

Examining the Impact of Game-Based Learning on Student
Engagement and Performance in an Introductory Computer
Programming Course at the University of the Southern
Caribbean (USC).



Fayola St. Bernard

January 2022

This thesis is submitted in partial fulfilment of the requirements for the degree
of Doctor of Philosophy.

Department of Educational Research,
Lancaster University, UK.

Declaration

This thesis results entirely from my own work and has not been offered previously for any other degree or diploma. I declare that the word length of this thesis is 44, 092 words and conforms to the permitted maximum length.

Fayola St. Bernard.

Abstract

Background: At the University of the Southern Caribbean (USC) students often struggle with learning programming. Because of this struggle, they often become disengaged with the programming courses, with some transferring to other degree programmes or withdrawing from the programme. While several strategies have been used to ensure that students can problem-solve, design, and develop coded solutions, it has not been enough to alleviate the issues. Game-based learning (GBL) emerged as a possible strategy that can potentially help students develop these skills while keeping them engaged with the course content.

Aim: Implementing such a strategy within the department requires evidence that it can be an effective technique for teaching and learning programming. Therefore, the aim of this study is to evaluate the impact of GBL on student engagement and overall performance in an introductory programming course.

Method: The research was designed as a deductive exploratory single case study research strategy and method. It approaches the aims and objectives from a pragmatic perspective, and as a result, uses a mixed methodological approach to data collection and analysis.

Findings: The findings show that while GBL does not alleviate the common negative reactions to learning programming, it does provide a learning environment engaging enough for students to overlook these. This results in students having an enhanced perception of the knowledge and improved performance.

Propositions: In implementing GBL in other programming courses, some features that are potentially the most impactful on students learning are immediate feedback, freedom to fail, user interface, code without limitations, and a visual representation of progress.

Table of Contents

Abstract	ii
Dedication	vii
Acknowledgements.....	viii
List of Figures and Tables.....	ix
List of Figures	ix
List of Tables.....	x
Introduction	1
Statement of the Problem	1
Previously Considered Teaching Strategies	4
<i>Robotics</i>	4
<i>Syntax-Free Approach</i>	5
<i>Appropriateness of Programming Language</i>	6
Games in Education.....	7
Focus and Scope of Research	7
Aims and Objectives of the Research	9
Research Questions.....	9
Structure of Thesis	11
Literature Review.....	14
Challenges and Issues in Learning Computer Programming	15
<i>Deep and Surface Learning</i>	16
Issues and Trends in Teaching Programming	17

<i>Trends in Teaching Programming</i>	18
Games in Education.....	20
Defining Gamification, Serious Games and Game-Based Learning	21
<i>Defining Gamification</i>	21
<i>Defining Serious Games and Game-Based Learning</i>	22
Games in Learning Computer Programming.....	24
<i>Engagement and Performance</i>	42
Learning Approaches in Current Research.....	47
<i>Collaborative Learning</i>	47
<i>Problem-Based Learning (PBL)</i>	48
<i>Creative-Based Learning</i>	49
Issues in Current Research	50
<i>Theoretical Underpinning for Games in Learning</i>	50
<i>Influence of Specific Game Attributes</i>	55
<i>Impact on Learning Outcomes</i>	57
Critique of Games in Education.....	59
Summary.....	61
Research Strategy and Design	63
Research Strategy	63
Case Study Design.....	66
Theoretical Framework.....	68
<i>Theoretical Frameworks that focused on Educational Games</i>	68
<i>Application of the Gamification for Student Engagement Framework</i>	76
Research Propositions and Questions	86
Case Study Approach	87

Data Collection.....	88
<i>Participation</i>	88
<i>Data Collection Process</i>	89
Data Analysis	92
<i>Paired Samples t-Test</i>	93
<i>Thematic Analysis</i>	94
Legitimation rather than Validity	95
Ethical Considerations.....	99
Findings.....	101
Prior Knowledge in Programming	102
Game Attributes.....	102
<i>Action Language</i>	102
<i>Assessment</i>	103
<i>Challenge</i>	104
<i>Game Fiction</i>	105
<i>Rules/Goals</i>	106
Student Engagement.....	106
<i>Engagement Themes</i>	106
<i>Linking Game Attributes to Engagement Themes</i>	108
Motivation.....	118
Perception of Knowledge	119
<i>Pre-Evaluation of Knowledge</i>	120
<i>Mid-Semester Reflection</i>	122
<i>Post-Evaluation of Knowledge</i>	122
Performance.....	125
<i>Pre-Test Scores</i>	125

<i>Competency in Programming</i>	127
<i>Post-Test Scores</i>	127
Linking Game Attributes, Engagement and Learning Outcomes Criteria	129
<i>Syntactical Knowledge</i>	129
<i>Problem-Solving and Design</i>	130
Summary of Findings.....	133
Discussion and Conclusions	137
Statement of the Problem	137
Answering Research Question 1.....	138
Answering Research Question 2	142
Answering Research Question 3	144
Answering Research Question 4	147
Answering Research Question 5	150
Propositions and Conclusions.....	152
<i>Proposition 1</i>	153
<i>Proposition 2</i>	153
<i>Proposition 3</i>	154
<i>Proposition 4</i>	154
Opportunities for Future Research	156
Limitations of the Research	156
Bibliography	158

Dedication

Michael St. Bernard

1961-2021

I dedicate this thesis to my father - my greatest love, comforter, and supporter. Through my
indescribable pain and never-ending tears, I finished this for you!

Acknowledgements

My God and my faith gave me the greatest comfort, strength, and purpose to keep persevering when it seemed overwhelming. For it all, I give praise!

I would like to express my deepest and heartfelt thanks to my supervisor - Professor Malcolm Tight; without his guidance and support, none of this would be possible. It has been an absolute pleasure and treasure having worked under Professor Tight.

Thank you to Dianne Auguste for taking time during her holidays to proofread this thesis. My cousin, but more a brother, Reeza Ramgattie. From childhood to adulthood, he has always been a part of every moment of my life. Thanks for putting up with me every week, when I took over your office as you worked, just because I needed a change of environment. My entire family deserves my thanks for all their love and unwavering support. We are a large, close-knit family and I love you all, but I cannot name you all. However, there are some that I would like to specifically mention: Dora and Lloyd Isaacs, Carmen Rooplal, Kelvin Rooplal, Inshan Rampersad, Wendy Rooplal, and Joyce Rooplal. The elite three - my mum, Annie St. Bernard; my sister, my 'hoss', Felisha St. Bernard; and my aunt, Clarise Rooplal.

I must mention Dr. Jerome Teelucksingh, my previous lecturer turned friend. I thank him for all the times he has encouraged and reminded me to stay focused. My study buddy Robert Tootell; his support, encouragement and our monthly progress meetings have kept me motivated and focused. His support throughout this process was more than I could ask for, and one that I appreciated. And my two best friends – my council members - Keron Venus and Alvin Ramsamooj, the best cheerleaders anyone can ask for!

List of Figures and Tables

List of Figures

Figure 1 Block-Based Programming vs. Syntax Programming.....	5
Figure 2 Google Trends Analysis - Game-Based Learning.....	19
Figure 3 Google Trends Analysis - Gamification.....	19
Figure 4 Google Trends Analysis - Gamification in Trinidad and Tobago.....	20
Figure 5 Bloom's Taxonomy taken from Armstrong (2010)	59
Figure 6 Application of Case-Study Schematics, adapted from Rosenberg & Yates (2007).....	67
Figure 7 Input-Process-Output Model (Garris et al., 2002).....	69
Figure 8 Theory of Gamified Learning (Landers, 2014)	70
Figure 9 Gamification for Student Engagement Framework - Application of Student Engagement Framework (top line, Kahu 2013) and Theory of Gamified Learning (bottom line, Landers 2014)	73
Figure 10 Student Engagement Framework (Kahu, 2013)	75
Figure 11 Application of the Gamification of Student Engagement Framework	76
Figure 12 Ozaria - Action Language	82
Figure 13 Ozaria – Assessment.....	82
Figure 14 Ozaria - Game Fiction.....	83
Figure 15 Ozaria - Rules and Goals.....	83
Figure 16 Thematic Analysis Process (Braun & Clarke, 2006)	94
Figure 17 Pre-Evaluation of the Perception of Programming Knowledge	120
Figure 18 Post-Evaluation of the Perception of Programming Knowledge	123
Figure 19 Paired Sample t-Test - Perception of Knowledge.....	125
Figure 20 Pre-Test Scores	126
Figure 21 Paired Samples t-Test - Pre/Post-Test Scores.....	127
Figure 22 Post Test Scores	128

Figure 23 Linking Learning Programming to Bloom's Taxonomy	146
---	-----

List of Tables

Table 1 Game Elements and its Definition.....	22
Table 2 Game Attributes as defined by Bedwell et al. (2012)	24
Table 3 Educational Games in Computer Programming - Review of Literature.....	26
Table 4 Introduction to Computer Programming Course Topics.....	77
Table 5 Aligning Ozaria Modules to Programming Topics within the Introduction to Computer Programming Course.....	80
Table 6 Game Attributes Implemented by Ozaria	81
Table 7 Affective, Behavioural, and Cognitive Engagement Indicators as outlined by Bond & Bedenlier (2019)	85
Table 8 Prior Knowledge and Skills in Programming Questionnaire.....	90
Table 9 Rating Understanding in Programming Concepts Questionnaire.....	90
Table 10 Mid-Semester Self-Assessment Questionnaire.....	90
Table 11 Reflections on Learning Experience Questionnaire	90
Table 12 Reflections on Specific Game Attributes Questionnaire	91
Table 13 Pre-Test Reflection Questionnaire - Students with Prior Knowledge.....	91
Table 14 Reflection on Engagement and Game Experience Questionnaire	92
Table 15 Code for Categorical Responses	93
Table 16 Frequency of Engagement Themes	108
Table 17 Game Attributes and the Engagement Themes it Influenced	109
Table 18 Frequency for Pre-Evaluation of Programming Knowledge.....	120
Table 19 Frequency for Post-Evaluation of Programming Knowledge	122

Chapter One

Introduction

This chapter provides an overview of the background and details the problem and rationale for undertaking this research. It begins by outlining the problem faced by the faculty of the University of the Southern Caribbean (USC), Trinidad and Tobago in the Department of Computing, Mathematics and Technology. The chapter then briefly introduces previously considered strategies to enhance teaching and learning programming within the department. It then defines and justifies the chosen method that will be used for this research. It follows this by briefly explaining the methods that guided this research, including research methodology and method, theoretical framework, and research questions. The chapter concludes by explaining the structure of the thesis, detailing each item contained within.

Statement of the Problem

The Department of Computing, Mathematics and Technology at University of the Southern Caribbean (USC), Trinidad and Tobago, offers students a software emphasis that prepares them for the field of software, mobile and web development. As a lecturer in the department, I teach several advanced level programming courses at the third and final year of the program. As the course progresses, students are introduced to many programming languages and topics that build in complexity.

Programming is dynamic in nature and requires students to integrate the syntax of a programming language, programming techniques, and algorithms in problem-solving and critical thinking skills to formulate solutions (Krpan et al., 2014; Thota & Whitfield, 2010). While students are known to have difficulties with using the syntax of a programming language, problem-solving is the most significant challenge faced when learning programming. Even if they know the syntax and understand the programming concepts, they still commonly face difficulties in solving problems, most often because they have fragmented knowledge of

programming patterns, they lack the knowledge of programming strategies, and fail to comprehend, write, and debug code (Qian & Lehman, 2017). The challenge in programming is not only in acquiring these necessary skills, but also in the emotions it can foster during the learning programming. Programming by its very nature can evoke several negative emotions, all of which usually stem from frustration. This frustration can come from course difficulties, frequent repetition of learning tasks, lack of support, lack of computer literacy or self-teaching skills, difficulty in identifying syntax errors to the misunderstanding features of the programming language, all of which can lead to rapid withdrawal, aversion to programming, and sense of failure (Bubica et al., 2014).

This is too often the experience in the advanced programming courses offered by the department. Students in these classes are solely responsible for learning new programming languages or frameworks on their own while learning advanced programming topics such as object-oriented programming and data structures and algorithms. Within my six years at the institution, I have seen students struggle the most with problem-solving and critical thinking when developing solutions. This often leads to them becoming despondent and frustrated with the learning process. While students do struggle to learn programming, a contributing factor to their struggle is the teaching methods adopted by the faculty. It is widely known that the traditional approach of using lectures and textbooks is largely ineffective for teaching programming. While research identifies teaching by doing as a more viable strategy, this often leads lecturers to focus more on the syntax of the programming language and fail to bridge the gap between the problem and its corresponding algorithmic solution (Cheah, 2020). Despite knowing the challenges in teaching programming, developing teaching strategies is difficult for a subject area as dynamic as programming because programming tools and its content are always changing making it harder to orchestrate. Therefore, any strategy adopted must entail a combination of facilitating the development of a mental model of what computers are, how they run code and how they interpret, trace and debug programs (Papadakis & Kalogiannakis, 2019).

At USC, the department mostly adopts the learning through a demonstration approach to teaching, with a combination of lectures and lab sessions that integrate theory and practical sessions designed as case studies and projects. While these efforts saw the movement away from purely demonstrative lecture-based teaching strategies, it is not without its limitations. With this teaching strategy, there is no guarantee that newly enrolled students have the capacity to develop the skills necessary at the early stages of learning programming. This is because new students range from having no previous knowledge or experience in programming to having some background, either attained informally or via formal study in a pre-university context.

This of course sparked a discussion among the faculty on students' aptitude for programming, and strictly adhering to requirements when enrolling students in the course. Considering aptitude before enrolling in introductory programming course, may help educators select potential students who are more likely to succeed while simultaneously avoid wasting time and effort on those who are unlikely to become good programmers (Cutts et al., 2006). However, in turning to empirical papers for evidence of its effectiveness, it showed that results were too varied to derive a consensus on its actual value. For instance, Barlow-Jones et al. (2014) attempted to establish a correlation between problem-solving ability and academic performance in introductory programming courses and found that there was a correlation between students' logical and numerical reasoning, verbal logic, and performance. In another example, Lacher et al. (2017) showed that there was not a strong correlation between aptitude and previous experience of students, noting that many students who reported having no previous experience had high aptitudes and one student with a reasonable level of experience was found to be in the low aptitude category. Also, Holbl et al. (2021) reported that having prior knowledge in programming led to a negative impact on students' self-esteem because they dedicated less effort to their studies. Similarly, Smith et al. (2019) found that while there was a significant correlation between prior knowledge and student success, this faded in significance over time.

In the face of such conflicting reports, aptitude was never pursued further by the department, but re-emerges as a topic of discussion from time to time. Despite this, the faculty agrees that a new strategy is needed for introducing newly enrolled students into programming in a way that keeps them engaged enough to overcome the negative perceptions because of the difficulties of programming while learning all the demanding programming concepts.

Previously Considered Teaching Strategies

Initially, some faculty members discussed the possibility of introducing robotics, syntax-free approach to teaching programming, and the use of a specific programming language for introductory courses.

Robotics

The department currently has several robotics kits that were once used to engage students in learning programming. Through educational robotics, students are given the chance to design and develop their own robots, with the purpose of improving their skills in both programming and problem solving, as well as enhance their motivation and engagement (Li et al., 2009; Medeiros et al., 2019). The literature on educational robotics supports the view that it can motivate students to acquire the skills and knowledge necessary to not only learn how the technology works, but also apply the knowledge in a meaningful and exciting way through hands-on learning (Eguchi, 2014).

However, integrating robotics into the existing program would prove to be challenging for two reasons. Firstly, while the department currently has approximately six robotics kits, they are outdated, and the software is not compatible with modern systems. Therefore, replacing them will be a major limitation, as these kits can cost approximately US \$200.00 each (Gage & Murphy, 2003; Shamlian et al., 2006). Secondly, another main challenge will be to develop a curriculum that supports learning and student engagement using robotics. Failure to do this effectively can result in students being unable to make connections to programming topics. For these reasons, robotics was not considered for complete implementation within the department.

Syntax-Free Approach

To ease the learning process, a syntax-free approach was proposed for teaching introductory programming classes. The syntax-free approach separates programming from coding, that is, it eliminates syntax dependency and instead allows students to focus solely on logical reasoning, problem-solving and computational thinking (Fidge & Teague, 2009; Trivedi et al., 2019).

More recently, block-based programming has become increasingly popular as a form of syntax-free approach using visual programming. Unlike a syntactical approach which speaks of coding or using specific programming languages, block-based programming is a visual environment where programming is done by dragging and dropping blocks of code, snapping them together by colour and shape of commands to perform functionalities (Brown et al., 2016; João et al., 2019; Tabet et al., 2016). The stark difference between these two approaches is better shown below (Figure 1).

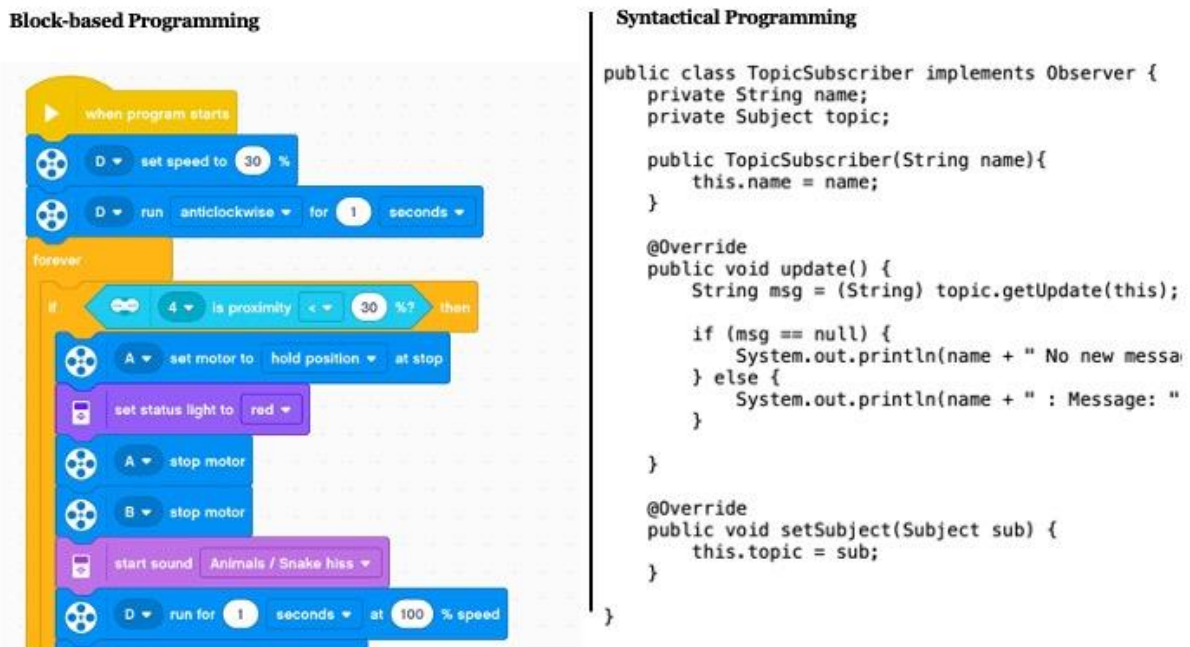


Figure 1 *Block-Based Programming vs. Syntax Programming*

The most widely used block-based applications are Scratch, Alice and MIT App Inventor (L. C. Begosso et al., 2020). These tools allow students the opportunity to program through visual blocks where they can concentrate on what to code rather than the notation that is used to write it (Bau et al., 2017). Using block-based programming is known to increase motivation and student engagement (Trilles & Granell, 2020), confidence (Deng et al., 2020), academic performance (Al-Linjawi & Al-Nuaim, 2010; Durak & Guyer, 2019; Hu et al., 2021; Jiang et al., 2019; Topalli & Cagiltay, 2018), computational thinking (Deng et al., 2020; Kwon et al., 2021; Smith et al., 2020), retention (Rizvi & Humphries, 2012) and a minimisation of programming misconceptions, such as loops (Mladenović et al., 2018). Despite its desired positive impact on learning that can benefit the students at USC, the concern with this was whether they can make the transition from block-based to syntactic programming.

Appropriateness of Programming Language

Since students at USC may still be faced with the common challenges of learning programming, even having gone through the block-based approach, it may not be advisable to abandon teaching a programming language at the introductory level, especially since the aim of learning programming is to develop the knowledge and skills to understand and be competent in a wide range of languages (Ivanovića & Budimac, 2013; Rubiano et al., 2015). When it comes to programming languages, there are over 300 programming languages being used in the industry, with only a fraction of these being taught at higher educational institutions. These include C++, C#, MATLAB, R, Java, and Python (Duffany, 2014; Kanika et al., 2020). Choosing an appropriate language is critical for satisfactory learning results, however, the reality is that programming languages were not developed for learning but to meet real world needs (Duffany, 2014; Minor & Gewali, 2004; Radošević et al., 2009). Thus, it is arguably no longer a discussion of whether programming languages should be taught at the introductory level, but rather how best to teach these languages to ensure a comprehensive theoretical and practical understanding to prepare students for learning other and future languages and environments without

discouraging them (Ivanovića & Budimac, 2013; Kaplan, 2010). A view agreed upon by the faculty of the department. For achieving this, a strategy that has received significant attention from both educators and researchers over the last 5 years is learning through gameplay.

Games in Education

Like games for commercial purpose, games for learning share some of the same design features: it is immersive, challenging, contains goals, is enjoyable, allows for the taking of chances, provides a channel for knowledge acquisition, refines skills, and encourages achievement, all wrapped within problem solving (Gros, 2007; Kinzie & Joseph, 2008). Three different terms, game-based learning (GBL), serious games, and gamification refer to the implementation of games in education. Both game-based learning (GBL) and serious games describe an environment where game content and game play enhance knowledge and skills acquisition, and where game activities involve problem-solving and challenges that provide players with a sense of achievement (Qian & Clark, 2016). GBL is slightly different but related to gamification, in that, gamification implements game-like elements in non-gaming contexts, integrating it into existing topics while GBL aims at fulfilling specific learning outcomes by changing the existing practice of learning, course and its content (Al-Azawi et al., 2016; Alhammad & Moreno, 2018; Deterding et al., 2011; Figueroa-Flores, 2016; Pivec et al., 2003; Plass et al., 2015; Scepanovic et al., 2015).

Focus and Scope of Research

Both gamification and game-based learning boasts of boosting students' knowledge acquisition, problem-solving capabilities, engagement, and motivation. This makes it an appealing strategy, as it addresses the key issues related to learning and teaching programming currently affecting the faculty and students of USC. Despite this, most of the research in GBL focuses on pre-k or k-12 education, with very few empirical studies that investigated higher education and even fewer that explored GBL in the context of Computer Science (Hosseini et al., 2019). Similarly, the research in gamification also suffers from the same lack of empirical

studies in computer science (Gari et al., 2018; Shahid et al., 2019). Justifying the use of this teaching strategy within the department needs empirical evidence, and the current literature lacks the verification to make a claim that it will be effective. Therefore, making a case for or against educational games will require definitive empirical evidence of its impact in the context of USC.

Implementing a new teaching strategy at the advanced level classes would most likely not reverse approximately two years of underdeveloped skills among the students. Dealing with this at the advanced stage is too late, because at this level, students should already have a solid foundation in the core topics of programming languages and techniques, problem-solving and critical thinking skills, and the ability to integrate all of these to develop coded solutions. The early stages of this knowledge must be nurtured from the most introductory level programming course, because it exposes students to the terminology and fundamental concepts of programming, it enlightens them about the best practices of programming, it teaches the process of designing, implementing, testing, and debugging a program, and it enhances students' computational and problem-solving competencies (Kanika et al., 2020).

For this reason, the first-year programming course, Introduction to Computer Programming was chosen to implement GBL. Designing and developing a game platform is outside the scope of this research, and for this reason pre-existing platforms of both gamification and GBL were evaluated for implementation. Ultimately, a GBL platform, Ozaria, was chosen and implemented in the Introduction to Computer Programming course. Despite using a GBL platform, this thesis still included both gamification and GBL literature due to the limited research available for each, and as to not narrow the discussion. From this point, the term "educational games" would be used to refer to both gamification and GBL, unless there is a need to be specific.

Largely, the research was designed as a case study with a mixed methodology approach, with propositions and its associated research questions being framed by the Gamification for

Student Engagement Framework. Although the framework is specific to gamification, its principles were easily transferable to the context of GBL.

Aims and Objectives of the Research

The overall aim of this research is to empirically assess the impact of GBL on student engagement and performance in an introductory computer programming course. The intention of this research is not meant to be generalisable, but to inform my professional practice. Using mixed methods, the research objectives are to:

1. Determine the impact of GBL on students' behaviours and attitudes towards learning programming.
2. Investigate how the changes in behaviours and attitudes affect the students' overall performance in the programming course.
3. Recommend features of game-based learning that are the most impactful on students' behaviours and attitudes for possible implementation in a GBL strategy for teaching programming at USC.

These aims and objectives were designed to also avoid the common gaps in current literature by:

1. Implementing a theoretical framework that frames the presentation and interpretation of the findings.
2. Informing *how* game attributes impacts students' learning using the data collected from the surveys and focus groups and not just reporting on the overall effect of the GBL environment.
3. Identifying the specific game attributes that are effective in both modifying student engagement and achieving the learning outcomes of the course.

Research Questions

This study implements the gamification for student engagement framework, which poses four propositions that can be tested when implementing a GBL strategy for teaching. From these four propositions, research questions were formed.

The first research question explores the impact of game attributes on student engagement. The study aligns with Bond and Bedenlier (2019) outlook on engagement, defining it from the perspective of affective, behavioural, and cognitive. Where affective is the reactions towards learning, behavioural is the involvement in learning activities, and cognitive is the mental effort placed on learning by students. The question identifies the specific engagement states under each category that have been modified by GBL. It goes further than current literature and identifies the specific game attributes of Ozaria that impacted each. The question only identifies the attributes and does not attempt to explain how this impact occurred.

The second research question examines students' perception of knowledge acquisition as a measurable consequence of these changes in engagement states. It uses a pre and post evaluation of students' perception of the programming topics of the course that includes algorithm, syntax errors, logic errors, debugging, looping statements, objects, methods, arguments, engineering cycle, variables, conditional statements, variable arithmetic, nesting statements, while loops, compound conditions, and functions. The purpose of which is to determine whether there is a statistical difference before the class begins and after using GBL.

The third research question evaluates GBL's support of the learning outcomes. In attempting to connect this to the limited available literature, it determines the statistical difference of the performance of the students from a pre and post-test that evaluated students' abilities in the areas of problem-solving, design, and knowledge of syntax. It also connects the findings to Bloom's taxonomy of learning outcomes by detailing the levels with which the findings of this research align. It also identifies the game attributes that were seen to have mostly supported the learning outcomes.

The fourth research question expands on the game attributes that impacted engagement states (research question one) and support of learning outcomes (research question three) by describing how both these influences occurred. Based on qualitative data, it explains the features of the game attributes that were the most impactful on students learning.

Outside the scope of the theoretical framework, the final research question addresses the debate of adhering to strict programming requirements for entry into the programme. More specifically, it informs on whether prior knowledge can be used as a factor for determining students' aptitude, by comparing the behaviours and attitudes, perception of knowledge and performance of students with prior knowledge and those without prior knowledge.

The research questions for each of the above explained are:

1. What is the impact of game attributes on student engagement states: affective, behavioural, and cognitive?
2. To what extent does engagement influence students' perception of knowledge?
3. To what extent does game attributes support learning outcomes?
4. How does student engagement affect the relationship between game attributes and learning outcomes?
5. How does having prior knowledge and not having prior knowledge compare in affecting students' ability to learn programming?

Structure of Thesis

This thesis is structured in five chapters, including the introduction chapter that presented the statement of the problem, previously considered teaching strategies considered by the department, the focus and scope of the study, the aims and objectives of the research, and finally the research questions this research intends to address.

Chapter Two - *Literature Review* - begins by briefly outlining the common challenges and issues in learning and teaching computer programming. It reviews GBL by placing it in the context of its popularity worldwide and in Trinidad and Tobago. As the chosen teaching strategy, GBL is then defined, and differentiated from serious games and gamification. This differentiation also introduces the ways game elements are considered in both GBL and gamification. The chapter then proceeds to evaluate the empirical studies that implemented both GBL and gamification at the higher education level in Computer Science between 2015 and

2021. It evaluates these on engagement and performance and learning approaches. It also reveals three commonly unexplored issues in the literature presented, that is, rare use of theories and theoretical frameworks, the lack of linking findings to specific game attributes, and a lack of connecting findings to learning outcomes. The chapter concludes with a critique of education games, this includes both gamification and GBL.

Chapter Three - *Research Strategy and Design* - details the considerations, approaches, and justifications for each decision made during the implementation process of this research. The first of which is the justification for the adoption of a case study approach. It follows by outlining the case study schematics used to design the research to ensure clarity and rigour of this study. The chapter then evaluates the theoretical frameworks commonly used in educational games research, and then presents, justifies, and details the application of the chosen theoretical framework to this research – the gamification for student engagement framework. Within the scope of the framework, it details the implementation in terms of engagement antecedents, engagement state, and engagement consequences. Next it outlines the five research questions that guided this research’s data collection and analysis. Four of which were based on the theoretical framework, and the final relating to the debate of student aptitude. The chapter then proceeds to explain the data collection and analysis process that includes participation selection, the various methods of data collection and the analysis strategies used for both the qualitative and quantitative data collected. Legitimation and ethical considerations are also addressed in this chapter.

Chapter Four – *Findings* - presents the results of the data collection and analysis process. It is organised by the structure outlined by the theoretical framework. It firstly reports on the students’ prior knowledge and experience in programming. Arranged by the engagement state part of the theoretical framework, the chapter reports on students’ opinions of the game attributes, the engagement states it influenced, and links the game attributes to the engagement states. The section uses the qualitative data to explain how these influences occurred. Using

both the quantitative and qualitative data, it presents and explains the perception of student knowledge as the measurable consequence of the change in student engagement states. It follows this by detailing on the students' overall performance – comparing the test scores before and after the course and depicts it further by the specific learning objectives: problem-solving, design and syntax.

Chapter Five – *Discussion and Conclusions* - interprets the findings and answers each of the five research questions posed by this research. The chapter then uses the propositions posed by the theoretical framework to make recommendations for a possible implementation of a GBL strategy for teaching programming at the university. It then concludes by outlining the limitations of the study and provides recommendations and directions for future research opportunities.

Chapter Two

Literature Review

This literature review used SCOPUS, Google Scholar, and Science Direct as the main search engine for articles using different combinations of the keywords, 'gamif*', 'higher education', 'computer programming', 'programming' and 'computer science', 'game-based learning', 'higher education', 'teaching', 'learning', and 'undergraduate'. The search was restricted to articles published between 2015 and 2021. While all articles found were limited to the 'English' language, the criteria for inclusion varied depending on the section of the literature review. The articles in this literature review that related to the nature and challenges in programming were included on the criteria of relating (1) strictly to computer programming, and (2) in higher education at the undergraduate level. The criteria for the sections relating to games and its various definitions in education, and its characteristics included (1) relating to gamification, game-based learning and/or serious games, and (2) strictly in the context of education.

Initially, the section that dealt specifically with game-based learning in computer science/programming, applied the inclusion criteria (1) relating to game-based learning, (2) in higher education at the undergraduate level, (3) implemented a digital game-based environment and (4) computer programming. However, this only returned 14 articles, and consequently greatly reduced the scope of the discussion. As a result, the first inclusion criteria were updated to 'relating to game-based learning and gamification'. Excluded from this were any articles relating to the development of a game-based or gamification environment. This combination returned 40 articles that matched the criteria specified. There were instances in which articles were included on a need's basis, particularly in the paragraphs dealing with engagement, learning approaches and theories. The chapter begins by briefly introducing the common issues and challenges in teaching and learning programming. As the focus of this literature review, it

details games in learning programming. In this regard, it firstly shows the state of popularity of GBL and gamification, using Google Trends. It goes on to define games in education, and differentiates between serious games, gamification, and GBL and how each defines game elements. Dealing specifically with literature in GBL and gamification, the chapter presents the common variables investigated (engagement, performance, and motivation), the learning approaches frequently mentioned and supported by both GBL and gamification, and the gaps found in the current literature. It concludes with the common criticism associated with using gamification and GBL.

Challenges and Issues in Learning Computer Programming

Jenkins (2002, p.55) stated, *“programming is a complicated business. It is not a single skill. It is not a simple set of skills.”* To learn programming, students must acquire a series of abilities that goes well beyond just learning the syntax of a programming language. It includes a combination of analysis, design, coding, code comprehension, verification, debugging, refactoring, and documentation (Ismail et al., 2010). Learning this myriad of skills can be particularly challenging and disorienting for first time programmers and many authors agree that problem-solving is the most significant challenge students face when learning programming (Medeiros et al., 2019; Soloway & Spohrer, 2013).

The literature showed that students often struggle with various aspects of problem-solving: deriving the problem from its description or requirements and decomposing this problem into sub-problems (de Raadt, 2007; Lister et al., 2004; McCracken et al., 2001; Robins et al., 2003), lacking the understanding and ability to use programming constructs for transforming the sub-problems defined into workable strategies and algorithms (de Raadt, 2007; Li & Watson, 2011; Lister et al., 2006; Lopez et al., 2008; Sajaniemi & Prieto, 2005), and not fully developing the capacity for program tracing which in turn affects their ability to author and comprehend and debug a program, that is, to find and fix both logical and syntax errors

(Ahmadzadeh et al., 2005; Fitzgerald et al., 2008; Holvikivi, 2010; Kolikant & Mussai, 2008; McCauley et al., 2008; Vainio & Sajaniemi, 2007; Xie et al., 2019).

Although not as common, some authors such as Altadmri and Brown (2015), Denny et al. (2011), Hristova et al. (2003), Jackson et al. (2005) and Sirkia and Sorva (2012) have reported students having difficulties in the programming concepts of control structures, logical operators, and syntax errors. Despite this, other authors such as Bosse and Gerosa (2017), Butler and Morgan (2007), and Lahtinen et al. (2005) argue that syntactical knowledge can be easily gained, and instead posit that students struggle with applying and combining the syntax to develop solutions.

Deep and Surface Learning

A fair number of articles found analysed the phases of problem-solving and implementation from a deep and surface learning approach. Deep learning is associated with understanding and engaging in the content in a meaningful way, whereas, in surface learning, students engage in selective memorisation or superficial way (Asikainen & Gijbels, 2017; Vanthournout et al., 2013). Relating this to programming, the research has distinguished surface learning as memorising the language syntax, and deep learning as understanding, problem-solving and integrating programming concepts (Mohorovičić & Strčić, 2011; Thota & Whitfield, 2010). Of the two strategies, authors such as Apiola and Tedre (2012), de Raadt (2007), and Fincher et al. (2006), argue that deep learning is ideal for learning programming with the latter two observing a positive correlation between deep learning and student performance. Arguably, the problem with deep and surface learning is that it often treats factual knowledge and problem-solving independently.

As previously discussed, programming is more than just mere factual knowledge, it is a skill that requires a combination of several activities that ranges from the basic knowledge of programming syntax to representations of program designs that must be understood, modified,

debugged, and documented. As a result, in learning programming, students need to grasp both the factual knowledge (syntax) and problem solving (Lister et al., 2006).

Issues and Trends in Teaching Programming

Teaching programming has been widely accepted by educators as a difficult task for its complexity, with teachers often being discouraged when students fail to understand the concepts being taught (Robins et al., 2003). Several articles were found that were consistent in the view that student challenges emanated from the traditional pedagogical design often used for teaching programming. This constrains students to become mere passive learners who often find it difficult to develop solutions, which in turn negatively affects their attitudes towards programming by reducing their persistency to continue learning and more importantly their motivation (Cheah, 2020). Authors such as Bennedsen and Caspersen (2012), Boyer et al. (2008), Bubica et al. (2014), Cheah (2020), Hegazi and Alhawarat (2016), Kolikant (2010), Massoudi (2019), Pears et al. (2007), Sharma et al. (2020), and Zhang et al. (2013), all agree that static teaching methods such as textbooks, lecture notes, slide presentations, labs, discussions, and audio/visual multimedia are not effective approaches for teaching a topic as dynamic as programming. In addition to teaching tools, some authors have also commented on the way solutions are presented to students as also contributing to their inability to develop problem-solving skills. Boyer et al. (2008), Gomes and Mendes (2015), and Cazzola and Olivares (2016) noted that teachers often present solved solutions in practical classes and then explain these solutions to students. According to the authors, this strategy encourages students to copy presented solutions. If errors occur, students merely randomly make modifications to the code instead of attempting to understand the problem and develop their own solutions.

These methods combined often lead to more emphasis being placed on syntax and semantics of a programming language, and the running of the programs rather than problem solving even if the lecturer's intention was such through the presentation of solutions (Gomes & Mendes, 2015; Ismail et al., 2010). Instead, any method chosen for teaching programming

should provide balance between factual knowledge (syntax) and problem solving, where students are made to concentrate on both understanding and problem-solving (Jenkins, 2002; Lister et al., 2006).

