Comparing Student Engagement in Educational Programming Games Utilising First-Person, Puzzle and Narrative Elements.

Theo Thesen
THSTHE004@myuct.ac.za
University of Cape Town

Cain Rademan
RDMCAI002@myuct.ac.za
University of Cape Town

1 PROJECT DESCRIPTION

Introductory programming has been shown to be a common point of attrition for students looking to study Computer Science [4, 5]. Part of this problem is that the current pedagogical approaches towards teaching introductory programming struggle to maintain student engagement. Solutions to this problem have been introduced, with Game-Based Learning (GBL) systems seeing a massive increase in popularity in recent years [18, 35, 36]. This is not without its faults though, as game-based learning in an introductory programming context has seen varied results in the literature [26]. This has been largely attributed to the fact that games, gamification and game-based systems are subject to too much variation and too vaguely defined to yield any pertinent research conclusions [25].This project intends to address the problem of student engagement in introductory programming courses by implementing two different game-based learning approaches. Each of these approaches are to be designed in such a way that they will focus on different elements of game-based systems that have been underrepresented in the literature. The first approach will focus on Puzzle and Point-of-View elements and the second will focus on Narrative elements in a game-based context.

2 PROBLEM STATEMENT

It has been definitively established that, in certain contexts, digital game-based learning can be an effective tool to increase student motivation and learning[6, 8, 13, 26, 42, 44]. However, less research has been done investigating, for a given learning domain, what type of game would be best suited to facilitate engagement. Digital games are typically complicated and varied systems utilising a host of different game design principles that are interconnected, making it difficult to benchmark or evaluate results when conducting research into educational games [25, 43]. This project hypothesises that when conducting research for a given learning domain, it is better to experimentally evaluate different game design elements in isolation to determine their impact on student engagement. In this case, the domain is that of introductory programming, an area of particular difficulty to first year Computer Science students.

Our research problem is the challenge of keeping students engaged in introductory programming courses using game-based learning. More specifically, we attempt to take meaningful steps toward solving the problem of student engagement in introductory programming courses by evaluating the impact that the following specific game design elements; point-of-view, puzzle, and narrative elements have on engagement, flow state, and fun. These will be explored in two separate studies, one focusing on point-of-view and puzzle elements, and the other focusing on narrative elements. The different motivations and approaches of each are discussed below, in sections 2.1 and 2.2 respectively.

2.1 Investigating Student Engagement in a First-Person Puzzle Game Designed to Teach Introductory Programming

First-person games are highly popular, with millions of players worldwide. [11]. Despite the prevalence of first-person perspective, there have been few programming games developed that utilise this point-of-view, either in academic studies or in a commercial context. Existing programming games tend to favour an isometric, top-down, or text-based approach [3, 26, 38]. First person perspective and its impact on player engagement is not well studied, though there is a general perception that utilising a first-person perspective increases immersion and interest in the game-world. There remains research to be done into whether a first-person programming game is effective in facilitating student engagement and increasing motivation.

It is possible that part of the reason why first-person serious games are under-represented in Computer Science Education is that there seems to be little conceptual link between first-person gaming and the activity of writing code. In this project, it is hypothesised that this disconnect can be solved by creating a first-person game utilising puzzle elements to more closely match the experience of coding. In the puzzle game genre, the player is required to solve puzzles and utilise their problem solving skills. This resembles the skills required to be an effective programmer - problem solving and programming are two closely related and mutually beneficial areas [23, 34]. Since puzzle elements offer players a chance to hone their problem solving abilities, and offer logical and conceptual challenges to a player, it can be

hypothesised that they can be used to create a compelling first-person programming game.

2.1.1 Research Question. How does the utilisation of first-person and puzzle elements affect engagement, flow state, and motivation of students in an educational game designed to teach introductory programming?

