover arrays using Java. The study stated that using games as an assignment can be more interesting than a typical task. Moreover, it forms an incentive for students to ask questions.

Corral et al (2014) conducted a research on 30 first-year students in the school of Engineering taking Fundamentals of Computer Science course. The students have been divided equally into two groups. An experimental group, where the students took a C# OOP course using Sifteo Cubes as a technological resource that offers interaction with tangible user interaction. The other group was a control group, where the students took a standard C# OOP course. Sifteo cubes is an interactive gaming platform, it consists of a small block with clickable screens that interact with users. The tasks of Sifteo cubes are controlled by code written in C# so it can be used in teaching as shown in figure 2.7. Both groups took two multiple-choice exams to evaluate the understanding of basic OOP concepts. The findings of the study showed that the experimental group students achieved a higher interest level and felt more motivated. In addition, they got overall better marks since they achieved an average of 7.44 out of 10, where the control group students achieved an average of 6.21 out of 10. Also, the experimental group spent less time on the test than the control group by 45 seconds, where they spent (12.35, 13.1 minutes, respectively). Similarly, Markham and King (2010) used personal robots for teaching introductory to programming course. The results showed an increase in the motivation and interest for the students who used the robots.

Cliburn (2006) conducted an experiment on 33 students, 9 of which were females, who were taking an introductory to programming course in C++. The students have to hand in five assignments during the semester and they were given the choice to choose from two different assignments each time. One of them is a game-based question while the other is just a regular question. The aim was to measure the effectiveness of computer games as programming projects over non-games projects. The

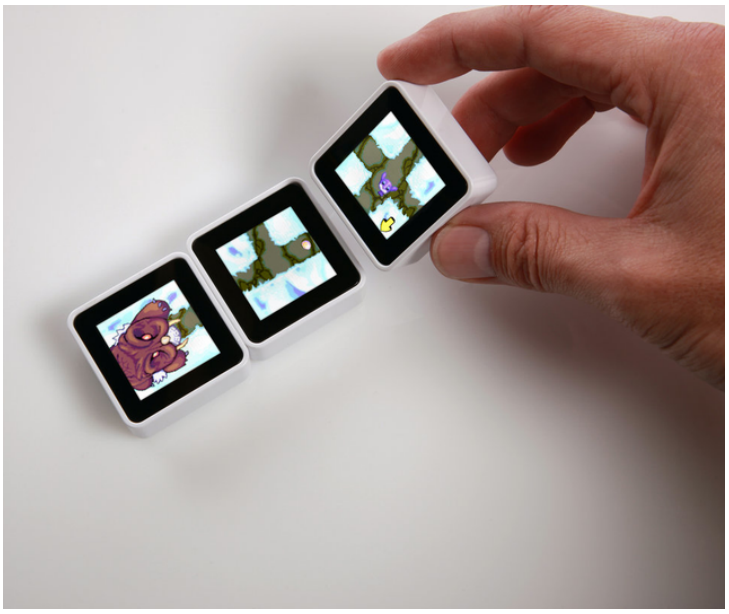


FIGURE 2.7: The Sifteo Cubes.

results showed that 78.9% of the submitted assignments were the game-based assignments; the male students chose the game-based questions more than the female over the regular questions (84.3% and 65.9% respectively). However, female students chose the game-based questions more frequently than the traditional questions. Similar to Leutenegger and Edgington (2007), the study concluded that using games in teaching has increased the attraction of students. Moreover, female students prefer game-based assignments than the traditional assignments, which mean they do like games. Although this may be true, Gurer and Camp (2002) stated that women don't like using games. The average score of the game-based assignments was 89.1%, where for the non-game was 95.1%. This showed that the game assignments did not improve the students score. However, the study refers that to the best students took intentionally the harder question. The harder question was the non-game question due to the analysis required to solve the problem because students don't have knowledge about it compared to a game. The students thought that solving the harder questions will benefit them more. This was based on an informal interview with one of the best students in the

class. They also conducted a survey on the students and all of the students preferred the idea of having a choice of the assignments they are doing. 80% of the students stated that game-based assignment is their favourite. Moreover, 84% of the students stated that the game questions provided extra motivation for them to complete the project with high quality.

Miljanovic and Bradbury (2017) designed and developed a serious game called RoboBUG that teaches debugging in C++ programming language and can be customised to different programming languages. The game was tested by three separated case studies with 23, 5 and 14 undergraduate students respectfully. The first case study was between a controlled group of 12 participants that completed an assignment and an experimental group of 11 participants that used the game. The results showed no significant difference in the achieved learning outcomes between the 2 groups. The second case was a 20 minutes interview with the participants after playing the game to evaluate the playability of the game. The results showed that the participants enjoyed playing the game despite some playability issues such as control problems. The third case was formed out of a group of 14 students who played the game for one hour and completed a pre- and post-tests. The results showed an improvement in the participants tests score after playing the game, specially for participants who scored a low mark in the pre-test.

In the study (Zhao et al, 2019), a 2D drag-and-drop serious game called the Restaurant Game that focuses on the topic of structure in C programming language was designed and implemented. The game was used by 90 first year students in the introductory software development module in which 77% of the participants were male and 33% were female. Pre- and post-tests were conducted, and each test was formed of 4 questions. A t-test was applied and showed a statistically significant improvement in the students' performance in the post test. A post game survey was

conducted which focused on the usability of the game and the participants feelings towards the game. 64% of the students agreed that the game helped them in understanding the covered topics. Further, the study (Baker et al, 2012) developed a Java programming game called Garden Gnomes From Planet 9. The game was developed to teach arrays and loops and it was used with Computer Programming 2 class. The study conducted pre-post-tests along with a survey of 19 students, 9 of them were females to measure the effectiveness of the game on enhancing students understanding of loops and arrays concepts. The study showed that most of the students exhibited a significant improvement and they left a positive feedback in the survey. Furthermore, the students showed that they are interested in learning concepts using this method. Figure 2.8 shows a screen shot of the game.

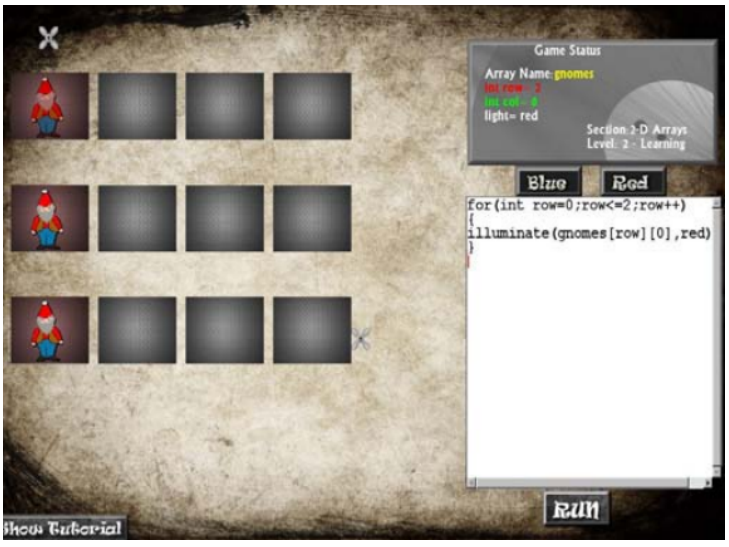


FIGURE 2.8: Garden Gnomes From Planet 9 screen shot.

The research work presented in (Mathrani et al, 2016) investigated the effectiveness of game-based learning in teaching computer programming by conducting a study on students taking a computing diploma. The study used a serious game called LightBot 2.0, and the students were divided into groups. Group 1 comprised of 20 students who didn't take the programming module yet. Group 2 contained 24 students who had

recently completed the programming module. The study included two surveys to be completed, one after completing playing the game and the other one after completing the degree. The results showed that game-based learning is a useful learning strategy before and after subject is taught. The students agreed that using games for learning is fun, effective and help in understanding difficult concepts. Inspired by Lightbot and another game called Cargo-bot, Agalbato and Loiacono (2018) developed a game called ROBO3. ROBO3 can be used to teach programming concepts, such as conditions, loops and functions. The game was played by 80 first year students taking fundamentals of computer science course. The students used the game to create simple programs that the robot will execute to solve the puzzle and the game was given to the students one month before their final exam. The game was tested by the students, in which they played the game and then filled in a survey. The results showed that most of the students played the game when they almost finished studying for the final exam. Moreover, the students found the game to be useful to improve their understanding of programming concepts. Similarly, Hinds et al (2017) explained the design and the development of a game called RunjumpCode that was developed to teach computer programming in C. The game is 2D and the idea was adopted from the famous game Super Mario Bros. The game provides challenges and puzzles, which enhance the programming knowledge of the user. Evaluating the effectiveness of the game is yet to be conducted using pre- and post-tests with a control and an experimental group.

Kumar and Sharma (2018) developed ProLounge (Programming Lounge), which is a hybrid online learning application that serves as a mobile and web app. ProLounge provides the users with three interactive modules

to learn, play and assess themselves. The results from running an experiment with 47 undergraduate students showed that students feel more focused and enthusiastic when learning difficult programming concepts using an online gamified platform. Another gamified approach was used by Zainol Abidin and Kamaru Zaman (2017), in which the study conducted a survey on 120 undergraduate students taking a computer programming course. The students learnt programming by using a game-based classroom response system called Kahoot. Kahoot provides a tool to create and share quizzes with other users. The results showed that more than 90% of the students enjoyed using Kahoot and it improved their understanding of computer programming. Similarly, the studies (Fotaris, 2016; Fotaris, 2015) used Kahoot and a class version of Who Wants to be a Millionaire in teaching computer programming. The results showed that using this approach helps in solving common computer programming language misconception and motivate students. Also, the results showed that the students attendance increased since this approach of teaching was introduced.

The study (Zhao et al, 2018) aimed at assessing whether the technology enhanced pedagogies help to engage students with STEM related subjects. The study covered a whole semester of 12 weeks with students in programming course in three universities across Europe with 150 participants. The study conducted pre- and post-tests, pre- and post-motivation questionnaire and a usability questionnaire. Four serious games were developed which are Variable game, Function/method call game, For loop game and Structure game. The results of the questionnaire reported that 54% of the mature students (aged 25 and above) and 74% of the young students (aged 17-20) would like to use more technology in the classroom when learning STEM subject. The study didn't report the results or the analysis of the pre- and post-tests. The study by Jordaan (2018) described

the design and use of a board game for year 1 students taking introductory to programming in Python programming language. The game was used in 8 lectures with 8 students participating in each lecture. Data were collected through semi-structured interviews. The results showed that students enjoyed playing the game and the game increased the communication and the socialisation within the participated students.

Malliarakis, Satratzemi and Xinogalos (2014) conducted a study on 22 university students of Applied Informatics Department. The students played a Massive Multiplayer Online Role Playing Game (MMORPG) named CMX to learn programming arrays in C programming language. A questionnaire was distributed to evaluate the game efficiency in teaching arrays. The study showed that students have increased their understanding and knowledge levels by playing the game. Equally important, the majority of the students stated that they want to use other educational games in learning programming due to enjoying this way of learning and because they felt motivated in completing the required tasks. The game was used again in which the study (Malliarakis et al, 2017) conducted an experiment on 76 first year students in the Applied Informatics department. The study aimed at evaluating the learning effectiveness as well as the entertaining and motivating elements of the serious game that teaches C programming. However, this study used pre- and post-tests with questionnaires and the results showed that most of the students were entertained by playing the game while learning. Further, the results showed that students had a positive attitude for using this learning approach to learn other computer programming concepts. Figure 2.9 shows a screen shot of the game and the CMX editor. Similarly, Eagle and Barnes (2008) conducted pre-post-tests along with a survey of 21 students taking the CS1 course. They divided the students into 3 groups where only 1 group that contains 9 students took both of the tests and played a game called

Wu's Castle that teaches arrays and loops in an interactive way. The study results showed that playing the game has statistically significant learning gains over the traditional way of teaching. Figure 2.10 shows a screen shot of the game.

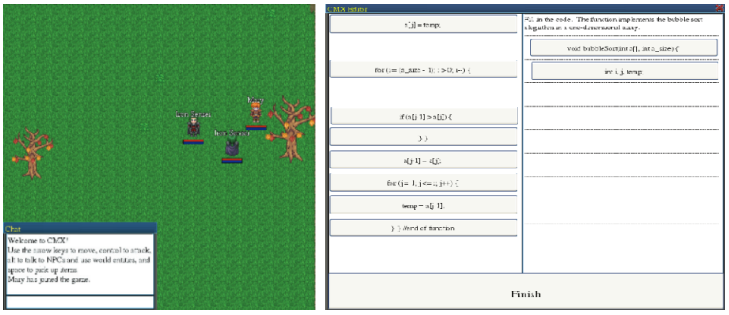


FIGURE 2.9: CMX game and editor.

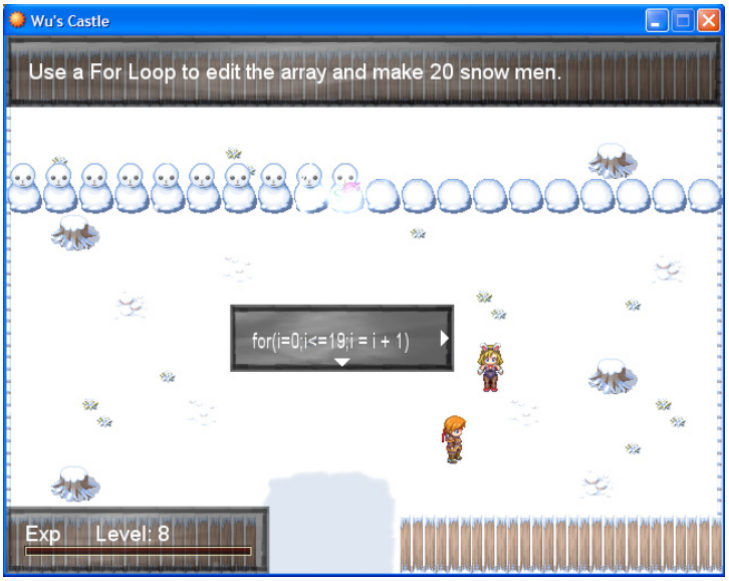


FIGURE 2.10: Wu's Castle game screen shot.

The study (Zhang et al, 2013) conducted pre-post-tests on 14 students after taking a game-like module called Java Ninja which is a game to help students understand inheritance in OOP. Figure 2.11 shows a screen shot of the game. A survey was conducted and generated 10 respondents to investigate whether such game-like module will enhance and improve students understanding. The overall students' feedback was positive, in which all the students enjoyed the game and want it to be involved in covering other concepts in learning. Also, most students showed significant

improvement, where the percentage of students who passed the post-test was 21% more than who passed it in the pre-test. Moreover, the students rated the game module as enjoyable to play, provided relevant feedback and they recommend it to others. Likewise, Al-Linjawi and Al-Nuaim (2010) used Alice in a Java programming course for undergraduate students in Computer Science Department at King Abdulaziz University in Saudi Arabia. 24 students volunteered to play the game and formed the treatment group and another 21 students were chosen based on their GPA similarity with the treatment group to form the control group. The treatment group used Alice two hours a week for 7 weeks and they were asked at the end to develop a project. Pre-Post-tests took place on both groups to measure the effectiveness of using Alice along with a survey for the treatment group. The study concluded that the treatment group showed better understating of the OOP concepts especially inheritance since the project was about it. Also, the study stated that Alice helped students to comprehend the intricacies of OOP.

FIGURE 2.11: Java Ninja game screen shot.

Watson and Lipford (2019) used a game called Ruby Warrior, which is a serious game that teaches Ruby programming language and few artificial intelligence concepts with software engineering students. The game has 10 levels represented by 10 floors where the game character

has to go through. It starts by solving problems by writing one line of code and advances to use concepts such as conditions and loops. The game has been used in two different semesters by 185 students, in which 154 were male and 32 were female. The game levels were divided into mandatory and optional sections, where only 2 students didn't complete the mandatory levels and 42.78% of the students completed 1 or 2 optional levels. A survey for the perception of the game was completed by 103 students. The results showed that 55.34% of the students were motivated by the game to learn more topics. While some students may not have found the game motivating, there were students who showed that the game motivates them to use it even if its not mandatory for the course credit. Another study by Erol and Kurt (2017) was conducted on university students taking the programming 1 course. 52 students were divided into two groups randomly (control and test). During the first 7 weeks, the students were taught the programming logic and learnt basic programming structures and then C programming language was introduced. Students in the test group were instructed using Scratch in the first 7 weeks, where students in the control group were instructed using flow-charting and problem-solving activities. Pre- and post-tests along with questionnaire were used to examine students' motivation and their programming achievements. The results showed that programming achievements scores for both groups increased. However, the increase was significantly different in favour of the test group. Moreover, the motivation scores decreased in the control group but increased for the test group. Scratch was also used in the study (Ouahbi et al, 2015), in which a survey was conducted on 69 high school science major student. The students were divided into groups and the students in the treatment group used Scratch to create simple games. The result of the survey conducted at the end of the experiment showed that students in the treatment group

desire to continue their studies in programming was 65%, where in the control group was only 10.3%. Using Scratch as an environment for learning programming highly motivated students. Similarly, Topalli and Cagiltay (2018) conducted a study during one academic semester period on students taking the introductory to programming module in C programming language. A group of 48 students took an enriched introduction to programming course, where the last 15 minutes of each laboratory activity were allocated to work on Scratch for 7 weeks. Then, the students were asked to develop a game for teaching English language or science. The results showed that students' progress improves slightly by enhancing the traditional course structure and that real-life problem-based game projects with Scratch improve learning programming concepts.

The study (Comber et al, 2019) conducted a questionnaire on three different secondary schools' students who used Unity game development environment. Unity was used to develop games to check if it improves engagement, motivation and learning. The students were asked to develop their own 2D games over a period of 3 months in which an average of 8 lessons, 100 minutes each per school were given. The students were given a document tutorial to help them build their games. The results showed that developing games is interesting and engaging for the majority of the students.

An example of using open-source serious games is Liu (2008), where Robocode was used as a java programming assignment for level 1 students taking Java programming course. Figure 2.12 shows a screen shot of the game. The students were given three weeks to explore the source code of the game to apply and extend their knowledge in Java programming. The students were asked to make a report highlighting their findings. Also, the students were asked to develop Java classes in the context of a real-world application. 25 students responded to the distributed

survey, which asks students to express their positive and negative experiences after completing the assignment. 20 students had positive and negative experiences while 5 students had only negative experience. The result of this study stated that students enhanced their research abilities by exploring the source code and the programming section opened a path for students to promote their creativity. Moreover, the scope of exploration was big and the students had realised the value of documenting the code.

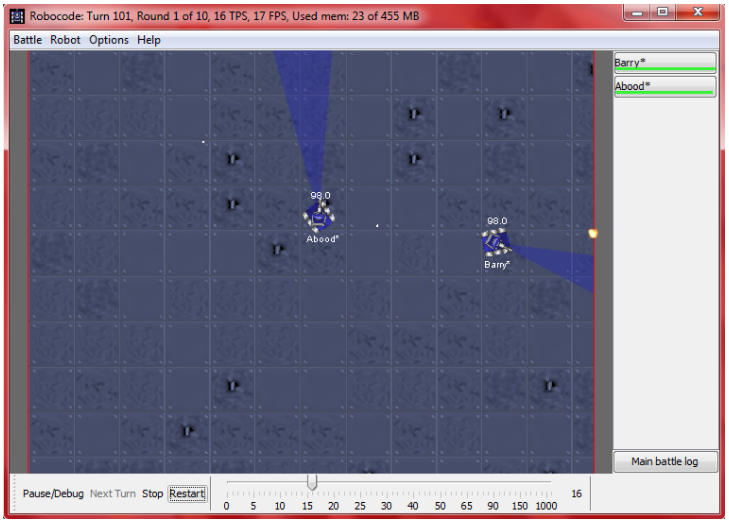


FIGURE 2.12: Robocode game screen shot.

For assessing Robocode, Hensman (2007) conducted a survey on 25 lecturers and 65 students from different universities and institutes of technology in Ireland to evaluate Robocode as a learning tool. The results of the survey showed that 83.3% of the lecturers agreed that students can use Robocode as a learning tool without any knowledge about objects. Also, 58.3% of the lecturers and 50% of the students rated the usefulness of Robocode as a teaching tool as excellent, where 33.3% of the lecturers and 42.9% of the students rated it as good. Moreover, 66.7% of the lecturers stated that they have or they would use Robocode in the classrooms, where 25% stated they would encourage students to use the game in their free time. Furthermore, the lecturers gave a rating of 100%

for two of the Robocode attributes, which are competitive aspects and social dimension. Also, they rated Robocode as a good tool for learning programming by 36.4%. 66.7% of the lecturers stated that the game suits level 1 students. The study sectioned the students into competition students for those who took part in the national competition and non-competition students who used the game to enhance their programming skills. When asked for the most enjoyable aspect of the game, 57.1% of the competition students chose the competitive nature of the game, where 71.4% of the non-competition students chose the helping to learn programming factor. Students were asked to rate the game in terms of improving programming skills, 42.9% of the competition students rated it as very good, 28.6% as good and 28.6% as it's just for fun. 12.5% of the non-competition students rated it as excellent, 37.5% as very good, 37.5% as good and 12.5% as it's just for fun. The study concluded that Robocode serves as a tool to help programming for weak programmers and a showcase for good programmers. Furthermore, the study stated that Robocode is being used as a tool to introduce objects and problem-based learning. Correspondingly, Long (2007) conducted a survey in the Robocode community forum and generated 83 responses. All of the participants were males and 80% of them informed that Robocode increased their programming skills (20% increased significantly and 60% increased somewhat). 75% of the participants' ages were between 18 and 34, 20% between 35 and 49 which means that Robocode as a game was enjoyable for older participants. 48.2% of the participants had a graduate education and 42.2% had college or undergraduate education which means that Robocode is pleasurable across different education levels. 23% of the participants were beginners, 40% intermediate, and 37% had advanced programming skills. 60% of the participants spent more than 1 hour a day playing Robocode with 16% spent more than 4 hours a day. 87%

of the participants played the game because it is fun and 54% to learn new skills. Both of the reasons are intrinsic motivators, where extrinsic motivators were chosen as follow, 32.5% to win the game, 16.3% to gain peer recognition and 11.3% to win the prize. Moreover, (to be able to solve problems on my own 88.3%) and (to be able to be creative 74.1%) were chosen by the participants as the factors that made Robocode enjoyable. Furthermore, (discovering algorithms 80%) and (designing the architecture 70.9%) were chosen by the participants as the most enjoyable activities. The study approved that Robocode could improve the user's motivation in the learning process and concluded that time spent playing the game is not subjective by the participants' age, education or skills levels.

Other innovative ways of using games in teaching computer programming was introducing chess to learning by Gusev (2018). This study introduced chess to learning computer programming for different students' levels with different programming languages such as C, C++ and Java. Open source chess engines were used by students to complete several assignments. The author stated that chess programming can be used effectively in different levels of undergraduate education if the students are sufficiently motivated. Further, Dush and Jaworski (2018) introduced the use of Arduino for students taking a fundamentals of programming course. Using such educational board can cover the use of conditional statements, loops, arrays, input and output and sorting algorithms. The authors stated that using development educational board such as Arduino will increase student' interest in programming activities.

To ensure that all previous studies on the use of serious games for teaching computer programming were covered. We ran a search string consisted of the following Boolean expression (A1 OR A2 OR A3) AND (B1 OR B2) AND (C1). Search expressions are presented in table 2.2.

The used databases included ACM Digital Library, IEEE Xplore, Scopus, SAGE Journals and Springer Journals. The search was limited to peer reviewed, full text and English papers. The search included the papers published between 01/01/2016 to 31/08/2019. The search was narrowed by the following subjects: problem solving, technology uses in education, video games, visualisation, electronic learning, computer simulation, gamification, learning, computer science education, serious games, computer programming, programming, computer games, game-based learning, educational technology, higher education, educational games, education and games.

TABLE 2.2: Search expressions

Search expressions	A	B	C
1	Serious games	Teaching	Computer programming
2	Game based learning	Learning	
3	Simulation software		

The initial result returned 681 papers. After reviewing the titles and excluding the non-relevant and the already cited papers, 26 papers were left. By reading the abstract 20 papers were excluded and 6 papers left. After reading all the papers, 2 papers were excluded. Although the already cited papers were excluded, the number of available studies is still low. The systematic review by Subhash and Cudney (2018) on gamified learning in higher education returned only 14 studies in the Computing subject area and the search covered the published studies between 2012 and 2017. The four studies are explored below.

Rozali and Zaid (2017) developed a mobile programming learning game to teach Action Script. A questionnaire was conducted with 10 post-graduate students who were enrolled in Authoring system course which

is compulsory in the Master of Educational Technology degree at Universiti Teknologi Malaysia. The questionnaire aimed to evaluate the students motivation towards learning computer programming using a mobile game. The results showed that most of the students that integrated the learning of programming in mobile games makes them motivated. Another online game was developed by Tsalikidis and Pavlidis (2016). They developed an online multiplayer game to teach programming with JavaScript called JLegends. The game was built with source code scalability so it can be expanded and edited on demand. The game has not been tested yet.

Rajeev and Sharma (2018) developed a game to teach linked lists and binary trees in the Python programming language. The results of a post survey with 57 undergraduate and graduate students showed that almost 79.63% of the students were motivated by the game. Also, 73.58% of the students found the game interesting. Galgouranas and Xinogalos (2018) developed a 2D game called jAVANT-GARDE to teach the basic concepts of computer programming in Java programming language. The game was evaluated by 42 high school students based on MEEGA+ framework to evaluate the player experience and shot-term learning. The students filled in a questionnaire after playing the game. The results showed that the students had a good experience playing the game. Positive results were recorded in terms of the game usability, challenge, satisfaction and fun. 69% of the students agreed that the game contributed in learning the basic concepts of computer programming.

Table 2.3 summarises the previously presented attempts of using serious game for teaching in terms of how effective they were, how many people benefited from them, the used programming language and the used data collection method.

TABLE 2.3: Serious games studies summary 1

Game Name/ Study	Programming language	Sample size	Data collection	Results
RoboBUG (Miljanovic and Bradbury, 2017)	C++	Case 1: 23 Case 2: 4 Case 3: 15	Pre-post-tests. Interviews.	Test scores improved after playing the game.
(Zhao et al, 2019)	C	90	Pre-post-tests. Survey	1- Statistically significant improvement after playing the game. 2- The game helped 64% of the students understand the covered topic.
Ruby Warrior (Watson and Lipford, 2019)	Ruby	185 over two semesters.	Survey (Filled in by 103)	55.34% of the students were motivated by the game to learn more topics.
ROBO3 (Agalbato and Loiacono, 2018)	Programming concepts	80	Survey	Most of the students found the game useful to improve their understanding of the covered topics.
Unity (Comber et al, 2019)	C#	NA	Questionnaire	Developing game is engaging for most of the students.
(Jordaan, 2018)	Python	8	Semi-structured interviews	1- Students enjoyed playing the game. 2- The game increases the students communication.
RunjumpCode (Hinds et al, 2017)	C#	NA	Pre-post-tests.	NA
(Erol and Kurt, 2017)	C#	52	Pre-post-tests. Questionnaire	1- Significant improvement in the test score after playing the game. 2- Motivation increase after playing the game.
Scratch (Topalli and Cagiltay, 2018)	C	48	Exam score	Students progress improves by enhancing the traditional course.
(Zhao et al, 2018)	Programming concepts	150 from three universities across Europe	Pre-post-tests. Motivation questionnaire Usability questionnaire	54% of the mature students (aged 25 and above) and 74% of the young students (aged 17-20) would like to use more technology in the classroom when learning STEM subject.
LightBot 2.0 (Mathrani et al, 2016)	Programming concepts	44	Survey	The results showed that game-based learning is a useful learning strategy before and after subject is taught
Scratch (Ouahbi et al, 2015)	Programming concepts	69	Survey	Using the game as an environment for learning programming highly motivated students
(Sprint and Cook, 2015)	C	15	Survey	All the students enjoyed the game and 83.33% preferred a gamified approach over the traditional one.
Sifteo cubes (Corral et al, 2014)	C#	30	Pre-post-tests	Overall better marks and higher interest after playing the game
Garden Gnomes From Planet (Baker et al, 2012)	Java	19	Pre-post-tests Survey	1- Statistically significant learning gains over the traditional way of teaching. 2- Students enjoyed playing the game.
(Malliarakis et al, 2014)	C	22	Questionnaire	Students have increased their understanding and knowledge levels by playing the game.
(Malliarakis et al, 2017)	C	76	Pre-post-tests. Questionnaire	1- Statistically significant improvement in the test score after playing the game. 2- Most of the students were entertained by playing the game while learning
Wus Castle (Eagle and Barnes, 2008)	Programming concepts	9	Pre-post-tests Survey	Playing the game has statistically significant learning gains over the traditional way of teaching.
Java Ninja (Zhang et al, 2013)	Java	14	Pre-post-tests	1- 21% pass rate in post test compared to the pre test. 2- Students enjoyed playing the game.
Alice (Al-Linjawi and Al-Nuaim, 2010)	Java	24	Pre-post-tests Survey	Students have increased their understanding after playing the game.
Robocode (Liu, 2008)	Java	25	Survey	Students enhanced their research abilities
(Rozali and Zaid, 2017)	Action Script	10	Questionnaire	Students' motivation increased
(Rajeev and Sharma, 2018)	Python	57	Survey	Students' motivation increased
jAVANT-GARDE (Galgouranas and Xinogalos, 2018)	Java	42	Survey	69% of the students agreed that the game contributed in learning the basic concepts of computer programming.

By observing the table, it can be seen that serious games have been used for teaching computer programming in different programming languages. The used programming language is affected by the course that the students are taking which justifies why Java, C and C# are used more. Hence, they are the first programming languages the students take in the introductory to programming courses in most universities. Further, some serious games were used to cover programming concepts without the use of a programming language to allow the students to focus on the concepts only, because the programming language syntax is considered difficult for new students (Sloan and Troy, 2008; Wilson, 2002). Figure 2.13 shows the frequency of the used programming languages.

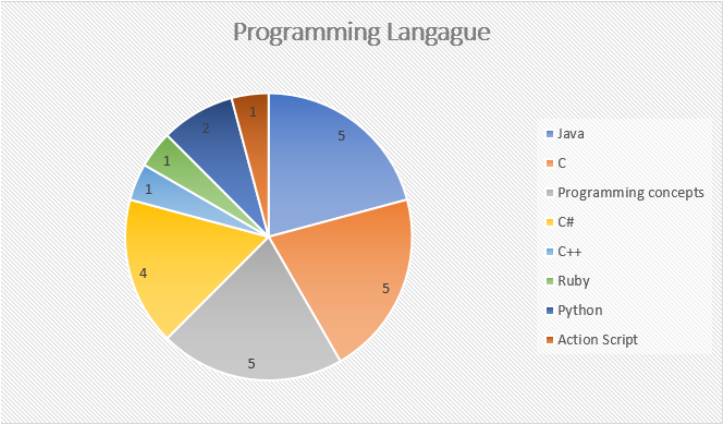


FIGURE 2.13: The used programming languages.

The used data collection methods are mainly surveys, pre-post-tests or both. The surveys are used to evaluate students' perception of the used game, the usability of the game, the students' motivation and/or students' satisfaction. Pre-post-tests are used to evaluate if there are any improvements in the students understanding of the covered topics which is represented by the exam scores. Figure 2.14 shows the different data collection methods used by the previously presented studies.

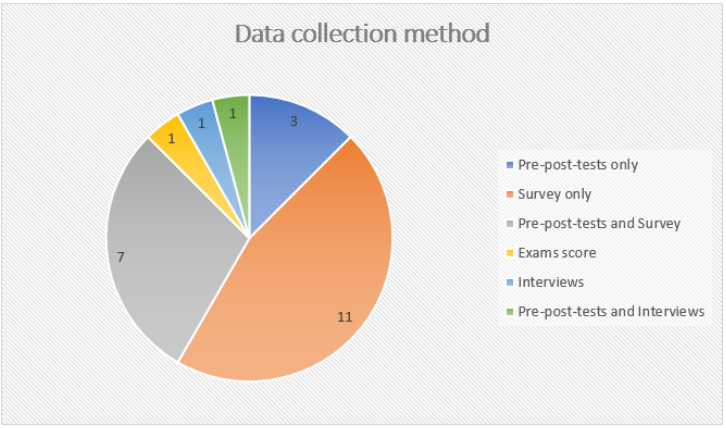


FIGURE 2.14: Data collection methods.

The following section will explore few examples of well-know computer programming serious games.

2.6 Serious games in teaching computer programming examples

Different serious games have been used for teaching computer programming, this section will highlight and describe few serious games and then will present a table that compares several aspects of the games against each other.

Alice is an innovative block-based programming environment, it's an open-source game written in Java. It is a 3D interactive environment that has visual and narrative aspects. Alice was developed by Carnegie Mellon University. Alice allows the users to drag and drop different objects like animals and buildings along with statements and expressions instead of writing the code to create their own virtual world. Moreover, Alice allows the user to create animations easily and control the animations using input methods like keyboard and mouse. Alice is designed to teach logical and computational thinking skills, fundamental principles of programming and to be the first exposure to object-oriented programming. Alice offers an instant visual output for the user without the need for compiling the project. Thus, the users can immediately see the changes they make. The game has been used in different studies such as (Aktunc, 2013; Al-Linjawi and Al-Nuaim, 2010; Cooper et al, 2000). Figure 2.15 shows a screen shot of Alice.

Scratch is a multimedia environment developed by the media lab at the Massachusetts Institute of Technology. Users can develop programs by fitting fragments of computer programs together. The game allows the users to drag and drop blocks to teach them about programming topics such as variables, conditions, loops and objects. Scratch can be used on Windows and Mac operation systems. Users can program their own games and animations and then share their projects online, where users can access each other projects and change them or build on them. Using

FIGURE 2.15: Alice screen shot.

Scratch can help the users to think creatively and work collaboratively and the game has been used in many studies such as (Bittencourt et al, 2015; Mishra et al, 2014; De Kereki, 2008). Figure 2.16 shows a screen shot of Scratch.

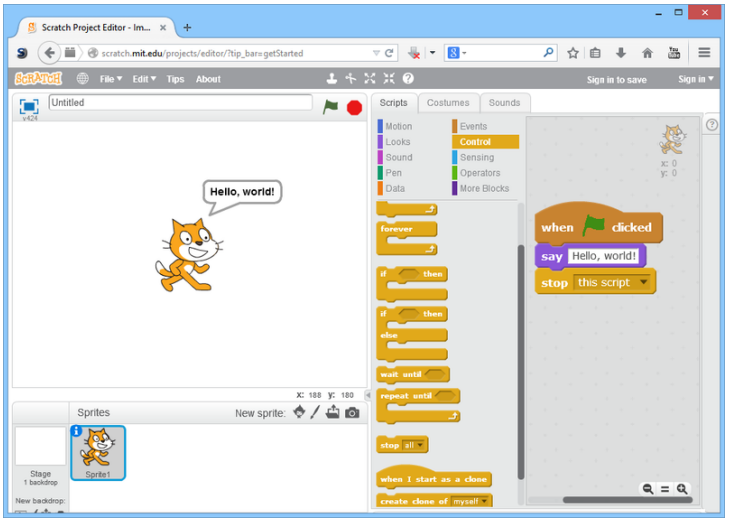


FIGURE 2.16: Scratch screen shot.

Robocode (short for Robot Code) is an open source Java-based virtual robot game created by Mathew Nelson. It is proposed to teach Java programming. The game consists of a development tool for developing the robots and a battlefield where the robots can battle. Robots can be programmed in Java and C#. The users can use the robot editor of the game to write the code or they can link the game to a separate IDE such as

Eclipse. Robocode develops students' skills for each stage of the software development process which are requirements' analysis, design, implementation and testing. The students will use all the stages in order to build the robot (O'Kelly and Gibson, 2006). Further, (Long, 2007) found that Robocode is enjoyable to play for various age groups, expertise levels and educational background. The study (Bierre et al, 2006) stated that besides Robocode being enjoyable, students produced very creative solutions. Further details about Robocode is available in section 3.4 in the Research Questions and Design chapter. Another example of a serious game for teaching Java is Jeroo. It is a Java-based serious game designed for teaching computer programming that runs on several platforms such as Windows, Mac and Linux. Jeroo has four main components which are an engaging metaphor, three programming language styles Java/C++/C#, VB.NET and Python, an integrated development environment and a runtime module. Jeroo helps in learning the fundamentals of OOP focusing on objects and methods. Sanders and Dorn, (2003) reported the use of Jeroo for teaching computer programming in different research papers.

Bomberman is a strategic maze-based computer game developed by Hudson Soft and it was published in 1983. Users of the game use C programming language to solve problems in the game. The game covers all C programming language concepts but it is mainly used for teaching novice students the simple concepts such as conditions and arrays. Bomberman has limited use reported in the literature such as the study (Chang et al, 2010). C-Sheep is another serious game for teaching computer programming in C programming language. Users control a sheep by writing code in C language. The game has a virtual environment called The Meadow, where the code can be executed and users can observe the behaviour of the cheap. C-Sheep was developed at the National Centre for Computer Animation at Bournemouth University in the UK. Limited studies have

used C-Sheep for teaching computer programming, such as (Anderson and McLoughlin, 2007; Anderson and McLoughlin, 2006). Figure 2.17 shows a screen shot of the game.

FIGURE 2.17: C-Sheep screen shot.

Prog & Play is a strategic serious game aimed for teaching computer programming. It covers various programming concepts and it has been used in studies, such as (Muratet, 2011). Robozzle and Lightbot are serious games that can be used for teaching functions/methods and recursion. The player chooses from a set of controls to move an object to cover a certain area. The player can use conditions and iterations to complete the objectives. Robozzle can be played online on http://robozzle.com/js// and Lightbot can also be played online on http://lightbot.com/flash.html. Figure 2.18 shows a screen shot of Robozzle and Lightbot.

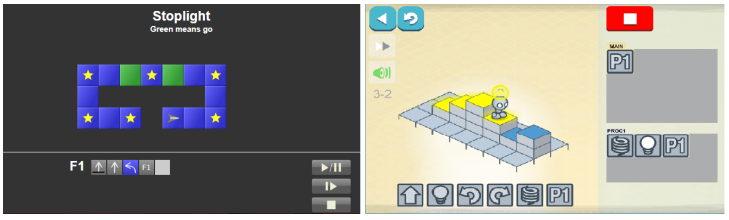


FIGURE 2.18: Robozzle and Lightbot screen shot.

Catacombs is a three-dimensional multiplayer game that aims to teach students programming concepts such as conditions and loops. The user represents a wizard that is going to save children by answering programming questions. Few studies used Catacombs for teaching computer programming, such as (Barnes et al, 2008; Barnes et al, 2007). Figure 2.19 shows a screen shot of the game.

FIGURE 2.19: Catacombs screen shot.

Turtle Logo is a game that can be considered as a mini-language which is used for teaching new students computer programming. The style of programming of Turtle is procedural and it can be used to teach conditions and loops. Users write simple lines of code and a turtle which act as a cursor will draw graphical lines on the screen. There are different versions of the game, some use triangle instead of the turtle to represent the cursor, but they all share the same commands and purpose.

From the presented examples of serious games for teaching computer programming. The differences in the game design can be observed from the provided screen shots, in which most of the games are for a single player while some of the games support multiplayer option. Further, 2D and 3D interfaces were used for the serious games and different platforms

were supported by the different games. Moreover, different genres were used in designing serious games. For example, Alice, Scratch are 3D games where Robocode and Bomberman are 2D games. Catacombs and Robocode are multiplayer games where C-Sheep and Lightbot are single player games.

Table 2.4 presents a comparison of different serious games for teaching computer programming in terms of the style of programming, the covered concepts, code representation and program construction.

Comparisons criteria		Robocode	Alice	Scratch	Jeroo	C-Sheep	Bomberman	Turtle Logo	Catacombs	Wu's Castle	Prog & Play	Robozzle	Lightbot	PlayLogo 3D
Style of programming	Procedural					X	X	X			Multiple	X	X	X
	Object-based		X	X										
	Object-oriented	X			X				X	X				
Concepts covered	Variables	Java language	X	X	X	X	C language	X		X	X			X
	Conditions		X	X	X	X		X	X	X	X			
	Loops		X	X	X	X		X	X	X	X			
	Functions/ Methods		X	X	X	X		X			X	X	X	X
	User-defined data type													
	Recursion					X						X	X	
	Collections/ array		X		X	X				X				
Code representations	Text	X	X	X	X	X	X	X	X	X	X			X
	Pictures											X	X	
Program construction	Typing code	X			X	X	X	X	X	X	X			X
	Assembling graphical objects		X	X								X	X	

TABLE 2.4: Different serious games comparison

There are serious games for the two styles of programming, which are procedural and object-oriented. Figure 2.13 in the previous section showed that the most covered programming languages by the serious games are C, C# and Java. Moreover, some games covered a whole programming language while others were limited to some selected programming concepts. Further, most of the serious games represented the code by text and the user was asked to type code to solve a specific problem. However, some games used pictures to represent the code. As mentioned in the previous section, the studies (Sloan and Troy, 2008; Wilson, 2002) stated that new students consider the programming language syntax is difficult to learn. Some serious games are using assembling graphical objects instead of typing code to solve problems. To conclude, all of the presented serious games were used to teach computer programming. Despite the various differences in the games design, they all achieved positive results in terms of enhancing students' understanding and /or increasing the motivation of the students. The selection for one of the serious games to be used will be based on the characteristics of the serious games and based on the teachers' requirements, such as the programming language or the scope of the game.

The following section will explore and describe the quality characteristics of serious games.

2.7 Serious games quality characteristics

The literature in evaluating serious games is diverse and various quality characteristics can be used to evaluate several aspects of serious games. A noticeable effort has been made by Calderon and Ruiz (2015). They investigated the literature and summarised the quality characteristics that have been used to evaluate serious games into 18 characteristics, which

are game design, user's satisfaction, usability, usefulness, understandability, motivation, performance, playability, pedagogical aspects, learning outcomes, engagement, users experience, efficacy, social impact, cognitive behaviour, enjoyment, acceptance and user interface. These quality characteristics will be explored describing what are they for and what do they measure. Also, highlighting their occurrences and usage in evaluating serious games to assess the relevance of each quality characteristic to this research.

1. Game design

   This refers to how the serious game is designed and the appealing and artistic visual design. The study (Deterding et al, 2011) stated that there are 5 levels of game design elements which are game interface design patterns, game design patterns and mechanics, game design principles and heuristics, game models and game design methods. Muratet et al (2011) considered game design and its appealing look as part of user's satisfaction. Game design has been measured and used in different studies, such as (Cederholm et al, 2011; Muratet et al, 2011).

2. User's satisfaction

   User's satisfaction does not have a unified definition, but rather depending on context and population subgroups (Doll et al, 2004). Several factors affect user satisfaction like perceived usefulness, perceived ease of use and perceived flexibility (Arbaugh, 2002; Arbaugh and Duray, 2002; Arbaugh, 2000). The studies (Hong, 2002; Thurmond et al, 2002) highlighted other factors like age, gender, users' initial computer skills and users' initial knowledge about e-learning technology. Moreover, Kanuka and Nocente (2003) and Piccoli et al (2001) underlined further factors like motivation and

attitude towards technology. The study (Sun et al, 2008) developed a six dimensions framework to assess e-learning satisfaction which are learner dimension, instructor dimension, course dimension, technology dimension, design dimension and environmental dimension. User's satisfaction has been measured and used in several studies, such as (Mortara et al, 2014; Torrente et al, 2009).

3. Usability

   Usability has been defined by the international standard ISO 9241-11 as the "extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use" (ISO/IEC 9126:1998). According to the definition, usability is a construct consisting of three dimensions: effectiveness, efficiency and satisfaction. Dumas and Redish (1999) defined the usability as "the people who use the product can do so quickly and easily to accomplish their own tasks"; further, the book identified four points that form the definition which are focusing on users, people use products to be productive, users are busy trying to accomplish tasks and users decide when a product is easy to use. Similarly, Nielsen (1993) stated that the usability is associated with five attributes which are learnability, efficiency, memorability, errors, and satisfaction. Many studies measured the usability characteristic in serious games, such as (Rodrguez-Cerezo et al, 2014; Couceiro et al, 2013; Zhang et al, 2013; Muratet et al, 2011; Olsen et al, 2011).

4. Usefulness

   This was defined as "The degree to which a person believes that using a particular system would enhance his or her job performance"

(Davis, 1989). The study (Alsabawy et al, 2016) stated that perceived usefulness is the main measure to assess the acceptance and success of e-learning systems. Moreover, Sela and Sivan (2009) listed usefulness as the first "Must-Have" success factor in e-learning. In the context of serious games, usefulness refers to the efficiency and suitability of the serious games for the designated audience, it has been measured as a quality characteristic in various studies, such as (Costantino et al, 2012; Backlund et al, 2011).

5. Understandability

   Understandability was defined in the ISO 9126 document as "the capability of the component to enable the user to understand whether it suitable, and how it can be used for particular tasks and conditions of use" (ISO/IEC 9126:2001). It's considered part of the usability characteristic along with learnability, operability, attractiveness and usability compliance. Another definition for understandability was listed by the study (Boehm et al, 1978), where software understandability was defined as a characteristic of software quality, which means ease of understanding software systems. It is considered the second level of Bloom's taxonomy as shown in figure 2.2. Further, the results of a questionnaire with 370 teachers showed that all of the teachers think that understandability is very important or at least important (Mazgon and Stefanc, 2012). Understandability is a crucial characteristic where if the serious game or the software is not understandable it will not achieve its educational goals. This characteristic has been explored and measured in studies, such as (Buttussi and Chittaro, 2010; Jung et al, 2004; Uchida and Shima, 2004).

6. Motivation

This refers to the ability of the used serious game to impact and affect the user's motivation where it encourages the user to use the serious game. According to Wrzesien and Alcaniz (2010), serious games boost the user's intrinsic motivation where it leads the users to curiosity and desire for challenge. Motivation is considered an important characteristic, in which several theories, such as the study (Garris et al, 2002) highlighted the potential of serious games to positively impact intrinsic motivation and furthermore the study stated that probably the primary purpose for using serious games is their assumed motivational appeal. Moreover, the study (Girard et al, 2013) analysis of previous studies on the effectiveness of serious games explored the motivational factor as a primary key for an effective use of a serious game. The motivation characteristic has been used and measured by numerous studies, such as (Wangenheim et al, 2012; Miller et al, 2011; Kebritchi et al, 2010; Papastergiou, 2009).

7. Performance

   Performance can be described as the functioning and the effectiveness of the serious game. The performance characteristic has been measured in studies by rating the functionalities of the serious game (Zhang et al, 2013); also, it has been used to evaluate a serious game in the development process to test the components of the game (Manero et al, 2013).

