Game-Based Learning of Data Structures Based on Analogies: Learning Gains and Intrinsic Motivation in Higher Education Environments

A thesis submitted to

Trinity College Dublin, The University of Dublin

in fulfilment of the requirements for the degree of Doctor of Philosophy

by

Alberto José Rojas-Salazar

March 2022

Declaration

I, Alberto José Rojas-Salazar, declare that this thesis has not been submitted as an exercise for a degree at this or any other university and it is entirely my own work.

Alberto José Rojas-Salazar

Dated: March 2022

Permission to Lend and/or Copy

I, Alberto José Rojas-Salazar, agree to deposit this thesis in the University's open access institutional repository or allow the Library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement.

I, Alberto José Rojas-Salazar, consent to the examiner retaining a copy of the thesis beyond the examining period, should they so wish (EU GDPR May 2018).

Alberto José Rojas-Salazar

Dated: March 2022

Abstract

Many researchers have considered video games as effective learning tools. According to them, video games can increase intrinsic motivation and promote active learning. However, video games are a flexible medium, and they allow the integration of many learning theories and pedagogical approaches. This dissertation is concerned with exploring and evaluating the use of video games for learning based on analogies to facilitate learning conceptual knowledge, specifically binary search tree conceptual knowledge. The hypothesis of the thesis is that video games for learning based on analogies are effective tools to teach non-intuitive conceptual knowledge.

To verify our hypothesis, first, a systematic literature review concerning video games for learning that focus on data structures was conducted. Specifically, the review focused on the game elements used to convey learning, the learning aspects of the game, and the evaluation methodologies followed. Second, a review of literature concerning design of learning experiences, entertainment game design, and design for video games for learning was performed. Informed by this review, a game design framework for video games for learning was developed. Third, an action-adventure video game based on analogies that focuses on binary search trees called *DS-Hacker* was designed and developed using our framework. Fourth, an assessment tool that measures conceptual knowledge about binary search trees was developed and validated. Fifth, two evaluations were performed to verify the effectiveness of our video game as a tool for learning.

Our findings show that video games for learning can teach abstract conceptual knowledge with an efficacy that is comparable to other “traditional” digital methods, such as video lectures. However, regarding intrinsic motivation, they are more motivating than “traditional” digital methods. Regarding the pedagogical approach, analogies embedded in the game rules and game mechanics are effective and facilitate the acquisition of new information. Furthermore, the level of realism of video games can slightly affect learning gains. However, this is caused by game elements that increase the external cognitive load.

The contributions of this thesis relate to the knowledge about game-based learning based on analogies that efficiently achieve learning objectives. First, the thesis identifies characteristics, such as strengths and weaknesses, of existing video games for learning that focus on data structures through the systematic literature review. Second, the thesis presents an approach for the design and development of games for learning, which emphasizes the alignment of the context, the learning aspects, and the game design aspects. Third, the thesis presents a validated assessment tool to measure binary search tree conceptual knowledge that generates measures with scale properties that can be used in computer science education research. Finally, an original video game for learning data structures was used to validate the game design framework and to evaluate the efficacy of game-based learning based on analogies.

Acknowledgements

I would like to express my gratitude to my supervisor, Dr. Mads Haahr, for his guidance, patience, feedback, support, and creative freedom that he has given me since the beginning of my Ph.D. He witnessed my academic growth, which was a very chaotic process, and even so, he trusted in my crazy ideas.

This thesis was funded by the University of Costa Rica. Also, I would like to thank the Office of International Affairs and External Cooperation of the University of Costa Rica and the kind people who work there, particularly Vivian Madrigal and Karol Cordero, who were always concerned for my well-being and necessities.

Special thanks to Lawrence Aymerich, Pamela Chinchilla, and Douglas Soto, who were my guarantors. They believed in me and supported me from the very beginning.

Many thanks to my colleagues and friends who provided support, advice and guidance or collaborate with me. Special thanks to: Alejandra Núñez, for the enjoyable conversations and advice; Paula Ramirez, for all her help during the initial and final stages of this thesis; and Prof. David Herrera and Prof. Alexei Ochoa from the National University of Colombia, for their availability and support provided during the game evaluations performed in this thesis.

I would like to express my gratitude to Shreya Prabhu for her unconditional support in good and bad times.

Finally, I would like to thank to my family in Costa Rica for their support. Special thanks to my brother, Joel. He has made important contributions to this thesis through his feedback and advice.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

This dissertation is concerned with the use of games for learning, specifically for the learning of data structures. This first chapter motivates the research by briefly discussing the challenges in relation to learning in science with specific attention to data structures and then proceeds to define the overarching research question and specific research questions, as well as the research approach adopted to address them. Next, we state the specific contributions that this dissertation makes to the field, before we outline the overall structure of the dissertation in the form of a document roadmap.

1.1 Motivation

Many areas of scientific knowledge are difficult to learn, understand, and teach (Johnstone, 1991; Treagust et al., 2000). In general, *scientific knowledge* is intangible, systematic, and hierarchical (Howe, 1993; Johnstone, 1991). In contrast, *common knowledge*, i.e., the knowledge that people use for their common tasks, is made up of tangible and recognizable instances (Howe, 1993; Johnstone, 1991). Due to the differences between the two types of knowledge, novice students often struggle to learn and understand science. Similarly, professors may find it difficult to teach science (Johnstone, 1991).

Computer science knowledge is also packed with concepts and models that students find difficult to learn and understand (Kaczmarczyk et al., 2010; Qian & Lehman, 2017). Studies related to misconceptions and learning difficulties in computer science suggest that novice students struggle to understand logic, programming, data structures, and computer architecture conceptual knowledge (Qian & Lehman, 2017; Taylor et al., 2014). For example, in relation to programming, students have problems understanding the concept of variables (Doukakis et al., 2007; Fleury, 1991; Kaczmarczyk et al., 2010), conditionals (Sirkiä & Sorva, 2012), loops (Sleeman et al., 1986), and Object-Oriented Programming concepts and principles (Sirkiä & Sorva, 2012). Regarding data structures, studies suggest that students struggle to understand how data structures are organized (Becker & Beacham, 2000; Zingaro et al., 2018), the use of references (Kaczmarczyk et al., 2010), and their functions (Danielsiek et al., 2012; Zingaro et al., 2018). As a consequence of these issues, at the end of introductory computer science courses, students may still not be able to understand basic principles of programming or to code simple programs (McCracken et al., 2001).

Generally, learning difficulties in science can be caused by four factors: (1) the knowledge and ideas that the learner already has are inadequate; (2) the knowledge and learning task complexity exceed the learner's current capacities; (3) the communication and language are not clear to the learner; and (4) the instructional approach does not match the learning style of the learner (Kempa, 1991). In relation to computer science specifically, studies suggest that learners face similar learning difficulties. In a comprehensive literature review relating to programming misconceptions, Qian and Lehman (2017) found six causes of learning difficulties: (1) complexity of the learning activities; (2) misconceptions caused by natural language; (3) student's lack of mathematical knowledge; (4) inaccurate mental models; (5) inadequate strategies to solve abstract problems; (6) environmental factors such as student's existing knowledge; and (7) inefficient instruction approaches. These findings suggest that students' previous knowledge, communication (e.g., how learning contents are presented), task complexity, and learning strategies play an important role when learning computer science.