Trends in Teaching Programming

The overall outcome, of the above-mentioned, is often a pedagogy that is demonstrative and teaches students about programming rather than how to problem solve, design, and implement solutions. As a result, there is a need for pedagogical changes that involve introducing a programming language in an interesting and engaging learning environment that is both creative and challenging (Apiola & Tedre, 2012; Bubica et al., 2014; Casey, 1997). Over the years, there has been a large amount of research conducted in teaching programming, yet still, it has had limited effect on classroom practice and even more so, it has not provided evidence to support any specific approach (Apiola & Tedre, 2012; Pears et al., 2007). Specific to the area of computer science at the higher educational level, many approaches have been researched over the years, with several articles found that made use of syntax-free approaches (Al-Linjawi & Al-Nuaim, 2010; Deng et al., 2020; Durak & Guyer, 2019; Trilles & Granell, 2020), collaborative/computer supported collaborative learning (Chowdhury et al., 2018; Florez-Aristizabal et al., 2021; Kavitha et al., 2018; Rahman et al., 2020; Silva et al., 2020), pair programming (Ayub et al., 2019; Celepkolu & Boyer, 2018; de Oliveira & Reboucas, 2018; Josko, 2021; Othman et al., 2019; Sobral, 2020), problem-based learning (Maenpaa et al., 2017; Martins et al., 2018; Topalli & Cagiltay, 2018; Yuliati et al., 2018), live coding (Raj et al., 2018; Rubin, 2013; Shannon & Summet, 2015), and robotics (J. Aparicio et al., 2019; de Lima et al., 2016; Oddie et al., 2010; Piedade et al., 2020).

Another approach that has gained popularity over the years among educators is learning through games. Google Trend analysis tool visually represents the popularity of google searches relative to a peak point of 100 for a given region within a user specified time frame. A worldwide search between the time frame 2015 to present on the key terms – ‘game-based learning’ and

narrowing the results to 'education' showed that between 2015 and early 2022, popularity was mostly under half of peak interest. By end of 2021, and early 2022, there was a drastic increase in popularity, with the search term peaking at the highest of 100 (Figure 2). Narrowing the results further to Trinidad and Tobago, yielded zero results, which - according to Google, indicates that there is not enough data available.

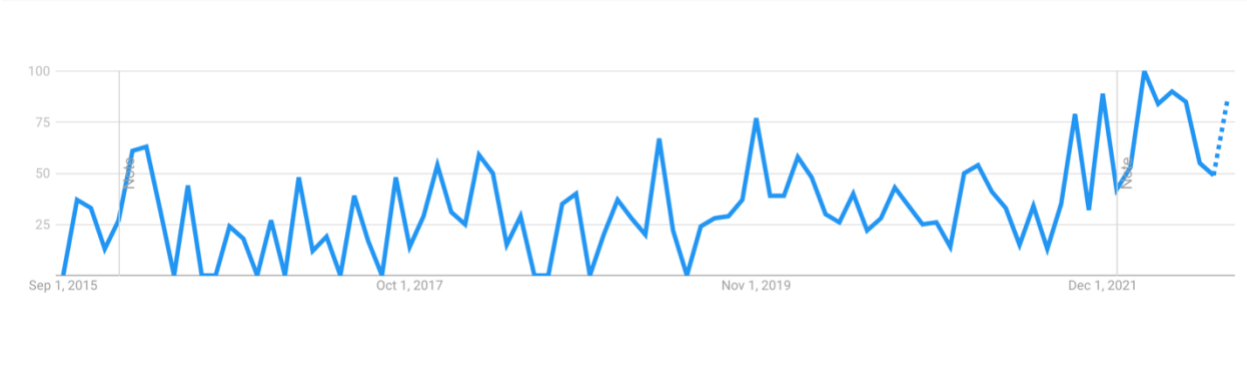


Figure 2 Google Trends Analysis - Game-Based Learning

The search results within the same period of 2015 to 2022 on the key terms – ‘gamification’ and ‘education’ showed a more consistent interest over the years, with most of the searches over half of the peak interest. Like game-based learning, interest reached its highest peak (100) by the end of the 2021 to early 2022 (Figure 3).

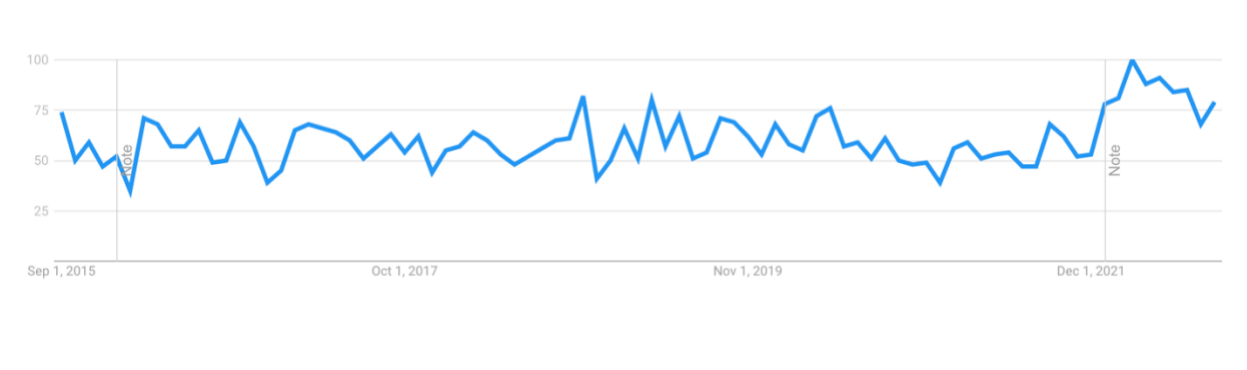


Figure 3 Google Trends Analysis - Gamification

Limiting the gamification results to the region of Trinidad and Tobago, showed sporadic popularity. With popularity reaching over half in 2016 and 2018, and under half in 2018, 2019, and 2020. Like the worldwide search, popularity reached its peak of 100 by the end of 2021 to the beginning of 2022 (Figure 4). Additionally, in both instances of gamification and game-

based learning, a search for research in gamification in education in Trinidad and Tobago – using SCOPUS, Google Scholar and Science Direct – had zero results, which suggests that this area is also largely unexplored in published works.



Figure 4 Google Trends Analysis - Gamification in Trinidad and Tobago

As previously mentioned, teaching programming requires targeting not only syntax and semantics but also the development of problem-solving and critical thinking skills, and even fostering cognitive factors such as motivation. Therefore, any teaching approach adopted for programming requires addressing all the above-mentioned.

Games in Education

Kinzie and Joseph (2008, p.2) offered the most comprehensive definition of game found in literature; “A game is an immersive, voluntary, and enjoyable activity in which a challenging goal is pursued according to agreed-upon rules. The game provides a safe environment for taking chances and the opportunity to develop the knowledge and refine the skills required to succeed”. Within this definition are found all the characteristics (underlined) and appeal of this method for use in education and have been applied to support a wide range of subject areas that include science, languages, culture, health, computer science, software engineering and business (Caponetto et al., 2014). Combined, the characteristics quoted promotes participation, feedback, knowledge acquisition, performance, engagement, co-operation, creativity, and the development of problem-solving strategies (Arango et al., 2008; Gros, 2007; Perrotta et al., 2013). All of these are considered desirable for students studying

computer science, as it provides the opportunity to target the core areas of programming: not only syntax and semantics but more importantly the development of problem-solving and critical thinking skills, and even fostering cognitive factors such as motivation.

Defining Gamification, Serious Games and Game-Based Learning

Distinct from games for pure entertainment, the terms gamification, game-based learning, and serious games were designed for the purposes of education (Hamari et al., 2016). The research has shown that the difference between these three are small and as a result they are often used interchangeably in literature (Almeida & Simoes, 2019; de Sousa Borges et al., 2014; Kim, 2015). It is important to note the differences between the two to better place into context the method of this research.

Defining Gamification

The two most quoted definitions of gamification in literature are that of Deterding et al. (2011), who defined gamification as the application of game-like elements and techniques used within a non-gaming context, and Werbach (2014) who stated that gamification is the process of making activities more game-like. The key aspect of both these definitions are the component of game-like elements, which both authors identified as a set of building blocks or features shared by games. Unlike Deterding and co-authors, Werbach's definition focusses on gamification as a process rather than game-like elements. According to the author, defining gamification as a process is necessary so as not to limit the definition, and most importantly places attention on the types of experiences it seeks to create, and the mechanism to do so. However, this definition is mostly directed to designers of gamified environments and its activities.

Examining further the concept of 'game elements' has shown that there is no agreed common definition; however, in the context of gamification, these elements are often defined by its most simple components. Authors such as Barata et al. (2013), Nah et al. (2014), Sailer et al. (2017), Huang and Hew (2018), and Denden et al. (2021) outlined varying features such as

points, leaderboards, badges, feedback, progress bars, avatars, levels/stages, and storyline as game-like elements (Table 1).

Game Elements	
Points	Measured record or calculation of success
Leaderboards	Comparison of outcomes to other players
Badges	Representation of success or achievement
Feedback	Evaluative or corrective information in response to a player's action
Progress bars	Visualisation of progression
Avatars	Graphical representation of a player (a persona or character)
Level/Stages	A section of a game in which a defined objective must be completed to progress
Storyline	Centred around a plot

Table 1 *Game Elements and its Definition*

Defining Serious Games and Game-Based Learning

The predecessor to gamification, game-based learning describes an environment where the game content plays a critical role in the acquisition and enhancement of knowledge and skills (Azadegan et al., 2012; Qian & Clark, 2016). The distinction between gamification, serious games, and game-based learning lies in this specific characteristic. Both serious games and game-based learning can be used as a pedagogical tool with the purpose of making learning fun, engaging, entertaining, and motivating (Al-Azawi et al., 2016; Arnab et al., 2015; Bellotti et al., 2013).

Serious games are considered fully fledged games and is mostly paired with game-based learning, in that serious games are the games themselves, while game-based learning provides the learning outcomes for using these games (Alhammad & Moreno, 2018). With gamification, the goal is not necessarily to change the existing practice of learning course, and content – like GBL – but instead to create an engaging and challenging environment using the game elements

outlined in Table 1 and integrating it into existing topics (Al-Azawi et al., 2016; Pivec et al., 2003; Plass et al., 2015; Scepanovic et al., 2015). On the other hand, serious games and game-based learning aims to fulfil specific learning outcomes (Alhammad & Moreno, 2018; Figueroa-Flores, 2016).

While game-based learning may utilise the components of game elements outlined in Table 1, in literature the game elements associated with game-based learning were mostly defined as attributes. Garris et al. (2002), Wilson et al. (2009) and Bedwell et al. (2012), were the most comprehensive studies found that outlined game elements as attributes. Garris et al. (2002) characterised game elements into six broad categories which the authors deemed as significant for learning, having previously conducted a literature review of game elements in 2001. These categories were: fantasy, rules/goals, sensory stimuli, challenge, mystery, and control. Wilson et al. (2009) in their literature review sought to determine the specific game attributes that impacted learning outcomes, and in doing so, expanded the categories initially put forward by Garris and co-authors, outlining 18 attributes. These were: adaptation, assessment, challenge, conflict, control, fantasy, interaction equipment, interaction interpersonal, interaction social, language/communication, location, mystery, players, progress and surprise, representation, rules/goals, safety, and sensory stimuli.

Though a comprehensive list, Bedwell et al. (2012) argued that this is also its drawback. That is, evaluating the relationship between game attributes and learning outcomes proved problematic because many of the attributes mentioned by Wilson and co-authors overlapped. Instead, Bedwell et al. (2012), through empirically testing this relationship, proposed a shortened overlap removed list of attributes (expressed as categories of the original list) that were more likely to affect learning outcomes (Table 2).

Category	Game Attribute	Description
Action Language	Language, Communication	Textual commands rather than use of an input device (e.g., joystick)
Assessment	Assessment, progress	Objectives and/or scores that are based on individual actions
Conflict/Challenge	Adaptation, conflict, challenge, surprise	Adapts based on how the individual performs
Control	Control, interaction (equipment)	Allows for the manipulation of objects in the game (e.g., throw, damage...etc)
Environment	Location	The actual location where the game takes place
Game fiction	Fantasy, Mystery	Simulation of reality
Human interaction	Interaction (interpersonal), interaction (social)	Networked game that allows communication with other individuals
Immersion	Players, representation, sensory stimuli, safety	Use of quality sounds and/or haptic feedback
Rules/Goals	Rules/Goals	Clearly outlines objectives

Table 2 Game Attributes as defined by Bedwell et al. (2012)

Games in Learning Computer Programming

Much of the appeal of implementing games in education is credited with the conviction that it boosts student's involvement, engagement, problem-solving capabilities, and motivation (Ahmad et al., 2019; Domínguez et al., 2013; Kapp, 2012). However, though game-based learning is popular in education, much of the research focuses on pre-K and K-12 education as a means of engaging children. Not much research has been done at higher educational levels, and even fewer have been done in computer programming (Hosseini et al., 2019). Between 2015 and 2022, only 15 of the 119 articles were found that was related to game-based learning in teaching and learning programming at an undergraduate level. With gamification, of the 36 articles found, a total of 25 articles met the criteria for inclusion. A combined 40 articles were included in this analysis. These articles predominantly examined some form of engagement, motivation,

and/or overall performance. 23 articles reporting a positive impact, 14 a mixed impact, and 3 having a negative impact. The articles also show a preference for quantitative data, with 24 articles using this form of data analysis and presentation. The remaining 16 articles used a mixed-methods approach, that is, both qualitative and quantitative forms of data. Table 3 outlines these articles and covers the following:

- The author/s of the article
- The variables that were the focus of the research
- Summary of the findings
- Any applied theories or theoretical frameworks used in the study
- The method of data collection and analysis. This includes qualitative, quantitative, or both (mixed).
- The overall impact of educational games on the variables under study. This included either having a positive, negative, or mixed impact.

The table distinguishes between gamification and game-based research, by marking gamification articles with '+' and game-based learning articles with '++'.

Table 3 Educational Games in Computer Programming - Review of Literature

+ Gamification

+ Game-Based Learning

Authors	Variables	Summary of Findings	Applied Theories	Method	Overall Impact
++ Abidin & Zaman (2017)	Engagement, Performance	<ul style="list-style-type: none"> • Enjoyable and fun • Enhanced their understanding • Felt that it will eventually enhance their thinking and problem-solving skills • Felt that the game scores will positively contribute to their overall performance • Preferred to work in teams rather than individually. Competition among the teams could enhance teamwork 		QUAN	Positive
+ Ahmad et al. (2020)	Engagement, Performance	<ul style="list-style-type: none"> • Students in the gamified environment were more satisfied than the traditional environment • Students in the gamified environment statistically significantly outperformed the students in the non-gamified environment 		QUAN	Positive

++ Akkaya & Akpinar (2022)	Performance	<ul style="list-style-type: none"> • Statistically significant increase in pre and post test scores for Object Oriented Programming and Computational Thinking • No statistically significant two-way interaction between the students' level of CPSS and attitude towards digital based learning of programming and achievement scores 	QUAN	Mixed
+ Begosso et al. (2018)	Engagement, Performance	<ul style="list-style-type: none"> • Students had a positive overview of using gamification for learning programming • Increase students' interest and enhanced pleasure. • Students in the gamified environment performed significantly higher than those in the traditional environment. 	QUAN	Positive
++ Butt (2016)	Engagement, Motivation, Performance	<ul style="list-style-type: none"> • Found the course enjoyable (not boring) • Some disparity between motivation and performance 	QUAL+QUAN	Mixed

		<ul style="list-style-type: none"> • Students are receptive of the idea of playing game to learn • Students found GBL helpful, interesting, challenging. Of those that disagreed with these statements, suggested that the students found the experience challenging but positive experience • Some students commented on how GBL helped develop their critical thinking and problem-solving skills. • The participants are not entirely convinced that it can substitute traditional methods 			
+ Call et al. (2021)	Motivation	<ul style="list-style-type: none"> • Students were motivated to finish their assignments earlier and commit code more frequently 	Self-Determination theory (SDT)	QUAL+QUAN	Positive
++ Chang et al. (2020)	Engagement, Motivation, Performance	<ul style="list-style-type: none"> • Statistically significant differences between pre and post test scores relating to the learning outcomes of understanding and application 		QUAL+QUAN	Positive

		<ul style="list-style-type: none"> Statistically significant on satisfaction, enjoyment, motivation 			
+ Cubukcu et al. (2017)	Engagement, Motivation	<ul style="list-style-type: none"> Most students understand related course knowledge better Most students were motivated: <ul style="list-style-type: none"> Extrinsic: most were motivated by badges and leader boards Intrinsic: most students were motivated by achievement Positive perception of gamification 	Intrinsic and Extrinsic Motivation	QUAN	Positive
+ Dambic et al. (2021)	Engagement	<ul style="list-style-type: none"> Student participation dropped as the course progressed 		QUAN	Negative
+ De Pontes et al. (2019)	Engagement	<ul style="list-style-type: none"> Students were fond of learning through games On average, the experimental group completed more assignments 		QUAN	Positive
+ de Sana Quaresma et al. (2020)	Engagement, Performance	<ul style="list-style-type: none"> Greater participation Greater collaboration Greater commitment to solving problems 		QUAN	Positive

		<ul style="list-style-type: none"> • Positive perception of learning through gamification • Improved performance 			
++ Dolgopolas et al. (2018)	Engagement, Motivation	<ul style="list-style-type: none"> • Better understanding was facilitated by reduced tension, improved self-confidence, and a boost in motivation • Authors noted that students had a challenge in translating what they did on App Inventor to actual code 		QUAL+QUAN	Mixed
+ Facey-Shaw et al. (2020)	Engagement, Motivation	<ul style="list-style-type: none"> • Mean scores for interest and enjoyment were not statistically significant between experimental and controlled group • Mean scores were statistically significant in the areas of perceived competence, effort/importance, and usefulness • The mean difference for interest/enjoyment trended downwards while the mean difference for the pressure/tension recorded an increase. 	Intrinsic Motivation	QUAN	Mixed

		<ul style="list-style-type: none"> • For the focus group results, students generally found the badges motivating, including those of their peers 		
+ Figueiredo & Garcia-Penalvo (2020)	Engagement, Performance, Motivation	<ul style="list-style-type: none"> • Students' attendance rate was higher with the gamified course • Students' performance in the activities increased between the years • Overall, qualitatively, students perceived the gamification environment positively leading to the conclusion that it was both motivating and important. 	QUAL+QUAN	Positive
+ Fotaris et al. (2016)	Engagement, Motivation, Performance	<ul style="list-style-type: none"> • EC (experimental class), had a slightly higher attendance than the control classed • EC students were more motivated to download the materials and do further reading. CC students seemed to have a lack of interest in doing same • EC students showed a small but steady weekly increase in their completion rate 	QUAL+QUAN	Positive

		<ul style="list-style-type: none"> • EC students had a higher academic performance 		
++ Gallego-Duran et al. (2017)	Engagement, Motivation	<ul style="list-style-type: none"> • Interest in the class increased • Motivation was increased 	QUAN	Positive
+ Garcia-Iruela et al (2020)	Engagement, Performance	<ul style="list-style-type: none"> • There was no significant difference in activity was observed between the gamified and non-gamified group • Student participation was the same between the two groups • No significant improvement in students grades before and after the activities 	QUAN	Negative
+ Harrington & Chaudhry (2017)	Engagement, Performance	<ul style="list-style-type: none"> • Improved attendance at the practical session • Improved retention rates and offered students the opportunity and motivation to overcome poor performance • There was a positive but not statistically significant correlation between the number of 	QUAN	Mixed

		challenges/points and final course grades			
		<ul style="list-style-type: none"> • There was strong relationship between system use and lower fail rate 			
+ Kasahara et al. (2019)	Engagement	<ul style="list-style-type: none"> • Students in the gamified environment improved code quality (modified programming habits) • In the gamified environment, code length was reduced 		QUAN	Positive
+ Khaleel et al. (2019)	Engagement, Motivation, Performance	<ul style="list-style-type: none"> • Significant change in students' knowledge gain compared to the control group • Students in the gamified environment were enthusiastic and interested • Statistically significant difference in motivational gains in the experimental group • There is a significant difference in the pre and post test scores in the experimental group 	ARCS Motivation	QUAN	Positive
+ Khaleel et al. (2020)	Engagement, Performance	<ul style="list-style-type: none"> • Continued to do work even after the achievement goal 		QUAL+QUAN	Positive

		<ul style="list-style-type: none"> • Students had a positive perception of the game environment • Students in the experimental group had a statistically significant higher result than the control group, indicating an increase in knowledge among students 		
+ Kumar & Sharma (2019)	Engagement, Motivation	<ul style="list-style-type: none"> • Students found the game environment more enjoyable, interesting. • Mean rating on fun, design of app, concept clarity, and content usefulness. This implies that the gamified approach enhanced student engagement, motivation and understanding 	QUAN	Positive
+ Landers & Landers (2015)	Engagement	<ul style="list-style-type: none"> • Gamification improved students' time on task (interacted more times than those in a traditional environment) 	QUAN	Positive
++ Lopez-Fernandez et al. (2021)	Engagement, Performance	<ul style="list-style-type: none"> • No statistical difference was found in pre-test scores between the control and experimental group. 	QUAN	Mixed

- The control group outperformed the experimental group in pre and post test scores difference. Neither method was statistically significant or more effective.
- Students preferred learning through game.
- In the experimental group of students, the GBL experience was: motivated and had fun

+ Lopez-Pernas et al. (2019)

Engagement, Performance

- Positive attitude towards the environment: Satisfaction was high among students, to which the authors also concluded that it fostered motivation
- Students were confident that their knowledge was increased in the gamified environment
- No correlation was found between students' self-perceived preparedness and learning effectiveness

QUAN

Mixed

		<ul style="list-style-type: none"> Although there was a statistically significant difference between pre and post test scores, no correlation was found between escape time and the increase in knowledge. Despite this, these students preferred learning in this environment 		
+ Marin et al. (2019)	Performance	<ul style="list-style-type: none"> Students had a positive perception of the gamified environment Grades were higher among students in the gamified platform 	QUAN	Positive
++ Martins et al. (2018)	Engagement, Motivation	<ul style="list-style-type: none"> Students came to enjoy the course The use of challenges was motivating, and enhanced personal satisfaction Students dedicated more hours to study because of this course The game environment motivated students to endeavour more in class 	QUAL+QUAN	Positive
++ Mathrani et al. (2016)	Engagement, Performance	<ul style="list-style-type: none"> Students found GBL fun, interesting, relevant, and effective in learning programming concepts 	QUAL+QUAN	Mixed

		<ul style="list-style-type: none"> • Some negative responses reported: boring, preferred to learn traditionally before playing the game, did not enjoy programming. Some students found that it did not help • There was relevance between the game elements and the programming modules, and increased self-confidence • The data showed no significant relationship between enjoyment and the level of difficulty / understanding • Performance was increased; students passed the module in their first attempt 		
+ Morales-Tujillo and Garcia-Mireles (2021)	Engagement, Performance	<ul style="list-style-type: none"> • Attention, relevance, and confidence were rated higher • Users in the gamified environment had a higher user experience rating - fun, interested, satisfied, challenged, competence; but also saw stressed, anxious 	QUAL+QUAN	Mixed

		<ul style="list-style-type: none"> Statistically significant improvement in student performance in the gamified group 			
+ Ortiz-Lopez et al. (2019)	Motivation, Self-efficacy, performance	<ul style="list-style-type: none"> Increased intrinsic motivation but not significant Intrinsic motivation did not mediate the changes in learning performance Lack of significant results in self-efficacy and intrinsic motivation. The authors attributed this to decreased motivation due to the programming challenges Students had a higher learning performance 	Intrinsic motivation, Self-Efficacy	QUAN	Mixed
++ Paiva et al. (2020)	Engagement	<ul style="list-style-type: none"> Positive reactions on each of the metrics (usefulness, ease of use, ease of learning, and satisfaction) Gradual increase in practice time, knowledge acquisition, and retention 		QUAL+QUAN	Positive
+ Rodrigues et al. (2021)	Motivation	<ul style="list-style-type: none"> Students achieved a higher learning gain 	Intrinsic Motivation	QUAN	Mixed

		<ul style="list-style-type: none"> No correlation between number of completed quizzes and intrinsic motivation 			
+ Rojas-Lopez et al. (2019)	Engagement, Motivation, Performance	<ul style="list-style-type: none"> Students were motivated to complete the challenges (intrinsic) Engagement was low among students who did not have the prior knowledge to do the challenges Increased emotions of joy and satisfaction, though some students were worried and anxious Many students favoured solving challenges in collaborative team Students of the control group performed better than those in the experimental group 	Intrinsic Motivation	QUAL+QUAN	Mixed
+ Shorn (2018)	Engagement, Performance	<ul style="list-style-type: none"> Although the experimental group had a slightly higher mean values on learning gains and engagement than the controlled group, it was not statistically significant. 		QUAN	Mixed

		<ul style="list-style-type: none"> Students in the gamified group performed better 			
+ Tasadduq et al. (2021)	Engagement, Intrinsic Motivation, Performance	<ul style="list-style-type: none"> No significant difference between gamified and non-gamified group in effort, satisfaction, and motivation Gamified group performed significantly better in assignments than the non-gamified group 	Intrinsic Motivation	QUAN	Mixed
++ Topalli & Cagiltay (2018)	Engagement, Performance	<ul style="list-style-type: none"> Students in the experimental group performed better than the control group in programming assessment Improved creativity, improve problem-solving, and made programming more enjoyable 		QUAN	Positive
++ Troussas et al. (2020)	Engagement, Performance	<ul style="list-style-type: none"> GBL was extremely effective for collaboration Students learning was enhanced through user interface GBL advanced students' knowledge Achieving the learning outcomes was significantly better among students in the GBL 		QUAL+QUAN	Positive

++ Zhao et al. (2022)	Engagement	<ul style="list-style-type: none"> • Students positively rated usability • Students agreed that it was helpful in knowledge acquisition • Students rated positively user experience 	QUAL+QUAN	Positive
++ Zhu et al. (2019)	Engagement	<ul style="list-style-type: none"> • Students were able to make connections between CPP and the programming representations of the game • Students enjoyed the game 	QUAL+QUAN	Positive
++ Zhu et al. (2020)	Self-Efficacy	<ul style="list-style-type: none"> • Students reported an increase in self-efficacy • Self-efficacy is correlated to the time spent in the game • Students utilised problem-solving strategies in their game play. 	QUAL+QUAN	Positive

Engagement and Performance

Although studies in student engagement have increased over the past few years, its definition is often varied, with the lack of clarity and consensus of its exact meaning being subjected to on-going debate (Ali & Hassan, 2018; Ferrer et al., 2022; Tight, 2020). As a result, wider theoretical and research literature in student engagement is often dynamic, complex, and multidimensional with various themes being used (Appleton et al., 2008; V. Trowler & Trowler, 2010). The current literature presented in Table 3 is an ideal example of how diverse engagement can be used. Examining the articles showed that engagement was frequently aligned to three categories: (1) engagement as a connection to the learning environment, (2) engagement as a measurable consequence, and (3) engagement as a form of involvement. Each of these encompassed several themes, with many articles evaluating a combination of themes across the different categories.

Engagement as a Connection to the Learning Environment. Within this category, the most quoted variable was *fun/enjoyment*. Other mentioned themes included, *satisfaction, interest, understanding, relevancy, competent, usefulness, boredom, challenging,* and *stressed/tension*. The research often reported the presence of several of these themes among students.

Gallego-Durán et al. (2017) showed that students' enjoyment and interest in the class increased in the GBL environment, with students shifting from hating programming activities to enjoying them. Zhu et al. (2019) learned that students not only enjoyed the game but were also able to make the connection between the representations in the game and the actual programming concepts. Similarly, Kumar and Sharma (2019) also discovered a mean rating on enjoyment, fun and interest for concept clarity and usefulness. Zhao et al. (2022) found that most students reported positive rating of the usability of the game, and an equal positive rating on students' perception of understanding. Martins et al. (2018) also observed that students enjoyed the course, and they preferred developing a project within a game environment rather

than projects that were traditional, such as, registration, payroll systems...etc. The authors also noted a high satisfaction rating, and based on the students' remarks, concluded that it was likely a result of the challenging game activities. The findings of Butt (2016) proved to be one of the most complex cases of mixed results. The research showed that while students enjoyed the course and found it interesting, they still found the activities challenging. Even faced with these challenges, they still rated their experience as positive, yet were not convinced that games in learning could be substituted for traditional learning methods.

Other studies linked the increase in engagement to students' understanding of the programming concepts and overall performance. For instance, Cubukcu et al. (2017) noted that most students had an increased understanding of the course contents and were able to better relate to the subject and this led to them having a positive perception of the use of gamification for learning. Similarly, Begosso et al. (2018), also showed that students had an overall positive perception of gamification, commenting that it increased their interest in programming. This led students in the gamified environment to perform better than the non-gamified class. Khaleel et al. (2019) found similar results, in that students in the experimental group developed a higher interest in learning when compared to those in the control group, which positively impacted knowledge gain and overall performance of the students. Comparably, Abidin and Zaman (2017) noted that students felt that the GBL environment was enjoyable and fun, a theme they attributed to their understanding of the programming concepts. Though the study did not statistically verify performance, it was documented that students felt that the game scores would positively influence their overall performance. Topalli and Cagiltay (2018) found that because of the increased enjoyment and fun, students in the experimental group performed better than the controlled group in the programming assessment. Marín et al. (2019) also reported better performance among students in a GBL environment, all of whom had a positive perception of learning programming through gamification. This was also confirmed in the study conducted by Ahmad et al. (2020), who found a statistically significant difference between pre and post test

scores. Based on students' responses the authors attributed this improved performance to increased satisfaction. Like Zhao and co-authors, Troussas et al. (2020) also found that usability was a contributing factor to students' feelings of knowledge gain. In statistically verifying the knowledge gain, the authors evaluated the performance of the GBL group with the non-GBL group and found that students in the GBL group performed significantly better in comparison.

However, this positive impact was not found in all cases. Authors Akkaya and Akpinar (2022), reported that while there was a statistically significant increase in learning gains on both object-oriented and computational thinking topics, there was no statistically significant interaction between students' engagement level and overall performance. Likewise, Dolgopolovas et al. (2018) concluded that while students had a better understanding of programming concepts that was facilitated by reduced tension, and improved self-confidence, they had a challenge in translating what they learnt in the GBL environment to actual code. The findings of Lopez-Fernandez et al. (2021) showed that students in the gamified group enjoyed the experience and had a positive perception of the learning environment, however, this did not affect their overall performance, and the control group outperformed the experimental group.

In comparing a class with games implemented (experimental), to a class being taught using traditional methods (control), Facey-Shaw et al. (2020) found no statistically significant difference in enjoyment between the groups. Unlike Martins and co-authors who found that challenges increased satisfaction, Facey-Shaw and co-authors noted the opposite; as the activities became more challenging, students felt more pressure/tension, that lead to a simultaneous downward trend in enjoyment as the course progressed. The findings of Morales-Trujillo and Garcíá-Mireles (2021), also saw students being stressed and anxious in a learning environment that was deemed enjoyable, fun, and feeling satisfied, competent, and challenged by the same students. Yet, the authors still found that there was a statistically significant improvement in performance among the students in the gamified group compared to the non-gamified.

Another case where a mixed impact was reported, was that of Rojas-López et al. (2019). The authors found that students in the control group outperformed those in the experimental group. Though there were feelings of joy and satisfaction, there were equal reports of heightened sense of worry and anxiousness among students. Furthermore, engagement was lower among students who did not have prior knowledge in programming. Shorn (2018) did find that students in a gamified environment performed better than those in a non-gamified environment. The author also found that there was a higher mean value in the gamified group's learning gains, however, this was not statistically significant. Tasadduq et al. (2021) showed that a high performance was not a necessarily a result of higher engagement. Although the gamified group performed better than the non-gamified group by comparison (in both assignments and exams), the author found no statistical difference in effort, and satisfaction between the groups.

Both Mathrani et al. (2016) and Lopez-Pernas et al. (2019) had a more complex mixed conclusion. Mathrani and co-authors discovered that while some students did rate their experience enjoyable and interesting, there was a minority that recalled the experience as being boring and commented that they preferred learning through traditional methods. Added to this, their results also showed that students were able to make the connection between what was learnt in the GBL environment and actual code (relevancy), and even found games effective for learning programming concepts, this led to an increase in performance among most students. Despite this, the authors found no significant relationship between enjoyment and the understanding of programming concepts. Lopez-Pernas and co-authors also found positive attitudes of satisfaction, confidence and a preference for GBL among students. However, there was no correlation between self-preparedness and learning effectiveness and although there was a statistically significant difference in pre and post test scores, there was no correlation between students' solutions and knowledge acquisition.

Engagement as a Measurable Consequence. Some themes that encompassed this category included *code quality, completed assignments, work continued/practice, and time on*

task. Researching from the perspective of writing code, Kasahra et al. (2019) found that students in the gamified environment code quality improved and their code length reduced. From the viewpoint of time on task, Landers and Landers (2015) found that students were likely to interact with the activities more times than the traditional learning environment, and overall improved the time spent on programming activities.

Like most of the findings of the other researchers relating to performance, the literature that focused on engagement as a measurable consequence was also sometimes linked to the influence of engagement as a connection to the learning environment. For instance, de Pontes et al. (2019) saw on average, students in the gamified group completing more assignments than their non-gamified counterparts. Among the gamified group, the authors noted that students were fonder of learning through games. Another study done by Khaleel et al. (2020) showed that with a positive perception of learning through games, students continued to practice programming even after the achievement goal, resulting in a higher achievement score than the control group. Paiva et al. (2020) reported a gradual increase in students' practice time, knowledge gains and retention, with equal positive impacts on usefulness, ease of learning, and satisfaction. Contrary to these findings, Harrington and Chaudhry (2017) evidenced a positive correlation between the number of completed assignments (challenges) and an improvement in course grades; however, this was not statistically significant.

Engagement as a Form of Involvement. Although not in the majority, a few articles viewed engagement as some form of involvement in class, mostly taking the form of participation and attendance. Assessing engagement from the standpoint of participation, authors de Sena Quaresma et al. (2020) found that students' participation and their commitment to solving exercises increased after implementing games, which also led to an improved overall performance. Dambic and co-authors' study was an example of how games can have an initial positive impact but decline overtime. The authors found that in implementing games students initially had a high enthusiasm, however, as the semester progressed there was a

loss of the novel effect and participation levels declined. Showing an overall negative impact, Garcia-Iruela et al. (2020) indicated that there is no significant difference in participation between the gamified and non-gamified classes. Likewise, there was also no significant improvement in student grades before and after the activities.

Other authors such as Figueiredo and Garcia-Penalvo (2020), Fotaris et al. (2016), and Harrington and Chaudhry (2017) also assessed engagement from the perspective of attendance. Thi study saw positive impact on completed assignments, and a higher attendance among students in the gamified group. These findings were comparable to that of Figueiredo and Garcia-Penalvo (2020). However, the works of Fotaris et al., (2016) did not see a positive impact, the authors only found that there was a slight increase in attendance of the gamified group, yet even with a small increase, there was an overall improvement in students' academic performance.

Learning Approaches in Current Research

Arnold (2014), Azmi et al. (2017), and Bíro (2014) argued that using games in learning has the potential to exponentially expand not only self-paced and life-long learning but also to bring with it the opportunity for facilitating other approaches to learning such as collaborative and problem-based learning. The recent studies in computer programming showed that game-based learning encouraged collaborative learning. Additionally, there were two variants in the way game-based learning was implemented in computer programming: (1) writing code to play a game, and (2) writing code to create a game. The former encouraged problem-based learning, while the latter encouraged creative-based learning.

Collaborative Learning

The studies of Abidin and Zaman (2017), de Sena Quaresma et al. (2020), Rojas-López et al. (2019), and Troussas et al. (2020) saw the potential of games to promote collaborative learning. Collaborative learning can be defined as an educational approach that involves learners working in groups to solve a problem, complete a task, or create a new product (Laal &

Laal, 2012). On its own, collaborative learning has the potential to develop understanding and knowledge, problem solving and critical thinking skills, all of which are generated from complex tasks rather than isolated activities (Laal & Ghodsi, 2012; Pivec et al., 2003). The use of games not only makes learning enjoyable, but it also affords the opportunity to engage more strongly in the process of learning - its content, its delivery and understanding (Amran et al., 2021). While most game environments encourage the development of individual learning skills (Romero et al., 2015), the four studies in the current literature showed that games also positively encouraged collaboration among students, especially when given complex tasks.

The findings of Abidin and Zaman (2017) showed that students found the course enjoyable and, with this increased engagement, they preferred working in groups rather than individually, and this collaboration helped in their understanding of the concepts. Similarly, de Sena Quaresma et al. (2020) also reported that students felt that collaboration was a central element in their learning and increased performance. Rojas-López et al. (2019) discovered that students favoured solving challenges in teams. Troussas et al. (2020) also noted that students made greater efforts to learn collaboratively and, like the findings of Abidin and co-authors, this advanced their knowledge in the programming concepts.

Problem-Based Learning (PBL)

As discussed previously, programming requires the deriving the problem from its description or requirements, decomposing this problem into sub-problems, and developing a technical and coded solution. This nature of programming makes it ideal for a problem-based learning approach. In problem-based learning, students are given more control over their learning by allowing them to focus on finding the problems, thinking ability, and the process of solving the problems (Chang et al., 2015; Walker & Leary, 2009). Games provide an ideal environment that allow students to explore, solve problems, attempt challenges, and make decisions (Figuroa-Flores, 2016).

Both Abidin and Zaman (2017) and Zhu et al. (2020) implemented GBL and PBL in time sensitive levels that increased in difficulty. Abidin and Zaman (2017) found that students believed that this feature enhanced their critical and problem-solving skills because it forced them to quickly develop solutions for the given problem. They reported that students felt that the game activities enhanced their thinking and problem-solving skills. Zhu et al. (2020) noticed that students' problem-solving strategies developed overtime. Butt (2016) noted that while most students commented on the overall game experience, 15% of them specifically credited the game activities for developing their critical thinking and problem-solving skills. Although Mathrani et al. (2016) used games to simulate problem-based scenarios, the authors focussed more on the student's perception of the GBL environment and understanding of the programming concepts but did not explore its impact on problem-solving and critical thinking skills.