2.2 Motivating Game-Based Systems With Narrative Elements as a Pedagogical Tool in an Introductory Programming Course

Narratives are utilized in games, other forms of popular media, and education to tie linked concepts together to form an over-arching thematic concept [17, 32]. They have also been shown to be reliable sources of engagement and emotional attachment [7, 30].

Research indicates that Game-based learning systems work best as a pedagogical tool when utilized hand-in-hand with other, more traditional pedagogical methods [20, 25]. Furthermore, it has been demonstrated that students struggle to maintain engagement with the material taught in introductory programming courses [16].

This suggests that the introduction of narrative elements into game-based learning systems may help to solve some of the problems facing game-based learning systems as a pedagogical tool for student engagement in introductory programming courses. This is because narrative elements have been shown to be reliable sources of sustained, intrinsic engagement, which is an indicator of improved knowledge acquisition in an introductory programming context, [31] and introductory programming courses struggle to maintain [16]. Furthermore, narrative elements are useful as a tool to tie linked concepts together to form a thematic context. Therefore, the introduction of narrative elements into a game-based learning context may allow for teachers to easily associate concepts within the game-based system with concepts taught using other pedagogical strategies.

2.2.1 Research Question. How do game-based learning systems with strong narrative elements effect engagement, flow state, and motivation of students in an introductory programming course?

3 PROCEDURES & METHODS

For the purposes of answering our research questions, we will develop two educational games, one utilising puzzle and POV game elements, and one utilising narrative elements. The games will be designed to reinforce basic programming concepts such as variables, if-statements and loops, using the language Python. It will be assumed that students understand basic Python syntax and have encountered the programming constructs (such as variables and if-statements) utilised in the game before. The intention of the games will not be to teach students coding from the beginning, but to reinforce or cement the ideas they have already learned in a visual, engaging way. The gameplay will consist of players writing and running code to manipulate the environment and advance in the game. Each game will be experimented on and evaluated independently, and each will ultimately aim to answer different research questions (listed above). They will however, share certain design and core technical elements, for the purposes of efficiency and non-duplication of work. In particular, they will share graphical models, and a Python to C# interpreter. The games will be implemented using the popular real-time development platform Unity, as it is recognized as one of the most effective tools to realize games and game-based systems. Overviews of each game and its core mechanics are detailed in the following sections 3.1 and 3.2.

3.1 Game-Based System with First-Person and Puzzle Elements

For the purposes of answering the research question in 2.1.1, an educational first-person puzzle game will be developed. The gameplay will involve the player writing Python code to navigate a 3D environment, using a first person point of view. The player will control the movements and actions of a rover/robot through code to explore an undiscovered planet. They will begin at a starting point and have to navigate the environment to reach an endpoint, potentially avoiding enemies or danger. The player will have to utilise their problem solving abilities to work out how to get the robot to do what they want.

If the essential functionality of the game is implemented and there is time remaining, the plan is to create at least 3 levels which the player progresses through sequentially. In the first level the player will learn the basics of how to move the robot. As the player moves through the levels, the difficulty may ramp up and the player may be expected to utilise the concepts they have learned to solve more challenging puzzles. The idea is that puzzle solving will simulate the problem solving required to code, and show the player that solving coding problems can be fun and exciting.

3.2 Game-Based System with Narrative Elements

To test the effect of narrative elements as a pedagogical tool for fostering student engagement in a game-based introductory programming context, a game-based 3D system will be built wherein a student controls a character and needs to navigate using the WASD keys. Two modes of this game-based system will be available. One, with narrative elements and one without. The narrative mode will provide the students with a cut-scene before they start playing the game, which will show an astronaut crashing on a hostile planet and their spaceship going dark. After this, the student will be encouraged to restore the electricity, water system and hydroponic farm on their crashed ship to 'win' the game. Each of these acts of restoration will require the student to write Python scripts that fix or build functionality for the aforementioned

tasks. Throughout the game-based system, a running narrative will be provided to the student, attempting to build some emotional connection and provide some sense of transportation to the game-based system. The mode without a narrative element will strip back on all of the world-building, context-based or narrative elements and simply have the students write Python code in a game-based environment with no overarching contextualization or attempts at forming an emotional connection.