8. Playability

   This means is the serious game playable. It is a crucial characteristic, as if the serious game is not playable then it will fail to accomplish its designated purpose. The study (Olsen et al, 2011) highlighted the importance of playability characteristic in serious games. Playability focuses on the complete functionality linked with the integration

of numerous usable tools tolerating for effective and pleasant interaction with the game. Heuristic evaluation has been developed to evaluate the playability of games (Desurvire et al, 2004) and it has been used to evaluate serious games (Petrasova et al, 2010). Also, the study (Manero et al, 2013) evaluated playability as a main quality characteristic to evaluate the effectiveness of the used serious game.

9. Pedagogical aspects

   This refers to the educational content provided by the serious game. It is vital that the educational content of the serious game is well adjusted and calibrated. As Quinn (2005) argued that serious games must be designed appropriately to create a harmony between the game-play and the learning objectives. (Manero et al, 2013; Hauge and Riedel, 2012) evaluated the pedagogical aspects in their evaluation of the serious game.

10. Learning outcomes

    Learning outcomes is considered the most important characteristic since its presence convert the game into a serious and educational game. Learning outcomes refers to the knowledge gained by the users after interacting with the serious game. Most of the studies evaluate the learning outcomes of the serious game first and then consider other dimensions or characteristics. In computer programming serious games, many studies evaluated the learning outcomes and used qualitative and/or quantitative methods and data in their analysis (Corral et al, 2014; Malliarakis et al, 2014; Baker et al, 2012; Wangenheim et al, 2012; Hillyard et al, 2010; Galves et al, 2009; Eagle and Barnes, 2008; Liu, 2008). Some studies divided learning outcomes into categories, such as the study (Wouters et al,

2009), where they divided learning outcomes into four categories cognitive, motor skills, affective and communicative.

11. Engagement

    This characteristic checks if the serious game engages users in using and playing. Engagement is complex as stated by the study (Dele-Ajayi et al, 2016), where they considered motivation as the root of engagement. The study (Pourabdollahiana et al, 2012) used a framework for evaluating the engagement characteristic based on two studies (Csikszentmihalyi, 1992; Malone and Lepper, 1987). The framework consisted of five factors that affect engagement, which are immersion, control, challenge, purpose and interest. Several studies evaluated the engagement characteristic in serious games, such as (Adamo-Villani et al, 2013; Dunwell et al, 2013).

12. User's experience

    User's experience was defined in the ISO 9241 document as "A person's perceptions and responses resulting from the use and/or anticipated use of a product, system or service" (ISO 9241-210:2010). In Wangenheim et al (2012) model to evaluate a serious game, the study recommended three factors to assess the used serious game which are motivation, learning and user experience, under user experience factor the study listed five aspects which are immersion, challenge, competence, fun and social interaction. A considerable number of studies evaluated user's experience characteristic, such as (Hou and Li, 2014; Law and Sun, 2012; Pavlas et al, 2012; Wangenheim et al, 2012).

13. Efficacy

Efficacy was defined as "the power or ability of the game to improve participants' knowledge" (Ingadottir et al, 2017). Evaluating the efficacy of the serious games involves evaluating the engagement, attention, involvement, enjoyment, difficulties, and time to complete the game (Harteveld et al, 2010; Thompson et al, 2010; Faria et al, 2009). Several studies evaluated the effectiveness of serious games, such as (Ingadottir et al, 2017; Malliarakis et al, 2014; Girard et al, 2013; Guillen-Nieto and Aleson-Carbonell, 2012; Mayo, 2007).

14. Social impact

    This can be defined as the impact or the influence that serious games create in the users or contribute to it. (Swain, 2007) listed 8 practices for designing and developing serious games to affect social change which are define intended outcomes, integrate subject matter experts, partner with like-minded organisations, build sustainable community, embrace "wicked problems", maintain journalistic integrity, measure transference of knowledge and make it fun. Hensman (2007) stated an advantage of the used serious game "as a platform for students to showcase their work in a socially interactive environment". Measuring social impact characteristic of serious games is limited in the literature and one example of measuring this characteristic is (Rebolledo-Mendez et al, 2009).

15. Cognitive behaviour

    Cognitive behaviour can be defined as the ability of serious games to create effects and changes in user's cognitive behaviour. Hauge and Riedel (2012) stated that cognitive analysis is a beneficial tool that allows game developers to well tune the serious games. Cognitive behaviour analysis is limited in the literature, few studies, such as

(Manera et al, 2015; Quick et al, 2013) measured this quality characteristic.

16. Enjoyment

    This can be defined as the joy and fun that the serious game produces to the users. Ricci (1994) stated that serious games increase the user interest because users enjoy this approach to learn. In Hensman (2007) study about a serious game called Robocode, the study questioned the enjoyable aspects of the game and the answers were its competitive nature and helping to learn computer programming. Enjoyment is considered as a key factor in games in general, where previous studies showed that enjoyment of playing is one of the main motivators for playing computer games (Sweetser and Wyeth, 2005; Hsu and Lu, 2004). Assorted studies explored the enjoyment characteristics, such as (Malliarakis et al, 2014; Zhang et al, 2013; Wrzesien and Alcaniz, 2010).

17. Acceptance

    Acceptance refers to the serious game being accepted and received by the user. According to Alsabawy et al (2016), the acceptance characteristic can be assessed by measuring the usefulness of the system, where acceptance is linked to perceived usefulness and perceived ease of use (Davis, 1989). Acceptance is considered as an important characteristic since if the user didn't accept the game and used it, he/she will not engage effectively and, thus, will not receive the specified knowledge. The study (Brom et al, 2010) identified five key factors that led to the positive acceptance of the used game, which are intelligibility, social role-playing, grounding in real data, story-telling and support for teachers. Technology Acceptance Model (TAM) has been used to predict the user acceptance of new

technology (Davis, 1989). Yusoff et al (2010) proposed a model for designing serious games and using TAM for validating the acceptance of the game. Few studies considered measuring the acceptance characteristic of serious games, such as (Eichenberg et al, 2016; Enah et al, 2014; Brom et al, 2010).

18. User interface

    This refers to the interaction that occurs between the user and the serious game. The study (Lanyi et al, 2012) stated that it is essential to design the user interfaces for maximum accessibility and usability. The study (Mikovec et al, 2009) investigated the designing of the user interface for serious games and the need for developing a user interface that is usable and understandable; furthermore, the study stated that serious games are barely successful without an appropriate and well-designed user interface. The study (Deterding et al, 2011) stated that game interface is one level of the elements of game design. Few studies measured user interface characteristic, such as (Lanyi et al, 2012).

After describing each of the different quality characteristics of serious games and highlighting their use in the literature. The following section will cover several attempts for evaluating serious games.

2.8 Serious games evaluation

This section will explore several evaluation frameworks that have been used in evaluating serious games. Moreover, it will highlight the advantages and disadvantages of the used frameworks.

De Freitas and Oliver (2006) developed a four-dimensional framework for the purpose of helping tutors to evaluate the potential of serious games.

Also, helping them in selecting a serious game that matches their needs. Further, the authors proposed that it can be used by game developers to design serious games that match the needs of the educational program. The four dimensions are context, learner specification, mode of representation and pedagogic considerations as shown in figure 2.20. The study emphasised the need to use the four dimensions together and not separately to illustrate the significance of the dimensions and how they relate to each other to support the learner experience. The context dimension focuses on where is the learning taking place in school, university, home or a combination of some. Also, it focuses on the need for specific resources and technical support and if it interferes with the context for learning. The learner specification dimension focuses on the learner attributes such as age, background and learning styles. The mode of representation dimension focuses on what does the game represent, the realism of the game, the interactivity and the level of fidelity. The fourth dimension focuses on the process of learning in terms of the used pedagogic models, the curricular objectives, the learning outcomes and activities. However, Robertson and Howells (2008) stated that the four-dimensional framework requires the tutors to have a good background in computer games. Furthermore, it doesn't sufficiently assist tutors in classifying the serious games that match their learning needs.

Ssemugabi and de Villiers (2007) developed a framework for evaluating web-based learning applications. The main category in the framework was developed based on Nielsen and Mack (1994) heuristics. The other two categories covered the website-specific criteria for educational websites and learner-centred instructional design, grounded in learning theory, aiming for effective learning. Heuristic Evaluation (HE) is an inspection technique where a set of experts evaluate whether a user interface conforms to defined usability principles, called heuristics (Dix et

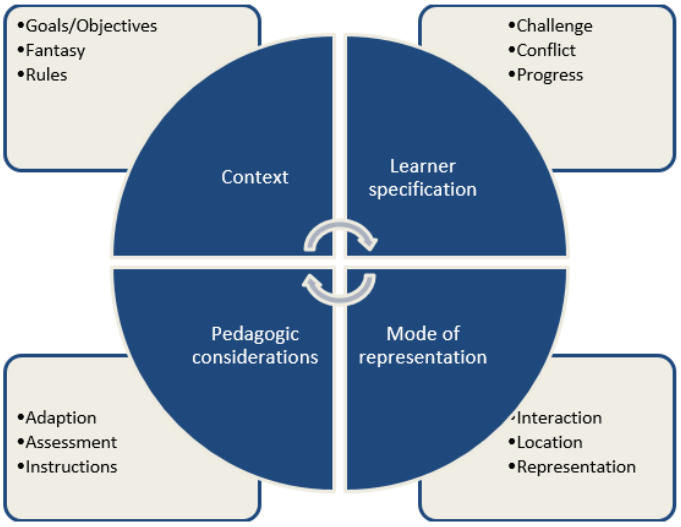


FIGURE 2.20: (De Freitas and Oliver, 2006) Four-dimensional framework.

al, 2004; Preece et al, 2002; Nielsen, 1992). Also, HE is used to identifying specific usability problems in the system. HE can be applied to different quality characteristics other than usability like an evaluation of the games playability (Desurvire et al, 2004) and game design (Song and Lee, 2007). HE is useful, however, it can be difficult, time-consuming and expensive (Kjeldskov, 2004). Moreover, it can lead to ignore any usability problems that are not covered by the listed heuristics. While this study focused on the usability factor, other studies focused on different factors. For example, the study (Fu et al, 2009) developed a scale to assess user enjoyment of e-learning games. The scale consists of eight dimensions, which are immersion, social interaction, challenge, goal clarity, feedback, concentration, control, and knowledge improvement. A questionnaire was conducted to validate the scale in which 166 valid responses were collected from students who took an online learning course called Introduction to Software Application.

The study (Xu et al, 2013) developed a serious game framework assessment named Serious Game Stakeholder Experience Assessment Method

(SGSEAM). It assesses serious games from the major stakeholders' experience perspective. The goal of this framework is to identify the strengths and shortcomings of a serious game framework for various stakeholders that are involved in the serious game life-cycle including players, system admins, game designers, game managers, community partners and funding organisations. There are a set of questions and other assessment approaches proposed for each stakeholder, which include the collection of quantitative and qualitative data. Using this framework can be helpful for identifying the usefulness of the serious game and its suitability to be used for a party and highlighting the potential improvements that can be applied for each stakeholder. However, applying this framework on a big serious game that involves multiple stakeholders can be difficult, expensive and time-consuming, because for each stakeholder, data should be collected and analysed.

The study (Schumann et al, 2001) construed and illustrated the four levels that formed Kirkpatrick's (1998) framework for evaluating and analysing the results of training and educational programs. The four levels are reaction, learning, behaviour and results. The framework was developed to explore the effectiveness of the simulation software from four diverse perspectives. Farjad (2012) used Kirkpatrick model to evaluate the effectiveness of a training course at a university level. Betas (2004) referred the popularity of using the Kirkpatrick model to provide a clear system or language and information needed to assess training programs and simplifying the complex evaluation process. On the other hand, Betas (2004) highlighted some limitations to this model like incompleteness to the model due to not considering individual or contextual influences and the assumption of causal linkages. For example, the model assumes that positive reactions lead to greater learning that delivers greater transfer and thus positive results. Another limitation is that the model assumes

that each level provides more informative data than the last level, which produced a perception for evaluators that collecting level four data, which is results will offer the most useful information as shown in figure 2.21. Furthermore, this model was designed for evaluating training programs and implementing it for evaluating serious games in teaching will ignore multiple quality characteristics.

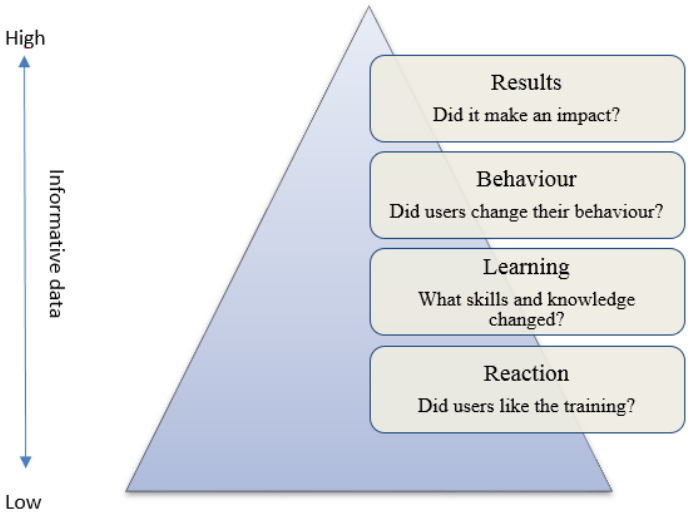


FIGURE 2.21: Kirkpatrick model.

The study (Connolly et al, 2009) developed a framework to evaluate the effectiveness of game-based learning. The framework consisted of seven major components, which are learner performance, motivation, perception, attitudes, preferences, collaboration and game-based learning environment. The study also specified that a game-based learning environment is divided into five categories, which are environment, scaffolding, usability, level of social presence and deployment. The study stated that this framework can be used at the development phase of the game or at a final product. The framework takes the pedagogical aspects into consideration and it provides an explanation and details about the chosen components and categories. However, it doesn't specify why they were chosen and why other characteristics have been withdrawn. Moreover, the study doesn't specify how the framework can be applied and it's yet

to be validated. Another framework was developed by the study (Mayer et al, 2014) for evaluating serious games. The framework is formed out of eight steps, which are framing, foundations and requirements, conceptual framework, quasi-experimental research design, contextualisation, research questions and hypothesis, operationalisation and data reduction and analysis. The framework can be identified as comprehensive in measuring learning outcomes. However, the framework is limited in the sense that the prime focus is on the learning outcomes only and it eliminates other vital characteristics. Similarly, a framework was proposed to evaluate the effectiveness of game-based learning by integrating the users' background parameters by the study (Ariffin et al, 2014). The framework consists of three background parameters, which are culture, ethnicity and native language. The framework aims to investigate if the motivation of the users will increase when different background parameters that suits the users are in the game. The framework was not tested on an actual game. It was tested as a conceptual framework by distributing a questionnaire on undergraduate students asking about the three different background parameters. The results showed that matching the game with the users' background will increase the motivation to play the game. The study and the framework are not comprehensive in evaluating serious games. Yet, it can be used as part of bigger evaluation framework. A big limitation of this study is that it was not tested on an actual game, but only by a general questionnaire in which the results obtained may not reflect an accurate response and reaction compared to users after playing the game.

The study (Wilson et al, 2016) designed an evaluation framework for large-scale and significant serious games to be applied and used in developing serious games. The framework highlights the evaluation during the development in four areas which are theoretical, technical, empirical and

external. The presented framework forms an excellent evaluation process to be followed during the development of big projects and serious games. However, the framework can't be applied to evaluate a current serious game for the aim of improving its results or to highlight its strengths and weaknesses. Another framework was proposed by Chew (2017) to be used when developing serious games. The framework measures the engagement characteristic with six factors, which are behaviour, cognitive, emotion, agentic, challenge and immersion. Design thinking process was jointly used in the framework since it enables the game to be designed form a users point of view. The framework was tested in an experiment through a questionnaire conducted with students in three simulated game scenarios. The results showed that the gaming activity designed using the framework is efficient in engaging the students to learn.

Further, Emmerich and Bockholt (2016) developed an evaluation framework for serious games to be used during the design and development process of serious games. It is named as the framework of evaluation-driven design. The framework consists of three phases, which are the preparation phase, the design phase and the evaluation phase, with the last two phases interlinked and used iteratively. The result of applying this evaluation framework will act as an input for the design process of the game. The authors stated that the framework doesn't consider important aspects of serious games such as engagement and fun. This is a major drawback of the framework, because engagement and fun are essential factors for the success of serious games. The study (Bellotti et al, 2013) stated that that assessment of a serious game must consider the aspect of fun/enjoyment along with the aspects of educational impact. Some studies, such as (Nacke et al, 2010) aimed at evaluating the game play of the serious games only, which means focusing on the enjoyment of the game. Similarly, the study (Yusoff et al, 2010) proposed

a model for designing serious games by using Technology Acceptance Model (TAM) for validating the acceptance of serious games. The aim of the model is to be used in the phase of developing and designing serious games to measure the learners' acceptance and willingness to use the serious game. Using this approach can be applied mainly in the development phase and it only evaluates one quality characteristic of serious games which is acceptance.

El Borji and Khaldi (2014) developed a tool aimed for assessing the quality of serious games in teaching. The tool consisted of four sections, which are:

1. Serious game identification that describes and identifies the serious game.
2. Pedagogical specification that consists of three elements, which are content that focuses on the signification and tenor of the game. Strategies that focus on the objectives, approaches and adaptability of the game. Assessment method, which focuses on the assessment procedure of the game.
3. Playful specifications, which aim to check if the game is exciting and attractive. It consists of two sections, which are attractiveness that meant to test the immersion and the motivation of the serious game. Playability aspects that focus on how the game can be played.
4. Technical Specifications and it consists of two elements, which are the technical efficiency that focuses on the graphics and the sounds of the game. The requirements that check the installing and system requirements for the game and any guides and instructions for installing the game.

This tool could be helpful in assessing serious games. However, the study doesn't specify how the tool was designed and what was the criteria based on. Further, applying the tool was conducted by the authors, in which they are not the actual users of the serious game, which mainly are students. Thus, the results could include bias or the actual users of the game view the serious game from a different point of view and different perspective. Thus, the results obtained from running the assessment tool could be not accurate and, perhaps, not relevant depending on the type of the serious game and its users.

Savi et al (2011) proposed a Model for the Evaluation of Educational Games (MEEGA), which is a model specifically developed for the evaluation of educational games. It focuses on the reaction of students after they have played an educational game, which is level one of Kirkpatrick model (Kirkpatrick, 1998). It evaluates three quality characteristics which are motivation, user experience and learning. The model uses questionnaire to collect data from learners after they play the game. The model was updated further by Petri (2018) and named MEEGA+, in which it evaluates motivation, user experience, usability, engagement, enjoyment and learning. MEEGA+ merged all the characteristics under two topics, which are usability and player experience, and it can be used by game designers, instructors and researchers to evaluate the quality of games for improvement.

Some of the evaluation frameworks are not completed or tested, such as the work by Carvalho (2012), in which an evaluation framework was developed to assess the efficiency of serious games in engineering education. The framework consists of three stages. Alpha testing to be examined by developers using tests and it examines game features during the development phase. Beta testing to be examined by a sample of end-users using questionnaires and interviews and it evaluates game play, game

story and mechanics. Gamma testing to be examined by end-users using tests and questionnaires and it evaluates knowledge, satisfaction and usability. The proposed framework has not been tested yet as reported by the study. Moreover, the study (Garcia-Mundo et al, 2015) proposed a quality model to allow designers and developers of serious games to evaluate and improve the quality of any serious game. The model consists of the eight factors, which are functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability and portability. The study reported that the validation of the proposed quality model is part of the future work of this research. Further, Ak (2012) developed a scale to evaluate the quality of any serious game in terms of three main categories, which are enjoyment, learning and usability. The scale can be used by teachers to identify the quality of the serious games. The study reports that the usability is not included in the scale yet. Moreover, the study doesn't present the scale and it is considered under development.

Table 2.5 summarise the different evaluation studies based on their approach, the data collection method and the evaluated factors.

Be observing the evaluation studies, it can be seen that various approaches have been used, such as assessment grids, scales, models and frameworks. Moreover, most of the studies have used questionnaires as a data collection method for evaluating serious games for different purposes.

The presented evaluation studies are diverse and they serve different purposes. For example, the studies (Chew, 2017; Emmerich and Bockholt, 2016; Wilson et al, 2016; Yusoff et al, 2010) focussed on the evaluation process of serious games during the development phase. Moreover, the studies (Garcia-Mundo et al., 2015; Carvalho, 2012; Ak, 2012; Connolly et al, 2009) presented evaluation studies that were either not completed or not validated. Further, some research focussed only on certain attributes

TABLE 2.5: Evaluation studies summary

Study	Approach	Data collection method	Evaluated factors
(Chew, 2017)	Framework	Questionnaire	Behavior, Cognitive, Emotional, Agentic, Challenge and Immersion
(Ak, 2012)	Scale	Questionnaire	Enjoyment Learning (Usability not completed)
(Carvalho, 2012)	Framework	Questionnaire Interviews Tests	Bets testing (Game play, Game story and Mechanics) Gamma testing (Knowledge, Satisfaction and Usability)
(Garcia-Mundo et al., 2015)	Model	NA	Functional suitability, Performance, Efficiency, Compatibility, Usability, Reliability, Security, Maintainability and Portability
(Fu et al., 2009)	Scale	Questionnaire	Immersion, Social interaction, Challenge, Goal clarity, Feedback, Concentration, Control, and Knowledge improvement
MEEGA (Savi, et al., 2011)	Model	Questionnaire	Motivation, User experience and Learning
MEEGA+ (Petri, 2018)	Model	Questionnaire	Motivation, User experience, Usability, Engagement, Enjoyment and Learning
Four Dimensional Framework (Freitas and Oliver, 2006)	Framework	NA	Pedagogic considerations, Learner specification, Context and Mode of representation
(Connolly et al, 2009)	Framework	NA	Learner performance, Motivation, Perception, Attitudes, Preferences, Collaboration and Game-based learning environment
(Ariffin et al, 2014)	Framework	Questionnaire	Learner background (culture, ethnicity and native language), Motivation and Learners' performance
(Ssemugabi and de Villiers, 2007)	Framework	Heuristic Evaluation	Interface usability
SGSEAM (Xu et al, 2013)	Framework	Experimental study Interviews System logs Engagement metrics	Sustainability, System admin assessment, Game designer experience, Game manager experience and Developer experience
(Mitgutsch and Alvarado, 2012)	Framework	Questionnaire	Content & Information, Game mechanics, Fiction & Narrative, Aesthetics & Graphics and Framing
(Schumann et al, 2001)	Framework	NA	Reaction, Learning, Behaviour and Results
(Yusoff et al, 2010)	Model	Questionnaire	Learnt skills, Learner control, Situated learning, Reward, Usefulness, Ease of use and Behavioural intention.
(El Borji and Khaldi, 2014)	Assessment grid	NA	Content, Strategies, Assessment method, Attractiveness, Playability aspects, Technical efficiency and Requirements.

or characteristics of serious games, such as enjoyment (Fu et al., 2009), usability (Ssemugabi and de Villiers, 2007), learning outcomes (Mayer et al, 2014), user background (Ariffin et al, 2014), users' reaction (Savi et al., 2011) and the serious game stakeholders (Xu et al, 2013).

The presented work by (El Borji and Khaldi, 2014; Freitas and Oliver, 2006) gave an example of evaluating serious games to investigate the games' suitability before using it with students. Their work covered multiple characteristics such as the playability of the game and the learning outcomes. Moreover, Mitgutsch and Alvarado (2012), presented a framework to evaluate serious games based on their purpose. The study stated

that the game design must reflects the game purpose in six components of the serious game, which are content & information, game mechanics, fiction & narrative, aesthetics & graphics and framing. This framework can be also used to evaluate the suitability of the game or can be used in the design and development process of the serious game.

From all the presented studies, MEEGA+ by Petri (2018) emerges on top as an evaluation study for the quality of serious games for the purpose of highlighting the games' strengths and weaknesses. However, the study omits one of the primary quality characteristics, which is understandability. Moreover, by analysing the MEEGA+ presented model, a problem of redundancy appeared. For example, when evaluating the ease of use of the game and that the game is easy to be played, the study asked the following redundant questions.

1. *Learning to play this game was easy for me.*
2. *I think that most people would learn to play this game very quickly.*
3. *I think that the game is easy to play.*
4. *When I first looked at the game, I had the impression that it would be easy for me.*

Having such redundancy in the model will lead to unbalanced evaluation. Further, to evaluate the learning outcomes of a serious game, an experiment with control and experimental groups should take place. However, MEEGA+ evaluates learning in the serious game using a questionnaire, where the users are asked questions such as *The game contributed to my learning in this course*.

MEEGA+ comes very close to achieve its purpose. Yet, there are some flaws and shortages in this model. Thus, there is a need for an evaluation framework to assess the quality of serious games for the purpose

of highlighting the games' strengths and weaknesses in order to improve them.

2.9 Summary

Serious games have been used for teaching and training purposes in various areas. Although there are many definitions for serious games, but they all agree on the importance of the entertaining factor for the success of a serious game in delivering its content to the designated audience. The pedagogy of serious game for teaching computer programming is missing in the literature. The design of serious games for most of the studies was based on well-established learning theories and the focus was on measuring different quality characteristics of the games, such as motivation and engagement.

The use of serious games for teaching computer programming has been limited to prove the effectiveness of using this approach over the traditional way of teaching and different research designs have been used. Moreover, most of the studies were conducted in developed countries and a limited number of studies were carried out in developing countries. To the best of our knowledge no previous research study has investigated the best approach for using serious games in teaching computer programming and conducted a study to compare implementing the use of serious games between developed and developing countries.

Different serious games for teaching computer programming were used. Each game has its own specifications and characteristics and they all achieved positive results when tested. There were also several attempts for evaluating serious games. Limited studies conducted evaluations on

serious games for the aim of highlighting the games strengths and weaknesses and there are observable flaws in the presented studies. Therefore, there is a need for a study to evaluate and assess serious games for the purpose of highlighting the games strengths and weaknesses in order to improve them.

Chapter 3

Research Questions and Design

The literature review covered the definition, pedagogy and usage of serious games. Then, it presented and compared the previous studies on the use of serious games in teaching computer programming. Also, the literature review chapter highlighted and compared previous evaluation studies for serious games. This chapter analyses further the obtained results to form the research questions that this study aims to answer. Firstly, the chapter highlights the need to collect and analyse data with regards to students attrition in computing schools and failure rates in computer programming courses. Then, the chapter highlights the need for collecting data about the possible usage of serious games in teaching computer programming and investigating the difficult concepts of computer programming. After that, the chapter emphasises the importance of finding the best approach for using serious games in teaching computer programming while investigating the feasibility and possibility of using serious games for teaching in developing countries. This chapter underlines the serious game that will be used in this research and then highlights the quality characteristics of serious games and the need to develop a framework to evaluate serious games for the purpose of improving them.

3.1 Attrition and failure rates

The introduction chapter presented several previous studies which highlighted the problems of high attrition rate for computing students, high failure rate in computer programming module and the declining number of students' enrolment. All of the studies were conducted in developed countries and yet, there are demands to prove these problems. For example, the studies (Watson and Li, 2014; Bennedsen and Caspersen, 2007) investigated the failure rate in the first programming module and yet, they asked for more results to be obtained to prove this problem. Further, Guzdial (2014) addressed a problem in the process of collecting data with regards to failure and retention rates. He stated that no one will publish a paper saying "Hey, we've had lousy retention rates for ten years running!". Thus, there is a need to contribute in collecting data about the attrition rate, failure rate and enrolments in computing majors in both developed and developing countries. The research questions that collecting and analysing the data will answer are:

1. What are the percentages of attrition in computing majors in developed and developing case study countries from the year 2010 to 2015?

2. What are the percentages of failure in the first programming module in developed and developing case study countries from the year 2010 to 2015?

3. Is there a decline in the total number of students or the new enrolled students in computing schools in developed and developing case study countries from the year 2010 to 2015?

The data collection period is set to start on 2010 and not earlier due to the limitation of the documented stats based on the initial communication with the universities in the developing case study country. The period will be extended if applicable when data are being collected subject to data availability and the time of data collection.

3.2 Computer programming teaching difficulties and serious games

The difficulties in teaching computer programming and the use of serious games in teaching computer programming have been addressed in the Introduction and Literature Review chapters. Surveys and interviews were used to highlight the difficulties in teaching computer programming.

Allsop and Jessel (2015) conducted a survey and semi-structured interviews to review the teachers' perception on the use of game-based learning in primary schools. The study was conducted in England and Italy. The majority of the 89 survey respondents reported that they would use digital games for teaching in the future. Developing problem solving skills and games motivational power were the two reasons by the teachers on why using games for educational purposes. When the teachers were asked about the barriers of using educational games for teaching. The responses varied but the most common answers were due the access to equipment in the classroom, lack of schools' ICT capability, relevance to the curriculum and teachers' subject knowledge. This study was general in terms of the covered topics by serious games. Similarly, Razak, Connolly and Hainey (2012) conducted a survey with primary school teachers to gauge the use of digital game-based learning in teaching in Scotland. 31 out of the 62 respondents never used educational games in teaching. Problem solving was ranked first with 100% in terms of the skills that can be obtained from computer games. The motivation of the

teachers for using games were mainly because students enjoy learning using computer games. The teachers rated the difficulty to assess learning gain and the lack of skill and training as the top two obstacles of using games for teaching. 91.4% of the teachers agreed that the benefits of using games in teaching that it transforms learning into a fun, motivating and engaging experience. Both studies targeted teachers and teaching in primary school. Thus, there is a need to check if there are any use of serious games in teaching at a university level and if there are any willingness to use them.

The study (Lahtinen et al, 2005) published a web-based questionnaire in which 34 responses were generated from teachers of programming modules. The questionnaire aimed at investigating the difficulties in learning programming and to find out what are the most difficult concepts to learn. The results showed that the most difficult concepts to learn are error handling, abstract data types, recursion, pointers and using data libraries which are considered as advanced programming concepts. Input-output handling, parameters and arrays were considered the most difficult basic programming concepts. The results found that dividing functionality into procedures and designing a program to solve a certain task as the most difficult issues in learning programming. Further, the results showed that according to the teachers, example programs and interactive visualisations are the most important kind of materials that would help students in learning programming. Similarly, Dale (2006) conducted an online survey, which generated 351 responses from instructors of CS1 module. Problem solving was highlighted as a main problem for students. Moreover, the results showed that the most difficult topics for the students varied but topics such as recursion, loops, arrays, inheritance and input-output handling were the most difficult. The results stated that the use of serious games for teaching computer programming topics could

be an answer to improve the teaching methods. Both studies were conducted at a university level and they recommended the use of serious games as a possibility to improve the teaching and enhancing students' understanding.

However, before applying and using serious games in teaching computer programming at a university level. There are several points that must be answered, such as if there were any previous use of such approach in teaching and if there are any willingness from the teachers to use such approach. Moreover, there is a need to investigate if there are any differences between developed and developing countries in terms of the possibility and willingness to use serious games in teaching computer programming. This information can be obtained through questionnaires or interviews. Interviews were chosen over questionnaire, because more information, details and elaboration can be obtained (Kothare, 2004). Bhattacherjee (2012) stated that "Interviews are a more personalised form of data collection method than questionnaires, and are conducted by trained interviewers using the same research protocol as questionnaire surveys (i.e., a standardised set of questions).". The interview consists of a set of questions where the interviewer read the questions for the interviewee and fill in the answer. According to Monette et al. (1986) "an interview involves an interviewer reading questions to respondents and recording their answers". The interviews will be answering the following research questions:

1. What is harder for students, problem solving or coding?
2. Have there been any use of serious games in teaching computer programming?
3. Are there any willingness from the teachers to use serious games in teaching computer programming?

4. What are the most difficult programming concepts for students to understand?

5. Are there any differences in the achieved results between the developed and developing case study countries?

3.3 The best approach for using serious games

In section 2.5 in the Literature Review chapter; previous studies about the use of serious games in teaching computer programming have been explored and compared. The comparison showed that various games were used and several programming languages were being taught. Further, the sample sizes for the studies varied greatly and different data collection methods have been used such as surveys, pre-post-tests or a combination of both.

Several factors affect the success of using simulation software and serious games in teaching. First, learning environment, which should be interactive, flexible and personalised (Malliarakis et al, 2015) regardless delivered through lectures, lab sessions or assignments (Sun et al, 2008). Teo (2014) stated that learning environment is a key factor that influences the success of using e-learning. Second, usage of space, the use of simulation software can overcome the problems associated with traditional learning related to space (Navimipour and Zareie, 2015; Trziu and Vrabie, 2015; Xu et al, 2014; Wang, 2014). Third, students' access, whether limited or unlimited. Simulation software and serious games deliver 24/7 access to learning materials, which has a massive influence on the success of the learning process (Zareie and Navimipour, 2016; Omar et al, 2012; Gunasekaran et al, 2002). Fourth, staff contacts' time, whether the use of simulation software and serious games requires extra staff time or not.

Once the software is perceived to be easy to use, then no previous staff experience is required (Capece and Campisi, 2013; Rubin et al, 2013) and, thus, no extra staff contact time. On the other hand, if the software is complicated then extra preparation is needed, which leads to extra staff time. However, the 21st century students who are raised in a digital world are familiar with complicated technologies and computer games are part of their everyday life (Malliarakis et al 2015). Fifth, cost, the use of simulation software provides a cost-effective approach for reaching different learners and meeting continuous learning requirements (Chen, 2014; Lee et al, 2011).

Overall students' experience and satisfaction is one of the most crucial factors for the success of using serious games in teaching and training. Another crucial factor is learning environment because it controls 4 other factors, which are staff contact time, students' access, usage of space and cost. Table 3.1 shows the controlled factors by learning environment.

TABLE 3.1: The controlled factors by the learning environment factor

Factors / Learning environment	Lectures	Assignments
Staff contact time	Time limited to usual lecture or extra lectures	Time to reply to students queries and questions
Students access	Limited	Unlimited
Usage of space	Use of computer lab	No usage of space
Cost	The cost of the serious game (usually free) and the cost of running the lab	The cost of the serious game (usually free)

Table 3.2 compares the previously presented attempts of using serious games for teaching computer programming in terms of the country, learning environment, staff contact time, student access, usage of space and cost.

Based on the World Economic Situation and Prospects book (2018), the countries were classified based on their economy. Figure 3.1 shows the percentage of the studies that were conducted in each country group.

TABLE 3.2: Serious games studies summary 2

Study	Country	Learning environment	Staff contact time	Student access	Usage of space	Cost
(Zhang et al., 2013)	USA	Tutorials alongside the traditional lectures	Extra lectures (tutorials)	Limited access	Lab	None
(Malliarakis et al., 2014)	Greece	Labs alongside the traditional lectures	Extra lectures (labs)	Limited access	Lab	None
(Corral et al., 2014)	Spain	Labs alongside the traditional lectures	Extra lectures	Limited access	Lab	None
(Eagle and Barnes, 2008)	USA	Traditional lectures	Extra lectures	Limited access	Lab	None
(Ross, 2002)	USA	Assignments	Regular lectures	Unlimited access	None	None
(Baker et al, 2012)	USA	Labs alongside the traditional lectures	Extra lectures	Limited access	Lab	None
(Liu, 2008)	Canada	Assignment	Regular lectures	Unlimited access	None	None
(Al-Linjawi and Al-Nuaim, 2010)	Saudi Arabia	Extra lectures	Extra lectures	Limited access	Lab	None
(Hillyard et al, 2010)	USA	Assignment	Regular lectures	Unlimited access	None	None
(Miljanovic and Bradbury, 2017)	Canada	Assignment	Regular lectures	Unlimited access	None	None
(Zhao et al, 2019)	Ireland	Labs alongside the traditional lectures	Regular lectures	Limited access	Lab	None
(Watson and Lipford, 2019)	USA	Labs alongside the traditional lectures	Regular lectures	Limited access	Lab	None
(Agalbato and Loiacono, 2018)	Italy	Assignment	Regular lectures	Unlimited access	None	None
(Comber et al, 2019)	Austria	Labs alongside the traditional lectures	Regular lectures	Limited access	Lab	None
(Jordaan, 2018)	South Africa	Labs alongside the traditional lectures	Regular lectures	Limited access	Lab	None
(Malliarakis et al, 2017)	Greece	Labs alongside the traditional lectures	Extra lectures	Limited access	Lab	None
(Erol and Kurt, 2017)	Turkey	Labs alongside the traditional lectures	Extra lectures	Limited access	Lab	None
(Topalli and Cagiltay, 2018)	Turkey	Labs alongside the traditional lectures	Regular lectures	Limited access	Lab	None
(Zhao et al, 2018)	3 European countries	Labs alongside the traditional lectures	Extra lectures	Limited access	Lab	None
(Mathrani et al, 2016)	New Zealand	Labs alongside the traditional lectures	Regular lectures	Limited access	Lab	None
(Ouahbi et al, 2015)	Morocco	Labs alongside the traditional lectures	Extra lectures	Limited access	Lab	None
(Rozali and Zaid, 2017)	Malaysia	Labs alongside the traditional lectures	Extra lectures	Limited access	Lab	None
(Rajeev and Sharma, 2018)	USA	Labs alongside the traditional lectures	Extra lectures	Limited access	Lab	None
(Galgouranas and Xinogalos, 2018)	Greece	Labs alongside the traditional lectures	Extra lectures	Limited access	Lab	None

The figure shows that 75% of the studies were conducted in developed countries, in which there are no barriers in terms of the computing infrastructure or technology budget. Only 25% of the studies considered developing countries to test the use of serious games in teaching computer programming. While applying serious games in teaching computer programming in developing countries, several barriers appeared, such as the fragile computing infrastructure, lack of technology budget (Talebian et al, 2014; AlAmmary, 2012) and cultural issues (Farid et al, 2015). There are not enough studies and research to prove the possibility of the successful implementation and usage of serious games for teaching in

developing countries. This formulates the need for more studies to investigate the feasibility and possibility of using serious games for teaching in developing countries.

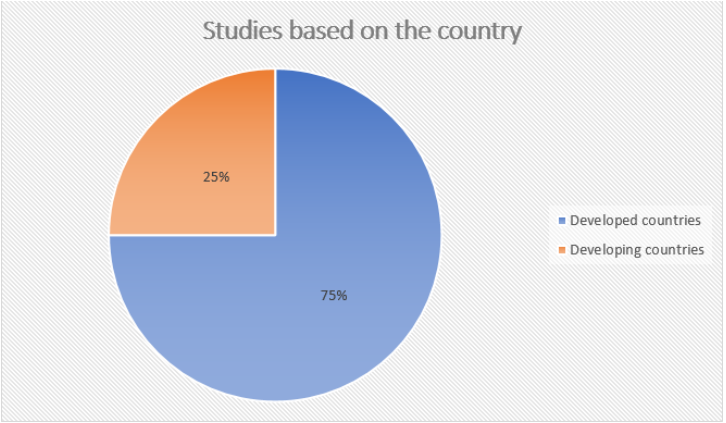


FIGURE 3.1: Studies by country.

Further, all of the presented studies concluded that serious games enhanced the students understanding of the covered topics and/or increased the motivation of the students. As shown in table 3.2, the usage of serious games can be done by either assignments or lectures. Any of the two choices can affect and change the staff contact time, the learning environment, usage of space and the students access of the serious game. Yet, no research compared the two approaches and investigated which approach achieves better results. This forms a gap that needs to be addressed, explored and analysed.

Thus, there is a need for designing an experiment to investigate the effect of using serious games in teaching computer programming and to find the best approach to use such alternative. Further, there is a need to conduct the experiments in a developing country to investigate the feasibility and possibility of using serious games for teaching in developing countries. Moreover, while conducting such experiment, there is a possibility to look at the gender perception of using serious games since some studies stated that female students do not like games. This forms an opportunity

to investigate the engagement of users from different genders. Designing and conducting the experiments will be answering the following research questions:

1. Does the use of serious games affect students' understanding in computer programming concepts?
2. What is the best approach for using serious games in teaching computer programming?
3. Are there any differences in using serious games in teaching computer programming between developed and developing countries?
4. Does gender affect the engagement in using serious games for teaching purposes?

3.4 The used serious game

In section 2.6 in the Literature Review chapter; several serious games for teaching computer programming were explored and compared. The selected serious game for the experiment was Robocode which is short for "Robot Code". It is an open source Java-based virtual robot game that is intended to teach object-oriented programming concepts. The Robocode game consists of a robot-development tool and it simulates a virtual battlefield where robots can battle against each other. The player programs the robot commanding it how to perform and respond to events arising in the battlefield. Thus, Robocode forms a space for students and learners to learn and apply their knowledge in OOP. It covers writing classes, reading, analysing and using existed code, event handling and message passing (Bonakdarian and White, 2004). Robocode battles are running in real-time and on-screen. The game starts at least with two robots and

each one of them starts with the energy of 100 and dies when it drops to zero. The game ends when there is only one robot left on the battlefield or when the time runs out. Robot class is automatically generated when creating a new robot and the class contains a run method with an infinite loop that defines the default behaviour for the robot. Additionally, the object has methods such as onScannedRobot, onHitWall, and onHitBy-Bullet which are responsible for handling a response to a particular event by calling other methods which will perform some actions either by moving the robot or investigating about the opponent robot. Table 3.3 shows few methods and the description for each one.

TABLE 3.3: Robocode methods

Method	Description
ahead(distance)	Moves the robot forward with the distance passed.
back(distance)	Moves the robot backwards with the distance passed.
turnLeft(degrees)	Turns the robot's body to the left by degrees passed.
turnRadarLeft(degrees)	Turns the robot's radar to the left by degrees passed.
turnGunLeft(degrees)	Turns the robot's gun to the left by degrees passed.
fire(power)	Fires a bullet with the firepower passed value.
scan()	Scans for other robots.
getDistance()	Returns the distance of the scanned robot

There are different alternatives for Robocode, such as Scratch, which was developed at the MIT Media Lab and its free to use. Scratch is a software that allows users to program and create interactive programs using animations and share their programs online. Therefore, Scratch helps students to think creatively and work collaboratively. Lamb and Johnson (2011) defined Scratch as "In computer software, scratching refers to reusable pieces of code that can easily be combined, shared, and adapted. Students can create stories, games, art, music, animations, and much more". The study (Resnick et al, 2009) stated that Scratch can be used from schools to universities as introductory to programming. In addition the studies (Ornelas Marques and Marques, 2012; Calder, 2010) referred to Scratch

programming language to overcome problems like critical thinking and problem solving. Moreover, Lee (2011) concluded that instructors can benefit from Scratch by developing creative and entertaining materials. However, Scratch doesn't involve writing code, where players use drag and drop to build the program which forms a disadvantage compared to Robocode. Also, Robocode in its nature is a competitive serious game, unlike Scratch.

What makes Robocode a good game for learning is that it represents the object with a visual activity. The students or learners can see the results and consequences of their implementation and calculations live on the battlefield. Furthermore, the rules of the game in terms of losing and gaining energy requires deep thinking and push students or learners to use different and analysed strategies. Because when a robot fires a bullet, the same robot loses energy with the same amount of the firepower, which makes the game not only about firing bullets. However, if the bullet hits another robot, the robot will gain back some energy and the opponent tank will lose four times the firepower. Hence, making a decent robot requires such an implementation that effectively fires and dodges bullets.

A notable feature of Robocode is that it allows the users to instantly view and test their robots' behaviour against different provided sample robots. This makes the testing and debugging easy and interactive. Thus, making the game covers several stages of the software development process. Moreover, the battling and challenging feature of the game provides a competitive and fun factor that acts as an attraction feature for the students. The game acts as a motivational element to impulse the students to study and understand different programming concepts to be able to create a robust robot, which will lead into adding to the overall students' experience. Robocode creator Mat Nelson stated that Robocode "like

chess, simple to learn, difficult to master" (Triplett, 2002), since a simple robot can be developed in minutes, where a sophisticated robot can take months of development. This means that Robocode can be used on various levels of students and even on experienced Computer Science graduates. Further, Hartness (2004) stated that Robocode can be used to encourage students to master difficult concepts and apply their knowledge in an interesting situation. Also, they can use Robocode to apply their knowledge without the need for mastering all the details of the game.

3.5 Evaluating serious games

In section 2.7 in the Literature Review chapter; 18 different quality characteristics of serious games were described and their use in the literature were highlighted. Table 3.4 presents the characteristics and divide them into primary and secondary characteristics. The following subsections will explain further the primary and secondary characteristics. If the absence or a poor representation for one of the characteristics will limit or eliminate the benefits of a serious game it will be considered a primary characteristic. Also, many characteristics are interlinked to each other. Thus, there might be some highly important characteristics and yet they are considered secondary because they are part of another primary characteristic.

3.5.1 Primary quality characteristics

Table 3.4 highlighted 8 primary quality characteristics based on the criteria that their absence will prevent serious games from delivering its

TABLE 3.4: Primary and secondary quality characteristics

Quality characteristic	Primary or Secondary	Reason
Game design	Secondary	Part of other characteristics such as usability and user's satisfaction.
User's satisfaction	Primary	Consists of several characteristics such as usefulness, user interface and efficacy
Usability	Primary	One of the most measured characteristics
Usefulness	Secondary	Part of other characteristics such as learning outcomes and user's satisfaction
Understandability	Primary	Forms the second level of Bloom's taxonomy and considered very important
Motivation	Primary	Considered as the main purpose of using serious games
Performance	Secondary	Part of other characteristics such as learning outcomes and user's satisfaction
Playability	Secondary	Part of other characteristics such as learning outcomes, usability and user's satisfaction
Pedagogical aspects	Primary	It forms the difference between video games and serious games
Learning outcomes	Primary	It forms the difference between video games and serious games
Engagement	Primary	Consists of several characteristics such as enjoyment, cognitive behaviour and social impact
User's experience	Primary	Consists of several characteristics such as user's satisfaction, usability and enjoyment.
Efficacy	Secondary	Part of other characteristics such as learning outcomes and user's satisfaction
Social impact	Secondary	Very limited use and part of the engagement and user's experience characteristics
Cognitive behaviour	Secondary	Very limited use and part of the engagement and motivation characteristic
Enjoyment	Secondary	Part of other characteristics such as motivation, engagement and user's experience
Acceptance	Secondary	Part of other characteristics such as learning outcomes, user's satisfaction and user's experience
User interface	Secondary	Part of other characteristics such as usability and user's satisfaction.

educational content to a designated audience effectively. Learning outcomes evaluation is widely considered as a primary characteristic along with pedagogical aspects characteristic. They form the difference between video games and serious game in which their occurrence makes games have educational content and purpose. For evaluating these characteristics an experiment must be conducted in which a control group and an experiment group should take a place to measure if a serious game has

succeeded in delivering its educational content by comparing the results of the two groups. Evaluating the learning outcomes of a serious game without conducting experiments could only be done to evaluate the potential of a serious game and its suitability for the selected audience only. Similarly, the user satisfaction characteristic must be measured by running an experiment and collecting feedback from users. This characteristic depends on many other quality characteristics such as game design and interface. Engagement, motivation and user experience are primary characteristics since their existence in a serious game pushes the user to engage effectively with a game and with other users. Another primary characteristic is usability, which is considered one of the most measured quality characteristics for serious games. Understandability characteristic refers to the clarity of a game and measuring this characteristic is limited in the literature. Yet, its absence prevents the serious game from delivering its content and thus not achieving its purpose. Because if the game is not easy to be understood and comprehended, then the user will not be able to use it. For that, understandability is considered a primary characteristic in this study.