Creative-Based Learning

Used mostly in the context of GBL, 3 articles applied learning through writing code to create the game. Often referenced as creative-based learning, this learning approach aims to transfer theoretical knowledge into practice by developing students' analytical, critical and communication thinking skills (Meeplat, 2020; Shabalina et al., 2016). In the context of programming, the objective of this approach is to produce original and valuable solutions to problems. This shared intention of developing the same cluster of skills makes creative-based learning and problem-based learning roughly the same. However, in this context, the difference lies in its application: writing code to create a game. Here, creative-based learning students are not necessarily faced with a problem to solve as is the case with problem-based learning, but rather students are developing and creating a game strategy with their acquired knowledge of programming.

Dolgopolovas et al. (2018), Martins et al. (2018), and Topalli and Cagiltay (2018) all implemented writing code to develop and create a game. In their study, Dolgopolvas and co-authors tasked students with developing an android educational game in C programming

language, however, the authors focused on evaluating the impact of this form on motivation and engagement. Although the approach proved to have a positive impact on self-confidence, motivation and reduced tension, the authors did not explore its impact on creativity or problem-solving abilities for which it is mostly credited. However, the works of Martins et al. (2018), and Topalli and Cagiltay (2018) did show that students developed a level of creative and critical thinking using this approach. Martins and co-authors' research required students to code a game, design the manual and produce an implementation report. The findings showed that students created more complex projects when compared to previous ones. Similarly, in Topalli and Cagiltay (2018) research, students were asked to design storyboards and code these designs and produce documentation for each, and the authors found that there was an increase in students' creativity and problem-solving ability.

Issues in Current Research

An analysis of the current research in educational games spotlighted three seldom explored issues: (1) theoretical underpinning for games in learning, (2) the influence of specific game attributes, and (3) impact on learning outcomes.

Theoretical Underpinning for Games in Learning

Some authors have argued that adherence to theories can result in an over-determination of the outcomes of the research, while strict adherence to theories does not allow for the consideration of alternatives (Ashwin, 2012; Trowler, 2012). Nonetheless there is a general agreement that the use of theories and theoretical frameworks not only justifies the importance and significance of the undertaken research but also plays a critical role in giving structure and meaning to the analysis and interpretation of data and placing the findings within the context of existing literature (Grant & Osanloo, 2014; Kivunja, 2018; Lederman & Lederman, 2015).

Despite the value of using theory and theoretical frameworks for analysing and framing the results of research, Seaborn and Fels (2015), in their survey study, found that 87% of applied

research found in games in learning were not grounded in theory or made use of any theoretical frameworks. Within the research that have adopted a theoretical foundation, there has been little cohesion in the theoretical underpinning, no doubt stemming from the abundance of educational theories through which an understanding in games in education can be realised (Landers et al., 2018; Seaborn & Fels, 2015). This may be because a significant portion of research at higher education is concerned with the development and improvement of practice, particularly in computer science (Nelson & Ko, 2018; Tight, 2014). This was certainly the case in the recent students outlined in Table 3. Collectively the research focused on enhancing students' engagement, motivation, and performance in computer programming. Within the few articles that have adopted a theory, gamification and game-based learning were examined through the lens of motivation and/or self-efficacy, with motivation being the more common of the two.

Motivation. Motivation in education is often regarded as a core factor in the learning process that concerns the activation and intention of students, particularly their energy, direction, and persistence (Ryan & Deci, 2000). The current outlined studies often discussed the findings through the lens of two motivational models: Self-Determination Theory (SDT), and Attention, Relevance, Confidence, and Satisfaction (ARCS) Motivation.

Self-Determination Theory (SDT). SDT is a macro motivational theory that distinguishes between two different reasons or goals that guides action: *intrinsic*, which refers to learners being interested in what they learn and in the learning process itself, and *extrinsic*, which refers to externally driven from the expectation of reward or punishment (Buckley & Doyle, 2016; Kapp, 2012; Nair & Mathew, 2019; Ryan & Deci, 2000). Deci and Ryan (2008) stated that SDT proposes that the basic psychological need for enhancing motivation is dependent on the degree of a person's feelings of:

- Autonomy – willingness to do a task.
- Competence – need for a challenge.
- Relatedness – feeling connected to others.

Of all the articles outlined, only Call et al. (2021) applied SDT to the interpretation of their findings. The authors concluded that while students did not start assignments earlier, games (using leaderboards) did motivate them to finish assignments earlier, committed code more frequently, and had more unit tests that passed, through the fulfilment of autonomy, competence, and relatedness needs.

Other studies evaluated motivation on the principles of intrinsic and/or extrinsic motivation. Cubukcu et al. (2017) found that most students were motivated both intrinsically and extrinsically using games. While students were extrinsically motivated by the visual indicator of their progress, they were opposed to the idea of having this accessible to the rest of the class. Intrinsically, students were motivated to get achievements as they progressed through the game. The findings of Facey-Shaw et al. (2020), Figueiredo and Garcia-Penalvo (2020), Fotaris et al. (2016), Gallego-Durán et al. (2017), Khaleel et al. (2019), Kumar and Sharma (2019), and Rojas-López et al. (2019) showed that students were intrinsically motivated by their achievements in the game to complete the game activities, even in the face of challenges as highlighted by Rojas-Lopez and co-authors.

Axelson and Flick (2010) and Ferrer et al. (2022) argue that there is often a direct connection between motivation and engagement. The works of Chang et al. (2020), Lopez-Fernandez et al. (2021), and Martins et al. (2018) are all testament to this, each establishing a connection between motivation and engagement, with motivation often leading to and/or facilitating some form of engagement. Although these authors, did not focus on motivational factors, each found that games increased motivation among students, often because of a positive impact on engagement attitudes of fun and enjoyment. Of these, Chang and co-authors went further and offered a more in-depth explanation, stating that the findings showed that the user interface of the game influenced both engagement and motivation, and often indirectly influenced satisfaction. This is likely because students are more likely to work harder and persist in their learning activities when engaged (Jarvela et al., 2008). Contrary to these findings, there

were instances of motivation having little to no relationship with engagement. For instance, in considering engagement (completed quizzes) as a measurable consequence, Rodrigues et al. (2021) found no correlation between the number of completed quizzes and motivation. Even further, the authors also found that depending on students' familiarity with programming, the impact can be negative. Tasadduq et al. (2021) showed that there was no significant difference in motivation between groups that were taught in a gamified environment and those that were taught using the traditional approaches.

The current outlined studies mostly showed a positive impact of using games to teach programming; even more so, it showed that oftentimes motivation can be linked to either influencing or being influenced by engagement in some form. However, the findings of Butt (2016) and Ortiz-Rojas et al. (2019) shows that this does not necessarily mean that student performance is improved as a result. Butt (2016) found that while students had a higher motivation, it had no impact on their performance. The findings of this study showed a disparity between motivation and performance. Similarly, Ortiz-Rojas et al. (2019) did observe an increase in intrinsic motivation, but this was not significant. Furthermore, intrinsic motivation did not mediate the changes in the students' learning performance.

Attention, Relevance, Confidence, and Satisfaction (ARCS) Motivation. Both a motivational and instructional model, ARCS is used for analysing motivational categories and then designing instructional material based on the analysis (Molae & Dortaj, 2015). The strategies that result from this analysis is intended to improve students' overall motivation in four components (K. Li & Keller, 2018). These categories include: *attention*, which is intended to stimulate curiosity, *relevance* which refers to the combination of student experiences (their interest and goals), *confidence* referring to the improvement of student's belief in their capability, and *satisfaction*, as evaluating student feedback (Goksu & Islam Bolat, 2021). Of all the articles that implemented games for teaching programming, only Khaleel et al. (2019) used ARCS to measure the effectiveness and motivational levels of students in a gamified learning

environment. Using motivational tests that evaluated all four categories of ARCS, the authors found a statistically significant gain in motivation among the students of the experimental group.

Self-Efficacy. The concept of self-efficacy is often defined as a person's belief in his or her capability to successfully perform specific tasks (Bandura, 2006). The belief in capabilities, more than skills and knowledge, have been theorised as having an influence on students' motivation, effort, confidence, persistence, strategizing, perception of the learning environment, and even performance (Heslin & Klehe, 2006; Lorschach & Jinks, 1999; van Dinther et al., 2011; Versland, 2016). Many of the findings showed that the higher the efficacy the more likely students are to view difficult tasks as something to be mastered, while students with low efficacy are more likely to avoid the challenges (Zimmerman, 2000).

In using this construct, two studies – Ortiz-Rojas et al. (2019), and Zhu et al. (2020) both assessed self-efficacy as an impact of educational games. Both these articles viewed self-efficacy from the standpoint of self-confidence. The findings of Ortiz-Rojas and co-authors showed that students' initial self-confidence was high, but this decreased over time as the programming activities became more challenging. This, however, did not have a direct effect on learning gains, as it was reported that there was still a statistically significant increase in this regard. Opposite to this, Zhu, and co-authors revealed a statistically significant increase in self-confidence for each of the programming topics assigned and a statistically direct connection to increased problem-solving capability.

Although self-efficacy was not a focussed variable in the research conducted by Dolgopolovas et al. (2018), and Facey-Shaw et al. (2020), they both reported findings of improved self-confidence, effort, and persistence. The findings of Dolgopolovas and co-authors showed that improved self-confidence was a contributing factor to students' better understanding of the programming concepts. Though Facey-Shaw and co-authors found a mixed impact of educational games, it was reported that there was a statistically significant

mean score in both perceived competence and effort in the experimental group compared to the control group.

Influence of Specific Game Attributes

As previously mentioned, game elements are often viewed in two ways – as elements and as attributes. Game elements include features such as points, leaderboards, badges, feedback, progress bars, avatars, level/stages, and storyline. Game attributes comprise categories of action language, assessment, conflict/challenge, control, environment, game fiction, human interaction, immersion, and rules/goals. While most articles evaluated the effectiveness of these elements on engagement, performance, and motivation, arguably the true impact of game elements is dependent on how the users are interacting with its features. Stott and Neustaedter (2013) argues that feedback, freedom to fail, progression and storytelling are dynamics that have shown to be consistently successful when applied to game learning environments. As one of the most critical elements, feedback – especially continuous and timely feedback - provides the means through which students can reflect on their learning and make the adjustments to make better progress at the learning stages. Freedom to fail encourages students to explore and experiment in the learning environment, providing the opportunity to focus on the learning process rather than results. Progression, such as levels or missions, frames, guides, and supports learning through the categorisation of information to focus their attention. Storytelling provides a context in which actions and tasks can be practiced, a feature considered effective for increasing engagement and motivation.

These game design features have also been noted in other articles. Lameris et al. (2017) commented not only on the importance of feedback to ensure that students embed learning, but also on ensuring that there is an alignment between pedagogy and storytelling to foster a balance between engagement, motivation, and learning. Agapito and Rodrigo (2018) also argues that the freedom to fail and recover works in favour of both weak and strong students – giving them the opportunity to improve. They also argue that the ability to visually monitor progress

gives the students the opportunity to identify their weaknesses. Dempsey et al. (2002) made similar arguments stating that games should provide the setting in which students can learn through the process of trial and error (freedom to fail). The authors also added that students emphasised that clear and concise instructions are needed. Even more so, while aesthetic features such as colour, screen design, sound and feedback sustain interest in learning, players should be given the opportunity to have control over difficulty levels, especially needed for those who are already familiar with the topic areas. The importance of the adjustment of difficulty levels was also made by Azadegan et al. (2012), who stated that challenges that are too rapid may lead to disengagement and degraded overall performance, while a game that is too passive may lead to boredom. Having the right amount of challenge as the students progress through the learning environment encourages persistence when dealing with more challenging tasks, which ultimately leads to feelings of competency. A feeling closely linked to self-efficacy.

While the current literature often reported the impact of educational games in teaching computer programming, several overlooked the features concerning how it is used by students. A contributing factor to this is most likely the preference for statistical analysis and presentation of data. Of the 40 articles, 26 used quantitative methods. The remaining 14 articles used a mixed methods approach, but only 12 of these tried explaining how the elements were used. It is also at this point that the research in gamification and game-based learning deviates.

Seven of the articles that used gamification techniques, implemented some form of visual representation of students' progress. The most widely used were leaderboards, badges, and points. Begosso et al. (2018), Call et al. (2021), Fotaris et al. (2016), Khaleel et al. (2020), and Morales-Trujillo and Garcíá-Mireles (2021), all found that seeing their achievements motivated students to put more effort into the course. Additionally, Begosso and co-authors also mentioned students had a greater sense of competence in seeing their progress. Khaleel and co-authors also commented that students felt that the ability to revisit the activities also motivated them to learn the content. Although Figueiredo and Garcia-Penalvo (2020) found that students

were motivated to work hard, they also noted that visual representations of progress can be demotivating to students that perform poorly. The authors also argued that gaining points rather than earning a grade reduced pressure. Much like Stott and Neustaedter (2013), Rojas-López et al. (2019) also showed that feedback provided several opportunities for students to improve their work and understanding of programming concepts, even feedback from their lecturers.

Only five articles used GBL, and each related mostly to the overall game experience and gameplay and made no direct link to the specific game attributes as gamification research did. However, some features that impacted were mentioned by four of the articles, and it saw some mention of features regarded highly by students. Paiva et al. (2020) noted that students credited immediate feedback to their understanding of the programming concepts. Similarly, Chang et al. (2020) found that not only did feedback motivate students to keep learning but allowed students to learn from their errors. This ability to learn from errors and revisit the game activities was also reported by Mathrani et al. (2016) who mentioned that this increased students' confidence. Different to the common results, Zhu et al. (2019) stated that while visualisations were instrumental in students understanding of the programming concepts, it also assisted in them making connections to the programming language.

Impact on Learning Outcomes

As previously referenced, a range of outcomes have been explored in the current research. Most studies explored various indicators of engagement; some evaluated motivation, and others used understanding or performance for determining the measurable outcome of implementing educational games in a computer programming course. Although each of these offered important evidence about the impact, it remains unclear in literature what influence educational games have on the specific learning outcomes of a course (Perrotta et al., 2013; van Staalduin & de Freitas, 2011).

Ahmad et al. (2020), Khaleel et al. (2020), Lopez-Pernas et al. (2019), Morales-Trujillo and García-Mireles (2021), and Troussas et al. (2020), all assessed students' fulfilment of learning outcomes from the perspective of overall performance. Each reported a significant improvement in students' performance in the gamified environment, that the authors argued was a fulfilment of the learning outcomes of the course. Conversely, Dambic et al. (2021) divided the course into topics that required students to complete each topic before progressing to the next. In their study, the fulfilment of the learning outcomes was evaluated on student participation. The authors reported a decrease in participation as the course progressed, thereby concluding that students did not entirely meet the learning outcomes.

Chang et al. (2020) also used performance for evaluating learning outcomes and concluded that students fulfilled this by the significant difference in pre and post-test scores. However, the authors went further by linking to Bloom's cognitive domain classification for assessing learning outcomes. Bloom's taxonomy provided a broad framework that allowed Chang and co-authors to generalise the learning outcomes that game attributes can affect.

Bloom's Taxonomy. Bloom identified six levels of learning: remember, understand, apply, analyse, evaluate, and create (Figure 5). Each of these gives meaning to the educational goals with each category defining a particular process of learning:

1. Remember – recalling facts
2. Understand – explaining the ideas or concepts
3. Apply – using the information learnt in new situations
4. Analyse – drawing connections among ideas
5. Evaluate – justifying a decision
6. Create – producing new or original work

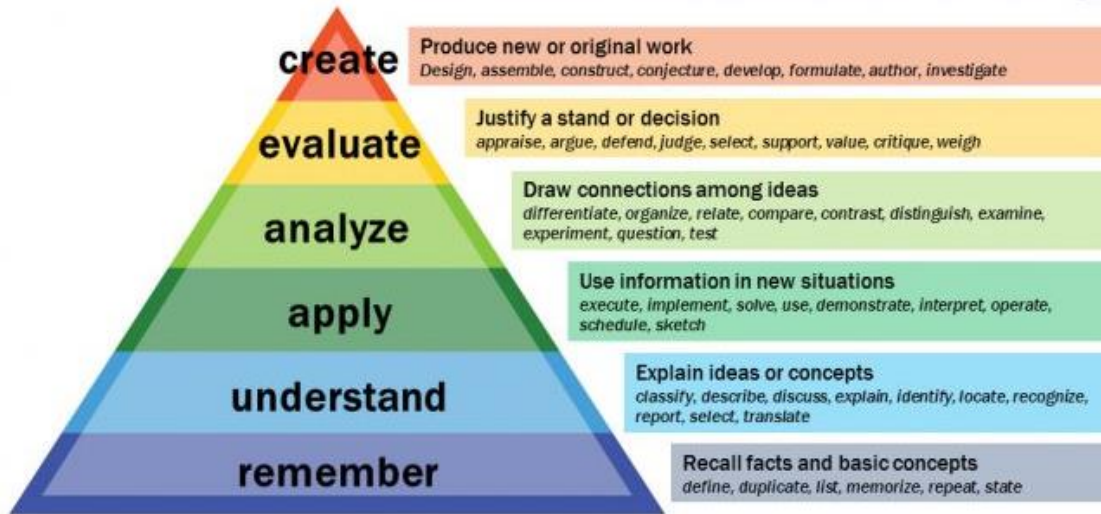


Figure 5 Bloom's Taxonomy taken from (Armstrong, 2010)

In this, 'remember' is the lowest order of skills that requires the least cognitive processing skill and 'create' is the highest order skill the requires a much greater cognitive processing skill (Adams, 2015; Starr et al., 2008). Using this framework, and based on the students' test scores, Chang and co-authors concluded that GBL impacted positively on both the 'understand' and 'apply' levels of the hierarchical model.

Critique of Games in Education

Although the implementation of games in education assumes that players learn, explore, and acquire knowledge during the game, in the wider literature, there are several criticisms.

Firstly, Todd's (2017) main criticism of educational games, particularly gamification is directly related to some of the gaps already presented. The authors argue that the research in this area remain shallow. One main reason is the lack of theoretical frameworks that addresses the application of educational games; the authors argue that this reduces the current literature to focus on tips and tricks rather than underpinning learning theories. Even among the literature in programming, it was shown that of the 40 articles presented only 8 articles used theories to underpin their research. Todd (2017) also argued that engagement and motivation were mostly reduced to students' perception and not the measurement of engagement. Although this may be a valid argument in other educational areas, the current literature in computer

science showed that this critique is not necessarily true. It has already been shown that many articles measured engagement and did so in several ways – either through completed assignments, continued work/practice, and time on task.

Secondly, authors Pohl et al. (2009) and Simkova (2014) criticize educational games on the point of design and development. The primary practical issue is that creating a highly engaging gamified environment is difficult, time-consuming, costly and requires technical infrastructure and pedagogical integration (Ahmad et al., 2020; Kapp, 2012). Overlooking any of these considerations are possibly the reason most educational software are of poor quality, badly edited and oftentimes unprofessional (Jenkins, 2002). While commercial software is also an option for educators, others have criticised its use of having little to no flexibility. Torrente et al. (2010) argues that commercial games are so rarely scalable that it does not allow educators the flexibility to adapt, reuse, maintain and share their materials, which may hinder them from fully integrating the learning objectives of the course.

Lastly, Toda et al. (2019) in their literature review criticised educational games on the negative side effects of their use – characterising them as loss of performance, and undesired behaviour; effects also reported in much of the literature presented. While there were instances where loss of performance was reported, the literature also showed a decline in engagement. These effects included little to no correlation between engagement and performance (Akkaya & Akpınar, 2022; Harrington & Chaudhry, 2017; Lopez-Pernas et al., 2019; Mathrani et al., 2016), an inability to translate what was learnt in the game to actual coding (Dolgopolovas et al., 2018), little to no improvement in performance (Garcia-Iruela et al., 2020; Lopez-Fernandez et al., 2021; Shorn, 2018), and overall lack of engagement (Dambic et al., 2021; Rojas-López et al., 2019; Tasadduq et al., 2021).

Shabalina et al. (2016), also shared Toda and co-authors' argument that there is the potential for students to experience undesirable emotions when using educational games, such as dissatisfaction, frustration, and the inability to complete the activity given. The current

literature also shows that educational games can have the opposite effect than intended; it was challenging (Chang et al., 2020), fostered feelings of tension, stress, and anxiety (Morales-Trujillo & Garca-Mireles, 2021; Rojas-Lopez et al., 2019), boredom (Mathrani et al., 2016). There were also additional negative effects reported that included lack of motivation (Ortiz-Rojas et al., 2019), and no correlation between motivation and engagement (Rodrigues et al., 2021).

Summary

Learning programming requires students to acquire a myriad of skills that range from analysis of problems, designing solutions, coding, comprehending code, refactoring, verifying, debugging, and documenting solutions. With a topic as dynamic as programming, facilitating its learning is equally challenging for teachers. It is widely known that traditional teaching methods are largely ineffective. Several strategies have been proposed over the years and includes pair programming, robotics, syntax-free approaches, live coding, problem-based learning, and computer supported collaborative learning. One strategy that has gained popularity over the years is the use of games. In the context of education, games often span three terms: gamification, game-based learning (GBL), and serious games. Serious games are considered the game itself, while GBL provides the learning outcomes of the game through the integration of components such as, action language, assessment, conflict/challenge, control, environment, game fiction, human interaction, immersion, and rules/goals. As the successor of game-based learning, gamification is the integration of elements such as leaderboards, badges, points, and the like into a course, with the purpose of creating an engaging environment rather than fulfilling specific learning outcomes, like GBL.

Although popular, it has not been widely adopted in the teaching of computer programming at an undergraduate level. As a result, there have been few published works in both gamification and game-based learning, and none that originates from Trinidad and Tobago. Of the few articles that related to educational games (gamification and GBL) in teaching

computer programming, most reported a positive impact of educational games on student engagement, performance, and motivation. However, closely behind was the report of mixed impacts of educational games on same. Linking to one of the main critiques of educational games, is that it has the potential to provoke undesirable emotions that includes boredom, anxiousness, frustration, and dissatisfaction. Apart from the outcome of implementing GBL, the literature also showed that GBL supports three approaches to learning: (1) collaborative learning, (2) problem-based learning, and (3) creative-based learning.

The articles found also revealed three mostly unexplored issues. Firstly, there has been a lack of theoretical frameworks or theories applied, and most that did, examined it from the standpoint of motivational theories - self-determination theory, intrinsic motivation, and attention, relevance, confidence and satisfaction (ARCS). Equally in the minority, was the analysis from the perspective of self-efficacy or self-confidence. Secondly, there has been a general lack of connecting the increase of engagement, motivation and/or performance to specific game attributes. While research based on gamification did attempt to make this connection using some form of visual progress, GBL research was the most culpable - frequently referring to the overall gaming experience with no connection to specific features or attributes that were the most influential. Thirdly, while some of the studies focused on the overall performance of students, few linked this to specific learning outcomes of the course under research.

Chapter Three

Research Strategy and Design

This chapter introduces the methodological approach adopted for this research and details how it connects to the answering of the research questions. It also details and justifies the theoretical framework adopted in this research that ultimately guided the analysis, presentation, and interpretation of the data. The chapter also describes the specific methods of data collection and the analysis of the data collected. It also addresses reliability and validity and outlines the ethical considerations in doing this research.

Research Strategy

This research was designed as a case study. Although the literature has several interpretations of case studies, the decision to use this strategy was largely based off the works of Denscombe (2014), Tight (2017), and Yin (2018). Derived from the writings of these researchers, a case study is defined as an in-depth study of a specific case in its own context – complex and bound – with an intention that focuses on how and why certain outcomes occur rather than on the outcomes themselves. It is within this understanding that the case study approach was judged the better option because it allowed for an in-depth insight into how games in education can impact engagement and performance through the first-hand experiences of students, the bounds of which was the Introduction to Computer Programming course at the University of the Southern Caribbean, making this research a single case study. As a single case study, this research examined and represented only one case (Gray, 2014) - Introduction to Computer Programming course.

One of the major strengths in choosing a case study lies in its definition. It allowed this research to be focused, in-depth, detailed and better able to grasp the complex relationship (Denscombe, 2014; Tight, 2017), between game elements, engagement, and performance. The insights gained from this can be interpreted and used for staff/individual development, within-

institutional feedback, and educational policy making (Cohen et al., 2018). This advantage aligned well with the one of the purposes of this research, that is, to inform professional practice and possibly transform teaching practices within the department at USC.

Despite the benefits gained from choosing to design this research as a case study, it is not without its criticisms. Tight (2017) classifies these pitfalls as generalisability, reliability, and validity. It is an on-going debate on the ability to generalise case studies, particularly, single case studies. The scepticism stems from the question of how far one can make assumptions about a wider population of cases from a small case study (Denscombe, 2014; Yin, 2018). Related to this research, the case, though limited in current literature, was not entirely unique – it implemented educational games to determine its impact on engagement and performance in an introductory computer programming course at the undergraduate level. Although the aim of this research was not meant to be generalisable, several strategies were implemented to ensure a meaningful case study, and one that can at least stand the scrutiny.

Firstly, a theoretical framework was adopted to ensure a deeper and fuller understanding was applied to the analysis and presentation of the findings (Meyer, 2001; Tight, 2017). Adopting a theoretical framework also placed this study among the few that have done so in literature related to educational games in computer programming. For this, a few frameworks were considered ranging from generic theories to theories specific to educational games. Ultimately, this research adopted the Gamification for Student Engagement Framework.

Secondly, the findings of this research were triangulated, as recommended by Tight (2017) – in the case of this research, methodological triangulated. Methodological triangulation refers to the use of more than one set of data collection methods (Wilson, 2014). As such, a mixed-methodology approach was adopted. Mixed-methodology combines the characteristics of both qualitative and quantitative research approaches where quantitative is measured data, expressed using statistics, and qualitative is categorised or descriptive data (Bryman, 2006; Johnson et al., 2007; Leavy, 2017). Borrowing from the strengths of both qualitative and

quantitative methods, the mixed methodology approach is considered as transcending the limitations of each (Heale & Forbes, 2013). Along with triangulation, this method allowed this research to provide a comprehensive understanding of the research data and its correlation (Cohen et al., 2018; Creswell, 2014; Denscombe, 2014), where the qualitative data was used to explain the quantitative data, and in some instances where quantitative data was used to verify qualitative data. This ensured that if differing results were found, it would highlight aspects of the phenomena, and lead to new and better explanations (Heale & Forbes, 2013). This strategy also addressed concerns over the lack of rigour relating to single case studies, from its focus on process rather than measurable outcomes (Denscombe, 2014; Zainal, 2007).

Lastly, to ensure that the collection of data, its analysis and the presentation of the findings are defensible, a formal schematic was used for designing this research (Tight, 2017). For this, the nine-step schematic outlined by Rosenberg and Yates (2007) was used (pg.66). This schematic provided the most comprehensive guideline and outlined a rigorous process for focused data collection and in-depth analysis of the findings. As part of the process, it considers the underpinning theoretical framework of the research – an essential component in addressing the critique of case studies and the gap in current research.

Another criticism of case studies is the lack of checks and balances surrounding reliability and validity - where validity is collecting of appropriate data for answering the research questions and reliability is the replicability of the research in the same way to produce the same results (Cohen et al., 2018; Tight, 2017). In dealing with reliability, Yin (2018) recommends that the procedures of the research be made explicit. Strategies already considered, such as using a formal schematic for designing the research and implementing a theoretical framework, ensured that the necessary details were provided to mitigate these criticisms. With validity, an interesting take on the issue have been put forward by authors such as Onwuegbuzie and Johnson (2006), who argued that the term 'validity' should be replaced by the term 'legitimation' for mixed-methodology research. According to the authors, mixed methods

combine the strengths of both qualitative and quantitative methods and their non-overlapping weaknesses. As a result, validity and reliability in these instances is complex, and instead requires its own requirements. They outlined nine legitimization types that overcome the problems of 'reliability and validity' of mixed methods research. Each was carefully considered in designing this research (pg.95).

Case Study Design

Rosenberg and Yates (2007) nine-step schematics deconstructs and identifies all the inter-related elements of a case study. The elements include: (1) pose the research question, (2) identify underpinning theories, (3) identify the case, its context, and the phenomena of interest, (4) determine the case study approach, (5) identify data collection methods, (6) select analysis strategies, (7) refine analysed data through an analytical filter, (8) use matrices to reduce the data (optional), and (9) determine conclusions.

To ensure clarity, rigour, and methodological integrity each of the above-mentioned elements was carefully considered and applied to designing this research. Its application to this research was represented by a visual map (Figure 6). For this research, the case consisted of 18 participants. As a result, the volume of data was not so large that it required the use of matrices and was therefore not applied to the design of this case study.

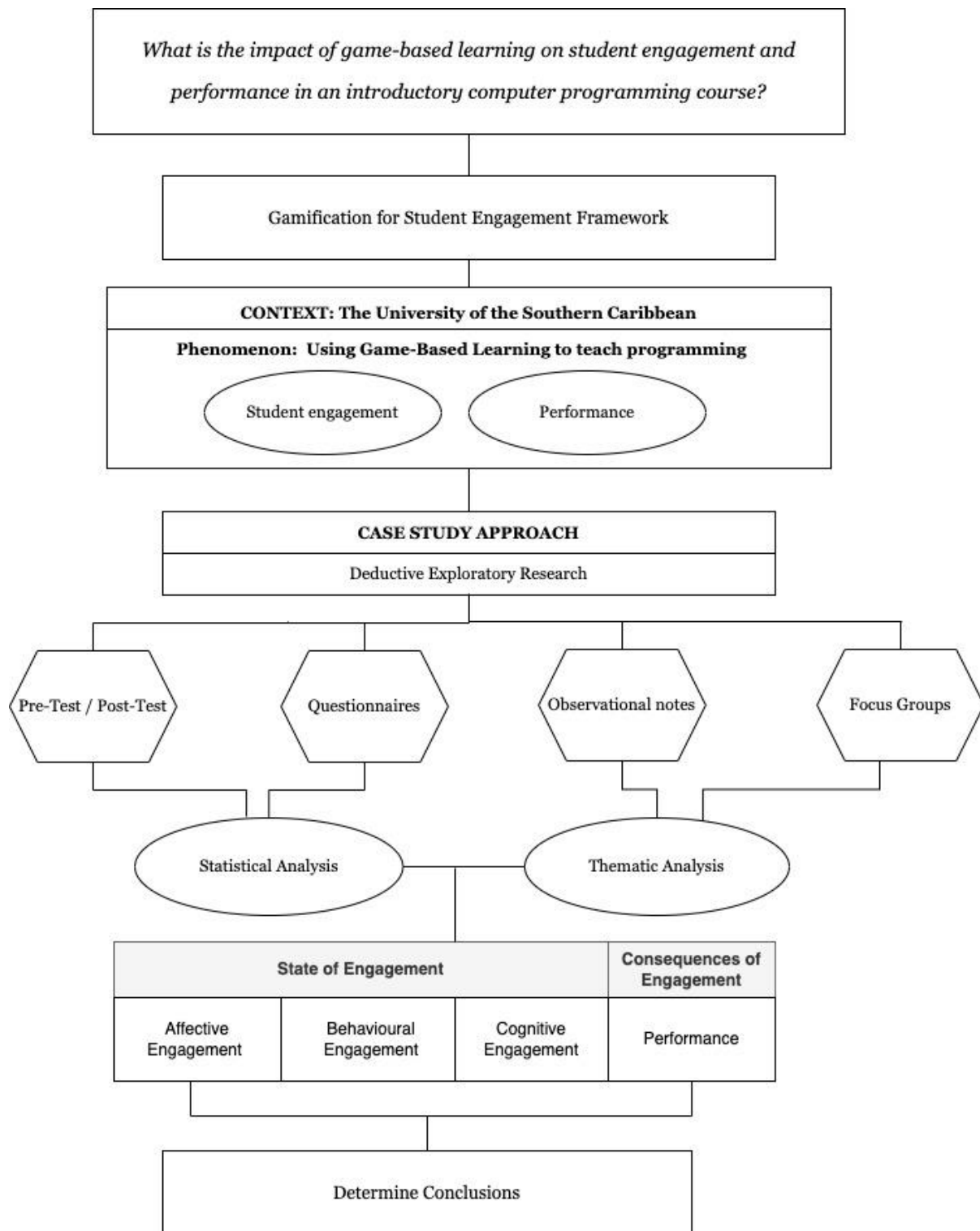


Figure 6 Application of Case-Study Schematics, adapted from Rosenberg & Yates (2007)

Theoretical Framework

Chapter 2 has shown that there were few research articles related to educational games in computer programming that implemented theories or theoretical frameworks in their study. Of the articles that did, they were based on generic motivational theories and models, such as Self-Determination Theory (SDT), Intrinsic and Extrinsic Motivation, and ARCS theory of motivation. The core aim of this research was to evaluate the impact of game-based learning on student engagement and performance. Generic theories and theoretical foundations were not considered because they often place emphasis on evaluating the impact of educational games on motivation and student engagement and not necessarily on student performance (Bai et al., 2020). Further to this, generic theoretical models also do not provide a mechanism in which to explore the impact of specific game elements or a combination of elements on engagement and performance (Landers, 2014; Loughrey & Broin, 2018; Wilson et al., 2009).

Theoretical Frameworks that focused on Educational Games

Separate from the use of the generic theories, other studies have also developed theoretical frameworks that are either blended versions of popularised motivational models with game elements or mostly untested gamification frameworks (Huang & Hew, 2018). Aparicio et al. (2012) developed a gamification framework based on self-determination theory. The authors proposed an iterative sequence of activities that included identification of the main objective, identification of the transversal objective (underlying objectives that are interesting to those involved), selection of game mechanics and effectiveness analysis (evaluating the effectiveness of gamification with the use of indicators such as fun and satisfaction). However, no articles were found that empirically tested this theory.

Hammerschall (2019) also proposed a gamification framework based on the outcomes of the self-determination motivational theory and the behaviour change theory; dubbed transtheoretical model of change (TTM). In combining these two theories, the framework derives a set of requirements for developing a long-term gamified strategy or process that

motivates students. In addition to the lack of empirical evidence of its validation, the defining feature of this framework is not in the evaluation of gamification in the learning environment but rather the development of applications that support the learning process.

Other authors have proposed gamification frameworks not solely rooted in popularised theoretical models: input-process-output model, and theory of gamified learning.

Input-Process-Output Model. Proposed by Garris et al. (2002), the model focuses on using gamification as a training intervention for motivating learners in three major objectives (Figure 7):

1. *Input.* The design of instructional content that incorporates features of gamification characteristics.
2. *Process.* Learning through an iterative process of user judgements and reactions (fun and enjoyment), user behaviours (such as, persistence and time of task) and system feedback. According to the authors, this cycle is the model's defining feature because it results in recurring, self-motivated game play once the game elements and instructional content are carried out correctly.
3. *Output.* The iterative engagement in the game play, which is likely to result in the achievement of the training objectives or learning outcomes.

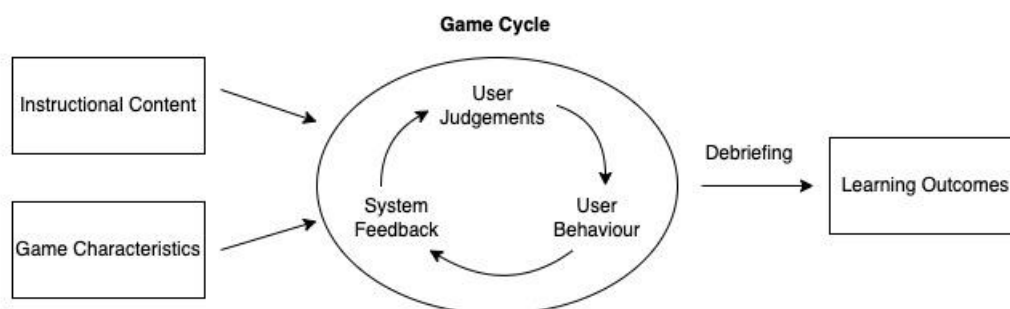


Figure 7 *Input-Process-Output Model (Garris et al., 2002).*

Empirically, this model has not been adopted or validated in current research, except for one study, that was found, and was done by Fan et al. (2015). The study also integrated Kolb's learning styles theory; the authors utilised the input-process-output model for designing activities for a gamified learning system with the purpose of assisting students in comprehending human blood circulation in a biology course for junior high school.

The Input-Process-Output was considered as it provided a framework for evaluating gamification on learning outcomes. However, the framework implies that the games assume the role of the instructor by providing the learning content, while the instructional content is framed by a debriefing process (Garris et al., 2002; Landers, 2014). Differing from this, the intention of this research leaned towards the position made by Tay (2010) who argues that the goal of implementing game elements is not to teach but rather to work in tandem with instructional content with the aim of influencing engagement and thereby improve learning.

Theory of Gamified Learning. The lack of representation of game elements in current theoretical models, especially in those popularly used, led Landers (2014) to propose a theoretical model that is specific to gamification, and based on serious games.

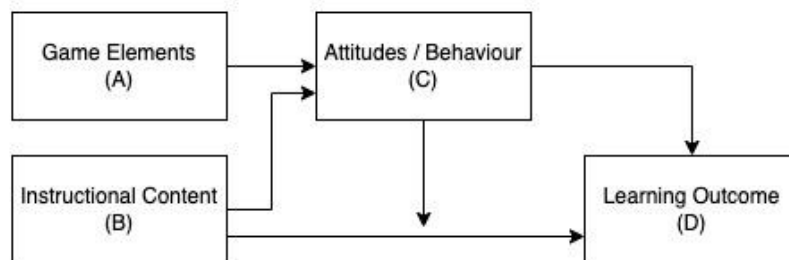


Figure 8 *Theory of Gamified Learning (Landers, 2014)*

Based on five propositions (Figure 8) where the directional arrows indicate a theorised path of causality, the theory of gamified learning was developed from the overall concept of gamification in serious games and focuses on the identification and use of game elements to existing instruction (Landers, 2014; Nair & Mathew, 2019).