3.3 Fundamental Programming Concepts

The 2013 ACM Software Development Fundamentals [37] identifies the essential competencies that an undergraduate computer scientist must develop. The competency "Fundamental Programming Concepts" encompasses basic concepts of programming languages that students must become proficient in. For the purposes of this project we have selected a subset of these fundamental programming concepts to be taught in the game-based systems developed. These are listed below:

1. Basic syntax and semantics of a higher-level language
2. Variables and primitive data types (e.g numbers, characters, Booleans)
3. Expressions and assignments
4. Conditional and iterative control structures
   - if-statements
   - for-loops
   - while-loops
5. Functions and parameter passing (time-permitting)
   - Players will not need to write functions, but may need to call existing functions

3.4 Python to C# Interpreter

Since scripts in Unity are written in the language C#, and players will be writing Python code, it will be necessary to make use of a Python-to-C# interpreter to execute the player's scripts. IronPython [14] is an open-source implementation of the Python programming language which is tightly integrated with .NET (the framework upon which C# is built). Using IronPython, Python programs can integrate with applications written in other .NET programming languages, such as C#. We will use IronPython to interpret the user's Python code during runtime, which will allow them to manipulate objects in the Unity environment. Although both games will be separate implementations and will facilitate different research questions, they will both use this interpreter, for the purposes of efficiency.

3.4.1 Writing Code to Manipulate the Environment.

Players will write Python code in a text editor and run it to manipulate the game environment. The text editor used will support syntax highlighting and error-checking. Players will reference different items in the game using an object name, and perform actions on it using the methods associated with that object. For example, to move their player forward, they might write "player.moveForward()". They will play through levels where the objective is to solve a problem by inputting the correct code. For each level, they will get information about what objects and methods are available to them. They will also be able to use other basic Python statements such as variables, if-statements, for-loops and while-loops.

On a technical level, the effect of the user writing code to interact with the game environment may be challenging to achieve. There will need to be some mapping of the objects and methods referenced in the user's Python code to those in the C# unity environment. GameObjects are the fundamental objects in Unity that represent characters, props and scenery. If the player inputs for example, "player.jump()", the interpreter will need to map the inputted Python object and method to the corresponding GameObject in Unity and call the "jump()" method in the C# code.

3.5 Implementation strategy

Although separate pieces of software, both games have a similar priority in terms of core components that must be provided. These are ranked according to the following priority system:

- 3 - Essential features that must be implemented
- 2 - Important to have
- 1 - Non-essential/extra

Below can be found a grouping by priority of the proposed features:

Essential features (Priority 3):

- Python to C# interpreter for Unity
- Basic Game environment and models
- Player is able to write and compile code in a text editor
- At least 1 level in which player writes code to control game objects
- Rudimentary narrative features or puzzle elements (game specific)

Important features (Priority 2):

- Menu/system where player can navigate the game
- Syntax highlighting
- Multiple levels

Non-essential features (Priority 1):

- Music and sound design
- Aesthetically pleasing graphics

For each game, an iterative development strategy will be used to deliver these features, with at least 2 cycles consisting of 1) analysis and design, 2) implementation and 3) testing. The first cycle will see the development of a high fidelity prototype implementing at least the essential "priority 3" features. If there is time remaining, lower priority features

may be implemented as well, which, although non-essential, will contribute to the cohesion and quality of the system. The second cycle will see the refinement of the prototype based on the feedback from the first cycle.

3.6 Testing and Evaluation

At the end of both cycles of development, user testing and evaluation will be done, using the prototypes developed. Participants recruited will be first year Computer Science students at the University of Cape Town. Since testing will be done in the latter half of the year, these students will have been exposed to Python and the programming constructs used in the game, while still being at a "beginner" level, making them ideal test subjects. Testing will be conducted online in light of the risks of in-person meetings due to the COVID-19 pandemic.