3.5.2 Secondary quality characteristics

Table 3.4 highlighted 10 secondary quality characteristics which are either part of other primary characteristics or they are not crucial to the success of serious games in delivering its educational content for the designated audience effectively. Social impact and cognitive behaviour quality characteristics have limited usage in evaluation frameworks and can be found mainly in evaluating serious games that are meant for children and thus they are considered as secondary characteristics. Evaluating game design and user interface characteristics are limited and these two characteristics are linked to each other. They depend on two primary

characteristics, which are usability and user satisfaction and, thus, there is no need to measure them independently if the other primary characteristics are measured. Acceptance, usefulness, performance, efficacy and playability characteristics are part of other primary characteristics. Measuring learning outcomes and user's satisfaction will provide an overall view of these characteristics. Enjoyment characteristic relays on several secondary characteristics like playability and game design. Also, it is part of the motivation and engagement characteristics and can be measured as part of the other primary characteristics rather than being measured on its own.

In section 2.8 in the Literature Review chapter; several previous evaluation studies were presented. Few studies focused on measuring the quality characteristics of serious games for the purpose of highlighting the games' strengths and weaknesses. Yet, there are flaws in these studies, which form a need to develop a framework that measures the primary characteristics of the game and applying it on the used serious game. Developing and applying this framework will be answering the following research questions:

1. What are the characteristics of the used serious game that needs to be improved?
2. How the used serious game can be improved?
3. What are the effects or changes on the used serious game evaluation after the changes, if any?

3.6 Summary

This chapter acted as a summary and conclusion for the Introduction and the Literature Review chapters. This chapter presented further comparisons of previous studies highlighting the need for data collection to fill in the gaps. This chapter formed all the research questions that this study aims to answer. The following is the list of all the research questions:

1. What are the percentages of attrition in computing majors in developed and developing case study countries from the year 2010 to 2015?
2. What are the percentages of failure in the first programming module in developed and developing case study countries from the year 2010 to 2015?
3. Is there a decline in the total number of students or the new enrolled students in computing schools in developed and developing case study countries from the year 2010 to 2015?
4. What is harder for students, problem solving or coding?
5. Have there been any use of serious games in teaching computer programming?
6. Are there any willingness from the teachers to use serious games in teaching computer programming?
7. What are the most difficult programming concepts for students to understand?
8. Are there any differences in the achieved results between the developed and developing case study countries?

9. Does the use of serious games affect students' understanding in computer programming concepts?
10. What is the best approach for using serious games in teaching computer programming?
11. Are there any differences in using serious games in teaching computer programming between developed and developing countries?
12. Does gender affect the engagement in using serious games for teaching purposes?
13. What are the characteristics of the used serious game that needs to be improved?
14. How the used serious game can be improved?
15. What are the effects or changes on the used serious game evaluation after the changes, if any?

The next chapter will describe the data collection methods and the analysis techniques that will be used to answer the previously mentioned research questions.

Chapter 4

Methodology

This research aims to study, analyse and propose a methodology to enhance the students understanding of programming concepts and help retaining students by using serious games. This will lead to a reduction of the attrition rate phenomena in computing majors, in which most of the attrition occur after taking the introductory programming module. Moreover, the research aims to improve the serious games by evaluating primary quality characteristics of the serious games and highlighting weak points that can be improved to increase the potential of the used serious game.

According to Kumar (1996), there are two main categories for data collection, which are secondary and primary. Secondary data refers to the data that is already collected and just needs to be extracted, such as government publication, earlier research and records. Primary data refers to data collected by observations, interviewing and questionnaires. Thus, secondary data will be collected, which are university records and primary data will be collected through interviews, experiments and questionnaires.

4.1 Secondary Data

Data will be collected from university case studies in the UK and Jordan for exploring the points that are listed below:

- The total number of students each year from 2010 to 2015 in computing majors.
- The number of new students each year from 2010 to 2015 in computing majors.
- The number of students who changed their major from computing to any other major or dropped from the university from 2010 to 2015.
- The failure rate for the first programming module each year from 2010-2015.

The first two points will provide numbers to be investigated if there is a drop in the number of students in computing majors or a decline in terms of new students enrolling in computing majors from the year 2010 to 2015. The third point will highlight the attrition that occurs in computing majors and will provide numbers to be compared to the total number of students in the computing school. The fourth point will highlight the failure rate for the introductory to programming module, which is believed to be the reason behind the high attrition rate. Data will be collected from universities in case study countries in the UK and Jordan. The results will be compared to highlight any similarities and differences.

4.2 Interviews

There are different types of interviews that can be used to collect data, such as the structured interviews, which is based on fixed questions directed to the interviewee. This type of interviews limits the interviewee answers and doesn't encourage justifications and comments and offers limited flexibility. According to (Berg, 2007), in structured interviews limited freedom is offered for both the interviewer and the interviewee. This type of interviews will limit the possibility of gaining extra information from the interviewees, such as attaining their opinions on different things. Another type of interviews is the open ended or unstructured interviews, which contrary to structured interviews it offers greater flexibility and freedom for interviewees (Gubrium and Holstein, 2002). However, the use of unstructured interviews can be time consuming and the major problem is the difficulties in comparing the data because the types of answers are unpredictable. A semi-structured interview is another type of interview that has the flexibility of the unstructured interviews and the formality of the structured interviews. Berg (2007) stated that "it allows for in-depth probing while permitting the interviewer to keep the interview within the parameters traced out by the aim of the study". The semi-structured interviews advantages are the freedom offered to the interviewees to express their opinions while maintaining a reliable and comparable data.

Since the freedom of expressing the teachers' opinions and point of views is highly important while sustaining comparable data. Semi-structured interviews will be used to attain information regarding the initiatives of using simulation software and serious games in teaching computer programming in universities in case study countries in the UK and Jordan. The interviews will show if there are any use of serious games in teaching

computing modules in general and specially computer programming and will open the opportunity for the teachers to express their opinions on the use of simulation software and serious games in teaching computing modules. Also, it will reveal the willingness of the teachers to use simulation software and serious games in teaching. Further, the interviews aim to investigate the most difficult programming concepts for first year computing students from the teachers' perspective. This will help in determining what serious game should be used in the experiments to match the students' needs.

The interviews will be conducted in universities in the two case study countries. One as a developed and one as a developing country.The experiments will be conducted in the same universities. This will allow the data attained from the interviews with the teachers to be compared with the data collected from running the experiments with the students. Further, the interviews will be conducted under the condition that only teachers who teach programming modules will be interviewed. This will lead to a small number of interviews but it will be representative and accurate. Moreover, the results will be comprehensive because the rating of the programming concepts will be based on their experience in teaching rather than guessing. Moreover, they are the teachers who may have used serious games in teaching before or may consider using it in the future. Results will be compared to highlight any differences in terms of teachers' opinions on the use of serious games in teaching computer programming and teachers' point of view on the most difficult computer programming concepts for students.

The interview questions (provided in Appendix A) starts by asking about the interviewee's degree and his/her experience in teaching in general and in teaching computer programming. The interviewee will be asked,

which is harder problem solving or coding and will express his/her opinion on what programming style should be introduced first for students (Object first or Procedural first). The rest of the questions were based on several studies. For example, the question about the difficulty in learning programming concepts was derived from the study (Lahtinen et al, 2005). In this question, the teachers are asked to use 5 Likert scale to rate the difficulty of learning the following programming concepts for students:

- Expressions and Conditions
- Methods and Encapsulation
- Objects and Classes
- Loops
- Arrays
- Inheritance
- Abstraction
- Polymorphism
- Recursion

The questions about whether serious games were used in teaching or not, if there are any willingness to use serious games in teaching and if using serious games in teaching will motivate and help students in understanding programming concepts were derived from the studies (Allsop and Jessel, 2015; Razak et al, 2012). The interviewee will be able to elaborate, explain and give examples for clarification purposes since this is a semi-structured interview.

4.3 The best approach for using serious games

The experiments aim to investigate the positive impact of using serious games in teaching computer programming. Further, the experiments aim to find the best approach for using serious games to increase the potential of the used software at a university level. Experiments will be conducted in universities in case study countries in the UK and Jordan.

Jordan was selected as a case study of a developing country because as shown in section 3.3 in the Research Questions and Design chapter, there are limited studies that applied the use of serious games in teaching in developing countries. Furthermore, several studies highlighted different barriers when implementing and using serious games in developing countries. Conducting the experiments in Jordan will investigate these barriers and examine the possibility and feasibility of using serious games for teaching in developing countries.

In this section, a description of students' sample that the framework will be applied on is presented. Then, the experimental design defines the 3 different groups, which are involved in the experiments, describes the framework and the tests and describes the process of evaluating the results. Next is a section that shows the used serious game and why it was chosen and a tasks section, which describes the tasks allocated to the different groups. Finally, there is a data analysis section that describes the analysis methods to be applied on the collected data.

The following sub-sections explain the study experiment.

4.3.1 Students Sample

Recruiting participants is a challenge, but since our target is university students, announcements will be made in the lectures about the experiment and the benefits that students will gain from participating. The announcement day will be after the teachers covered the loops, arrays, objects and classes concepts and not at the beginning of the semester. Because students will get distracted from all the information they will get at the start of the semester and to ensure that all the students understand that this experiment and its tests are independent of the module itself. Ethical approval is required to carry out this study. Thus, in the case study university in the UK, an ethical approval form will be filled in and submitted to the school for confirmation (provided in Appendix B). For the case of universities in Jordan, a letter will be prepared from Queen's University Belfast requesting the permission to conduct the study (provided in Appendix C). After obtaining ethical approval, the experiment will be described for all the students on the announcement day highlighting all the benefits students can gain from participating along with the certificates for participants and rewards. Students who want to participate will read and sign a consent form (provided in Appendix D) that describes the experiment from the beginning to the end.

To decide who to recruit, the population must be analysed. Since the study focuses on teaching programming concepts, we need to check the population that want to participate in the experiment in terms of any previous programming experience and any experience in using the serious game that we will use. Students with previous programming experience apart from high school programming materials and any previous knowledge of the chosen serious game will be excluded. The study (Feigenspan

et al, 2012) found that self-estimation of programming language experience for undergraduate students correlates with programming tasks. Thus, participants will be asked about their programming experience and about any familiarity with the chosen serious game before including them in the experiment.

With regards to the student sample and as Singh (2006) stated "the size of the sample depends upon the precision the researcher desires in estimating the population parameter at a particular confidence level. There is no single rule that can be used to determine sample size. The best answer to the question of size is to use as large a sample as possible". Moreover, the study (All et al, 2016) conducted a semi-structured interview with experts who were defined as "staff members of an organisation with a specific professional function and a specific experience and knowledge for this purpose". The experts were chosen to have at least a PhD degree or still conducting relevant research, which evaluates educational interventions. The aim of the interviews was to define the preferred methods for conducting digital game-based learning effectiveness studies. The results reported that an absolute minimum suggested by the experts is 20 participants per condition. Therefore, the study aims to recruit as many students as possible and ensure that a minimum of 20 participants is allocated to each group.

4.3.2 Experimental Design

This section will describe the design of the experiment. The chart in figure 4.1 presents the study experiment. Following a Comparative design, which is used to compare the effectiveness of different treatment modalities (Kumar, 1996). For example, to compare the effectiveness of three

teaching models (A, B and C) on the level of comprehension of students in a class. The experiment design consists of three groups as follow:

1. The Control Group (CG): students will only take Pre- and Post-Tests and the questionnaire.
2. The First Experiment Group (EG1): students will take the Pre- and Post-Tests and the questionnaire. Students in this group will take an induction lecture followed by an assignment, where they will use the serious game to learn and apply their knowledge of programming. The group will take 2 weeks to complete the assignment.
3. The Second Experiment Group (EG2): students will take the Pre- and Post-Tests and the questionnaire. Students in this group will take lectures, where they will be guided to use the serious game to finish the same assignment as students in EG1. The group will take 4 lectures, 2 hours each to complete the assignment.

All, Castellar and Looy, (2016) conducted interviews with experts to define the preferred methods for conducting digital game-based learning effectiveness studies. The results of the study showed that randomisation has been accepted by all experts as the preferred method for assigning the participants to condition. However, matching has been suggested by most of the experts (12 out of 13) as a method to guaranteeing similarity between conditions and controlling for certain variables. Table 4.1 provides an overview of variables to match participants in different conditions as suggested by the experts.

The students' distribution in the groups will be based on their previous knowledge represented by the pre-test scores (provided in Appendix E). Students will be divided equally into the three groups according to their scores instead of randomly distributing them to avoid bias. Part 1 in

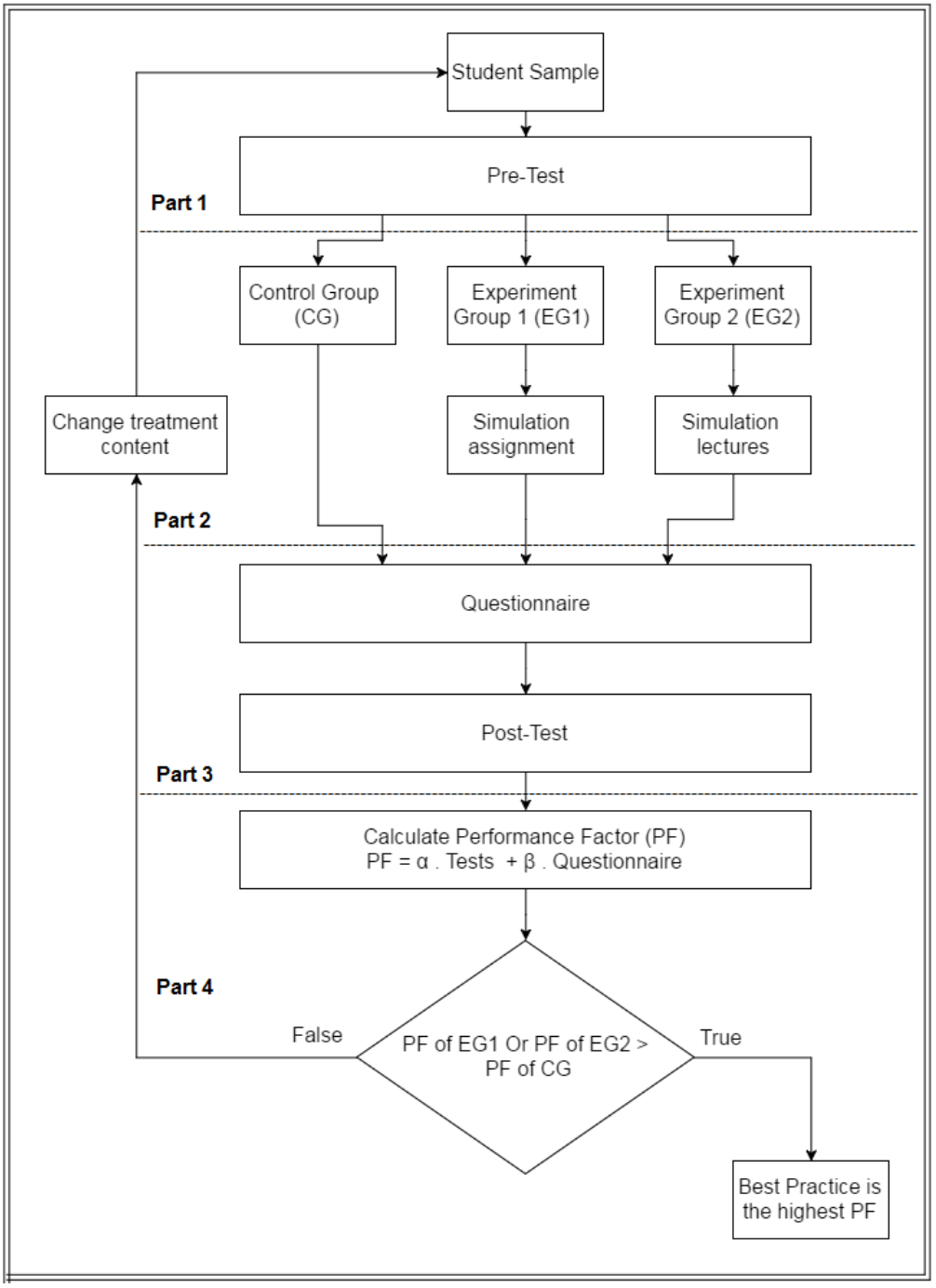


FIGURE 4.1: Research experiment.

figure 4.1 shows that the chosen student sample will take a pre-test and based on the results, the students will be divided into the three groups shown in part 2 of the figure.

Part 3 of figure 4.1 shows that the students will be asked to fill in a questionnaire (provided in Appendix F) about how satisfied they are with their

TABLE 4.1: Variables suggested to match on.

Variable	Description
Previous knowledge	Matching on prior academic achievement or pre-test scores
Ability	Matching on different ability levels (e.g. Low, medium and high achievers)
Motivation	Matching on motivation towards the learning content
Game experience	Matching on previous experience with games
Gender	Matching on gender (male/female)
Age	Matching on age/age categories
SES	Matching on socio-economic status

programming skills and their understanding with their effort before taking the post-test. This is to ensure that the students' answers are not based on them being able to answer the post-test questions or not. The students will also take a post-test (provided in Appendix G). After completing the test and as shown in part 4 of figure 4.1. The Performance Factor (PF) will be calculated for all three groups using the formula 4.1. Tests is the result of the analysis of the post-test. Questionnaire is the result of the questionnaire that students filled in. α and β are weight factors and the weight was chosen to be 50%, where students satisfaction will be given the same weight as the students understanding.

$$PF = (\alpha * Tests) + (\beta * Questionnaire) \tag{4.1}$$

Since we have two experimental groups and one controlled group, and the data collection will be based on a between-subject design, where different groups of people are assigned to each group. The tasks that will be assigned to the groups are different from one group to another. But the common task will be the pre-post-tests and the questionnaire. All three groups will take the same tests and fill in the same questionnaire. The EG1 involves playing a serious game as an assignment, where students will be given a short induction lecture and they will be provided with the required tools to complete the task. They will be given 2 weeks to complete their task. The students in this group will have a 24/7 access to the

serious game either from the university or from their personal computer devices. On the other hand, the EG2 is limited to 4 lectures, 2 hours each, where students will be guided on how to use the serious game to complete the required task. The students will have access to the serious game through the university computers only and have limited hours. Table 4.2 shows the location, covered topics and the duration for the experimental groups.

TABLE 4.2: Task design

	Experiment group 1 (EG1)	Experiment group 2 (EG2)
Location	Any	University lab
Covered topics	Variables, methods, object, string, loops and arrays.	Variables, methods, object, string, loops and arrays.
Duration	2 weeks.	4 lectures, 2 hours each.

4.3.2.1 Pre- and Post-tests

The pre- and post-test (provided in Appendix E and G) will be used to measure students understanding of a specific computer programming concepts, such as variables declaration, methods calling, strings, loops and arrays. Both tests are formed of 20 multiple choice questions each and designed using (Alliger and Horowitz, 1989) concept of multiple-choice tests to eliminate the guessing from changing the score of the test. Each question has a subsequent question asking if the student knows the answer or the student is just guessing. Figure 4.2 shows an example. In the pre- and post-tests, the questions are divided as follow:

1. The first 2 questions are about variables and methods declaration.
2. Questions (3,4,6) are about calling methods.
3. Question 5 is about creating an object.
4. Questions (7,8,9,10,11) are about Strings and its methods.

5. Questions (12,13,14,15,16) are about Loops.

6. Questions (17,18,19,20) are about Arrays.

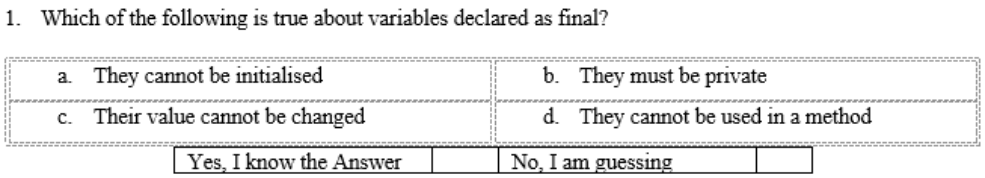


FIGURE 4.2: Questions example.

Using this concept in the tests allows to eliminate the guessing factor or knowing the correct answer by luck. Also, by using this concept we can get the answers in the traditional way by just marking the right answer or we can add the filter "Yes, I know the answer" so we can get the mark based on the actual understanding of the topic that is being tested. Moreover, using this concept can reveal the misunderstanding of a certain topic when the answer is wrong but marked as "Yes, I know the answer". When using this concept of multiple-choice tests in a pre-and post-tests to check for possible improvements for a certain group after taking the treatment. Another factor can be identified and analysed, which is either the increase or the decrease of confidence in the knowledge when answering the questions between the pre-and the post-tests.

4.3.2.2 Questionnaire

Hays (2005) stated that comparing a treatment group with a control group that does not engage in any educational exercises will lead to an overestimation of the beneficial effects of the used serious game. The same applies when comparing the level of students' satisfaction in the method of learning if they didn't use it. Further, the study (Butt and Rehman,

2010) presented a conceptual framework showing that students' satisfaction is affected by teachers' expertise, courses offered, learning environment and classroom facilities. Thus, the students' satisfaction questionnaire should be general and not specific to the used serious game because one of the three groups will not use the serious game. The questions for the questionnaire were derived from the study (Lahtinen et al, 2005). The questions aimed to find out the difficulties in learning computer programming and what kind of materials the students find most effective for learning.

The questionnaire is provided in Appendix F, and it consists of 5 questions. The first question asks about the gender of the student. The second question asks the students if they have any prior programming experience whether from a previous course, a hobby or self-learning. Any student with an extensive previous programming experience will be excluded from the analysis because the focus in this research is on fresh students taking introductory to programming. When the students were recruited, they were asked about their programming experience and about any familiarity with the chosen serious game before including them in the experiment. This question acts as an extra filter for the inclusion criteria. Questions 3,4 and 5 are designed as 5 Likert-scale type questions. Question 3 asks the students to evaluate the following statements:

- I enjoy programming
- I enjoy problem solving
- I have good programming skills
- I am interested in alternatives to lecture-based teaching styles

The results of this question will provide an overview of the students' satisfaction and confidence in their programming skills. The students

then will be asked to use 5 Likert scale to rate the difficulties they face when learning computer programming, which are:

- Using program development environment (e.g. eclipse)
- Learning the programming language syntax
- Solving a certain problem
- Writing code for a solved problem
- Finding bugs from my own work
- Lack of visualisations

This question will highlight the difficulties that students may face when learning computer programming. Finally, the students will be asked to use 5 Likert scale to rate the difficulty of learning the following programming concepts:

- Expressions and Conditions
- Methods and Encapsulation
- Objects and Classes
- Loops
- Arrays
- String

This question will be used to highlight the difficult programming concepts from the student perspective. This question is the same one included in the interview questions. However, students will not be asked to rate all the programming concepts as the teachers because the students are in their first year and they didn't cover all the programming concepts.

The results obtained from the students will be then analysed and compared between the three different groups, to test if the results will be affected by the use of the serious game or not, and how the serious game will be presented for the students. The results attained from the questionnaire are given 50% of the weight of the evaluation, where it provides the satisfaction level of the students and self-evaluation in the knowledge of computer programming.

4.3.3 Variables

We have one Independent Variable (IV) with three levels, each level assigned to a group. The CG has class lectures as the first level. The EG1 has the serious game assignment along with the class lectures as the second level. The EG2 has the extra serious game lectures along with the class lectures as the third level. Table 4.3 shows the three IV levels, their assigned group and their allocated task.

TABLE 4.3: Independent variable levels

Independent variable (IV) levels	Group	Task
Level 1	Control group (CG)	Attend class lectures.
Level 2	First experimental group (EG1)	Attend class lectures and take the serious game assignment.
Level 3	Second experimental group (EG2)	Attend the class lectures and extra serious game lectures.

The study has one dependent variable (DV), which is students' understanding that will be measured by the marks of the post-test. Students in all three groups will take the same test at the end of the experiment as shown in part 3 in figure 4.1.

4.3.4 The used serious game

In section 2.6 in the Literature Review chapter; several serious games for teaching computer programming were explored and compared. The

serious game that will be used is Robocode. Section 3.4 in the Research Questions and Design chapter highlights the advantages of the serious game Robocode over other games and justifies why it is best suited for this experiment.

4.3.5 Tasks

The common tasks for the students in all three groups are the pre-post-tests and questionnaire, which are part 1 and 3 in figure 4.1. Students in the first and second experimental groups will be asked to develop a robot using the serious game Robocode. The students will use all their programming knowledge and skills in developing their robots. Table 4.2 shows the different tasks assigned for the two experimental groups, which represents part 2 in figure 4.1.

Students in the first experiment group will be given a tutorial document, which covers the basics of the serious game Robocode to start with. They will be given the freedom of time and resources to use for the development of their robots. Students in the second experimental group will take 4 tutorial lectures by a tutor who has good knowledge in Robocode development. The first two lectures will cover the basics of Robocode and will help students to build a basic robot. In the last two lectures, the students will be given the choice to choose their robot design and behaviour and work on it with the help of the tutor. The goal for the students in the two experimental groups is to develop a robot that can compete in a battlefield against other robots, so it can enter the competition.

4.3.6 Data Analysis

The analysis for the experiment will be divided into two parts. The first part is a statistical analysis part. Since we want to compare the mean of different groups. A T-test can be used to compare the different groups in pairs and repeat this process to test the multiple groups. However, using multiple t-test comparison is not appropriate statistical practice because the chance of committing a type I error (false positive, rejecting the null hypothesis when it is true) is high (Weaver et al, 2017). Instead, Analysis of Variance (ANOVA) can be used to compare the mean of multiple groups. For example, the study (Ashby et al, 2011) used ANOVA to compare the student success between developmental maths courses offered online, blended and face-to-face.

Since we have three different groups, a one-way analysis of variance (ANOVA) will be used to compare the effect of the method of learning programming on the students understanding represented by the multiple-choice post-test they will take. Weaver et al (2017) stated five assumptions that must be satisfied before using ANOVA, which are:

1. Data type: "The dependent variable must be interval or ratio. Additionally, the independent variable should have two or more categorical groups". This means that the dependent variable must be continuous, such as height measured by cm or exam scores measured by numbers. The independent variable represents the independent groups and normally ANOVA is used when there are three or more categories or groups, such as ethnicity or treatment type. ANOVA can be used for two groups but using t-test is more common in this situation.

2. Distribution of data: “The data follow a normal distribution. This includes the dependent variable’s distribution within each category (group) of the independent variable”. This means that a test for data normality must be applied on the dependent variable for each independent variable level separately. The one-way-ANOVA is robust to violations of normality, which means that this assumption can be a little violated and still provide valid results.

3. Independent samples: “The samples are independent (with the exception of rANOVA); independent samples have no effect on one another”. It can be also referred as independence of observations, which means that there must be different participants in each group with no participant being in more than one group. This assumption is a matter of design, in which using a between-subject design will satisfy this assumption. The between-subject design means that different groups of people are assigned to each group unlike the with-in subject design were the same group will do all the tasks.

4. Homogeneity: “The variance of the populations must be equal, meaning the populations must have an equal spread around the mean”. This means that there needs to be homogeneity of variances. This assumption can be checked using Levene’s test for homogeneity of variances.

5. Random sampling: “The observations are randomly sampled and are independent from one another”. This means that the groups variances must be homogeneous. The independent samples part has been described in point 3.

If all the previous assumption were met and ANOVA was used. The result will only show if there is a significant difference in the mean between the tested groups or not. If the result showed that there is a significant

difference in the mean. It will not determine which group is statistically different from the other. In order to specify where the differences lie, there is a need to use post-hoc tests, such as Tukey honestly significant test (HSD) (Weaver et al, 2017).

The second part of analysing the data will be following the framework in calculating the performance factor for each of the three groups as shown in part 4 in figure 4.1. The results of the test and questionnaire are combined. Three questions were chosen from the questionnaire that reflects students' satisfaction. All three questions are 5 Likert-scale type. Table 4.4 shows the number of elements in each question and the best answer.

TABLE 4.4: Questions description

	Question 1	Question 2	Question 3
Number of elements	4	6	6
Maximum value	5	5	5
Minimum value	1	1	1
Best answer	5	1	1

To calculate the result for each question, the median will be employed as the measure of the central tendency to calculate the result of each element in the question because we are dealing with ordinal data (Clegg, 1998). The average for each question will be calculated by adding the values of each element and dividing it by the number of elements for each question.

Following the performance factor formula and to achieve a result out of 100 for students' satisfaction, the following formula will be implemented, where valx represents the question number. Each question is multiplied by a value that serves as a percentage to give weight for each question. The weights were divided equally based on the number of elements in each question and the weights given for the three questions are (30%, 35%, 35%) respectively. Assigning the weights was based on the number of elements in each question. Question two and three contain 6

elements each while question one contains only 4 elements. Thus, more weight has been given to question two and three.

The best answer for question one is 5 while for question 2 and three is 1. In order to make 5 the best answer for all the questions, subtraction has been applied for the results of the second and third questions.

$$Questionnaire = val1 * 6 + (6 - val2) * 7 + (6 - val3) * 7 \quad (4.2)$$

The average results for the post-test will be calculated out of 100 for the three groups and then the performance factor formula 4.1 will be applied.

In the studies of using serious games for teaching computer programming and as shown in section 2.5 in the Literature Review chapter. Several studies used questionnaires, where others used tests to evaluate the used serious game. None of the studies provided a proof on why one data collection method is better than the other. Thus, combining the two different data collection methods will offer a better and more comprehensive understanding for the effect of the used serious game.

Since this study is using two different data collection methods and it aims to combine their analysis together. Weights must be assigned to the two data collection methods. α and β in the performance factor formula are the weights. The question that is being faced here is how to assign the weights.

There is no rule to follow to assign the weights because none of the listed previous studies combined the analysis of questionnaire and tests. The studies conducted the analysis separately in the cases, when they used both of the data collection methods. Most of the previous studies used either pre-post-tests or questionnaires to measure the effect of serious

games. This shows that both data collection methods are informative in this research area and can be used individually. Thus, this study treats the two collection methods the same and assigns the same weigh, which is 50% for both α and β.

4.4 Evaluation questionnaires

The evaluation consists of analysing all the measured quality characteristics of serious games. Then, developing a framework to evaluate serious games for the aim of highlighting weak characteristics and points that can be improved, which will lead to improve and enhance the serious game. In this section, key quality characteristics are explored, explained and described how it fits into the framework. Then, students sample is described and the goals of this experiment is listed. Then, a brief description of the used serious game is presented and tasks section, which describes the tasks allocated to the participants. Finally, a data analysis section, which shows how the collected data will be analysed.

4.4.1 Framework development

The literature evaluating serious games is diverse and various quality characteristics can be used to evaluate several aspects of serious games. In section 2.7 in the Literature Review chapter; 18 different quality characteristics of serious games were described and their use in the literature were highlighted. In section 3.5 in Research Questions and Design chapter; the characteristics were divided into primary and secondary characteristics as shown in table 3.4. The following section will show how the framework was designed.

4.4.1.1 The framework

The aim of the framework is to evaluate serious games to highlight the games' strengths and weaknesses. This will allow the researchers and developers to enhance the serious games by improving the weak characteristics.

After dividing the quality characteristics of serious games into primary and secondary characteristics. A framework will be developed to evaluate serious games, which is formed out of the primary quality characteristics. The quality characteristics that were marked as primary are user's satisfaction, usability, understandability, motivation, pedagogical aspects, learning outcomes, engagement and user's experience.

The evaluation framework can be designed from all 8 primary quality characteristics. However, the learning outcomes and the pedagogical aspects characteristics can't be measured by a questionnaire. You can only measure their potential in terms of the covered topics. For example, a study can examine a serious game to evaluate how many programming concepts are covered and how many details are offered. But to evaluate the learning outcomes and pedagogical aspects, an experiment should take place to measure the effect of using the serious game on students while comparing their improvement with a control group using different data collection methods. This experimental design for assessing the learning outcomes and also student's satisfaction for using a serious game is available in section 4.3 in the Methodology chapter. Thus, learning outcomes, pedagogical aspects and user's satisfaction will be excluded from the evaluation framework.

The framework will consist from the five remaining primary quality characteristics, which are usability, understandability, motivation, engagement and user's experience as shown in figure 4.3. The design of the framework will be based on MEEGA evaluation model (Savi et al., 2011; Savi et al., 2012). The MEEGA model consists of a questionnaire to collect data on the reaction of the students after playing the game.

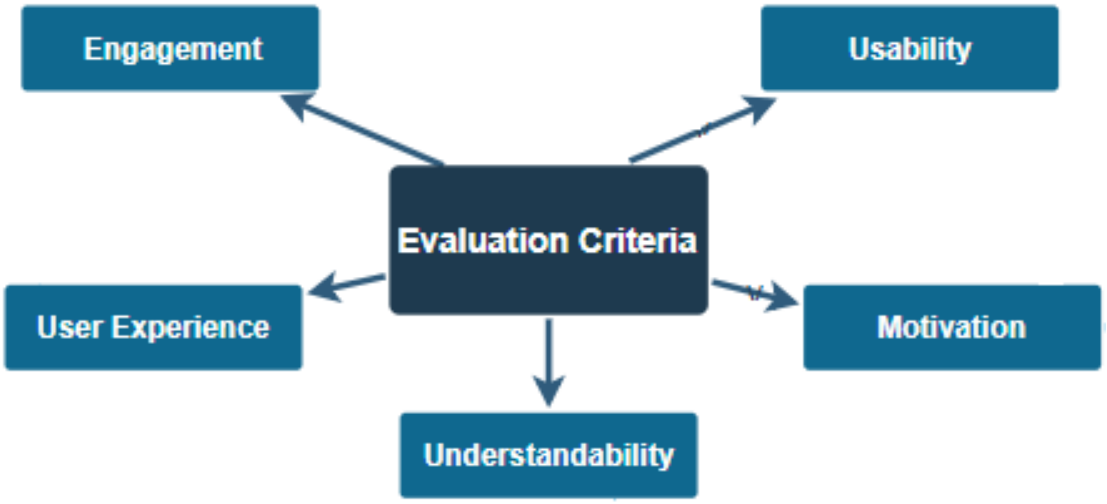


FIGURE 4.3: Five Dimensions Evaluation Criteria.

4.4.1.2 The evaluation framework questions

The quality characteristics evaluation framework is formed of the five chosen primary quality characteristics, which are usability, motivation, engagement, user experience and understandability. After analysing each of the five characteristics. Three factors were chosen to represent each characteristic and the framework represents each factor by a question. Only three factors were chosen to avoid redundancy and to ensure the same weights are given to the factors and the characteristics.

The evaluation questionnaire is provided in Appendix H. Below is the list of the characteristics, the factors and the questions, which the students will be asked to rate.

1. Usability and of measures learnability & usefulness, errors and ease of use. Under the usability characteristic, the following questions are asked:

   (a) The game is a useful learning tool for computer programming.

   This question measures the Learnability & usefulness factor and checks if the game is a useful learning tool and if it offers a learnable content.

   (b) The game does not contain errors.

   This question measures the errors factor and checks if the game contains errors, which may affect the usability of the game.

   (c) The game is easy to use.

   This question measures the ease of use factor and checks if the game is easy to be played and used.

2. Motivation and of measures challenge, enjoyment and curiosity. Under the motivation characteristic, the following questions are asked:

   (a) The game is challenging.

   This question measures the challenge factor and checks if the game offers an interesting environment to the users and challenge them.

   (b) Playing the game is enjoyable.

   This question measures the enjoyment factor and checks if the game is enjoyable since serious games primary purpose is educational and not entertainment and, thus, some serious games might not be entertaining to play.

   (c) Playing the game sparked my curiosity.

   This question measures the curiosity factor and checks if the game increases the user's desire to learn and immerse him/her.

3. Engagement and of measures purpose, interest and control. Under the engagement characteristic, the following questions are asked:

   (a) The purpose of the game is appealing.

   This question checks if the objectives and goals of the game are engaging, appealing and attractive.

   (b) The idea and the story-line of the game is interesting.

   This question checks if the game's scenario and theme are interesting and exciting.

   (c) The game is self-controlled by the user and allows making decisions.

   This question checks if the game is self-controlled by the user and allows making decisions and choices that will lead to particular results and consequences.

4. User experience and of measures competence, social interaction and immersion. Under the user experience characteristic, the following questions are asked:

   (a) The game is competitive.

   This question checks if the game is competitive by allowing and promoting the comparison and/or differentiation between different users work.

   (b) The game allows social interaction.

   This question checks if the game endorses and stimulates interaction between different users.

   (c) The game promotes the involvement of the learner.

   This question checks if the game encourages the involvement of the users and increases the participation.

5. Understandability and of measures clarity, simplicity and independence. Under the user experience characteristic, the following questions are asked:

   (a) The game goals are clear and understood easily.

   This question checks if the game and its goals are clear and easy to be understood.

   (b) The game offers a set of straightforward steps to be followed.

   This question checks if the game offers a set of straightforward steps to be followed.

   (c) The game can be played individually without the need of assistance.

   This question checks if the game can be played individually without the need for assistance or support.

Each quality characteristic has 3 statements, in which each statement represents a factor. Figure 4.4 shows the evaluation criteria including the factors for each quality characteristic. The students will rate each statement from 1-10 based on their experience after using the serious game. The following section will explore the students sample that will be included.

4.4.2 Students Sample

As shown in the previous section, the framework doesn't measure learning outcomes or students' understanding of programming topics. But rather, it focuses on the students' reaction after using and playing the serious game. Thus, there is no need for the students to be at a certain level of knowledge in computer programming. However, the level of the participants will affect their rating of the characteristics of the serious game.

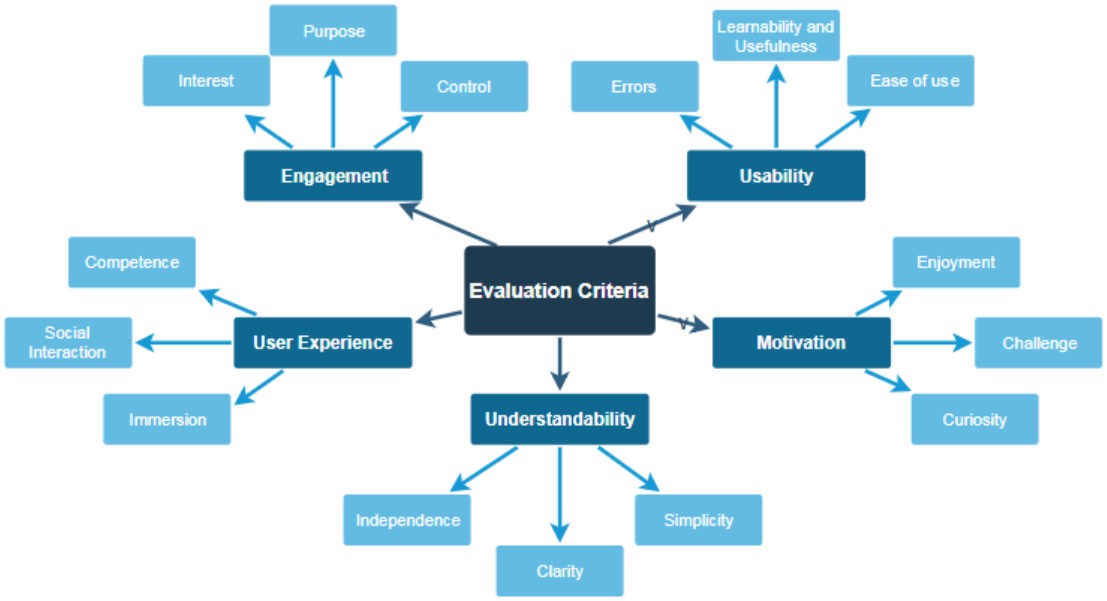


FIGURE 4.4: Five Dimensions Evaluation Criteria with the factors.

Experienced participants with computer programming and/or computer software knowledge will perceive and apprehend the serious games differently than less experienced participants. Therefore, to achieve more accurate results and to cover different level of students. The evaluation will be conducted with both university and school students, in which the results will include students with different backgrounds and varied experience with computer programming.

The evaluation targets are both university and school students. Thus, arrangements will be made with the module coordinator or teacher at the university level and with the school teacher at the school level. Ethical approval will be verbally agreed with the module teacher in the university and the teacher in the school and the study will be conducted under their supervision. Students with any previous knowledge of the used serious game will be excluded because we want to ensure that the evaluation will not be affected by any prior experience of using the serious game.

4.4.3 Goals

The goal of this experiment is to have an overview evaluation for key quality characteristics of serious games. This will help to identify and highlight weak points of the used serious game, which will mark a route for changes and amendments that will lead to improve the serious game. Evaluating Robocode based on the chosen quality characteristics with university and school students will offer a clear view of the game from students' perspective. The results obtained from the evaluation will be used as an input for the improvement phase, in which the weak quality characteristics, if any, will be addressed in the game and proper changes will be applied. Upon completing the changes and improvements. Another experiment for evaluating the edited version will be conducted the same way the first evaluation experiment was completed to monitor any changes in the evaluation.

4.4.4 The used serious game

Robocode is chosen for the evaluation because it will be used in other parts of this study. More importantly, it is an open source game, which will permits the code to be accessed and allows for any changes to be made. The only difficulty that will be faced that there is no documentation for the source code of Robocode. This makes it difficult to navigate through the code and make changes if required based on the results of the evaluation. Further description about Robocode is available in section 3.4 in the Research Questions and Design chapter.

4.4.5 Tasks

The students will be asked to use the chosen serious game Robocode for 20 minutes. After finishing, the students will be asked to fill in an evaluation framework that is based on the 5 chosen quality characteristics. Each characteristic has 3 statements and the students will rate each statement with a value from 1-10. If there will be any changes and amendments made to the serious game as stated in the Goals section number 4.4.3. Another evaluation for the edited version of the game will take place with the same tasks but with different students. This is to ensure that the evaluation will not be affected by any pre-knowledge of the game from the students.

4.4.6 Data analysis

The analysis of the evaluation questionnaire will be done by calculating the average for each quality characteristic out of 100% by combining the results of the three evaluation factors that forms each quality characteristic for each student. Then, the average result for all the students will be calculated. After completing the calculations, the results will be observed and if it was found that there is low result for a particular quality characteristic. Amendments and changes will be added to the serious game to try to improve the weak quality characteristic. Then, the same analysis will be applied using the same framework. In this case, we need to compare the evaluation before and after the changes have been made to the serious game. Since we have two groups, we need to apply a statistical analysis test, such as t-test. However, t-test is a parametric test and it can't be used to compare questionnaire results, which are considered ordinal data. A non-parametric test that is an alternative to the t-test can be applied, which is Mann-Whitney U test. For example, the study (Klasen et

al, 2000) used the Mann-Whitney U test to compare the German version of the strengths and difficulties questionnaire.

Thus, Mann-Whitney U test will be applied to compare the results before and after the changes to the serious game. Weaver et al (2017) stated four assumptions for using Mann-Whitney U test, which are:

1. Data type: "The Mann-Whitney U test can be used with a dependent variable that is ordinal, interval, or ratio.". For example, the Mann-Whitney U test have been used for Likert scale analysis (Barr et al, 2006).

2. Distribution of data: "The Mann-Whitney U test and Wilcoxon signed-rank test are non-parametric applications, therefore, the data can fall into a non-normal distribution". Even if the data are not normally distributed, the Mann-Whitney U test can be used. Unlike Wilcoxon signed-rank test, the Mann-Whitney U test can be used even if the shape of the differences between the groups is asymmetrical. For Mann-Whitney U test, if the shapes of the distributions are the same in the groups then the test compares the medians of the two groups. However, if the distributions are of different shapes, then the test compares the mean ranks of the groups.

3. Sampling groups and observations: "The Mann-Whitney U test assumes the two variables are independent". This means that the data collected from one group will not influence the other group.

4. Random sampling: "The observations are collected at random".

When using Mann-Whitney test, the null hypothesis depends on the assumption of data distribution. If the data distribution shapes were the same, then the null hypothesis H_0 will mean that the distribution of the two groups are equal. And the alternative hypothesis H_1 means that the

medians of the two groups are significantly different. However, if the data distribution shapes were not the same, then then the null hypothesis H_0 will mean that the distribution of scores for the two groups are equal. And the alternative hypothesis H_1 means that the mean ranks of the two groups are significantly different.

4.5 Limitations

The research will have few limitations, which are listed below:

- The secondary data and the interviews from the two countries serves as a case study between developed and developing countries and the results may not match other institutions in the case study countries or different countries.
- The secondary data may represent a short period of only five years because of limitations of the availability of data, which means that the results may be affected by the changes of this period.
- The timing of the experiment for finding the best approach for using serious games will be near the end of the semester because the students must cover several programming concepts in the course before being involved in the experiment. This forms a limitation in terms of the number of students that will participate in the experiment because the students by that time will have a lot of assignments and projects deadlines. Moreover, the students will be at the stage of preparing for exams. This will lead to fewer students participating in the experiment and will potentially cause students dropping from the experiment.
- The experiment for finding the best approach for using serious games will take a place over two weeks, in which a group of students will be

assigned to attend a 2 hours tutorial session twice a week. Finding a suitable time for the whole number of students can't be achieved, in which it will lead to students dropping from the experiment because they can't attend the tutorial sessions. This issue will affect the experiment in case study universities in Jordan more than the UK. Because in Jordan the students have the freedom to enrol in different modules including university and faculty optional and compulsory module, which makes it very hard to set a time for the tutorials that match all the students' timetables.

- The study will focus on a particular serious game for teaching computer programming and, thus, the framework and/or the results may not apply to other computer programming serious games or other disciplines serious games.

Delimitation in terms of ignoring some years in the secondary data collected from universities, because some universities don't have data that cover all the years that other universities do. To maintain a comparable data, the data will be only collected for a specified period. Moreover, the evaluation framework is consisted of five quality characteristics. Only three factors represent each quality characteristics because a lot of factors are interlinked. Adding more factors to the quality characteristics while avoiding redundancy will result in having different weight for different quality characteristics, which will result in an unbalanced analysis.

4.6 Summary

This chapter described the data collection methods to answer the research questions. The chapter highlighted how the frameworks were developed and how the data will be collected. Further, this chapter listed the analysis

techniques that will be used to analyse the data to answer the research questions. This chapter offered a detailed description for designing and conducting the two different experiments, which includes the design of the frameworks, the choice of the data collection methods, how the data will be collected, the tasks that are assigned to the participants and the data analysis mechanism. The next chapter presents the analysis of the data after conducting the experiments and collecting the data.