According to constructivist learning theories, learning and understanding a new topic requires the creation of connections to previously acquired knowledge (Aubusson et al., 2006). In this view, learning is a *relearning process* in which the learner's previous experiences, beliefs, and knowledge are modified or used as a base to create new knowledge (D. A. Kolb, 2014). Additionally, it is important that learners engage actively in the learning process (Biggs & Tang, 2007). Consequently, learning strategies must consider learners' previous knowledge and facilitate the active engagement of learners to integrate new knowledge effectively.

Analogies and *digital game-based learning* are approaches that have shown potential to increase learning gains and reduce misconceptions. Analogies support linking new information with the learner's previous knowledge through comparisons of structures from two domains; those structures must share symmetrical relations among their components (Duit, 1991). In science, analogies have been used to explain natural phenomena and to develop theory through analogical model representations (Mellor, 1968). In educational environments, analogies have been utilized to create relations between non-intuitive concepts and familiar knowledge (Aubusson et al., 2006), and they have been extensively used in science teaching (Duit, 1991).

Concerning digital game-based learning, it is a flexible, interactive, and motivating learning approach that can facilitate the representation and understanding of abstract and complex knowledge. Like more traditional mediums, video games can communicate messages through narrative elements, such as text, audio, and images. However, video games can also communicate ideas through simulations (Frasca, 2003; Squire, 2011). Simulations are operating representations based on models or abstracted representations of reality (Hays & Singer, 1989) in which a model can be considered "an approximation, representation, or idealization of selected aspects of the structured, behaviours, operations, or other characteristics of a real-world process, concept, or system" ('IEEE Standard Glossary of Modeling and Simulation

Terminology', 1989). According to Seel and Blumschein (2009), one of the main objectives of simulation is to facilitate the understanding of the interaction that occurs among the internal elements and outputs of the model represented. In video games for learning, models should be simple enough to facilitate the comprehension of the learning contents (Squire, 2011). In this regard, video games have the potential to represent complex and abstract knowledge in an approachable way using simplified models of reality. Furthermore, video games for learning allow players to engage with and perceive the models represented from different perspectives, which a purely narrative depiction cannot.

Furthermore, studies suggest that "good" video games can embody many types of learning theories and methodologies (Gee, 2007; Squire, 2011). For example, behaviourist approaches have been used through reward systems to boost players' motivation to perform repetitive tasks that otherwise would be boring (Gee, 2007). Cognitive theories, such as the multistore model (Atkinson & Shiffrin, 1968) and the cognitive load theory (Sweller, 2012), have been employed to design the game's challenges and user interfaces (Hodent, 2017). Constructivist theories have been used to encourage the active creation of knowledge through active reflection or socially through communities of practice (Gee, 2007). Also, many studies suggest that video games are intrinsically motivating (Malone, 1980; Yee, 2006), which is a desired characteristic to increase learning quality (R. M. Ryan & Deci, 2000a). Therefore, video games constitute a flexible and intrinsically motivating medium that has the potential to employ a variety of pedagogical techniques, and they can represent models of reality through simulation, an approach that other mediums cannot.

For the reasons outlined above, this dissertation is concerned with exploring and evaluating the use of digital game-based learning and analogies to facilitate learning computer science conceptual knowledge. Specifically, we investigate the use of analogies embedded in game mechanics and game model to teach Binary Search Tree (BST) conceptual knowledge. Data structures and algorithms are a fundamental area of study in computer science, and they are extensively employed in the software industry. Data structures and algorithms are important because their correct application considerably improves the efficiency of computational systems (Sedgewick & Wayne, 2014). However, BST knowledge exhibits the same learning difficulties as other complex scientific fields, and for beginning students, developing a deep understanding of advanced data structures and their algorithms is a difficult task because they are abstract and difficult to relate to familiar knowledge (Becker & Beacham, 2000). This dissertation anticipates that analogies and video games for learning can facilitate linking the BST abstract knowledge with students' familiar video game knowledge. Consequently, digital game-based learning has the potential to mitigate the learning difficulties related to students' previous knowledge and communication as pointed out by Qian and Lehman (2017). Furthermore, digital game-based learning provides an additional learning approach that can match students' learning styles.

Additionally, this dissertation investigates the effect of video game realism on learning gains of conceptual knowledge and intrinsic motivation. According to Adams (2014), game realism is a "continuous scale upon which the game's relationship to the real world is measured" (p. 501). At one end of the realism scale is *abstract*. Abstract games do not share a close similarity with the real world, and the game rules are arbitrary. At the other end of the scale is *representational*. Representational games share high similarity with the real world, and it is expected that players use their understanding of the real world to play the game. Game elements, such as the graphics and physics, can be categorized using the scale. Higher levels of realism can facilitate the creation of analogies because common and familiar knowledge can be used to create the analogical models. However, a higher representational level in video games can increase the cognitive load (Schrader & Bastiaens, 2012).

1.2 Research Methodology

This thesis follows the case study methodology. Cases studies are in-depth studies of a specific instance (e.g., a person, a family, students enrolled in a course, etc) within a real-life context. Case studies are useful when the researchers have a small sample and usually employ quantitative data collection methodologies which are useful with small samples (Lazar et al., 2017). However, cases studies may also use quantitative approaches. In those cases, the researchers should be aware of the limitations of quantitative methods with small samples (Lazar et al., 2017). Furthermore, the case study methodology relies on theoretical frameworks to guide the collection and analysis of data (Yin, 2018).

According to Yin (2018), case studies are useful when: (1) your research questions are "how" or "why" questions; (2) you have little control or no control over behavioural events; and (3) your focus of study is a contemporary phenomenon. Furthermore, case studies' main objectives can be categorized into four categories: (1) exploration; (2) explanation; (3) description; and (4) demonstration of success application of new tools (Lazar et al., 2017).

Concerning the scope of the results obtained through this methodology, case studies cannot be used to make broad conclusive claims based on their results. Nevertheless, they "can be used to build understanding, generate theories and hypothesis, present evidence for the existence of certain behaviour, or to provide insight that would otherwise be difficult to gather." (Lazar et al., 2017).

According to Lazar et al. (2017), cases studies consist of at least four components: (1) the research questions; (2) hypotheses; (3) units of analysis; and (4) a data analysis plan. The research questions are descriptions of the main aims of the study. The hypotheses are the expected results of the study. The units of analysis are descriptions of what is under study (e.g., an individual, a group of people, or individual activities). Based on components one and three, the researcher should select an appropriate data collection approach. The data analysis plan consists of the methods used to analyse the collected data. Finally, all these components should be grounded in a theoretical model.

Concerning this thesis, the case study methodology was selected because of three reasons. First, the case study methodology is useful to study instances in a real-life context, which is a desirable characteristic for this study. The aim of this thesis is to demonstrate that video games for learning computer science knowledge are effective learning tools. Generally, learning of academic knowledge occurs during classes or self-driven study. It does not occur in a laboratory or controlled environment. We consider that to verify the efficacy of a game for learning, it is necessary to test it in real-life.

Second, case studies are appropriate for small samples. Finding participants to evaluate a new learning approach may be a difficult task. Usually, beginning engineering and computer science students have a busy schedule due to the academic workload. Similarly, not many professors are willing to concede one of their classes to evaluate a new learning tool. For this reason, a methodology that allow small samples is convenient for this thesis.

Third, the case study methodology is useful to explore and provide understanding of new problems or situations. Exploration and explanations of the problems are guided by theory. In this regard, game-based learning is still a relatively new field of study that provides situations and problems that have not been explored. Similarly, new games for learning and the related game design approaches may be explored and understood before implementing them in larger scenarios. In this regard, the case study methodology is suitable for this thesis where we explore a new approach to teach scientific knowledge through video games based on analogies.