- *Proposition 1: Instructional content influences learning outcomes and behaviours.* This suggests that instructional content affects a student's ability to learn. Therefore, the central role of gamification is not to replace instruction, but rather to improve it.
- *Proposition 2: Behaviours and attitudes affect learning.* It implies that learners' behaviours and attitudes can directly influence their ability to learn. Previous research remarked that learning is directly proportional to the level of cognitive effort and engagement students put into their work (Carini et al., 2006; Paas et al., 2005). Consequently, gamification is theorised as providing positive behaviours and attitudes that are likely to translate into improved learning.
- *Proposition 3: Game elements influence changes in behaviour and attitudes.* This implies that using specific game elements or a combination of game elements can affect student behaviours and attitudes. For instance, both O'Neil et al. (2005) and Wilson et al. (2009) theorised that gamification influences cognitive and behavioural attitudes that will in turn affect learning, interactivity and most importantly, student's motivation for learning.

The final fourth and fifth propositions are the key elements of this model, where Landers (2014) makes a distinction between two pathways: mediating and moderating.

- *Proposition 4: Game elements affect behaviours and attitudes that moderate instructional effectiveness.* Identified by the pathway of C -> B -> D (Figure 8), the goal of this proposition is "moderating" the relationship between instructional content and learning outcomes using game elements. According to Baron and Kenny (1986), moderation is present when the effect of one variable on another depends upon the value of the moderating variable. In other words, the moderating variable would not independently influence the outcome but rather either strengthen or weaken its interrelationship. In the case of gamification, Landers (2014) argues that

instructional content (moderating variable) must already be effective, making the incorporation of game elements a motivating factor to increase participation.

- *Proposition 5: The relationship between game elements and learning outcomes is mediated by behaviours and attitudes.* Identified by the pathways of A → C → D and B → C → D (Figure 8), this mediating pathway depicts the casual relationship where changes in the independent construct can significantly account for variations in the dependent construct (Baron & Kenny, 1986). Landers (2014) relates this to gamification by hypothesising that the mediator – game characteristics – would likely affect learning outcomes but only because game characteristics affect behaviour and attitudes, and behaviour and attitudes also affect learning outcomes.

Like the other theoretical foundations highlighted in this section, the Theory of Gamified Learning is limited in the number of available empirical validation, with only 3 articles found that implemented this theory as its theoretical foundation. One study was done by Landers and Landers (2015), where the authors evaluated the game element – leaderboards – as a strategy to influence the amount of time students spend on performing a task. Denny et al. (2018) using the mediating pathway, assessed the gamification elements of points and badges on self-testing as a means of evaluating exam performance. Garcia-Marquez and Bauer (2020) using the moderating pathway examined the game attributes of assessment and progress on self-efficacy as a means of evaluating learning outcomes. It is worth noting that the authors extended Landers theory by exploring goal orientation as the moderating variable.

According to Denscombe (2014), the true value of a case study lies in giving equal attention to the relationship and processes that led to the outcome, rather than focusing solely on the outcome itself. For this, the theory of gamified learning provides one of the best frameworks for evaluating *why* and *how* game-based learning affect learning related behaviour and its impact on learning outcomes in a mediating or moderating process. The decision of whether to adopt this framework came down to evaluating two of its shortfalls. Firstly, the

theory does not define a construct for defining behaviour. This was a minor factor and would have been remedied by defining learning behaviour using an existing student engagement framework. Thus, the deciding factor was based on the second shortfall – it does not connect what game elements affect which state of engagement. One of the study’s objectives is to inform my professional practice, which means it would be important to identify the specific game elements that were or were not effective influencers on learning behaviour and the achievement of the learning outcomes of the course. This was instrumental for determining the plausibility of the implementation of GBL within the programme. As a result, the decision was made to frame this research on the Gamification for Student Engagement Framework, rather than the Theory of Gamified Learning.

Gamification for Student Engagement Framework. Authors Rivera and Garden (2021) developed the Gamification for Student Engagement Framework to address the shortfalls of Landers’ theory by outlining a comprehensive definition for student engagement, connecting the theory to game attributes, and more importantly linking these attributes to learning outcomes. At its topmost part, Rivera and Garden (2021) takes Kahu’s (2013) students engagement framework of engagement antecedents, engagement state, and engagement consequences and applies it to the mediating pathway of Landers’ (2014) theory of gamified learning (Figure 9).

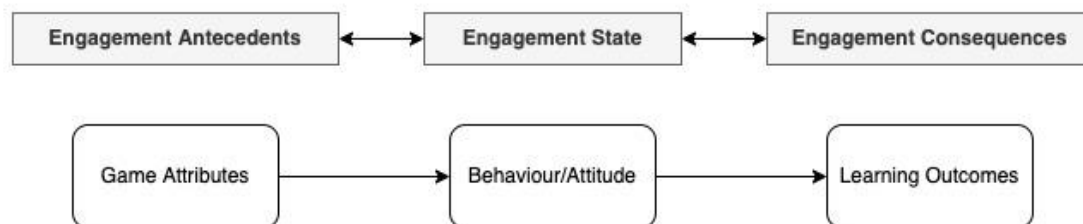


Figure 9 *Gamification for Student Engagement Framework - Application of Student Engagement Framework (top line, Kahu 2013) and Theory of Gamified Learning (bottom line, Landers 2014)*

After Kahu (2013), the framework encompasses six elements: structural and psychosocial influences, student engagement, and proximal and distal consequences, all

encompassed within sociocultural influences (Figure 10). According to Kahu (2013), both structural and psychosocial influences (antecedents of engagement) are characterised by university and student relationships from differing perspectives. Structural influences include characteristics such as curriculum, culture, assessment (university), and background, support, family (student). Psychosocial influences include features such as teaching, support, workload (university) and motivation, skills, and identity (student). In reconciling this with the game attributes component of Landers' theory of gamified learning, Rivera and Garden (2021) explained that game attributes connect to university structural influences, that is, the curriculum design that implements game attributes, this in turn influences student engagement (Kahu, 2013).

Relating to engagement (engagement state), the framework's view sought to incorporate all aspects of the students' experience that spans affective, behavioural, and cognitive domains. Taking this view of engagement can greatly extend the understanding of how games affect student learning (Rivera & Garden, 2021). Fredericks et al. (2004) defines affective, behavioural, and cognitive engagement as:

1. *Affective or emotional engagement* comprises students' reactions to teachers and peers, as well as the willingness to do the work.
2. *Behavioural engagement* includes participation and involvement in academic activities.
3. *Cognitive engagement* encompasses the effort students place on learning activities necessary to comprehend complex ideas and difficult skills.

Landers' view of engagement was approached from the perspective of cognitive engagement. Even though Kahu's view of engagement is far more complex than Landers', both recognise engagement as complex and far-reaching, making the incorporation of state of engagement (Kahu) and behaviours/ attitudes (Landers) easily reconcilable.

The consequences of student engagement, from the point of view of Kahu includes proximal and distal consequences (engagement consequences). Proximal consequences comprise the obvious academic outcomes of engagement including some measurable form of achievement, and overall satisfaction. While the academic outcomes are also reflected in distal consequences, it also considers long term effects of engagement, such as retention and work success. Though not as complex, Landers' view of the consequences of engagement considers the fulfilment of learning outcomes, an attribute parallel to the proximal consequences of Kahu's framework.

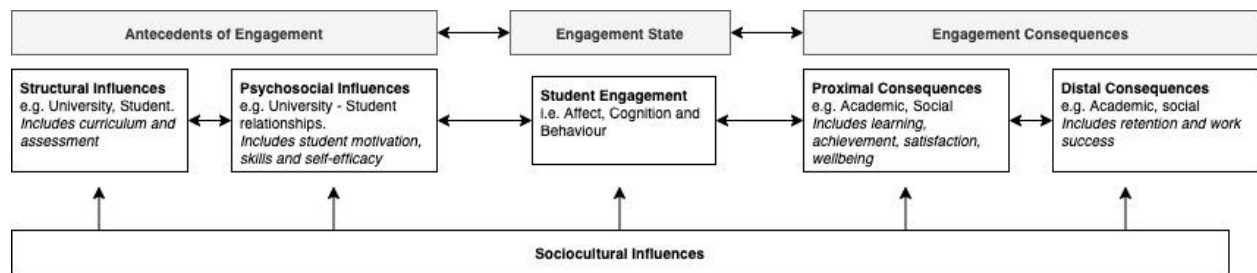


Figure 10 Student Engagement Framework (Kahu, 2013)

Antecedents of engagement, engagement state, and engagement consequences can all be potentially influenced by sociocultural influences. Kahu (2013) defines this as the impact of broader social context on student experience that includes disciplinary power, academic culture, and excessive focus on performance, all of which can possibly lead to a disconnection of students within higher education.

The key strength in adopting this framework is that it presents a detailed and holistic view of engagement, its relationship, as well as depicting the unique nature of students' experiences (Rivera & Garden, 2021). Applied to the theory of gamified learning, it allows for the exploration of game attributes modifying the structural, sociocultural, and psychosocial influences on student engagement. It also provides the opportunity to understand the impact from the perspective of 'how' and 'why'. A perspective that is often overlooked in current research relating to educational games and teaching and learning computer programming. For this, the framework outlines four propositions to test:

1. Gamification is the process through which student engagement states can be modified to support the achievement of learning outcomes.
2. The achievement of learning outcomes can be a measurable consequence of the state of student engagement which spans affective, cognitive, and behavioural domains.
3. It is possible to select game attributes appropriate to support the achievement of specific learning objectives categorised into the three domains of learning: cognitive, affective, and psychomotor.
4. It is possible to select a game attribute for employment in a gamification strategy by identifying the psychological domain shared between the learning outcome/educational objective and the desired, modifying student experience of engagement.

Application of the Gamification for Student Engagement Framework

Although the gamification for student engagement framework is specific to gamification, the principles were easily transferable to the context of GBL. In applying the framework, this research: (1) implemented game attributes into the introduction to computer programming course (structural influence), (2) reviewed students' prior knowledge in programming, (3) examined the impact of this on students' affective, behavioural, and cognitive engagement (engagement state), and (4) determined the consequences of this impact on students' performance (proximal consequences) (Figure 11).

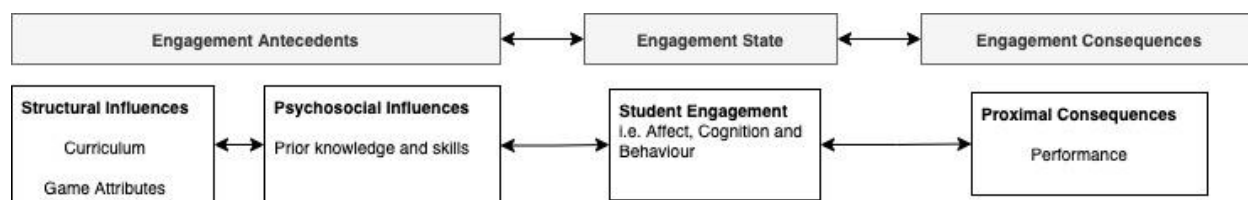


Figure 11 *Application of the Gamification of Student Engagement Framework*

The factors that included sociocultural influences, distal consequences, and psychosocial factors relating to teaching staff, student structural influences of background, and support were not included in this research. The time frame in which the research was conducted did not allow

for the exploration of these additional factors. As a result, only the factors directly related to the overall aim of this research was investigated; that is, how game attributes influenced student engagement and performance. Prior knowledge and skills in programming was included as a factor researched because of the debate on having strict requirements for entry into the programme.

Engagement Antecedents. Although it was noticed that students of the higher-level classes struggle with problem-solving and implementing solutions, the development of these skills should be nurtured at the very onset of their introduction to programming. Developing these skills at the introductory stage ensures that students can successfully design and implement solutions at the more advanced programming classes. Therefore, the Introduction to Computer Programming course was chosen as the case for this research because it is the first programming course introduced to students at the University of the Southern Caribbean. The course covers programming topics on problem-solving, algorithmic development and design as well as the fundamentals of the Python programming language that includes data types, logic control structures of sequence, selection, and functions (Table 4).

Topic 1	<ul style="list-style-type: none"> • Problem-solving and Algorithm development - pseudocode, flowchart, IPO charts, Desk checking • Debugging • Objects, methods, and arguments • Looping structures
Topic 2	<ul style="list-style-type: none"> • Variables and data types • Conditional structures and Boolean logic
Topic 3	<ul style="list-style-type: none"> • Advanced looping structures • Advanced control structures
Topic 4	<ul style="list-style-type: none"> • Functions

Table 4 *Introduction to Computer Programming Course Topics*

As previously mentioned, creating a game environment is difficult, time-consuming, costly, and requires technical infrastructure and pedagogical integration (Ahmad et al., 2019;

Kapp, 2012). Developing such an environment is beyond the scope of this research and instead existing platforms were considered. In evaluating the platforms, two criteria were topmost: the students' potential experience in programming, and programming language offered.

1. *Students' potential experience in programming.* Based on previous semester intake, many students enrolled in the Introduction to Computer Programming course have little to no experience in programming. Therefore, the platform chosen needed to sufficiently introduce students to programming at the most basic level.
2. *Programming language.* The programming language used for the Introduction to Computer Programming course was Python, therefore, the platform also needed to offer Python.

With these in mind, the platforms considered were:

1. *CodinGame.* This is a challenge-based platform that offers 25 languages, including Python. It encourages competitive and collaborative learning through its games. While the challenges were fun and engaging, it does not provide an order to learning and its more geared towards intermediate and advanced programmers.
2. *Code Academy.* This is an online, interactive platform that offers coding lessons and challenges in Python programming language. Unlike CodinGame, it is designed for beginners. Its lessons were hands-on and immersive with the inclusion of a sandbox (online code editor). However, the lessons were mostly based on written instructions and text-based learning.
3. *Ozaria via Code Combat.* This is a story-driven, adventure game where each level is unlocked by achieving specific goals through problem-solving and coding the solution. It is intended for beginners and the Python language is available. However, the platform is intended for student in grades 4 to 12 and coded solutions are not based on real-world applications.

4. *Check.io*. This is an interactive and educational and competition-based platform through gamified puzzles, for beginners and advanced programmers. However, the platform was often too difficult and at times confusing to navigate.

From the above evaluation, Ozaria was decided upon for use with the students of the Introduction to Computer Programming course. Although the platform's intended audience is much younger than those of the introductory programming class, it was still chosen because of its beginner focus for students with no programming knowledge. Also, the progression of the challenges was easily aligned to the course content of the introduction to computer programming class (Table 5).

Module #	Concepts covered	Aligned to topic/s
1	<ul style="list-style-type: none"> • Problem-solving • Sequences and algorithms • For loops • Debugging • Syntax • Objects • Methods Game Design 	1, 2
2	<ul style="list-style-type: none"> • Sequences and Algorithms • Syntax • Debugging • Variables • Boolean Logic • Conditionals 	1, 2, 3
3	<ul style="list-style-type: none"> • Data types • For loops • Iteration • Nesting • While loops 	1, 2, 3

4	<ul style="list-style-type: none"> • Functions • Compound conditionals • Comparators • Data and Analysis • Objects • Game Design 	2, 3, 4
---	--	---------

Table 5 *Aligning Ozaria Modules to Programming Topics within the Introduction to Computer Programming Course*

Ozaria also integrates two recommended strategies for implementing problem-solving activities in programming: problems that are ill-structured and challenging (Kay et al., 2000; Wu, 2010), and problems that are well-defined and uncomplicated (Bawamohiddin & Razali, 2017). For well-defined and uncomplicated activities, each of the levels of Ozaria presented students with structured challenges, and well-defined goals to achieve in order to progress through the game. For unstructured and challenging activities, Ozaria required students to develop a game through a capstone project. The project is a summative project that provide students with the opportunity to apply all concepts learnt in the module by creating a game strategy. An important aspect of this is that the project is not limiting and allows students to create unique solutions. This feature of learning to code through gameplay and learning to code by creating games was also another reason Ozaria was chosen for implementation.

In aligning with the works of both Rivera and Garden (2021), and Landers (2014), the game attributes were identified by the classifications outlined by Bedwell et al. (2012). Ozaria implemented five (5) attributes that had the potential of affecting learner engagement: action language, assessment, challenge, game fiction, and rules/goals (Table 6).

Category	Definition	Ozaria's Application
Action Language	Language with which the player communicates with the system	Students communicate with the game using the Python programming language (Figure 12)
Assessment	The nature and content of feedback provided by the game	Feedback is provided to students in the form of identifying and correcting syntax errors when the code is run (Figure 13)
Challenge	The difficulty and the nature of the problem within the game	As the students complete the goals, each level gets increasingly difficult, where students must problem-solve using a combination of previously learnt programming concepts
Game fiction	Using fantasy, this refers to how the game is presented to the player	Students are presented with a fantasy world, storyline, and characters (Figure 14)
Rules/Goals	Having clear rules, goals, and information on progression	For each level, students must complete each goal to unlock the next level. Students' progress is visually represented using a map (Figure 15)

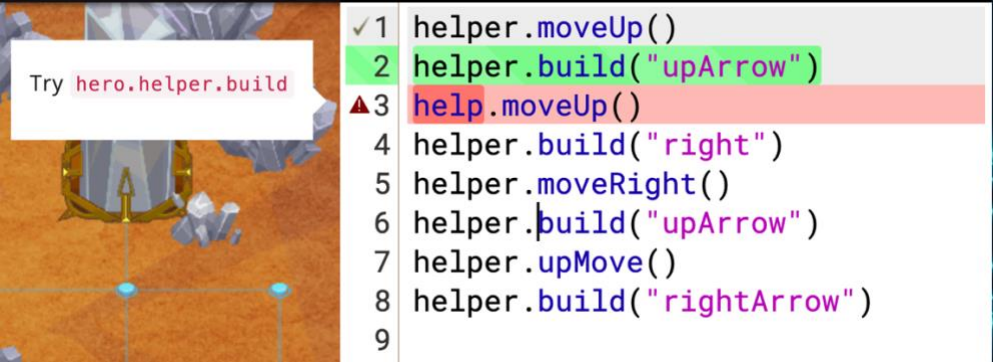
Table 6 *Game Attributes Implemented by Ozaria*

```

1 enemy = hero.findNearestEnemy()
2 steps = hero.getDistanceTo(enemy)
3
4 mouse.moveUp(steps)
5 mouse.bark()
6
7 mouse.moveRight(2)
8 mouse.digRight()
9 remainingSteps = 5 - steps
10 mouse.moveUp(remainingSteps)
11
12 hero.moveRight(1)
13 hero.jumpRight()
14 hero.jumpRight()

```

Figure 12 Ozaria - Action Language



The screenshot shows a game interface on the left with a hint box that says "Try hero.helper.build". On the right, a code editor displays a list of actions:

- ✓ 1 helper.moveUp()
- 2 helper.build("upArrow")
- ▲ 3 help.moveUp()
- 4 helper.build("right")
- 5 helper.moveRight()
- 6 helper.build("upArrow")
- 7 helper.upMove()
- 8 helper.build("rightArrow")
- 9

Figure 13 Ozaria – Assessment



Figure 14 Ozaria - Game Fiction

<p>Goals : Ran out of time</p> <ul style="list-style-type: none"><input type="checkbox"/> Use subtraction with a variable.<input type="checkbox"/> Get to the exit.<input type="checkbox"/> Get Mouse to the exit.	
<p>Goals</p>	<p>Map</p>

Figure 15 Ozaria - Rules and Goals

Engagement State. Students of programming must be able to grasp multiple knowledge and skills, the first of which is problem-solving, the design of the solution and finally translate the solution to code using their understanding and knowledge of the programming language. An important factor in the success of students is their behaviour and attitudes in learning. Based on the suggestions of the theoretical framework, this research used Bond and Bedenlier (2019) as a model for classifying student affective, behavioural, and cognitive engagement. For each of these, the authors provided a comprehensive list of indicators, and in acknowledging that students are influenced differently, these indicators were used as pre-codes for identifying the states of engagement from the participants reflections (Table 7).

Affective Engagement	Behavioural Engagement	Cognitive Engagement
Purposeful	Enthusiasm	Effort
Integrating ideas	Sense of Belonging	Attention/focus
Critical thinking	Satisfaction	Developing agency
Self-learning goals	Curiosity	Attendance
Self-Regulation	Sees relevance	Attempting
Operational reasoning	Interest	Homework completion
Trying to understand	Sense of well-being	Positive conduct
Reflection	Vitality/Zest	Action/initiation
Focus/Concentration	Feeling appreciated	Confidence
Deep learning	Manages expectations	Participation/Involvement
Learning from peers	Enjoyment	Asking teacher or peers for help
Justifying decisions	Pride	Assuming responsibility
Understanding	Excitement	Identifying opportunities/challenges
Doing extra to learn	Desire to do well	Developing multidisciplinary skills
Follow through/care	Positive interactions with peers and teachers	Support and encouraging peers
Positive self-perceptions and Self Efficacy	Sense of connectedness to school/university/within	Interaction (peers, teacher, content)
Preference for challenging tasks	classroom	Study habits/accessing course material
Teaching self and peers	Positive attitude about learning/values learning	Time on task/persistence
Use of sophisticated learning strategies		
Positive perceptions of teacher support		

Table 7 *Affective, Behavioural, and Cognitive Engagement Indicators as outlined by Bond & Bedenlier (2019)*

Engagement Consequences. Student performance was assessed using a programming test that evaluated three skill areas: problem-solving, design and syntactical knowledge.

1. *Problem Solving* – ability to understand, analyse and structure the reasoning.
2. *Design* – ability to communicate and diagram the solution to a given programming problem.
3. *Syntactical knowledge* – ability to use programming concepts in building a working and efficient solution.

Research Propositions and Questions

The overall research question sought to determine: “what is the impact of game-based learning on student engagement and performance in an introductory computer programming course?” For answering this overarching question, the gamification for student engagement framework proposed four testable propositions (pg.76), which were altered for the specific context of this research. Since most of the current literature reports positive findings, proposition one and two theorised that a positive impact would be found.

1. GBL and its game attributes are the process through which student engagement states (affective, behavioural, and cognitive) can be positively modified.
2. The perception of students’ knowledge can be positively altered by the state of engagement which spans affective, behavioural, and cognitive domains.
3. Game attributes support the achievement of learning objectives that span problem-solving, design and syntactical knowledge.
4. It is possible to select a game attribute for a GBL strategy by identifying the domain shared between the learning objectives and the desired modifying student experience of engagement.

To evaluate each of these propositions, four research questions were formed from each of the propositions outlined above:

1. What is the impact of game attributes on student engagement states: affective, behavioural, and cognitive?
2. To what extent does engagement influence students' perception of knowledge?
3. To what extent does game attributes support learning outcomes?
4. How does student engagement affect the relationship between game attributes and learning outcomes?

Outside the domain of the theoretical framework, this research also sought to address the debate of requiring students to have prior knowledge in programming as a prerequisite for entry into the program.

5. How does having prior knowledge and not having prior knowledge compare in affecting students' ability to learn programming?

Case Study Approach

Determining an appropriate case study approach initially placed this research in a complex position between explanatory and exploratory approaches. From the literature, an exploratory approach is one that can be used as a pilot to a larger study for the purpose of generating hypotheses (Cohen et al., 2018; Yin, 2018). A defining feature of this type of research is its focus on answering the question of “what” for the purposes of better understanding and gaining insights into the phenomena under study (Babbie, 2010; Yin, 2018). The exploratory approach is generally considered to be inductive and qualitative (Stebbins, 2001). In contrast, explanatory is often done when testing theories, seeking to answer the “how” and “why” questions, especially when little is known of the phenomena under investigation (Cohen et al., 2018; Swanborn, 2018; Yin, 2018). A key feature of explanatory studies is dealing with both surface and deep level explanations of the phenomena – the intent of which is to trace a process over time (Yin, 2018; Zainal, 2007). For this reason, the explanatory approach is often closely tied to hypothesis testing using deductive reasoning (Babbie, 2010).

While this research sought to answer “what” is the impact of gamification with the objective of communicating the outcomes, its intention is not to generate a hypothesis nor is it a pilot study, as in the case of exploratory research. Instead, guided by the theoretical framework that made assumptions, this research sought to test four propositions which are also aligned to characteristics of explanatory research. Therefore, the nature of this research required the deductive reasoning of the explanatory approach and the inductive nature of the exploratory approach. As a result, the approach adopted was *deductive exploratory research*. To do this conceptually, the propositions were treated as a ‘working hypothesis’; in other words, the propositions that guided this research were tested in action and were subject to change based on the findings (Casula et al., 2021). Adopting this approach not only delimited the area of research – keeping it focused – it also dictated the choice of data collection and data analysis strategies.

Data Collection

Framing this research as deductive exploratory influenced the decision to adopt a mixed methodology approach – collecting both quantitative and qualitative data.

Participation

All students enrolled in this class were eligible to participate in this research. Permission to use their data was sought from each student, and in most cases from their parents when the student was under the age of 18. Initially, 22 students agreed to have their data used for this study. However, the final number of participants was reduced to 18 students. Three students seemed to have dropped out of the program by mid-semester, and 1 student’s data was removed because their responses in three of surveys were often one word/one sentence responses and was considered not enough to provide the valuable insight being sought after. The same student also never completed the final survey and was unavailable for the focus group. With the 18 students, participation remained consistent throughout the multiple stages of data collection.

Data Collection Process

For this study, four methods of data collection were used: (1) pre/post-test, (2) questionnaires, (3) focus groups, and (4) observational notes. Data was collected using a parallel strategy, where it was gathered roughly at the same time, and continuously integrated in the interpretation of the overall results (Creswell, 2014). The process with which the data was collected was as follows:

1. At the start of the semester, students were given a programming test and a pre-course questionnaire that gauged their demographics, prior skills and knowledge in programming, and their perception of understanding in programming concepts.
2. At the end of each topic, a sample of the students participated in a focus group.
3. At the end of the semester, students completed a post-test and a post-course reflective questionnaire.
4. During the semester, observations were made on students' knowledge and progression, and a mid-semester questionnaire was given.

Pre/Post Test. Before the official start of classes, participants were given a programming problem for which they had one hour to design and implement a solution. The rubric for the exam evaluated students in problem-solving, design and syntactical knowledge (pg.86). At the end of the semester, the participants were given the same exam to verify whether they could successfully apply the skills, knowledge, and ability to problem-solve.

Questionnaires. In all, the participants were given 4 questionnaires that consisted of both closed and open-ended questions. The first was given at the start of the semester and their prior knowledge and experience in programming (Table 8). As mentioned, the first questionnaire determined students' prior knowledge and experience in programming. In this questionnaire, questions 2 and 3 were rated questions on a Likert scale that ranged from poor to excellent.

1. Do you have any prior knowledge in programming?
2. How would you rate this knowledge?
3. Do you have any knowledge in the Python programming language?
4. How would you rate this knowledge?

Table 8 *Prior Knowledge and Skills in Programming Questionnaire*

As part of the first questionnaire, participants were asked to rate their knowledge in the content areas covered in the course and was rated on the same Likert scale that ranged from poor to excellent (Table 9). This questionnaire was again administered to participants at the end of the course, as a method of self-assessment.

1. Rate your knowledge in the following areas:
Algorithms, Syntax errors, Logic errors, Debugging, Engineering Cycle, Variables, Variable arithmetic, Looping statements, Conditional Statements, Nesting statements, Compound conditionals, Objects, Methods, Arguments, Functionals

Table 9 *Rating Understanding in Programming Concepts Questionnaire*

The mid-semester questionnaire evolved based on the observations during the course. It was a self-assessment questionnaire that used a Likert scale that ranged from ‘strongly agree’ to ‘strongly disagree’ (Table 10).

Questionnaire for mid-semester self-assessment
1. I feel like my skills are improving
2. I feel enthusiastic about learning programming
3. I want to explore programming further
4. I am interested in learning programming
5. I am having fun learning programming

Table 10 *Mid-Semester Self-Assessment Questionnaire*

By the end of the semester, another questionnaire was administered to determine the students’ overall experience (Table 11) and evaluate each of the game attributes (Table 12).

1. I feel competent in programming
2. I feel the course was frustrating

Table 11 *Reflections on Learning Experience Questionnaire*

-
1. What are your thoughts on <action language, assessment, challenge, game fiction, rules/goals>?
 2. How has the <action language, assessment, challenge, game fiction, rules/goals> impacted your learning programming?
 3. What are your thoughts on the Capstone project?
-

Table 12 *Reflections on Specific Game Attributes Questionnaire*

Focus Groups. Four focus groups were conducted at the end of each topic to gain a deeper understanding of the students’ opinion of learning through the game elements and how they thought it affected their learning. Initially, the focus group was organised based on 22 students; however, due to 3 dropouts and the removal of the data for 1 student, the focus groups participants were adjusted.

- G1 – 5 students
- G2 – 4 students (1 student was removed)
- G3 – 5 students (1 student removed)
- G4 – 4 students (2 students removed)

The above-mentioned groups were intentionally formed so as not to limit the ability to gain as many ideas and perspectives from the students at any given time. While the students of groups G2 to G4 were randomly selected, the five students chosen for G1 were deliberate. It comprised students who indicated some prior experience in programming and rated their knowledge above fair yet scored below 10 (out of 30) in the pre-test. This focus group sought to understand the reason for this (Table 13). G1 and all other focus groups were also asked questions that triggered further discussion on their experience (Table 14).

-
1. What challenges did you have in developing an algorithm?
 2. What challenges did you have in designing the solution?
 3. What challenges did you have in writing the code for the solution?
-

Table 13 *Pre-Test Reflection Questionnaire - Students with Prior Knowledge*

-
1. Describe your experience in 4 words or less.
 2. What some negative aspects of the game?
 3. What are some positive aspects of the game?
 4. What game attribute would you say helped you learn?
-

Table 14 *Reflection on Engagement and Game Experience Questionnaire*

Observational Notes. As the semester progressed, observations were recorded on the students who agreed to take part in the study. This included observations on the students' performance, their progression in the game, and their understanding of the programming concepts as they submitted the assignments given.

Data Analysis

Descriptive statistics are numerical functions or graphical techniques used to organise and describe the characteristics of data (Fisher & Marshall, 2009). In working with descriptive statistics, missing data can affect the accuracy of the results (Cohen et al., 2018), hence the decision was made to remove the incomplete data from the 3 students that dropped out and the student that did not complete the questionnaire or take part in the focus group. Data for this research was represented using two methods: a measure of tendency (mode) and graphical representations.

1. *Mode* is defined as the most frequently occurring value in the dataset (Lee, 2020). Best used on categorical data, this research presented the mode for the engagement themes resulting from the thematic analysis of the students' responses in the focus group and questionnaires. This analysis was also used to present the students' rating of their perception of knowledge in the programming concepts being taught. Using this method allowed for determining which engagement states were the most impactful.
2. *Graphical representations.* Some of the data was represented using pie charts, as these were helpful for presenting the categorical data as proportions (Cohen et al.,

2018). Data relating to perception of knowledge (before and after) was presented using this method, making it easier to summarise the data in a simple visual form.

Paired Samples t-Test

The t-test is often used to uncover statistically significant differences between two groups or the same group under different conditions, and when the sample size is reasonably small – less than 30 (Cohen et al., 2018; King & Eckersley, 2019). For this research, the statistical difference was calculated between the same group of participants at two different periods in time (before and after) for both the participants’ perceived knowledge of the programming concepts and the pre/post-test. For perceived knowledge, ordinal data or counts was based on categorical data (Denscombe, 2014) – and was also used to calculate the statistical differences. Each of the categorical responses were coded as follows:

Categorical response	Code
(5-point scale)	
Poor	0
Fair	1
Good	2
Very good	3
Excellent	4

Table 15 Code for Categorical Responses

Using the codes (Table 15), a cumulative score for the scale items for both the pre and post questions of each student was calculated. Using SPSS, the total scores (variables: pre and post) were entered for each student, and the t-test calculated. The pre and post-test was calculated using the same process; the total scores for each was used for the paired samples t-test. The major criticism of significance testing is that on its own it does not give any indication of the impact of the phenomena or whether the findings are highly likely or important (Cohen et al., 2018). Determining impact is one of the core purposes of this research; as a result, significance testing was paired with thematic analysis.

Thematic Analysis

To explain the statistical data, a thematic analysis was used to analyse the qualitative data gained from the questionnaire, focus group, and observational notes. Thematic analysis is defined as a descriptive method for identifying, analysing, and reporting patterns within data (Castleberry & Nolen, 2018). An important characteristic of the thematic analysis is its ability to be used in both an inductive approach and deductive approach (Quintão et al., 2020). To give structure, the analysis was done using the process outlined by Braun and Clarke (2006) on the data obtained through the students' survey, focus groups, and observational notes (Figure 16).

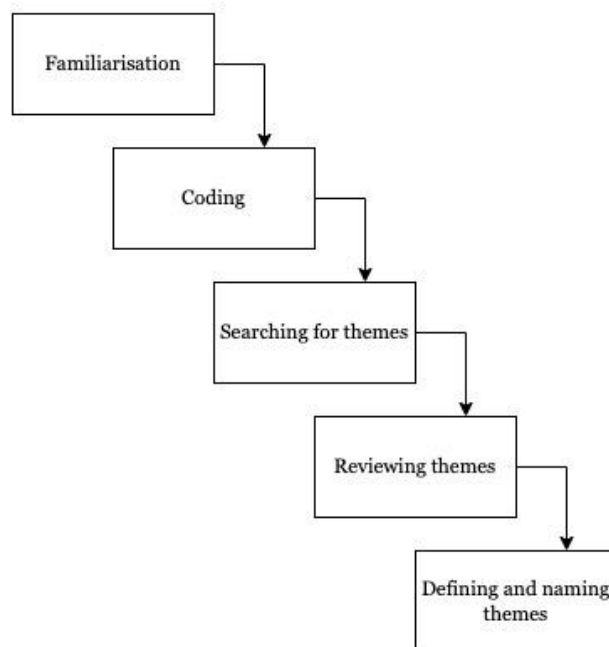


Figure 16 *Thematic Analysis Process (Braun & Clarke, 2006)*

1. *Familiarisation*. The data gained from the focus group was manually transcribed and combined with the data of the survey and observational notes. It was read, re-read and in-depth notes were made.
2. *Coding*. A concept driven approach to coding was used to identify the themes within the data, that is, most of the themes were gained from the research literature (Gibbs,

2018). For this, student engagement themes were pre-coded based on the indicators outlined by Bond and Bedenlier's (2019) – (pg.85).

3. *Searching for themes.* The data was read and re-read where keywords and various statements of interest were highlighted, and patterns identified.
4. *Reviewing themes.* The keywords and statements made by students identified in the previous phase were categorised and those that exhibited a good fit for a particular pre-defined code was identified. Therefore, both searching for themes and reviewing themes became an iterative process. To ensure the accuracy of the themes identified, the transcript (with identifying data removed) and its associated codes and themes were verified by two colleagues.
5. *Defining and naming themes.* Once the iterative process of searching and reviewing themes were completed, and all major themes were identified and rechecked again. Each was given a name and definition, and its associated data was extracted for presentation.

Legitimation rather than Validity

As mentioned previously, Onwuegbuzie and Johnson (2006), argued that the term 'validity' should be replaced by the term 'legitimation' for mixed methodology research. They outlined nine legitimation types that attempt to overcome the problems of mixed-methods research. These are: (1) sample integration, (2) inside-outside, (3) weakness minimisation, (4) sequential, (5) conversion, (6) paradigmatic mixing, (7) commensurability, (8) multiple validities, and (9) political. Each was considered and applied to this research.

Sample Integration. Explained as the extent to which the relationship between qualitative and quantitative sampling yields quality inferences. For this research, the same samples were used for both quantitative and qualitative methods: all students who consented completed a pre and post-test, questionnaires, and focus group.

Inside-Outside. Defined as the extent to which the researcher accurately describes and explains the insider and observers view. Considering the nature of the problem being addressed, it was natural that this research targeted the institution that I am currently employed at, and the students enrolled in the introductory programming class during an official semester for which I was their lecturer. As an insider researcher, I had both a direct connection and involvement to the research environment and had a similar background to the participations (Greene, 2014; Saidin, 2016). On one hand, this presented the advantage of having an already established understanding of the nuances of learning programming, leading to a lower chance of misinterpreting, or taking students comments out of context. At the same time, there was also the concern that this familiarity could potentially influence how the data was interpreted. That is, reporting data that favoured my preconceived notions and biases and taking for granted possible patterns because of my familiarity with the topic.

Another concern arising from the insider role was the possibility of unintentionally influencing student responses during the interviews. As their lecturer and interviewer, there was the potential for students to speak positively of game-based learning and its influence on their learning, rather than be entirely truthful. For this, careful attention was made to not share my own experiences or contribute to the conversation. This meant allowing the discussion to evolve naturally as students detailed their experiences. Input from myself was only done to ensure that the interview remained on topic or to ask for clarification on some comments made.

Balancing these concerns also meant taking on a dual role as the researcher. The quantitative data was collected using online questionnaires and surveys, and during this process my role was that of an outsider (objectivity). This role changed to an insider (subjectivity) for the interpretation of the data collected from the open-ended component of the survey, the focus group interviews and observational notes. This inside-outside legitimization was further established through reviewing of the data, its interpretation and overall presentation being reviewed by my supervisor to ensure that there were no premature or overstated conclusions.