The testing methodologies used at the end of both cycles will be identical, so that the two prototypes can be compared and it can be ascertained whether the second attempt improves upon the first. Participants will be sent an executable version of the game, and will run and play the game, giving feedback and comments as they play. They will share their screen while doing so to allow the tester to record their reactions to specific game elements. Participants will play the game without extensive instruction from the tester - any struggles that they have will expose ambiguities or issues in the design. Afterwards, they will fill out a qualitative survey assessing their impressions of the legibility and usability of the software, as well as their subjective feelings of engagement, fun and flow state while playing. The survey will incorporate the User Engagement Scale (UES) [28, 29, 40] in a game-based context [40], as well as the GameFlow questionnaire [22] to measure effects of transportation [17], identification [9] and flow [39] within the game-based system. The survey will also ascertain their level of enthusiasm towards this implementation of game based learning, and whether they think it would have been beneficial and motivating as part of their introductory programming course.

4 ETHICAL, LEGAL & PROFESSIONAL ISSUES

4.1 Ethical

Ethical considerations with regards to the process of testing our game-based systems have been accounted for. Students from the CSC1010H, CSC1011H and potentially CSC1016S Computer Science courses at the University of Cape Town will be contacted. No compensation will be offered and students will only undertake the exercise through willing participation.

Students will be asked to engage with one of the game-based learning systems for around an hour and then will undertake a short questionnaire that has been designed to utilize the User Engagement Scale, as explained above.

Furthermore, all students will be informed of their role within the research and will provide free, informed, prior consent before they are to be involved. Their confidentiality will also be maintained as all data will be obtained without identifiers. For data gathering purposes, each student will be identified by the order in which they underwent the questionnaire. E.g., the first student to take the questionnaire will be referred to as 'Student 1', and so on.

Taking student safety and the COVID-19 pandemic into account, there will be absolutely no face-to-face interaction and the entire process of recruiting, sending the game-based learning system to the student and conducting the questionnaire will be done online.

4.2 Legal

The system will be entirely comprised of either open source software or software and assets that have been licensed or bought.

We will be using Unity Personal Edition to make both of the game-based systems. Unity Personal Edition is free to use and share creations on if the individual using it makes less than USD 100'000 a year. We will also make use of the Unity Asset store to purchase additional assets which operate under the Standard Unity Asset Store EULA. Any additional assets will be created using Blender, a free and open source 3D creation software.

5 RELATED WORK

5.1 Game-based Learning

Game-based learning systems have received significant attention in the world of pedagogy, specifically as applied to Computer Science education [15, 26, 27, 41, 42, 44]. They operate in a similar domain to gamification, which is a strategic approach that involves incorporating game design mechanics, that are inherently motivating, into a system [18]. However, they differ from conventional gamification in the sense that they do not borrow features and mechanics from game design but rather leverage video games themselves as a distinct medium of conveying information [2, 10]. There is much literature covering different use cases of game-based learning systems but the results are mixed, although largely positive [24]. This has been ascribed to the fact that games, and therefore game-based systems are too loosely defined to yield consistent results [25, 43]. It has therefore been proposed that a better approach towards evaluating the effectiveness of game-based learning systems in an introductory programming context is to isolate specific elements of games and test their effectiveness individually [43].

5.2 Narrative Elements in Game-based Learning

Narrative elements in the context of game-based learning are underrepresented in the literature. This can be explained, in part, by the lack of rigorous, peer-reviewed research on the effects of individual elements that constitute games, as described in the section above. Another factor to consider is the fact that narrative elements are not exclusive to games and are more closely associated with other forms of media (film, print, etc.) [32].