In addition to the case study methodology, this thesis uses the methodology suggested by Kitchenham and Charters (2004) for carrying out systematic literature reviews in computer science, which were later adapted by Calderón and Ruiz (2015) and Petri and von Wangenheim (2017) for reviewing serious games. According to Kitchenham and Charters, systematic literatures reviews should follow four stages: (1) the planning stage; (2) the reviewing stage; (3) the data extraction stage; and (4) the reporting stage. The use of this methodology was necessary to enhance the theoretical framework used for analysis in the study methodology.

1.3 Research Questions

This dissertation aims to answer the following overarching research question: ***Can analogies embedded into the game rules and game mechanics effectively teach computer science conceptual knowledge?***

To help answer the research question, we define five specific research questions (SRQ) that will guide the dissertation:

SRQ 1. What are the key game elements and pedagogical aspects of video games for learning data structures conceptual knowledge, and how is learning evaluated?
SRQ 2. How can game rules and game mechanics be designed to teach computer science conceptual knowledge effectively using analogies?
SRQ 3. How can conceptual knowledge learning be measured accurately?
SRQ 4. Are video games for learning that use analogies to teach conceptual knowledge more or less effective and motivating than other, more traditional, digital approaches?
SRQ 5. Does the realism level of video games for learning that use analogies affect the learning gains and motivation?

As mentioned in the previous section, cases studies are guided by theory. Therefore, before answering the specific research questions, we performed a small review of literature related to learning, motivations, play, and video games for learning (see Chapter 2). This literature review provides the foundations to guide our study (e.g., data collection, analysis, and discussion) which will lead to the answers to our research questions.

To answer the main research question, a video game for learning BST fundamental concepts was designed and developed. Two versions of the game were required (one realistic and one abstract), and the game needed to use analogies. Consequently, it was required to gather insights about game elements that facilitate conceptual knowledge and how pedagogical approaches are embedded into those game elements. Furthermore, it was necessary to obtain insights regarding how learning gains and cognitive changes were evaluated. SRQ 1 addresses these topics, and it is covered in Chapter 3 through a systematic literature review (SLR). The SLR considers studies about video games for learning data structures and algorithms and classifies them based on their subject, game elements, pedagogical approaches, and evaluation aspects. The SLR provided a starting point for considering potential research areas, game elements, and pedagogical approaches that were used to design and develop the video game for learning BST concepts. Additionally, it provided a summary of the methods utilized to evaluate cognitive changes used in previous research. This summary provided insights about the strengths and limitations of each method.

Video game design is a complex task. Many aspects, such as game elements, technological factors, pedagogical approaches, and target audience, should be considered when designing and developing a “good” video game for learning. To facilitate these tasks, it is necessary to adopt a framework or a set of design principles that guide the processes. Due to the objectives of this study, the guideline or principles must consider the use of analogies. SRQ 2 captures these concerns, and it is addressed in Chapter 4 and 5, which focus on learning, games, and design frameworks. Chapter 4 presents a review of literature regarding learning activities design, entertainment video games design, and games for learning design. The review presented in Chapter 4 was used to create a framework that intends to guide the design process of our video game for learning. The design framework is a general-purpose framework. In other words, it can be used to create games that cover different topics, using different pedagogical approaches and game genres. In Chapter 5, a description of the design and development process of a video game for learning is presented. The process is an exemplar of the use of the framework, and it constitutes a validation of the framework. The topic covered by the game is Binary Search Tree conceptual knowledge; the main pedagogical approach is analogies; and the game genre is action-adventure.

To evaluate the effectiveness of a learning tool, it is necessary to measure learning of conceptual knowledge accurately. SRQ 3 addresses this matter, and it is covered in Chapter 6. Chapter 6 provides a description of the development process of a validated concept assessment tool with scale properties (e.g., it can be used as interval data) that can be used to measure learning of conceptual BST knowledge accurately. The content validity of the assessment tool was evaluated through a qualitative survey. The statistical validity was verified through Item Response Theory statistical methods. Specifically, the Rasch model was employed.

SRQ 4 focuses on the effectiveness of our video game as a tool for learning, and the first section of Chapter 7 addresses this matter through a game evaluation. In the evaluation, the game (experimental activity) and two video tutorials (control activity) were assessed. The evaluation follows a repeated measurements design. Data collection tools used were the BST concept assessment tool described in Chapter 6 and the Intrinsic Motivation Inventory (IMI; R. M. Ryan, 1982) survey. Data collected was analysed using descriptive statistics and Mann Whitney U test.

As mentioned above, analogies use common and familiar knowledge to facilitate the creation of new knowledge. Considering this, understanding how much the level of realism of a video game affects the learning process can be beneficial to improve the design process of digital game-based learning experiences. SRQ 5 addresses this matter, and it is answered through a game evaluation in the second section of Chapter 7. In the evaluation, a 2D and 3D version of the game were assessed. This evaluation followed a repeated measure design. Data collection tools used are the BST assessment tool and the IMI survey. As the first experiment, data was analysed using descriptive statistics and Mann Whitney U test.

Table 1.1 summarizes the information given above and presents the research question ID, chapters that address the research question, data collection tools used, and the data analysis methods.

Table 1.1: Research questions, chapter addressed, data collection tools, and analysis methods.

Question	Chapter	Data collection tools	Data analysis method
SRQ 1	Chapter 3	- Systematic literature review	- Classification of game elements, learning aspects for game-based learning, and evaluation aspects.
SRQ 2	Chapter 4 Chapter 5	- Review of literature	- Synthesis of learning activities design guidelines. - Synthesis of entertainment game design frameworks. - Synthesis of game-based learning frameworks.
SRQ 3	Chapter 6	- Qualitative Survey - Test	- The Rasch model
SRQ 4	Chapter 7.1	- Informal concept inventory - Qualitative Survey - BST concept inventory - IMI	- Descriptive statistics - Mann Whitney U test
SRQ 5	Chapter 7.2	- BST concept inventory - IMI	- Descriptive statistics - Mann Whitney U test

1.4 Contributions

This dissertation evaluates the use of analogies embedded in game mechanics and the effect of realism in video games for learning computer science conceptual knowledge. Our contribute to the areas for Digital Game-Based Learning and Computer Science Education. The specific contributions are:

1. A systematic literature review related to digital video games for learning data structures and recursive algorithms.
2. A framework to design game video games with analogies embedded in their game mechanics.
3. An original adventure video game for learning binary search trees called *DS-Hacker*. The game has two versions, a 2D and 3D version.
4. A binary search tree concept assessment tool with scale properties.

Some parts of this thesis were published in the following peer reviewed conferences:

- The initial concept of *DS-Hacker* was presented in Digital Game Research Association Conference (DiGRA 2020); available in (Rojas-Salazar & Haahr, 2020a).
- The pilot evaluation of *DS-Hacker 3D* (Appendix 3) was presented in International Computer Programming Education Conference (ICPEC 2020); available in (Rojas-Salazar et al., 2020).
- The second version of *DS-Hacker 3D* (Chapter 5) was presented in The International Conference on the Foundations of Digital Games (FDG 2020); available in (Rojas-Salazar & Haahr, 2020b).
- The systematic literature review related to video game for learning data structures and recursive algorithms was presented (Chapter 3) Joint Conference of Serious Games (JCSG 2020); available in (Rojas-Salazar & Haahr, 2020c).