Specific components, such as themes identified, and the statistical analyses were also reviewed by colleagues. It should be noted that the data provided did not contain any identifying characteristics of students, such as names. This ensured that the privacy, confidentiality, and ethical strategies outlined for this research was strictly followed.

Weakness Minimisation. Referring to the extent to which the strengths of one approach compensate for the weaknesses of another. The remedy of which is the main benefit of conducting a mixed-methodology research: triangulation. Triangulation is one of the most quoted benefits for undertaking a mixed-methodology approach to research because it produces a more thorough understanding of the phenomena under investigation (Almalki, 2016; Cohen et al., 2018; Yin, 2018). This understanding also includes uncovering any possible contradictions between the qualitative and quantitative data (Johnson et al., 2007).

Sequential. Denoting the extent to which problems with inferences were mitigated by reversing the sequence of quantitative and qualitative phases of data collection. This research used a parallel strategy to data collection and therefore, this step was not considered.

Conversion. Indicating the degree to which qualitative and quantitative data is analysed and represents quality inferences. For this research, qualitative data was analysed using a thematic analysis, while quantitative data was analysed statistically, using descriptive statistics and paired samples t-test.

Paradigmatic Mixing. Relating to the consideration of epistemology and ontology that underlies the qualitative and quantitative approaches. The ontology of this research was already considered and covered in the step inside-outside. Using a mixed methodology approach meant that this research did not fit into either a positivism or interpretivism philosophy.

Positivism seeks to identify causal relationships. It is often considered observable, measurable, and objective, and for this reason it is closely associated with quantitative methods (Ansari et al., 2016; Collis & Hussey, 2009; Doyle et al., 2009). The interpretivism relies on interpreting and understanding the meanings that humans attach to their actions, it is

subjective, and is associated with qualitative data (Ansari et al., 2016; Collis & Hussey, 2009). Addressing validity and reliability issues from either of these was not suited. Instead, the epistemology standpoint that underlined this research was pragmatism.

Pragmatism is based on the premise of solving practical problems rather than making assumptions about the nature of the knowledge (Hall, 2013). This research was conceptualised and managed based on the three principles outlined by Kelly and Cordeiro (2020).

1. Emphasis on actionable knowledge, meaning a desire to produce useful and actionable knowledge. This aligns with the purpose of this research, which is to inform my professional practice, that is, determining the possible benefit to students of a GBL environment, as well as recommending a game strategy that would be the most impactful on students.
2. Exploring the interconnectedness of experience, and knowledge through my own experience with both the organisation under research, and the experience of students (inside-outside, p.96).
3. Inquiry as an experiential process.

Pragmatism is based on the premise that the research questions are eclectic in its design, methods of data collection and analysis (Cohen et al., 2018). A hallmark of this research is based on a mixed design (use of qualitative and quantitative data), methods of data collection (pre/post-test, questionnaires, surveys, and observational notes) and analysis (thematic and statistical analyses). Using this approach afforded the benefit of providing the opportunity to combine macro and micro levels of understanding of the research issue, having qualitative data inform quantitative data, and vice versa; allowing for the corroboration and clarification of results, discovering contradictions and paradoxes, and having qualitative data inform quantitative data, and vice versa (Onwuegbuzie & Leech, 2005).

Commensurability. Referring to the extent to which inferences reflect a mixed view based on the cognitive process of switching and integration. For this research, this was achieved

by iteratively going between the qualitative and quantitative data sets to get a complete understanding of the impact of GBL from the perspective of the students, verify and report on whether a positive impact truly existed via the integration of these perspectives with student performance and more intentionally connect the results of this study to wider research.

Multiple Validity. Meaning the extent to which qualitative and quantitative data is gathered. As previously mentioned, data for this research was gathered using several methods that included questionnaires, interviews, observational notes, and pre/post-test.

Political. Denoting the extent to which readers accept the inferences of the mixed methods research. There was a deliberate attempt to ensure that the propositions were adequately answered using both the qualitative and quantitative data. Part of this process included updating the questionnaire based on the observational notes.

Ethical Considerations

One of the major ethical concerns was the collection and use of confidential and personal information from the participants. To ensure adherence to proper ethical guidelines, the following was be done:

1. Ethical approval was sought from Lancaster University and the University of the Southern Caribbean.
2. All participants were thoroughly informed about the intention of the research, the data that would be collected and how their data would be used and stored. Additionally, consent was gained from all participants before any data was collected. Ten students were under the age of 18, therefore, permission was sought from their parents.
3. All the data collected from the questionnaires, focus group interviews, observational notes, and pre/post-test scores were stored in a secure and encrypted device.

4. Participant names were excluded from the presentation of data, and every effort was made to not include any information that may have identified the students who took part in this research.

Another ethical concern is in the fact that the GBL environment was implemented in a professional and educational setting, using my own students as the participants of this research. To mitigate this concern, the GBL labs were not used as a means of grading, evaluating, or contributing to the students' academic standing with the university.

Chapter Four

Findings

This chapter presents the data collected; it does not attempt to interpret the data. The interpretation of the data is presented in the Discussion chapter (p.137). The chapter begins with presenting the psychosocial influences, that is, it reports on the students' prior programming knowledge to provide a better understanding of the representativeness of the background knowledge of the participants. It then presents the structural influences, as students' rating of the specific game attributes of Ozaria – action language, assessment, challenge, game fiction, and rules/goals. Relating to engagement states, the section 'Student Engagement' firstly identifies the engagement themes most frequently mentioned by students, classified as affective, behavioural, and cognitive engagement. Motivation, although not an intended variable under research was discovered to have occurred and is presented. Although the themes were pre-coded based on the works of Bond and Bedenlier (2019), the findings offer evidence of additional themes not included in the pre-coded list. Following this, it links the game attributes to the engagement themes which it impacted; and using the comments from students, it explains the reason for this influence. The chapter then reports on the measurable consequence of engagement using students' perception of knowledge before and after the course, using descriptive statistics and a paired samples t-test. Within this, it presents the students mid-semester reflections of their learning by considering the students' perceptions of their skills and enthusiasm in learning programming. After this, the students' overall performance is presented by comparing the results of the pre and post-test using both the descriptive statistics and a paired samples t-test. The overall performance is then analysed further by the learning outcomes: syntactical knowledge, problem-solving and design. Contained in this section, are students' perception of competency.

Prior Knowledge in Programming

The students were fairly divided in having prior knowledge in programming with 10 of students having some prior knowledge, and 8 not having prior knowledge. After examining the academic background of the participants with prior knowledge, only 8 of the 10 had formal education that derived over a two-year period from secondary school education. At this level, computing is not specialised in programming. The curriculum exposes students to a wide range of computing topics that include programming, basic operations of the computer, and computer hardware and software. From the findings of the focus groups, the remaining two students explained that their prior knowledge in programming was derived from self-study, using online tutorials prior to enrolling in the course. Among the 10 students that had prior knowledge in programming, 9 felt that they had a fair knowledge of programming, while 1 rated their knowledge as good.

Since the Introduction to Computer Programming course is strictly Python, those who had prior knowledge in programming were asked if they had any prior knowledge in Python. Six said they had no prior knowledge, while 4 did. Of the 4 that had prior knowledge, 1 classified this knowledge as poor, 1 as good, and 2 felt that their knowledge was fair.

Game Attributes

The comments – gained from the questionnaire – on each of the game attributes were used to determine whether they viewed the attributes positively, neutrally, or negatively. More importantly it identified the specific features of the game that made an impact.

Action Language

Referring to language with which players communicate with the game, Ozaria implemented this attribute through the Python programming language. 100% (N=18) of the students commented positively on action language. During the interviews, the discussion surrounding *action language* often centred around two main sentiments. Firstly, using Python in the game environment helped bridge the gap between what was learnt in class and developing

solutions in Python. Secondly, it provided a safe environment to practice programming through trial and error.

“It made it easier to get over more difficult concepts through trial and error.”

“Just class and then going straight to Python would have been intimidating, so the game bridged the gap and allowed me to practice in a safe environment.”

Assessment

Defined as the nature and content of feedback, Ozaria implemented this by providing feedback when debugging, that is, providing prompts to identify syntax and logic errors in the solution. This game attribute was rated mostly positively by 16 students. While none rated *assessment* negatively, 2 were neutral in their remarks.

The positive comments on *assessment* all stemmed from similar opinions. That is, the feedback in the form of prompts, notifications and underlined code was often deemed helpful in highlighting the syntax and logic errors. This made the process of identifying and fixing errors (debugging) easier as the course progressed, especially for errors that can be easily missed when developing solutions and, for some, when the in-class explanations seemed confusing. Despite mostly positive comments, the neutral comments made by the remaining 2 students were varied. For one, the displeasure came from having to do further research to fix the errors. For the other, the feedback was a hindrance to progressing because technical problems meant having to quickly catch up with the rest of the class. This student noted that had this not been the case, their rating of the attribute may have been different.

Positive

“...all the feedback and prompts has helped by letting me know where my error is and what needs to be done to fix is.”

Neutral

“It was not entirely helpful, but at times it was. It highlighted the issue, but I had to research and come back to it again.”

“It helped, but it could have helped more and my reason for saying this was because I was a bit behind due to my computer being down. If it wasn’t, it might have been better.”

Challenge

Meaning the merging of programming concepts. As the game progressed, comments on this attribute were fairly dividing, with 10 comments being positive, while 8 were negative.

The most mentioned programming concepts that students struggled with were variables, conditionals, looping statements, and functions. The negative comments surrounding *challenge* was mostly focused on the difficulties in trying to use these concepts when developing solutions. Especially since each challenge became increasingly difficult by requiring the merging of multiple programming concepts to develop solutions. Students who spoke positively about *challenge*, also made similar comments about having the same difficulties; however, what gave them a different and positive outlook, was the appreciation that the challenge encouraged them to pause and analyse what needed to be done.

Positive

I would say that my favourite part was that it made me really stop and analyse what had to be done.

Negative

The parts with using loops made me struggle a bit.

Game Fiction

Students played Ozaria through a fantasy world complete with storyline, and characters. Although this attribute was mostly met with positive comments by 16 students, 1 was neutral in the comments, and another commented negatively.

The positive comments on *game fiction* were directly linked to the attention-grabbing nature. It is this nature that ensured that students remained focused on learning programming. Despite this, the visual representations did not have the same effect on all. One remarked that they tried to skip the storyline, when possible, throughout the game. The further comments on this read as having a lack of interest in going through a storyline while learning. For another, there was an acknowledgement that the storyline was mildly appealing, but in reflection was unable to comment further on how it helped their learning process.

Positive

A fun interactive way making me want to play

Neutral

"The storyline was okay. It's not dull or WOW, so honestly I don't know if it helped."

Negative

“I don't necessarily have thoughts on it, I normally try to skip it.”

Rules/Goals

Visually represented as a map, students had to complete specific goals to unlock each level and progress throughout the game. Like action language, 100% (N=18) commented positively on *rules/goals*. Many of the comments concerning *rules/goals* were similar, in that students felt that it provided them with clear objectives to achieve, which focused their attention on the problem given, especially when applying newly learnt programming concepts.

“...it helps, gives us direct objectives to achieve, helps us focus on our problems.”

“Having goals to meet is important to me to make sure I was applying the different techniques and principles they were trying to convey.”

Student Engagement

The analysis of the data relating to how game attributes influence engagement was answered in two stages: (1) it identified and defined the engagement themes frequently mentioned by students, and (2) it identified the game attributes that students found to have influenced their learning programming.

Engagement Themes

For this research, student engagement was defined within the scope of affective, behavioural, and cognitive engagement. Where affective engagement is the emotional responses to learning, behavioural means the actions during learning, and cognitive refers to the effort in learning. The themes were derived from the student responses and were pre-coded using the indicators outlined by Bond and Bedenlier (2019). Overall, three themes were identified under

affective engagement, two for behavioural engagement, and one was identified for cognitive engagement. The themes and its definition are as follows:

1. Affective Engagement
 - a. *Fun/Enjoyment*. Having fun while playing the game.
 - b. *Interest*. Wanting to learn more about programming.
 - c. *Sees relevance*. Ability to make the connections between the in-class explanations, the game programming practice, and the assignments.
2. Behavioural Engagement
 - a. *Persistence*. Continued efforts in learning even in the face of difficulties.
 - b. *Asking peers for help*. Asking for assistance and working with other students.
3. Cognitive Engagement
 - a. *Understand*. Ability to grasp the programming topics being taught.

Although the engagement themes were pre-coded, there are two that emerged outside of the list.

These were:

4. *Frustrated*. Feelings of distress or annoyance with the content or learning process.
5. *Tedious*. Continuously revisiting/repeating the programming concepts to the point of annoyance.

The interview responses revealed that *fun/enjoyment*, *understand*, and *frustrated* were the topmost frequently mentioned engagement themes. With *understand* occurring 25 times, *fun/enjoyment* occurring 18 times, and *frustrated* occurring 12 times. Occurring a fair number of times was *persistence*, *tedious* and *see relevance*, with 10, 8 and 6 occurrences, respectively. Both *interest* and *asking peers for help* were the least mentioned theme, with only 5 and 3 mentions respectively (Table 16).

Themes	Occurrences
Understand	25
Fun/Enjoyment	18
Frustrated	12
Persistence	10
Tedious	8
Sees relevance	6
Interest	5
Asking peers for help	3

Table 16 Frequency of Engagement Themes

Linking Game Attributes to Engagement Themes

As the gamification for student engagement framework suggested, game attributes influence engagement states: affective, behavioural, and cognitive. While it was important to map the specific game attributes to the engagement state it impacted, it was also essential to understanding the reasons - the *how* and *why* this occurred. The students' statements were valuable in this regard. Based on these, engagement themes were seen to have been influenced by either a single or a combination of game attributes. Table 17 shows the engagement themes found and the game attributes that influenced it:

1. *Fun/Enjoyment* -> *Game Fiction*
2. *Interest* -> *Game Fiction*
3. *Sees Relevance* -> *Action language*, *Challenge*, and *Rules/Goals*
4. *Persistence* -> *Challenge* and *Rules/Goals*
5. *Asking peers for help* -> *Challenge*
6. *Understand* -> *Action language*, *Assessment*, *Challenges*, *Game Fiction*, and *Rules/Goals*
7. *Frustrated* -> *Action language*, *Assessment*, *Challenge*, and *Rules/Goals*
8. *Tedious* -> *Action language*, *Challenge*, and *Rules/Goals*

	Fun/Enjoyment	Interest	Sees Relevance	Persistence	Asking Peers for help	Understand	Frustrated	Tedious
Action Language			✓			✓	✓	✓
Assessment						✓	✓	
Challenge			✓	✓	✓	✓	✓	✓
Rules/Goals			✓	✓		✓	✓	✓
Game Fiction	✓	✓				✓		

Table 17 *Game Attributes and the Engagement Themes it Influenced*

Understand. Recognised as the ability to grasp the programming topics being taught, *understand* had the most occurrence of mentions at 25. Most students generally agreed that the GBL approach helped them grasp the programming concepts taught. Expanding this furthershowed that it was the only engagement theme that students accredited all the attributes (*action language*, *assessment*, *challenge*, *game fiction*, and *rules/goals*) in some way to their understanding. Students mostly rated *assessment* positively on the game's ability to point out both the syntax and logic errors in the code. It is this very feature that students attributed to the engagement theme *understand*. According to most, the identification of errors, both syntactical and logic helped them better grasp the rules of the programming language, and thereby assisted in the identification of errors and in understanding their mistakes. For some students, this also meant removing the pressure of having to ask the lecturer for assistance, and even encouraging them to figure out the solution on their own.

Yes, this game has helped me in grasping the concepts of programming, since it helps debug the code entered to ensure that are no syntax and logical error, and if per chance there are any, it would point it out and allow me rectify, and correct it.

The bonus of pointing out all your errors because I struggled a lot with syntax errors previously, and instead of having to have an instructor constantly telling me, the game tells you.

I didn't want to ask, I really wanted to figure it out on my own

This did not mean that students did not require the lecturer's assistance. Some commented that even though the game made it easier to understand, at times, this understanding required the intervention of the lecturer.

With my lecturer's help and proper reading, I was able to understand it.

Although game fiction was proclaimed by students as being attention grabbing, linking the attribute to the engagement state showed two additional features that were impactful for students. Firstly, the use of the storyline and characters to explain the programming concepts as they played through the levels. This according to students helped them remember the programming concepts learnt.

“Yes, it did help me quite a lot due to how entertaining it was. I was able to easily remember all the different concepts.”

“...how they explain the code is very informative.”

“New programming concepts were explained. Some concepts that I thought were difficult became much easier after interacting with the game. I got a real in depth and informative explanation whilst having fun.”

Secondly, game fiction combined with action language also positively influenced the engagement state *understand*. Just as the students commented on the attribute action language, the feature of bridging the gap between the game and actual coding was instrumental in their understanding of the programming concepts. With game fiction students isolated the interface in this regard, that is, the characters, environment, and movement. In combining these two features, students mentioned gaining a better understanding by seeing how the code they wrote translated into the movement and actions of the character on screen. This helped them to conceptualise how the different programming syntax works, such as conditional and looping statements.

It has given me a more illustrated view of programming and coding concepts since I am able to easily see and understand the result of any code, I write.

The findings also showed that understanding was not limited to grasping the programming concepts, but it also extended to students enhanced problem-solving and design skills. In this regard, the comments revealed that a combination of the attributes *challenges* and *rules/goals* were the most impactful. Although *challenge* was met with mixed responses and was mostly associated with frustration, some positive comments centred around the feature of encouraging them to analyse the problem presented to develop a solution. The feature of *rules/goals* which students reflected positively, remains the feature that they attributed to improving their problem-solving and design skills, that is, it encouraged them to focus on the developing and designing solutions to fulfil the goals. In associating this with *understand*, most commented on the activities giving them the opportunity to analyse and develop solutions, and as the goals became more complex by integrating more programming concepts, it developed their ability to problem-solve and design solutions to meet each goal.

“The game presents a series of different obstacles on the same topic. This shows me that there are many ways problems can arise and it also forces me to think of a solution for each.”

“The more difficult levels required more goals allowing the use of multiple concepts which made problem solving key to completing the levels. All these things helped improve my problem-solving skills.”

Fun/Enjoyment. Defined as having fun while playing the game, *fun/enjoyment* had the second most occurrences – at 18. *Game Fiction* was the most quoted attribute to have influenced *fun/enjoyment* and, just as mentioned, it was a result of its attention-grabbing

nature. The findings showed that students felt frustrated with the learning process (p. 114). Despite this, the mid-semester reflection questionnaire showed that most agreed that Ozaria provided them with a learning platform that made learning programming fun and enjoyable. With 11 students strongly agreeing, 2 agreeing, 1 neutral, and 1 disagreeing.

Although most students agreed that learning programming through Ozaria was fun, there was 1 student who was neutral, and another disagreed with the statement that the game was fun. The student who was neutral was the same student who mentioned being mildly interested in the storyline and attempted to skip it when possible. The student that disagreed was the same that had technical difficulties during the semester.

Interest. Linked with the engagement theme *fun/enjoyment*, *interest* was defined as wanting to learn more about programming after having played the game. This theme was the second least mentioned with 5 occurrences. Much like, *fun/enjoyment*, students mentioned this mostly in relation to *game fiction*, particularly the audio/visual features (storyline, characters, music...etc.).

“The amazing storyline, visuals, and music made for a super cool learning experience. I’ve always felt like I wanted to learn more.”

Many of the comments relating to the increased interest in programming, also showed its close link to *fun/enjoyment*. That is, the enjoyment these students gained from the *game fiction* led them to feel interested enough to continue to play and learn programming.

“It kept me entertained which made me want to keep learning.”

By the mid semester, the feeling of wanting to explore programming continued to be true for most students, despite already expressing feelings of frustration and finding the learning process tedious. When asked, 10 students strongly agreed to the statement that they felt like

they wanted to continue learning programming, 7 agreed, and 1 was neutral. The student who remained neutral in their comment, was the same that found the course frustrating overall.

Frustrated. Defined as feelings of distress with the learning content or process, *frustrated* was the third highest occurrence at 12. Several comments alluded to a combination of *action language*, *assessment*, *challenges*, and *rules/goals*. The negative remarks surrounding *challenge* is as mentioned; they had struggles with developing solutions, especially when it combined programming concepts that were initially difficult to grasp. When discussing their challenges, the students commented on having difficulties with using the programming concepts variables, conditional statements, looping statements, and functions. *Action language* in this case was not about bridging the gap or having a safe environment to practice, but in using the language. As shown previously, the evaluation of the single attribute *assessment* saw mostly positive remarks regarding the assistance in the identification of syntax errors in the code, however, in discussing *challenges* it was revealed that this very feature was also a source of frustration among students. Overall, in using the programming language concepts (*action language*), students had difficulties in fixing errors, even though feedback was provided (*assessment*), causing a situation where students were often stuck on some levels – unable to fulfil the goals and progress further (*rules/goals*), particularly when the level became more difficult, with students needing to use a combination of the programming concepts learnt (*challenges*).

*“Some levels would frustrate me, because it would be the simplest error and I would spend like an hour on one level.” **

“I was getting stuck at a certain level for longer than 5 mins because a small error was made, or I didn't indent my code where needed.”

** This comment was found to be an exaggeration on the part of the student; there was no evidence that showed the student spent an hour on one level.*

The observational notes also support these claims. Relating to errors in debugging, errors were often made with the use of spacing, and misspelt syntax. Although they mentioned mostly the syntax errors in their comments, the observational notes showed that students also had difficulties with logic errors related to the use of looping and conditional statements, and functions. In other words, students often used these programming concepts incorrectly in such a way that it did not fulfil the goals of the level. In the game activities related to these, 12 students spent – on average– 27 minutes on the activities.

Even though students were *frustrated* with the learning process, the end of semester reflection questionnaire showed that the frustrations felt during the course did not have a lasting impact. Of the 12 that explicitly mentioned being frustrated, when asked about their overall experience, 2 strongly disagreed with being frustrated overall, 7 disagreed, 2 were neutral and 1 agreed that the course was frustrating. It was apparent from a few comments made by the 7 that disagreed, that the engaging nature of the platform, particularly its influence on *fun/enjoyment* and *understand* made the learning process less stressful.

“It provided a fun and less stressful environment for me to learn about the concepts presented. Programming is seemingly a difficult topic, but the game makes it easier to grasp the concepts.”

Persistence. Meaning the continued efforts in learning even in the face of difficulties, *persistence* had the fourth highest occurrence with 10. Although the students felt *frustrated* with the learning process, the game also led to their continued efforts in learning. The game attributes *challenge* and *rules/goals* were frequently mentioned as being the reason for their *persistence*. Relating to *challenge*, although students had difficulties, it was not something they saw as being negative, but rather, one which helped propel their understanding of the programming topics and made them determined to finish.

“...it was not bad challenging; it was good challenging.”

For other students, this *persistence* stemmed from a sense of satisfaction after overcoming the challenges the game presented to them. While one detailed their frustrations with the challenges, the student also added that there was a feeling of satisfaction after completing activities that were difficult, to which the group agreed.

“...but when you finally overcome the level, it's so satisfying”.

While some saw *challenge* as initiating feelings of *persistence*, others attributed *rules/goals* as the game attribute that impacted the same. For these students, having clear goals to achieve at each level allowed them to persist in learning in two ways. Firstly, it focused their ability to gain knowledge of the programming concepts, much like the reasons given for a positive outlook on *rules/goals*. Secondly, it offered students the ability to measure progress which gave them a sense of accomplishment rather than feeling discouraged with the learning process.

“It helped give us direct objectives to achieve, helped us focus on solving the problems.”

“I'll be finding myself going back to the map and seeing how much I had to do. It just makes me feel like, yes, I accomplished this.”

Tedious. Stemming from continuously revisiting the programming concepts to the point of annoyance, *tedious* had 8 occurrences. In discussing this, students generally agreed that *action language*, *challenges*, and *rules/goals* were the game attributes that contributed to their feeling of, “*going through the motions*”. While *challenges* saw students having difficulties in developing solutions using a combination of programming concepts, it was also revealed that

the repetition of developing similar solutions also triggered a negative response. Overall, students made it clear that the reiteration of using the same programming concepts at varying difficulty levels (*action language* and *challenges*) and having to meet similar goals (*rules/goals*) became monotonous.

“If you write the code and go to the next level and it's basically the same thing.”

“My only criticism is that at times it did start feeling a bit monotonous and repetitive.”

Despite these criticisms, the same students also saw it as beneficial. They understood that learning programming relies on repetition and helped in their understanding of programming concepts.

“I disliked the slight repetitiveness of some of game levels. However, I do not hate it since I understand that this is a method by which I ground the concepts that I learned into my memory.”

“When you're doing programming, if you already think about what programming truly is, it really is a repeat of certain stuff.”

Sees Relevance. Occurring 6 times, the engagement state *sees relevance* was seen to have been influenced by the combination of *action language*, and *rules/goals*, allowing students to be able to translate what they learnt in game to real-world solutions. The two reviews on *action language* saw students commenting that Ozaria provided an environment in which they were able to bridge the gap between what was learnt in the game and actual programming, and an environment in which they can practice, and learn through trial and error. It is these very features that most students linked to the engagement theme *sees relevance*. They reported being able to better apply the programming concepts learnt to real world problems after having played

the game. In applying these concepts to assignments, students mentioned revisiting levels of the game to refamiliarize themselves with the application of the language and then attempt the assignments.

“It has helped me when writing my code, because there were times I would go back to let the game explain how to do something I had some problems with.”

Despite a mostly positive influence, this was not the case for one student with no prior knowledge in programming, who mentioned having a difficult time in translating what was learnt and practiced in the game to coding and developing solutions to real-world problems.

“I didn't quite grasp using the codes that I learnt in the game, and then having to put it into Idle [programming development environment], it was kind of a tough transition, at least for me”.

Asking Peers for Help. Although this theme only appeared 3 times, it was still reported as it was not initially considered as a possible indicator. Because these students were first year students, meeting for the first time in a strictly remote class setting and scattered across the Caribbean region, it was not entirely expected that they would develop a good enough relationship for collaboration. However, there was a small group of 3 students who explained that the same *challenges* that some reacted to in *persistence*, also prompted them to reach out to their peers when stuck on the levels. As stated by one student: *“When I got stuck on the levels real long, it irritated me, but I would speak with [--] and [--] and they would help.”*

Motivation

Although not originally intended to be a focus of this research, motivation had 18 occurrences in the focus group interviews, with students mentioning being motivated to learn because of Ozaria. When asked to elaborate on why they felt motivated, 15 linked it to some form

of engagement. With comments crediting motivation to have fun while learning, while others attributed it to increased understanding.

“It provided a fun and less stressful environment for me to learn about the concepts presented. Programming is seemingly a difficult topic, but the game makes it easier to grasp the concepts.”

“...with the game, is that it allowed me to understand fully on the basic concepts of programming whilst being excited to see the storyline of the game as I progress through each coding method.”

The remaining 3 comments were all similar in their sentiment. Students discussed motivation in relation to having better understanding of the requirements and applicability of programming. For these, the motivation to learn stemmed from knowing that they can use the skills learnt to create something of their own.

“There are many things that can be done with programming, and being able to see how it's applied intrigued me.”

Perception of Knowledge

The perception of knowledge was gained from the questionnaire that required students to rate their knowledge in the programming topics: algorithm, syntax errors, logic errors, debugging, looping statements, objects, methods, arguments, engineering cycle, variables, conditional statements, variable arithmetic, nesting statements, while loops, compound conditionals, and functions. The questionnaire was given at the beginning of the course and at its end.

Pre-Evaluation of Knowledge

The general overview of students' perception of knowledge showed that the most frequently occurred rating was poor with 127, the second most frequent was fair at 93 (Table 18).

Rating	Frequency
Poor	127
Fair	93
Good	55
Very good	8
Excellent	5

Table 18 Frequency for Pre-Evaluation of Programming Knowledge

With a frequency occurrence of 127, poor was highest rating in 12 of the 16 topics. Similarly, with the second highest frequency rating of 93, fair was also the second highest rating in 8 of the 16 programming topics. The topic, syntax errors, was equal in its rating with 7 students rating their knowledge as poor and fair (Figure 17).

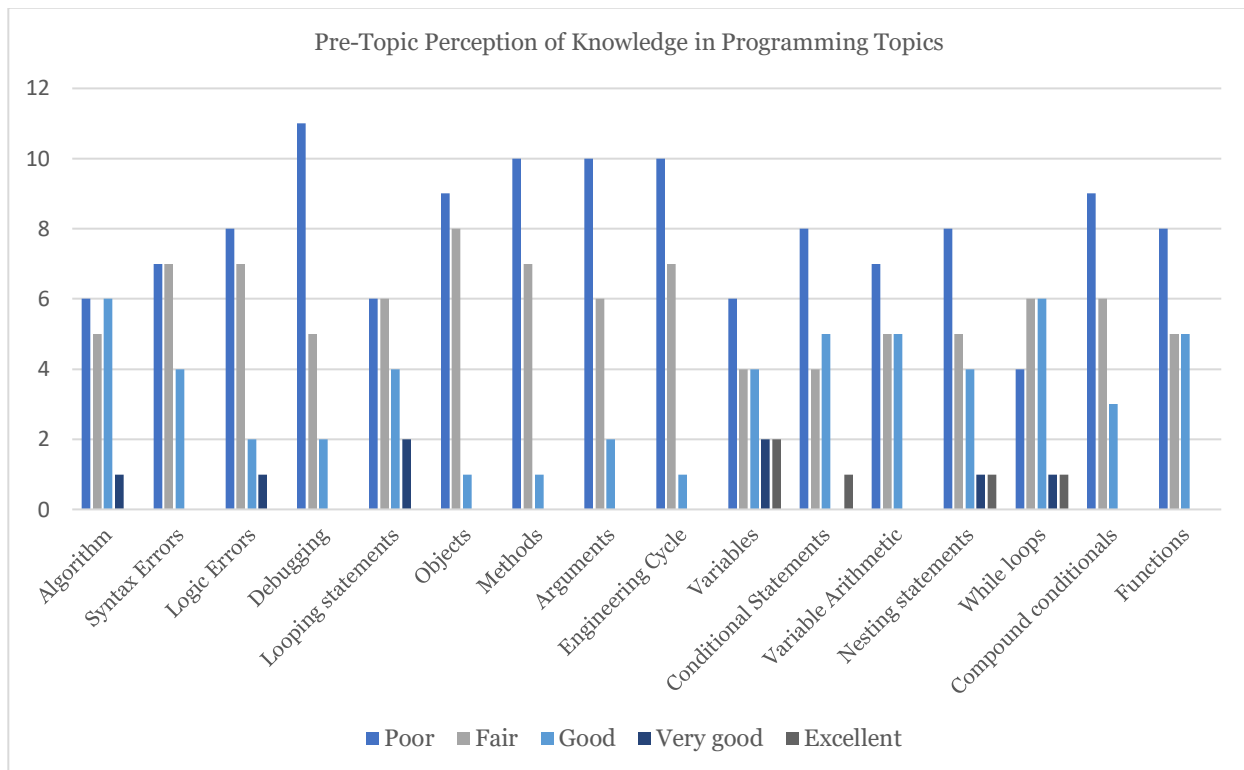


Figure 17 Pre-Evaluation of the Perception of Programming Knowledge

Using good as the lowest benchmark for what was considered a positive marker for knowledge, the responses were further analysed by students who had prior knowledge in programming and those who did not. Among the 10 students who had programming knowledge, it was surprising to discover that most were not familiar with more than half of the programming topics: syntax errors (6), logic errors (7), debugging (8), objects (9), methods (9), arguments (8), software development cycle (9). For the topics looping statements and functions, the students were equal in those who were familiar and those who were not. Most were more familiar with the topics: algorithms (7), variables (8), conditional statements (6), and while loops (7).

As expected, the students who were not familiar with programming mostly rated their knowledge as poor in all the 16 programming topics. However, though in the minority, an interesting occurrence was identified where some rated their knowledge as good in 4 programming topics: looping statements (1), variable arithmetic (2), nesting statements (2), and while loops (1). These students stated that prior to starting the course, they self-studied and understood some of the programming concepts. However, they did not consider this prior knowledge and therefore did not rate themselves as having such. As one stated, *“I don't really have experience in it because it's something I picked up three months ago.”* In another notable response, *“I researched videos on programming to get ready for class, so I had a good idea on the basic concepts. I managed with those but putting it into effect was more difficult, especially developing algorithms.”* Their learning originated from online videos and tutorials, and while they understood the programming concepts, they were unable to problem solve and by extension implement a solution. The evidence of this was further perceived where they positively rated their knowledge in some programming concepts, but with problem-solving techniques such as algorithms and the engineering cycle, they rated their knowledge negatively. Additionally, these same students were entirely unsuccessful in the pre-test.

Mid-Semester Reflection

By the mid-semester, the topics: algorithm, syntax and logic errors, debugging, looping statements, objects, and methods were already covered. In reflection, students generally had a positive outlook on the progression of their skills. All agreed that their skills were improving, with 10 agreeing to the statement, and 8 strongly agreeing.

Post-Evaluation of Knowledge

By the end of the course, there was a noticeable change in the way students perceived their overall knowledge. Where previously, poor, and fair were the two most frequently occurred rating at 127 and 93, by the end of the course, this was significantly reduced to a frequency of 1 and 24, respectively. The final ratings showed that after the course, the most frequently occurred rating became excellent, with 112 and the second was very good with 94 (Table 19).

Rating	Frequency
Poor	1
Fair	24
Good	57
Very good	94
Excellent	112

Table 19 *Frequency for Post-Evaluation of Programming Knowledge*

Topics such as logic errors, debugging, objects, methods, arguments, and the engineering cycle all saw a shift from a high poor rating to a high excellent rating. Conditional statements, variable arithmetic, nesting statements, compound conditions and function also saw a change from a high poor rating to a high very good rating (Figure 18).

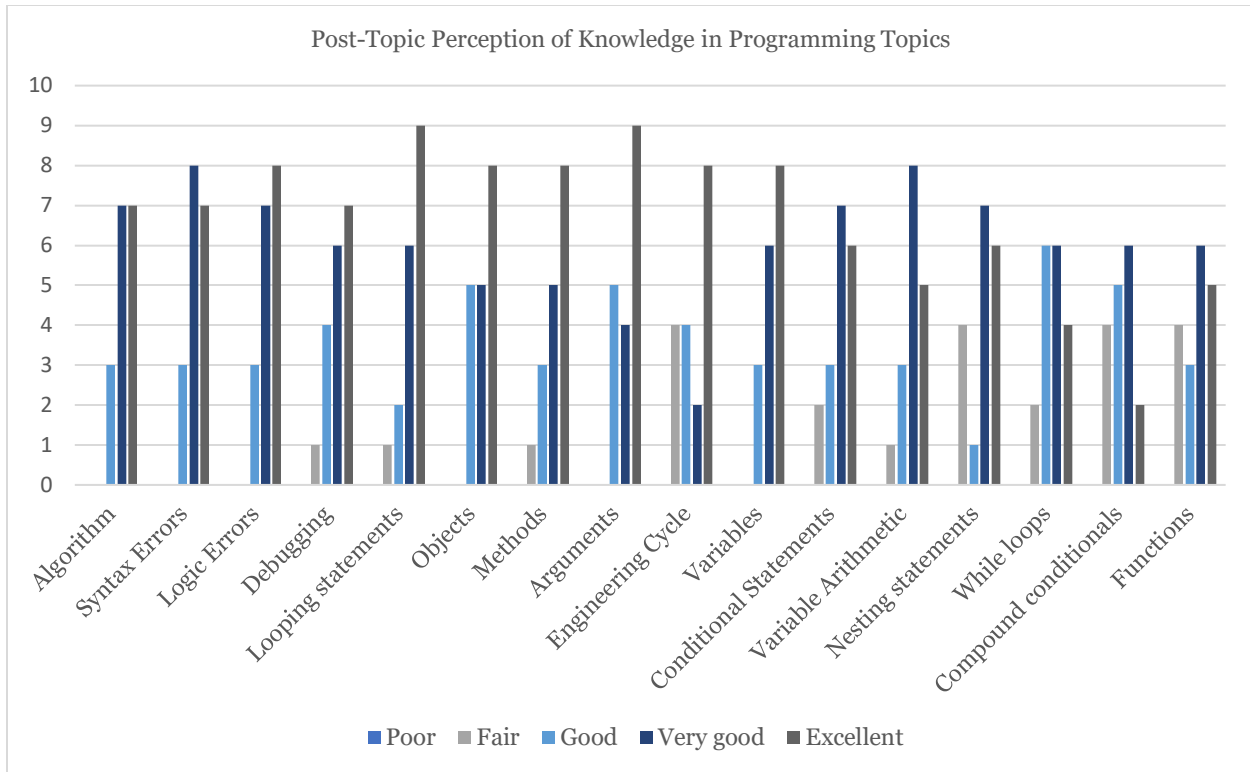


Figure 18 Post-Evaluation of the Perception of Programming Knowledge

Previously, students who did have prior knowledge rated their knowledge negatively in the topics: syntax errors (6), logic errors (7), debugging (8), objects (9), methods (9), arguments (8), engineering cycle (9). After the course, these topics showed that all rated their understanding positively.

The results for those who indicated having no prior knowledge in programming were more diverse. Initially, these students mostly rated their knowledge negatively in all 16 programming topics, but after the course, five of the topics – syntax error, logic error, objects, arguments, and variables, - saw all rating their understanding positively. The topics debugging (7), looping statements (7), methods (7), and variable arithmetic (7) changed to mostly a positive rating with only 1 negative rating each. Although a negative rating, it was a slight improvement from an initial poor rating (before) to fair (after). In the topic of debugging, the student’s understanding remained at a consistent fair rating both before and after. Second to this, were the topics – conditional statements (6), and while loops (6), that also saw a change to a positive

rating, with only 2 negative ratings each. Like the previous topics, there was only a minimal improvement from poor to fair in one response for the topics, conditional statements and while loops, while the other rating of understanding remained the same both before and after.