Regardless of the reasons, there has not been substantial research into the effect of narrative elements in a game-based learning context [20]. However, it has been shown that factors that keep students engaged in a game-based learning system are related to how emotionally invested a user is with the game-based learning environment [1]. This is significant as narrative elements in games have been shown to be the greatest source of emotional investment with a game [9]. It then makes sense that this emotional attachment as a result of narrative elements would follow through from gaming to game-based systems.

5.3 Point of View Elements in Game-based Learning

There has been little research into point-of-view as a feature of educational games, both in the realm of Computer Science and otherwise. In a literature review conducted prior to this study [33], it was found that a large number of game based systems in research utilise an isometric, top-down, or text-based approach, but there was little motivation for choice of these systems. There were no examples found that utilised a first person perspective.

5.4 Puzzle Elements in Educational Programming Games

There have been several useful literature reviews on the use of game based learning to teach Computer Science [3, 26, 38]. These provide an overview of the content taught and methodologies used in serious programming games thus far. Blanco and Engstrom provide analyses of game genre used [3]. Their findings showed that in commercial programming games, puzzle games are the overwhelming majority. They noted that in academic papers, this is not as pronounced, although there is still a strong representation of traditional puzzle-type games. Based on the success of programming games utilising puzzle elements [12, 19, 21], there is evidence that the puzzle genre is ideal for teaching coding. This could be attributed to the fact that programming and puzzle games both involve problem solving. The link between problem solving and coding is backed up in the literature. It has been shown that programming improves cognitive reasoning skills[34]. It has also been shown that teaching problem solving before programming in an effective strategy in introductory programming courses to improve learning in students [23].

6 ANTICIPATED OUTCOMES

6.1 System

In both cases, game-based learning systems will be created with a particular emphasis on the isolated elements of gaming for which research is being conducted.

6.1.1 Narrative driven Game-based System. The narrative-driven game-based system is expected to increase a users emotional attachment and therefore engagement with the game-based system. Furthermore, the introduction of narrative elements will hopefully allow students to more easily associate concepts taught within the game-based system to concepts that are introduced using more traditional pedagogical methods.

6.1.2 Puzzle and POV Game-based System. The POV and puzzle game-based system is expected to facilitate engagement by utilising a first-person perspective to increase immersion and interest in the game-world. Additionally, it is hoped that the puzzle elements will exercise students' problem solving abilities, and improve their perceptions of coding as a fun activity.

6.2 Expected Impact

There is currently no significant research on the effect of particular elements of game design on student engagement in an introductory programming context. We are looking to address that gap in the research by producing new findings on the effect of narrative, puzzle and point-of-view elements on student engagement. We anticipate that this research will bolster similar studies conducted about the effectiveness of game-based systems as a learning tool and how they interact with student engagement. If these experiments prove to be successful, we hope to present point-of-view, puzzle and narrative elements as effective elements of games to utilize when creating effective game-based learning systems.

6.3 Success Factors

- Successful implementation of two game-based learning systems focused around different and specific elements of gaming.
- Yielding meaningful and positive results associated with the use of point-of-view and puzzle elements in game-based systems and student engagement.
- Yielding meaningful and positive results associated with the use of narrative elements and student engagement.

7 PROJECT PLAN

7.1 Risks

A risk-matrix is included in Appendix A. This lists potential risks for this project, as well as each risk's likelihood of occurrence, impact and consequences. Also included are strategies for risk mitigation, monitoring and management.

7.2 Timeline

Detailed Gantt chart provided in Appendix B.

7.3 Resources

- Computers capable of running Unity.
- Reliable access to the internet.
- IronPython open-source framework.
- Unity Asset Store access.
- Blender open-source modelling software.
- Access to an appropriate C# IDE.

7.4 Deliverables

The primary deliverables for this research are the two final research papers which will analyze, in detail, the implementation, execution and results of both of the aforementioned methods to hopefully foster student engagement in introductory programming using game-based learning systems with a primary focus on different elements of games. Other noteworthy deliverables are:

- A literature review of puzzle and point-of-view elements in game-based learning systems.
- A literature review of narrative elements in game-based learning systems.
- A project proposal.
- A project feasibility presentation.
- Drafts of final research papers.
- Source code for both projects.
- Overall project posters and a website.