1.5 Document Roadmap

The remainder of the dissertation is structured as follows:

Chapter 2 presents a review of literature related to learning and digital game-based learning. The chapter is divided in two sections. The first section addresses learning topics from a general stand. In this section, the learning paradigms, theories, and pedagogical approaches are presented. The second section provides a review of literature concerning play, games, and game-based learning definitions and concepts. The aim of this chapter is to present a learning and pedagogical framework for the construction of the learning tool and the analysis of the results obtained in the evaluations.

Chapter 3 provides a systematic literature review of video games for learning data structures and recursive algorithms. The review considers all video games reported in academic journals between 1999 to 2019, and it focuses on three areas: (1) game aspects, (2) pedagogical approaches, and (3) evaluation of the game.

Chapter 4 provides a review of the literature concerning design principles and game design frameworks, and at the end of the chapter, a synthesis of those design frameworks and Chapter 3’s learning theories is presented. Chapter 4 is divided into four sections. The first section provides a review of literature about design principles for higher education learning activities. The second section addresses methodologies of game design and key elements of game elements that should be considered during the game design phase. The third section presents a review of design frameworks for video games for learning. The fourth section provides a discussion that integrates all sections into a comprehensive game design framework.

Chapter 5 presents a description of the design and development process of a video game for leaning BST conceptual knowledge. This chapter is divided in four sections. The first section explains the design and development process of the video game. The second section describes the learning content of the game. The third section provides a detailed description of the elements of the game and how they teach the learning contents. The fourth section provides a discussion of the expected outcomes and limitations of the game.

Chapter 6 provides a description of the design, development, and calibration of a BST concept assessment tool with scale properties that measures learners' conceptual knowledge. This section provides an explanation of how the inventory items were constructed and validated. Then, it provides an explanation of the methodology and materials used during the statistical validation of the assessment tool. Finally, it presents the results obtained and limitations of the assessment tool.

Chapter 7 presents two game evaluations. The first evaluation assesses the effectiveness of the game's final version and compares it against two video tutorials. The third experiment evaluates the 2D and 3D versions of game, and its aim is to verify the effect of realism on learning gains.

Chapter 8 concludes the thesis and presents a summary of the contributions, how the objectives are met, and how the research questions are answered. Additionally, the chapter presents a discussion related to future work and limitations of the dissertation results.

Chapter 2: Learning and Video Games

Chapter 2 presents a review of literature related to learning theories, pedagogical approaches, and concepts of play, games, and game-based learning. The review of game concepts is used in Chapter 4 to answer SRQ 2: *How can game mechanics and game models be designed to teach computer science conceptual knowledge effectively using analogies?* The review of learning concepts is used in Chapter 5 to make game design decisions and in Chapter 7 to discuss and analyse the results obtained during the game evaluations.

2.1 Learning

In this section, the learning theories and pedagogical methodologies that underpin our game are explained. First, we give a brief explanation of the main perspectives on learning. Then, Kolb's experiential learning theory is presented, which is followed by a discussion about the concept of analogies and its importance in learning. Then, a definition and classification of the different types of motivations are presented. Finally, the cognitive load theory is explained.

2.1.1 Behaviorism, Cognitivism and Constructivism

Behaviourism is a set of psychology approaches to understanding behaviour. It appeared at the beginning of the 20th century as a rejection of the introspective methods, stating that behaviours (including learning) of humans are caused as a response to environmental stimuli (Phillips, 2012). In other words, behaviours follow a stimulus-response principle (Bates, 2016). Learning theories under this paradigm explain learning without any description of the mental processes that occur in the individuals (Harteveld, 2011). Advocates of this paradigm suggest that people need to be guided through their learning processes and encourage the usage of reinforcement and repetition. Furthermore, desired behaviours are strengthened through rewards, and in similar way, undesired behaviours are be discouraged through punishment or lack of rewards (Bates, 2016; Phillips, 2012).

Cognitivism is a learning theory that became popular at the end of the 1950s (Whitton, 2007), and it emerged as a rejection of the behaviourist theories (Harteveld, 2011). Advocates of this theory state that internal mental operations cannot be ignored. Cognitivism rejected the idea that the mind is a black box and started to pay attention to internal processes that occur in people while they are learning. To explain learning, cognitive theories focus on sensory receptors, executive control, working memory and long-term memory. Furthermore, cognitivism suggests that new information is actively processed, and learning occurs while people search for relations with familiar knowledge (Harteveld, 2011).

Constructivism is a theory of knowledge that suggests that humans actively create and re-create their knowledge using their previous experiences, mental structures, and beliefs (Gogus, 2012). In education environments, this means that educators cannot simply transfer the information to students. Learners will not understand information explained by an educator if they do not reflect about what is being taught or if they cannot relate new information with what they already know (Gogus, 2012). Learners must interpret and organize their experiences in order to create meaning and transform the experiences into knowledge. Under this premise, educators should provide experiences and environments where students can construct and reconstruct their knowledge through reflection, critical thinking and knowledge already acquired.

Constructivist theories have evolved in different forms. Two of the most influential trends of constructivism are cognitive constructivism and social constructivism. Cognitive constructivism was initially developed by Jean Piaget and focusses on how learners actively modify and expand their knowledge relying on their cognitive structures (Gogus, 2012). On the other hand, social constructivism was initiated by Lev Vygotsky and suggests that knowledge is constructed and reconstructed through social processes. Furthermore, social constructivism suggests that culture and context play a key role for cognitive development (Gogus, 2012).

2.1.2 Kolb’s Experiential Learning Theory

Kolb’s experiential learning theory (KELT) is a constructivist learning theory that is based on several well-known theories proposed by scholars such as William James, Kurt Lewin, Jean Piaget, and Paulo Freire. KELT suggests that experience plays a fundamental role in every learning process, and that learners should be in the centre of the learning system. The theory also suggests that learning occurs through the resolution of two dual dialectics: action/reflection and experience/abstraction. According to Kolb and Kolb (2012), KELT is aligned with six principles:

1. Learning is a process. Ideas and knowledge are not fixed. Learning and understanding are always evolving and restructuring.
2. Learning is re-learning. Learners use their previous experiences, beliefs, and knowledge to create new knowledge. KELT is guided by the constructivist principle of learning.
3. “Learning requires the resolution of conflicts between dialectically opposed modes of adaptation to the world” (p. 1216). The modes of adaptation are action/reflection and experience/abstraction.
4. Learning is a holistic process. Learning is more than just cognitive processes. Learning requires involvement of learners’ bodies and minds, such as senses, emotions, beliefs,

thoughts, and behaviours. Furthermore, learning is not limited to one stage of life. People learn during their whole life.

5. Learning is the result of transactions between the learner and the environment. Experience (objective) is caused by the environment, but it is perceived and understood by the learner (subjective).
6. Learning is a process of creating knowledge. People build their personal knowledge by constructing and reconstructing social knowledge. At the same time, social knowledge is built from personal knowledge. Therefore, knowledge is constructed by the interaction between social and personal knowledge.