Chapter 5

Results and Discussion

This chapter explores the results achieved by collecting the data, conducting the experiments and the evaluation questionnaire. Further, the chapter discusses the results and answers the research questions highlighted in the methodology chapter. Section 4.1 reports the data collected from universities in case study countries in the UK and Jordan with regards to students' attrition and failure rates and the number of students in computing majors and new enrolments. Then analyses and discusses the data. Section 4.2 presents the results of the conducted interviews with teachers of the programming modules in case study universities in the UK and Jordan and then discusses the obtained results. Section 4.3 explains the conducted experiments in the UK and Jordan, describes how the framework was applied and then analyses and discusses the results obtained from running the experiments. Section 4.4 highlights how the evaluation framework was applied to the case serious game to evaluate several dimensions of the game and identify possible improvements. Then the section analyses and discusses the obtained results and highlights the changes applied to the game based on the framework recommendations. Finally, this section presents the results of validating the applied changes by conducting the evaluation on the edited version of the game.

5.1 Secondary Data

Secondary data were collected from case study countries in the UK and Jordan. Data were collected from Queen's University Belfast in the UK as a case of a developed country and from Applied Science University and Petra University from Jordan as a case of developing country. The data that were asked for from all the universities in the case study countries are as follow:

1. The total number of students in the computing faculty each year from 2010 to 2015.
2. The number of newly registered students in computing faculty each year from 2010 to 2015.
3. The failure rate in the first programming module each year from 2010 to 2015.
4. The number of students who changed their major from computing to any other major or dropped from the university each year from 2010 to 2015.

As noted in section 3.1 in Research Questions and Desgin chapter. The data collection period is set to start on 2010 and not earlier due to the limitation of the documented stats based on the initial communication with the universities in the developing case study country. The period will be extended if applicable when data is being collected subject to data availability and the time of data collection.

The collected data aimed to investigate the enrolment and retention rates in computing majors. Also, the data will provide the number of students who dropped or changed major and the failure rate for the introductory

to programming module. The following sub-sections aims to answer the research questions assigned to the secondary data collection.

5.1.1 What are the percentages of attrition from computing majors in developed and developing case study countries from the year 2010 to 2015?

Data were collected from Petra University for the number of students who changed their major from computing majors to any other major in another faculty. The data were collected in 2016. Petra University offers 4 different majors in the Information Technology faculty, which are Computer Science, Computer Information Systems, Software Engineering and Computer Networks. Figure 5.1 shows the percentage of attrition in Petra University from 2010 to 2015.

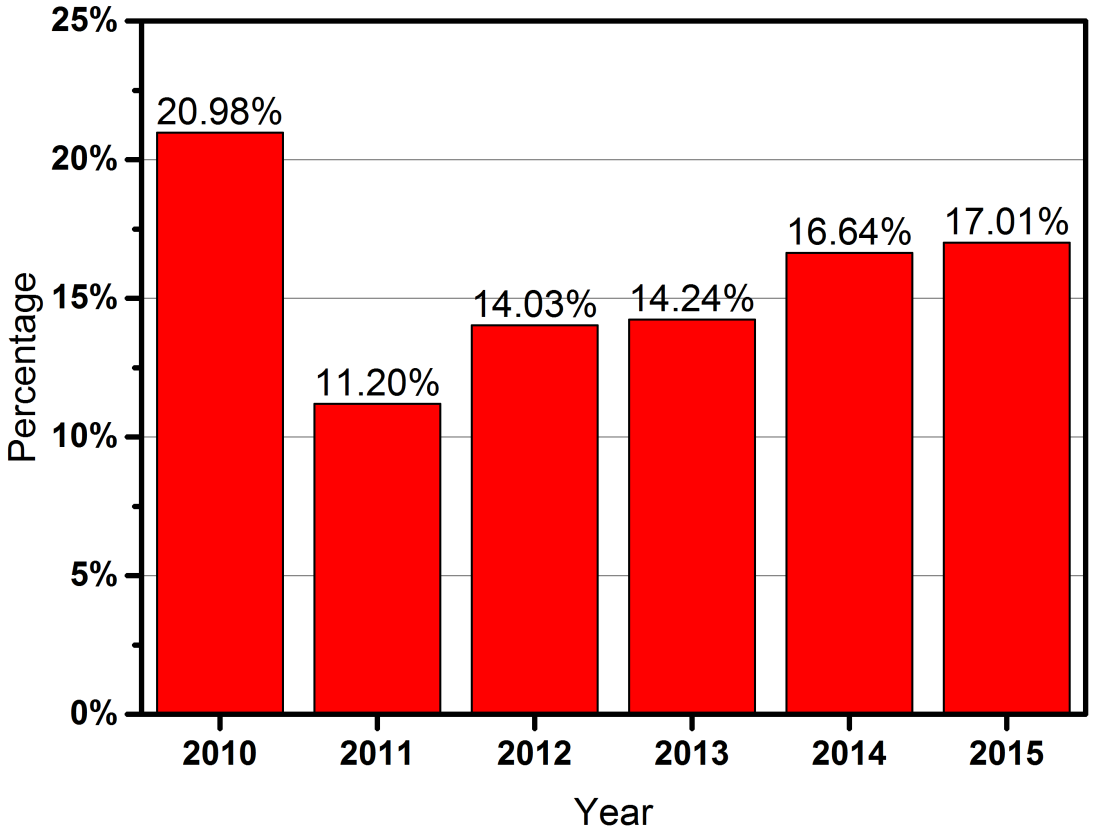


FIGURE 5.1: Petra University attrition rate from (2010-2015).

Data were also collected from Applied Science University for the number of students who changed their major from computing majors to any other major in another faculty. The data were collected in 2016. Applied Science University offers 4 different majors in the Information Technology faculty, which are Computer Science, Computer Information Systems, Software Engineering and Computer Networks Systems. Figure 5.2 shows the percentage of attrition in Applied Science University from 2010 to 2015.

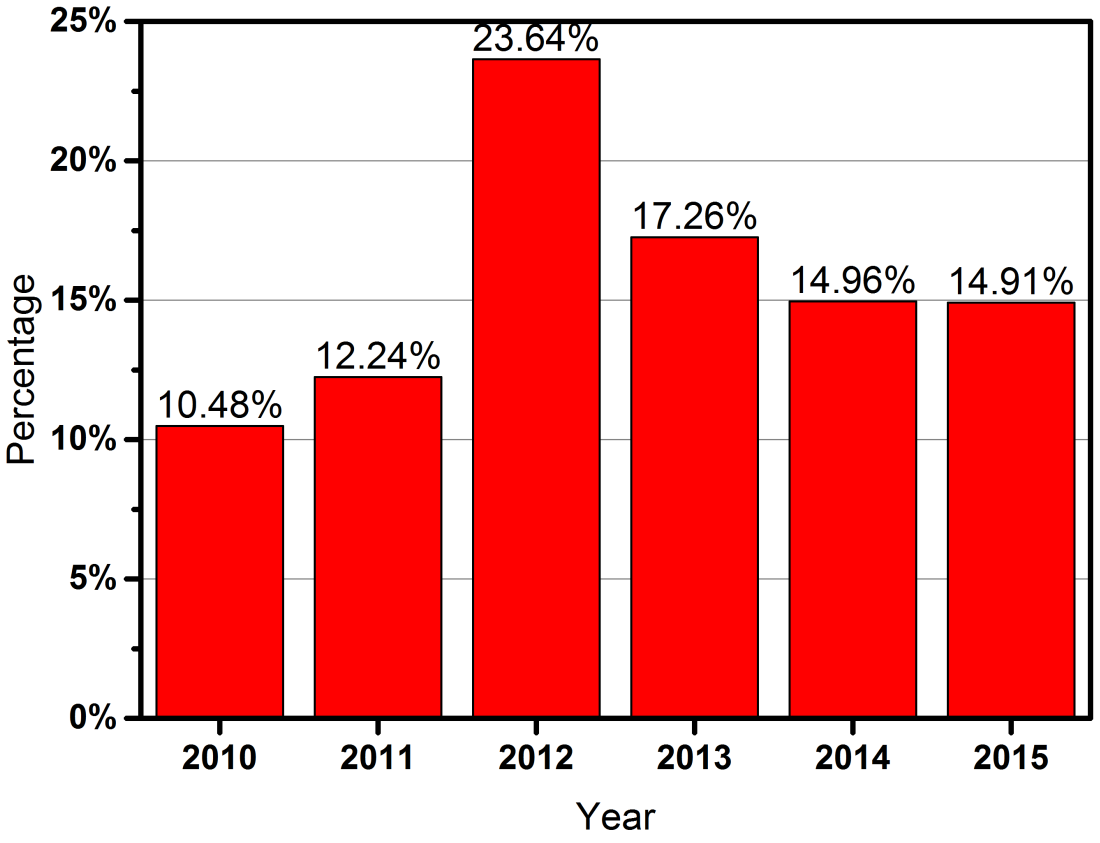


FIGURE 5.2: ASU attrition rate from (2010-2015).

Figure 5.3 shows the number of students who changed their major from computing majors to any other major in another faculty in Applied Science University and Petra University from the year 2010 to 2015.

By observing the previous three figures, it can be seen that the attrition rate is increasing in both universities. In Petra University, the peak

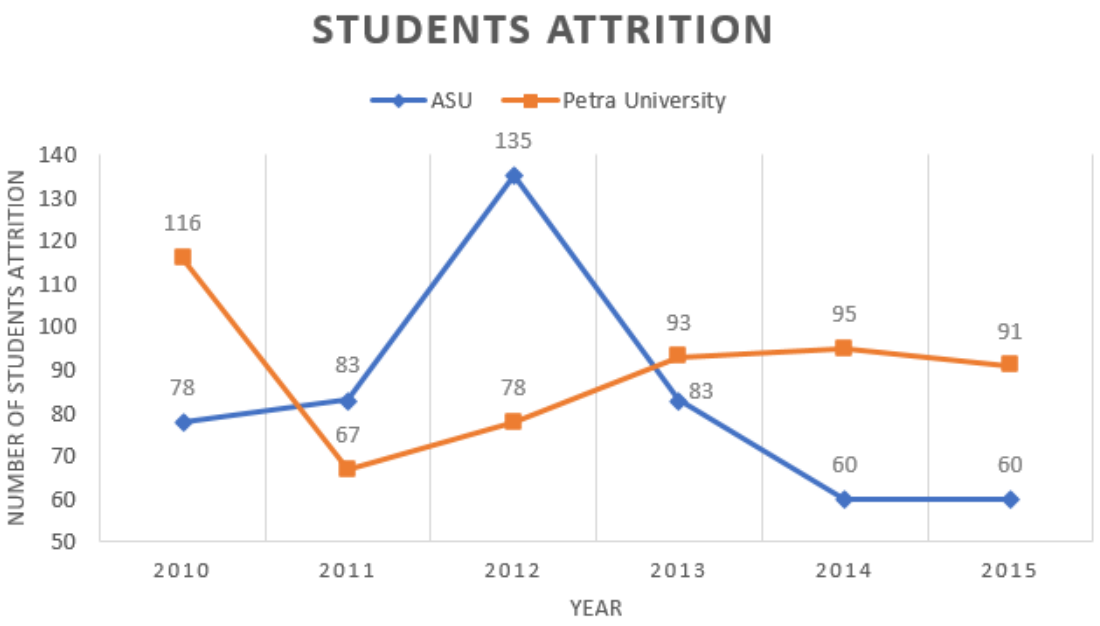


FIGURE 5.3: ASU and Petra University attrition by students numbers from (2010-2015).

of attrition was in 2010; in 2011 there was a huge decline in the attrition. However, after 2011 there was a steady increase in the attrition rate achieving 17% in 2015, which was the last measured year. In Applied Science University, there was an increase in the attrition from the year 2010 to 2015 with a peak in attrition occurs in 2013. The attrition rates in the case study universities in the developing country are high and increasing and match the numbers reported by the study (Azad and Shubra, 2010).

Data were collected in 2017 and 2018 in EEECS at Queen's University Belfast as a case study of a developed country. We have managed to collect the number of students who left the computing major for 2 years only, which are 2016 and 2017 and the numbers were 109 and 92 respectively. The number of students who left the major is alarmingly high. Bearing in mind that most of the attrition usually happens by the first-year students and in Queen's University Belfast the number of new students in 2016 and 2017 was 486 and 470 respectively. It can be seen that the number of students who left the major forms around 20% of the number of the

newly enrolled students and if all the students who left the major were newly enrolled students, which it is most likely the case. Then this forms a critical problem that must be highlighted, analysed and solved.

5.1.2 What is the percentage of failure in the first programming module in developed and developing case study countries from the year 2010 to 2015?

The failure rates in the introductory to programming module were collected from Applied Science University as a case study of a developing country from the year 2013 to 2015 due to the limitation of accessing previous years. The pass mark for introductory to programming module is 50 out of 100 and table 5.1 shows the failure rates from the year 2013 to 2015.

TABLE 5.1: Pass and Failure rate for the introductory to programming module in Applied Science University.

Year	Pass Rate	Fail Rate
2013/2014	81.5%	18.5%
2014/2015	85.6%	14.4%
2015/2016	75.8%	24.2%

The failure rates in the introductory to programming module were collected from Queen's University Belfast as a case study of a developed country. The pass mark for introductory to programming module is 40 out of 100. Table 5.2 shows the failure rates from the year 2010 to 2015.

TABLE 5.2: Pass and Failure rate for the introductory to programming module in Queen's University Belfast.

Year	Pass Rate	Fail Rate
2010/2011	80.6%	19.4%
2011/2012	84.6%	15.4%
2012/2013	89.6%	10.4%
2013/2014	78.7%	21.3%
2014/2015	79.5%	20.5%
2015/2016	77.2%	22.8%

It can be seen that the failure rates in the introductory to programming module are very high in both case study countries. This result confirms

the previous findings in the literature regards to the difficulties in teaching computer programming, which leads to low pass rates. The failure rates in Queen's University Belfast for the introductory to programming module are high, despite that the pass mark is 40. For example, in 2014/2015 the percentage of students who achieved a mark less than 40 was 20.5% and the percentage of students who achieved a mark between 40 and 49 was 15%. Also, in 2015/2016 the percentage of students who achieved a mark less than 40 was 22.8% and the percentage of students who achieved a mark between 40 and 49 was 10%. The failure rates for the introductory to programming module in both developed and developing case study countries are high and, thus, it must be analysed further to highlight possible causes of the problem and try to target and solve these problems.

5.1.3 Is there a decline in the total number of students or the new enrolled students in the computing schools in developed and developing case study countries from the year 2010 to 2015?

Data about the number of students in the computing faculties were collected from Applied Science University and Petra University as a case of a developing country from 2010 to 2015. Figure 5.4 shows the number of students. By observing the figure, it can be seen that the number of students is decreasing in both universities, which matches the results reported in the studies (Ali, 2009; Benokraitis et al, 2009).

There is no decline in the total number of students or the newly enrolled students in Queen's University Belfast. However, there is a problem, in which the number of newly enrolled students in the years 2015/2016 and 2016/2017 was decreasing. Yet, the total number of students for the same years was increasing, which highlights a problem where students do not graduate on-time and get stuck in the university. This issue must

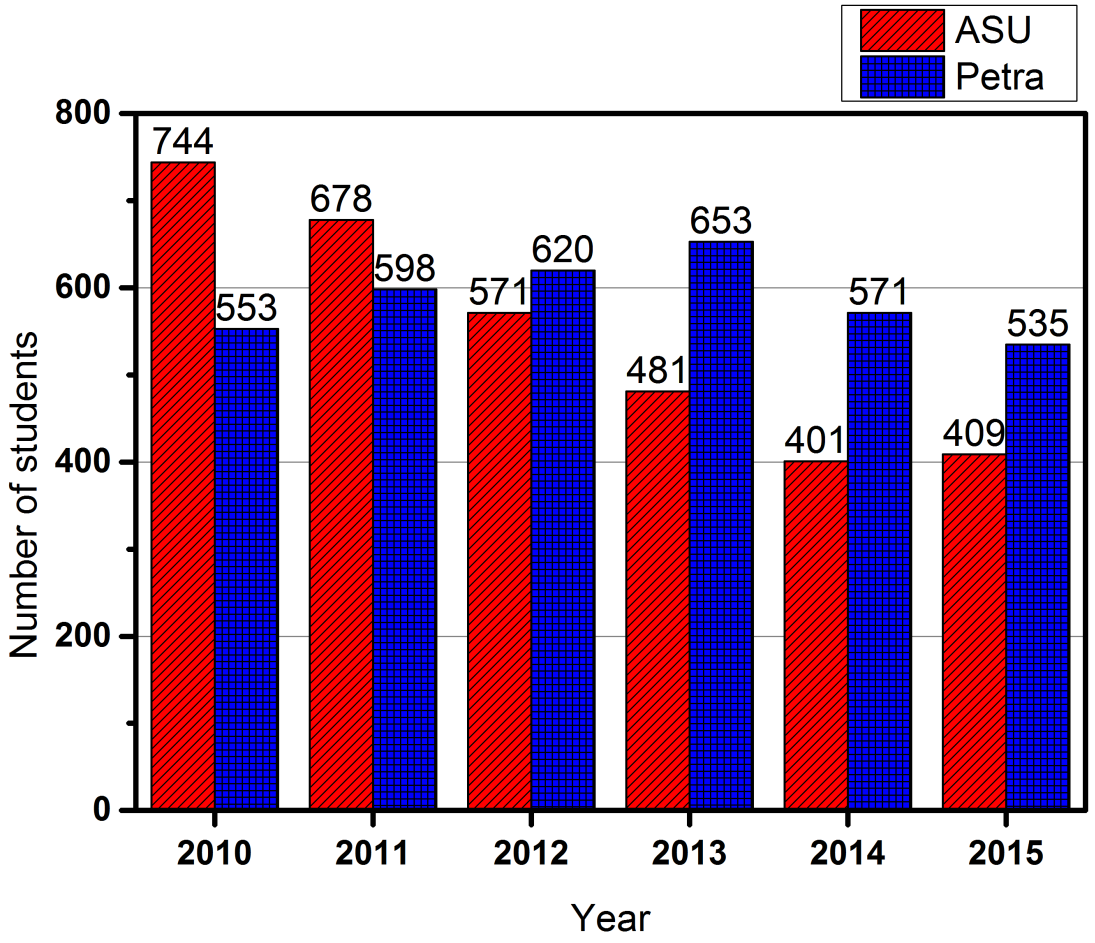


FIGURE 5.4: Number of IT Students in ASU and Petra University from (2010-2015).

be investigated further to identify its reasons and solve them. Table 5.3 shows the total number of students and the newly enrolled students in EEECS at Queen's University Belfast from 2010/2011 to 2017/2018.

TABLE 5.3: The total number of students and the newly enrolled students in EEECS at Queen's University Belfast from 2010\2011 to 2017\2018.

Year	2010/2011	2011/2012	2012/2013	2013/2014	2014/2015	2015/2016	2016/2017	2017/2018
Newly enrolled students	292	349	421	499	519	486	470	500
Total number of students	1048	1012	1266	1501	1689	1733	1756	1783

Several issues have been identified from collecting the data starting from the problem of high attrition rate to the complication of high failure rate in universities in case study countries in the UK and Jordan. Further, the problem in the declining of the number of students in the computing faculty in the universities in Jordan and the issue of students not graduating

on time in Queen's University Belfast in the UK.

5.2 Interviews

Interviews were conducted with teachers of programming modules in the UK and Jordan. A total of 26 interviews were completed, where 6 teachers were interviewed in Queen's University Belfast in the UK and 11 in Applied Science University, 4 in Petra University, 5 in the University of Jordan in Jordan. This sample size can be identified as representative because it reached the level of saturation, which means that additional participants will not provide any additional insights. Guest, Bunce and Johnson (2006) stated that the saturation often occurs around 12 participants in homogeneous group. The study meant by homogeneous group that the group has a particular position or level.

The interviews were conducted in 2016 and 2017. All of the interviewees have a degree in Computer Science, either master's degree or PhD with experience in teaching between 3 to 20 years. Figure 5.5 shows teachers experience in teaching. The interviews aimed at addressing the use of simulation software and serious games in teaching and to see if there were any use of this approach in teaching or if there are any willingness to be used in the future. The interviews also aimed at highlighting the most difficult programming concepts, which the students struggle to understand. The results of the conducted interviews can be accessed through the links provided in Appendix I. The following sub-sections aims to answer the research questions assigned to the interviews.

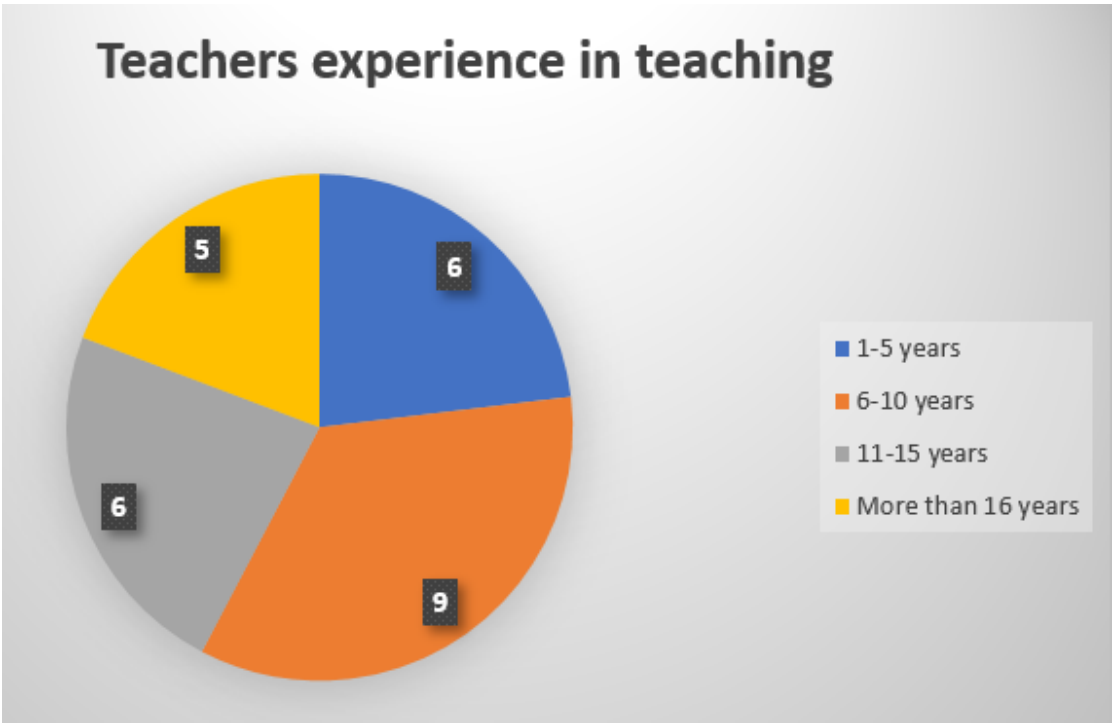


FIGURE 5.5: Teachers years of experience in teaching.

5.2.1 What is harder for the students, problem-solving or coding?

Most of the interviewed teachers stated that the students face difficulties in solving the problems more than coding the solution. One of the teachers referred that to the lack of confidence and knowledge as a major difficulty. Another teacher referred that to the poor background education. Only one teacher stated that coding is more difficult than problem-solving. Also, three teachers stated that problem-solving and coding have the same difficulty.

5.2.2 Have there been any use of serious games in teaching computer programming?

All the interviewed stated that no simulation software or serious games have been involved in teaching computer programming courses apart from one teacher who used Flash in teaching programming before. However, serious games have been used for teaching other computing topics.

Two teachers stated that they have used Scratch for teaching a multimedia course.

5.2.3 Are there any willingness from the teachers to use serious games in teaching computer programming?

Despite not using simulation software or serious games in teaching, teachers were enthusiastic to use such approach if it was proved to have good results and impact on students. Some teachers specified that it would have better results when used for students with no programming background. However, one teacher stated that he will not use such a method on a university level but on school students only. Further, all the interviewed agreed that using serious games and simulation software in teaching will motivate students and help them understand some programming concepts. Three teachers specified that there were some initiatives for using serious games in teaching programming. But the lack of time to try them prevented it from happening. The games were Mine craft, Alice, and Scratch.

The results achieved from this question matches the result obtained from a survey with 76 school teachers in Jordan (Assaf et al, 2019). The results showed that 84.2% of the teachers showed willingness to use serious games in teaching in classrooms. The rest 15.8% stated that they might use serious games and none of the teachers stated that they won't use serious games in classrooms. Further, the study results showed that 97.30% of the teachers believes that serious games can be used to acquire knowledge.

5.2.4 What are the most difficult computer programming concepts for students to understand?

The teachers were asked to evaluate the students' difficulty of few programming concepts based on their experience. The evaluation was on a scale from 1 to 5 (1 is easy, 5 is very hard). In the first interview, the teacher asked to make the scale from 1 to 10 so it can be more representative. Thus, the scale was changed from (1 to 5) to (1 to 10). Part of the interviewed teachers made a range for some concepts; the average was calculated for the range. Figure 5.6 shows the teachers evaluation for the difficulty of the programming concepts. Table 5.4 shows the average and the standard deviation of the teachers evaluation for the difficulty of the programming concepts.

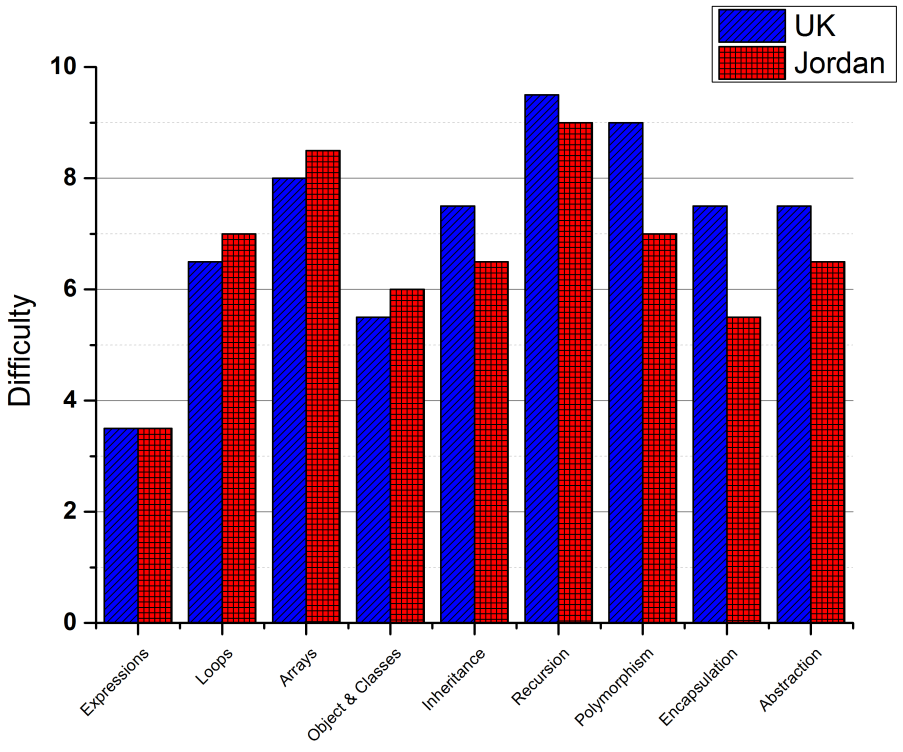


FIGURE 5.6: Teachers rating for the difficulty of programming concepts.

The interviews results marked loops and arrays as the hardest topics for students that are taken in the first programming module. Despite the result that shows concepts, such as recursion and polymorphism as the

TABLE 5.4: Teachers rating for the difficulty of programming concepts

Programming concept	Average	Std
Expressions	3.519	0.91
Loops	6.884	0.919
Arrays	8.384	0.605
Objects & Classes	5.903	1.157
Inheritance	6.730	1.358
Recursion	9.115	1.306
Polymorphism	7.461	1.567
Encapsulation	5.961	1.648
Abstraction	6.73	1.511

hardest topics, these topics are taken in the second semester. Students who struggled in understanding the first semester concepts will face extra difficulties understanding the second semester concepts, because the concepts will be harder and based on the first semester topics. Similarly to the study (Dale, 2006), which conducted an online survey with members of higher education institutions. It identified loops and arrays in the top three difficult themes for beginner programming students. Moreover, students tend to have difficulties in creating classes and objects and in creating and calling methods at the early stage of learning computer programming and some teachers referred that to the lack of practice.

Consequently, a number of serious games for teaching programming concepts were identified for the purpose of conducting experiments, such as Robocode, Alice, and Scratch. The selection of the appropriate game will be based on the comprehensiveness of the covered programming concepts, consideration of students' capabilities differences and attractiveness of the game.

5.2.5 Are there any differences in the achieved results between the developed and developing case study countries?

Overall, results obtained from the two case study countries showed that there were no use of simulation software and serious games in teaching

computer programming apart from one attempt done by a teacher in Jordan. Same results were obtained, which showed that there is willingness from teachers in the UK and Jordan to use simulation software and serious games for teaching computer programming if it was proven to be beneficial for students' understanding. In rating the difficulty of the programming concepts, results obtained from Jordan and UK were close to each other with a slightly more difficulty observed in Jordan for loops, arrays and object and classes. Further, there were a marginally difference highlighted in the second programming module topics, such as inheritance, recursion and polymorphism where the UK teachers ranked the difficulty of those topics more than the teachers in Jordan. No major differences were observed from running the interviews with teachers of programming courses in the UK and Jordan.

To conclude, although there was no real use of simulation software and serious games in teaching computer programming. There is willingness for using serious games in teaching computer programming from all the interviewed teachers. Furthermore, there were few initiatives and attempts to use this approach in teaching. It must be noted that out of the 26 interviewed teachers, only 10 teachers supported the object first approach which means the use of an object-oriented language as a first programming language to teach. 16 teachers supported the procedural first approach and in all four universities in which the interviews were conducted the approach used in teaching computer programming is object first. This highlights an issue that should be investigated further as it could be one of the reasons for the high failure rate of the programming modules.

5.3 The best approach for using serious games

The experiments were conducted in universities in case study countries in the UK and Jordan. Experiments were conducted in Queen's University Belfast in the UK as a case of a developed country and in Applied Science University, Petra University and the University of Jordan in Jordan as a case of a developing country. The experiments aimed to investigate the positive impact of using simulation software and serious games for teaching computer programming and to identify the best approach of using this method in teaching to increase the potential outcomes.

The following sub-sections describes conducting, analysing and the results of the experiments in the UK and Jordan.

5.3.1 UK experiment

After obtaining an ethical approval from EEECS in Queen's University Belfast. Meeting with the coordinator of the programming course for year 1 students took a place, where topics like how to interact with students and how to announce the competition were discussed. The transcript of the meeting is provided in Appendix J. Moreover, attaining a sufficient students' engagement was discussed and analysed in terms of how to make students participate in the experiment. Below are 6 points that were highlighted as the most important parts that will make the students take the decision to be involved in the experiment:

1. Enhance the student knowledge and learn new skills in programming.

The student will learn new programming skills and earn extra knowledge in OOP, including the use of APIs and reading code documentation.

2. Show and apply student skills in a visual way.

   The student will be able to program a robot and control its behaviour in the battle arena. This forms a transformation from viewing the code results in a black screen to an interactive visualisation.

3. A new experience for undergraduate students in programming games.

   The student will be programming the artificial intelligence (AI) of the robot, which is new for students at this level. Students will understand how games are programmed and will understand how the simple commands and concepts they learn in programming can be put together to build a game.

4. Competition.

   The students will program a tank, then they will let their robots fight each other in a competition, which creates a competitive factor for the students.

5. Certificates.

   The student will be awarded a certificate supported by EventMAP Ltd for participating in the competition.

6. Prizes.

   The experiment involves a programming competition and winners will be awarded prizes.

Permissions were granted from the teachers of the programming modules for year 1 students in all majors to make a presentation in the lecture to introduce the students with the competition. The interested students who

want to participate were asked to sign in the consent form (provided in Appendix D). Out of almost 500 students, 82 students showed interest and signed in for the experiment. 61 students were male and 19 were female, 2 students didn't specify their gender. Following the framework, a pre-test has been prepared to evaluate the students' programming skills and knowledge and it was given to all year 1 students as a quiz in the module lab slot were 335 students completed the pre-test. The test results can be accessed through the links provided in Appendix I. Following the framework and as shown in Part 1 in figure 4.1. The students were divided into two groups equally based on their marks in the test to ensure that each group has students on the same level. After that, students were informed, in which group they are. The students in the first experimental group who meant to work on their own were given access to a website, which was built and hosted on Queen's University website hosting service. Where students can download the software development kit they need to build their own robots and to use it for submitting the final work. Also, the students in the first experimental group were given a document, which act as a short tutorial to help them start with the game and it is provided in Appendix K.

The students in the second experimental group who meant to take tutorials to build their robots were informed of the times that a computer laboratory was booked to start developing the robots. The students took 4 lectures or tutorials to complete their work and the content of the tutorials is as follow:

1. Tutorial 1: The students were welcomed and guided through the steps to download the Robocode software development kit and install in on the machines. Then, the students were shown how to start the game and how to do simple and basic stuff, such as starting a

battle and creating new robots. The students were asked to change the initial code, which is automatically generated when a new robot is created and see the results of the changes by playing the game. Some rules were explained to the students, such as the rules of firing bullets and the energy calculation, information about the battlefield and its coordination, the movement methods and its parameters and the anatomy of the robots, which consists of three main parts, which are body, gun and radar.

2. Tutorial 2: Reaching the second tutorial, the students were already familiar with the basics of Robocode so they were introduced to the events in the game and how can the events be used and what data can be retrieved from the occurrence of a certain event. The students were introduced to the getters, in which they can retrieve information about other robots in the battlefield, such as the robot's energy, distance and X and Y axis. The students were then asked to start targeting the other robots by calculating the distance between the two robots and accordingly assign a suitable firepower to the fired bullets.

3. Tutorial 3: In the third tutorial, the students were asked to create their own robot behaviour and to draw a design. After that, the students started developing their robots based on the created design. During the development, the students were given assistance and guidance to help them in completing the objectives.

4. Tutorial 4: During the final tutorial, the students continued the work on their robot's design and at the end of the tutorial the students tried their robots against each other to help them tune their robots. Finally, the students submitted their final work using the provided website.

Reaching the final stage of the experiment, only 8 students have submitted their final work, out of which 7 were male and 1 was female. Thus, it was the reason not to proceed with the experiment, because such a small number of students will not allow to accept or reject the hypothesis and answer the research questions. Larger number is required to validate the findings.

The reason behind the small number of students has been investigated further by contacting the students. It has been found that the timing of the experiment, which was at the end of the semester is an issue, where many students dropped because of the big load of assignments and exams at that time. Many students expressed the difficulty to manage the mandatory work of assignments and tests they have with an optional experiment. Thus, they didn't continue with the experiment. Some students didn't attend the tutorials and when they were asked about the reason, different responses were attained, such as work commitments, attending lectures and illness. However, several reasons could have played a role in students' decisions, such as the serious game itself or how it was presented. Many students didn't even respond to the inquiries about why they dropped from the experiment so no final conclusion on the reasons behind dropping could be found.

Students not completing the experiment was expected, but the number of students who dropped from the experiment was very high. The main reason was the timing of the experiment, which is at the end of the semester as stated previously in section 4.5 in the Methodology chapter. However, the timing of the experiment can't be changed. Because the students must be taught the different programming concepts in the lectures, such as loops, arrays and strings before being involved in the experiment. This was the reason for limiting the experiment to two week only. Since

adding more time for the students to work will result in major conflicts with students deadlines and exams.

5.3.2 Jordan experiment

The letter from the head of EEECS school at Queen's University Belfast (Provided in Appendix C) was presented to the deans of Information Technology faculties in three universities in Jordan, which are Applied Science University, Petra University and the University of Jordan. Ethical approval was given to proceed with conducting the experiment. Thus, meetings with the coordinators of the programming courses for Year 1 students took a place in three universities. Topics, such as how to interact with students and how to announce the competition were discussed. Permissions were granted from the teachers of the programming modules for year 1 students in all majors to make a presentation in the lectures to introduce the students to the competition. Interested students who want to participate were asked to sign in the consent form.

123 out of roughly 400 students showed interest and signed in for the experiment, in which 81 were males and 42 were females. Following the framework, a pre-test has been conducted to evaluate the students' programming skills and knowledge and it was given to all available year 1 students. 174 students completed the pre-test to ensure we have a control group that matches the two experimental groups. The results of the test can be accessed through the links provided in Appendix I.

Following the framework and as shown in Part 1 in figure 4.1. Students who signed for the experiment were divided into two groups equally based on their marks in the test to ensure that each group has students on the same level. Then students were informed, in which group they are and the students in the first experimental group who meant to work on

their own were given access to a website, where they can download the software development kit they need to build their own robots and to use it for submitting the final work. Also, the students were given a document, which acts as a short tutorial to help them start with the game. The students in the second experimental group who meant to take tutorials to build their robots were informed of the times that a computer laboratory was booked to start developing the robots. The structure of the tutorials is the same as used in the UK experiment and it was discussed earlier in section 5.3.1 in the Results chapter.

43 students from the two experimental groups completed the tasks and submitted their final work. Following the framework, the students were asked to fill in a questionnaire and then they took a post-test along with 20 students who represent the control group. The small number of students who completed the experiment can be justified by the timing of the experiment, in which most of the students who drooped from the experiment referred that to the timing of experiment being at the end of the semester, in which the students had many assignments and assessments. But as highlighted earlier, the timing of the experiment can't be changed. Because students must cover different programming concepts before getting involved in the experiment. The results of the questionnaire and the post-test can be accessed through the links provided in Appendix I.

The following subsections presents the analysis for the obtained results.

5.3.2.1 Statistical analysis

ANOVA statistical test was chosen as highlighted in section 4.3.6 in the Methodology chapter. Five assumptions must me be satisfied before using ANOVA. The assumptions will be checked and make sure they are all met as shown below:

1. Data type.

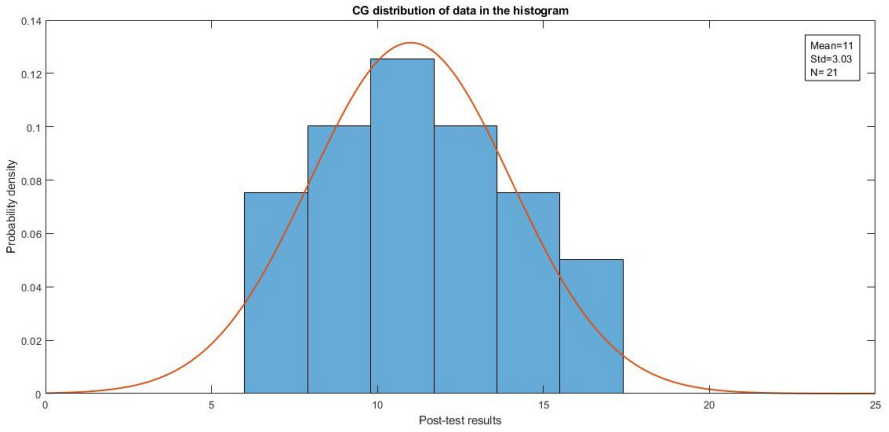
   The first assumption is data type. The dependant variable data must be interval or ratio. The dependant variable data we have is interval, which is represented by the exam scores.

2. Distribution of data.

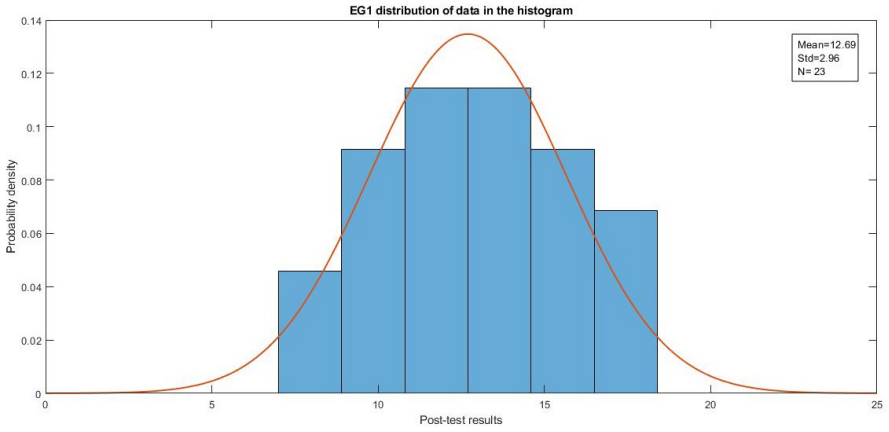
   The data must be tested to check if it is normally distributed. The normality tests can be done by a visual inspection or by a statistical test. Mazlan (2012) stated that depending only on visual examination of diagrams may lead to erroneous interpretation. Thus, three different methods using both visual and statistical tests were used to check the normality of the data, which are Histogram, Quantile-Quantile plot (QQ plot) and Shapiro-Wilk test. Figure 5.7 shows three figures, which represent the Histogram for the three different groups we have, which are the control group, the first experimental group and the second experimental group.

   It can be seen from all three figures that the histograms follow the normal distribution of the data and they does not skew to the left or the right, which means the data came from a normally distributed population. Further, QQ plot is used to test the validity of a distributional assumption for a data set (Christensen, 2011). It was applied on the data from the three groups against standard normal data and figure 5.8 shows the results of applying the QQ plot.

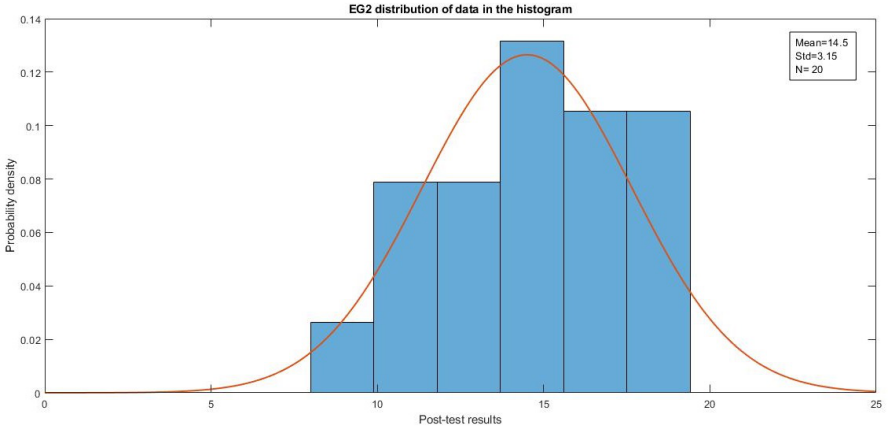
   It can be observed from the three figures that the data in the three groups are around the linear line, which indicates that the data came from a normally distributed population. The final normality test that was used is the Shapiro-Wilk test. Shapiro-Wilk test was chosen because several studies stated that Shapiro-Wilk test is the most powerful normality test, such as (Farrel and Stewart, 2006; Keskin, 2006).



(Control group)



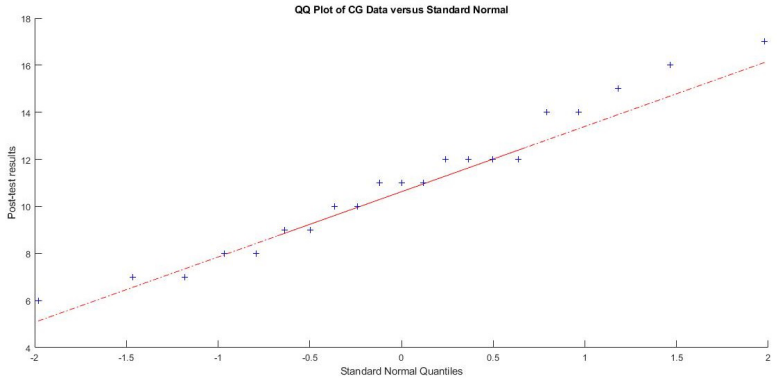
(First experimental group)



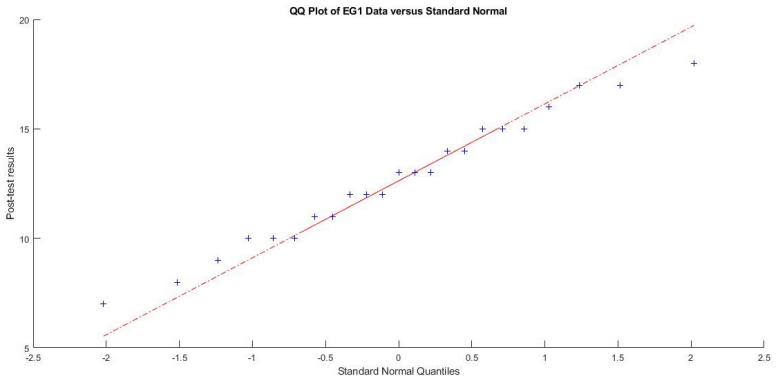
(Second experimental group)

FIGURE 5.7: Histogram showing the distribution of data for the post-test

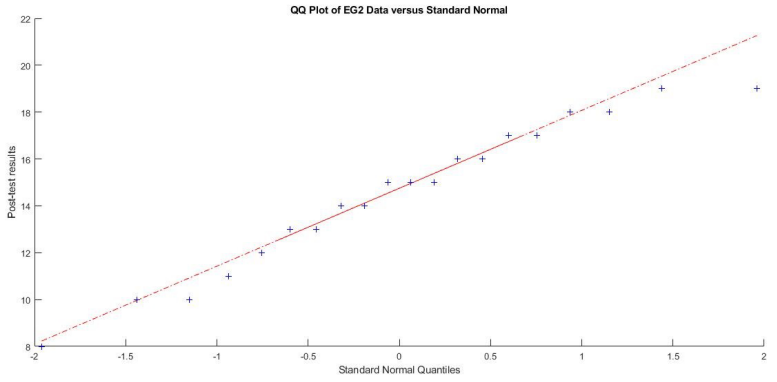
In Shapiro-Wilk test, a null hypothesis (H_0) means the population is normally distributed and the alternative hypothesis (H_1) means the population is not normally distributed. If the significant value obtained from the results is greater than 0.05 that means that the data



(Control group)



(First experimental group)



(Second experimental group)

FIGURE 5.8: QQ plot showing the distribution of data for the post-test

came from a normally distributed population and if the significant value was less than 0.05 that means the null hypothesis is rejected and the sample population is not normally distributed. Table 5.5 shows the results of applying Shapiro-Wilk test on all three groups.

As shown in table 5.5, the significance value for all three groups are greater than 0.05, which means we accept the H_0 that the data came

TABLE 5.5: Shapiro-Wilk test results.

Normality Test	Shapiro-Wilk test	
	Statistic value	Significant (P-value)
Control group	0.969	0.725
First experimental group	0.979	0.904
Second experimental group	0.960	0.552

from a normally distributed population.

3. Independent samples.

   The research design followed the between-subject design, in which different groups of people are assigned to each group. Thus, there are different participants in each group, which means the samples are independent.