7.5 Milestones

The primary milestones of this research are the implementation of two game-based learning systems that take different approaches to foster student engagement and the submission of two final research papers.

7.6 Work Allocation

Theo Thesen will create all 3D models, shared Unity design patterns and develop the game-based learning system which focuses on narrative elements. Cain Rademan will implement the Python to C# interpreter, syntax highlighting and error labelling for said interpreter, any necessary sound design in the project and will develop the game-based learning system that focuses on puzzle and point-of-view elements.

References

[1] Azita Iliya Abdul Jabbar and Patrick Felicia. Gameplay engagement and learning in game-based learning: A systematic review. *Review of educational research*, 85(4):740–779, 2015.

[2] Rula Al-Azawi, Fatma Al-Faliti, and Mazin Al-Blushi. Educational gamification vs. game based learning: Comparative study. *International Journal of Innovation, Management and Technology*, 7(4):132–136, 2016.

[3] Ander Areizaga Blanco and Henrik Engström. Patterns in mainstream programming games. *International Journal of Serious Games*, 7(1):97 – 126, Mar. 2020.

[4] Jessica D Bayliss. Using games in introductory courses: tips from the trenches. In *Proceedings of the 40th ACM technical symposium on Computer science education*, pages 337–341, 2009.

[5] Jens Bennedsen and Michael E Caspersen. Failure rates in introductory programming. *AcM SIGcSE Bulletin*, 39(2):32–36, 2007.

[6] Elizabeth A Boyle, Thomas Hainey, Thomas M Connolly, Grant Gray, Jeffrey Earp, Michela Ott, Theodore Lim, Manuel Ninaus, Claudia Ribeiro, and João Pereira. An update to the systematic literature review of empirical evidence of the impacts and outcomes of computer games and serious games. *Computers and education*, 94:178–192, 2016.

[7] Rick Busselle and Helena Bilandzic. Measuring narrative engagement. *Media Psychology*, 12(4):321–347, 2009.

[8] Chi-Cheng Chang, Chaoyun Liang, Pao-Nan Chou, and Guan-You Lin. Is game-based learning better in flow experience and various types of cognitive load than non-game-based learning? perspective from multimedia and media richness. *Computers in human behavior*, 71:218–227, 2017.

[9] Jonathan Cohen. Defining identification: A theoretical look at the identification of audiences with media characters. *Mass communication & society*, 4(3):245–264, 2001.

[10] Brianno D Coller and David J Shernoff. Video game-based education in mechanical engineering: A look at student engagement. *International Journal of Engineering Education*, 25(2):308, 2009.

[11] Valve Corporation. Steam & game stats.

[12] Michael Eagle and Tiffany Barnes. Experimental evaluation of an educational game for improved learning in introductory computing. In *Proceedings of the 40th ACM technical symposium on computer science education*, SIGCSE '09, pages 321–325. ACM, 2009.

[13] Mohd. Elmagzoub Eltahir, Najeh Rajeh Alsalhi, Sami Al-Qatawneh, Hatem Ahmad AlQudah, and Mazan Jaradat. The impact of game-based learning (gbl) on students' motivation, engagement and academic performance on an arabic language grammar course in higher education. *Education and information technologies*, 26(3):3251–3278, 2021.

[14] .NET Foundation. Ironpython.

[15] Stefanos Galgouranas and Stelios Xinogalos. javant-garde: A cross-platform serious game for an introduction to programming with java. *Simulation gaming*, 49(6):751–767, 2018.

[16] Anabela Gomes and Antonio Mendes. A teacher's view about introductory programming teaching and learning: Difficulties, strategies and motivations. In *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*, pages 1–8. IEEE, 2014.