According to KELT, learning occurs as a cyclical or spiral process where learners transform their experience into knowledge (Figure 2.1). The learning cycle consists of four interrelated stages, and ideally, learners should pass over all the stages in order to maximise the learning experience (D. A. Kolb, 2014). Simply put, the learning cycle works like this: A learner experiences a concrete or immediate experience. Then, he or she observes and reflects about the experience. Then, he or she integrates his or her reflections with previous knowledge and believes and transforms those reflections into abstract concepts and theories. Finally, these abstracts concepts and theories can be used or tested in new applications or scenarios that will generate new experiences.

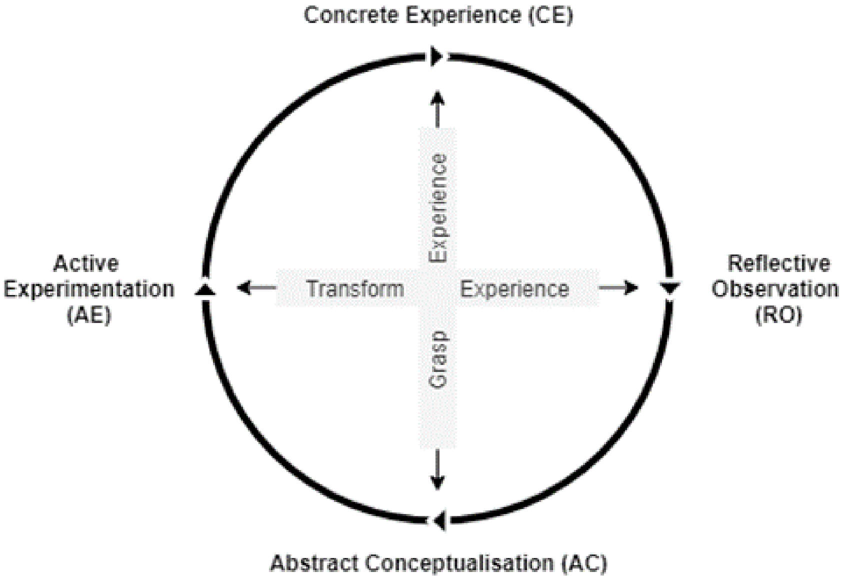

Concrete Experience (CE)
Experience
Active Experimentation (AE)
Transform
Experience
Reflective Observation (RO)
Grasp
Abstract Conceptualisation (AC)

Figure 2.1: Experiential learning cycle; adapted from D. A. Kolb (2014).

The learning cycle distinguishes two dimensions (axes) of learning: the prehension and the transformation (D. A. Kolb, 2014). The prehension dimension corresponds to two opposed ways of grasping experiences: comprehension (grasping from abstract experiences) and apprehension (grasping from concrete experiences). The transformation dimension corresponds to the two opposed ways of transforming the grasped experiences: intention (transforming through internal reflection) and extension (transforming through active external manipulation).

Furthermore, the cycle has four stages that correspond to one of the sides of the two dimensions (axes) mentioned before. These stages are:

- *Concrete Experience* (CE). Learning occurs through senses and feelings; learners should engage sincerely and openly to the new experience.
- *Reflective Observation* (RO). Learners understand their experiences through observation and reflection.
- *Abstract Conceptualization* (AC). Learners conscientiously create new concepts and theories by thinking and relying on their observations and previous knowledge and experiences.
- *Active Experimentation* (AE). People learn through active experimentation and testing their theories and concepts.

To summarize, KELT defines learning as "the process whereby knowledge is created through the transformation of experience" (D. A. Kolb, 2014, p. 49).

Concerning video games and KELT, Whitton (Whitton, 2014) suggests that video games embody several aspects of this theory. For example, in video games, learning occurs as a cycle of probing. First, players experience some aspects of games; then, they reflect about their experience and formulates theories and hypothesis from those experiences; and finally, they probe the new ideas generating new experiences (Gee, 2007). The learning game cycle has a similar structure than Kolb's learning cycle.

Another aspect of KELT reflected in video games is that video games allow learning of procedural knowledge (Gee, 2007). Unlike other types of learning experiences that only provide conceptual and factual knowledge, learning with video games can occur through repetition of concrete experiences and active experimentation. Finally, video games provide immediate feedback, which is necessary for the KELT learning cycle.

2.1.3 Motivation

Motivation is a "psychological force that enables action" (Touré-Tillery & Fishbach, 2014, p. 328). In other words, motivation can be understood as how much effort and how often a person will act or seek a goal (Svinicki & Vogler, 2012). According to Ryan and Deci (2000b, 2000a), it is possible to recognise two types of motivation: intrinsic and extrinsic motivation.

Intrinsic motivation refers to the psychological forces that induce a person to perform certain activities just because they are interesting or enjoyable (R. M. Ryan & Deci, 2000a). However, whether an activity is intrinsically motivating depends on the individual who is doing the activity. For example, some people find it interesting to play video games; however other people do not.

Two factors can produce variability in intrinsic motivation: the *feeling of competence* and the *sense of autonomy* (Deci & Ryan, 1985a). The feeling of competence occurs when an individual performs an activity efficiently. The sense of autonomy occurs when a person freely chooses to do an activity. Consequently, promoting the feeling of competence (e.g., provide optimal challenges and constructive feedback) and autonomy (e.g., opportunities, choice, and self-direction) increases the intrinsic motivation. On the other hand, diminishing the feeling of competence (e.g., demeaning evaluations and negative feedback) and autonomy (e.g., deadlines, threats, and competition pressure) decrease the intrinsic motivation.

According to Ryan and Deci (R. M. Ryan & Deci, 2000a), there are many ways to measure intrinsic motivation; however there are two methods that have been used most often: “free choice” measure and self-reports. The “free choice” approach is a laboratory experiment design where participants are exposed to a target task and distractor activities. Then, the experimenter observes which activities the participant prefers. Self-reports can be surveys or interviews that asks about the interest or enjoyment of a task.

Extrinsic motivation refers to the psychological force that induces a person to perform certain activities because they will lead to an outcome that is not directly related to the actions performed (R. M. Ryan & Deci, 2000a). Because many necessary activities are not intrinsically motivating, two aspects are fundamental to increasing motivation without external pressure: *internalization* and *integration*. Internalization “is the process of taking in a value or regulation” (p. 60). Integration “is the process by which individuals more fully transform the regulation into their own so that it will emanate from their sense of self” (p. 60). Furthermore, according to Deci and Ryan (Deci & Ryan, 1985b), extrinsic motivation can be classified into four categories depending on the level of autonomy.

- *External regulation*. It occurs when an individual performs an activity to obtain an imposed reward, avoid a punishment, or satisfy an external demand.

- *Introjection*. It occurs when internal pressures, such as guilt, anxiety, pride, or ego, regulate (motivate) a person to perform an activity.
- *Identification*. It occurs when an individual understands the personal importance of a behaviour or activity and accepts its regulation as to his or her own.
- *Integration*. It occurs when after self-examination and congruence with his or her values and needs, a person fully accepts an identified regulation.

2.1.4 Analogies and Learning

Analogies are comparisons of parts of the structures of two domains (Duit, 1991). Their objective is to transfer ideas and concepts from one familiar domain (called *source* or *base*) to another unfamiliar domain (called *target*) (Glynn, 1994). Figure 2.2 is a formal representation of an analogy. R_1 represents the source, and R_2 represents the *target*. R_m represents the structures of R_1 and R_2 that share similarities. Due to this relation, it is possible to state that there is an analogical relation between the two domains (Duit, 1991). Furthermore, those structures must share symmetrical relations among some of their components.