4. Homogeneity.

   To check this assumption, Levene's test of homogeneity was used. The original Levene's test used only the mean. Brown and Forsythe (1974) extended the test to use the median or 10% trimmed mean. The 10% trimmed mean is the mean of the observations after removing the largest and smallest 10% values in that group. Levene's test null and alternative hypothesis are defined as:

   - H_0: The data samples have equal variances.
   - H_1: The data samples don't have equal variances.

   Levene's test was applied on the data using the mean, median and 10% trimmed mean to test the hypothesis that the groups' variances are equal. Below are the results:

   (a) Use of mean: The p-value was 0.6686.

   (b) Use of median: The p-value was 0.7258.

   (c) Use of 10% trimmed mean: The p-value was 0.6725.

All three tests failed to reject the null hypothesis at the 0.05 significance level. There is insufficient evidence to claim that the variances are not equal and, thus, this assumption for using ANOVA is satisfied.

5. Random sampling

   The samples were randomly selected and were randomly assigned to the groups. The matching in the groups was based on the marks but the process of assigning the participants to the groups was done randomly.

After all five assumptions were satisfied, ANOVA will be applied on the results of the post-test to test if there are any differences between the three groups.

Sullivan and Feinn (2012) defined effect size as "the magnitude of the difference between groups". It helps readers understand the magnitude of differences found. The P-value can inform the reader whether an effect exists while the effect size will reveal the size of the intervention effect. For illustrative purposes, effect sizes were calculated using the sample size, mean and variance of the three groups. The results showed the following:

1. The effect size for EG1 vs EG2 is f=0.2288
2. The effect size for EG1 vs CG is f=0.4343
3. The effect size for EG2 vs CG is f=0.2178
4. The overall effect size f=0.4343

Sullivan and Feinn (2012) defined statistical power as "the probability that your study will find a statistically significant difference between interventions when an actual difference does exist". If statistical power is

high, the likelihood of deciding there is an effect, when one does exist, is high. The power was calculated using the achieved overall effect size, which is 0.4343 and with the significance level set to 0.05. The result showed that the power is 0.8683 which means that the significant results are reliable due to the reduction in probability of type II error.

The one-way analysis of variance (ANOVA) is used to determine whether there are any statistically significant differences between the means of groups. ANOVA generates two hypotheses, which are:

- H_0: There are no statistically significant differences between the means of the groups.
- H_1: There are statistically significant differences between the means of the groups.

Running ANOVA showed that the effect of teaching approach on students results in the post-test was significant, $F(2,61) = 6.01$, $p = 0.004$. This means we reject the null hypothesis and accept the alternative hypothesis. Table 5.6 shows the mean and standard deviation for all three groups. Table 5.7 shows the generated table from running ANOVA and figure 5.9 shows the ANOVA boxplot.

TABLE 5.6: Groups mean and standard deviation

	Experimental group 1 (EG1)	Experimental group 2 (EG2)	Control group (CG)
Mean	12.695	14.5	11
SD	3.308	3.12	3.24

TABLE 5.7: Result of running ANOVA

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	125.4898	2	62.7449	6.019	0.004
Within Groups	635.8696	61	10.42409		
Total	761.3594	63			

ANOVA test will only show that there is a significant difference in the mean. It will not determine which group is statistically different from the

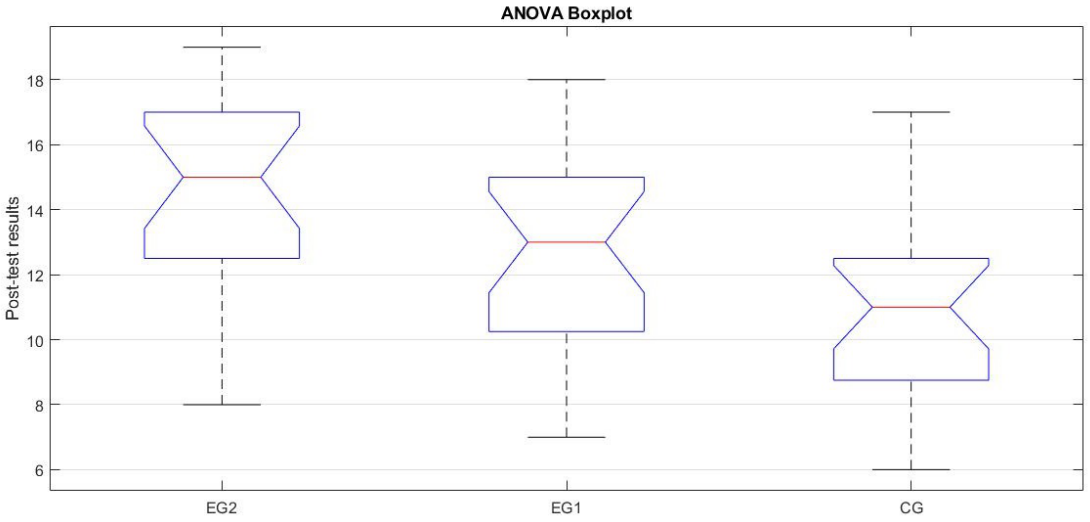


FIGURE 5.9: ANOVA boxplot.

other. In order to specify where the differences lie. There is a need to use post-hoc tests, such as Tukey honestly significant test (HSD) (Weaver et al, 2017). First, multiple comparison has been conducted using MATLAB, which showed that there is a statistically significant difference between the Control group and the second experimental group as shown in figure 5.10.

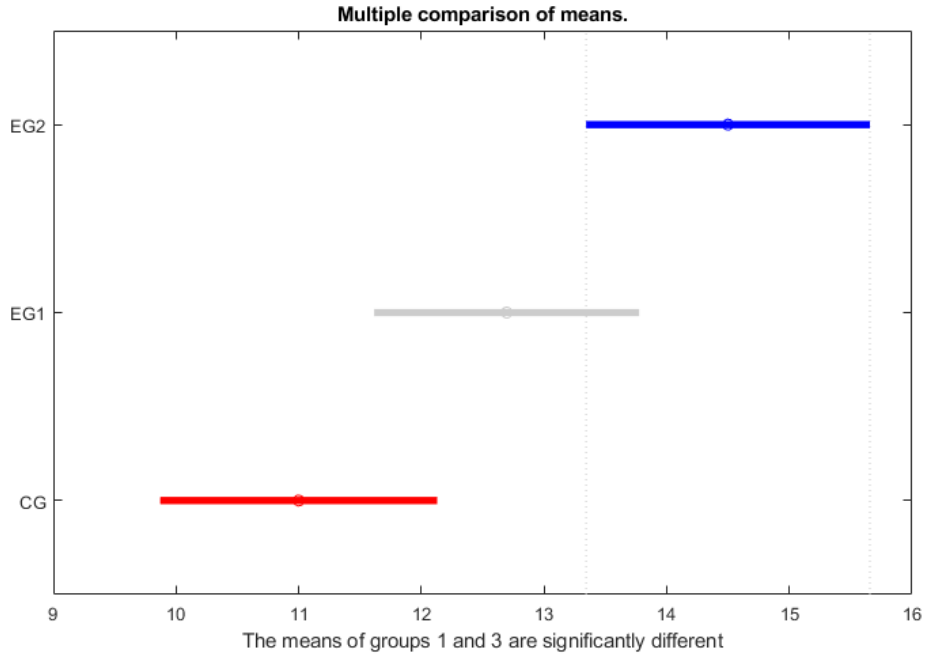


FIGURE 5.10: Multiple comparison of means.

HSD was used as a post-hoc test to investigate where the differences occurred between the three groups. HSD is designed to compare each of the conditions (groups) to every other condition (group). Thus, it will compare EG1 with EG2, EG1 with CG and EG2 with CG. HSD will run three times and each time it generates two hypotheses, which are:

- H_0: There are no statistically significant differences between group 1 and group 2.
- H_1: There are statistically significant difference between group 1 and group 2.

Running HSD showed the following:

- EG2 and CG differed significantly at $p<.05$, in which the p value was 0.002. Thus, H_0 is rejected and the alternative hypothesis H_1 is accepted.
- EG2 and EG1 were not significantly different, in which the p value was 0.16 and, thus, the null hypothesis H_0 is accepted.
- EG1 and CG were not significantly different, in which the p value was 0.19 and, thus, the null hypothesis H_0 is accepted.

The statistical test ANOVA showed that there is a statistically significant difference between the three groups. When post-hoc test was applied to investigate further. The results showed that students in the second experimental group, where students used the serious games and took the tutorials (M=14.5, SD=3.12) achieved significantly higher marks in the post-test compared to the control group, where students didn't use the serious game (M=11, SD=3.24). The post-hoc test didn't identify any significant difference between the students in second experimental group and students in the first experimental group (M=12.695, SD=3.308). Also, the

results showed that there is no significant difference between the students in the first experimental group and students in the control group.

The statistical analysis shows that the test results are significantly different for the three groups. The significant difference occurred between the control group and the group that used serious game through tutorials. The difference between the other groups are big but not significant. Considering the test results, which represents students' understanding of the covered concepts. It can be concluded that using serious games through tutorials is the best approach for using serious games in teaching computer programming.

5.3.2.2 Performance factor analysis

The performance factor analysis was based on the framework where the results of the questionnaire and the tests were combined through the formula 4.1 as shown in part 4 of figure 4.1. The questionnaire results needs to be calculated following the formula 4.2 as described in section 4.3.6 in the Methodology chapter.

The last question in the questionnaire asks the student to rate the difficulty of few programming concepts. It is the same as the last question in the teachers interviews with slight difference in the listed programming concepts. Since the scale in the question in the interviews was changed from (1 to 5) to (1-10). The same change will be applied to the question in the students questionnaire to allow for a comparison between students and teachers rating for the difficulty of the programming concepts. Thus, the formula 4.2 will be slightly changed to accommodate this change. Formula 5.1 is the new formula that will be used to calculate the questionnaire result.

$$Questionnaire = val1 * 6 + (6 - val2) * 7 + (11 - val3) * 7 \quad (5.1)$$

Only the last part of the formula has changed, in which instead of subtracting val3 by 6, its subtracted by 11 because the scale has changed and the maximum possible value is 10 not 5. First the questionnaire results were calculated as shown in table 5.8. The description for the calculation of the questionnaire formula is provided in section 4.3.6 in the Methodology chapter.

TABLE 5.8: Questionnaire result.

Group Name	Questionnaire formula	Result
Control group	3*6 + (6-3.5)*7 + (11-5.3) * 3.5	56.85
First experimental group	3.25*6 + (6-3)*7 + (11-5)*3.5	61.5
Second experimental group	3.62*6 + (6-2.41)*7 + (11-4.83)*3.5	68.44

The average results for the post-test were calculated out of 100 for the three groups and then the performance factor formula was applied, table 5.9 shows the results.

TABLE 5.9: Performance factor result.

Control group	First experimental group	Second experimental group
(α * 55) + (β * 56.85)	(α * 63.45) + (β *61.5)	(α * 72.5) + (β * 68.44)
55.92	62.47	70.47

Based on the results and the analysis of the framework, it has been shown that using Robocode in tutorials as a supportive tool for teaching computer programming has better results in terms of students' understanding and satisfaction with an overall score of 70.47%. Using Robocode as an assignment achieved an overall score of 62.47%. The lowest result was not using Robocode and only teaching computer programming by lectures and labs, which achieved an overall score of 55.92%.

The results from running the statistical analysis and running the performance factor analysis showed the same results. Both concluded that using serious games through tutorials is the best approach for using serious games in teaching computer programming. Further, the results showed that using serious games through assignments is better than not using serious games.

5.3.3 Answering the research questions

This section answers the research questions allocated to the experiments. The first and second questions are answered in two ways. First, using the statistical tests. Second, is based on the performance factor, which is the ideal way because this way takes into consideration the students' results in the post-test as well as the questionnaire results, which represents the student's satisfaction.

5.3.3.1 Does the use of serious games affect students' understanding in computer programming concepts?

After conducting the statistical tests, it has been shown that there is a significant difference between the three groups. Further, when the difference was being investigated, the results showed that using serious games through lectures or tutorials is significantly different than not using serious games at all. However, there were no significant deference between using serious games through assignments and not using serious games at all.

Further, the performance factor was applied to the data achieved from the three groups and using serious games through lectures achieved a score of 70.47% and using serious games through assignments achieved

a score of 62.47%; while not using serious games at all achieved a score of 55.92%.

As a result, and based on the statistical analysis and the performance factor analysis. We can conclude that the use of serious games in teaching computer programming affects and improves students' understanding of programming concepts.

5.3.3.2 What is the best approach for using serious games in teaching computer programming?

There was no statistically significant difference between the two groups that represents the two approaches of using serious games in teaching computer programming. Yet, the results of the performance factor analysis showed that using serious games through lectures achieved an overall score of 70.47%, where using serious games through assignments achieved a score of 62.47%. It can be concluded from running the experiments that the best approach of using serious games for teaching computer programming is through tutorials.

The analysis of the results showed that using the serious game through lectures or tutorials is better than using the serious game through assignments. However, it should be taken into consideration that using the serious game through tutorials requires extra staff contact time and extra cost in terms of using and booking labs. Moreover, this experiment has limited access for students in the tutorials group to the serious game, which may affect the study and the results could be different if students had unlimited access.

5.3.3.3 Are there any differences in using serious games in teaching computer programming between developed and developing countries?

This question can't be answered because the experiment in the UK as a case of a developed country wasn't completed due to the small number of students submitting their final work. Yet, running the experiment showed that using serious games for teaching computer programming in Jordanian universities as a case study of a developing country is applicable despite the concerns related to the computing infrastructure and cultural issues.

5.3.3.4 Does gender affect the engagement in using serious games for teaching purposes?

The gender effect will be measured by the continuation rate for the students who participated in the experiment and since the UK experiment wasn't completed. The results will be only based on the experiment that has been conducted in Jordan. The experiment in Jordan started with 123 students, in which 81 were males and 42 were females. 43 students out of the 123 completed the experiment and out of the 43 students 29 were males and 14 were females forming a continuation rate of 35.8% for male students and 33.3% for female students. Despite the low continuation rate for the students, males and females continuation rate was almost the same. We can conclude from that the gender doesn't affect the engagement in using simulation software and serious games in teaching, although several studies stated that female students doesn't engage with serious games as males do. This finding must be investigated further by conducting other experiments to generalise the achieved result. Also, conducting experiments using other serious games.

5.4 Evaluation questionnaires

The evaluation questionnaires were conducted at university and school level students in the UK. The questionnaires were conducted at Queen's University Belfast and at Ballyclare High School and Shimna Integrated College. The evaluation questionnaires aimed to evaluate several dimensions of serious games and to identify possible improvements that can be applied on the tested serious game for the goal of improving the game.

The following sub-sections describes conducting and analysing the results of the evaluation questionnaires before and after the changes to the serious game.

5.4.1 Pre-evaluation

After obtaining an ethical approval from EEECS at Queen's University Belfast. Meeting with the coordinator of IT Enterprise Project module took place discussing topics on how to engage with the students. After granting permission from the module teacher. A presentation took place at the end of a lecture presenting students with a short introduction about the research and its goals. Then, students who wanted to participate with the evaluation were presented with the evaluation framework and were given more details about the evaluation criteria and the factors.

Sixteen final year Business Information Technology students out of 42 students voluntarily participated by using and playing the case study serious game Robocode. The students were given access to download the game and they were asked to play the game. Then, the students were asked to rate 15 evaluation factors on a scale from (1-10). Every factor was represented by a statement and each quality characteristic from the

proposed framework was represented by 3 statements. Fifteen valid responses were obtained, in which 10 were males and 5 were females. All the respondents were in the age group between 18 and 23. The results of the evaluation questionnaire can be accessed through the links provided in Appendix I.

The average was calculated for each quality characteristic out of 100 by combining the results for the three evaluation factors that represent the quality characteristic as shown in table 5.10.

TABLE 5.10: University students rating of the quality characteristics.

No	Quality Characteristic	Average out of 100
1	Usability	71.3%
2	Understandability	62.2%
3	Motivation	69.5%
4	Engagement	66%
5	User experience	66.4%

The usability characteristic achieved the highest rating from the students with 71.3%, while the understandability scored the lowest with 62.2%. As shown in table 5.10, the scores indicate that understandability is an issue for using the serious game Robocode in teaching. To attain more understanding about the details of the rating. The average was calculated also for each factor and the results are presented in table 5.11.

Under the usability characteristic, the ease of use factor received the lowest rating (60.6%). It can be justified due to the interlinks between the understandability and the usability characteristic and particularly the ease of use factor. Boehm et al (1978) defined software understandability as a characteristic of software quality, which means ease of understanding software systems. This definition shows that the ease of use factor can be included as a factor for the usability and the understandability characteristics. However, it was added to the usability characteristic because Dumas and Redish (1999) defined usability as “the people who use the

TABLE 5.11: University students rating for each factor of the quality characteristics.

Criteria	Evaluation factor	Rating (score/10)
Usability	The game is a useful learning tool for computer programming.	71.3%
	The game contains errors.	82%
	The game is easy to use.	60.6%
Understandability	The game goals are clear and understood easily.	62.6%
	The game offers a set of straightforward steps to be followed.	63.3%
	The game can be played individually without the need of assistance.	60.6%
Motivation	The game is challenging.	70.6%
	Playing the game is enjoyable.	68.6%
	Playing the game sparked my curiosity.	69.3%
Engagement	The purpose of the game is appealing.	68%
	The idea and the storyline of the game is interesting.	53.3%
	The game is self-controlled by the user and allows making decisions.	76.6%
User experience	The game is competitive.	71.3%
	The game allows social interaction.	51.3%
	The game promotes the involvement of the learner.	76.6%

product can do so quickly and easily to accomplish their own tasks". Further, the book identified four points that form the definition and "users decide when a product is easy to use" was one of the four points. The other two factors of the usability characteristic achieved a high rating.

The understandability was measured by three factors, which are independence, clarity and simplicity and all the factors received almost the same rating with an average of 62.2%. The engagement characteristic received a rating of 66%. But the interest factor, which measures if the game is interesting to play attained a low rating of 53%. Similarly, the user experience characteristic received a rating of 66.4%. But the social interaction factor received the lowest rating of all the factors with a rating of 51.3%. This might be because the students only played the game for a short time and they did not compete. The students only developed simple robots for themselves, but they did not try their robots against each other, which may result in a higher social interaction and increase the interest of the students in the game.

Furthermore, a student added this comment "The main problem was that

I had no idea what was going on when I first started it. A little bit of an explanation would help". The understandability is a complex characteristic that affects other characteristics, such as engagement and user experience. Thus, these results suggest the need for more emphasis on improving the understandability characteristic of the serious games. Based on the three understandability factors used in this study, the problem is understanding the game goals and how to use the game itself. This issue for Robocode could be solved by two ways. Either by assigning tutors to explain the game goals for the students and by showing them how to use the game. Or by adding a tutorials option for the game that students can follow so they can grasp the required information that will allow them to understand the game and use it on their own.

Another evaluation was planned to validate the results attained from running the evaluation with university students. The choice was to conduct another evaluation with school students, so the evaluation can cover different students' skills and expertise. Thus, discussions with teachers and coordinators in Ballyclare High School and Shimna Integrated College in the UK took a place where topics, such as students' programming experience, students' usage of software and serious games and students' skills were discussed. After obtaining approvals from the schools' panels. Presentation was made for students about the research that is being conducted and a brief introduction about serious games in terms of its definition and usage. Students were introduced to the case serious game Robocode and students who wanted to participate were given access to play the game on a pre-prepared lab.

17 secondary school students in the age group between 14 and 18 participated, in which 12 students were males and 5 were females. The students were taking programming modules in the school using the language C#. The students played the game Robocode and then they were asked to fill

in the same evaluation questionnaire as the university students. The results of the evaluation questionnaire can be accessed through the links provided in Appendix I. The same analysis method was applied by calculating the average for each quality characteristic out of 100 and the results are listed in table 5.12.

TABLE 5.12: School students rating of the quality characteristics.

No	Quality Characteristic	Average out of 100
1	Usability	69.2%
2	Understandability	53.9%
3	Motivation	61.3%
4	Engagement	66%
5	User experience	61.7%

The usability characteristic scored the highest rating with 69.2% and understandability scored the lowest with 53.9%. The results achieved from Robocode evaluation with school students were the same attained from university students with usability scoring the highest rating (69.2%, 71.3%) respectively and understandability scoring the lowest (53.9%, 62.2%). To achieve more understanding about the details of the rating, the average was calculated for each factor and the results are presented in table 5.13.

TABLE 5.13: School students rating for each factor of the quality characteristics.

Criteria	Evaluation factor	Rating (score/10)
Usability	The game is a useful learning tool for computer programming.	70%
	The game contains errors.	77%
	The game is easy to use.	60.5%
Understandability	The game goals are clear and understood easily.	53.5%
	The game offers a set of straightforward steps to be followed.	47%
	The game can be played individually without the need of assistance.	61.1%
Motivation	The game is challenging.	59.4%
	Playing the game is enjoyable.	64.7%
	Playing the game sparked my curiosity.	60%
Engagement	The purpose of the game is appealing.	65.8%
	The idea and the storyline of the game is interesting.	59.4%
	The game is self-controlled by the user and allows making decisions.	72.9%
User experience	The game is competitive.	65.2%
	The game allows social interaction.	52.3%
	The game promotes the involvement of the learner.	67.6%

Similar to the university students rating, the ease of use factor of the usability characteristic received the lowest rating (60.5%), the other two factors of the usability characteristic achieved a high rating. The understandability characteristic received a very low rating for all three factors and the simplicity factor received the lowest rating of 47%. The rating for the other three characteristics, which are motivation, engagement and user experience was 61.3%, 66% and 61.7% respectively. Similar to the university students rating, the interest factor of the engagement characteristic and the social interaction factor of the user experience characteristic received a low rating of 59.4% and 52.3% respectively. However, the challenge factor of the motivation characteristic received a low rating of 59.4%. This can be justified for the same reason that was proposed for the university students rating, which is that the students did not have enough time to play and explore the game and try their robots against each other. This eliminates the competitive factor of the game, which possibly led to the poor rating of factors, such as challenge and interest.

It can be observed that there is an understandability issue in the game, where university students faced difficulties understanding the game and school students found it even harder to understand the game, which was reflected on the low score of the understandability characteristic.

“Great concept, possibly could do with a short starting guide/ tooltips to help when first trying it” was a comment left by a school student, which confirms the findings for the need of adding tutorials to the game or assigning tutors to help the students understanding the game and how to use it at the beginning.

Therefore, these results propose the necessity for more emphasis on improving the understandability characteristic of the serious game. Since Robocode can be used either in a class or as an assignment, it is better to

add tutorials for users, where they can follow a pre-prepared tutorial to walk them through the game and familiarise them with the needed information to start with. Thus, adding tutorials for Robocode was the choice for improving the game. Further, understandability characteristic might affect other quality characteristics, such as engagement and user experience, in which if the understandability of a serious game is improved it may affect other quality characteristics. This can be tested by running the evaluation again after applying appropriate changes to increase the understandability characteristic of the used serious game.

The following section describes the addition of the tutorials to the serious game Robocode.

5.4.2 Tutorial addition

A thorough study on Robocode has been conducted to highlight the important parts of the game that must be included in the tutorials to allow the inexperienced users to understand the game, how to use it and provide them with ideas to start with.

Creating a robot in Robocode is easy since the game automatically creates a new simple functional robot with simple implementation for few methods. However, creating and developing a good robot is tricky and hard, because it's not only about programming and writing code but about the analysis and problem-solving. The game can be very complex depending on the scope the user is looking at, where there are various variables that play a crucial role in the game and how the game itself can be played. For example, the robot is formed of three parts as shown in figure 5.11. Each part serves a different purpose and they can be linked together in terms of the movement or they can be controlled separately. The result of the movement of one of the parts controls the other. For

example, the movement of the radar, which detects other robots in the battlefield can control the movement of the gun based on the direction of another robot in the battlefield. These three parts form the fundamentals of Robocode, where the body represents the movement, gun represents the targeting and radar represents the scanning.

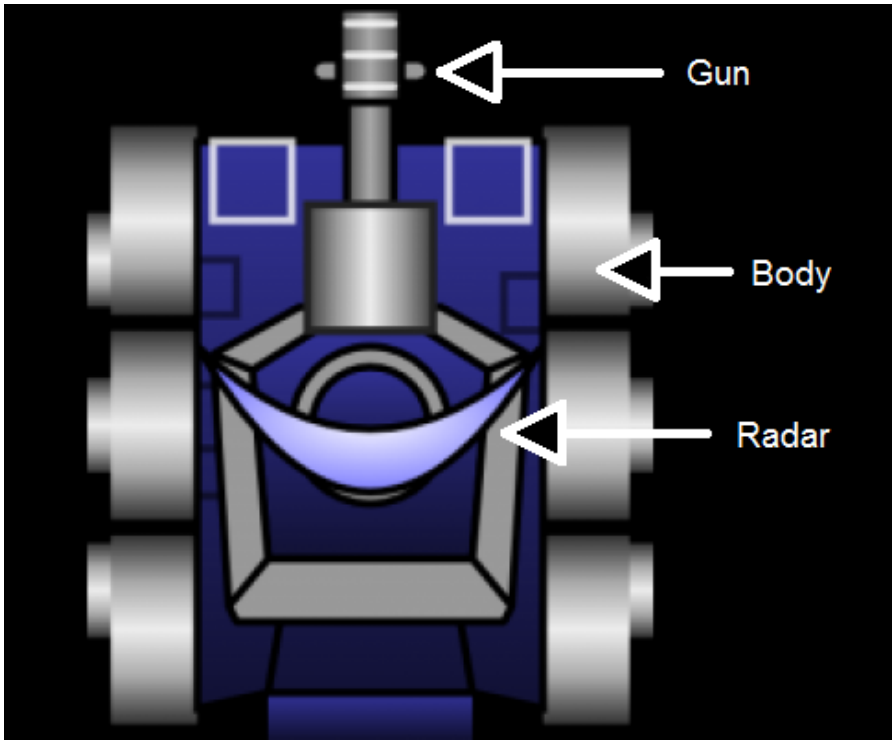


FIGURE 5.11: Robocode robot anatomy.

Robocode battlefield is designed as a coordinate graph with the bottom left of the battlefield forming the origin (0,0). The maximum height and width of the battlefield depend on the settings as there are multiple options to choose from. At any time a robot is scanned, information can be retrieved, such as the distance between the two robots, its energy, its velocity, its x and y-axis. The numbers that are retrieved can be used to do a specific action,. For example, the robot x and y-axis can be used to calculate the angle between the two robots and move the gun or the body accordingly.

Furthermore, the scoring of the game is very important in order to boost

the problem-solving skills of the users and improve their thinking. Every bullet has a firepower between 0.1 and 3. Each time a robot fires a bullet the same robot loses energy the same amount of the firepower. However, if the fired bullet hits another robot, the shooting robot gets back energy three times the firepower and the robot that has been hit loses energy four times the firepower. There is also additional damage depends on the firepower of the bullet. Also, the velocity of the bullet depends on the firepower, in which the velocity increases when the firepower decreases and vice versa. It can be seen from that the game is not about firing random bullets, but about being precise and efficient as much as possible. To be able to do that the user must be able to use all three parts of the robot effectively and make use of the information that can be gathered from scanning other robots. Based on that, a tutorial must be developed to cover multiple topics of the game starting from the simple robot movement to taking actions based on the opponent robot position.

Since Robocode is an open-source game, the code has been downloaded, which contains numerous classes and packages. However, there was no documentation for the code, which makes it hard and complicated to navigate through the code. A tutorials option has been added to the main screen of the game as shown in figure 5.12. When the option is clicked, it opens a modified source editor screen that contains two tutorial options as shown in figure 5.13. The first option for movement and contains four different tutorials, where the user is asked to complete movement tasks and it has a hints option to help the user complete each tutorial. The second option is for scanning and it contains four tutorials where the user is requested to complete different scanning and targeting assignments and includes retrieving other robots' information. Each tutorial has its own hints to help the users in completing the objectives. The tutorials were designed and developed to help the inexperienced users of Robocode to

understand the basics of the game and walk them through the main parts of the game allowing them to start building their robots. The tutorials promote some robot design ideas and help users to create their own ideas and then code them. Appendix L lists all the added classes to the game and the classes that have been changed and modified. Below is a description of the added tutorials.

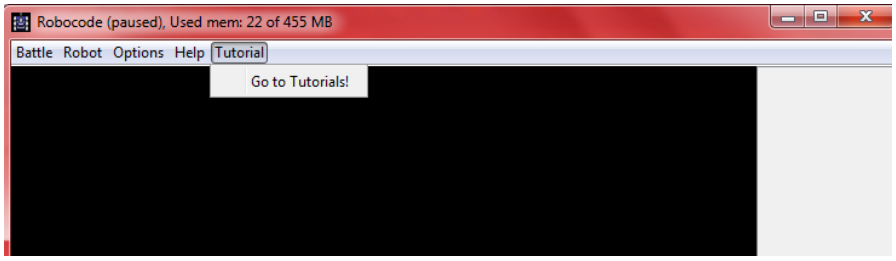


FIGURE 5.12: Robocode main screen with the tutorial option.

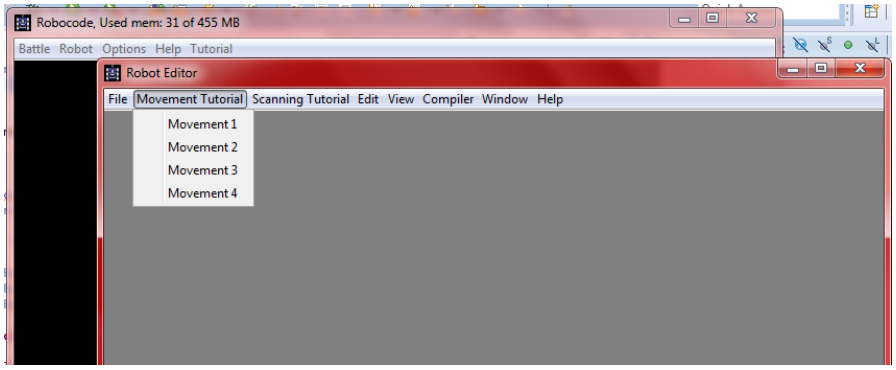


FIGURE 5.13: Robocode robot editor showing movement and scanning tutorials option.

5.4.2.1 Movement tutorials

The movement tutorials aim to teach the new users how to control their robots starting from the very basic movement. Each tutorial introduces the use of different programming concepts, such as loops and arrays. Below is a description of all four movement tutorials.

- Movement 1: the first tutorial asks the user to draw a square with the robot using only the ahead and turnRight or turnLeft methods. The class only contains an infinite loop, which is the default behaviour

of the robot. The user should write the code inside this loop. Since this is a loop and the user should draw a square one time only, the use of if statement is needed.

- Movement 2: the second movement tutorial objective is to draw a square with the robot five times. This tutorial builds on the previous tutorial, where the user should only add a for or while loop to repeat the code he/she wrote in the first tutorial.

- Movement 3: the third movement tutorial asks the user to do the same thing that has been done in the second tutorial by drawing a square with the robot five times. But asks the user to add the code in a method and then call the method. Also, the tutorial asks the user to print out how many squares have been completed and how many left each time a square is completed.

- Movement 4: the fourth tutorial asks the user to draw a square with the robot five times by adding the code in a method and then calling the method. The user is asked to declare two arrays of type double and size four. Then the user is asked to store the robot's x and y position in the two arrays each time the robot turns. The user is asked to print out the two arrays at the end of the loop.

Hints option is available for each movement tutorial that will help student to complete their objectives for each specific tutorial as shown in figure 5.14.

5.4.2.2 Scanning tutorials

The scanning tutorials aim to teach the new users how to control their robots' radar and gun from scanning other robots in the battlefield to targeting and firing bullets towards them. During the tutorials, the users

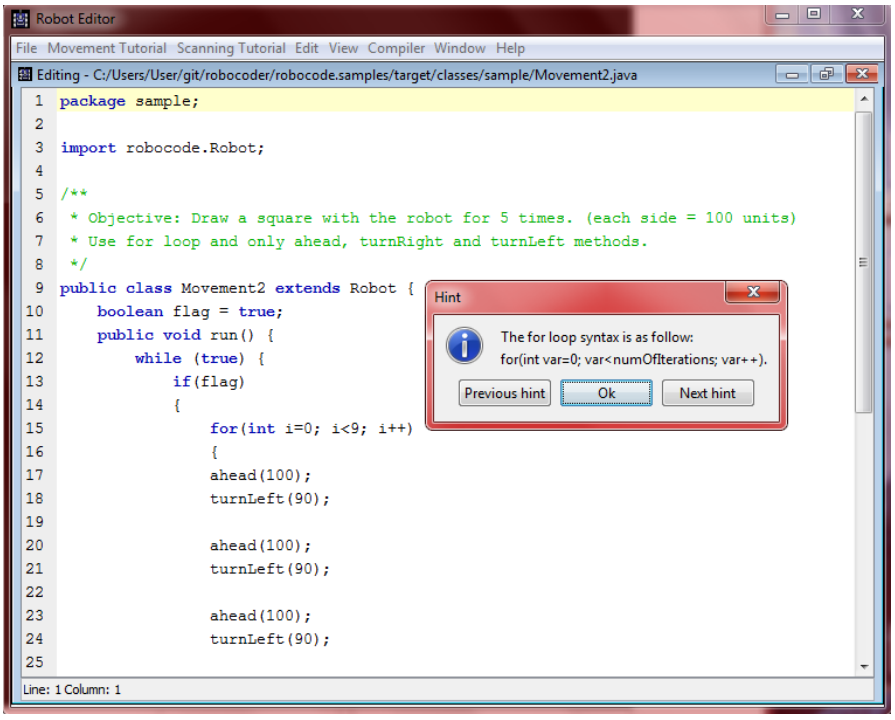


FIGURE 5.14: Hint screen to help complete the objectives for the second movement tutorial.

will be using different programming concepts like nested if statements and method calling. Below is a description of all four scanning tutorials.

- Scanning 1: the first scanning tutorial asks the user to make a simple movement then do a 180-degree turn that will turn the body and the radar of the robot. A new method is declared here, which is (OnScannedRobot) that acts as an event handler and will be called once a robot has been scanned. The user is then required to print out a statement if a robot was scanned and this event was triggered.

- Scanning 2: the second scanning tutorial builds up on the first one, where after conducting the simple movement and the turnaround to scan the battlefield. On successfully scanning another robot, the user is asked to retrieve information about the scanned robot, such as its name, energy and the distance between the two robots and then prints them out. In this tutorial, the user will start designing a

functional robot that can move, scan and retrieve information about the other robots.

- Scanning 3: the third tutorial includes targeting and firing, where it asks the user first to make a simple movement and scanning. Then it asks the user to fire a bullet with a firepower of 1 whenever it scans a robot and checks the robot's energy. The other robot in all the scanning tutorials is an inactive robot, which does not move or fire at all to help the players scan it easily and target it effectively.
- Scanning 4: the fourth tutorial highlights the importance of targeting and firing bullets efficiently, where it asks the user to make simple movement then scan for other robots. Once a robot is scanned, the user is asked to retrieve the distance between the two robots and according to the distance, the user should fire a bullet with a certain firepower. If the other robot is close fire a bullet with a high fire power and if it's far a smaller firepower.

Hints option is available for each scanning tutorial that will help student to complete their objectives for each specific tutorial as shown in figure 5.15.

After completing all eight tutorials, the users should be able to start working on their own robots since they tried the game and its controls. Also, they understand the basics of movement and scanning. Moreover, the users will understand the importance of designing a robot rather than just start coding and they received the basics that will help them to sketch their own ideas and then code and test them accordingly.

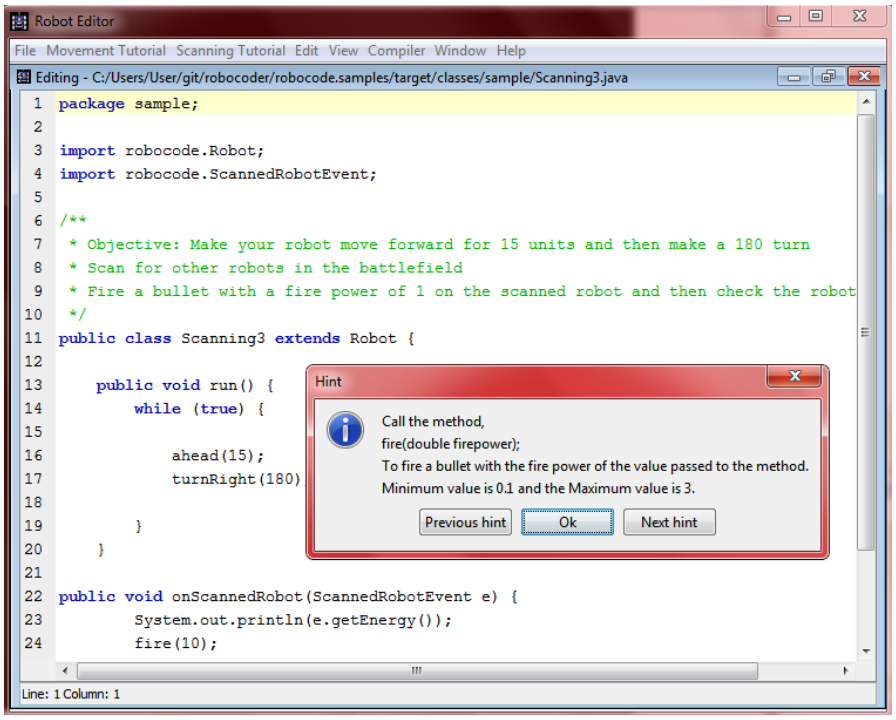


FIGURE 5.15: Hint screen to help complete the objectives for the third scanning tutorial.

5.4.3 Post evaluation

Running the evaluation framework on the case serious game Robocode highlighted understandability as the weakest quality characteristic of all tested characteristics. It has the lowest ranking among all the other characteristics by both university students with a rating of (62.2%) and by school students with a rating of (53.9%). The proposed and chosen solution was to add tutorials to the game as described in the previous section. Yet, to ensure that these changes applied to Robocode will increase the understandability of the game. Another evaluation took a place to inspect the improvements on the understandability characteristic if any and to investigate the possibility of affecting other quality characteristics.

Permission was granted from Ballyclare High School staff and students were presented with brief introduction about the game. Students who wanted to participate were given access to play the game only after insuring that the students did not participate in the first conducted evaluation

as their experience playing the game will affect their evaluation.

24 students participated, in which 19 were males and 5 were females and they were in the age group between 15 and 18. The students played the game and then filled in the same evaluation questionnaire. The results of the evaluation questionnaire can be accessed through the links provided in Appendix I. The same analysis method was applied by calculating the average for each quality characteristic out of 100 and the results are listed in table 5.14.

TABLE 5.14: School students rating of the quality characteristics after adding the tutorials.

No	Quality Characteristic	Average out of 100
1	Usability	79.7%
2	Understandability	77.7%
3	Motivation	77.6%
4	Engagement	78.3%
5	User experience	77.1%

The usability characteristic scored the highest with 79.7%, similarly to the previous evaluations with school and university students. Motivation scored the lowest with an overall rating of 77.6%, which is very close to the highest rating of the usability. To achieve more understanding about the details of the rating, the average was calculated for each factor and the results are presented in table 5.15.

The usability characteristic rating has increased significantly to achieve a rating of 79.7%. It can be observed that the easy of use factor of the usability characteristic has increased considerably for the school students rating from 60.5% to 67.9% after adding the tutorials. Further, the interest factor of the engagement characteristic has increased to achieve a rating of 67.9% after adding tutorials, in which it was 59.4% before adding the tutorials. The reason of this increase could be due to students understanding the game after completing the tutorials, which led the students to engage more with the game and increase their interest. However,

TABLE 5.15: School students rating for each factor of the quality characteristics after adding the tutorials.

Criteria	Evaluation factor	Rating (score/10)
Usability	The game is a useful learning tool for computer programming.	78.7%
	The game contains errors.	92.5%
	The game is easy to use.	67.9%
Understandability	The game goals are clear and understood easily.	79.5%
	The game offers a set of straightforward steps to be followed.	81.2%
	The game can be played individually without the need of assistance.	72.5%
Motivation	The game is challenging.	76.6%
	Playing the game is enjoyable.	77%
	Playing the game sparked my curiosity.	79.1%
Engagement	The purpose of the game is appealing.	81.2%
	The idea and the storyline of the game is interesting.	67.9%
	The game is self-controlled by the user and allows making decisions.	85.8%
User experience	The game is competitive.	72.5%
	The game allows social interaction.	53.3%
	The game promotes the involvement of the learner.	87.5%

the social interaction received a low rating as before adding the tutorials and this could be referred to the reason that the students did not compete against each other and by completing the evaluation in a brief period, the students did not have the chance to socialise between each other.

Observing the understandability characteristic, it can be seen that the overall rating has increased significantly for the school students' evaluation from 53.9% before the changes to 77.7% after the changes have been made. Adding tutorials to the case serious game Robocode had a significant effect on the understandability characteristic of the game, which can be observed from the substantial difference of the scores for the characteristic before and after adding the tutorials. Thus, it can be concluded that adding the tutorials to the game led to improve the game. Moreover, the increase of the understandability characteristic affected other quality characteristics as shown in table 5.16.

The average of all the evaluated quality characteristics has increased at least 10% after adding the tutorials. The understandability characteristic average increased the most by 23.8%, which is expected because

TABLE 5.16: School students rating of the quality characteristics before and after adding tutorials.

No	Quality Characteristic	School students average out of 100 before adding tutorials	School students average out of 100 after adding tutorials	The increase
1	Usability	69.2%	79.7%	10.5
2	Understandability	53.9%	77.7%	23.8
3	Motivation	61.3%	77.6%	16.3
4	Engagement	66%	78.3%	12.3
5	User experience	61.7%	77.1%	15.4

the addition of the tutorials targeted the understandability characteristic. Further, the motivation and user experience characteristics have increased significantly with an overall average increase of 16.3 and 15.4 respectively. The overall average increase for all the quality characteristics despite the fact that adding tutorials targeted the understandability characteristic can be justified. Because if the game and its goals are clear, the game can be played individually and the game offers steps and guidelines. This will lead to more motivation, better engagement and user experience.

5.4.3.1 Statistical analysis

To statistically compare the differences in the evaluation before and after the changes. Mann-Whitney U test was used to compare the 2 levels of the independent variable, which are students evaluation before and after the changes. Since post-evaluation was conducted only with school students and to ensure the homogeneity of groups. Only pre-evaluation with school students will be used in the test. Before using the Mann-Whitney U test. All the assumptions must be checked and make sure they are all satisfied as shown below:

1. Data type.

   The data must be ordinal, interval or ratio. The data that we are comparing is ordinal.

2. Distribution of data.

   The data will be checked for its shape only. This will determine how to interpret the results of the test. Histograms and boxplots were used to check the shapes of the data. All the figure are provided in Appendix M.

3. Sampling groups and observations.

   The research design followed the between-subject design, in which different groups of people are assigned to each group. Thus, there are different participants in each group, which means the samples are independent.

4. Random sampling

   The samples were randomly selected.

After all four assumption were satisfied. Mann-Whitney U test was used to compare the evaluation of the serious game before and after the changes. The shapes of the data distribution determines the hypotheses, thus, the shapes of the data were compared using histograms and boxplots (provided in Appendix M). Table 5.17 shows the null and the alternative hypotheses for each question. The table has a code column that reference each question. The first three letters refers to the quality characteristic and the number refers to the question number. (Use refers to Usability, Und refers to Understandability, Mot refers to Motivation, Eng refers to Engagement and Exp refers to User experience).

The results of running Mann-Whitney U test on all 15 questions of the evaluation questionnaire are provided in table 5.18. The table shows the question, the median for each question in both groups, the p-value, the U value and the code which is used to reference each question.

TABLE 5.17: Mann-Whitney U test hypotheses

Code	Null hypothesis H_0	Alternative hypothesis H_1
Use1	The distribution of scores for the two groups are equal	The mean ranks of the two groups are not equal
Use2	The distribution of scores for the two groups are equal	The mean ranks of the two groups are not equal
Use3	The distribution of scores for the two groups are equal	The mean ranks of the two groups are not equal
Und1	The distribution of scores for the two groups are equal	The mean ranks of the two groups are not equal
Und2	The distributions of the two groups are equal	The medians of the two groups are not equal
Und3	The distributions of the two groups are equal	The medians of the two groups are not equal
Mot1	The distribution of scores for the two groups are equal	The mean ranks of the two groups are not equal
Mot2	The distribution of scores for the two groups are equal	The mean ranks of the two groups are not equal
Mot3	The distribution of scores for the two groups are equal	The mean ranks of the two groups are not equal
Eng1	The distribution of scores for the two groups are equal	The mean ranks of the two groups are not equal
Eng2	The distribution of scores for the two groups are equal	The mean ranks of the two groups are not equal
Eng3	The distribution of scores for the two groups are equal	The mean ranks of the two groups are not equal
Exp1	The distribution of scores for the two groups are equal	The mean ranks of the two groups are not equal
Exp2	The distributions of the two groups are equal	The medians of the two groups are not equal
Exp3	The distribution of scores for the two groups are equal	The mean ranks of the two groups are not equal

By observing table 5.18, it can be seen that questions with code Use2, Und1, Und2, Eng1, Eng3 and Exp3 have a p-value less than 0.05. This shows that the statistical test showed a significant difference for the results between the groups for these questions. Since the shape of data distribution for the question with code Und2 for the two groups are the same and the test showed a significant difference. We accept the alternative hypothesis H_1, which says that the medians of the two groups are not equal. The significant difference indicates that the rating of question Und2 (The game offers a set of straightforward steps to be followed) was

TABLE 5.18: Mann-Whitney U test results.

Code	Question	Pre evaluation median	Post evaluation median	Mann P-value	U value
Use1	The game is a useful leaning tool for computer programming.	7	8	.13362	147
Use2	The game contains errors.	8	10	**.04444**	127.5
Use3	The game is easy to use.	5	7	.12852	146
Und1	The game goals are clear and understood easily.	6	8	**.00058**	73.5
Und2	The game offers a set of straightforward steps to be followed.	5	8	**.00006**	53
Und3	The game can be played individually without the need of assistance.	6	7.5	.1902	154
Mot1	The game is challenging.	6	8	.08544	138.5
Mot2	Playing the game is enjoyable.	7	8	.23404	158.5
Mot3	Playing the game sparked my curiosity.	6	8	.0784	137
Eng1	The purpose of the game is appealing.	7	8.5	**.04444**	127.5
Eng2	The idea and the storyline of the game is interesting.	6	7	.14986	149
Eng3	The game is self-controlled by the user and allows making decisions.	8	9	**.0271**	120
Exp1	The game is competitive.	8	7	.71138	189.5
Exp2	The game allows social interaction.	6	6	.88076	198
Exp3	The game promotes the involvement of the learner.	8	9	**.04338**	127

greater in the post evaluation (Mdn=8) than in pre evaluation (Mdn=5), U=35, p=.0006.