Other topics such as the engineering cycle, nesting statements, compound conditionals and functions saw a more divided response. Both nesting statements and functions saw a positive change in rating among 5 students and only 3 rating their understanding negatively at the end of the course. For nesting loops, two made a nominal improvement from poor to fair; however, one who previously positively rated their knowledge as good, after the course downgraded the rating to fair. For the topic of functions, there was a minimal improvement from poor to fair among 2 students, and one perceived no change in the understanding, rating it fair both before and after. The topics, engineering cycle and compound conditionals, were equally divided with 4 rating their understanding positively and 4 negatively. Among those that negatively rated their understanding, beginning with the engineering cycle, 2 perceived only a slight improvement (from poor to fair) in their understanding, while the other 2 viewed their understanding as having remained consistent. With compound conditionals, 3 perceived no improvement in their understanding and rated it the same before and after; however, 1 felt that their understanding had a slight improvement, rating from poor to fair.

Placing this difference using a paired samples t-test showed that there was a statistically significant difference in students' perception of their understanding of the programming topics before starting the topic and after completing the course. The results showed that students did perceive their understanding to be better after the gamified course ($M=48.22$, $SD=13.40$) when compared to when they first started ($M=13.72$, $SD=11.20$). This improvement, 34.50, was statistically significant, $t(17)=9.85$, $p<.001$ (Figure 19).

		Mean	N	Std. Deviation	Std. Error Mean
Pair 1	PreTopics	13.7222	18	11.20822	2.64180
	PostTopics	48.2222	18	13.40130	3.15872

		Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the Difference		t	df	Significance	
Paired Differences					Lower	Upper			One-Sided p	Two-Sided p
Pair 1	PreTopics - PostTopics	-34.50000	14.85716	3.50187	-41.88829	-27.11171	-9.852	17	<.001	<.001

Figure 19 Paired Sample t-Test - Perception of Knowledge

Performance

Performance was evaluated by testing students at the beginning of the semester and the same test was administered at the end of the semester. Each of the tests assessed students on their knowledge and skills in problem solving, design, and syntactical knowledge (coding), with a total of 25 points. 9 points for problem-solving, 6 points for design and 10 points for coding.

Pre-Test Scores

For the pre-test, students scored well below average with a score of 1.5. It was expected that the 8 with no prior programming knowledge would earn zero points, and this did occur. Initially, it was expected that those who had prior knowledge would have been able to, for the least, successfully problem-solve, and design the solution to the programming problem. However, this was not the case. Of the 10 students who expressed having prior knowledge in programming, 4 of these scored zero points, while the remaining 6 all scored significantly low in the pre-test. Of the 4 who noted having some prior knowledge in Python, none coded the solution – thereby earning zero points in syntax. Of the 6, (and out of a total of 25 points) the two highest scores were 9 points, while the remaining 4 students earned under 5 points. Although significantly low in the overall score, 5 of the 6 students attempted the problem-

solving and design questions. 2 were capable of decomposing the problem into sub-problems, earning 5 and 4 out of 6 points. The remaining 3 ranged from 3 points to 1 point. In the area of design, only 3 students were able to identify the programming concepts needed to develop an algorithm for the problem. One earned the highest score of 5 of 6 points, while the remaining 2 earned 4 points (Figure 20).

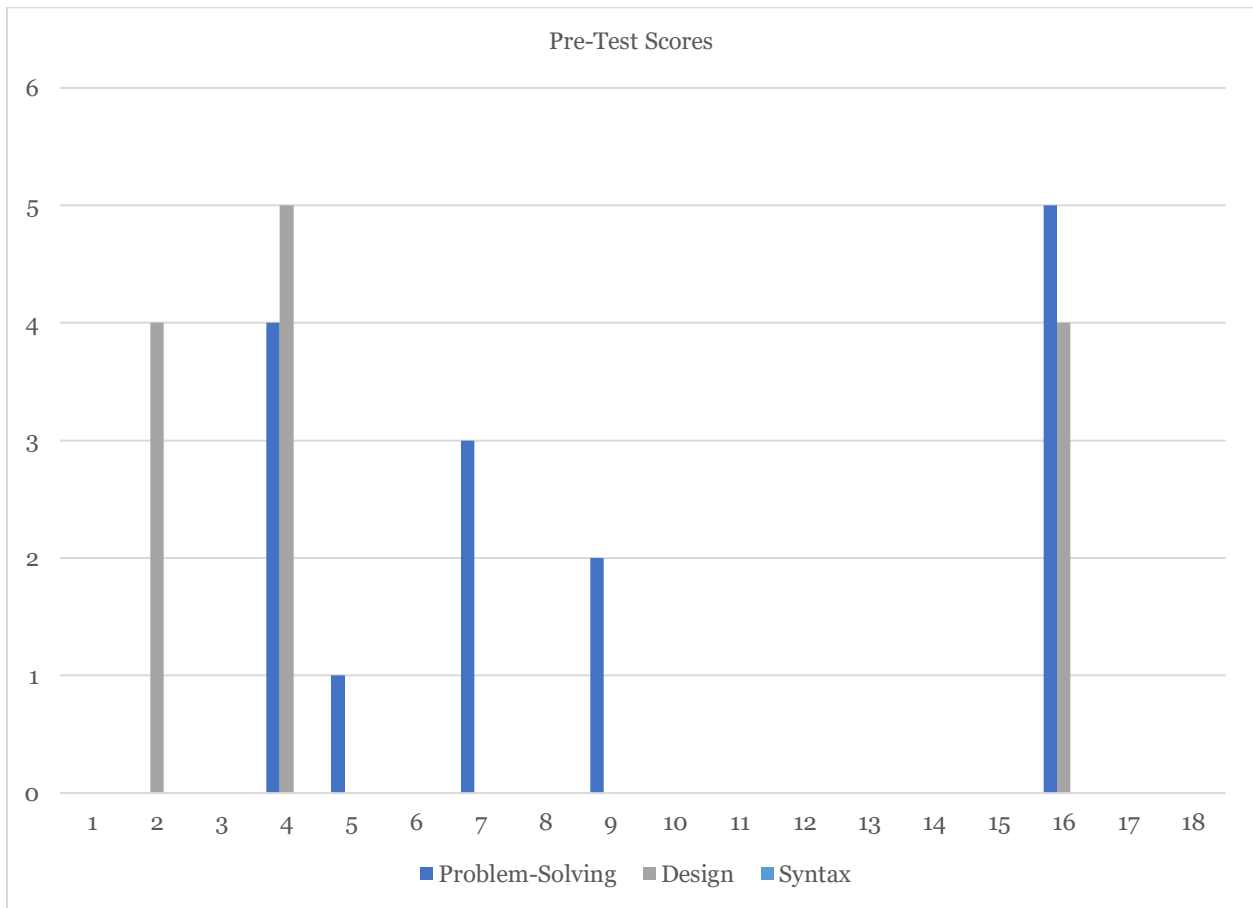


Figure 20 Pre-Test Scores

The first focus group targeted a sample of 5 of these students in attempt to understand why those with prior knowledge scored low in the pre-test. It was found that while they were knowledgeable in problem solving programming problems, the challenge was the application of this knowledge in designing and coding a solution, especially having not practiced programming between the time they completed the Caribbean Secondary Education Certificate (CSEC) and beginning the programme. As one respondent noted, *“I honestly couldn’t remember how to do*

it. After CSEC I stopped practicing for some time so although I was familiar with everything in the test, I just couldn't remember how to do it." Another mentioned, "I know everything about problem-solving [programming problems], but I just couldn't remember how to do it." To these two comments, all the other students agreed.

Competency in Programming

Before the post-test was administered, students were given a reflective questionnaire that asked them to rate their perceived level of competency in programming having finished the course. Students' general outlook was that they felt more competent in programming. While 1 was neutral in their rating, the majority rated their competency positively, with 11 strongly agreeing to the statement, and 6 agreeing.

Post-Test Scores

As shown above, most students perceived themselves to be more competent in programming, a finding that was statistically verified by the post-test administered at the end of the course. The post-test results showed a statistically significant difference between the scores before and after the course. The paired samples t-test showed that students did not perform well before ($M=1.67, SD=3.27$), however, by the end, students performed better ($M=22.61, SD=1.97$). This improvement of 20.94 was statistically significant $t(17) = 29.63, p < .001$ (Figure 21).

		Mean	N	Std. Deviation	Std. Error Mean
Pair 1	PreTest	1.6667	18	3.27198	.77121
	PostTest	22.6111	18	1.97451	.46540

		Paired Differences					Significance			
		Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the Difference		t	df	One-Sided p	Two-Sided p
					Lower	Upper				
Pair 1	PreTest - PostTest	-20.94444	2.99946	.70698	-22.43604	-19.45285	-29.625	17	<.001	<.001

Figure 21 Paired Samples t-Test - Pre/Post-Test Scores

Further to this, the average score increased from an average of 1.5 to an average of 22.6. Where previously, 12 students earned zero points (4 of whom had prior knowledge), with the two highest scores being 9 points, the post-test saw a significant change with 5 scoring the full 25 points, 3 of whom had prior knowledge in programming and 2 did not. The lowest scores were 19 points (no prior knowledge) and 20 points (had prior knowledge). In the pre-test, it was previously stated that 4 students who had prior knowledge in programming scored zero points. In the post test, these said students earned scores greater than 20, the lowest being 20 points and the highest being 23 points. Decomposing these results further into the individual assessment criteria – problem-solving, design and syntax, also showed a major change (Figure 22).

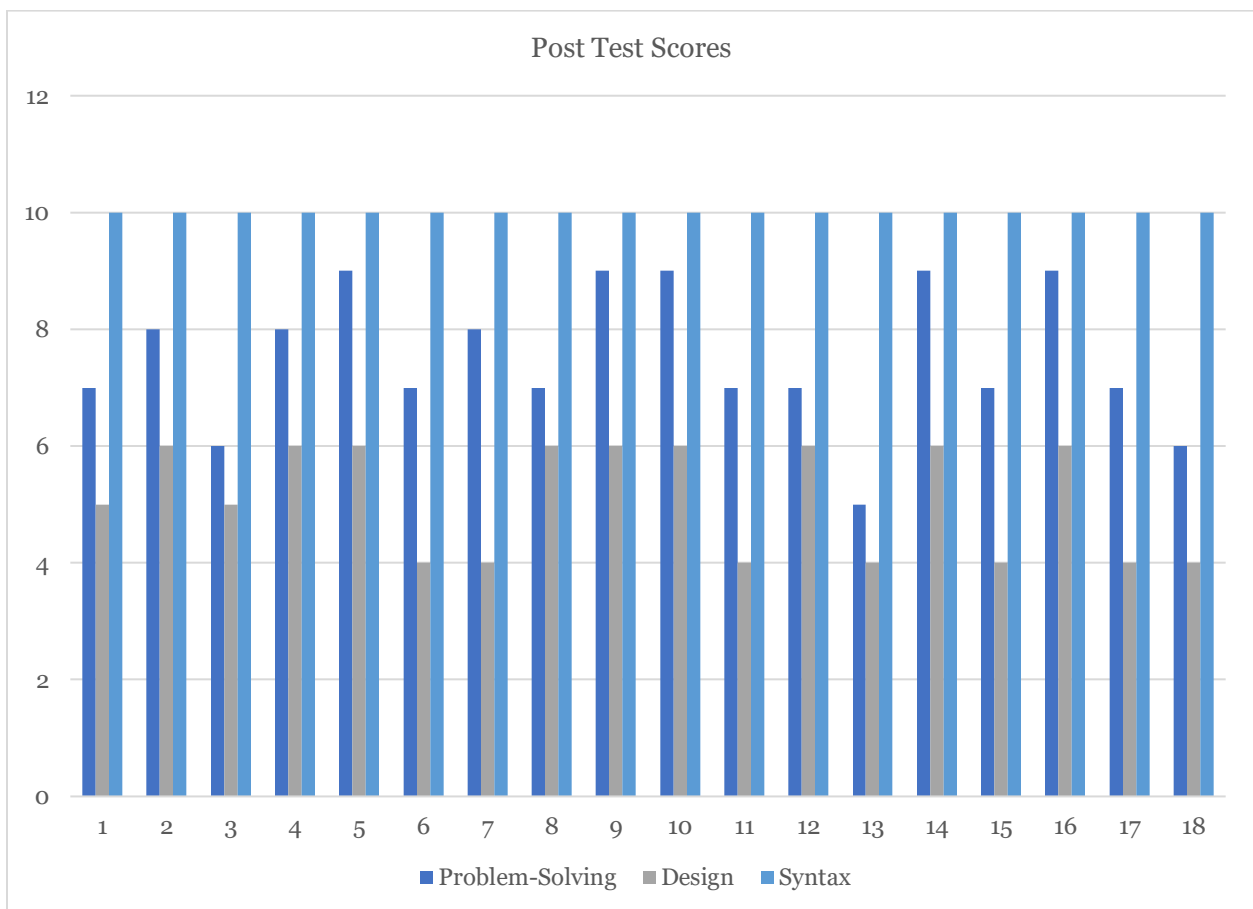


Figure 22 Post Test Scores

As it relates to *syntax*, all students scored full 10 points, in the use of the programming concepts to code the solution. With *design*, in the pre-test, only 3 earned points with the highest being 5 points, and the other two 4 points each. Among these, the student who earned 5 points and the other who earned 4 points, both improved, earning full marks in the post test. The third student saw no improvement in design, earning 4 points in both the pre and post-test. Overall, in the post test, 9 students got the full 6 points for design, 4 of which did not have prior knowledge in programming and 5 did. The lowest marks awarded was 4 points, for which 7 students earned – 3 had prior programming knowledge, while 4 did not. Of the 9 points that was assigned for *problem-solving*, 5 got full points– 2 of whom did not have prior knowledge in programming and 3 had prior knowledge. The lowest mark earned was 5 points, and this student did not have any prior knowledge in programming.

Linking Game Attributes, Engagement and Learning Outcomes Criteria

Linking the performance to the learning outcomes criteria was done by cross referencing the observations, notes, students' comments, and the result of the post-test scores. Overall, the fulfilment of the learning outcomes was evaluated on two criteria:

1. Syntactical knowledge.
2. Problem-solving and design.

Syntactical Knowledge

Linking this to the engagement themes previously presented, *sees relevance*, and *understand* themes were mostly seen to have contributed to enhancing students' performance in syntax related topics. According to the students, their understanding of the syntax of the programming knowledge was improved by seeing how the code they wrote translated into the movement and actions of the character on screen. This not only extended to the use of the syntax, but also in the process of identifying and fixing logic and syntax errors. Among those who had challenges in syntax errors, logic errors and debugging, the in-game feedback or prompts when errors were detected in the code were particularly helpful for identifying their

errors, understanding their mistakes, and providing the hints to correct these errors. As one student noted: *“Before I started playing the game, I was very absent minded with the concepts like syntax error and debugging. The game has made me understand these concepts way more than when I started.”* With *sees relevance*, students’ syntactical knowledge was achieved by the feature of revisiting levels to improve understanding of the use of the syntax.

In discussing syntax related knowledge, those that had prior knowledge in programming indicated that the game characteristics solidified their knowledge of the programming topics: *“I relearnt some of the basic concepts of programming that were previously difficult, but the game really cleared up the terms which allowed me to understand it better.”* For students who did not have previous knowledge in programming, the majority expressed that the game characteristics helped them easily understand the topics which were new to them. As one commented, *“I did not understand programming and this way of learning has made it so much easier to understand.”* Noticeable in their comments were the continuous reference to increased understanding. Therefore, the game attributes support of the learning outcomes did have some influence in how students perceived their understanding, and in this case, the change in perceived knowledge consequently led to a statistically significant difference in test scores, before and after the course.

Problem-Solving and Design

Ozaria implemented problem-solving challenges by providing structured activities with specific goals to achieve, as well as projects that required students to develop their own game using the programming concepts learnt. The observational notes showed that students became more creative in their solutions as the semester progressed. Solutions became more diverse, using a combination of different programming concepts and even some that were not taught in class. This was also evident in the design of the solutions for the post-test. While 10 students developed using a standard solution, 2 coded the acceptance of two types of input – full string

and characters. 6 students implemented the solution using functions. 1 implemented a looping statement to display the questions, and 1 implemented a countdown at runtime.

In the end of semester reflection, students were asked to rate their problem-solving skills. The results show that most rated their problem-solving ability positively, with 13 students rating their knowledge as very good, 3 as good, 1 as excellent and only 1 rating their knowledge as fair. Examining this further showed that 17 students credited the game for enhancing their ability to problem solve. The remaining one remarked that problem-solving was a skill already attained with their prior knowledge and experience in programming, at the secondary school level. Students also attributed the engagement theme *sees relevance* not only to their problem-solving and design skills, but also their creativity in developing solutions. Associated with structured tasks with specific goals to fulfil, students commented highly on the feature of coding without restrictions. As the goals became more complex by integrating more programming concepts, it developed their ability to problem-solve and design solutions. Merging this with no restrictions on how the syntax was used also enhanced their creativity in developing solutions.

“The game presents a series of different obstacles on the same topic. This shows me that there are many ways problems can arise and it also forces me to think of a solution for each.”

“The more difficult levels required more goals allowing the use of multiple concepts which made problem solving key to completing the levels. All these things helped improve my problem-solving skills.”

In the game, there were instances where there weren't restrictions in the code that you use. Due to this I was able to not only recognize that there were multiple ways to do a particular action but also practice doing those actions.

The capstone project allowed flexibility in developing solutions, by allowing students to create their own game using the programming concepts learnt. This also had the same effect of increasing problem-solving skills, design skills and creativity, as the structured activities. The comments on the capstone project showed that most viewed it positively, with 13 students. However, 4 were neutral in their comments, and only 1 saw it negatively. Among those that rated the capstone project positively, most commented on it contributing not only to their problem-solving skills, but also credited it for allowing them to be creative in programming.

“...the capstone project lets me be creative.”

“Yes, it has because I get to show my creativity and portray what I have learnt.”

“Yeah, I think I did because it gives us an opportunity to be creative with the different programming aspects.”

For those that were neutral in their comments, they believed the capstone project enhanced their knowledge in some way, but was not entirely convinced of such, with most describing it as “okay” and “kind of helpful”. In detailing the experience with the capstone project, one student noted that it was confusing at times and would require the lecturer’s intervention to understand the requirements of the project.

“I didn't quite understand what was going on but with my lecturer's help and proper reading I was able to understand what to do.”

Like this student, the one who rated the project negatively, also found it confusing and added that it did not contribute to enhancing their knowledge or skill in anyway.

“I don't exactly enjoy the capstone projects. They haven't solidified my knowledge in anyway either. They're a bit confusing too.”

Summary of Findings

Students who enrol in program are from varying backgrounds: some have prior knowledge and experience in programming, often from formal education at the secondary school level or self-study prior to enrolment, and others have no prior knowledge or experience. Among the students that do, while the students are aware of the syntax of the programming language, and are capable of designing a solution, the major challenge – coming into the university – is the connecting and application of this knowledge in developing solutions.

Ozaria exposed students to five game attributes *action language*, *assessment*, *challenge*, *game fiction* and *rules/goals*. In their interaction with these, attributes *action language*, *assessment*, *game fiction* and *rules/goals* were mostly positively discussed. With *action language*, it was the game's ability to provide them with an environment that allowed them to practice programming through trial and error. It also gave them the ability to bridge the game between the use of the syntax of the language in game and developing solutions to their actual assignments. With *assessment*, students appreciated the feature of highlighting syntax and logic errors as they played the game. With *game fiction*, most found it made the learning experience attractive enough to entice them to continue playing. Students also pinpointed specific features such as the explanation of the programming concepts being incorporated into the storyline, and translation of the code written into the movement of the characters within the interface. With *rules/goals*, all agreed that providing goals for each level ensured that they remained focused on solving the problem provided. *Challenge* saw a more mixed response with some students being encouraged by the increasing difficulty levels that ensured they analysed the problem before

attempting to develop the solutions, while others struggled with developing solutions and the use of the syntax.

Most times a combination of these features significantly impacted the affective, behavioural, and cognitive engagement of students, with affective engagement states *fun/enjoyment*, *interest* and *sees relevance*, behavioural engagement states of *persistence*, and *asking peers for help*, and cognitive engagement state of *understand* being mentioned. Outside of the scope of the pre-coded themes, two other themes emerged, *frustrated* and *tedious*.

Above all others, *understand* was the most impacted by game attributes and the only one that saw all five attributes and its features being the most impactful. This was not limited to students' ability to grasp the syntax of the programming language, and the process of identifying the fixing errors, but it also impacted positively their ability to problem-solve and develop solutions (design). Second to this, was the affective engagement response *fun/enjoyment*, which was also closely linked to the state of *interest* (ranked seventh in frequency), both of which saw the attribute *game fiction* and its features of attention-grabbing and the interface (storyline, music, characters...etc.) being the sole influencer. Also being positively impacted was *sees relevance*; most students except for one without prior knowledge in programming were able to translate what they learnt through the game to developing real-world solutions. A feature that helped in this regard was the ability to revisit levels to reinforce the concepts learnt, with the attributes *action language*, and *rules/goals* being credited to this engagement state.

While students were mostly positively engaged when learning with Ozaria, it also shows that it did not eliminate the common learning challenges associated with programming. That is, students were still *frustrated* with the learning process and felt it was *tedious*, and this at times defined their learning process. *Frustrated* originated from the difficulties in identifying syntax and logic errors and being stuck on levels because of their inability to complete the goals necessary to progress. Apart from *challenges* that saw a mixed response, other attributes mentioned positively – *action language*, *assessment*, and *rules/goals*, were also seen to have

contributed to feelings of frustration. However, most reacted to these frustrations with *persistence*, and some of the very attributes that made students *frustrated* also contributed to their feelings of *persistence* (*challenge*, and *rules/goals*). This occurred because students felt a sense of satisfaction and accomplishment after completing the difficult challenges. While some reacted to the difficulties with *persistence*, a few turned to their peers for assistance when stuck on some levels, attributing *challenges* to this response. With *tedious*, the combination of *action language*, *challenges*, and *rules/goals* and its feature of being repetitive in its activities were quoted as the main reason. However, because they understood the benefit of the repetition of the activities, though it was discussed negatively, students still saw it as being beneficial to their learning. In addition to student engagement, *motivation* was also frequently mentioned, with as much occurrences as *fun/enjoyment* (18). While this was not a focus of this research, interviews did ask students to elaborate on their “motivation comments”. In doing so, many linked motivated to two forms of engagement: *fun/enjoyment* and *understand*.

In an environment that was marked by a frustrating and tedious learning process, a statistical comparison of students’ perception of the knowledge showed a significant difference before and after, with many attributing other forms of engagement – *fun/enjoyment* and *understanding* – as its cause. Unpacking further showed that students with prior knowledge rated their perception mostly positively, and as expected, those without prior knowledge rated their perception negatively. By the mid-semester, students felt their skills were improving and by the end of the course, there was little difference between those who had prior knowledge and those who did not, as most rated their knowledge positively.

This significant increase in students’ perception of knowledge was validated by an equally statistical increase in students’ test scores when comparing pre and post-test results. Both students with prior knowledge and without prior knowledge in programming performed poorly in the pre-test results. While this was expected from students without prior knowledge, it was not of those with prior knowledge – they were able to identify the programming concepts

needed to create the solution, and were able to design the algorithm, but none were able to code the solution. By the end of the semester, most were sure that they felt more competent in programming than when they started the course, and this was even more evident in the post-test scores, with students performing significantly better. All, in spite of their prior knowledge, were able to problem-solve, design and code the solution, even having unique and creative solutions.

Comments showed that the engagement themes *understand* and *sees relevance* were both directly related to the learning outcome criteria of syntactical knowledge. Features of highlighting errors, a visual output of written code, and the ability to revisit the levels for revising the use of the syntax of the programming language were most impactful on acquiring syntactical knowledge. These features were helpful to students who had prior knowledge in programming as a recap of what was already learnt; and for those with no prior knowledge, it aided their understanding of concepts that were new to them.

The learning outcomes criteria, problem-solving and design skills were all achieved through learning to code through game play and learning to code by developing a game strategy. Both also contributed to students developing creativity, as evidenced by the varying coded solutions submitted in the post-test. With structured activities, students credited features related to *sees relevance* as enhancing these skills. Many mentioned being able to better analyse the problem to develop solutions to meet a specific goal. More than this, they were allowed to be creative in their solutions because the game did not restrict their use of the programming syntax. The unstructured activities were the main feature of the capstone project. While there were comments from students who found the project confusing and needed the lecturer's intervention to understand, and another did not find it helpful to their knowledge, most commented positively, remarking that it contributed to them achieving the learning outcomes criteria of problem-solving and design skills. This was attributed to having flexibility in developing their own game using any number of programming concepts learnt throughout the module.

Chapter Five

Discussion and Conclusions

This chapter interprets and describes the significance of the findings. In placing this research in the wider literature, the findings were aligned to empirical studies in both game-based learning and gamification due to the limited research found relating to game-based learning in computer programming. The chapter begins by reiterating the statement of the problem investigated and the main findings after the implementation of Ozaria in the Introduction to Computer Programming course at USC. Guided by the gamification for student engagement framework, the chapter is organised by the four research questions. Following this, Research Question 5 is answered, which deals with the debate of adhering to strict requirements regarding students' aptitude for programming for enrolment into the program. The chapter concludes the findings by using the propositions as a means for informing my professional practice for possible implementation of game-based learning, the features of attributes that can be the most beneficial to students, and how they can help achieve the learning outcomes of programming courses within the department. The chapter ends with a discussion on the limitations of this research and recommendations for future research.

Statement of the Problem

The University of the Southern Caribbean, Trinidad and Tobago, offers a Computer Science degree that has a software emphasis that exposes students to a wide range of courses that cover areas of software, mobile and web development. At the more advanced classes, students must navigate – on their own - the complexity of multiple programming languages and the syntax of each, while simultaneously being taught more advanced programming topics. For the most part, students do not commonly struggle with the programming language and its syntax, but rather they struggle with applying programming logic, problem-solving, and developing solutions to problems. Currently, the department largely adopts a problem-based

approach to assignments through projects and case studies and lectures that follow a demonstration approach to teaching, with a combination of lectures and lab sessions that integrate theory and practical sessions. However, these strategies remain largely ineffective in helping students develop their programming skills. The department has had discussions concerning this issue and much of it surrounded changing the way programming is taught and the possibility of holding to strict aptitude requirements for entry into the program. This research attempts to address this issue by examining the impact of game-based learning on student engagement and performance, the intention of which is to provide empirical evidence on the viability of implementing this approach as a teaching strategy for programming. The impact of aptitude was also addressed to advise on whether having prior knowledge in programming can have an impact on learning programming compared to those that do not.

Answering Research Question 1

Research Question 1 examines the effect the game attributes (action language, assessment, challenge, game fiction, and rules/goals) have on the engagement states: affective, behavioural, and cognitive, where affective is the reaction to teachers or peers, behavioural is the involvement in academic activities, and cognitive is the effort students place on learning. The answer discusses the various engagement states that have been influenced by GBL, and only indicates the game attributes that influenced it. It does not address how this influence occurred, as this analysis is explored in research Question 4. Research Question 1 asks: *“What is the impact of game attributes on student engagement states: affective, behavioural, and cognitive?”*

Much of the findings of this research corroborates current literature. That is, GBL positively impacts engagement states: fun/enjoyment, interest, understand, and sees relevance. Fun/enjoyment, and interest is the most quoted form of engagement that educational games influence, with authors Abidin and Zaman (2018), Begosso et al. (2018), Butt (2016), Chang et al. (2020), De Pontes et al. (2019), Gallego-Duran et al. (2016), Khaleel et al. (2019), Kumar and

Sharma (2019), Lopez-Fernandez et al. (2021), Mathrani et al. (2016), Martins et al. (2018), Topalli and Cagiltay (2018), and Zhu et al. (2019), all reporting students having an increased fun/enjoyment, interest, and/or satisfaction in learning. The findings of this study suggest the same, but more so on fun/enjoyment than interest being highly regarded by students. The direct cause of this being the audio/visuals of the game, that is the storyline, characters, and overall interface (game fiction).

Again, supporting the findings of the current literature, understanding was one of the most impacted engagement states that resulted from the implementation of GBL, with authors Abidin and Zaman (2017), Chang et al. (2020), Cubukcu et al. (2017), Dolgopolovas et al. (2018), Khaleel et al. (2019), Kumar and Sharma (2019), Troussas et al. (2020), and Zhao et al. (2022), all reporting an enhanced understanding of programming concepts resulting from the game environment. However, in the case of this research, this understanding extended further than just the syntax of the programming language, but also in identifying and fixing syntax and logic errors, and problem-solving. The data shows that the likely cause of this enhanced understanding was a combination of attributes: action language, assessment, and game fiction.

Not as common as understand, fun/enjoyment, and interest, the findings also suggest that students are able to make connections between what was learnt in the game environment to developing real-world solutions (sees relevance). This ability to see the relevancy supported the findings of Mathrani et al. (2016), and Zhu et al. (2019), who both found that students were able connect the game elements to the programming modules being learnt. More than this, a few comments indicate that there is a correlation between sees relevance and understanding. Students are able to properly make connections between what is learnt in the game and actual programming because they are better able to remember, and understand the programming concepts being taught, as well as analyse problems and develop solutions. In identifying what facilitated this relevancy between gameplay and course topics, this research finds it to be action language and rules/goals, with the latter playing a significant part in understanding the syntax

related topics, and the former contributing to students problem-solving abilities. This is evidenced by comments that centred around students revisiting the levels, showing that they attempted to acquire a better understanding of the programming concepts. Also, that meeting goals ensured that they worked towards problem-solving and developing specific solutions.

Despite the data showing that GBL impacted positively on engagement states fun/enjoyment, interest, and sees relevance and understanding, it also shows that there is the potential for students to experience undesirable emotions as Shabalina et al. (2016) argues. Relating specifically to programming, the results indicate that even a GBL environment did not eliminate the common negative emotions caused by programming and the findings are just as Bubica & Boljat (2014) report: it results from frequent repetition of learning tasks, difficulty in identifying syntax errors, and misunderstanding the features of the programming language. The reason for this was the combined characteristics of challenge, action language, and rules/goals of requiring students to reuse the same programming concepts to solve similar problems to complete each level. This became tedious for students, with a few comments suggesting that there is a potential for this to evoke feelings of boredom, as Mathrani et al. (2016) also found.

On the point of having difficulty in identifying errors and misunderstanding the features of the programming language, students became frustrated with the learning process. The reason for this being a combination of action language, challenge, and rules/goals - the challenges students experienced with the process of debugging resulted in them being stuck on the levels, unable to fulfil the goals to progress. An occurrence that is also not uncommon in current literature, with Morales-Tujillo and Garcia-Mireles (2021), and Rojas-Lopez et al. (2019), showing that even in an environment that is marked by feelings of fun, enjoyment and satisfaction, students still had a heightened sense of tension, worry and anxiousness. Facey-Shaw et al. (2020) saw a downward trend in enjoyment as tension increased; a conclusion not supported by the findings of this study. Students were indeed frustrated as they learnt programming, yet by the mid-semester they were still enthusiastic about learning programming,

and at the end they did not find the course overall frustrating. Unlike Facey-Shaw and co-authors, the findings instead show a causal relationship: frustration can lead to an increase in persistence in learning, and even an effort towards learning collaboratively (asking peers for help). The frustrations students felt, and the tediousness of the learning process ultimately caused students to persist in learning, an engagement indicator modified because of the attributes challenge and rules/goals.

A likely underlying influence of this is intrinsic motivation. Although motivation was not an intended focus of this study, it was mentioned several times during the focus groups, with several students mentioning being motivated by mostly feelings of fun/enjoyment, and interest. This is just as Axelson and Flick (2010), and Ferrer et al. (2022) mentioned; engagement and motivation often share a direct connection. A connection Butt (2016), Chang et al. (2020), and Gallego-Dunram et al. (2017) cited: states of fun and enjoyment and the resulting positive change in attitude is an indication of increased motivation among students. This is what the findings of this study observed: students worked harder and persisted in their studies despite the challenges, frustrations, and tediousness of the learning environment. In other words, the states of fun/enjoyment, and interest results in a positive change in attitude from frustrating to persistence and allowed students to overlook the tediousness of the learning environment – a possible indication of increased motivation among students. This further supports the claims made by Facey-Shaw et al. (2020), Figueiredo & Garcia-Penalvo (2020), Fortaris et al. (2016), Gallego-Duran et al. (2017), Khaleel et al. (2019), Kumar and Shamar (2019), and Rojas-Lopez et al. (2019), who all reported that students who were intrinsically motivated were more inclined to complete the activities presented to them, even in the face of challenges.

Although not the initial intention of the research, the findings show that not only did frustrations have the capacity to lead students' persistence in learning, but it also has the potential to encourage learning through collaboration. The reason for which resulted from the challenge attribute. There is already evidence that a game environment did encourage individual

learning, as most game environments do. A characteristic noted by Romero et al. (2015). This is evidenced by the students' comments on having to research further to understand programming concepts and code solutions. However, in instances where the activities were too complex, some students did reach out to their colleagues to get help in understanding the programming concepts. Though there was no further analysis into whether students favoured working collaboratively as Abidin and Zaman (2017), de Sana Quaresma et al. (2020), and Rojas-Lopez et al. (2019) claim, the findings do support partially the works of Abidin and Zaman (2017), and the conclusions of Troussas et al. (2020) in that learning collaboratively did advance the students' knowledge in the programming concepts.

Answering Research Question 2

This research question examines the measurable consequence of game attributes influence on engagement states. Unlike the current literature that measured education games influence on completed assignments (De Pontes et al., 2019; Harrington & Chaudhry, 2017), time on task (Landers & Landers, 2015), work continued/practice (Khaleel et al., 2020; Paiva et al., 2020), code quality (Kasahara et al., 2019), this research evaluates students' perception of knowledge as the measurable consequence of the state of engagement. By comparing students' perception of their knowledge in algorithm, syntax error, logic error, debugging, looping statements, objects, methods, arguments, engineering cycle, variables, conditional statements, variable arithmetic, nesting statements, while loops, compound conditionals, and functions, before and after the course, the research question answers: *"To what extent does engagement influence students' perception of knowledge?"*

The findings show that students changed from a mostly poor rating, to feeling like their skills were improving, and feeling enthusiastic in learning by the mid-semester. At the end of the semester, students rated their perception of knowledge as mostly excellent and most agreed that they felt more competent in their programming skills. Overall, a comparison of students' perception of knowledge before the course began and after its completion shows a statistically

significant difference. These findings support that of Cubukcu et al. (2017), who also noted an increased understanding of the topics within a course, and a better connection to the course contents. Dolgopolovas et al. (2018) presented findings that suggested that changes in students' perceived understanding – such as this - is facilitated by reduced tension. This is not supported by the findings of this research. Students' learning process was marked by frustrations and tediousness. Much like Butt (2016) who reported students having challenges but still found the learning experience positive, the evidence of this study confirms this. It shows that possible negative indicators - tediousness and frustration - did not have a lasting impact on how students view the development of their skills. From the data, emerged two possible reasons.

Firstly, the data indicates that there is likely a direct and indirect relationship between engagement and perceived knowledge. Themes such as fun/enjoyment, and interest have an indirect influence on students' perceived knowledge. This is unlike the results of the studies done by Abidin and Zaman (2017), Begosso et al. (2018), and Topalli and Cagiltay (2018) who all determined a possible direct connection between fun/enjoyment, interest and the measurable consequence of knowledge gains and performance. In this case, students mostly mentioned these themes in relation to their overall learning experience, explaining that the game environment helped them to retain their attention and focus while being entertained. From their comments, the themes that saw a more direct relation to students' perceived knowledge are understand, sees relevance, and on a smaller scale asking peers for help. For instance, students mentioned being able to better grasp the syntax and use each of the concepts of the programming language. In using the gaming platform, students were not only able to better transfer syntactical knowledge into coding real-world solutions but was also able to transfer the problem-solving skills into designing solutions. Through repetition, students were able to reinforce their knowledge of the programming concepts. For a small group of students, learning from their peers helped them understand concepts that were too challenging.

Secondly, the findings indicate that students possibly became more self-confident. Ortiz-Rojas et al. (2019) found that self-confidence decreased over time as the programming challenges became more difficult. However, there was no evidence of this. Instead, as the course progressed, students went from rating their knowledge mostly negative, to feeling like their skills were improving by mid-semester, and by the end of the course, students felt more competent in their programming skills, despite being frustrated with the learning process and at times finding it tedious. This mostly supports Dolgopolas et al. (2018), who suggested that self-confidence is likely a reason why students' understanding increased in a game learning environment. Students were often quoted as experiencing feelings of satisfaction after overcoming challenges. This also supports the claims made by Zimmerman (2020) – students with higher self-confidence see difficult tasks as something to be mastered. It is likely that students' self-confidence increased when finishing the levels that were challenging, changing how they viewed these tasks. Not as something discouraging, but something to overcome through continued practice and learning, ultimately becoming a possible contributor to the change in how students perceive their programming knowledge.

Answering Research Question 3

This research question explores how this has impacted the students' fulfilment of the learning outcome objectives of the Introduction to Computer Programming course. These objectives include problem-solving, design and syntactical knowledge. It poses the questions:

“To what extent does game attributes support learning outcomes?”