[17] Melanie C Green and Timothy C Brock. The role of transportation in the persuasiveness of public narratives. *Journal of personality and social psychology*, 79(5):701, 2000.

[18] Juho Hamari, Jonna Koivisto, and Harri Sarsa. Does gamification work?–a literature review of empirical studies on gamification. In *2014 47th Hawaii international conference on system sciences*, pages 3025–3034. Ieee, 2014.

[19] Andrew Hicks. Towards social gaming methods for improving game-based computer science education. In *Proceedings of the Fifth International Conference on the foundations of digital games*, FDG '10, pages 259–261. ACM, 2010.

[20] Maria-Blanca Ibanez, Angela Di-Serio, and Carlos Delgado-Kloos. Gamification for engaging computer science students in learning activities: A case study. *IEEE Transactions on learning technologies*, 7(3):291–301, 2014.

[21] Cagin Kazimoglu. Enhancing confidence in using computational thinking skills via playing a serious game: A case study to increase motivation in learning computer programming. *IEEE Access*, 8:221831–221851, 2020.

[22] Kristian Kiili and Timo Lainema. Foundation for measuring engagement in educational games. *Journal of Interactive Learning Research*, 19(3):469–488, 2008.

[23] Theodora Koulouri, Stanislao Lauria, and Robert D Macredie. Teaching introductory programming: A quantitative evaluation of different approaches. *ACM transactions on computing education*, 14(4):1–28, 2015.

[24] Ming-Chaun Li and Chin-Chung Tsai. Game-based learning in science education: A review of relevant research. *Journal of Science Education and Technology*, 22(6):877–898, 2013.

[25] Katie Larsen McClarty, Aline Orr, Peter M Frey, Robert P Dolan, Victoria Vassileva, and Aaron McVay. A literature review of gaming in education. *Gaming in education*, pages 1–35, 2012.

[26] Michael Miljanovic and Jeremy Bradbury. A review of serious games for programming. 11 2018.

[27] Mathieu Muratet, Patrice Torguet, Jean-Pierre Jessel, and Fabienne Viallet. Towards a serious game to help students learn computer programming. *International journal of computer games technology*, 2009(1):1–12, 2009.

[28] Heather L O'Brien, Paul Cairns, and Mark Hall. A practical approach to measuring user engagement with the refined user engagement scale (ues) and new ues short form. *International Journal of Human-Computer Studies*, 112:28–39, 2018.

[29] Heather L O'brien and Elaine G Toms. Examining the generalizability of the user engagement scale (ues) in exploratory search. *Information Processing & Management*, 49(5):1092–1107, 2013.

[30] Trena M Paulus, Brian Horvitz, and Min Shi. 'isn't it just like our situation?'engagement and learning in an online story-based environment. *Educational Technology Research and Development*, 54(4):355–385, 2006.

[31] Nikolaos Pellas. Exploring interrelationships among high school students' engagement factors in introductory programming courses via a 3d multi-user serious game created in open sim. *J. UCS*, 20(12):1608–1628, 2014.

[32] Hua Qin, Pei-Luen Patrick Rau, and Gavriel Salvendy. Measuring player immersion in the computer game narrative. *Intl. Journal of Human–Computer Interaction*, 25(2):107–133, 2009.

[33] Cain Rademan. Literature review of game-based learning in computer science education. pages 3–4, 2021.

[34] Ronny Scherer, Fazilat Siddiq, and Bárbara Sánchez Viveros. The cognitive benefits of learning computer programming: A meta-analysis of transfer effects. *Journal of educational psychology*, 111(5):764–792, 2019.

[35] Katie Seaborn and Deborah I Fels. Gamification in theory and action: A survey. *International Journal of human-computer studies*, 74:14–31, 2015.

[36] Gabriela Silva-Maceda, P David Arjona-Villicana, and F Edgar Castillo-Barrera. More time or better tools? a large-scale retrospective comparison of pedagogical approaches to teach programming. *ieee Transactions on Education*, 59(4):274–281, 2016.