The shapes of data distribution for questions Use2, Und1, Eng1, Eng3 and Exp3 were different. The result of Mann-Whitney U test showed a significant difference. Thus, the alternative hypothesis H_1 is accepted and it can be interpreted as there are a significant difference in the mean ranks for the two groups for each question.

The results of running the statistical test showed that the changes applied to the serious game Robocode resulted in a significant difference rating. Although the changes were applied to improve the game's understandability characteristic. The statistical test showed that the changes affected four characteristics, which are Usability, Understandability, Engagement and User experience. The rating for 6 out of 15 factors increased significantly. These factors are Errors, Clarity, Simplicity, Purpose, Interest and Immersion. Two out of the six factors represent understandability and another two factors represent engagement, which indicates that there is a relation between these two quality characteristics. It can be concluded that since the game was understandable for the users, the users'

engagement increased alongside with their motivation and their experience playing the game.

5.4.4 Answering the research questions

This section answers the research questions allocated to the Evaluation questionnaires. The first and second questions are answered based on running the evaluation framework on the case serious game Robocode. The third question is the result of the changes applied to the serious game based on the output result from the first two questions.

5.4.4.1 What are the characteristics of the used serious game that needs to be improved?

To answer this question, a framework that evaluates primary quality characteristics was developed. After applying the developed framework on the used serious game Robocode. The results of the pre evaluation and as shown in table 5.10 and table 5.12 showed that the understandability characteristic achieved the lowest rating. Other characteristics, such as user experience achieved a low rating. However, because the understandability characteristic rating was the lowest, it is recommended to improve this characteristic as a first step.

5.4.4.2 How the used serious game can be improved?

From the analysis of the results, the understandability characteristic for the serious game Robocode can be enhanced by two ways: (I) by assigning tutors when presenting and using the game for the first time, (II) by adding tutorials to the game so new users can follow a pre-prepared

guide to help them in understanding the game goals and tools. Enhancing the understandability characteristic will improve the game and may positively affect other quality characteristics.

5.4.4.3 What are the effects on the used serious game evaluation after the changes, if any?

The addition of tutorials for the used serious game Robocode has been made to enhance the understandability characteristic of the serious game. After running the evaluation on the edited version of the game. It has been found that the understandability characteristic of the game has increased significantly as shown in table 5.14. Moreover, adding tutorials resulted in increasing the rating of all evaluated characteristics. Motivation and User experience characteristics had the biggest rating increase after the understandability. When the results of the pre and post evaluation were statistically analysed. The results showed that there was a statistically significant difference in the rating of 6 factors, which are errors, clarity, simplicity, purpose, interest and immersion.

5.5 Summary

This chapter presented the results of the data collection and analysis. The results from collecting data from the universities showed that the attrition rate is increasing in the universities in the developing country. Further, the failure rate for the computer programming module is high in the universities in the case studies of developed and developing countries. Moreover, the number of students is decreasing in computing faculties in universities in the case study of a developing country. While in the case of a developed country, there is a problem of students not graduating on time.

The interviews with teachers of programming module showed that serious games weren't used in teaching computer programming. Yet, the teachers showed willingness to use this approach in teaching. The results marked loops and arrays as the hardest topics for students in the first programming module. There were no differences in the results obtained from conducting the interviews in Jordan and the UK.

Conducting the experiments for finding the best approach of using serious games in teaching computer programming were done in universities in developed and developing case study countries. The study was conducted in the UK as a case study of a developed country. However, the experiment wasn't completed due to students dropping from the experiment. The small number of students who completed the tasks will not allow the study to produce strong arguments. Thus, no data analysis was provided. The same study was conducted in three universities in Jordan as a case study of a developing country. 64 students completed the tasks and the results showed that using the serious game through tutorials enhanced students understanding of programming concepts significantly, when compared to not using serious games at all. Further, the analysis of the performance factor, which combines students understanding with students satisfaction showed that using serious games through tutorials is better than using serious games as an assignment or not using serious games at all. Moreover, the study found that female students perception of using serious games in teaching is positive, which is reflected by the same continuation rate for males and females in the experiment that was conducted in Jordan.

A framework was developed to evaluate serious games. The framework consists of five quality characteristics, which are usability, understandability, engagement, motivation and user experience. Evaluation of the serious game Robocode was conducted with 15 university students and

17 school students. The results showed that the understandability characteristic achieved the lowest rating in both of the evaluations. In order to improve the serious game and particularly improve the understandability of the game. Tutorials were added to the serious game so the user can follow a set of pre-prepared tasks to familiarise himself/herself with the game. After the changes have been made, a post evaluation took place using the same framework with 24 school students. The results showed an improvement for all quality characteristics with the maximum increase occurs for the understandability characteristic. When the results were compared with the previous school evaluation result. A statistically significant improvement was found for 6 out of the 15 tested factors, which are errors, clarity, simplicity, purpose, interest and immersion.

Chapter 6

Conclusion

This chapter summarises the work in this study. The following sections present a summary and the outcome of the work completed in this research. The contribution of this study is highlighted and finally a section that presents potential paths for future work.

6.1 Summary of the research

The study highlighted an issue of the students high attrition rate in computing majors. Further, another issue was found represented by the high failure rate in computer programming modules. The issues were investigated further by collecting data from universities in case study countries. The results showed that the attrition rate is high in computing majors and the failure rate is high in computer programming modules.

The use of serious games was suggested by many studies as a solution for the difficulties in teaching computer programming, which could help in decreasing the attrition and failure rates. Numerous studies have used serious games for teaching computer programming. The results showed that serious games enhanced students understanding of the covered concepts and/or increased the students motivation. However, a gap was

found since no study compared the different approaches of using serious games to increase the effectiveness of the used serious game. This study compared two different approaches for using serious games in teaching, which are assignments and tutorials. The results showed that using serious games through tutorials achieves better results in terms of students understanding and satisfaction.

Further, several attempts have been made to evaluate serious games. Yet, limited studies were conducted to evaluate serious games for the aim of highlighting the games strengths and weaknesses. Moreover, there are observable flaws in the presented studies. Therefore, this study developed a framework to evaluate serious games for the purpose of highlighting the games strengths and weaknesses to improve the games. The framework evaluates the serious games usability, understandability, motivation, engagement and user experience. The framework was applied on a serious game and the results highlighted areas that once changed the serious game will be improved. Changes were applied on the serious game and the game was re-evaluated. The results of the evaluation showed a significant improvement for the evaluated characteristics of the serious game.

6.2 Contribution

The study collected data from universities in Jordan as a case study of a developing country and from a university in the UK as a case study of a developed country. The data contributed to prove that there is a problem in retaining students in computer majors. Moreover, the data showed that the failure rate in programming module is high.

Further, the study conducted interviews with teachers of computer programming only in universities in the UK and Jordan. The interviews results showed that there were no use of serious games in teaching computer programming. However, the teachers showed the willingness to use serious games in teaching if the games proved to be beneficial to the students. Moreover, the interviews contributed to highlight the most difficult programming concepts for students. The results showed that loops and array are the most difficult concepts that students take in the first programming module. There were no significant differences between the results achieved from the interviews in the two case study countries.

A main contribution of this study was identifying the best approach for using serious games in teaching computer programming. A framework was designed by combining five key factors for the success of using simulation software and serious games. The factors are learning environment, usage of space, students access, staff contacts time and cost. The framework consists of 2 experimental groups and 1 control group. The first experimental group used a serious game called Robocode to complete an assignment and they had unlimited access to the game. The second experimental group used the same serious game through lectures to complete the same assignment, but they had limited access to the game. The control group didn't use the serious game and they only attended the module lectures and labs. After completing the assignment, students in all groups filled in a questionnaire and completed a test to measure their understanding of the covered concepts and their satisfaction. The results showed that using serious games through lectures is significantly better than not using serious games at all. Although there was no statistically significant difference between the two experimental groups. The analysis of the performance factor that combines the test results and the satisfaction questionnaire showed that using serious games through tutorials is

better than using them as an assignment.

Another main contribution of this study was the presented evaluation framework. The framework was developed by analysing 18 different quality characteristics that were used to evaluate serious games. Five primary quality characteristics formed the framework, which are usability, understandability, motivation, engagement and user experience. Each characteristic consists of 3 factors and the framework represented each factor with a statement. The evaluation framework was applied on a serious game called Robocode. University and school students used the game and filled in a questionnaire to rate the different factors. The results showed that the understandability of the game achieved the lowest rating. To improve the games understandability and after analysing the comments left by the students. Tutorials were added to the game to help new users with the game. The tutorials focused on movement and scanning, which are the two main parts of Robocode. To validate the changes, another evaluation for the edited version of the game was conducted. The results showed that that the understandability characteristic improved significantly. Moreover, adding tutorials improved the other measured quality characteristics.

6.3 Recommendations and Future work

Since the study highlighted issues in the computer programming module. Improvements must be applied to the methods of teaching computer programming. Using serious games proved to be beneficial for increasing students understanding and satisfaction. This study recommends the use of serious games in computer programming modules. Although the study found that the best approach for using serious games is through tutorials. Serious games can be used as assignments to reduce the staff

contact time. Further, changes can be applied to the serious games to prevent the students from needing direct supervision for using the serious game. Moreover, the study recommends using the evaluation framework to highlight the strengths and weaknesses of serious games. This can help in identifying the best serious game to be used based on the targeted audience.

There are multiple potential future paths that can build on the findings of this study. The study recommends the following future work:

1. Apply the designed framework and conduct experiments in different developing countries to view if the same results can be achieved or there are barriers that haven't been found in this research.
2. Conduct the experiment in the United Kingdom as a case study of a developed country in a more formal way. Perhaps by letting the module teachers present the experiment or include it in coursework to obtain more numbers.
3. Apply the framework using different serious games other than Robocode and check if the same results can be achieved.
4. Investigate further if gender affects the engagement in using serious games.
5. Use the evaluation framework to evaluate other serious games for the aim of highlighting the game's strengths and weaknesses.

This page is intentionally blank.

Appendix A

Interview Questions

What is your degree	
How many years you have been teaching	
How many years you have been teaching programming modules	
Object First or Procedural first	
Have you used simulation or games in teaching programming or was there any initiatives for using them	
Do you think using a game as an assignment will motivate students and help them understand some programming concepts	
Will you use simulation and games as part of teaching in the future	
What is harder for students, solving problems or coding	

On scale from 1 to 10 (1 is easy, 10 is very hard), how hard are the following programming concepts.

Concept	Difficulty
Conditions & Expressions	
Loops	
Arrays	
Recursion	
Objects & Classes	
Encapsulation	
Inheritance	
Polymorphism	
Abstraction	

Appendix B

Ethical Approval Form

Name: Abdelbaset Abdellatif

Staff/Student Number: 40167348

Email: aabdellatif01@qub.ac.uk

Staff position/Student level: 2nd year PhD student

PI/Supervisor (if applicable): Dr. Barry McCollum

Funding Source (if applicable):

Please answer the following questions. If the answer to any of them is "yes", you *might* need ethical approval from the university before you start the project, and should contact the school's Ethics Officer (brian.murphy@qub.ac.uk) to discuss it.

	YES	NO	N/A
Does the project involve animals, or have any potential environmental consequences?		✓	
Does the project involve methods or technologies that could be of use in espionage, terrorism or military applications?		✓	
Does this project involve any external companies/organisations or other parts of the university beyond EEECS?	✓		
Does the project directly involve people (e.g. do you do interviews, surveys, experiments, recordings, test tasks, usability studies)?	✓		
Are any of the people "vulnerable" in any way – e.g. NHS patients, people with health or cognitive issues, children, prisoners?		✓	
Would all of the people be in a position to fully understand your study, and freely decide whether to take part?	✓		
Does any part of the project put people's health or physical person at risk?		✓	
Are there any elements of the project that deceive people, or could be otherwise unpleasant (e.g. frightening, embarrassing, distressing, disgusting, etc)		✓	
Will you use data which includes information that comes from individual people in your study? (termed "personal data")		✓	
Does the personal data include identifying information like names, addresses, or phone numbers? (e.g. rather than anonymizing arbitrary codes			✓

like “AHX174”)			
Is any of the personal data sensitive? That means information that people usually regard as private, like their financial details, health status, sexual orientation, etc.			✓
Is any the information as a whole “identifying”? That means, might it be possible to indirectly identify any of the people who are included in your data? (e.g. if you knew a person's home-town, age, gender and profession, you might be able to guess who it is)			✓

Outline of the project:

What do you propose to do? (please answer briefly, in a few sentences)

The research aims to investigate the effect of using simulation software and serious games in teaching programming concepts for year 1 students, taking into consideration enhancing students and understanding and students’ satisfaction. We are using a serious game called Robocode, where one group of the involved students will be asked to develop a robot, we will only provide the students in this group with the game and answer their questions. The second group will take 4 lectures, 2 hours each to build their robots. All of the students who are taking the programming course will take the Pre and Post tests, we have agreed with the teachers of the module to give the tests in the practical lectures of the module. After the completion of building the robots, students will be involved in a competition held in the university where robots will fight against each other. Students will fill in a questionnaire regards to their satisfaction on how the module have been taught for them.

Prizes for the winners are supported and sponsored by EventMap which is a spin out company of the university.

Appendix C
Head of School letter

School of Electronics, Electrical Engineering and Computer Science

Computer Science Building
14-18 Malone Road
Belfast BT9 5BN
Northern Ireland
Tel +44 (0)28 9097 4620
www.qub.ac.uk/schools/eeecs

Ref: dsn/eh

8 December 2016

TO WHOM IT MAY CONCERN

Dear Sir/Madam

Re: Letter requesting permission to conduct research: Mr Abdelbaset Jamal Abdellatif

Mr Abdelbaset Jamal Abdellatif is a registered PhD student in Electronics, Electrical Engineering and Computer Science School (EEECS) in Queen's University Belfast, working under the supervision of Dr Barry McCollum, Senior Lecturer.

His research relates to investigating and optimising the use of simulation software and serious games in teaching. The School kindly seeks your help to support him in conducting an experiment in the Information Technology faculty.

The experiment involves interacting with first year students taking the introduction to programming course, where students who want to participate in the experiment will be supported with a programming serious game called "Robocode" and will be asked to develop robots. The experiment will be carried out over two weeks after object-oriented programming concepts are covered in the module. Upon completion of developing the robots, a competition will be held between the students' robots and winners will be awarded prizes; certificates will also be awarded to all participants. Pre-Post-tests and a questionnaire will be conducted with the participants.

The identity and information gathered from students will be kept strictly confidential and will be used for the study purpose only, and you will receive a copy of the research upon completion.

Your contribution will be invaluable to this research and is very much appreciated.

If you have any further questions, please do not hesitate to contact Mr Abdelbaset Abdellatif via email at aabdellatif01@qub.ac.uk.

Thank you.

Yours faithfully

Professor Dimitrios S Nikolopoulos
Head of the School of Electronics, Electrical Engineering and Computer Science

Appendix D

Robocode Experiment Consent Form

Experiment Purpose and Procedure

The purpose of the experiment is to investigate the effect of using Robocode on enhancing students understanding of programming concepts.

The experiment consists of 3 groups:

1. Control group that will only take the Pre and Post tests and will be asked to complete a questionnaire.
2. First Experiment group that will take the Pre and Post tests, asked to complete a questionnaire and will use Robocode to build a robot in two weeks.
3. Second Experiment group that will take the Pre and Post tests, asked to complete a questionnaire and will take 4 extra lectures about Robocode to build their robots.

Students in the two experiment groups will be involved in a competition after they finish developing their robots.

Confidentiality

The following data will be recorded: student number, Email, test results, questionnaire.

All data will be coded so that your anonymity will be protected in any research papers and presentations that result from this work.

Finding out about results

If interested, you can find out the results of the study by contacting the researcher Abdelbaset Abdellatif, after dare 01-10-2017. He can be contacted via email: aabdellatif01@qub.ac.uk.

Record of Consent

Your signature below indicates that you have understood the information about the Robocode experiment and consent your participation. The participation is voluntary, and you may refuse to answer certain questions on the questionnaire and withdraw from the study an any time with no penalty. This does not waive your legal rights. You should have received a copy of the consent form for your own records, if you have further questions related to this research please contact the researcher.

Number	Student #	Student Email	Gender	Major	Signature	Date
[1]						/ /2017
[2]						/ /2017
[3]						/ /2017
[4]						/ /2017
[5]						/ /2017
[6]						/ /2017

Appendix E

The Pre-test

Email: Date: / /2017
Major:

(Circle your answer and check the box under each question.)
Assume you can create methods as you want

1. Which of the following is true about variables declared as final?

   a. They cannot be initialised
   b. They must be private
   c. Their value cannot be changed
   d. They cannot be used in a method

Yes, I know the Answer		No, I am guessing	

2. Which of the following declarations is incorrect?

   a. String s;
   b. integer i;
   c. double d;
   d. char c;

Yes, I know the Answer		No, I am guessing	

3. Will this code execute?

```
public void jump(int x){   // some instructions }
        public void test(){
        jump("left"); }
```

   a. Yes
   b. No
   c. No, there is a compile-time error
   d. Yes, but there is a run-time error

Yes, I know the Answer		No, I am guessing	

4. Will this code execute?

```
public void walk(int  x){   // some instructions  }
        public void test(){
        jump(5.23); }
```

   a. Yes
   b. No
   c. No, there is a compile-time error
   d. Yes, but there is a run-time error

Yes, I know the Answer		No, I am guessing	

5. Which of the following statements is correct?

   a. new Mobile = m1;
   b. Mobile 1m = new Mobile();
   c. Mobile m1 = Mobile();
   d. Mobile m1 = new Mobile();

Yes, I know the Answer		No, I am guessing	

6. Will this code execute?

```
public void run(String y){   // some instructions   }
        public void test(){ String str = "Hello World";
        run (str.subString(1));  }
```

a. Yes
b. No
c. No, there is a compile-time error
d. Yes, but there is a run-time error

Yes, I know the Answer		No, I am guessing	

7. Which of the following methods does not belong to the String class?

a. length()
b. substring()
c. charOf()
d. equals()

Yes, I know the Answer		No, I am guessing	

8. Given the code:

```
String s1 = "Yes";
String s2 = "Yes";
String s3 = new String (s1);
```

Which of the following is true?

(A) s1 = s2
(B) s1.equals(s2);
(C) s1.equalsIgnoreCase(s3);
(D) s1=s2=s3

a. (B)
b. (A) & (C)
c. (A) & (D)
d. (B) & (C)

Yes, I know the Answer		No, I am guessing	

9. What will be the output of the following code:

```
String s1 = "s1 = "+ "123"+"456";
String s2 = "s2 = "+ (123+456);
System.out.println(s1+" , "+ s2)
```

a. s1=123456 , s2=123456
b. s1=123456 , s2 = 579
c. s1=579 , s2=123456
d. s1= 579 , s2=579

Yes, I know the Answer		No, I am guessing	

10. Which of the following operators can be used to concatenate two or more Strings ?

a. +
b. +=
c. &
d. &&

Yes, I know the Answer		No, I am guessing	

11. What will be the output of the following code:

```
String str = “Hello World”;
System.out.println(str.substring(2,7));
```

a. llo W
b. llo Wo
c. lloW
d. llo World

Yes, I know the Answer		No, I am guessing	

12. What will be the output of the following code:

```
int x=9;
while(x>5) { System.out.print(x);
x--;  }
```

a. 9876
b. 98765
c. 987654
d. Error

Yes, I know the Answer		No, I am guessing	

13. What will be the output of the following code:

```
for(int i=10; i>0; i-=2)
        System.out.print(i+” ”);
```

a. 10 8 6 4 2 0
b. 10 8 6 4 2
c. 10 9 8 7 6 5 4 3 2
d. Error

Yes, I know the Answer		No, I am guessing	

14. What will be the output of the following code:

```
x = 5;
while(x > 1) {
        x--; }
System.out.print(x+” ”);
```

a. 5 4 3 2
b. 5 4 3 2 1
c. 1
d. Error

Yes, I know the Answer		No, I am guessing	

15. What will be the output of the following code:

```
int z = 0;
for(int i=0; i<3; i++)
        z++;
System.out.print(i);
```

a. 0123
b. 012
c. 123
d. Error

Yes, I know the Answer		No, I am guessing	

16. What is the error in the following code:

```
for(int k=0 , k<=12 , ++k)
```

a. The increment should be k++
b. The variable must be i not k
c. There should be a semicolon at the end
d. The commas should be semicolons

Yes, I know the Answer		No, I am guessing	

17. Consider the following code:

```
int[] x = {5,6,7,8,9};
System.out.println(x[3]);
```

What will be printed to the console upon execution?

a. 3
b. 7
c. 8
d. 9

Yes, I know the Answer		No, I am guessing	

18. Which of the following statements are valid in terms of an array declaration?
(A) int marks ();
(B) float marks [];
(C) double [] marks;
(D) marks int []

a. (A)
b. (B) & (C)
c. (C) & (D)
d. (B) & (D)

Yes, I know the Answer		No, I am guessing	

19. What is the default value for the elements of an array of integers?

a. 0
b. 1
c. Null
d. "0"

Yes, I know the Answer		No, I am guessing	

20. Array indexing starts with which one of the following?

a. 0
b. 1
c. Null
d. 0 or 1

Yes, I know the Answer		No, I am guessing	

Appendix F

Experiments Questionnaire

Email: Date: / /2017
Major:

(Circle your answer.)

1. What is your gender?

   - Male
   - Female
   - Prefer not to say.

2. Previous programming experience?

   - No previous programming experiences
   - Hobby
   - Self-learning
   - Previous programming course

3. Rate the following statements from 1 to 5? (1 is strongly disagree, 5 is strongly agree).

   - I enjoy programming.
   - I enjoy problem solving.
   - I have good programming skills.
   - I am interested in alternatives to lecture-based teaching styles.

4. In learning computer programming, rate the following on scale from 1 to 5? (1 is very easy, 5 is very hard).

   - Using program development environment (e.g. eclipse).
   - Learning the programming language syntax.
   - Solving a certain problem.
   - Writing code for a solved problem.
   - Finding bugs from your own work.
   - Lack of visualizations.

5. Rate the difficulty of the following programming concepts on scale from 1 to 10? (1 is very easy, 10 is very hard).

   - Expressions and Conditions.
   - Methods and Encapsulation.
   - Objects and Classes.
   - Loops.
   - Arrays.
   - String

Appendix G

The Post-test

Email: Date: / /2017

Major:

Post-Test

(Circle your answer and check the box under each question.)

Assume you can create methods as you want

1. Which of the following is true about methods declared as final?

   a. They cannot be called
   b. They cannot be overridden
   c. They cannot hold parameters
   d. They cannot return a value

Yes, I know the Answer		No, I am guessing	

2. Which of the following declarations is incorrect?

   a. float f;
   b. String s;
   c. double 5d;
   d. int i;

Yes, I know the Answer		No, I am guessing	

3. Will this code execute?

```
public void jump(int x){   // some instructions }
        public void test(){
        jump("10"); }
```

   a. Yes
   b. No
   c. No, there is a compile-time error
   d. Yes, but there is a run-time error

Yes, I know the Answer		No, I am guessing	

4. Will this code execute?

```
public void walk(double  d){   // some instructions  }
        public void test(){
        walk(5.23); }
```

   a. Yes
   b. No
   c. No, there is a compile-time error
   d. Yes, but there is a run-time error

Yes, I know the Answer		No, I am guessing	

5. Which of the following statements is correct for creating an object of type Fish?

   a. new Fish = fish;
   b. Fish 3f = new Fish();
   c. Fish f1 = new Fish();
   d. Fish f1 = Fish();

Yes, I know the Answer		No, I am guessing	

6. Will this code execute?

```
public void run(String y){   // some instructions   }
        public void test(){ String str = "Hello World";
        run (str.subString(1,5));  }
```

a. Yes
b. No
c. No, there is a compile-time error
d. Yes, but there is a run-time error

Yes, I know the Answer		No, I am guessing	

7. Which of the following methods does not belong to the String class?

a. indexAt();
b. substring();
c. toLowercase();
d. equalsIgnoreCase();

Yes, I know the Answer		No, I am guessing	

8. Given the code:

```
String s1 = "Bye";
String s2 = "Bye";
String s3 = new String (s1);
```

Which of the following is true?

(A) s1=s2=s3;
(B) s1.equals(s2);
(C) s1.equalsIgnoreCase(Bye);
(D) s1 = = s2;

a. (B)
b. (A) & (B)
c. (A) & (D)
d. (B) & (D)

Yes, I know the Answer		No, I am guessing	

9. What will be the output of the following code:

```
String s1 = "s1=" + "123"+"456";
String s2 = "s2=" + (123+456);
System.out.println(s1+" , "+ s2);
```

a. s1=123456 , s2=123456
b. s1= 579 , s2=579
c. s1=579 , s2=123456
d. s1=123456 , s2 = 579

Yes, I know the Answer		No, I am guessing	

10. Which of the following can be used to concatenate two or more Strings?

a. s1.concat(s2);
b. +=
c. &
d. s1.concatenate(s2);

Yes, I know the Answer		No, I am guessing	

11. What will be the output of the following code:

```
String str = “Hello World”;
System.out.println(str.substring(6));
```

a. Hello W
b. World
c. o World
d. Hello World

Yes, I know the Answer		No, I am guessing	

12. What will be the output of the following code:

```
int x=8;
while(x>=4) { System.out.print(x);
x--;  }
```

a. 8765
b. 876543
c. 87654
d. 88888

Yes, I know the Answer		No, I am guessing	

13. What will be the output of the following code:

```
for(int i=0; i<10; i+=2)
        System.out.print(i+” ”);
```

a. 0 1 2 3 4 5 6 7 8 9
b. 0 2 4 6 8 10
c. 2 4 6 8
d. 0 2 4 6 8

Yes, I know the Answer		No, I am guessing	

14. What will be the output of the following code:

```
int x = 2;
while(x < 7)
        x++;
System.out.print(x+” ”);
```

a. 7
b. 3 4 5 6 7 8
c. 3 4 5 6 7
d. 2 3 4 5 6 7

Yes, I know the Answer		No, I am guessing	

15. What will be the output of the following code:

```
int z = 0;
for(int i=0; i<3; i++)
        z=i;
System.out.print(z+10);
```

a. 13
b. 12
c. 3
d. 10

Yes, I know the Answer		No, I am guessing	

16. What is the error in the following code:

```
for(int k=0 ; k<=12 ; k+)
```

a. The increment should be k++
b. The variable must be i not k
c. There should be a semicolon at the end
d. The semicolons should be commas

Yes, I know the Answer		No, I am guessing	

17. Consider the following code:

```
int[] x = {50,60,70,80,90,100};
System.out.println(x[2+2]);
```

What will be printed to the console upon execution?

a. 60
b. 7070
c. 90
d. 80

Yes, I know the Answer		No, I am guessing	

18. Which of the following statements are valid in terms of an array declaration?
(A) int marks ();
(B) double marks [];
(C) int [] marks;
(D) marks int []

a. (A)
b. (B) & (C)
c. (C) & (D)
d. (B) & (D)

Yes, I know the Answer		No, I am guessing	

19. What is the default value for the elements of an array of doubles?

a. 0
b. 0.0
c. Null
d. 1.0

Yes, I know the Answer		No, I am guessing	

20. Array indexing starts with which one of the following?

a. 0
b. 1
c. Null
d. 0 or 1

Yes, I know the Answer		No, I am guessing	

Appendix H

The Evaluation Questionnaire

Email: Date: / /2017

Circle your answer and fill in Question 3 and 4

1. I accept the data to be used anonymously for research purposes only?

   - Yes
   - No

2. What is your gender?

   - Male
   - Female
   - Prefer not to say.

3. How old are you?

   -

4. Rate Robocode according to the following criteria on the scale from 1-10? (1 is strongly disagree, 10 is strongly agree)

Criteria	Evaluation factor	Rating (score/10)
Usability	The game is a useful learning tool for computer programming.	
	The game does not contain errors.	
	The game is easy to use.	
Understandability	The game goals are clear and understood easily.	
	The game offers a set of straightforward steps to be followed.	
	The game can be played individually without the need of assistance.	
Motivation	The game is challenging.	
	Playing the game is enjoyable.	
	Playing the game sparked my curiosity.	
Engagement	The purpose of the game is appealing.	
	The idea and the storyline of the game is interesting.	
	The game is self-controlled by the user and allows making decisions.	
User experience	The game is competitive.	
	The game allows social interaction.	
	The game promotes the involvement of the learner.	

Appendix I

Data Links

Interviews results link:

https://data.mendeley.com/datasets/42tkbjjw9c/1

UK Pre-test results link:

https://data.mendeley.com/datasets/2w76mhptgf/1

Jordan Pre-test results link:

https://data.mendeley.com/datasets/tydgzv2wxf/1

Jordan Post-test results link:

https://data.mendeley.com/datasets/xpwphsy59h/1

Jordan questionnaire results link:

https://data.mendeley.com/datasets/bxzf44ygdz/1

Pre evaluation with university students results link:

https://data.mendeley.com/datasets/52fz76c5zy/1

Pre evaluation with school students results link:

https://data.mendeley.com/datasets/vchx43gm8j/1

Post evaluation with school students results link:

https://data.mendeley.com/datasets/wndp94j248/1

Appendix J

Meeting transcript

Project title: Serious Games in Teaching Computer Programming: Usage and Evaluation.

Date: 17/10/2016

Note-taker: Abdelbaset Abdellatif.

Present: Dr. Barry McCollum, Dr. Paul McMullan, Ms. Angela Allen, Abdelbaset Abdellatif.

Agenda

1. What is the experiment
2. How and when to announce the experiment.
3. How to motivate students to participate.

1. What is the experiment.

Abdelbaset described the framework of the experiment and presented a detailed plan of the data collection methods.

2. How and when to announce with experiment.

Dr. Barry focused on the importance of announcing the experiment as a competition to gain more students interest. Angela stated that it is better to announce the experiment in the lab. This will make an opportunity for students to take the pre-test and sign in the experiment.

Abdelbaset highlighted that the time of the experiment will be towards the end of the semester as students must cover several chosen programming concepts before they can be involved in the experiment. Angela suggested that week 8 or 9 is suitable and the chosen programming concepts will be covered at that time.

3. How to motivate students to participate.

Angela expressed the need to highlight the game for the students and that they will be developing a game which will encourage them. Further, Dr. Barry stated that the presentation with the students should focus on the visualisation offered by the game which can't be find by using the traditional methods of teaching.

Dr. Paul suggested certificates and prizes for the winners of the competition. Angela suggested that the prize to be relate to Information Technology. Angela preferred the prize to be Raspberry Pi 4. Dr. Barry acknowledged Angela's suggestion and stated that he will provide the certificates and the prizes.

Appendix K

Robocode Tutorial

Build the best, Destroy the rest

- ❖ It's a game to help students **learn** and **apply** their knowledge in Java in a fun and a competitive way.

- ➢ **Download the game and start a battle**
  Battle → New → (select some robots) →Start Battle
  - ❖ You can see unique design and behavior for each robot.

- ➢ **Robot Anatomy**
  - ❖ **Body**
  - ❖ **Gun**
  - ❖ **Radar**

To view Radar arcs:
Options → Preferences → View Options → **tick** Visible Scan Arcs

- ➢ **Battlefield has coordinates**
  - ❖ **All coordinates are positive.**
  - ❖ **Coordinates represented as (x,y).**
  - ❖ **The origin (0,0) is at the bottom left.**

- ➢ **Movements**
  - ❖ **ahead and back methods takes distance value as parameter (distance measured by pixels) (accepts positive and negative integers)**
  - ❖ **turn gun right and left takes degrees value as parameter (accepts positive and negative integers)**
  - ❖ **All of the methods are called and controlled in the infinite loop (while(true))**

- ➢ **Events**
  - ❖ **onScannedRobot(ScannedRobotEvent e) → when your robot scan another robot**
  - ❖ **onHitByBullet() → when your robot got hit by a bullet**
  - ❖ **onHitWall() → when your robot hits a wall**

- ➢ **Getters**
  - ❖ **getEnergy() → return your current energy**
  - ❖ **getX() → return your current X position (getY() for your Y position)**

- **getHeading() (Where its facing)**
- **getVelocity() (speed)**
- **You can use getters to get information about other tanks by using the object name before calling the method (e) getDistance(), getEnergy(). (This methods are used in onScannedRobot(ScannedRobotEvent e))**

- **Bullets**
  - **Fire method takes one double parameter which is firepower (minimum value 0.1 – maximum value 3)**
  - **Bullet damage = 4 * firepower**
  - **Additional damage if firepower is greater than 1. Damage = 2*(firepower - 1)**
  - **Velocity (speed of the bullet) = 20 – 3 * firepower**
  - **Energy lost in firing = firepower**
  - **Power returned on successful hit of another robot = 3 * firepower**

- **Create a new Robot**
  - **Robot → Source Editor**
  - **File → New → Robot**
  - **Name the robot then name the package**
  - **Before using the robot you must save the class and compile it.**

Appendix L

Robocode Editing

Location: Robocode.ui > src/main/java > net.sf.robocode.ui

Class name: WindowManager - IWindowManager

Added methods to match the changes and opens the Tutorial editor window.

Location: Robocode.ui > src/main/java > net.sf.robocode.ui.dialog

Class name: MenuBar - NewBattleDialog

Added Tutorial Menu to the main Robocode screen menu to access the developed and added tutorials, Added methods to NewBattleDialog class to load and setup the battle for one robot.

Location: Robocode.ui.editor > src/main/java > net.sf.robocode.ui.editor

Class name: Module

Registered new classes as Robocode uses Pico Container for code injection.

New Classes: TutorialEditWindow - TutorialFindReplaceDialog - TutorialMoreWindowDialog - TutorialRobocodeCompiler - TutorialRobocodeCompilerFactory - TutorialRobocodeEditor - TutorialRobocodeEditorMenuBar - TutorialWindowMenuItem

New classes to setup a new editor with a new menu to show the tutorials for the user where they can access several tutorials, write the code, compile it and then run. During this process, the users have access to a hints menu which is unique for each tutorial to help them complete the different objectives.

Location: Robocode.battle > src/main/java > net.sf.robocode.battle

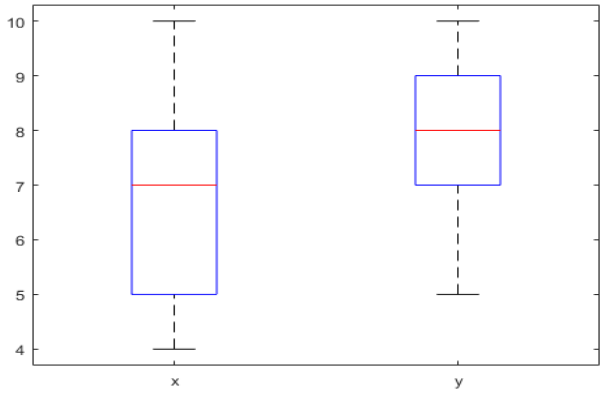
Class name: BaseBattle

Changed variables values to make the battle starts with only one robot in the battlefield.

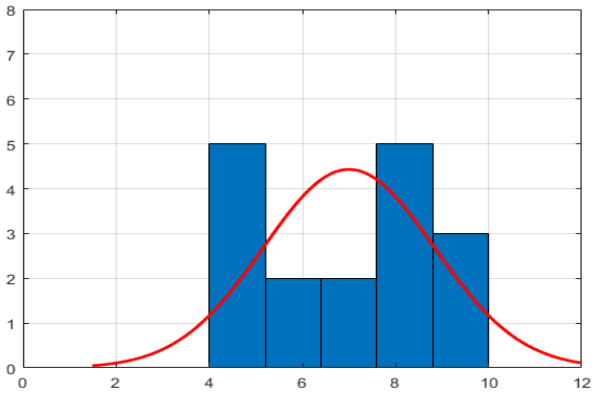
Appendix M

Histograms and Boxplots of the Pre and Post evaluation questionnaire

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

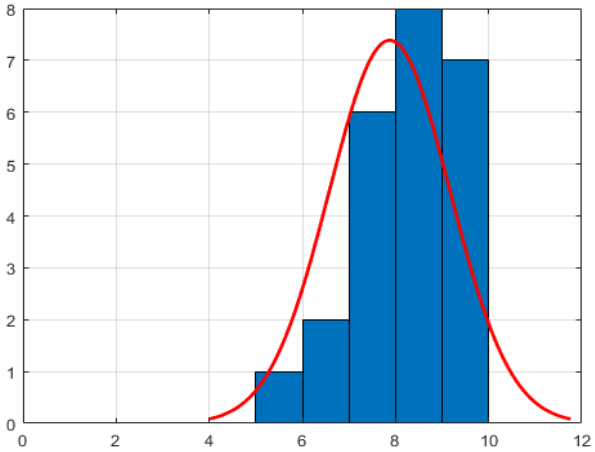
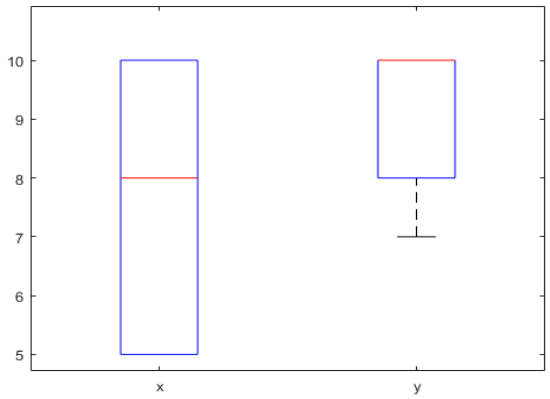
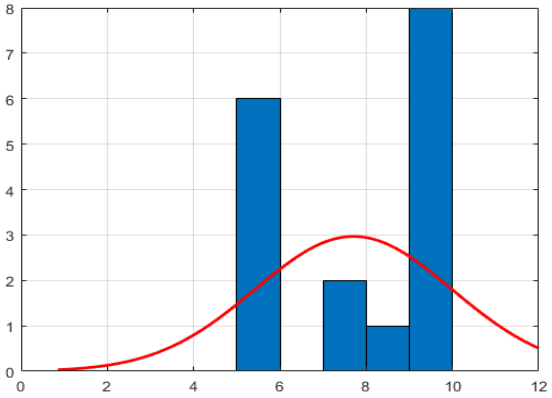


Figure 1: Histogram and boxplot for Use1

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

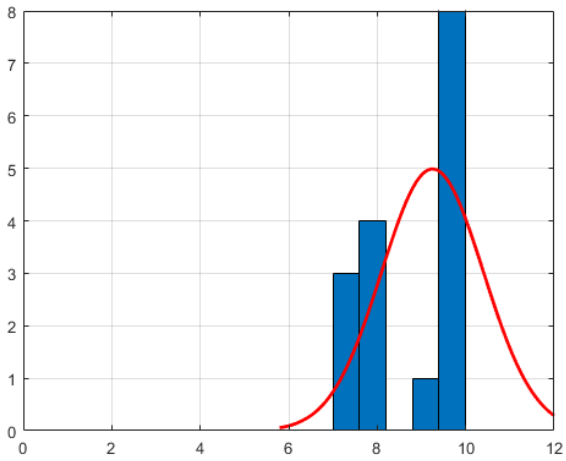
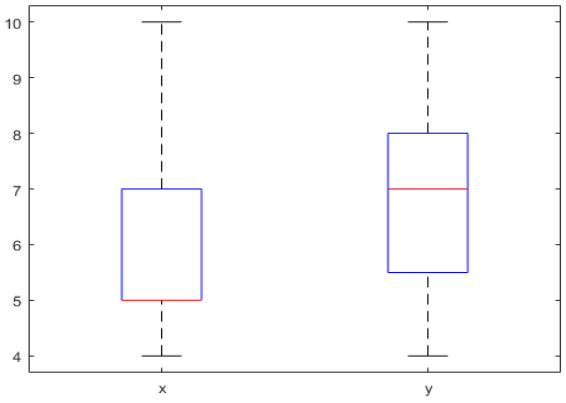
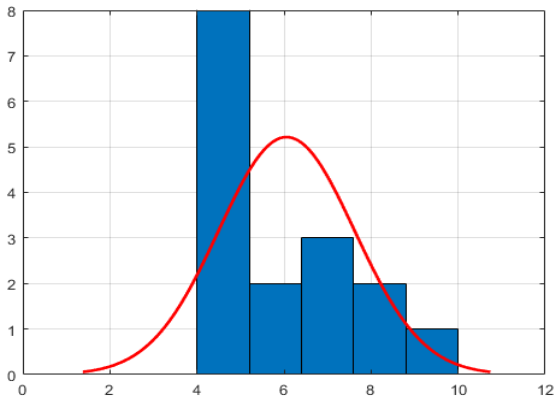


Figure 2: Histogram and boxplot for Use2

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

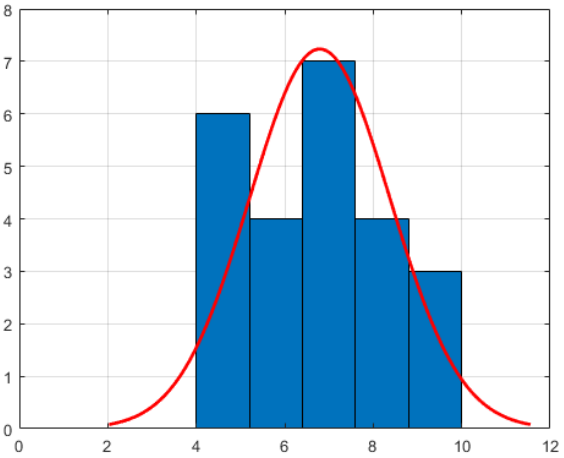
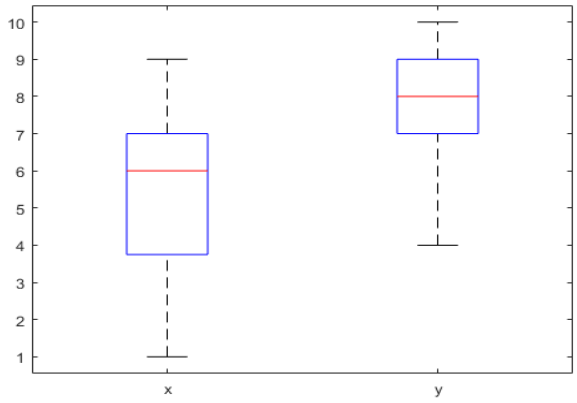
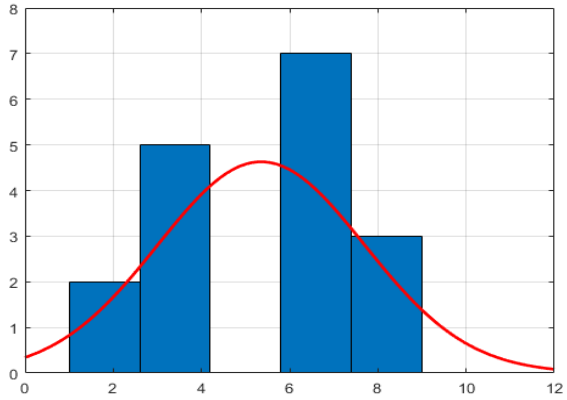


Figure 3: Histogram and boxplot for Use3

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

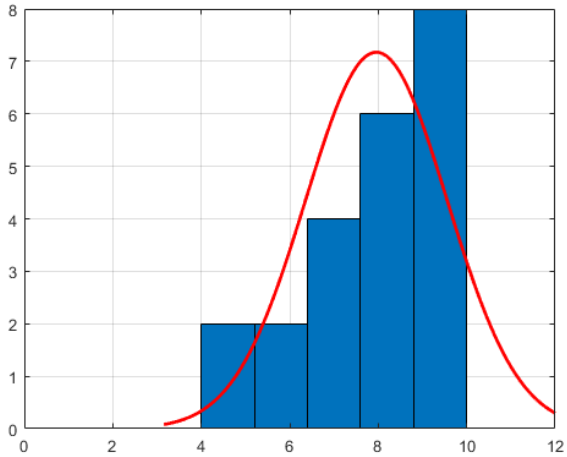
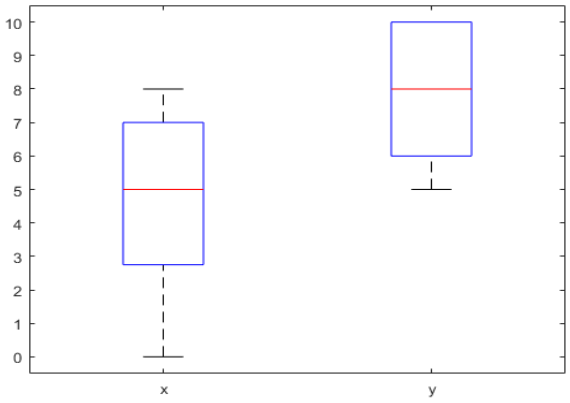
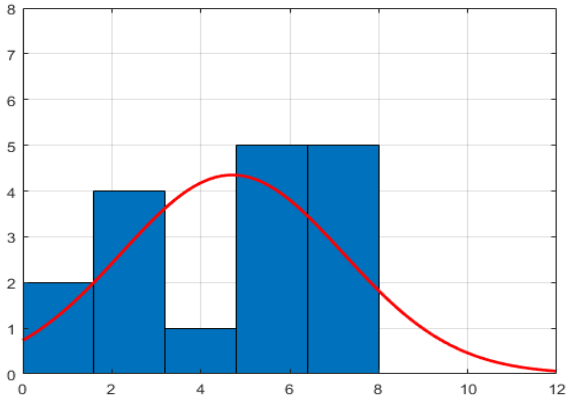


Figure 4: Histogram and boxplot for Und1

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

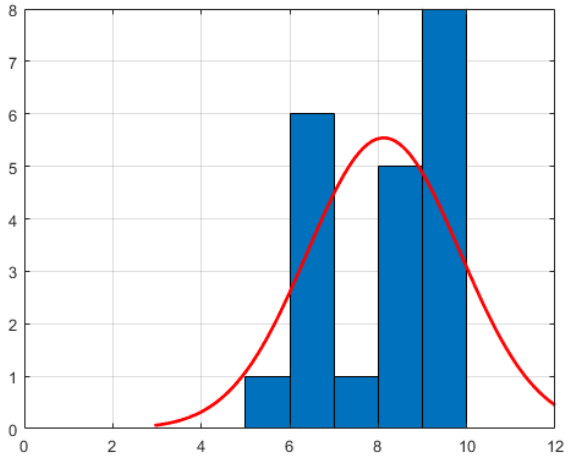
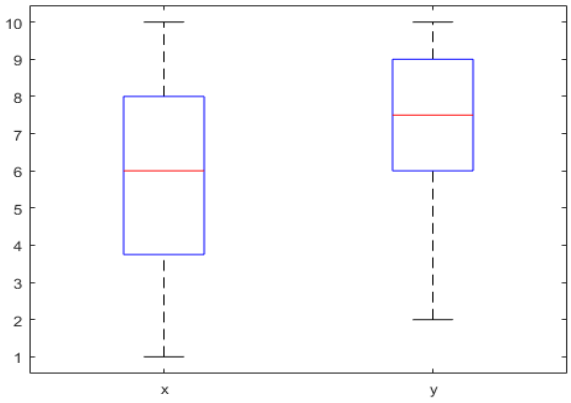
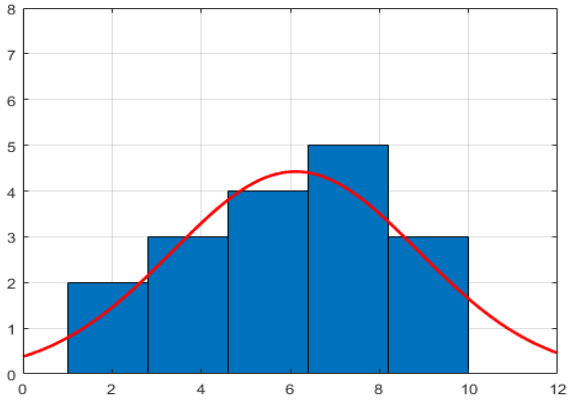