Ahmad et al. (2020), Chang et al. (2020), Khaleel et al. (2020), Rojas-Lopez et al. (2019), Shorn (2018), and Troussas et al. (2020), were the only articles found to have used overall performance to evaluate the achievement of learning objectives. Each evaluated this on students' performance, and reported that students performed well in the game environment, leading to the conclusion that it does support the learning outcomes of the course in which it is applied. In basing the fulfilment of learning outcomes through overall performance using a pre

and post-test then, this study does support the author's findings. The results did show that overall, students had a statistically significant difference in the test scores after the course. On these results, it is reasonable to conclude that GBL can support the learning outcomes of a course. Dambic et al. (2021), taking performance further, concluded that students did not meet the learning outcomes after evaluating each topic of the course. This research does not support this conclusion, in that, dividing the course into its specific objectives (problem-solving, design and syntactical knowledge) shows that students can achieve the learning objectives of the course. On average, most students performed better in the specific areas of problem-solving, design, and syntax related areas when comparing test scores before and after the course.

Taking this a step further and basing the fulfilment of learning outcomes on Bloom's taxonomy as Chang et al. (2020) did, indicates that like these authors this research also found that GBL was seen to have supported the levels of 'understand' and 'apply'. However, it also reveals that these are not the only two that are possible - it can correspond to the levels of 'remember' and 'create' (Figure 23). Based on the findings, the link to Bloom's taxonomy is as follows:

1. Remember – recall the syntax of the programming language
2. Understand – proper use of the programming concepts
3. Apply – use a combination of the programming concepts to develop solutions
4. Create – develop unique solutions to real-world problems.

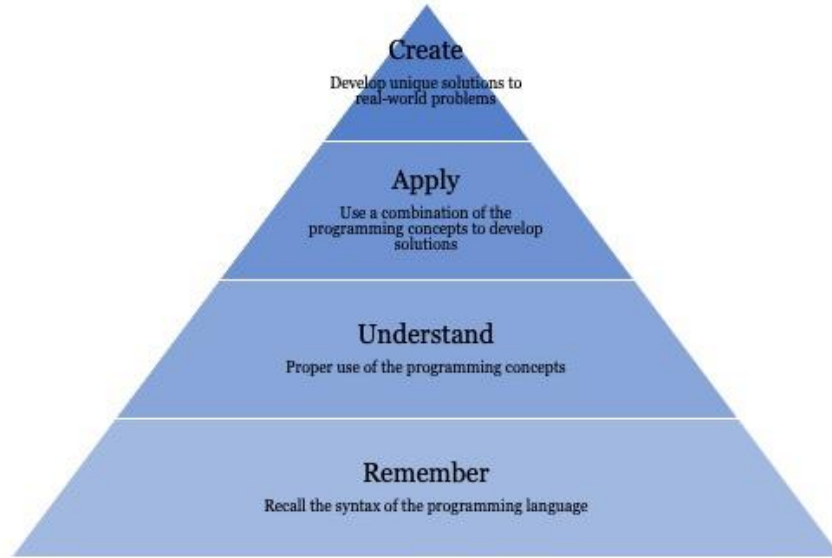


Figure 23 *Linking Learning Programming to Bloom's Taxonomy*

The reason for this lies in the game attributes. As previously mentioned, connecting the game attributes to learning outcome objectives have rarely been explored in current literature. As a result, much of the following discussion focuses on the findings of this study with little connection to current literature. The comments made by students evidenced the influence of game attributes on the learning outcomes of the course, showing each attribute – action language, assessment, challenge, game fiction, and rules/goals – contributing in its own way. Connecting to remember and understand, attribute assessment contributed by highlighting syntax and logic errors allowing students to understand the syntax of the programming language. Action language and game fiction by merging programming lessons into the storyline and seeing how the code written translates into movement on the interface. Thereby connecting code from within the game, to using code to build real-world solutions. Rules/goals by its repetitious nature ensured continuous practice, thereby allowing students to recall the syntax of the language.

In relation to apply and create, there were two opposing views on how problem-solving and design capabilities can be achieved. Kay (2002) and Peng (2010) stated that problems should be ill-structured and challenging, while Bawamohidden and Razali (2017) argued that it should be well-defined and uncomplicated. Ozaria implemented both, with the capstone project being ill-structured and challenging and the activities of the game being well-defined and uncomplicated. Both were effective in fostering the ability to problem-solve and design solutions. Structured activities with increasing difficulty levels that provide specific goals to achieve can assist students in analysing the problem to develop solutions. Unstructured activities, with no restrictions on implementation also have the potential to allow students to analyse and develop solutions, but also to be creative in those solutions.

Answering Research Question 4

This research addresses the link between game attributes and the learning outcomes in relation to the student engagement states. It analyses this by discussing the features that facilitated this relationship. The question asks, *“How does student engagement affect the relationship between game attributes and learning outcomes?”*

Akkaya and Akpınar (2022), Lopez-Fernandez et al. (2021), and Tassaduq et al. (2021) all concluded that there is no correlation between increased engagement and overall performance. The findings of this research do not support this view, instead, it shows that students' comments directly related engagement, particularly understand and sees relevance, to their understanding of the programming language and concepts, as well as their improved problem-solving and design capabilities, and by extension their overall performance.

More than just the game attributes, it is the features of these that have the most impact. The findings of this research mostly confirm the proposal made by Stott and Neutaedter (2013), that a successful game learning environment should incorporate the dynamics of feedback, freedom to fail, progression, and storytelling. In this context, the data shows that feedback, and freedom to fail were seen to have influenced engagement states in a way that allowed students to

fulfil the learning outcomes of the course. However, progression and storytelling were indirectly related. It created an environment that was engaging enough for students to overcome the common undesirable emotions related to learning programming.

Feedback emerged as an important feature for two reasons. Firstly, the prompts for identifying errors, according to students, was the main facilitator for better understanding the syntax of the programming language, assisting in easily identifying the syntax and logic errors in the code, and overall making the process of debugging easier. Secondly, a critical aspect to students remembering and understanding is visual feedback from writing code. Zhu et al. (2019) mentioned that visualisations are instrumental for understanding programming concepts and making connections to the programming language. In supporting this claim, the findings of this study show that the audio/visual representations enhanced students' understanding by allowing them to visually see how the code written translates to movement of their character, rather than just seeing the output like a traditional development environment. This indicates that students are not only able to understand the code, but also able to make connections to the application of the programming language. Again, showing a strong connection between the engagement themes of understanding and sees relevance. Linking this further to the scores gained in the post-test shows that students did improve their ability to code, were able to learn from their errors during the learning process and apply what they learnt to problem-solving, designing and coding real-world solutions. A finding that supports the same conclusions made by Chang et al. (2020), Paiva et al. (2020), and Rojas-Lopez et al. (2019).

Ozaria provided students the ability to revisit and review the activities and concepts that were challenging, rather than present it to them as one-off activities. Like Stott and Neutaedter (2013), Dempsey et al. (2002), and Mathrani et al. (2016) argued that learning through a game environment should include the freedom to fail, revisit and learn from the errors. Based on the findings of this research, this feature is found to be beneficial for students to make connections (sees relevance) not only in understanding the syntax related topics, but also in applying their

problem-solving capabilities to create and design solutions. This is evidenced by the students' comments on relying on the process of trial and error to understand the syntax and applying concepts such as conditional and looping statements. This process not only facilitated the learning outcomes, but it is likely that it is also a contributing factor in increasing students' self-confidence, a theme shown to directly relate to students improved perception of knowledge. Agapito and Rodrigo (2018) claims that features such as these favour both strong and weak students – a claim the findings hint at being true. Both students with and without prior knowledge benefited - students with prior knowledge stating that it assisted in revising what is already known, and students with no prior knowledge citing that it afforded them the opportunity to learn something new.

It is clear from the findings that GBL does not eliminate the common drawback related to learning programming – frustration and a tedious learning process. However, engagement states, fun/enjoyment, and interest – although not directly related to the learning outcomes – did enhance engagement and motivate students to continue to learn despite the negatives. For this reason, it is worth mentioning the features of the attributes that results in students' persistence in learning, and ultimately ensures remembrance and understanding of the topics. These include the storyline aspect, and visual of students' progressions in the form of the map.

Stott and Neutaedter (2013) claimed that storytelling provides the aesthetics for which actions and tasks can be practiced, and this seemed true - students did show a greater interest in learning because of the appealing nature of the storyline. However, the data shows that for programming, more is required than just aesthetics; students are more engaged and more inclined to remember and understand when aesthetics is merged with the learning of the content, as evidenced by the mention of an easier learning process when theory and practice is learnt as the game is played. However, it is also important to note that storytelling may not be entirely effective for all students. The comments stating an indifference and disinterest in the

storyline, suggest that this feature may be subjective and not have the same or intended effect among all students.

A similar assumption can also be made for the visible representation of progress. Done through the map, Figueiredo and Garcia-Penalvo (2020) argued that such visuals can be demotivating to students who perform poorly. This was not the case in this context. Students were still enthusiastic to learn by mid-semester and was overall motivated to learn programming. These findings align more with those of Begosso et al. (2018), Call et al. (2021), Fortaris et al. (2016), Khaleel et al. (2020), and Morales-Tujillo and Garcia-Mireles (2021), who all cited that students are more motivated to put effort into the course after seeing their progress. A likely reason for not seeing demotivation in any of the students is that the map is only viewed by the student and not the entire class. Therefore, progress could not be compared, and students were able to concentrate on their own learning without the added pressure of competing with their peers.

Answering Research Question 5

Independent of the theoretical framework adopted, Research Question 5 addresses the discussion on the issue of student aptitude and adhering to strict requirements for determining students' acceptance into programme. It asks, *"How does having prior knowledge and not having prior knowledge compare in affecting students' ability to learn programming?"* The findings suggest two interpretations, (1) having prior knowledge in programming is not an indicator of competency in learning programming, and (2) having no prior knowledge is not a marker of inability to be competent in programming.

Most students who had prior knowledge were formally educated and the others self-studied before the start of the programme. In the pre-test, these students –despite having some previous knowledge in programming – were unable to perform well. Overall, all students scored below half, with scores in the problem-solving and design questions, but earned no points in the coded question. This was further supported by the students' pre-perception of their knowledge,

where most students negatively rated their knowledge in most syntax related topics but rated positively their knowledge in algorithms.

Students felt that they had a good enough grasp of syntax related topics, and in the pre-test, were – to some extent – able to answer the problem-solving and design questions, yet they were unable to complete the coded questions. This contradicts the findings of Barlow-Jones and van der Westhuizen (2014) and the partial findings of Smith et al. (2019), who found a correlation between prior knowledge and performance. Derived from the students' comments, an explanation is that students understood the concepts but were unable to remember its application, thereby corroborating the claims of Bosse and Gerosa (2017), Butler and Morgan (2007), and others: the struggle for students is not in understanding the concepts but in applying and combining these concepts to develop and code solutions. The findings also indicate that the problem does not occur at the undergraduate level, but instead, students enter university suffering from the same problem-solving difficulties widely reported in literature by authors such as de Raadt, (2007), Lopez et al. (2008), Xie et al. (2019) and others: decomposing problems into sub-problems, transforming the sub-problems into workable solutions.

As expected, students that had no prior knowledge in programming rated their perception of knowledge in the programming topics negatively and failed the pre-test. However, by the end of the course, these students' perception of their knowledge and overall performance was on par with that of their peers who had prior knowledge. Most who had prior knowledge and no prior knowledge were able to earn close to full points in the post-test, earning high marks in the problem-solving and design questions, as well as the coded question. Added to this, their post perception of knowledge saw both categories of students mostly rating positively their knowledge in problem-solving and design topics as well as syntax related topics. These findings suggest that students can perform well overall despite not having prior knowledge in programming, thereby, supporting the findings of Lacher et al. (2017) who noted that there was no significant difference between aptitude and previous experience and final grades. Some

students with high aptitude can perform poorly, and some students with a low aptitude can perform well. This also extends to engagement. Rojas-Lopez and co-authors, in their study found that engagement was low among students who did not have prior knowledge in programming. However, the findings of this study show no indication of this occurring; engagement was high among most students notwithstanding their prior knowledge and experience.

Ultimately, students of the Introduction to Computer Programming class all came from different backgrounds - students who self-learned prior to enrolling in the program, students who had no prior knowledge in programming and students who had formal prior knowledge in programming. Yet there was little difference between each in their performance both before and after the course, and even in their level of engagement. Therefore, the findings suggest that it is not entirely accurate to use any perceived relationship between prior knowledge and performance to distinguish students through aptitude as those who can program from those who cannot. Since aptitude may not be able to be accurately measured or assumed, the faculty should not consider aptitude as a possible method through which students are selected for the programme.

Propositions and Conclusions

Each of the research questions were formed from the propositions theorised by the gamification for student engagement framework:

1. GBL and its game attributes are the process through which student engagement states (affective, behavioural, and cognitive) can be positively modified.
2. The perception of students' knowledge can be positively altered by the state of engagement which spans affective, behavioural, and cognitive domains.
3. Game attributes support the achievement of learning objectives that span problem-solving, design and syntactical knowledge.

4. It is possible to select a game attribute for a GBL strategy by identifying the domain shared between the learning objectives and the desired modifying student experience of engagement.

Proposition 1

The proposition in which Research Question 1 was formed theorised that “*GBL and its game attributes are the process through which student engagement states (affective, behavioural, and cognitive) can be positively modified.*” Based on the analysis, this proposition is assumed true.

The department can stand to benefit from the implementation of GBL for teaching programming; it fosters several forms of engagement that is hoped for when teaching and learning programming. That is, having fun while learning, having a continued enthusiasm to learning, being motivated to persevere, becoming more self-confident in their skills, understanding the content, and being able to make connections to real world applications of code. What GBL is not is a strategy that can eliminate the frustrations associated with programming, as faced by students at present during the learning process. While GBL does promote some level of individual learning, which is an essential characteristic for learning programming, it is still prudent to ensure that the lecturers offer additional feedback or explanations where necessary to ensure complete understanding of the course content.

Proposition 2

Research Question 2 was formed from this proposition which suggests that: “*The perception of students’ knowledge can be positively altered by the state of engagement, which spans affective, behavioural, and cognitive domains.*” The findings suggest that this proposition is also assumed true.

GBL can create a highly engaged learning environment, that will likely positively influence how students view the development of their programming skills both directly and indirectly. In using GBL, indirectly, the department stands to create a positive learning

environment that promotes a meaningful learning experience that ensures that students remain attentive and focused. Directly, with attention and focus, students are more willing and motivated to participate and invest their time in trying to understand, translate the concepts learnt into designing and developing solutions, and overcome the difficult activities despite feeling frustrated in a learning process defined as being tedious at times. Combined students are more likely to feel more self-confident, leading to feelings of enhanced perception of knowledge gain and competency.

Proposition 3

The third proposition, from which Research Question 3 was formed theorised that, “*Game attributes support the achievement of learning objectives that spans problem-solving, design and syntactical knowledge.*” This proposition is assumed true.

GBL does have the potential to see students within the department perform better overall in their courses. With GBL, students will most likely be able to recall the syntax of the programming language, properly use programming concepts, combine these two for developing solutions, and develop unique and creative solutions to real-world problems, aligning to the remember, understand, apply, and create levels of Bloom’s taxonomy. In attaining each of these, there is the potential of GBL to positively impact students’ overall performance. Narrowing on the specific learning objectives, most programming courses within the department are evaluated on the outcomes of problem-solving, design skills and syntactical knowledge. The implementation of GBL and its attributes have the potential to support the achievement of each of these. Most importantly, GBL offers the department two ways to address the problem-solving issues of students: structured activities with clearly defined goals, and unstructured and flexible activities.

Proposition 4

Proposition four speculated that “*it is possible to select a game attribute for a GBL strategy by identifying the domain shared between the learning outcomes/educational*

objective and the desired, modifying student experience of engagement.” This is also assumed true based on the data.

Using GBL has been shown to influence students’ engagement positively, and this influence ultimately impacts their overall performance and the achievement of the learning outcomes objectives of the programming courses. While Ozaria implemented game attributes action language, assessment, challenge, game fiction and rules/goals, the true influence of game attributes are the features that it provides. These features are ultimately what influences students’ ability to be engaged with the course material, and what will affect their performance. It is a fair statement that Ozaria cannot be implemented at any other level programming course within the department, however, in choosing a GBL strategy for more advanced level courses, there are some features that are highly recommended.

1. *Feedback.* The game should provide immediate and timely feedback, especially for syntax and logic errors.
2. *Freedom to fail.* Students should be given the opportunity to revisit the levels of the game as many times as needed to understand the programming concepts.
3. *User Interface.* While storytelling is not completely necessary, it is important that students are able to see how their code is translated into some visual representation. This gives students a better understanding of how concepts such as looping statements, and conditional statements operate.
4. *Code with no limitations.* Students should not be restricted in how they implement solutions. Of course, this does not mean allowing inefficient code, but instead, not restricting the use of specific programming concepts. This gives them the opportunity to implement concepts learnt on their own or implement a combination of concepts in different ways to achieve the same outcome.

5. *Visual representation of progress.* Students should have some visual representation of their progress, not necessarily public to the class. This has the potential for encouraging a sense of accomplishment, confidence, and motivation.

Opportunities for Future Research

The findings of this research also outlined three areas that are opportunities for future research at the University. Firstly, there is the opportunity to explore collaborative learning using educational games in the Introduction to Computer Programming course. Secondly, there is also the opportunity to research the lasting effects of students who have already completed a course using educational games, especially at the more advanced level programming courses, to verify whether students are truly capable of developing programmable solutions. Lastly, the research implemented GBL at the introductory level; however, it would be interesting to evaluate how educational games can impact engagement and performance at the more advanced level programming courses. Some possible programming classes that are likely candidates are Computer Science 1, Computer Science 2, and Object-Oriented Design and Programming.

Limitations of the Research

This study has three limitations. Firstly, because of the time frame in which this research was conducted (one semester - approximately 3 months), the full scope of the theoretical framework was not implemented. Aspects relating to student-university relationship, attitudes of students prior to the commencement of the course (such as motivational level, self-efficacy...etc.), and retention rate after the conclusion of the course were not addressed.

Secondly, the sample size of this research was small, with 18 students from the Introductory Computer Programming class participating in this research. Such a small sample limited the use of statistical methods. Initially, it was planned to statistically test the relationship between students perceived knowledge and actual performance. However, the sample size was insufficient to yield viable results. That is, the data could not produce statistical evidence of

correlation between the two, and instead determining a relationship had to be done by cross referencing the paired samples t-test results with the qualitative data.

Thirdly, there was a lack of previous studies in game-based learning and computer programming. As a result, this not only limited the identification of the scope of work but also limited the overall discussion. Game-based learning lacked discussion on the game elements that influence student attitudes and behaviour, and the specific features that are the most impactful. The theoretical framework adopted required addressing these, which was important for informing possible game strategies within the department. For this reason, gamification research was included to ensure that each proposition and its associated research question could be sufficiently addressed.

Bibliography

- Abidin, H. Z., & Zaman, F. H. K. (2017). Students' perceptions on game-based classroom response system in a computer programming course. *Proceedings of the 2017 IEEE 9th International Conference on Engineering Education, IEEE ICEED 2017, 2018-January*, 254–259.
- Adams, N. E. (2015). Bloom's taxonomy of cognitive learning objectives. *Journal of the Medical Library Association*, 103(3), 152–153.
- Agapito, J., & Rodrigo, M. M. (2018). Identifying Meaningful Gamification-Based Elements Beneficial to Novice Programmers. *Proceedings of the 26th International Conference on Computers in Education*, 619–624.
- Ahmad, A., Zeshan, F., Khan, M. S., Marriam, R., Ali, A., & Samreen, A. (2020). The Impact of Gamification on Learning Outcomes of Computer Science Majors. *ACM Transactions on Computing Education*, 20(2).
- Ahmad Tuan, Hussin, A., & Yusri, G. (2019). A Review of Learning Theories for Gamification Elements in Instructional Games. *Malaysian International Conference on Academic Strategies in English Language Teaching*, 1–14.
- Ahmadzadeh, M., Elliman, D., & Higgins, C. (2005). An Analysis of Patterns of Debugging Among Novice Computer Science Students. *ACM SIGCSE Bulletin*, 84–88.
- Akkaya, A., & Akpınar, Y. (2022). Experiential serious-game design for development of knowledge of object-oriented programming and computational thinking skills. *Computer Science Education*.
- Al-Azawi, R., Al-Faliti, F., & Al-Blushi, M. (2016). Educational Gamification Vs. Game Based Learning: Comparative Study. *International Journal of Innovation, Management and Technology*, 131–136.

- Alhammad, M. M., & Moreno, A. M. (2018). Gamification in software engineering education: A systematic mapping. *Journal of Systems and Software*, *141*, 131–150.
- Ali, M. M., & Hassan, N. (2018). Defining Concepts of Student Engagement and Factors Contributing to Their Engagement in Schools. *Creative Education*, *09*(14), 2161–2170.
- Al-Linjawi, A. A., & Al-Nuaim, H. A. (2010). Using Alice to Teach Novice Programmers OOP Concepts. *Journal of King Abdulaziz University-Science*, *22*(1), 59–68.
- Almalki, S. (2016). Integrating Quantitative and Qualitative Data in Mixed Methods Research—Challenges and Benefits. *Journal of Education and Learning*, *5*(3), 288.
- Almeida, F., & Simoes, J. (2019). The role of serious games, gamification and industry 4.0 tools in the education 4.0 paradigm. *Contemporary Educational Technology*, *10*(2), 120–136.
- Altadmri, A., & Brown, N. C. C. (2015). 37 million compilations: Investigating novice programming mistakes in large-scale student data. *SIGCSE 2015 - Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, 522–527.
- Amran, A., Jalil, H. A., Muhamad, M., & Nasharuddin, N. A. (2021). Factors Contributing to Collaborative Game-Based Learning (CGBL) Effectiveness. *Asian Social Science*, *17*(11), 1.
- Ansari, S., Hameed Panhwar, A., Akbar Mahesar, G., Panhwar, A. H., Akbar, G., & Ariel, M. /. (2016). Mixed Methods Research: Ontological, Epistemological and Methodological underpinnings. In *ARIEL An International Research Journal of Language and Literature* (Vol. 27).
- Aparicio, A. F., Vela, F. L. G., Sánchez, J. L. G., & Montes, J. L. I. (2012). Analysis and application of gamification. *ACM International Conference Proceeding Series*.
- Aparicio, J., Pereira, S., Aparicio, M., & Costa, C. (2019). Learning Programming Using Educational Robotics. *2019 14th Iberian Conference on Information Systems and Technologies (CISTI)*.
- Apiola, M., & Tedre, M. (2012). New perspectives on the pedagogy of programming in a developing country context. *Computer Science Education*, *22*(3), 285–313.

- Appleton, J. J., Christenson, S. L., & Furlong, M. J. (2008). Student engagement with school: Critical conceptual and methodological issues of the construct. *Psychology in the Schools*, 45(5), 369–386.
- Arango, F., Aziz, E.-S., Esche, S., & Chassapis, C. (2008). A Review of Applications of Computer Games in Education and Training. *38th ASEE/IEEE Frontiers in Education Conference*, 1–6.
- Armstrong, P. (2010). *Bloom's Taxonomy*. Vanderbilt University Center for Teaching.
- Arnab, S., de Freitas, S., Bellotti, F., Lim, T., Louchart, S., Suttie, N., Berta, R., & de Gloria, A. (2015). *Pedagogy-driven design of Serious Games: An overall view on learning and game mechanics mapping, and cognition-based models*.
- Arnold, B. J. (2014). Gamification in Education. In *Proceedings of ASBBS* (Vol. 21).
- Ashwin, P. (2012). How often are theories developed through empirical research into higher education? In *Studies in Higher Education* (Vol. 37, Issue 8, pp. 941–955).
- Asikainen, H., & Gijbels, D. (2017). Do Students Develop Towards More Deep Approaches to Learning During Studies? A Systematic Review on the Development of Students' Deep and Surface Approaches to Learning in Higher Education. *Educational Psychology Review*, 29(2), 205–234.
- Axelson, R. D., & Flick, A. (2010). Defining Student Engagement. *Change: The Magazine of Higher Learning*, 43(1), 38–43.
- Ayub, M., Karnalim, O., Risal, R., Senjaya, W. F., & Wijanto, M. C. (2019). Utilising pair programming to enhance the performance of slow-paced students on Introductory Programming. *Journal of Technology and Science Education*, 9(3), 357–367.
- Azadegan, A., Riedel, J., & Hauge, J. (2012). Serious Games Adoption in Corporate Training. In M. Ma, M. Oliveira, J. Hauge, H. Duin, & K.-D. Thoben (Eds.), *Serious Games Development and Applications* (pp. 74–85). Springer.

- Azmi, S., Ahmad, N., Iahad, N. A., & Yusof, A. F. (2017, August 3). Promoting students' engagement in learning programming through gamification in peer-review discussion forum. *International Conference on Research and Innovation in Information Systems, ICRIIS*.
- Babbie, E. (2010). *The Practice of Social Research* (12th ed.). Wadsworth, Cengage Learning.
- Bai, S., Hew, K. F., & Huang, B. (2020). Does gamification improve student learning outcome? Evidence from a meta-analysis and synthesis of qualitative data in educational contexts. In *Educational Research Review* (Vol. 30). Elsevier Ltd.
- Bandura, A. (2006). Guide for Constructing Self-Efficacy Scales. In F. Pajares & T. Urdan (Eds.), *Self-Efficacy Beliefs of Adolescents* (Vol. 5, pp. 307–337). Information Age Publishing.
- Barata, G., Gama, S., Jorge, J., & Gonçalves, D. (2013). Improving participation and learning with gamification. *ACM International Conference Proceeding Series*, 10–17.
- Barlow-Jones, G., van der Westhuizen, D., & Coetzee, C. (2014). An Investigation into the Performance of First Year Programming Students in Relation to their Grade 12 Computer Subject Results. *Ed-Media*, 77–83.
- Baron, R. M., & Kenny, D. A. (1986). The Moderator-Mediator Variable Distinction in Social Psychological Research. Conceptual, Strategic, and Statistical Considerations. *Journal of Personality and Social Psychology*, 51(6), 1173–1182.
- Bau, D., Gray, J., Kelleher, C., Sheldon, J., & Turbak, F. (2017). Learnable programming: Blocks and beyond. In *Communications of the ACM* (Vol. 60, Issue 6, pp. 72–80). Association for Computing Machinery.
- Bawamohiddin, A. B., & Razali, R. (2017). Problem-based learning for programming education. *International Journal on Advanced Science, Engineering and Information Technology*, 7(6), 2035–2050.

- Bedwell, W. L., Pavlas, D., Heyne, K., Lazzara, E. H., & Salas, E. (2012). Toward a taxonomy linking game attributes to learning: An empirical study. *Simulation and Gaming, 43*(6), 729–760.
- Begosso, L. C., Begosso, L. R., & Christ, N. A. (2020). An Analysis of Block-Based Programming Environments for CS1. *2020 IEEE Frontiers in Education Conference (FIE)*.
- Begosso, L. R., Begosso, L. C., Cunha, D., Pinto, J. V., Lemos, L., & Nunes, M. (2018). The Use of Gamification for Teaching Algorithms. *Communication Papers of the 2018 Federated Conference on Computer Science and Information Systems, 17*, 225–231.
- Bellotti, F., Kapralos, B., Lee, K., Moreno-Ger, P., & Berta, R. (2013). Assessment in and of serious games: An overview. *Advances in Human-Computer Interaction, 1–11*.
- Bennedsen, J., & Caspersen, M. E. (2012). Persistence of elementary programming skills. *Computer Science Education, 22*(2), 81–107.
- Bíro, G. I. (2014). Didactics 2.0: A Pedagogical Analysis of Gamification Theory from a Comparative Perspective with a Special View to the Components of Learning. *Procedia - Social and Behavioral Sciences, 141*, 148–151.
- Bond, M., & Bedenlier, S. (2019). Facilitating student engagement through educational technology: Towards a conceptual framework. *Journal of Interactive Media in Education, 2019*(1).
- Bosse, Y., & Gerosa, M. A. (2017). Why is programming so difficult to learn? *ACM SIGSOFT Software Engineering Notes, 41*(6), 1–6.
- Boyer, N., Langevin, S., & Gaspar, A. (2008). Self Direction and Constructivism in Programming Education. *Proceedings of the 9th ACM SIGITE Conference on Information Technology Education., 89–94*.
- Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*(2), 77–101.

- Brown, N. C. C., Mönig, J., Bau, A., & Weintrop, D. (2016). Panel: Future Directions of Block-Based Programming. *SIGCSE 2016 - Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, 315–316.
- Bryman, A. (2006). Integrating Quantitative and Qualitative Research: How is it done? *Qualitative Research*, 6(1), 97–113.
- Bubica, N., Edu, M., & Boljat, I. (2014). Teaching of Novice Programmers: Strategies, Programming Languages and Predictors. *Proceedings International Conference on Information Technology and Development of Education ITRO 2014*.
- Buckley, P., & Doyle, E. (2016). Gamification and student motivation. *Interactive Learning Environments*, 24(6), 1162–1175.
- Butler, M., & Morgan, M. (2007). Learning challenges faced by novice programming students studying high level and low feedback concepts. *Proceedings Ascilite*, 99–107.
- Butt, P. (2016). Students' Perceptions of Game-Based Learning using CodinGame. *The International Conference on Information Communication Technologies in Education*, 151–158.
- Call, M. W., Fox, E., & Sprint, G. (2021). Gamifying Software Engineering Tools to Motivate Computer Science Students to Start and Finish Programming Assignments Earlier. *IEEE Transactions on Education*, 64(4), 423–431.
- Caponetto, I., Earp, J., & Ott, M. (2014). Gamification and Education: A Literature Review. *ITD-CNR*.
- Carini, R. M., Kuh, G. D., & Klein, S. P. (2006). Student engagement and student learning: Testing the linkages. *Research in Higher Education*, 47(1), 1–32.
- Casey, P. J. (1997). Computer programming: A medium for teaching problem solving. *Computers in the Schools*, 13(1–2), 41–51.

- Castleberry, A., & Nolen, A. (2018). Thematic analysis of qualitative research data: Is it as easy as it sounds? In *Currents in Pharmacy Teaching and Learning* (Vol. 10, Issue 6, pp. 807–815). Elsevier Inc.
- Casula, M., Rangarajan, N., & Shields, P. (2021). The potential of working hypotheses for deductive exploratory research. *Quality and Quantity*, 55(5), 1703–1725.
- Cazzola, W., & Olivares, D. M. (2016). Gradually learning programming supported by a growable programming language. *IEEE Transactions on Emerging Topics in Computing*, 4(3), 404–415.
- Celepko, M., & Boyer, K. E. (2018). Thematic analysis of students' reflections on pair programming in CS1. *SIGCSE 2018 - Proceedings of the 49th ACM Technical Symposium on Computer Science Education, 2018-January*, 771–776.
- Chang, C. S., Chung, C. H., & Chang, J. A. (2020). Influence of problem-based learning games on effective computer programming learning in higher education. *Educational Technology Research and Development*, 68(5), 2615–2634.
- Chang Chiung-Sui, Chen, J.-F., & Chen, F.-L. (2015). Development and Design of Problem-Based Learning Game-Based Courseware. *International Association for Development of the Information Society*, 217–219.
- Cheah, C. S. (2020). Factors contributing to the difficulties in teaching and learning of computer programming: A literature review. *Contemporary Educational Technology*, 12(2), 1–14.
- Chowdhury, B., Bart, A., & Kafura, D. (2018). Analysis of Collaborative Learning in a Computational Thinking Class. *SIGCSE '18 : Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 143–148.
- Cohen, L., Manion, L., & Morrison, K. (2018). *Research Methods in Education* (8th ed.). Routledge.
- Collis, J., & Hussey, R. (2009). *Business Research: A Practical Guide for Undergraduate and Postgraduate Students* (3rd ed.). Palgrave Macmillan.

- Creswell, J. (2014). *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches* (4th ed.). SAGE Publications, Inc.
- Cubukcu, C., Wang, B., Goodman, L., & Mangina, E. (2017). Gamification for Assessment of Object-Oriented Programming. *The International Conference on Information Communication Technologies in Education*, 226–237.
- Cutts, Q., Haden, P., Sutton, K., Box, I., Hamer, J., Fincher, S., Anthony Robins, kentacuk, Baker, B., de Raadt, M., Hamilton, M., Lister, R., Tolhurst, D., Petre, M., & Tutty, J. (2006). Computing Education. *Eighth Australasian Computing Education Conference*, 52, 181–188.
- Dambic, G., Kesscec, T., & Kucak, D. (2021). A Blended Learning with Gamification Approach for Teaching Programming Courses in Higher Education. *2021 44th International Convention on Information, Communication and Electronic Technology, MIPRO 2021 - Proceedings*, 843–847.
- de Lima, J. P. C., Carlos, L. M., ScharDOSim Simão, J. P., Pereira, J., Mafra, P. M., & da Silva, J. B. (2016). Design and implementation of a remote lab for teaching programming and robotics. *IFAC-PapersOnLine*, 49(30), 86–91.
- de Oliveira, T. A. N., & Reboucas, A. D. (2018). The use of pair programming to support introductory programming teaching: A qualitative study. *Proceedings - 13th Latin American Conference on Learning Technologies, LACLO 2018*, 65–68.
- de Pontes, R. G., Guerrero, D. D. S., & de Figueiredo, J. C. A. (2019). Analyzing gamification impact on a mastery learning introductory programming course. *SIGCSE 2019 - Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 400–406.
- de Raadt, M. (2007). A review of Australasian investigations into problem solving and the novice programmer. *Computer Science Education*, 17(3), 201–213.

- de Sena Quaresma, J. A., Ronaldo Bezerra Oliveira, S., & Eliasquevici, M. K. (2020). Validation of a Gamified Framework for Teaching and Learning for Algorithms Subject from a Control Group. *Proceedings - Frontiers in Education Conference, FIE, 2020-October*.
- de Sousa Borges, S., Durelli, V. H. S., Reis, H. M., & Isotani, S. (2014). A systematic mapping on gamification applied to education. *Proceedings of the ACM Symposium on Applied Computing*, 216–222.
- Deci, E. L., & Ryan, R. M. (2008). Self-determination theory: A macrotheory of human motivation, development, and health. *Canadian Psychology*, 49(3), 182–185.
- Dempsey, J. v., Haynes, L. L., Lucassen, B. A., & Casey, M. S. (2002). Forty simple computer games and what they could mean to educators. *Simulation and Gaming*, 33(2), 157–168.
- Denden, M., Tlili, A., Essalmi, F., Jemni, M., Chen, N. S., & Burgos, D. (2021). Effects of gender and personality differences on students' perception of game design elements in educational gamification. *International Journal of Human Computer Studies*, 154.
- Deng, W., Pi, Z., Lei, W., Zhou, Q., & Zhang, W. (2020). Pencil Code improves learners' computational thinking and computer learning attitude. *Computer Applications in Engineering Education*, 28(1), 90–104.
- Denny, P., Luxton-Reilly, A., Tempero, E., & Hendrickx, J. (2011). Understanding the Syntax Barrier for Novices. *Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education.*, 208–212.
- Denny, P., McDonald, F., Empson, R., Kelly, P., & Petersen, A. (2018). Empirical support for a causal relationship between gamification and learning outcomes. *Conference on Human Factors in Computing Systems - Proceedings, 2018-April*.
- Denscombe, M. (2014). *The Good Research Guide For Small-Scale Social Research Projects* (5th ed.). Open University Press.

- Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). From Game Design Elements to Gamefulness: Defining “Gamification.” *Proceedings of the 15th International Academic MindTrek Conference Envisioning Future Media Environments.*, 9–15.
- Dolgopolovas, V., Jevsikova, T., & Dagiene, V. (2018). From Android games to coding in C—An approach to motivate novice engineering students to learn programming: A case study. *Computer Applications in Engineering Education*, 26(1), 75–90.
- Domínguez, A., Saenz-De-Navarrete, J., De-Marcos, L., Fernández-Sanz, L., Pagés, C., & Martínez-Herráiz, J. J. (2013). Gamifying learning experiences: Practical implications and outcomes. *Computers and Education*, 63, 380–392.
- Doyle, L., Brady, A. M., & Byrne, G. (2009). An overview of mixed methods research. *Journal of Research in Nursing*, 14(2), 175–185.
- Duffany, J. L. (2014). Choice of Language for an Introduction to Programming Course. *Twelfth LACCEI Latin American and Caribbean Conference for Engineering and Technology (LACCEI'2014)*, 1–9.
- Durak, H. Y., & Guyer, T. (2019). Programming with Scratch in primary school, indicators related to effectiveness of education process and analysis of these indicators in terms of various variables. *Gifted Education International*, 35(3), 237–258.
- Eguchi, A. (2014). Robotics as a Learning Tool for Educational Transformation. *International Workshop Teaching Robotics, Teaching with Robotics & International Conference Robotics in Education*, 27–34.
- Facey-Shaw, L., Specht, M., van Rosmalen, P., & Bartley-Bryan, J. (2020). Do Badges Affect Intrinsic Motivation in Introductory Programming Students? *Simulation and Gaming*, 51(1), 33–54.
- Fan, K. K., Xiao, P. wei, & Su, C. H. (2015). The effects of learning styles and meaningful learning on the learning achievement of gamification health education curriculum. *Eurasia Journal of Mathematics, Science and Technology Education*, 11(5), 1211–1229.