[37] The Association for Information Systems (AIS) The Joint Task Force: Association for Computing Machinery (ACM) and IEEE Computer Society. Computer science curricula 2013: Curriculum guidelines for undergraduate degree programs in computer science, December 2013.

[38] Adilson Vahldick, Antonio Jose Mendes, and Maria Jose Marcelino. A review of games designed to improve introductory computer programming competencies. In *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*. IEEE, 2014.

[39] Joar Vittersø. Mihaly csikszentmihalyi, finding flow. the psychology of engagement with everyday life. *Journal of Happiness Studies*, 1(1):121–123, 2000.

[40] Eric N Wiebe, Allison Lamb, Megan Hardy, and David Sharek. Measuring engagement in video game-based environments: Investigation of the user engagement scale. *Computers in Human Behavior*, 32:123–132, 2014.

[41] Stelios Xinogalos. Programming serious games as a master course: Feasible or not? *Simulation gaming*, 49(1):8–26, 2018.

[42] Mirac Yallihep and Birgul Kutlu. Mobile serious games: Effects on students' understanding of programming concepts and attitudes towards information technology. *Education and information technologies*, 25(2):1237–1254, 2020.

[43] Michael F Young, Stephen Slota, Andrew B Cutter, Gerard Jalette, Greg Mullin, Benedict Lai, Zeus Simeoni, Matthew Tran, and Mariya Yukhymenko. Our princess is in another castle: A review of trends in serious gaming for education. *Review of educational research*, 82(1):61–89, 2012.

[44] Dan Zhao, Cristina Hava Muntean, Adriana E Chis, and Gabriel-Miro Muntean. Learner attitude, educational background, and gender influence on knowledge gain in a serious games-enhanced programming course. *IEEE transactions on education*, pages 1–9, 2021.

A Risk Matrix

Risk	Consequence	Probability	Impact	Mitigation	Monitoring	Management
Initial scope chosen to be too large	Inability to deliver on promised scope, resulting in unfinished project and potential inability to evaluate research questions	High	Critical	Under-promise on scope instead of over-promising. Make sure to implement essential features first	Measure actual progress against the project timeline to determine whether features are being implemented too slowly	Should it become clear that the scope is too large, decrease scope to more manageable level
Bottleneck while one team member waits for work from the other	Project disruption - cannot continue project until work is finished. Progress is delayed.	High	Marginal	Share as few components as possible between game-based systems. Prioritise tasks that other team members are dependent on	Undertake regular meetings to communicate about progress	Should a bottleneck occur, both team members will work on the task to ensure it gets done quickly
Inability to get Unity C# environment to integrate with Python scripts	Students will be unable to write code, resulting in an incomplete programming game. Research questions will not be testable	Medium	Critical	Start on this feature early, focusing on getting a working prototype as soon as possible	Check whether this feature is being developed in accordance with the project timeline	Switch to a different environment that is more integrated with Python, such as Blender or Panda3D.
Ethical clearance for user testing obtained late	Inability to proceed with testing, resulting in project delay	Medium	Critical	Submit ethical clearance form as soon as possible, and in accordance with the deadline	Regularly check in on whether the form has been returned yet. Follow up with ethics committee if the process is taking very long	If ethical clearance is not obtained in time, conduct testing later than planned. It may be necessary to forgo testing for the first prototype
Team member contracts COVID-19	Infected team member cannot work, resulting in massive delays in output. Other team member may be left waiting for work they are dependant on.	Low	Critical	Practise COVID-19 safety and take recommended precautions to limit chance of contracting the virus.	Monitor physical health closely. Team members must get tested if they suspect they may have the virus	Team members should keep logs of their work, communicate to partner, and back up regularly so that in the event of infection, the other can help to finish their work

Table 1. Risk Matrix

B Gantt Chart