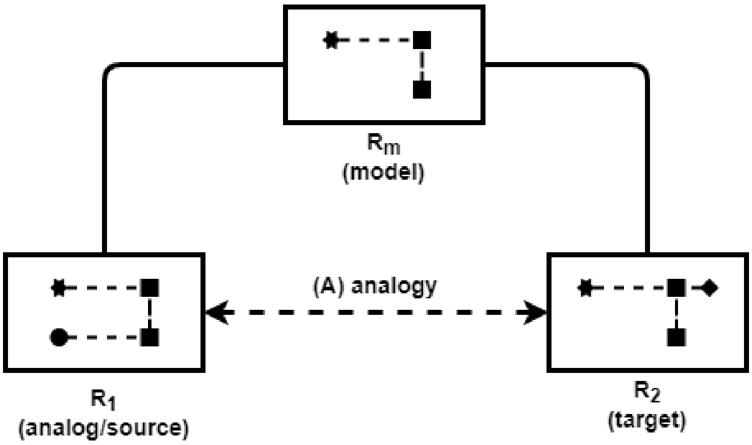


Figure 2.2: Formal representation of an analogy; adapted from Duit (1991).

Analogies have proved to be an efficient teaching method, and for this reason, they have been extensively used in educational environments to facilitate acquisition of non-intuitive and abstract knowledge (Podolefsky & Finkelstein, 2006). From a constructivist perspective, this may be caused by two fundamental principles: (1) knowledge is actively built by learners, and (2) new knowledge is grounded on previously acquired knowledge. Accordingly, analogies are aligned with these principles. Analogies facilitate the acquisition of knowledge through an active process where learners must actively create relations between familiar knowledge and unfamiliar information (Duit, 1991). Consequently, analogies are quite popular in science education (Glynn, 1994), for example physics (Podolefsky & Finkelstein, 2006), astronomy

(Donnelly & McDaniel, 1993), mathematics (Indurkhya, 1991) and, even in other fields such as sports (Lee et al., 2019).

In order to successfully utilize analogies in educational environments, Glynn (Glynn, 1994) carried out a study that included theoretical considerations, analysis of physics books and a set of empirical studies. From his results, he identified six operations that may facilitate the application of analogies. The operations are:

1. Introduce target concept.
2. Cue retrieval of analog concept.
3. Identify relevant features of target and analog.
4. Map similarities between target and analog.
5. Indicate where analogy breaks down.
6. Draw conclusions.

Despite the analogies' strengths, analogies may possess potential risks. One threat is that the base domain never exactly fits the target domain. This situation may lead learners to make incorrect deductions by applying aspects of the familiar domain that do not correspond to the unfamiliar domain (Donnelly & McDaniel, 1993; Duit, 1991). A second threat is that analogical reasoning requires considerable guidance (Duit, 1991). To guarantee students' understanding, learning tools and professors must ensure that learners understand surface similarities and deep structure aspects of both domains.

2.1.5 Cognitive Load Theory

The cognitive load theory is a learning theory that states that human cognitive processing power is limited and suggests that instruction and learning activities should be presented in a way that maximize the usage of the cognitive power available (Sweller, 2012). The theory is based on the cognitive architectural multistore model (Atkinson & Shiffrin, 1968; Baddeley & Hitch, 1974). Specifically, it focuses on the long-term memory and working memory.

The cognitive load theory suggests that there are two types of knowledge: *biologically primary knowledge* and *biologically secondary knowledge* (Sweller, 2011). Biologically primary knowledge is the knowledge that humans learn naturally, unconsciously, and without explicit instruction; humans evolved to learn these skills. Examples of these skills are speaking the mother language, listening, psychomotor skills, recognizing faces and patterns. Biologically primary knowledge is modular and not closely related to other primary skills.

On the other hand, secondary knowledge is knowledge that is culturally important (Sweller, 2011). Acquiring secondary knowledge tends to be effortful, conscious, and enhanced by explicit instruction; humans need motivation to learn this type of knowledge. Examples of this type of knowledge are reading, writing, or doing advanced mathematical calculations. Biologically secondary knowledge requires a general cognitive architecture applicable to a wide variety of areas. The cognitive load theory applies to this type of knowledge.

According to the cognitive load theory, the human cognitive system deals with biologically secondary information in the same way that evolution by natural selection deals with living things. The cognitive system creates novel information, stores it, and propagates it (Sweller, 2011); however, novel knowledge suffers a process of selection, evolution, or elimination, depending on whether it fits the context. The cognitive load theory suggests five principles that describe this process (Sweller, 2011).

- *Information store principle*. This principle suggest that long-term memory plays a major role. Expertise on a domain depends on how much biologically secondary knowledge about that domain one has stored in our long-term memory. In this regard, the role of education is to maximize the amount of knowledge stored in the long-term memory.
- *Borrowing and reorganization principle*. This principle suggests that the most efficient way to acquire new knowledge is borrowing (and reorganizing) knowledge from long-term memory of other people. This process occurs by imitating what they do, listening to what they say, and reading what they write. Information borrowed through this process can be altered, depending on what is previously stored in the long-term memory.
- *Randomness as genesis principle*. This principle suggests that new knowledge is created during problem solving situations through a random generate and test mechanism. This mechanism is limited by the knowledge stored in long-term memory, which reduces the range of possible moves or variations. In other words, this principle suggests that new knowledge is ultimately created by trial-and-error processes.
- *Narrow limits of change principle*. This principle suggests that the reduced capacity of the working memory when dealing with novel information helps to reduce the range of possible moves and the complexity of these movements. This suggests that learning occurs gradually and in small portions. Additionally, it suggests that information that it is not fully processed in the working memory will not be stored in long-term memory.
- *Environmental organizing and linking principle*. This principle states that when processing new information, working memory uses cues from the environment to organize and determine how and when to use that information. If the new information was properly

stored and linked in long-term memory, the limitations of working memory when dealing with novel information disappear, allowing it to manage a large amount of information.

According to the cognitive load theory, the purpose of instruction is to increase biological secondary knowledge stored and organized in long-term memory, considering all the limitations mentioned above and avoiding overloading the cognitive system, specifically the working memory.

Furthermore, the theory suggests that there are two types of cognitive loads: intrinsic, germane, and extraneous (Sweller, 2012). Intrinsic cognitive load is caused by the natural difficulty of the topics covered. Germane cognitive load refers to the resources used by the working memory to process the new information into advanced and more complex schemas. Extraneous cognitive load is caused by extraneous information or the way that a topic is presented. According to the theory, this is the only type of cognitive load that can be reduced. Research in cognitive load has found a set of effects that can be used to reduced extraneous cognitive load (Sweller, 2011, 2012).

- *The goal-free effect*. The effect occurs when learners are presented with problems that do not have a specific goal and ask to solve as much problems as the learner can. This later type of problems maximizes learning gains.
- *The worked example effect*. According to this effect, students learn more by studying a problem and its solution rather than solving the problem themselves.
- *The split-attention effect*. It occurs when information is presented in multiple physically separate forms, and to understand the topic, students must split their attention and dedicate part of it to each form.
- *The modality effect*. This effect occurs when information is better presented by a dual mode of presentation (e.g., audio-visual) than a single (e.g., just written text).
- *The redundancy effect*. It occurs when information is presented by multiple redundant and unnecessary sources, causing a cognitive load that reduce learning gains.
- *The element interactivity effect*. When the intrinsic cognitive load of a topic is low, it is possible to increase the extraneous cognitive load without having instructional consequences.
- *The expertise reversal effect*. This effect occurs when instructional methods for novice learners decrease in effectiveness as learners become more proficient.
- *Problem completion effect*. Students have a better performance when presented with a partially completed exercise than when they must solve the whole exercise.