Figure 5: Histogram and boxplot for Und2

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

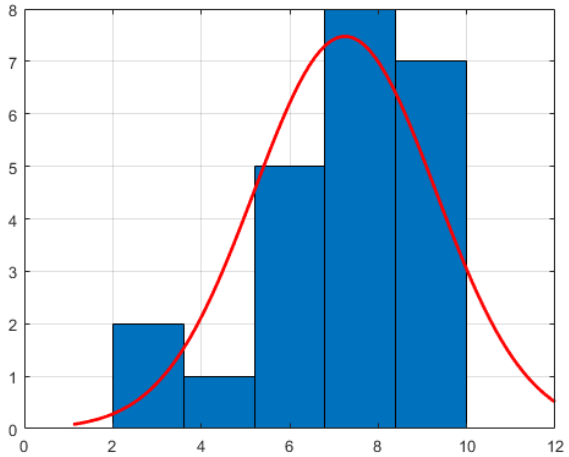
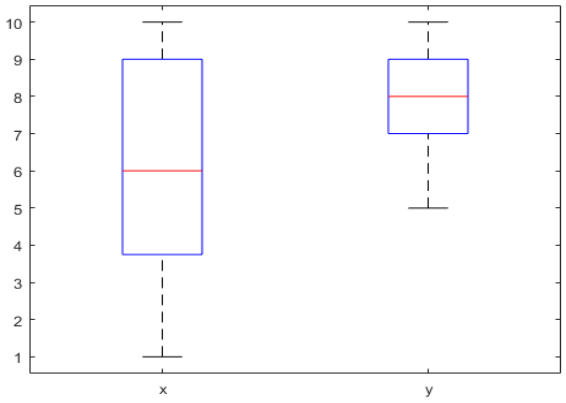
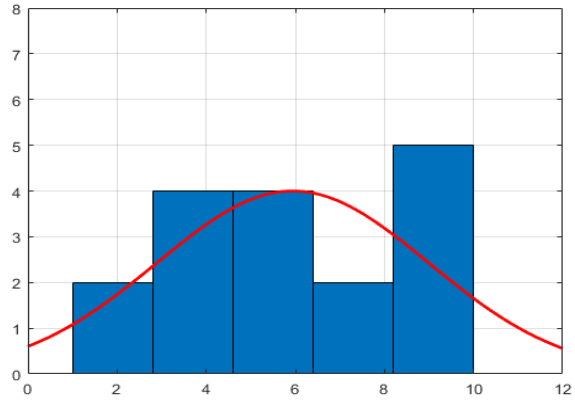


Figure 6: Histogram and boxplot for Und3

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

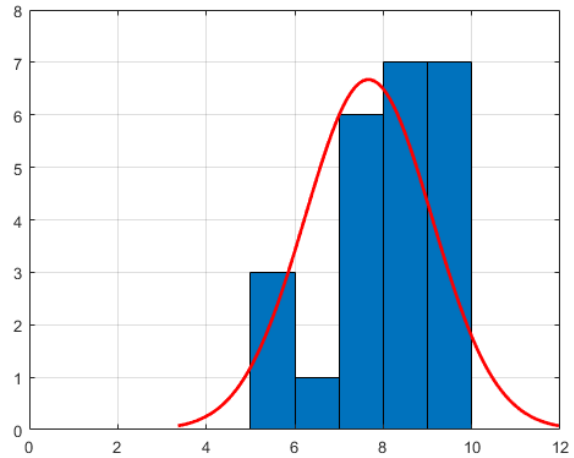
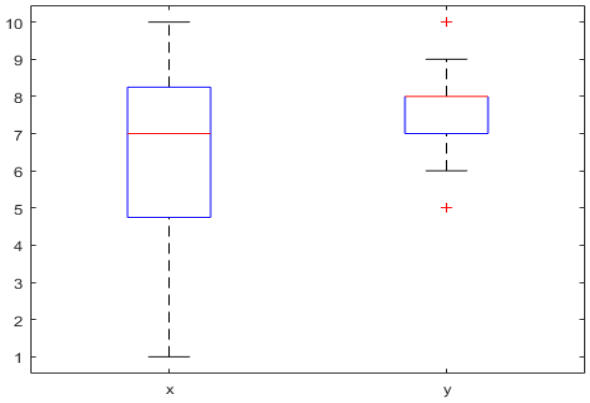
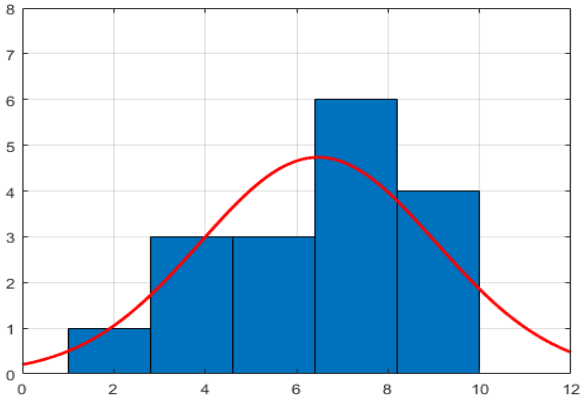


Figure 3: Histogram and boxplot for Mot1

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

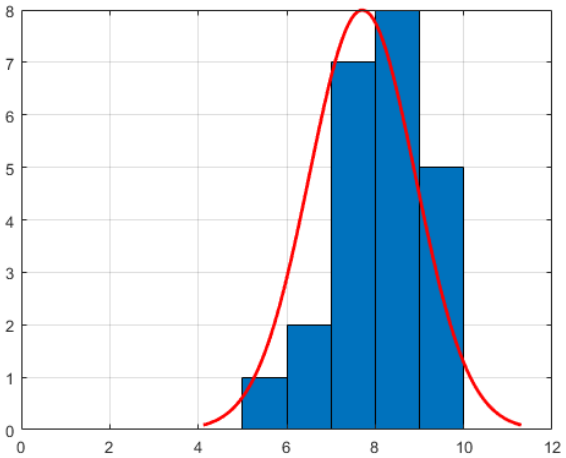
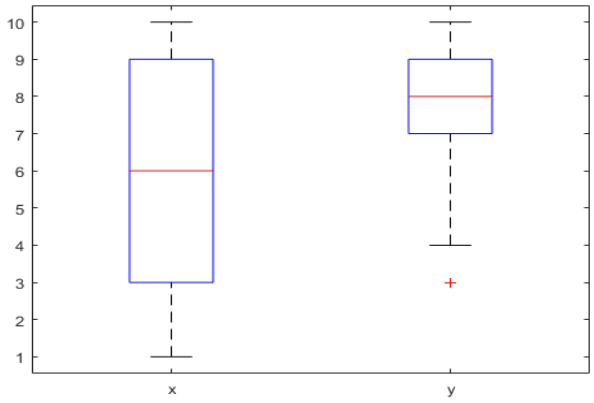
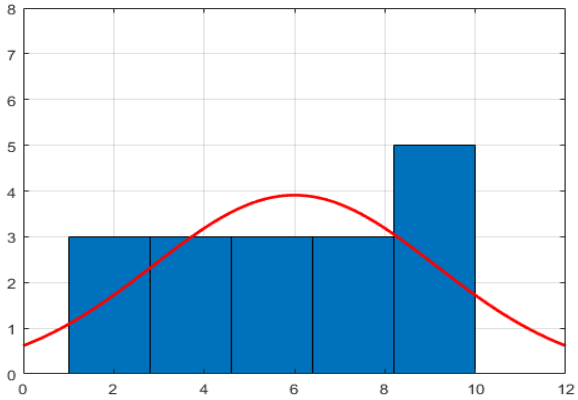


Figure 3: Histogram and boxplot for Mot2

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

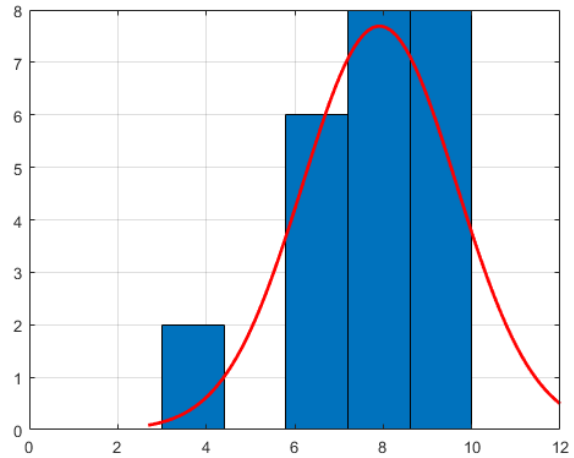
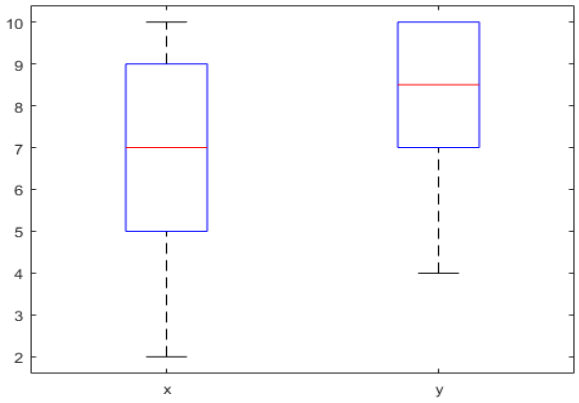
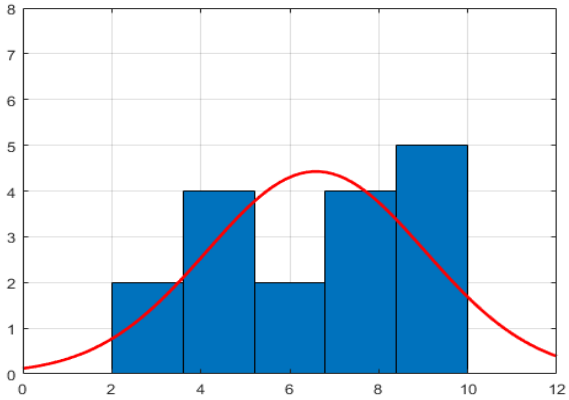


Figure 9: Histogram and boxplot for Mot3

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

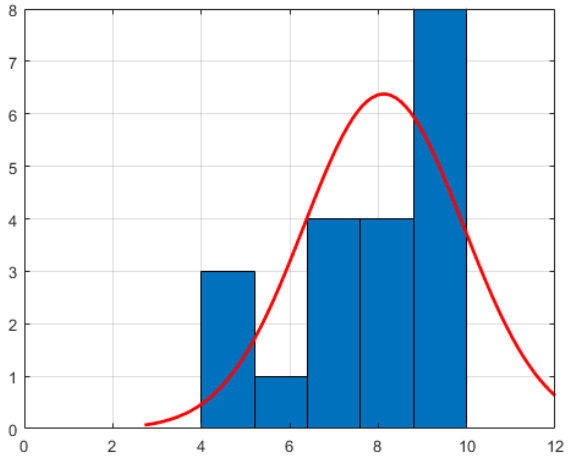
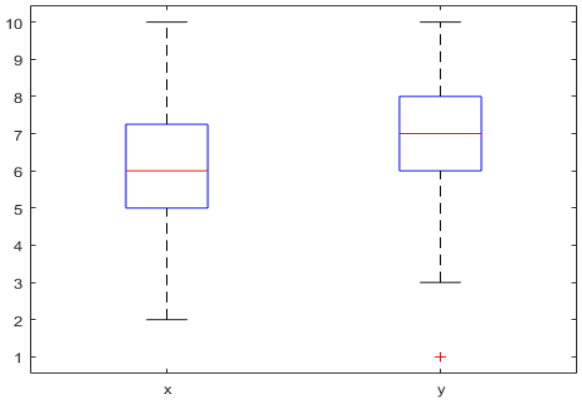
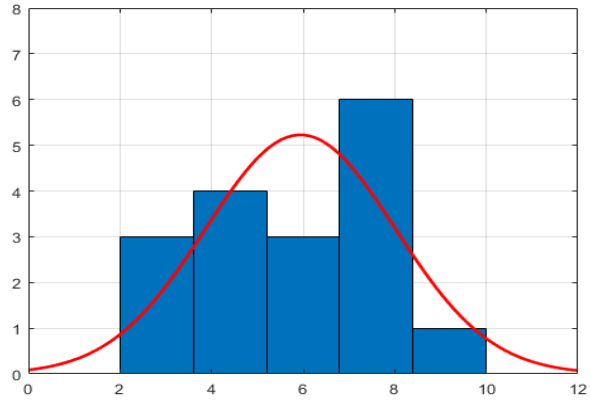


Figure 10: Histogram and boxplot for Eng1

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

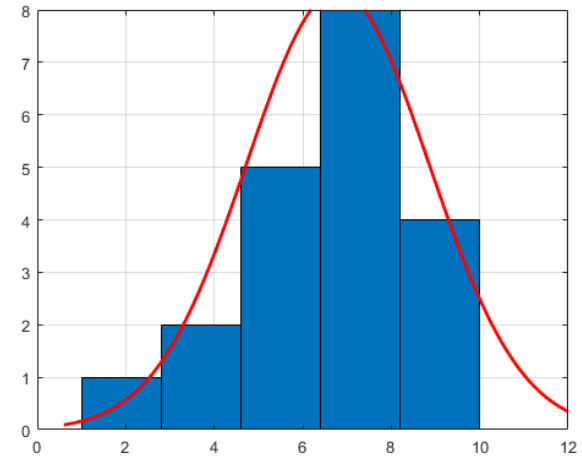
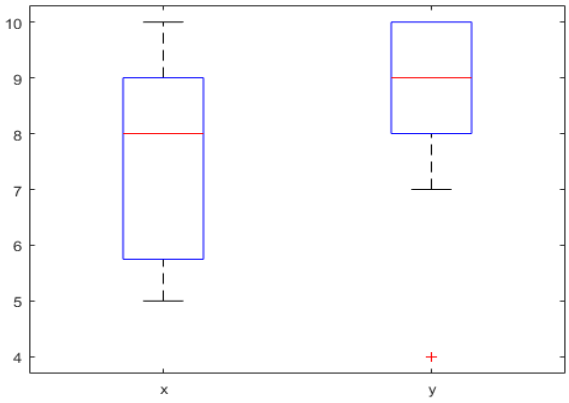
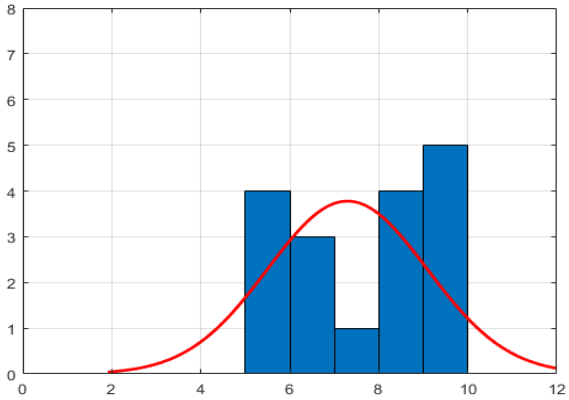


Figure 11: Histogram and boxplot for Eng2

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

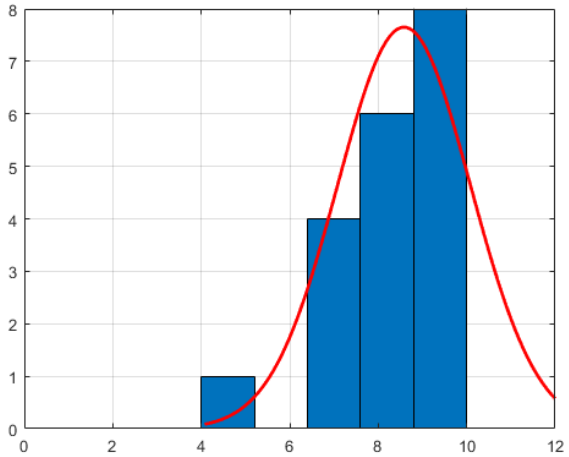
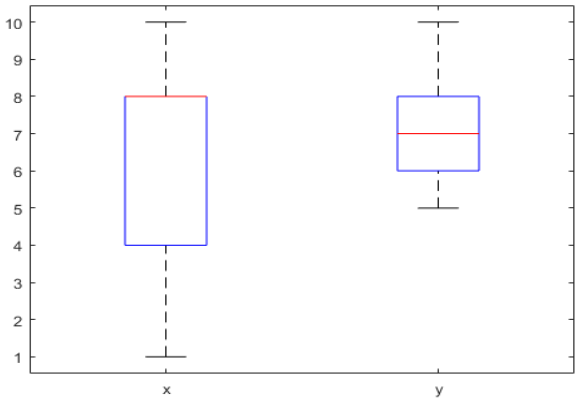
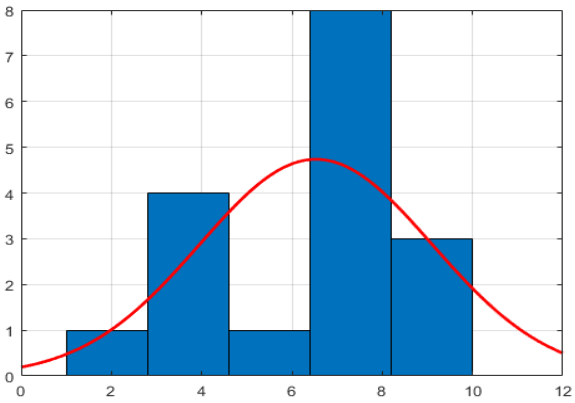


Figure 12: Histogram and boxplot for Eng3

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

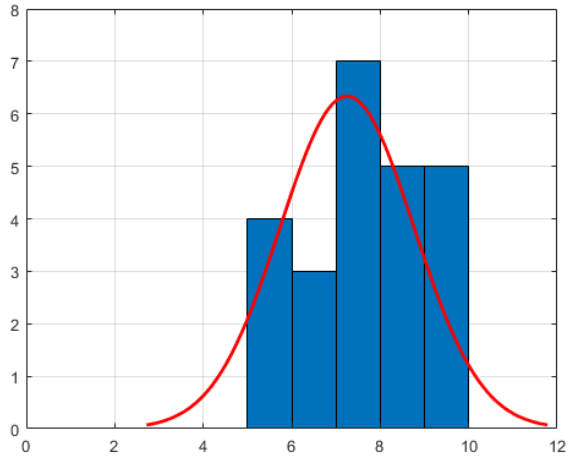
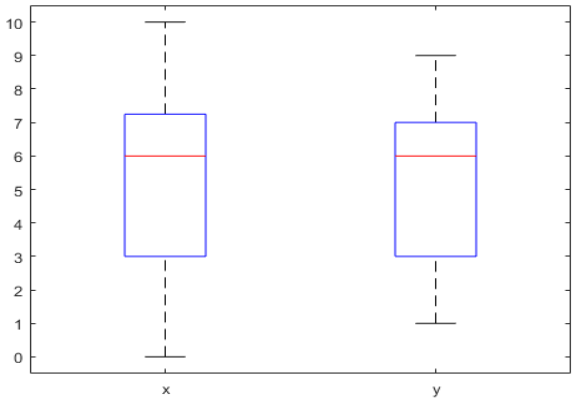
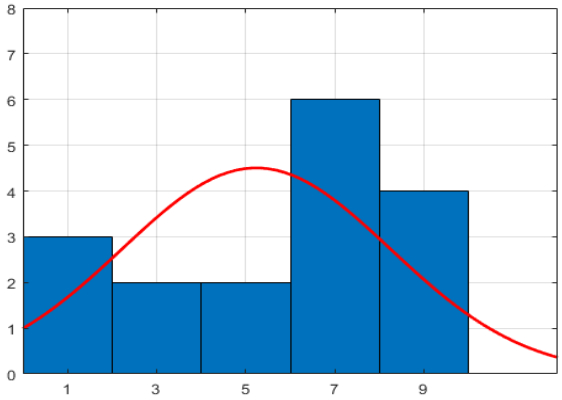


Figure 13: Histogram and boxplot for Exp1

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

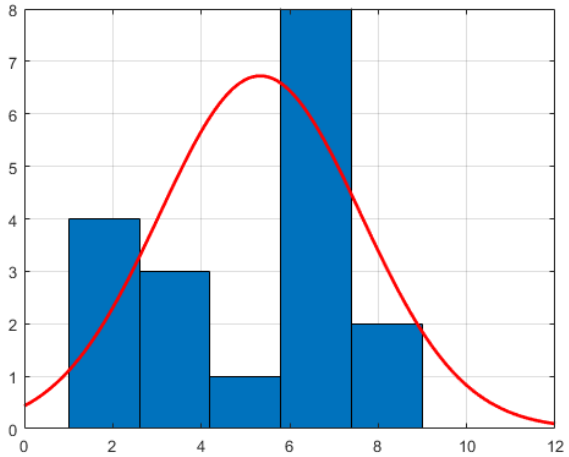
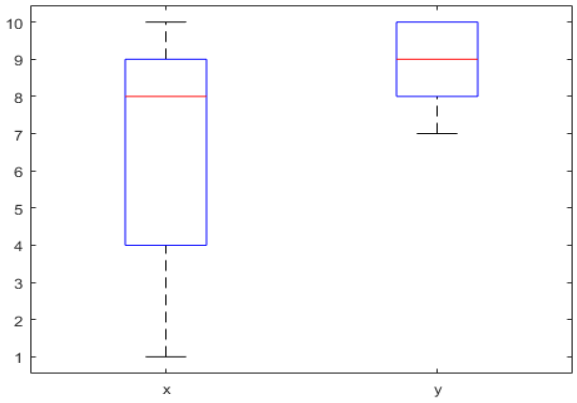
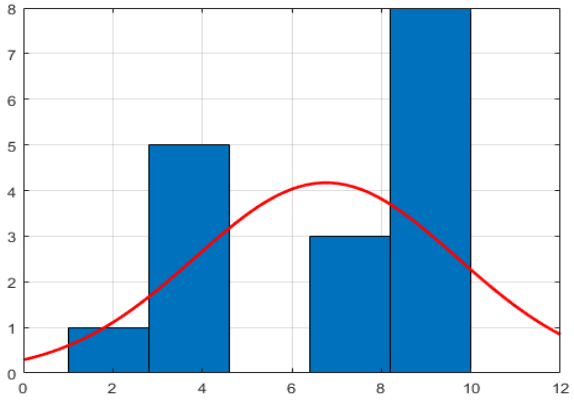


Figure 14: Histogram and boxplot for Exp2

A: Boxplot



B: Histogram for Pre evaluation



C: Histogram for Post evaluation

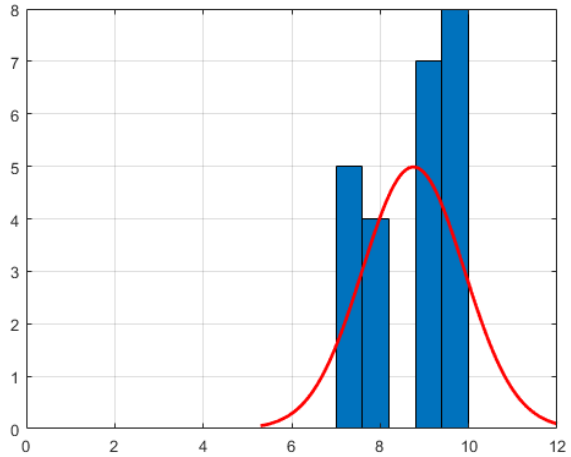


Figure 15: Histogram and boxplot for Exp3

This page is intentionally blank.

Bibliography

Abt, Clart., (1970). “Serious Games”. *Lanham, MD: University Press of America 1987. (Originally published: New York: Viking Pres 1970).*

Adamo-Villani, N., Haley-Hermiz, T. and Cutler, R., (2013). “Using a serious game approach to teach 'operator precedence' to introductory programming students”. *In Proceedings of the 17th International Conference on Information Visualisation, pp. 523-526.*

Agalbato, F. and Loiacono, D., (2018). "ROBO3: a puzzle game to learn coding". *IEEE Games, entertainment, Media Conference (GEM), pp. 359- 366.*

Age of Learning, Inc, (2018). “Full Online Curriculum FOR CHILDREN AGES 2-8.” *ABCmouse.com, 2018, www.abcmouse.com/abt/homepage.*

Ak, O., (2012). "A game scale to evaluate educational computer games". *Procedia Social and Behavioral Sciences, Vol. 46, pp. 2477-2481.*

Aktunc, O., (2013). “A Teaching Methodology for Introductory Programming Courses using Alice”. *International Journal of Modern Engineering Research, Vol. 3, pp. 350-353.*

AlAmmary, J., (2012). “Educational Technology: A way to enhance students achievement at the University of Bahrain”. Procedia – Social and Behavioral Sciences, *Vol. 55, pp. 248-257.*

Ali, A., (2009). “Successful efforts in recruiting women into technology courses – A case study”. *Issues in Information System, pp. 225-231.*

All, A., Castellar, E. P. and Looy, J., (2016). “Assessing the effectiveness of digital game-based learning: Best practices”. *Computers & Education, 92-93, pp. 90-103.*

Alliger, G. M. and Horowitz, H. M. (1989). “IBM takes the guessing out of testing”. *Training and Development Journal*, *Vol. 43, Issue, 4, pp. 69-73.*

Al-Linjawi, A. A. and Al-Nuaim, A. H., (2010). “Using Alice to Teach Novice Programmers OOP Concepts”, *Journal of King Abdulaziz University-Science, Vol. 22, No. 1, pp. 59-68.*

Allsop, Y. and Jessel, J., (2015). "Teachers' Experience and Reflections on Game-Based Learning in the Primary Classroom". *International Journal of Game-Based Learning, Vol. 5, no. 1, pp. 1-17.*

Alsabawy, Y. A., Cater-Steel, A. and Soar, J., (2016). "Determinants of perceived usefulness of e-learning systems". *Computers in Human Behavior, pp. 843-858.*

Alvarez, J., Alvarez, A., Djaouti, D. and Michaud, L., (2010). "Serious Games: Training & Teaching – Healthcare - Defence & security - Information & Communication". *2nd Edition, IDATE.*

Alyaz, Y., Spaniel-Weise. D. and Gursoy. E., (2017). "A Study on Using Serious Games in Teaching German as a Foreign Language". *Journal of Education and Learning, Vol. 6, No. 3, pp. 250-264.*

Anderson, E. F. and McLoughlin, L. (2006). "C-Sheep: Controlling Entities in a 3D Virtual World as a Tool for Computer Science Education". *Poster in Proceedings of Future Play.*

Anderson, E. F. and McLoughlin, L. (2007). "Critters in the Classroom: A 3D Computer-Game-Like Tool for Teaching Programming to Computer Animation Students". *ACM SIGGRAPH 2007 Educators Program, ACM Press.*

Anderson, E. F., McLoughlin, L., Liarokapis, F., Peters, C., Petridis, P. and De Freitas, S., (2009). "Serious Games in Cultural Heritage". *In International Symposium on Virtual Reality, Archaeology and Intelligent Cultural Heritage.*

Anderson, L. W., Krathwohl, D. R. and Bloom, B. S. (2001). "A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives". *Allyn & Bacon.*

Arbaugh, J. B. and Duray, R., (2002). "Technological and structural characteristics, student learning and satisfaction with web-based courses – An exploratory study of two on-line MBA programs". *Management Learning, Vol. 33, Issue. 3, pp. 331–347.*

Arbaugh, J. B., (2000). "Virtual classroom characteristics and student satisfaction with internet-based MBA courses". *Journal of Management Education, Vol. 24, Issue. 1, pp. 32–54.*

Arbaugh, J. B., (2002). "Managing the on-line classroom: a study of technological and behavioral characteristics of web-based MBA courses". *Journal of High Technology Management Research, Vol. 13, pp. 203–223.*

Ariffin, M. M., Oxley, A. and Sulaiman, S., (2014). "Evaluating Game-Based Learning Effectiveness in Higher Education". *Procedia - Social and Behavioral Sciences, Vol. 123, pp. 20-27.*

Ashby, J., Sadera, W. A. and McNary, S. W., (2011). "Comparing student success between developmental math courses offered online, blended, and

face-to-face". *Journal of Interactive online Learning, Vol. 10, Number. 3, pp. 128-140.*

Assaf, M., Hillegersberg, J., Spil, T. and Arikat, N., (2019). "Teachers' Perceptions about using Serious Games in Formal Education in Jordan: Possibilities and Limitations". IEEE Global Engineering Education Conference (EDUCON), pp. 436-441.

Azad, A. and Shubra, C., (2010). "Efforts to Reverse the Trend of Enrollment Decline in Computer Science Programs". *Issues in Informing Science and Information Technology, Vol. 7, pp. 209.224.*

Backlund, P., Taylor, A., Engstrom, H., et al., (2011). "Evaluation of usefulness of the Elinor Console for homebased stroke rehabilitation". *In VS-Games 2011-Proceedings of the 3rd International Conference on Games and Virtual Worlds for Serious Applications, pp. 98-103.*

Baker. A., Zhang. J. and Caldwell. R. E., (2012). "Reinforcing Array and Loop Concepts Through a Game-Like Module". *The 17th International Conference on Computer Games (CGAMES), pp. 175-179.*

Barker, J. L., McDowell, C., and Kalahar, K., (2009). "Exploring Factors that Influence Computer Science Introductory Course Students to Persist in the Major". *ACM SIGCSE Bulletin, Vol. 41, Issue. 1, pp. 153-157.*

Barnes, T., Chaffin, A., Powell, E., Lipford, H., (2008). "Game2Learn: Improving the motivation of CS1 students". *Proceedings of the 3rd international conference on Game development in computer science education, pp. 1-5.*

Barnes, T., Richter, H., Powell, E., Chaffin, A. and Godwin, A., (2007). "Game2Learn: Building CS1 learning games for retention". *ITiCSE '07 Proceedings of the 12th annual SIGCSE conference on Innovation and technology in computer science education, Vol. 39, Issue. 3, pp. 121-125.*

Battistella, P. E. and von Wangenheim, C. G., (2016). "Games for Teaching Computing in Higher Education – A Systematic Review". *IEEE Technology and Engineering Education (ITEE), Vol. 1, No. 3.*

Bayliss, J. D., (2007). "The Effects of Games in CS1-3". *Journal of Game Development, Vol. 2, Issue. 2, pp. 7-17.*

Beaubouef, T., and Mason, J., (2005). "Why the high attrition rate for computer science students: some thoughts and observations". *ACM SIGCSE Bulletin, Vol. 37, Issue 2, pp. 103-106.*

Becker, K. (2005). "How Are Games Educational? Learning Theories Embodied in Games". *Proceedings of DiGRA 2005 - the Digital Games Research Association's 2nd International Conference.*

Bellotti, F., Kapralos, B., Lee, K., Moreno-Ger, P. and Berta, R., (2013). "Assessment in and of Serious Games: An Overview". *Advances in Human-Computer Interaction - Special issue on User Assessment in Serious Games and Technology-Enhanced Learning, Vol. 2013.*

Bellotti, F., Ott, M., Arnab, S., Berta, R., De Freitas, S., Kiili, K. and De Gloria, A. (2011). "Designing Serious Games for education: from Pedagogical principles to game mechanisms". *5th European Conference on Games Based Learning, pp. 26-34*

Bennedsen, J. and Caspersen, M., (2007). "Failure Rates in Introductory Programming". *ACM SIGCSE Bulletin archive, Vol. 39, Issue. 2, pp. 32-36.*

Benokraitis, V., Bizot, B., Brown, R. and Martens, J., (2009). "Reasons for CS decline: Preliminary evidence". *Journal of Computing Sciences in Colleges, Vol. 24, Issue. 3, pp. 161-162.*

Berg, B. L., (2007). "Qualitative research methods for the social sciences". *London: Pearson.*

Bergeron, B. P., (2006). "Developing Serious Games". *Charles River Media, Hingham, Mass, USA, 2006.*

Bergin, S. and Reilly, R., (2005). "The influence of motivation and comfort-level on learning to program". *In Proceedings of the 17th Annual Workshop of the Psychology of Programming Interest Group, pp. 293-304.*

Bergomi, M., Ludovico, L. A. and Barate, A., (2013). "Development of Serious Games for Music Education*". Journal of E-Learning and Knowledge Society, Vol. 9, pp. 89-104.*

Betancur, A. J., Rodríguez, C., and Ezparragoza, I., (2011) "An Undergraduate collaborative design experience among institutions in the Americas," *presented at the 8th WSEAS International Conference on Engineering Education, pp. 263-267.*

Betas, R., (2004), "A critical analysis of evaluation practice: the Kirkpatrick model and the principle of beneficence". *Evaluation and Program Planning, Vol. 27, pp. 341-347.*

Bhattacherjee, A., (2012). "Social Science Research: Principles, Methods, and Practices". *Textbooks Collection, 3.* *http://scholarcommons.usf.edu/oa_textbooks/3*

Bierre, K., Ventura, P., Phelps, A. and Egert, C., (2006). “Motivating OOP by blowing things up: An exercise in cooperation and competition in an introductory java-programming course”. *ACM SIGCSE Bulletin, Vol. 38, Issue. 1, pp. 354-358.*

Biju, S. M., (2013). "Difficulties in understanding object oriented programming concepts." In *Innovations and Advances in Computer, Information, Systems Sciences, and Engineering, pp. 319-326.*

Bittencourt, R. A., dos Santos, D. M. Rodrigues, C. A., Batista, W. P. and Chalegre, H. S., (2015). "Learning programming with peer support, games, challenges and scratch". *IEEE Frontiers in Education Conference (FIE).*

Bloom, B. S. (1956). “Taxonomy of educational objectives: The classification of educational goals”. *New York: David McKay Company.*

Boehm, W., Brown, R., Kaspar, H. et al., (1978). “Characteristics of Software Quality”. *Amsterdam, North Holland.*

Bonakdarian, E. and White. L., (2004). “Robocode throughout the curriculum”. *Journal of Computing Sciences in Colleges, Vol. 19, Issue. 3, pp. 311-313.*

Brom, C., Sisler, V. and Slavik, R., (2010). “Implementing Digital Game-Based Learning in Schools: Augmented Learning Environment of ‘Europe 2045’”. *Multimedia Systems, Vol. 16, No. 1, pp.23-41.*

Brown, M. B. and Forsythe, A. B., (1974). “Robust Tests for the Equality of Variances”. *Journal of the American Statistical Association, Vol. 49, pp. 364-367.*

Butt, B. Z. and Rehman, K., (2010). "A study examining the students satisfaction in higher education". *Procedia Social and Behavioral Sciences 2, pp. 5446-5450.*

Buttussi, F. and Chittaro, L., (2010). “Smarter phones for healthier lifestyles: an adaptive fitness game”. *IEEE Pervasive Computing, Vol. 9, Issue. 4, pp. 51-57.*

Calder, N. (2010). “Using Scratch: an integrated problem-solving approach to mathematical thinking”. *Australian Primary Mathematics Classroom, 15(4), 9–14.*

Calderon, A. and Ruiz, M., (2015). “A systematic literature review on serious games evaluation: An application to software project management”. *Computers & Education, Vol. 87, pp. 396-422.*

Cantwell, B., (2002). “A study of factors promoting success in computer science including gender differences”. *Computer Science Education, Vol. 12, pp. 141-164.*

Capece, G., and Campisi, D., (2013). “User satisfaction affecting the acceptance of an elearning platform as a mean for the development of the human capital”. *Behaviour & Information Technology, Vol. 32, Issue. 4, pp. 335–343.*

Carvalho, C. V., (2012). "Is Game-Based Learning Suitable for Engineering Education?". *Global Engineering Education Conference (EDUCON), pp. 808-815.*

Cederholm, H., Hilborn, O., Lindley, C., et al., (2011). “The aiming game: using a game with biofeedback for training in emotion regulation”. *In Proceedings of the 5th International Conference on Digital Research Association: Think Design Play.*

Cesare, R. V., (2013). “Sorting Algorithms: Digital Game Based Learning”. *Available at: http://sortingalgorithmsgame.webs.com/. [Accessed 22 July 2018].*

Chang, W., Chou, Y. and Chen, K., (2010). "Game-based digital learning system assists and motivates C programming language learners". *The 6th International Conference on Networked Computing and Advanced Information Management.*

Chen, T. L., (2014)., “Exploring e-learning effectiveness perceptions of local government staff based on the diffusion of innovations model”. *Administration & Society, Vol. 46, Issue. 4, pp. 450–466.*

Chew, B., (2017). "An Efficient Framework for Game-Based Learning activity". *IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE), pp. 147-150.*

Christensen, R., (2011). "Plane Answers to Complex Questions: The Theory of Linear Models". *Springer, 4th Edition*.

Clegg, F. (1998)., “Simple statistics A Course Book for the Social Sciences”. *Cambridge: Cambridge University Press.*

Cliburn. C. D., (2006). “The Effectiveness of Games as Assignments in an Introductory Programming Course”. *Proceedings Frontiers in Education, 36th Annual Conference, pp. 6-10.*

Comber, O., Motsching, R., Mayer, H. and Haselberger, D., (2019). "Engaging Students in Computer Science Education through Game Development with Unity". *IEEE Global Engineering Education Conference (EDUCON), pp. 199-205.*

Connolly, T. M., Stansfield, M. and McLellan, E., (2006). “Using an Online Games-Based Learning Approach to Teach Database Design Concepts”. *The Electronic Journal of e-Learning, Vol. 4, Issue. 1, pp. 103-110.*

Connolly, T., Stansfield, M. and Hainey, T., (2009). “Development of a general framework for evaluating games-based learning”. *In T. Connolly, M. Stansfield, & L. Boyle (Eds.), Games-Based Learning Advancements for Multi-Sensory Human Computer Interfaces: Techniques and Effective Practices, pp. 251-273.*

Cooper, S., Dann, W. and Pausch, R. (2000). “ALICE: A 3-D Tool For Introductory Programming”. *Journal of Computing Sciences in Colleges, Vol. 15, Issue. 5, pp. 107-116.*

Corney, M., Teague, D. and Thomas, R., (2010). “Engaging Students in Programming”. Proceedings of the Twelfth Australasian Conference on Computing Education, pp. 63-72.

Corral, R. M. J., Balcells, C. A., Estévez, M. A., Moreno, J. G. and Ramos, F. J. M., (2014). "A game-based approach to the teaching of object-oriented programming languages". *Computers & Education, Vol. 73, pp. 83-92.*

Costantino, F., Di Gravio, G., Shaban, A. et al., (2012). “A simulation based game approach for teaching operations management topics”. *In Proceedings of the 2012 Winter Simulation Conference, pp. 1-12.*

Couceiro, M. R., Papastergiou, M., Kordaki, M. and Veloso, I. A., (2013). “Design and evaluation of a computer game for the learning of Information and Communication Technologies (ICT) concepts by physical education and sport science students”. *Education and Information Technologies, Vol. 18, Issue. 3, pp. 531-554.*

Csikszentmihalyi, M., (1992). “Flow: the Psychology of Happiness”. *London: Random House.*

Dahalan, N., Hassan, H., and Atan, H., (2012). “Student engagement in online learning: learners attitude toward e-mentoring”. *Procedia-Social and Behavioral Sciences, Vol. 67, pp. 464-475.*

Dale, D., N., (2006). "Most Difficult Topics in CS1: Results of an Online Survey of Educators". *ACM SIGCSE Bulletin, Vol 38, Issue 2, pp. 49-53.*

Dann, W., Cooper, S. and Pausch, R. (2000). “Making the Connection: Programming with Animated Small World”. *Proceedings of the 5th annual SIGCSE/SIGCUE ITiCSE conference on Innovation and technology in computer science education, Vol. 32, Issue. 3, pp. 41-44.*

Dauski, S. and Quaye, A., (2016). "Undergraduate Students' Failure in Programming Courses in Institutions of Higher Education in Developing Countries: A Nigerian Perspective". Electronic Journal of Information Systems in Developing Countries, Vol. 76, pp. 1-18.

Davies, S. P. (1993). "Models and theories of programming strategy". *International Journal of Man-Machine Studies, Vol. 39, pp. 313-320.*

Davis, D. F., (1989). "Perceived usefulness, perceived ease of use and user acceptance of information technology". *MIS Quarterly, Vol. 13, Issue. 3, pp. 319–340.*

De Freitas, S. and Oliver, M., (2006). "How can exploratory learning with games and simulations within the curriculum be most effectively evaluated?". *Computers & Education, Vol. 46, pp.249-264.*

De Freitas, S., (2004). "Learning through Play. Internal report". *London: Learning and Skills Research Centre.*

De Freitas, S., (2005). "Review of the uptake and embedding of digital content". *Internal report. Coventry: Becta.*

De Kereki, I. F., (2008). "Scratch: Applications in Computer Science 1". *38th Annual Frontiers in Education Conference.*

De Paolis, L. T., (2012). "Serious game for laparoscopic suturing training". *In Proceedings of the 6th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS '12), pp. 481-485.*

Debabi, W. and Champagnat, R., (2017). "Towards Architecture for Pedagogical and Game Scenarios Adaptation in Serious Games". *International Conference e-Learning, pp. 63-70.*

Dele-Ajayi, O., Strachan, R., Sanderson, J. and Pickard, A., (2016). "Learning Mathematics Through Serious Games: An engagement framework". *In Frontiers in Education Conference (FIE).*

Desurvire, H., Caplan, M. and Toth, A. J., (2004). "Using Heuristics to Evaluate the Playability of Games". *In CHI '04 Extended Abstracts on Human Factors in Computing Systems, pp. 1509-1512.*

Deterding, S., Dixon, D., Khaled, R. and Nacke, L., (2011). "From game design elements to gamefulness: defining "gamification"". *MindTrek '11 Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments, pp. 9-15.*

Dix, J., Finlay, E., Abowd, D. et al., (2004). “Human-Computer Interaction”. 3rd Ed. Harlow Assex: Pearson Education Limited.

Djaouti, D., Alvarez, J., Jessel, J. P. and Rampnoux, O., (2011). “Origins of serious games”. *In book: Serious Games and Edutainment Applications Edition: Chapter: Origins of Serious Games, Publisher: Springer London.*

Do, T. and Lee, J., (2009). “A multiple-level 3D-LEGO game in augmented reality for improving spatial ability”. *In: Jacko J.A. (eds) Human-Computer Interaction. Interacting in Various Application Domains. HCI 2009. Lecture Notes in Computer Science, Vol. 5613.*

Doll, J. W., Deng, X., Raghunathan, S. T., et al., (2004). “The Meaning and Measurement of User Satisfaction: A Multigroup Invariance Analysis of the End-User Computing Satisfaction Instrument”. *Journal of Management Information System, Vol. 21, Issue. 1, pp. 227-262.*

Driskell JE, Dwyer DJ. (1984). Microcomputer videogame based training. Educational Technology, 24, 11–17.