- Ferrer, J., Ringer, A., Saville, K., A Parris, M., & Kashi, K. (2022). Students' motivation and engagement in higher education: the importance of attitude to online learning. *Higher Education, 83*(2), 317–338.
- Fidge, C., & Teague, D. (2009). Losing Their Marbles: Syntax-Free Programming for Assessing Problem-Solving Skills. *Proceedings of the Eleventh Australasian Computing Education Conference, 95*, 75–82.
- Figueiredo, J., & Garcia-Penalvo, F. (2020). Increasing Student Motivation in Computer Programming with Gamification. *2020 IEEE Global Engineering Education Conference (EDUCON)*, 997–1000.
- Figuroa-Flores, J. F. (2016). Gamification and Game-Based Learning: Two Strategies for the 21st Century Learner. *World Journal of Educational Research, 3*(2), 507.
- Fincher, S., Anthony Robins, kentacuk, Baker, B., Cutts, Q., Haden, P., Hamilton, M., Petre, M., Tolhurst, D., Box, I., de Raadt, M., Hamer, J., Lister, R., Sutton, K., & Tutty, J. (2006). Predictors of Success in a First Programming Course. *Proc. Eighth Australasian Computing Education Conference, 52*, 189–196.
- Fisher, M. J., & Marshall, A. P. (2009). Understanding descriptive statistics. *Australian Critical Care, 22*(2), 93–97.
- Fitzgerald, S., Lewandowski, G., McCauley, R., Murphy, L., Simon, B., Thomas, L., & Zander, C. (2008). Debugging: Finding, fixing and flailing, a multi-institutional study of novice debuggers. *Computer Science Education, 18*(2), 93–116.
- Florez-Aristizabal, L., Burbano, C., & Moreira, F. (2021). Towards a Computer Supported Collaborative Learning Approach for an Object-Oriented Programming Course. *World Conference on Information Systems and Technologies, 163–172*.
- Fotaris, P., Mastoras, T., Leinfellner, R., & Rosunally, Y. (2016). Climbing Up the Leaderboard: An Empirical Study of Applying Gamification Techniques to a Computer Programming Class. *Electronic Journal of E-Learning, 14*(2), 95–110.

- Fredericks, J., Blumenfeld, P., & Paris, A. (2004). School Engagement: Potential of the Concept, State of the Evidence. *Review of Educational Research*, 74(1), 59–109.
- Gage, A., & Murphy, R. R. (2003). Principles and Experiences in Using Legos to Teach Behavioral Robotics. *33rd Annual Frontiers in Education, 2003. FIE 2003*, 23–28.
- Gallego-Durán, F. J., Villagra-Arnedo, C., Llorens-Largo, F., & Molina-Carmona, R. (2017). PLMan: A Game-Based Learning Activity For Teaching Logic Thinking And Programming. *International Journal of Engineering Education*, 33(2), 807–815.
- Garcia-Iruela, M., Fonseca, M. J., Hijon-Neira, R., & Chambel, T. (2020). Gamification and computer science students' activity. *IEEE Access*, 8, 96829–96836.
- Garcia-Marquez, C., & Bauer, K. N. (2020). An Examination and Extension of the Theory of Gamified Learning: The Moderating Role of Goal Orientation. *Simulation and Gaming*, 1–28.
- Gari, M., Walia, G., & Radermacher, A. (2018). Gamification in Computer Science Education: a Systematic Literature Re-view. *American Society for Engineering Education*.
- Garris, R., Ahlers, R., & Driskell, J. E. (2002). Games, motivation, and learning: A research and practice model. *Simulation and Gaming*, 33(4), 441–467.
- Gibbs, G. R. (2018). Thematic Coding and Categorizing. In *Analyzing Qualitative Data* (pp. 53–74). SAGE Publications Ltd.
- Goksu, I., & Islam Bolat, Y. (2021). Does the ARCS motivational model affect students' achievement and motivation? A meta-analysis. *Review of Education*, 9(1), 27–52.
- Gomes, A., & Mendes, A. (2015). A teacher's view about introductory programming teaching and learning: Difficulties, strategies and motivations. *Proceedings - Frontiers in Education Conference, FIE, 2015-February*(February).
- Grant, C., & Osanloo, A. (2014). Understanding, Selecting, and Integrating a Theoretical Framework in Dissertation Research: Creating the Blueprint for Your “House.” *Administrative Issues Journal Education Practice and Research*, 4(2).

- Gray, D. (2014). *Doing Research in the Real World*. SAGE Publications Ltd.
- Greene, M. J. (2014). On the Inside Looking In: Methodological Insights and Challenges in Conducting Qualitative Insider Research. *The Qualitative Report*, 19(29), 1–13.
- Gros, B. (2007). Digital games in education: Me design of games-based learning environments. *Journal of Research on Technology in Education*, 40(1), 23–38.
- Hall, R. (2013). Mixed Methods: In Search of a Paradigm. In T. Le & Q. Le (Eds.), *Conducting Research in a Changing and Challenging World* (pp. 71–78). Nova Science Publishers Inc.
- Hamari, J., Shernoff, D. J., Rowe, E., Coller, B., Asbell-Clarke, J., & Edwards, T. (2016). Challenging games help students learn: An empirical study on engagement, flow and immersion in game-based learning. *Computers in Human Behavior*, 54, 170–179.
- Hammerschall, U. (2019). A Gamification Framework for Long-Term Engagement in Education Based on Self Determination Theory and the Transtheoretical Model of Change. *2019 IEEE Global Engineering Education Conference (EDUCON)*, 95–101.
- Harrington, B., & Chaudhry, A. (2017). TrAcademic: Improving participation and engagement in CS1/CS2 with gamified practicals. *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE, Part F128680*, 347–352.
- Heale, R., & Forbes, D. (2013). Understanding triangulation in research. In *Evidence-Based Nursing* (Vol. 16, Issue 4, p. 98).
- Hegazi, M. O., & Alhawarat, M. (2016). The challenges and the opportunities of teaching the introductory computer programming course: Case study. *Proceedings - 2015 5th International Conference on e-Learning, ECONF 2015*, 324–330.
- Heslin, P. A., & Klehe, U. C. (2006). Self-Efficacy. In *Encyclopedia of Industrial/Organisational Psychology* (Vol. 2, pp. 705–708).
- Holbl, M., Welzer, T., & Zlatolas, L. N. (2021, September 1). Students' Background Knowledge Influence on Learning Computer Programming. *Proceedings of the 2021 30th Annual*

Conference of the European Association for Education in Electrical and Information Engineering, EAEEIE 2021.

- Holvikivi, J. (2010). Conditions for Successful Learning of Programming Skills. In N. Reynolds & M. Turcsanyi-Szabo (Eds.), *Key Competencies in the Knowledge Society* (pp. 155–164). Springer.
- Hosseini, H., Hartt, M., & Mostafapour, M. (2019). Learning IS child's play: Game-based learning in computer science education. *ACM Transactions on Computing Education*, *19*(3).
- Hristova, M., Misra, A., Rutter, M., & Mercuri, R. (2003). Identifying and Correcting Java Programming Errors for Introductory Computer Science Students. *Proceedings of the Thirty-Fourth SIGCSE Technical Symposium on Computer Science Education : SIGCSE 2003*, 153–156.
- Hu, Y., Chen, C. H., & Su, C. Y. (2021). Exploring the Effectiveness and Moderators of Block-Based Visual Programming on Student Learning: A Meta-Analysis. *Journal of Educational Computing Research*, *58*(8), 1467–1493.
- Huang, B., & Hew, K. F. (2018). Implementing a theory-driven gamification model in higher education flipped courses: Effects on out-of-class activity completion and quality of artifacts. *Computers and Education*, *125*, 254–272.
- Ismail, M., Ngah, N., & Umar, I. (2010). Instructional Strategy in the Teaching of Computer Programming: A Need Assessment Analyses. *TOJET: The Turkish Online Journal of Educational Technology*, *9*(2), 125–131.
- Ivanovića, M., & Budimac, Z. (2013). First programming language - Never-ending story. *AIP Conference Proceedings*, *1558*, 353–356.
- Jackson, J., Cobb, M., & Carver, C. (2005). Identifying Top Java Errors for Novice Programmers. *Proceedings Frontiers in Education 35th Annual Conference*, 24–27.

- Jarvela, S., Jarvenoja, H., & Veermans, M. (2008). Understanding the Dynamics of Motivation in Socially Shared Learning. *International Journal of Educational Research*, 47(2), 122–135.
- Jenkins, T. (2002). On the Difficulty of Learning to Program. *3rd Annual HEA Conference for the ICS Learning and Teaching Support Network*, 1–8.
- Jiang, B., Zhao, W., Zhang, N., & Qiu, F. (2019). Programming trajectories analytics in block-based programming language learning. *Interactive Learning Environments*, 30(1), 113–126.
- João, P., Nuno, D., Fábio, S. F., & Ana, P. (2019). A Cross-Analysis of Block-Based and Visual Programming Apps with Computer Science Student-Teachers. *Education Sciences*, 9(3).
- Johnson, R., Onwuegbuzie, A., & Turner, L. (2007). Towards a Definition of Mixed Methods Research. *Journal of Mixed Methods Research*, 1(2), 112–133.
- Josko, J. M. B. (2021). Mixing Cognitive and Affective Approaches in Teaching Introductory Programming. *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*, 641.
- Kahu, E. R. (2013). Framing student engagement in higher education. *Studies in Higher Education*, 38(5), 758–773.
- Kanika, Chakraverty, S., & Chakraborty, P. (2020). Tools and Techniques for Teaching Computer Programming: A Review. *Journal of Educational Technology Systems*, 49(2), 170–198.
- Kaplan, R. (2010). Choosing a First Programming Language. *SIGITE '10 : Proceedings of the 2010 ACM Conference on Information Technology Education*, 163–164.
- Kapp, K. (2012). Theories Behind Gamification of Learning and Instruction. In *The Gamification of Learning and Instruction: Game-Based Methods and Strategies for Training and Education* (pp. 51–74).

- Kasahra, R., Sakamoto, K., Washizaki, H., & Fukazawa, Y. (2019). Applying Gamification to Motivate Students to Write High Quality Code in Programming Assignments. *ITiCSE '19: Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*, 92–98.
- Kavitha, R. K., JalajaJayalakshmi, V., & Rassika, R. (2018). Collaborative learning in Computer Programming Courses using E-Learning Environments. *International Journal of Pure and Applied Mathematics*, 118(8), 183–189.
- Kay, J., Barg, M., Fekete, A., Greening, T., Hollands, O., Kingston, J. H., & Crawford, K. (2000). Problem-Based Learning for Foundation Computer Science Courses. *Computer Science Education*, 10(2), 109–128.
- Kelly, L. M., & Cordeiro, M. (2020). Three principles of pragmatism for research on organizational processes. *Methodological Innovations*, 13(2).
- Khaleel, F. L., Ashaari, N. S., & Wook, T. S. M. T. (2019). An empirical study on gamification for learning programming language website. *Jurnal Teknologi*, 81(2), 151–162.
- Khaleel, F. L., Ashaari, N. S., & Wook, T. S. M. T. (2020). The impact of gamification on students learning engagement. *International Journal of Electrical and Computer Engineering*, 10(5), 4965–4972.
- Kim, B. (2015). Gamification in Education and Libraries. In *Understanding Gamification* (pp. 20–28). Amer Library Assn.
- King, A., & Eckersley, R. (2019). *Statistics for Biomedical Engineers and Scientists: How to Visualise and Analyse Data*. Academic Press.
- Kinzie, M. B., & Joseph, D. R. D. (2008). Gender differences in game activity preferences of middle school children: Implications for educational game design. *Educational Technology Research and Development*, 56(5–6), 643–663.

- Kivunja, C. (2018). Distinguishing between theory, theoretical framework, and conceptual framework: A systematic review of lessons from the field. *International Journal of Higher Education*, 7(6), 44–53.
- Kolikant, Y. B. D. (2010). Innovative teaching in computer science: What does it mean and why do we need it? In *Computer Science Education* (Vol. 20, Issue 2, pp. 73–78).
- Kolikant, Y. B. D., & Mussai, M. (2008). “So my program doesn’t run!” Definition, origins, and practical expressions of students’ (mis)conceptions of correctness. *Computer Science Education*, 18(2), 135–151.
- Krpan, D., Rosić, M., & Mladenović, S. (2014). Teaching Basic Programming Skills to Undergraduate Students. *Proceedings of CIET*, 147–158.
- Kumar, B., & Sharma, K. (2019). A Gamified Approach to Achieve Excellence in Programming. *Proceedings - 4th International Conference on Computing Sciences, ICCS 2018*, 107–114.
- Kwon, K., Ottenbreit-Leftwich, A. T., Brush, T. A., Jeon, M., & Yan, G. (2021). Integration of problem-based learning in elementary computer science education: effects on computational thinking and attitudes. *Educational Technology Research and Development*, 69(5), 2761–2787.
- Laal, M., & Ghodsi, S. M. (2012). Benefits of collaborative learning. *Procedia - Social and Behavioral Sciences*, 31, 486–490.
- Laal, M., & Laal, M. (2012). Collaborative learning: What is it? *Procedia - Social and Behavioral Sciences*, 31, 491–495.
- Lacher, L., Jiang, A., Zhang, Y., & Lewis, M. (2017). Aptitude and Previous Experience in CS1 Classes. *Int’l Conf. Frontiers in Education: CS and CE*, 87–95.
- Lahtinen, E., Ala-Mutka, K., & Järvinen, H. M. (2005). A study of the difficulties of novice programmers. *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, 14–18.

- Lameras, P., Arnab, S., Dunwell, I., Stewart, C., Clarke, S., & Petridis, P. (2017). Essential features of serious games design in higher education: Linking learning attributes to game mechanics. *British Journal of Educational Technology*, *48*(4), 972–994.
- Landers, R. N. (2014). Developing a Theory of Gamified Learning: Linking Serious Games and Gamification of Learning. *Simulation and Gaming*, *45*(6), 752–768.
- Landers, R. N., Auer, E. M., Collmus, A. B., & Armstrong, M. B. (2018). Gamification Science, Its History and Future: Definitions and a Research Agenda. *Simulation and Gaming*, *49*(3), 315–337.
- Landers, R. N., & Landers, A. K. (2015). An Empirical Test of the Theory of Gamified Learning: The Effect of Leaderboards on Time-on-Task and Academic Performance. *Simulation and Gaming*, *45*(6), 769–785.
- Leavy, P. (2017). *Research Design: Quantitative, Qualitative, Mixed Methods, Arts-Based, and Community-Based Participatory Research Approaches*. The Guilford Press.
- Lederman, N. G., & Lederman, J. S. (2015). What Is A Theoretical Framework? A Practical Answer. In *Journal of Science Teacher Education* (Vol. 26, Issue 7, pp. 593–597). Springer Netherlands.
- Lee, J. (2020). Statistics, Descriptive. In *International Encyclopedia of Human Geography* (pp. 13–20). Elsevier.
- Li, F., & Watson, C. (2011). Game-Based Concept Visualization for Learning Programming. *Proceedings of the Third International ACM Workshop on Multimedia Technologies for Distance Learning*, 37–42.
- Li, K., & Keller, J. M. (2018). Use of the ARCS model in education: A literature review. *Computers and Education*, *122*, 54–62.
- Li, L.-Y., Chang, C.-W., & Cheng, G.-D. (2009). Researches on Using Robotics in Education. In *Learning by Playing. Game-based Education System Design and Development* (pp. 479–482).

- Lister, R., Adams, E. S., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M., McCartney, R., Moström, J. E., Sanders, K., Seppälä, O., Simon, B., & Thomas, L. (2004). A multi-national study of reading and tracing skills in novice programmers. *Proceedings of the Conference on Integrating Technology into Computer Science Education, ITiCSE*, 119–150.
- Lister, R., Simon, B., Thompson, E., Whalley, J. L., & Prasad, C. (2006). Not Seeing the Forest for the Trees: Novice Programmers and the SOLO Taxonomy. *ITICSE '06: Proceedings of the 11th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, 118–122.
- Lopez, M., Whalley, J., & Lister, R. (2008). Relationships Between Reading, Tracing and Writing Skills in Introductory Programming. *ICER '08: Proceedings of the Fourth International Workshop on Computing Education Research*, 101–112.
- Lopez-Fernandez, D., Gordillo, A., Alarcon, P. P., & Tovar, E. (2021). Comparing Traditional Teaching and Game-Based Learning Using Teacher-Authored Games on Computer Science Education. *IEEE Transactions on Education*, 64(4), 367–373.
- Lopez-Pernas, S., Gordillo, A., Barra, E., & Quemada, J. (2019). Analyzing Learning Effectiveness and Students' Perceptions of an Educational Escape Room in a Programming Course in Higher Education. *IEEE Access*, 7, 184221–184234.
- Lorsbach, A. W., & Jinks, J. L. (1999). Self-Efficacy and Learning Environments SELF-EFFICACY THEORY AND LEARNING ENVIRONMENT RESEARCH. In *Learning Environments Research* (Vol. 2).
- Loughrey, K., & Broin, D. (2018). Are We Having Fun Yet? Misapplying Motivation to Gamification. *2018 IEEE Games, Entertainment, Media Conference (GEM)*, 529–533.
- Maenpaa, H., Varjonen, S., Hellas, A., Tarkoma, S., & Mannisto, T. (2017). Assessing IOT projects in university education - A framework for problem-based learning. *Proceedings - 2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering and Education Track, ICSE-SEET 2017*, 37–46.

- Marín, B., Frez, J., Cruz-Lemus, J., & Genero, M. (2019). An empirical investigation on the benefits of gamification in programming courses. *ACM Transactions on Computing Education*, 19(1).
- Martins, V. F., de Almeida Souza Concilio, I., & de Paiva Guimarães, M. (2018). Problem based learning associated to the development of games for programming teaching. *Computer Applications in Engineering Education*, 26(5), 1577–1589.
- Massoudi, M. (2019). A Review on Challenges and Solutions in Learning Programming Courses at Undergraduate Level. *International Journal of Applied Research and Studies*, 5(8), 146–149.
- Mathrani, A., Christian, S., & Ponder-Sutton, A. (2016). International Forum of Educational Technology & Society PlayIT: Game Based Learning Approach for Teaching Programming Concepts. *Journal of Educational Technology & Society*, 19(2), 5–17.
- McCauley, R., Fitzgerald, S., Lewandowski, G., Murphy, L., Simon, B., Thomas, L., & Zander, C. (2008). Debugging: A review of the literature from an educational perspective. In *Computer Science Education* (Vol. 18, Issue 2, pp. 67–92).
- McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Ben-David Kolikant, Y., Laxer, C., Thomas, L., Utting, I., & Wilusz, T. (2001). A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. *ITiCSE-WGR '01: Working Group Reports from ITiCSE on Innovation and Technology in Computer Science Education*, 125–180.
- Medeiros, R. P., Ramalho, G. L., & Falcao, T. P. (2019). A Systematic Literature Review on Teaching and Learning Introductory Programming in Higher Education. *IEEE Transactions on Education*, 62(2), 77–90.
- Meeplat, N. (2020). A model of creativity based learning for computer teaching to enhance creative skills of undergraduate students. *ACM International Conference Proceeding Series*, 184–188.

- Meyer, C. (2001). A Case in Case Study Methodology. *Field Methods*, 13(4), 329–352.
- Minor, J. T., & Gewali, L. P. (2004). Pedagogical Issues in Programming Languages. *International Conference on Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004.*, 562–565.
- Mladenović, M., Boljat, I., & Žanko, Ž. (2018). Comparing loops misconceptions in block-based and text-based programming languages at the K-12 level. *Education and Information Technologies*, 23(4), 1483–1500.
- Mohorovičić, S., & Strčić, V. (2011). An Overview of Computer Programming Teaching Methods. *Proceedings of the 22nd Central European Conference on Information and Intelligent Systems*, 47–52.
- Molaei, Z., & Dortaj, F. (2015). Improving L2 Learning: An ARCS Instructional-motivational Approach. *Procedia - Social and Behavioral Sciences*, 171, 1214–1222.
- Morales-Trujillo, M. E., & García-Mireles, G. A. (2021). Gamification and SQL: An Empirical Study on Student Performance in a Database Course. *ACM Transactions on Computing Education*, 21(1).
- Nah, F., Zeng, Q., Telaprolu, V., Ayyappa, A., & Eschenbrenner, B. (2014). Gamification of Education: A Review of Literature. In *HCI in Business* (pp. 401–418).
- Nair, S., & Mathew, J. (2019). A theoretical framework for gamified learning. *International Journal of Innovative Technology and Exploring Engineering*, 8(12), 2846–2851.
- Nelson, G. L., & Ko, A. J. (2018). On use of theory in computing education research. *ICER 2018 - Proceedings of the 2018 ACM Conference on International Computing Education Research*, 31–39.
- Oddie, A., Hazlewood, P., Blakeway, S., & Whitfield, A. (2010). Introductory Problem Solving and Programming: Robotics Versus Traditional Approaches. *Innovation in Teaching and Learning in Information and Computer Sciences*, 9(2), 1–11.

- O'Neil, H. F., Wainess, R., & Baker, E. L. (2005). Classification of learning outcomes: Evidence from the computer games literature. *Curriculum Journal*, 16(4), 455–474.
- Onwuegbuzie, A. J., & Johnson, R. B. (2006). The Validity Issues in Mixed Research. *Research in Schools*, 13(1), 48–63.
- Onwuegbuzie, A., & Leech, N. (2005). On becoming a pragmatic researcher: The importance of combining quantitative and qualitative research methodologies. In *International Journal of Social Research Methodology: Theory and Practice* (Vol. 8, Issue 5, pp. 375–387).
- Ortiz-Rojas, M., Chiluzia, K., & Valcke, M. (2019). Gamification through leaderboards: An empirical study in engineering education. *Computer Applications in Engineering Education*, 27(4), 777–788.
- Othman, M., Rosmani, A. F., Mohd Fauzi, S. S., & Mazlan, U. H. (2019). The Impact of Pair Programming on Students Logical Thinking_A Case Study on Higher Academic Institution. *Social and Management Research Journal*, 16(1), 85.
- Paas, F., Tuovinen, J. E., van Merriënboer, J. J. G., & Darabi, A. A. (2005). A motivational perspective on the relation between mental effort and performance: Optimizing learner involvement in instruction. *Educational Technology Research and Development*, 53(3), 25–34.
- Paiva, J. C., Leal, J. P., & Queirós, R. (2020). Fostering programming practice through games. *Information (Switzerland)*, 11(11), 1–20.
- Papadakis, S., & Kalogiannakis, M. (2019). Evaluating the effectiveness of a game-based learning approach in modifying students' behavioural outcomes and competence, in an introductory programming course. A case study in Greece. *International Journal of Teaching and Case Studies*, 10(3), 235.
- Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J., Devlin, M., & Paterson, J. (2007). A Survey of Literature on the Teaching of Introductory Programming. *ACM SIGCSE Bulletin*, 204–223.

- Perrotta, C., Featherstone, G., Aston, H., & Houghton, E. (2013). *Game-Based Learning: Latest Evidence and Future Directions*. National Foundation for Educational Research.
- Piedade, J., Dorotea, N., Pedro, A., & Matos, J. F. (2020). On teaching programming fundamentals and computational thinking with educational robotics: A didactic experience with pre-service teachers. *Education Sciences*, *10*(9), 1–15.
- Pivec, M., Dziabenko, O., & Schinnerl, I. (2003). Aspects of Game-Based Learning. *Proceedings of I-KNOW'03*, 216–225.
- Plass, J. L., Homer, B. D., & Kinzer, C. K. (2015). Foundations of Game-Based Learning. *Educational Psychologist*, *50*(4), 258–283.
- Pohl, M., Rester, M., & Judmaier, P. (2009). Interactive Game-Based Learning: Advantages and Disadvantages. In C. Stephanidis (Ed.), *Universal Access in Human-Computer Interaction* (pp. 92–101). Springer.
- Qian, M., & Clark, K. R. (2016). Game-based Learning and 21st century skills: A review of recent research. *Computers in Human Behavior*, *63*, 50–58.
- Qian, Y., & Lehman, J. (2017). Students' misconceptions and other difficulties in introductory programming: A literature review. In *ACM Transactions on Computing Education* (Vol. 18, Issue 1). Association for Computing Machinery.
- Quintão, C., Andrade, P., & Almeida, F. (2020). How to Improve the Validity and Reliability of a Case Study Approach? *Journal of Interdisciplinary Studies in Education*, *9*(2), 273–284.
- Radosevic, D., Orehovački, T., Lovrencic, A., Radošević, D., & Lovrenčić, A. (2009). New Approaches and Tools in Teaching Programming. *20th Central European Conference on Information and Intelligent Systems (CECIIS 2009)*.
- Rahman, M., Paudel, R., & Sharker, M. (2020). Effects of Infusing Interactive and Collaborative Learning to Teach an Introductory Programming Course. *2019 IEEE Frontiers in Education Conference*.

- Raj, A. G. S., Patel, J. M., Halverson, R., & Halverson, E. R. (2018, November 12). Role of live-coding in learning introductory programming. *ACM International Conference Proceeding Series*.
- Rivera, E. S., & Garden, C. L. P. (2021). Gamification for student engagement: a framework. *Journal of Further and Higher Education, 45*(7), 999–1012.
- Rizvi, M., & Humphries, T. (2012). A Scratch-based CSO course for at-risk computer science majors. *Proceedings - Frontiers in Education Conference, FIE*.
- Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *International Journal of Phytoremediation, 21*(1), 137–172.
- Rodrigues, L., Toda, A. M., Oliveira, W., Palomino, P. T., Avila-Santos, A. P., & Isotani, S. (2021). Gamification Works, but How and to Whom?: An Experimental Study in the Context of Programming Lessons. *SIGCSE 2021 - Proceedings of the 52nd ACM Technical Symposium on Computer Science Education, 184–190*.
- Rojas-López, A., Rincón-Flores, E. G., Mena, J., García-Peñalvo, F. J., & Ramírez-Montoya, M. S. (2019). Engagement in the course of programming in higher education through the use of gamification. *Universal Access in the Information Society, 18*(3), 583–597.
- Romero, M., Usart, M., & Ott, M. (2015). Can serious games contribute to developing and sustaining 21st century skills? In *Games and Culture* (Vol. 10, Issue 2, pp. 148–177). SAGE Publications Inc.
- Rosenberg, J. P., & Yates, P. M. (2007). Schematic representation of case study research designs. *Journal of Advanced Nursing, 60*(4), 447–452.
- Rubiano, S. M. M., López-Cruz, O., & Soto, E. G. (2015). Teaching Computer Programming: Practices, Difficulties and Opportunities. *2015 IEEE Frontiers in Education Conference (FIE)*.

- Rubin, M. (2013). The Effectiveness of Live-Coding to Teach Introductory Programming. *SIGCSE'13 : Proceedings of the 44th ACM Technical Symposium on Computer Science Education*, 651–656.
- Ryan, R. M., & Deci, E. L. (2000). Intrinsic and Extrinsic Motivations: Classic Definitions and New Directions. *Contemporary Educational Psychology*, 25(1), 54–67.
- Saidin, K. (2016). Insider Researchers: Challenges & Opportunities. *Proceedings of The ICECRS*, 1(1).
- Sailer, M., Hense, J. U., Mayr, S. K., & Mandl, H. (2017). How gamification motivates: An experimental study of the effects of specific game design elements on psychological need satisfaction. *Computers in Human Behavior*, 69, 371–380.
- Sajaniemi, J., & Prieto, R. N. (2005). Roles of variables in experts' programming knowledge. *17th Workshop of the Psychology of Programming Interest Group*, 145–159.
- Scepanovic, S., Zaric, N., & Matjevic, T. (2015). Gamification in Higher Education Learning - State of the Art, Challenges, and Opportunities. *The Sixth International Conference on E-Learning*.
- Seaborn, K., & Fels, D. I. (2015). Gamification in theory and action: A survey. *International Journal of Human Computer Studies*, 74, 14–31.
- Shabalina, O., Malliarakis, C., Tomos, F., & Mozelius, P. (2016). Game-Based Learning as a Catalyst for Creative Learning. *European Conference on Game Based Learning*, 589–598.
- Shahid, M., Wajid, A., & ul Haq, K. (2019). A Review of Gamification for Learning Programming Fundamental. *3rd International Conference on Innovative Computing (ICIC)*, 1–8.
- Shamlan, S. v., Killfoile, K., Kellogg, R., & Duvallet, F. (2006). Fun with robots: A student-taught undergraduate robotics course. *Proceedings - IEEE International Conference on Robotics and Automation, 2006*, 369–374.
- Shannon, A., & Summet, V. (2015). Live Coding in Introductory Computer Science Courses. *Journal of Computing Science in Colleges*, 31(2), 158–164.

- Sharma, M., Biroş, D., Ayyalasomayajula, S., & Dalal, N. (2020). Teaching Tip Teaching Programming to the Post-Millennial Generation: Pedagogic Considerations for an IS Course. *Journal of Information Systems Education*, 31(2), 96–105.
- Shorn, S. (2018). Teaching Computer Programming using Gamification. *Proceedings of the 14th International CDIO Conference*.
- Silva, L., Mendes, A., & Gomes, A. (2020). Computer-Supported Collaborative Learning in Programming Education: A Systematic Literature Review. *2020 IEEE Global Engineering Education Conference*.
- Simkova, M. (2014). Using of Computer Games in Supporting Education. *Procedia - Social and Behavioral Sciences*, 141, 1224–1227.
- Sirkia, T., & Sorva, J. (2012). Exploring Programming Misconceptions: An Analysis of Student Mistakes in Visual Program Simulation Exercises. *Koli Calling '12: Proceedings of the 12th Koli Calling International Conference on Computing Education Research*, 19–28.
- Smith, A., Mo, B., Taylor, S., Cheuoua, A. H., Minogue, J., Oliver, K., & Ringstaff, C. (2020). Designing block-based programming language features to support upper elementary students in creating interactive science narratives. *SIGCSE 2020 - Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 1327.
- Smith, D. H., Hao, Q., Jagodzinski, F., Liu, Y., & Gupta, V. (2019). Quantifying the Effects of Prior Knowledge in Entry-Level Programming Courses. *CompEd 2019 - Proceedings of the ACM Conference on Global Computing Education*, 30–36.
- Sobral, S. R. (2020). Is pair programming in higher education a good strategy? *International Journal of Information and Education Technology*, 10(12), 911–916.
- Soloway, E., & Spohrer, J. C. (2013). *Studying the Novice Programmer* (E. Soloway & J. C. Spohrer, Eds.). Psychology Press.

- Starr, C., Manaris, B., & Stalvey, R. (2008). Bloom's Taxonomy Revisited: Specifying Assessable Learning Objectives in Computer Science. *SIGCSE '08: Proceedings of the 39th ACM Technical Symposium on Computer Science Education*, 261–265.
- Stott, A., & Neustaedter, C. (2013). *Analysis of Gamification in Education*. 1–8.
- Swanborn, P. (2018). *Case Study Research: What, Why and How?* SAGE Publications, Inc.
- Tabet, N., Gedawy, H., Alshikhabobakr, H., & Razak, S. (2016). From alice to python. Introducing text-based programming in middle schools. *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE, 11-13-July-2016*, 124–129.
- Tasadduq, M., Khan, M. S., Nawab, R. M. A., Jamal, M. H., & Chaudhry, M. T. (2021). Exploring the effects of gamification on students with rote learning background while learning computer programming. *Computer Applications in Engineering Education*, 29(6), 1871–1891.
- Tay, L. (2010). *Employers: Look to Gaming to Motivate Staff*.
- Thota, N., & Whitfield, R. (2010). Holistic approach to learning and teaching introductory object-oriented programming. *Computer Science Education*, 20(2), 103–127.
- Tight, M. (2014). Discipline and theory in higher education research. *Research Papers in Education*, 29(1), 93–110.
- Tight, M. (2017). *Understanding Case Study Research: Small-scale Research with Meaning*. SAGE Publications Ltd.
- Tight, M. (2020). Student retention and engagement in higher education. *Journal of Further and Higher Education*, 44(5), 689–704.
- Toda, A. M., Oliveira, W., Klock, A. C., Palomino, P. T., Pimenta, M., Gasparini, I., Shi, L., Bittencourt, I., Isotani, S., & Cristea, A. I. (2019). A taxonomy of game elements for gamification in educational contexts: Proposal and evaluation. *Proceedings - IEEE 19th International Conference on Advanced Learning Technologies, ICALT 2019*, 84–88.
- Todd, A. (2017). *Why Gamification is Bullshit Malarkey*.

- Topalli, D., & Cagiltay, N. E. (2018). Improving programming skills in engineering education through problem-based game projects with Scratch. *Computers and Education, 120*, 64–74.
- Torrente, J., del Blanco, Á., Marchiori, E. J., Moreno-Ger, P., & Fernández-Manjón, B. (2010). <e-Adventure>: Introducing educational games in the learning process. *2010 IEEE Education Engineering Conference, EDUCON 2010*, 1121–1126.
- Trilles, S., & Granell, C. (2020). Advancing preuniversity students' computational thinking skills through an educational project based on tangible elements and virtual block-based programming. *Computer Applications in Engineering Education, 28*(6), 1490–1502.
- Trivedi, P., Kajgaonka, P., Kulkarni, A., Kolte, N., & Kanawade, B. (2019). 2019 Global Conference for Advancement in Technology (GCAT) : Bangalore, India, Oct 18-20, 2019. *2019 Global Conference for Advancement in Technology (GCAT)*, 1–5.
- Troussas, C., Krouska, A., & Sgouropoulou, C. (2020). Collaboration and fuzzy-modeled personalization for mobile game-based learning in higher education. *Computers and Education, 144*.
- Trowler, P. (2012). Wicked Issues in Situating Theory in Close Up Research. *Higher Education Research and Development, 31*(3), 273–284.
- Trowler, V., & Trowler, P. (2010). *Deliverable 2 for the Higher Education Academy Student Engagement Project*.
- Vainio, V., & Sajaniemi, J. (2007). Factors in Novice Programmers' Poor Tracing Skills. *ITiCSE 2007 : 12th Annual Conference on Innovation & Technology in Computer Science Education : Inclusive Education in Computer Science*, 236–240.
- van Dinther, M., Dochy, F., & Segers, M. (2011). Factors affecting students' self-efficacy in higher education. In *Educational Research Review* (Vol. 6, Issue 2, pp. 95–108).

- van Staalduinen, J.-P., & de Freitas, S. (2011). A Game-Based Learning Framework: Linking Game Design and Learning Outcomes. In M. Khine (Ed.), *Learning to Play: Exploring the Future of Education with Video Games* (pp. 29–54). Peter Lang.
- Vanthournout, G., Coertjens, L., Gijbels, D., Donche, V., & van Petegem, P. (2013). Assessing students' development in learning approaches according to initial learning profiles: A person-oriented perspective. *Studies in Educational Evaluation*, 39(1), 33–40.
- Versland, T. M. (2016). Exploring Self-Efficacy in Education Leadership Programs: What Makes the Difference? *Journal of Research on Leadership Education*, 11(3), 298–320.
- Walker, A., & Leary, H. (2009). A Problem Based Learning Meta Analysis: Differences Across Problem Types, Implementation Types, Disciplines, and Assessment Levels. In *The Interdisciplinary Journal of Problem-based Learning* • (Vol. 3, Issue 1).
- Werbach, K. (2014). (Re)Defining Gamification: A Process Approach. *International Conference on Persuasive Technology*, 266–272.
- Wilson, K. A., Bedwell, W. L., Lazzara, E., Salas, E., Burke, S. C., Estock, J. L., Orvis, K. L., & Conkey, C. (2009). Relationships between game attributes and learning outcomes: Review and research proposals. In *Simulation and Gaming* (Vol. 40, Issue 2, pp. 217–266).
- Wilson, V. (2014). Research Methods: Triangulation. *Evidence Based Library and Information Practice*, 9(1), 74–75.
- Wu, P. (2010). Practice and experience in the application of problem-based learning in computer programming course. *ICEIT 2010 - 2010 International Conference on Educational and Information Technology, Proceedings*, 1.
- Xie, B., Loksa, D., Nelson, G. L., Davidson, M. J., Dong, D., Kwik, H., Tan, A. H., Hwa, L., Li, M., & Ko, A. J. (2019). A theory of instruction for introductory programming skills. *Computer Science Education*, 29(2–3), 205–253.
- Yin, R. (2018). *Case Study Research and Applications: Design and Methods* (6th ed.). SAGE Publications Inc.

- Yuliati, L., Fauziah, R., & Hidayat, A. (2018). Student's critical thinking skills in authentic problem based learning. *Journal of Physics: Conference Series*, 1013(1).
- Zainal, Z. (2007). Case study as a research method. *Jurnal Kemanusiaan*.
- Zhang, X., Zhang, C., Stafford, T. F., & Zhang, P. (2013). Teaching Introductory Programming to IS Students: The Impact of Teaching Approaches on Learning Performance. In *Journal of Information Systems Education* (Vol. 24, Issue 2).
- Zhao, D., Muntean, C. H., Chis, A. E., Rozinaj, G., & Muntean, G. M. (2022). Game-Based Learning: Enhancing Student Experience, Knowledge Gain, and Usability in Higher Education Programming Courses. *IEEE Transactions on Education*.
- Zhu, J., Alderfer, K., Furqan, A., Nebolsky, J., Char, B., Smith, B., Villareale, J., & Ontañón, S. (2019, August 26). Programming in game space: How to represent parallel programming concepts in an educational game. *ACM International Conference Proceeding Series*.
- Zhu, J., Alderfer, K., Smith, B., Char, B., & Ontañón, S. (2020). *Understanding Learners' Problem-Solving Strategies in Concurrent and Parallel Programming: A Game-Based Approach*.
- Zimmerman, B. (2000). Self-Efficacy: An Essential Motive to Learn. *Contemporary Educational Psychology*, 25, 82–91.