- *Guidance fading effect*. When learners increase their capabilities, guidance should gradually disappear. For example, the problem competition effect only works for novice students but not for advanced students.
- *The imagination effect*. Students who imagine concepts or procedures obtain better learning gains than students who just study the concepts or procedures.

2.2 Play, Games, and Game-Based Learning

This section provides a review of literature related to definitions and concepts of play, games, and game-based learning. The aim of the section is to understand how play and games can be used to teach contents. To achieve this objective, first, the concept of play is defined, and its relationship with learning is presented. Then, definitions of games, video games, and their fundamental characteristics are provided. Finally, definitions and features of game-based learning are presented. The definition of game-based learning gives us a reference for which characteristics a video game for learning should possess.

2.2.1 Play

Play is intrinsic to human nature, and it is fundamental for human development (Rieber, 1996). According to the Dutch anthropologist Johan Huizinga (1955, p. 28),

> play is a voluntary activity or occupation executed within certain fixed limits of time and place, according to rules freely accepted but absolutely binding, having its aim in itself and accompanied by a feeling of tension, joy and the consciousness that it is 'different' from 'ordinary life'.

This definition highlights that play is a fun activity, and it is done with joy. It also highlights the importance of rules. Play requires rules that cannot be broken by the players. Trespassing the rules can lead to an early conclusion of the activity. Furthermore, playing requires a certain amount of imagination; the players should pretend that they are doing something different from what they are actually doing. Pretending is the cognitive skill that allows us to consciously create, modify, and interact with an imaginary reality (Adams, 2014). The boundary that divides what is real from what is played has been described as a "magic circle", inside which actions have a different meaning from what they mean in real life (Huizinga, 1955).

Similarly, Rieber (1996) suggests that play has four fundamental characteristics: (1) it is usually voluntary; (2) it is intrinsically motivating; (3) it requires actions from the players; and (4) it has a make believe quality. However, Rieber mentions that playing is not always an enjoyable activity. An individual can be forced to play by external pressure (e.g., peers or culture), and bulling behaviours in relation to play can reduce the feeling of joy with which play is otherwise associated.

Roger Caillois (1961) suggests that play can be divided into two types: *paidia* and *ludus*. *Paidia* is all types of free play activities with no pre-established rules, for example, a child playing with his and her dolls or children running around, chasing each other. On the other hand, *ludus* refers to all play activities that happen within a game that has pre-established rules, for example, children playing basketball, Monopoly, or Tetris. Consequently, the term play can be applied to games and other types of spontaneous activities that cannot be classified as games.

Salen and Zimmerman (2004) distinguish between three types of play: *Game Play*, *Ludic Activities*, and *Being Playful*. *Game Play* refers to all types of play that take place within a formal set of rules defined by a game. *Ludic Activities* denotes the type of playing that occurs in a non-game setting. *Being Playful* occurs when a person has a playful state of mind and interacts with bigger and more general systems of rules like language or physics. For example, rhymes, some types of jokes, or avoiding stepping on cracks in the pavement can be considered as *Being Playful*. According Salen and Zimmerman, all these types of play take place within a more rigid system of rules, and the emotional and engaging experiences happen when players can move freely inside the system of rules.

Concerning learning, research suggests that humans learn intuitively through play (Bueno, 2018; Rieber, 1996). Play is a meaningful activity that requires a mental effort (Huizinga, 1955) and intense concentration, which allows appreciating the abstract and creative capabilities of the human mind (Matthews & Hua Liu, 2012). According to Matthews and Hua Liu (2012), play in children is associated with learning outcomes in the following areas: physical, social-emotional, linguistic, and cognitive. In young adults, play is associated with creativity and imagination (Matthews & Hua Liu, 2012).

Table 2.1 summarizes the six characteristics of play that were found in this review.

Table 2.1: Characteristics of play.

Characteristic	Definition
Pretending	Playing requires that the players pretend that they are doing something different from what they are actually doing. This boundary is known as the “magic circle”.
Rule based	While playing, there are implicit or explicit rules that guide the course of action. These rules cannot be broken without a consensus.
Intrinsically motivating	Playing produces joy to the player, and the player do not expect external rewards to engage it.
Voluntarily	Usually, playing is a voluntary activity.
Active engagement	Playing requires that the player performs activities.
Learning	Playing is an intuitive way of learning and developing skills.

2.2.2 Games

According to Crawford (1984, p. 9), games are “closed formal systems that subjectively presents a subset of reality.” As formal systems, all games have explicit rules, which define the interaction among all their components. Games do not objectively represent reality; accuracy is only used if it helps to enhance players’ fantasy. Games allow interaction, and they permit the observation the causes and consequences of the interactions. Games possess conflicts in the form of goals and challenges. Additionally, Crawford adds that games provide a safe environment to practice without serious consequences.

Salen and Zimmerman (2004) define games as artificial activities that allow one or more players to play. Unlike spontaneous ludic activities, a game is a designed activity that creates an artificial context for play. Inside this context, objects and actions can have different meanings, behaviours, and relationships of what they really mean, behave, and relate in real life. The rules of the game define the meaning and outcomes inside the artificial context. Consequently, games are systems based on rules (Juul, 2005; Salen & Zimmerman, 2004), and, as systems, games are made of smaller elements that interact with each other in order to create something greater than the individual elements (Crawford, 1984).

From a systemic point of view, Salen and Zimmerman (2004) suggest that games are composed of four elements: objects, attributes, internal relationships, and an environment. Objects refer to the elements and variables inside a game. For example, a token of a board game like *Risk* (1959) is an object. Attributes are the characteristics of a game object such as meaning, behaviours, and outcomes. For example, a *Risk* token (object) represents an army (attribute). Internal relationships arise from the objects’ interaction when the game is played. For example, an army of more than two tokens attacking a region defended by another army is a interaction in *Risk*. Finally, the environment refers to the context of the game. It is when the game is being played that the whole system materializes. For example, a group of friends playing *Risk*.

Adams (2014) suggests that games are created to fulfil the human necessity for play and our capacity to pretend. According to him, “a game is a type of play activity, conducted in the context of a pretended reality, in which the participant(s) try to achieve at least one arbitrary, nontrivial goal by acting in accordance with the rules” (p. 3). For Dempsey et al. (2002), a game is an activity involving one or more players and some aspects of competition. This activity has goals, constraints, payoffs, and consequences, and rules guide the interactions and outcomes of the game.

Schell (2014) distinguishes ten features that games have: (1) games are voluntarily; (2) they have goals; (3) they have a conflict; (4) they have rules; (5) they can be won or lost; (6) they are interactive; (7) they have challenges; (8) they have internal value; (9) they are engaging; and (10) they are closed formal systems.

According to Juul (2005, p. 36), a game is

> is a rule-based system with a variable and quantifiable outcome, where different outcomes are assigned different values, the player exerts effort in order to influence the outcome, the player feels emotionally attached to the outcome, and the consequences of the activity are negotiable.

This definition suggests that games have six characteristics (Juul, 2005): (1) they have rules; (2) the outcome of the game can vary, and it is quantifiable; (3) the outcomes of a game have different values and meanings, some positive and other negative; (4) the player should do certain effort in order to achieve the game goals; (5) the outcome has an emotional meaning for the player; and (6) the consequences can extend to real-life if the players agree to do it.