Dumas, J. and Redish, J., (1999). “A Practical Guide to Usability Testing”. *Intellect Books Exeter, UK.*

Dunwell, I., Lameras, P., Star, K., et al., (2013). “Metycoon: a game-based approach to career guidance”. *In VS-Games 2013- Proceedings of the 5th International Conference on Games and Virtual Worlds for Serious Applications, pp. 1-6.*

Dush, P., Jaworski, T., (2018). "Enriching Computer Science Programming Classes with Arduino Game Development". *11th International Conference on Human System Interaction (HIS).*

Eagle, M., and Barnes, T., (2008), "Wu’s Castle: Teaching Arrays and Loops in a Game". *ACM SIGCSE Bulletin, volume 40, pp. 245-249.*

Eagle, M., and Barnes, T., (2009). "Experimental Evaluation of an Educational Game for Improved Learning in Introductory Computing". *SIGCSE '09 Proceedings of the 40th ACM technical symposium on Computer science education, Vol. 41, pp. 321-325.*

Egenfeldt-Nielsen, S., (2006). “Overview of research on the educational use of video game”. *Digital Kompetanse, pp. 184-213.*

Eichenberg, C., Grabmayer, G. and Green, N., (2016). "Acceptance of Serious Games in Psychotherapy: An Inquiry into the Stance of Therapists and Patients". *Telemed J E Health, Vol. 22, Issue. 11, pp. 945-951.*

El Borji, Y. and Khaldi, M., (2014). “Comparative Study to Develop a Tool for the Quality Assessment of Serious Games Intended to be used in Education”. *International Journal of Emerging Technologies in Learning, Vol. 9, Issue. 9, pp. 50-55.*

Emmerich, K. and Bockholt, M., (2016). “Serious Games Evaluation: Processes, Models, and Concepts”. *In: Dörner R., Göbel S., Kickmeier-Rust M., Masuch M., Zweig K. (eds) Entertainment Computing and Serious Games. Lecture Notes in Computer Science, Vol. 9970, pp. 265-283.*

Enah, C., Piper, K. and Moneyham, L., (2014). “Qualitative evaluation of the relevance and acceptability of a web-based HIV prevention game for rural adolescents”. *Journal of Pediatric Nursing, Vol. 30, Issue. 2, pp. 321-328.*

Erol, O. and Kurt, A. A., (2017). “The effects of teaching programming witch Scratch on pre-service information technology teachers’ motivation and achievement”. *Computers in Human Behavior, Vol. 77, pp. 11-18.*

Faria, A. J., Hutchinson, D., Wellington, W. J. and Gold, S., (2009). “Developments in business gaming: a review of the past 40 years”. *Simulation and Gaming, Vol. 40, Issue. 4, pp. 464–487.*

Farid, S., Ahmad, R., Niaz, A.I., Arif, M., Shamshirband, S. and Khattak, D.M., (2015) "Identification and prioritization of critical issues for the promotion of e-learning in Pakistan", *Computers in Human Behavior, Vol. 51, pp. 161-171.*

Farjad, S., (2012). "The Evaluation Effectiveness of training courses in University by Kirkpatrick Model (case study: Islamshahr university)". *Procedia - Social and Behavioral Sciences Vol. 46, pp. 2837–2841.*

Farrel, P. K. and Stewart, K. R., (2006). "Comprehensive Study of Tests For Normality And Symmetry: Extending The Spiegelhalter Test*". Journal of Statistical Computation and Simulation, Vol. 76, Issue. 9, pp. 803-816.*

Feigenspan J, Kastner C, Liebig J, Apel S, Hanenberg S (2012). “Measuring programming experience”. *International Conference on Program Comprehension, pp. 73–82.*

Ferriera, N., (2008). “Serious Games”. *Available at: https://paginas.fe.up.pt/~aas/pub/Aulas/DiCG/NunoFerreira.pdf*

Fotaris, P., Mastoras, T., Leinfellner, R. and Rosunally. Y., (2016). “Climbing up the Leaderboard: An Empirical Study of Applying Gamification Techniques to a Computer Programming Class”. *Electronic Journal of e-Learning, Vol. 14, pp. 94-110.*

Fotaris, P., Mastoras, T., Leinfellner, R. and Rosunally. Y., (2015). “From Hiscore to High Marks: Empirical Study of Teaching Programming Through Gamification”. *9th European Conference on Games Based Learning ECGBL.*

Fotiadis, K.A. and Sigala, M., (2015) "Developing a framework for designing an Events Management Training Simulation (EMTS)". *Journal of Hospitality, Leisure, Sport & Tourism Education 16, pp 59-71.*

Fredricks, J. A., Blumenfeld, P. C. and Paris, A. H. (2004). “School engagement: Potential of the concept, state of the evidence”. *Review of Educational Research, Vol. 74, No. 1, pp. 59–109.*

Fu, F., Su, R. and Yu, S., (2009). "EGameFlow: A scale to measure learners’ enjoyment of e-learning games". *Computers & Education, Vol. 52, pp. 101-112.*

Gagné, R. M., (1965). “The conditions of learning”. *New York: Holt, Rinehart and Winston.*

Galgouranas, G. and Xinogalos, S., (2018). “jAVANT-GARDE: A Cross-Platform Serious Game for an Introduction to Programming with Java”. *Simulation & Gaming, Vol. 49, pp. 751-767.*

Galves. J., Guzman. E. and Conejo. R., (2009). “A blended E-learning experience in a course of object oriented programming fundamentals”. *Knowledge-Based Systems, Vol. 22, pp. 279-286.*

Garcia-Mundo, L., Genero, M. and Piattini, M., (2015). "Towards a Construction and Validation of a Serious Game Product Quality Model". *7th International Conference on Games and Virtual Worlds for Serious Applications (VS-Games).*

Garris, R., Ahlers, R. and Driskell, J. E., (2002). "Games, Motivation, and Learning: A Research and Practice Model". *Simulation and Gaming, Vol. 33, pp. 441-467.*

Gee, J., (2003). “What video games have to teach us about learning and literacy”. *New York: Palgrave Macmillan.*

Gibson, B. and Bell, T., (2013). “Evaluation of games for teaching Computer Science”. *The 8th Workshop in Primary and Secondary Computing Education, pp. 51-60.*

Girard, C., Ecalle, J. and Magnant, A., (2013). "Serious games as new educational tools: how effective are they? A meta-analysis of recent studies". *Journal of Computer Assisted Learning, Vol. 29, pp. 207-219.*

Gomes, A. and Mendes, A. J. N., (2007). "Learning to program-difficulties and solutions". *Proceedings of the International Conference on Engineering Education, pp. 283-287.*

Goosen, L. and Pieterse, V., (2005). "Roller coaster riding: highs and lows of understanding OO". *Proceedings of the 35th conference of SACLA.*

Green, S., Chan, C. and Plant, N., (2016). "Student at Risk: Identification and Remedial Action System for Improving Retention on Computer Science Programmes". *New Directions in the Teaching of Physical Sciences, Vol. 11, Issue. 1.*

Greitzer, F. L., Kuchar, O. A. and Huston, K., (2007). "Cognitive science implications for enhancing training effectiveness in a serious gaming context". *ACM Journal on Educational Resources in Computing, Vol. 7, No. 3, article 2.*

Gubrium, J. F. and Holstein, J. A., (2002). "Handbook of Interview Research: Context and Method". *Thousand Oaks, CA: Sage.*

Guest, G., Bunce, A. and Johnson, L., (2006). "How many interviews are enough? An experiment with data saturation and variability". *Field Methods, Vol. 18(1), 24.*

Guillén-Nieto, V. and Aleson-Carbonell, M., (2012). "Serious games and learning effectiveness: The case of It's a Deal!". *Computers & Education, Vol. 58, pp. 435-488.*

Gunasekaran, A., McNeil, R. D. and Shaul, D., (2002). "E-learning: research and applications". *Industrial and Commercial Training, Vol. 34, Issue. 2, pp. 44-53.*

Gurer D. and Camp T., (2002), "An ACM-W literature review on women in computing". *ACM SIGCSE Bulletin - Women and Computing, Vol. 34, Issue. 2, pp. 121-127.*

Gusev, D. A., (2018). "Using Chess Programming in Computer Education". *Association Supporting Computer Users in Education (ASCUE), pp. 92-102.*

Guzdial. M., (2014). "A Biased Attempt at Measuring Failure Rates in Introductory Programming". *Computing Education Research Blog. https://computinged.wordpress.com/2014/09/30/a-biased-attempt-at-measuring-failure-rates-in-introductory-programming/. Accessed 2019 August 30*

Hakulinen, L, (2011). "Using Serious Games in Computer Science Education". *The 11th Koli Calling International Conference on Computing Education Research, pp. 83-88.*

Hanks, B., McDowell. C., Draper, D. and Krnjajic. M., (2004). “Program quality with pair programming in CS1”. *In ITiCSE '04: Proceedings of the 9th annual SIGCSE conference on Innovation and technology in computer science education, pp. 176-180.*

Hanzu-Paraza, R., and Barsan, E., (2010). “Teaching techniques –modern bridges between lecturers and students”. *In Proc. 7th WSEAS International Conference on Engineering Education, pp. 176-181.*

Harteveld, C., Guimarães, R., Mayer, S., et al., (2010). “Balancing play, meaning and reality: the design philosophy of LEVEE PATROLLER”. *Simulation & Gaming, Vol. 41, Issue. 3, pp. 316–340.*

Hartness, K., (2004). “Robocode: Using Games to Teach Artificial Intelligence”. *Journal of Computing Sciences in Colleges, Vol. 19, Issue. 4, pp. 287-291.*

Hauge, J. B. and Riedel, J. C., (2012). “Evaluation of simulation games for teaching engineering and manufacturing”. *Procedia Computer Science, Vol. 15, pp. 210-220.*

Hays, R. T., (2005). “The effectiveness of instructional games: a literature review and discussion”. *(Technical Report No. 2005-004). Orlando, FL: Naval Air Warfare Centre, Training Systems Division.*

Hensman, A., (2007). “Evaluation of Robocode as a Teaching Tool for Computer Programming”. *Proceedings of the National Digital Learning Repository (NDLR) Symposium, Trinity College Dublin, Ireland.*

Hill, J. M. D., Ray, C. K., Blair, J.R.S. and Carver, C.A., (2003). “Puzzles and Games: addressing different learning styles in teaching operating systems concepts”. *Proc. of the 34th SIGCSE technical symposium on Computer science education, pp. 182-186.*

Hillyard. C., Angotti. R., Panitz. M., Sung. K., Nordlinger. J. and Goldstein. D., (2010). “Game-Themed Programming Assignments For Faculty: A Case Study”. *SIGCSE '10 Proceedings of the 41st ACM technical symposium on Computer science education, pp. 270-274.*

Hinds, M. Baghaei, N., Ragon, P., Lambert, J., Rajakaruna, T., Houghton, T. and Dacey, S., (2017). "RUNJUMPCODE: AN EDUCATIONAL GAME FOR EDUCATING PROGRAMMING". *13th International Conference Mobile Learning, pp. 109-113.*

Hong, K. S., (2002). “Relationships between students’ and instructional variables with satisfaction and learning from a Web-based course”. *Internet and Higher Education, Vol. 5, pp. 267–281.*

Hou, H. and Li, M., (2014). “Evaluating multiple aspects of a digital educational problem-solving-based adventure game”. *Computers in Human Behavior, Vol. 30, pp. 29-38.*

Hsu, C. L. and Lu, H. P., (2004). “Why do People Play On-Line Games? An Extended TAM with Social Influences and Flow Experience”. *Information and Management, vol.41, Issue. 7.*

Hwang, G. J., and Wu, P. H., (2012). “Advancements and trends in digital game-based learning research: A review of publications in selected journals from 2001 to 2010”. *British Journal of Educational Technology, Vol. 43, Issue. 1, pp. 6-10.*

Ingadottir, B., Blondal, K., Thue, D., et al., (2017). "Development, Usability, and Efficacy of a Serious Game to Help Patients Learn About Pain Management After Surgery: An Evaluation Study". *JMIR Serious Games, Vol. 5, No. 2.*

ISO 9241-210:2010 Ergonomics of human-system interaction—Part 210: Human-cantered design for interactive systems. (2010).

ISO/IEC 9126:1998 Ergonomic requirements for office work with visual display terminals (VDTs) - Part 11: Guidance on usability. (1998).

ISO/IEC 9126:2001 Software engineering - Product quality - Part 1: Quality model. International Organization for Standardization and International Electrotechnical Commission. (2001).

Jenkins, T., (2002). “On the Difficulty of Learning to Program”. *In Proceedings for the 3rd Annual conference of the LTSN Centre for Information and Computer Sciences, pp. 53-58.*

Jordaan, D. B., (2018). "Board Games in the Computer Science Class to Improve Students’ Knowledge of the Python Programming Language". *International Conference on Intelligent and Innovative Computing Applications (ICONIC).*

Jordan, C., Knapp, M., Mitchell, D., Claypool, M. and Fisler, K., (2011) “CounterMeasures: a game for teaching computer security”, *Proc. of the 10th Annual Workshop on Network and Systems Support for Games.*

Jung, H., Kim, S. and Chung, C., (2004). "Measuring software product quality: a survey of ISO/IEC 9126". *IEEE Software, Vol. 21, Issue. 5, pp. 88-92.*

Kanuka, H. and Nocente, N., (2003). “Exploring the effects of personality type on perceived satisfaction with web-based learning in continuing professional development”. *Decision Education, Vol. 24, Issue. 2, pp. 227–245.*

Kapralos, B., Fisher, S., Clarkson, J. and van Oostveen, R., (2015). "A course on serious game design and development using an online problem-based learning approach". *Interactive Technology and Smart Education, V.12, pp. 116-136.*

Kathleen, F. F., Morales, V. C., Dunn, S. L., Godde, K. and Weaver, P. F., (2017). "An Introduction to Statistical Analysis in Research: With Applications in the Biological and Life Sciences". John Wiley & Sons, Inc.

Kebritchi, M., Hirumi A. and Bai H., (2010). "The effects of modern mathematics computer games on mathematics achievement and class motivation". *Computers & Education Vol. 55, pp. 427–443.*

Kelleher, C. and Pausch, R., (2005). "Lowering the barriers to programming: A taxonomy of programming environment and languages for novice programmers". *ACM Computing Surveys, Vol. 37, Issue. 2, pp. 83-137.*

Keskin, S., (2006). "Comparison of Several Univariate Normality Tests Regarding Type I Error Rate and Power of the Test in Simulation Based Small Samples". *Journal of Applied Science Research, Vol. 2, Issue. 5, pp. 296-300.*

Kiili, K., (2005). "Digital Game-based Learning: Towards an Experiential Gaming Model". *The Internet and Higher Education, Vol. 8, pp. 13-24.*

Kinnunen, P. and Malmi, L., (2006). "Why students drop out CS1 course?". *ICER '06 Proceedings of the second international workshop on Computing education research, pp. 97-108.*

Kirkpatrick, D. L., (1998). "Evaluating Training Programs: The Four Levels (2nd Ed)". *San Francisco: BerrettKoehler.*

Kjeldskov, J., Skov, B. and Stage, J., (2004). "Instant Data Analysis: Conducting Usability Evaluations in a Day". *Proceedings of the third Nordic Conference on Human Computer Interaction, pp. 233-240.*

Klasen, H., Woerner, W., Wolke, D., Meyer, R., Overmeyer, S., Kaschnitz, W., Rothenberger, A. and Goodman, R., (2000). "Comparing the German version of the Strengths and Difficulties Questionnaire (SDQ-Deu) and the Child Behavior Checklist". *European Child and Adolescent Psychiatry, Vol. 9, pp. 271–276.*

Kothare, C. R., (2004). "Research Methodology Methods and Techniques". *2nd edition, New Age International Publishers.*

Kumar, B. and Sharma, K., (2018). "A Gamified Approach to Achieve Excellence in Programming". *4th International Conference on Computing Sciences, pp. 107-114.*

Kunkle, W. and Allen, R., (2016). “The Impact of Different Teaching Approaches and Languages on Student Learning of Introductory Programming Concepts”. *ACM Transactions on Computing Education, Vol. 16, Issue. 1, Article No. 3.*

Laamarti, F., Eid, M. and El Saddik, A., (2014). “An Overview of Serious Games”. *International Journal of Computer Games Technology. International Journal of Computer Games Technology, Vol. 2014.*

Lahtinen, E., Ala-Mutka, K. and Jarvinen, H., (2005). "A study of the difficulties of novice programmers". *ITiCSE '05 Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education, pp. 14-18.*

Lamb, A., Johnson, L. (2011). Scratch: computer programming for 21st century learners. *Teacher Librarian, Vol. 38, Issue. 4, pp. 64–68.*

Lanyi, S., Brown, J., Standen, P. et al., (2012). “Results of user interface evaluation of serious games for students with intellectual disability”. *Acta Polytechnica Hungarica, Vol. 9, pp. 225-245.*

Laporte, L. and Zaman, B., (2018). "A comparative analysis of programming games, looking through the lens of an instructional design model and a game attributes taxonomy". *Entertainment Computing, Vol. 25, pp. 48-61.*

Law, L. E. and Sun, X., (2012). “Evaluating user experience of adaptive digital educational games with Activity Theory”. *International Journal of Human-Computer Studies, Vol. 70 Issue. 7, pp. 478-497.*

Lee, Y. H., Hsieh, Y. C. and Ma, C. Y., (2011). “A model of organizational employees’ elearning systems acceptance”. *Knowledge-Based Systems, Vol. 24, Issue. 3, pp. 355–366.*

Lee, Y., (2011). “Scratch: multimedia programming environment for young gifted learners”. *Gifted Child Today, Vol. 34, Issue. 2, pp. 26–31.*

Lepper, M. R. and Malone, T. W. (1987). “Intrinsic motivation and instructional effectiveness in computer-based education”. *In R. E. Snow & M. J. Farr (Eds.) “Aptitude, learning, and instruction: Vol. 3. Cognitive and affective process analysis” Hillsdale NJ: Erlbaum, pp. 255-286.*

Leutenegger, S. and Edgington J., (2007). “A games first approach to teaching introductory programming”. *Proceedings of the 38th SIGCSE technical symposium on Computer science education, Vol. 39, Issue. 1, pp. 115-118.*

Liu, P., (2008). “Using open-source Robocode as a java programming assignment”. *SIGCSE Bulletin, Vol. 40, Issue. 4, pp. 63–67.*

Livovsky, J. and Poruban, J., (2014). “Learning object-oriented paradigm by playing computer games: concepts first approach”. *Central European Journal of Computer Science, Vol. 4, pp. 171-182.*

Long, J., (2006). "Why Not Have Fun While Learning: Using Programming Games in Software Programming Education". *ISECON 2006, Vol. 23, pp. 1-10.*

Long, J., (2007). “Just For Fun: Using Programming Games in Software Programming Training and Education – A Field Study of IBM Robocode Community”. *Journal of Information Technology Education, Vol. 6, pp. 279-290.*

Loon, M., Evans, J. and Kerridge, C., (2015) "Learning with a strategic management simulation game: A case study", *The International Journal of Management Education 13, pp 227-236.*

Lopez, M., Whalley, J., Robbins, P. and Lister, R., (2008). “Relationships between reading, tracing and writing skills in introductory programming”. *Proceedings of the Fourth International Workshop on Computing Education Research, pp.101-112.*

Lu, J. and Fletcher, G., (2009). "Thinking about computational thinking". *ACM SIGCSE Bulletin - SIGCSE '09, Vol. 41, Issue. 1, pp. 260-264.*

MacLean, L., (2010). “Recruitment and Retention of Women in Computer Science and Information Systems: How and Why”. *In Proc. 2nd International Conference on Education and New Learning Technologies*, *pp. 1585-1591.*

Malliarakis, C., Satratzemi, M. and Xinogalos, S., (2014). "CMX: Implementing an MMORPG for Learning Programming". *Proceeding of the European Conference on Games-based Learning, Vol. 1, pp. 346-355.*

Malliarakis, C., Satratzemi, M. and Xinogalos, S., (2017). “CMX: The Effects of an Educational MMORPG on Learning and Teaching Computer Programming”. *IEEE Transactions on Learning Technologies, Vol. 10, No. 2, pp. 219-235.*

Malliarakis, C., Shabalina, O. and Mozelius, P., (2015). "How to Build an Ineffective Serious Game: Worst Practices in Serious Game Design". *The 9th European Conference on Games Based Learning, ECGBL.*

Malone, T. W. and Lepper, M., (1987). “Making Learning fun: A Taxonomy of intrinsic motivations for learning”. In Aptitude, learning and instruction. Conative and affective process analysis". *Lawrence Erlbaum, Hillsdale, N.J, pp. 223-253.*

Manaris, B., (2007). “Dropping CS Enrollments: Or The Emperor’s New Clothes?”. *ACM SIGCSE Bulletin, Vol. 39, Issue. 4, pp. 6-10.*

Manera, V., Petit, D., Derreumaux, A. et al., (2015). “Kitchen and cooking,' a serious game for mild cognitive impairment and Alzheimer's disease: a pilot study”. *Frontiers in Aging Neuroscience, Vol. 7.*

Manero, B., Fernandez-Vara, C. and Fernandez-Manjon, B., (2013). “E-Learning Takes the Stage: From La Dama Boba to a Serious Game”. *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje, Vol. 8, Issue. 4, pp. 179-204.*

Maria, M., De Oliveira, M. F. D. and Waddell, G., (2016). "Game-Based Learning of Musical Instruments: A Review and Recommendations". *In the 10th European Conference on Games Based Learning.*

Marin-Garcia, J. A., and Mauri, J. L., (2007). “Teamwork with University Engineering Students. Group Process Assessment Tool”. In *Proc. the 3rd WSEAS/IASME International Conference on Educational Technologies, pp. 391–396.*

Markham, S. A. and King, K. N., (2010). "Using Personal Robots in CS1: Experiences, Outcomes, and Attitudinal Influences". *ITiCSE’10, pp. 204-208.*

Mathrani, A., Christian, S. and Ponder-Sutton, A., (2016). “PlayIT: Game Based Learning Approach for Teaching Programming Concepts”. *Educational Technology & Society, Vol. 19, pp. 5-17.*

Mavromihales, M., Holmes, V. and Racasan, R., (2018). “Game-based learning in mechanical engineering education: Case study of games-based learning application in computer aided design assembly”. *International Journal of Mechanical Engineering Education, Vol. 47, pp. 156-179.*

Mayer, I., Bekebrede, G., Harteveld, C., Warmelink, H., Zhou, Q., Ruijven, T., Lo, J., Kortmann, R. and Wenzler, I., (2014). "The Research and Evaluation of Serious Games: Toward a Comprehensive Methodology". *British Journal of Educational Technology, Vol. 45, No. 3, pp. 502-527.*

Mayo, M. J., (2007). “Games for science and engineering education”. *Communications of the ACM, Vol. 50, Issue. 7, pp. 31–35.*

Mazgon, J and Stefanc, D., (2012). “Importance of the various characteristics of educational materials: Different opinions, different perspectives”. *The Turkish Online Journal of Educational Technology, Vol. 11, Issue. 3, pp. 174-188.*

Mazlan, M. N. A., (2012). “Students’ Perception of Motivation to Learn: Does an avatar Motivates?”. *Durham University.*

McDowell, C., Werner, L., Bullock, E. H., and Fernald, J., (2006). “Pair programming improves student retention, confidence, and program quality,” *Communications of ACM, Vol. 49, Issue. 8, pp. 90-95.*

Melero, J., Hern'ndez-Leo, D. and Blat, J., (2012). “Considerations for the design of mini-games integrating hints for puzzle solving ICT-related concepts”. *Proc. of the 12th International Conference on Advanced Learning Technologies.*

Mestadi, W., Nafil, K., Touahni, R. and Messoussi, R., (2018). "An Assessment of Serious Games Technology: Toward an Architecture for Serious Games Design". *International Journal of Computer Games Technology, Vol. 2018*.

Michael, D. and Chen, S., (2006). “Serious Games: That Educate, Train, and Info”. *Thomson Course Technology.*

Mikovec, Z., Salvik, P. and Zara, J., (2009). "Cultural Heritage, User Interfaces and Serious Games at CTU Prague". *15th International Conference on Virtual Systems and Multimedia, pp. 211-216.*

Miljanovic, M. A. and Bradbury, J. S., (2017). "RoboBUG: A Serious Game for Learning Debugging Techniques". *Proceedings of the 2017 ACM Conference on International Computing Education Research, pp. 93-100.*

Miller L. M., Chang C.-I., Wang S., Beier M. E. and Klisch Y., (2011). “Learning and motivational impacts of a multimedia science game”. *Computers & Education Vol. 57, pp. 1425–1433.*

Misfeldt, M., (2015) "Scenario Based Edcation as a Framework for Understanding Students Engagement and Learning in a Project Management Simulation Game", *The Electronic Journal of e-Learning Volume 13 issue 3, pp 181-191.*

Mishra, S., Balan, S., Iyer, S. and Murthy, S., (2014). “Effect of a 2-week Scratch Intervention in CS1 on Learners with Varying Prior Knowledge”. *ITiCSE '14 Proceedings of the 2014 conference on Innovation & technology in computer science education, pp. 45-50.*

Mitgutsch, K., Alvarado, N., (2012). “Purposeful by design?: a serious game design assess-ment framework”. *Proceedings of the International Conference on the Foun-dations of Digital Games,* pp. 121–128.

Mittermeir, R. T., Hochmüller, E., Bollin, A., Jäger, S. and Nusser, M., (2003) “AMEISE – A media education initiative for Software Engineering”, *Proc. of the Int. Workshop on Interactive Computer Aided Learning.*

Monette, D. R., Thomas J. S. and Cornell R. D., (1986). “Applied Social Research: Tools for the Human Services”. *Forth Worth, TX, Holt, Rinehart, and Winston.*

Mortara, M., Catalano, E., Fiucci, G., et al., (2014). “Evaluating the effectiveness of serious games for cultural awareness: the Icura user study”. *Lecture Notes in Computer Science, pp. 276-289.*

Muratet, M., Torguet, P., Viallet, F., et al., (2011)., “Experimental feedback on Prog&Play: a serious game for programming practice”. *Computer Graphics Forum, Vol. 30, pp. 61-73.*

Nacke, L., Drachen, A., and Gobel, S., (2010). "Methods for Evaluating Gameplay Experience in a Serious Gaming Context". *International Journal of Computer Science in Sport, Vol. 9, Issue. 2, pp. 1-12.*

Navimipour, J.N., and Zareie, B., (2015). "A model for assessing the impact of e-learning systems on employees' satisfaction". *Computers in Human Behavior Vol. 53, pp. 475-485.*

Nguyen, N. T., (2015). "Motivational Effect of Web-Based Simulation Game in Teaching Operations Management". *Journal of Education and Training Studies Volume 3 No. 2.*

Nielsen, J., (1992). “Finding Usability Problems through Heuristic Evaluation”. *In: Proceedings of the SIGCHI Conference on Human factors in Computing Systems: 373- 380. Monterey: ACM Press.*

Nielsen, J. and Mack, R. L., (1994). “Usability Inspection Methods”. *John Wiley & Sons, New York, NY.*

Nielsen, J., (1993). "Usability Engineering". *Morgan Kaufmann Publishers Inc. San Francisco, CA, USA.*

O'Kelly, J. and Gibson, J. P., (2006), "RoboCode & Problem-Based Learning: A non-prescriptive approach to teaching programming". *ACM SIGCSE Bulletin, Vol. 38, Issue. 3, pp. 217-221.*

Olsen, T., Procci, K. and Bowers, C., (2011). “Serious games usability testing: how to ensure proper usability, playability, and effectiveness”. *Lecture Notes in Computer Science, Vol. 6770, pp. 625-634.*

Ornelas Marques, F., Marques, M.T. (2012). “No problem? No research, little learning ... big problem!”. *Systemics, Cybernetics and Informatics, Vol. 10, Issue. 3, pp. 60–62.*

Ouahbi, I., Kaddari, F., Darhmaoui, H., Elachqar, A. and Lahmine, S., (2015). "Learning Basic Programming Concepts by Creating Games with Scratch Programming Environment". *Social and Behavioral Sciences, 191, pp. 1479-1482.*

Papadopoulos, Y., Tegos, S., (2012). "Using microworlds to introduce programming to novices". *Proceedings of the 2012 16th Panhellenic Conference on Informatics, pp. 180–185.*

Papastergiou, M., (2009). "Digital game-based learning in high school computer science education: impact on educational effectiveness and student motivation". *Computers & Education Vol. 52, pp. 1–12.*

Paras, B. and Bizzocchi, J. (2005). "Game, Motivation, and Effective Learning: An Integrated Model for Educational Game Design". *Proceedings of DiGRA 2005 – the Digital Games Research Association's 2nd International Conference.*

Pascarella, E. T. and Terenzini, P.T. (2005). "How college affects students: A third decade of research". *San Francisco: John Wiley & Sons, Inc.*

Pavlas, D., Jentsch, F., Salas, E., Fiore, S. M. and Sims, V., (2012). "The Play Experience Scale: development and validation of a measure of play". *The Journal of the Human Factors and Ergonomics Society, Vol. 54, Issue. 2, pp. 214-225.*

Petrasova, A., Cza, S., Chalmers, A., Farrer, J. and Wolke, D., (2010). "The playability evaluation of virtual baby feeding application". *In VS-Games 2010-Proceedings of the 2nd International Conference on Games and Virtual Worlds for Serious Applications, pp. 95-100.*

Petri, G. (2018). "A Method for the Evaluation of the Quality of Games for Computing Education" (Doctoral dissertation). *Federal University of Santa Catarina.*

Phuong, D. D., Harada, F., Takada, H., and Shimakawa, H., (2008). "Collaborative Learning Environment with Convincing Opinions for Novice Programmers". *In Proc. 5th WSEAS/IASME International Conference on Engineering Education, pp. 88-94.*

Piccoli, G., Ahmad, R. and Ives, B., (2001). "Web-based virtual learning environments: a research framework and a preliminary assessment of effectiveness in basic IT skill training". *MIS Quarterly, Vol. 25, Issue. 4, pp. 401–426.*

Pinelle, D., Wond, N., and Stach, T., (2008). "Heuristic Evaluation for Games: Usability Principles for Video Game Design". *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1453-1462.*

PlayGen, (2016). “Serious Games for Government & Politics”. *Available at: http://playgen.com/serious-games-for-government-and-politics/*

Porter, L., Guzdial, M., McDowell, C. and Simon, B., (2013). “Success in Introductory Programming: What Works?”. Communications of the ACM, Vol. 56, Issue. 8, pp. 34-36.

Porter, L., Lee, C., Simon, B., and Guzdial, M., (2017). “Preparing tomorrow’s faculty to address challenges in teaching computer science”. *Communications of the ACM, Vol. 60, Issue. 5, pp. 25-27.*

Pourabdollahiana, B., Taischa, M. and Kergaa, E., (2012). "Serious Games in Manufacturing Education: Evaluation of Learners' Engagement". *Procedia Computer Science, Vol. 15, pp. 256-265.*

Preece, J., Rogers, Y. and Sharp, H., (2002). “Interaction Design: Beyond Human-Computer Interaction”. *New York: John Wiley & Sons.*

Quick, V., Corda, K., Chamberlin, B., et al., (2013). “Ninja kitchen to the rescue: evaluation of a food safety education game for middle school youth”. *British Food Journal, Vol.115, Issue. 5, pp. 686-699.*

Quinn, N. C., (2005). “The Seven Step Program for eLearning Improvement”. *Available at: https://quinnovation.com/EnhancedIDWP.pdf.*

Ragonis, N. and Ben-Ari, M., (2005). “A long-Term Investigation of the Comprehension by Novices”. *Computer Science Education, Vol. 15, No. 3, pp. 203-221.*

Rajeev, S. and Sharma, S., (2018). "Educational Game-Theme Based Instructional Module for Teaching Introductory Programming". *44th Annual Conference of the IEEE Industrial Electronics Society, pp. 3039-3044.*

Ranjit, K., (1996). "Research Methodology a step-by-step guide for beginners". *3rd edition, SAGE Publishing.*

Razak, A., Connolly, T. and Hainey, T., (2012). "Teachers’ Views on the Approach of Digital Games-Based Learning within the Curriculum for Excellence". *International Journal of Game-Based Learning Vol. 2, pp. 33-51.*

Rebolledo-Mendez, G., Avramides, K., de Freitas, S. and Memarzia, K., (2009). “Societal impact of a serious game on raising public awareness: the case of FloodSim”. *In Sandbox 2009-Proceedings of the 2009 ACM SIGGRAPH Symposium on Video Games, pp. 15-22.*

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E, Siver, J., Silverman, B., Kafai, Y. (2009). "Scratch: programming for all". *Communications of the ACM, Vol. 52, pp. 60–67.*

Ricci, K.E., (1994). "The use of computer-based videogames in knowledge acquisition and retention*". Journal of Interactive Instruction Development, Vol. 7, Issue. 1, pp. 17–22.*

Roberts, J. and Styron, R., (2011). "Student Satisfaction and Persistence: Factors Vital to student retention," *Research in Higher Education Journal, vol. 6, pp. 1-18*.

Robertson, J. and Howells, C., (2008). "Computer game design: Opportunities for successful learning". *Computers & Education, Vol. 50, Issue. 2, pp. 559-578.*

Robins, A., Rountree, J. and Rountree, N., (2003). "Learning and Teaching Programming: A Review and Discussion". *Journal of Computer Science Education, Vol. 13, Issue. 2, pp. 137-172.*

Rodríguez-Cerezo, D., Sarasa-Cabezuelo, A., Gomez-Albarran, M. and Sierra, J., (2014). "Serious games in tertiary education: a case study concerning the comprehension of basic concepts in computer language implementation courses". *Computers in Human Behavior, Vol. 31, pp. 558-570.*

Rondon, S., Sassi, C. F., and Andrade, F. R. C., (2013). "Computer game-based and traditional learning method: a comparison regarding students' knowledge retention". *BMX Medical Education, Vol. 13.*

Rosenberg, J., and Kölling, M., (1997). "Testing Object-Oriented Programs: Making it Simple". *ACM SIGCSE Bulletin, Vol. 29, Issue. 1, pp. 77-81.*

Ross, M. J., (2002). "Guiding Students Through Programming Puzzles: Value and Examples of Java Game Assignments". *ACM SIGCSE Bulletin, Vol. 34, Issue. 4, pp. 94-98.*

Rossiou, E. and Papadakis, S., (2007). "Educational games in higher education: a case study in teaching", *Proc. of the International Conference on Education in a Changing Environment, pp. 149-157*.

Rozali, N. F. and Zaid, N. M., (2017). "Code Puzzle: ActionScript 2.0 Learning Application Based on Problem Based Learning Approach". *6th ICT International Student Project Conference (ICT-ISPC).*

Rubin, B., Fernandes, R. and Avgerinou, M. D., (2013). "The effects of technology on the Community of Inquiry and satisfaction with online courses". *The Internet and Higher Education, Vol. 17, pp. 48–57.*

Sabri, H., Cowana, B., Kapralosa, B., Porte, M., Backsteinc, D. and Dubrowskiee, A., (2010). “Serious games for knee replacement surgery procedure education and training”. *In Proceedings of the World Conference on Educational Sciences (WCES ’10), Vol. 2, pp. 3483-3488.*

Sanders, D. and Dorn, B., (2003). "Classroom experience with Jeroo". *Journal of Computing Sciences in Colleges, Vol. 18, Issue. 4, pp. 308-316.*

Sardi, L., Idri, A. and Fernandez-Akeman, J. L., (2017). "A systematic review of gamification in e-Health". *Journal of Biomedical Informatics, Vol. 71, pp. 31-48.*

Savi, R., Gresse Von Wangenheim, C. and Borgatto, F. A., (2011). “Model for the Evaluation of Educational Games for Teaching Software Engineering”. *25th Brazilian Symposium on Software Engineering. pp. 194–203.*

Savi, R., Wangenheim, C. G., Borgatto, A. F., Buglione, L. and Ulbricht, V. R., (2012). “MEEGA – A Model for the Evaluation of Games for Teaching Software Engineering”. *Technical Report INCoD/GQS.01.2012. E. Brazilian Institute for Digital Convergence, Department of Informatics and Statistics, Federal University of Santa Catarina, Brazil.*

Schumann, P., Anderson, H. P., Scott, W. T. and Lawton, L., (2001). "A Framework For Evaluating Simulations As Educational Tools". *Developments in Business Simulation and Experiential Learning, Vol. 28, pp. 215-220.*

Sela, E. and Sivan, Y. Y., (2009). “Enterprise e-learning success Factors: An analysis of practitioners' perspective”. *Interdisciplinary Journal of E-learning and Learning Objects, Vol. 5, pp. 335-343.*

Seng, W. Y. and Yatim, M. H. M. (2014). "Computer Game as Learning and Teaching Tool For Object Oriented Programming in Higher Education Institution". *Procedia - Social and Behavioral Sciences 123, pp. 215- 224.*

Seyal, A. H., Mey, Y. S., Matusin, M. H., Norzainah, H. H. S. and AbdulRahman, A., (2015). “Understanding Students Learning Style and Their Performance in Computer Programming Course: Evidence from Bruneian Technical Institution of Higher Learning”. International Journal of Computer Theory and Engineering, Vol. 7, No. 3, pp. 241-247.

Shabalina, O., Malliarakis, C., Tomos, F. and Mozelius, P., (2017). “Game-Based Learning for Learning to Program: From Learning Through Play to Learning Through Game Development”. *European Conference on Games Based Learning, Vol. 11.*

Sheng, S., Magnien, B., Kumaraguru, P., Acquisti, A., Cranor, L. F., Hong, J. and Nunge, E., (2007). “Anti-Phishing Phil: the design and evaluation of a game

that teaches people not to fall for phish", *Proc. of the 3rd Symposium on Usable Privacy and Security.*

Shernoff, D. J. Csikszentmihalyi, M. Schneider, B. and Shernoff, E. S., (2014). "Student engagement in high school classrooms from the perspective of flow theory". *Applications of Flow in Human Development and Education: The Collected Works of Mihaly Csikszentmihalyi, pp. 475–494.*

Singh, Y. K., (2006). "Fundamental of Research Methodology and Statistics". *New Age International Publishers.*

Sitzmann, T., (2011). "A Meta-Analytic Examination of The Instructional Effectiveness of Computer-Based Simulation Games". *Personnel Psychology, Vol. 64, pp. 489–528.*

Sizmann, T., (2011). "A meta-analytic examination of the instructional effectiveness of computer -based simulation games". *Personnel Psychology, Vol. 64, pp. 489-528.*

Sloan, R. H. and Troy, P., (2008). "CS 0.5: A better approach to introductory computer science for majors". *ACM SIGCSE Bulletin - SIGCSE 08, Vol. 40, Issue. 1, pp. 271-275.*

Smith, P., (2013). "Serious Games 101". STO-EN-MSG-115.

Song, S. and Lee, J., (2007). "Key factors of heuristic evaluation for game design: Towards massively multi-player online role-playing game". *International Journal of Human-Computer Studies archive, Vol. 65, Issue. 8, pp. 709-723.*

Sprint. G. and Cook. D., (2015). "Enhancing the CS1 Student Experience with Gamigication". *Integrated STEM Education Conference (ISEC), pp. 94-99.*

Squire, K., (2002). "Cultural framing of computer/video games". *Game Studies, Vol. 2. Available at: http://www.gamestudies.org/0102/squire/ Last accessed 21/November/ 2017.*

Ssemugabi, S. and de Villiers, R., (2007). "A comparative study of two usability evaluation methods using a web-based e-learning application". *In Proceedings of the 2007 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries, pp. 132-142.*

Stokes, B., (2005). "Video games have changed: time to consider, serious games". *The Development Education Journal, Vol. 11, No. 108.*

Subhash, S. and Cudney, E. A., (2018). "Gamified learning in higher education: A systematic review of the literature". *Computers in Human Behavior, Vol. 87, pp. 192-206.*

Sullivan, G. M. and Feinn., (2012). "Using Effect Size-or Why the P Value is not Enough". *Journal of Graduate Medical Education, pp. 279-282.*

Sun, P., Tsai, R., Finger, G. et al., (2008). "What drives a successful e-Learning? An empirical investigation of the critical factors influencing learner satisfaction". *Computers & Education, Vol. 50, Issue. 4, pp. 1183-1202.*

Susi, T., Johannesson, M. and Backlund, P., (2007). "Serious Games- An Overview". University of Skövde. *Available at: http://www.diva-portal.org/smash/get/diva2:2416/FULLTEXT01.pdf.*

Swain, C., (2007). "Designing Games to Effect Social Change". *Proceedings of DiGRA 2007 Conference.*

Sweetser, P. and Wyeth, P., (2005). "GameFlow: a Model for Evaluating Player Enjoyment in Games". *Computers in Entertainment, vol.3, Issue. 3.*

Syberfeldt, A. and Syberfeldt, S., (2010). "A serious game for understanding artificial intelligence in production optimization". *Proc. of the Conf. on Computational Intelligence and Games, pp. 443-449.*

Talebian, S., Mohammadi, M. H. and Rezvanfar, A., (2014). "Information and communication technology (ICT) in higher education: advantages, disadvantages, conveniences and limitations of applying e-learning to agricultural students in Iran". *Social and Behavioral Sciences, Vol. 152, pp. 300-305.*

Talton, O. J., Peterson, L. D., Kamin, S., Israel, D., and Al-Muhtadi, J., (2006). "Scavenger Hunt: Computer Science Retention Through Orientation" *ACM SIGCSE Bulletin, Vol. 38, Issue. 11, pp. 443-447.*

Tang, S., Hanneghan, M. and El Rhalibi, A., (2007a). "Pedagogy Elements, Components and Structures for Serious Games Authoring Environments". 5th International Game Design and Technology Workshop, pp. 26-34.

Tang, S., Hanneghan, M. and El Rhalibi, A., (2007b). "Describing Games for Learning: Terms, Scope and Learning Approaches". The Fifth Annual *International Conference in Computer Game Design and Technology*.

Teo, T., (2014). "Preservice teachers' satisfaction with e-learning". *Social Behavior and Personality: An International Journal, Vol. 42, Issue. 1, pp. 3–6.*

Thompson, D., Baranowski, T., Buday, R., et al., (2010). “Serious video games for health: how behavioral science guided the development of a serious video game”. *Simulation Gaming, Vol. 41, Issue. 4, pp. 587–606.*

Thurmond, A., Wambach, K. and Connors, R., (2002). “Evaluation of student satisfaction: determining the impact of a web-based environment by controlling for student characteristics”. *The American Journal of Distance Education, Vol. 16, Issue. 3, pp. 169–189.*

Tîrziu, M. A. and Vrabie, C., (2015). “Education 2.0: E-Learning Methods” *Procedia - Social and Behavioral Sciences, Vol. 186, pp. 376–380.*

Tobias, S. and Fletcher, J. D., (2007). “What research has to say about designing computer games for learning”. *Educational Technology, Vol. 47, pp. 20–29.*

Topalli, D. and Cagitaly, N. E., (2018). “Improving programming skills in engineering education through problem-based game projects with Scratch”. *Computers & Education, Vol, 120, pp. 64-74.*

Torrente, J., Moreno-Ger, P., Fernandez-Manjon, B. et al., (2009). “Game-like simulations for online adaptive learning: a case study”. *Lecture Notes in Computer Science, Vol. 5670, pp. 162-173.*

Triplett, D., (2002). “An Interview with Robocode creator Mat Nelson”. *Developer Works, IBM’s Resource for Developers.*

Tsalikidis, K. and Pavlidis, G., (2016). "JLegends Online game to train programming skills". *7th International Conference on Information, Intelligence, Systems & Applications (IISA).*

Uchida, S. and Shima, K., (2004). "An Experiment of Evaluating Software Understandability". *Systemics, Cybernetics and Informatics, Vol. 2, Number. 5, pp. 7-11.*

Ulicsak, M., (2010). "Games in Education: Serious Games". *Available at: https://www.nfer.ac.uk/publications/FUTL60/FUTL60.pdf*

United Nations, (2018). “World Economic Situation and Prospects”. *United Nations.*

Vogel, J., Vogel, D. S., Cannon-Bowers, J., Bowers, C. A., Muse, K. and Wright, M., (2006). “Computer gaming and interactive simulations for learning: A meta-analysis”. *Journal of Educational Computing Research, Vol. 34, pp. 229–243.*

Wallace, S., McCartney, R. and Russell, I., (2010). “Games and machine learning a powerful combination in an artificial intelligence course”. *Computer Science Education, Vol. 20, pp. 17-36.*

Wang, T. H., (2014). “Developing an assessment-centered e-Learning system for improving student learning effectiveness”. *Computers & Education, Vol. 73, pp. 189–203.*

Wangenheim, G. C., Savi, R. and Borgatto, F. A., (2012). “DELIVER! – An educational game for teaching Earned Value Management in computing courses”. *Information and Software Technology, Vol. 54, pp. 286-298.*

Watson, C. and Li, F. W. B., (2014). “Failure Rates in Introductory Programming Revisited”. *Proceedings of the 2014 conference on Innovation technology in computer science education (ITiCSE'14), pp. 39-44.*

Watson, S. and Lipford, H. R., (2019). "Motivating Students Beyond Course Requirements with a Serious Game". *50th ACM Technical Symposium on Computer Science Education, pp. 211-217.*

Wein, J. Kourtchikov, K., Cheng, Y., Gutierez, R., Khmelichek, R. and Topol, M., (2009). “Virtualized games for teaching about distributed systems”. *Proc. of the 40th ACM Technical Symposium on Computer Science Education, pp. 246–250.*

Wilson, D., Jenkins, J., Twyman, N., Jensen, M., Valacich, J., Dunbar, N., Wilson, S., Miller, C., Adame, B., Lee, Y., Burgoon, J. and Nunamaker, J., (2016). “Serious Games: An Evaluation Framework and Case Study”. *49th Hawaii International Conference on System Sciences, pp. 638-647.*

Wolz, U., Barnes, T., Parberry, I. and Wick, M., (2006). “Digital gaming as a vehicle for learning”. *Proceedings of the 37th SIGCSE technical symposium on Computer science education, Vol. 38, Issue. 1, pp. 394-395.*

Woodfield, R., (2014). “Undergraduate retention and attainment across the disciplines*”. York: Higher Education Academy.* https://www.heacademy.ac.uk/resource/issues-retention-and-attainment-computer-science.

Wouters, P., van der Spek, E. and van Oostendorp, H., (2009). "Current practices in serious game research: A review from a learning outcomes perspective". *Games-based learning advancements for multisensory human computer interfaces: Techniques and effective practices, pp. 232-255.*

Wrzesien, M. and Alcaniz, M., (2010). “Learning in serious virtual worlds: evaluation of learning effectiveness and appeal to students in the E-Junior project”. *Computers & Education, Vol. 55, pp. 178–187.*

Xu, D., Huang, W. W., Wang, H. and Heales, J., (2014). “Enhancing e-learning effectiveness using an intelligent agent-supported personalized virtual learning environment: an empirical investigation”. *Information and Management, Vol. 51, Issue. 4, pp. 430-440.*

Xu, Y., Johnson, M. P., Moore, A. C., Brewer, S. R. and Takayama, J., (2013). "SGSEAM: assessing serious game frameworks from a stakeholder experience perspective". *In Proceedings of the First International Conference on Gameful Design, Research, and Applications, pp. 75-78.*

Yan, L., (2009) “Teaching object-oriented programming with games”. *ITNG 2009 6th international conference on information technology: New generations, pp. 969-974.*

Yassine, A., Chenouni, D., Berrada, D. and Tahiri, A., (2017). "A Serious Game for Learning C Programming Language Concepts Using Solo Taxonomy". *iJET, Vol. 12, No. 3, pp. 110-127.*

Yusoff, A., Crowder, R. and Gilbert, L., (2010). "Validation of Serious Games Attributes Using the Technology Acceptance Model". *In 2010 Second International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES)*.

Zainol Abidin, H. and Kamaru Zaman, F. H., (2017). “Students’ Perceptions on Game-based Classroom Response System in a Computer Programming Course”. *IEEE 9th International Conference on Engineering Education.*

Zapusek, M. and Rugelj, J. (2013). "Learning programming with serious games". *EAI Endorsed Transactions on Game Based Learning, Vol. 13.*

Zareie, B., and Navimipour, J. N., (2016). “The effect of electronic learning systems on the employee's commitment”. *The International Journal of Management Education Vol. 14, pp. 167-175.*

Zhang, J., Caldwell, R. E. and Smith, E., (2013). "Learning the Concept of Java Inheritance in a Game". *The 18th International Conference on Computer Games (CGAMES), pp. 212-216.*

Zhao, D., Chis, A. E., Muntean, G. M. and Muntean, C. H., (2018). “A Large-Scale Pilot Study on Game-Based Learning and Blended Learning Methodologies in Undergraduate Programming Courses”. *10th International Conference on Education and New Learning Technologies*.

Zhao, D., Muntean, C. H. and Muntean, G., (2019). “The Restaurant Game: a NEWTON PROJECT Serious Game for C Programming Courses”. *Proceedings of Society for Information Technology & Teacher Education International Conference, pp. 2121-2128.*

Zweben, S. and Bizot. B., (2017). “2017 CRA Taulbee Survey Another Year of Record Undergrad Enrollment; Doctoral Degree Production Steady While Master’s Production Rises Again”. *Computing Research Association. Available at:* https://cra.org/resources/taulbee-survey/

Zweben, S., (2008). "2006-2007 Taulbee Survey", *Computing Research News, Volume 20, No 3.*

Zweben, S., (2009). "2007-2008 Taulbee Survey: Upward Trend in Undergraduate CS Enrollment; Doctoral Production Continues at Peak Levels". Computing Research *News. Available at:* https://cra.org/resources/taulbee-survey/

Zweben, S., (2010). "2008-2009 Taulbee Survey Undergraduate CS Enrollment Continues Rising; Doctoral Production Drops". *Computing Research News. Available at:* https://cra.org/resources/taulbee-survey/.

Zyda, M., (2005). “From visual simulation to virtual reality to games”. Journal *Computer, Vol. 38, Issue. 9, pp. 25-32.*