Concerning the rules, they are one of the main features of games and distinguish games from other types of activities that allow playing. In games, rules are the means for creating play (Salen & Zimmerman, 2004); they are the blueprints that enable a specific type of play. Rules state what can the players do inside the game and what are the outcomes of each action (Salen & Zimmerman, 2004). Rules must be precise and repeatable (Juul, 2005). Adams (2014) suggests that games' rules define the following aspects:

1. The semiotics of the game (meanings and relationships among game elements).
2. The gameplay (challenges and available actions).
3. The sequence of play.
4. The goals of the game.
5. The termination condition (winning and losing condition).
6. Metarules (exceptions).

Results of this review shows that game definitions share many aspects in common. Table 2.2 summarizes the characteristics that are shared in the definition presented and that we consider important.

Table 2.2: Key components of games.

Characteristic	Description
System	Games are formal systems that consist of smaller components that interact among each other.
Rules	Rules dictate the meaning, interaction, and outcomes of the game components and players.
Fantasy	Games require a make-believe environment or story.
Interaction	Games allow interaction among their elements. Rules state the relationships among players and game elements. These relationships allow the game elements and players to perform actions that can modify the state of the game.
Outcomes	They are the result of the interactions among game elements and players. They are stated by the rules.
Goals	Games have one or more goals that players must achieve.
Challenges	Challenges are the obstacles that the player must overcome to achieve the goals. Challenges must not be trivial.

2.2.3 Video Games

Video games are a subset of games mediated by computers (Adams, 2014). Computers include desktop and laptop computers, game consoles, cellphones, tablets, VR headsets, etc. In video games, computers store and compute the rules, and they determine the winning or losing condition. Due to computers' capabilities (e.g., store large amount of data and process information extremely fast), they can handle complex systems with large sets of rules and still provide immediate feedback (Salen & Zimmerman, 2004). Also, computers allow video games to use entertainment techniques from other media (Adams, 2014). In other words, video games can be much more complex than traditional games.

Usually, video games do not require that players enforce the rules (Juul, 2005), and in many situations, players are required to discover and learn the rules while playing. This allows the player to deeply immersed in the game while the computer hides and revels information in interesting ways (Adams, 2014; Salen & Zimmerman, 2004).

The following sections provide a review of key features that are important for this thesis.

2.2.3.1 Game Mechanics

The game mechanic is a core concept of video games. However, it has been defined in different ways. On one hand, some scholars define game mechanics as equivalent to the rules of the video game. However, other scholars define them as the rules that allow the player to interact with the game system. This section provides a brief review of definitions related to game mechanics.

Concerning game mechanics as rules, many game scholars, such as Adams (2014), Hunicke et al. (2004), and Schell (2014), define game mechanics as a detailed and precise version of the rules. In other words, mechanics are “the data and the algorithms that precisely define the game’s rules and internal operations” (Adams, 2014, p. 351). The game mechanics state what the game environment and game elements’ relationships are and how they behave. Additionally, the game mechanics state what actions can be performed by the player and how he or she can use those actions.

Regarding game mechanics as means of interaction, Salen and Zimmerman (2004) define core mechanics as the activity or set of activities that the player should perform in a game. They are the means that allow the player to perform meaningful choices. According to Fernández-Vara (2014), game mechanics are “the rules of the game that refer to what the player can do, as opposed to the parts of the game system that act independently of player input” (p. 250). Sicart (Sicart, 2008) suggests a more formal definition, borrowing terminology of Object-Oriented Programming. He defines game mechanics as “methods invoked by agents, designed for interaction with the game state.” According to Sicart, methods are the actions that the agents can perform. The actions are the means of interaction with the game world and are constrained by the rules. Furthermore, game mechanics can be invoked any agent in the game; these includes the player and non-player characters. For the thesis purposes, we understand game mechanics as defined by Sicart.

2.2.3.2 Game World

The game world (fictional world, or game environment) is the artificial place where the playing interactions occur (Adams, 2014). It includes the physical space, the characters that live there, and the events that occur inside the game world (Fernández-Vara, 2014). The game world is the place in which the player pretends to be. It is the internal zone of the magic circle. The purpose of the game world is to increase immersion and engagement. However, as the player increases his or her understanding of the game mechanics and rules, his or her interest in the game world decreases (Adams, 2014). Furthermore, Adams suggest five dimensions that help to describe a game world. These dimensions are summarized in Table 2.3.

2.2.3.3 Level of Realism

According to Adams (2014), all videogames require a certain degree of abstraction, even the most realistic ones. The degree of realism in videogames can be understood as a continuous scale. On one end of the scale is *abstract*. A video game is considered abstract when its elements share little relationship to the real world. On the other side of the scale is *representational*. Representational games depict ideas and relationships familiar from the real world. Players are expected to apply their understanding of the real world while playing the game. Furthermore, "different aspects of the game may have their own levels of realism (such as the graphics and the physics), which combine to form the game's overall level of realism" (p. 520).

Table 2.3: Dimensions of the game world (Adams, 2014).

Dimension	**Description**
Physical	It consists of the simulated spaces where the player's avatar can move, and it has three properties: 1. *Spatial dimensionality*. It refers to the number of dimensions that the game world possesses. The most common are 2D, 2.5D, and 3D. 2. *Scale*. This property considers the *absolute* size of the physical space represented and the *relative* size of objects in the game. 3. *Boundaries*. It refers to the edges of the physical space simulated.
Temporal	This dimension defines how the game world treats time and the ways that it differs from time in real world. The time dimension has three properties: 1. *Variable time*. It refers to the scale of time in relation to real world time (e.g., daytimes of just some minutes). Additionally, this property considers time jumps that occurs when nothing happens in the game. 2. *Anomalous time*. This property occurs when time seems to move at a different speed at different parts of the game. 3. *Adjustable time*. This property allows the player to adjust the time of the game world (e.g., speed up the time or decrease the time of a match).
Environmental	This dimension describes the game world's appearance and atmosphere, and it has two properties: 1. *Cultural context*. It describes the culture of the individuals, institutions, social organizations of the game world. This includes aspects such as beliefs, attitudes, and values. 2. *Physical surroundings*. It addresses aspects related to the look and feel of the game world. It includes the level of detail and the style.
Emotional	This dimension defines the emotions of the people who live in the game world and the intended emotions that the player should feel while he or she interacts with the game world.
Ethical	The ethical dimension defines the ethical and moral values of the game world. In other words, it defines what is good or wrong and how the player should behave inside the game world.

Another approach to understanding realism in video games is proposed by Fernández-Vara (2011). She suggests the use of an analytical framework to understand the level of abstraction in videogames based on Juul's (2005) concepts of fictional worlds, rules, and simulation. Fictional worlds provide "a frame of reference to the player's actions and become the setting of the story of the game" (Fernández-Vara, 2011, pp. 131–132). The rules govern the video game's behaviours and player actions. The simulation is the representation of the fictional world by means of the rules. However, not all the fictional world elements are simulated. Some of them are represented by cut scenes, informative text, or the visual environment in the video game. Additionally, there are some common game rules that are not part of the fictional world, such as saving, loading, and score systems.

According to Fernández-Vara (2011), the intersection between fictional worlds and rules defines the videogame's level of abstraction: a larger overlap results in a more detailed simulation, meanwhile a smaller overlap results in more abstract interactions (Figure 2.3).

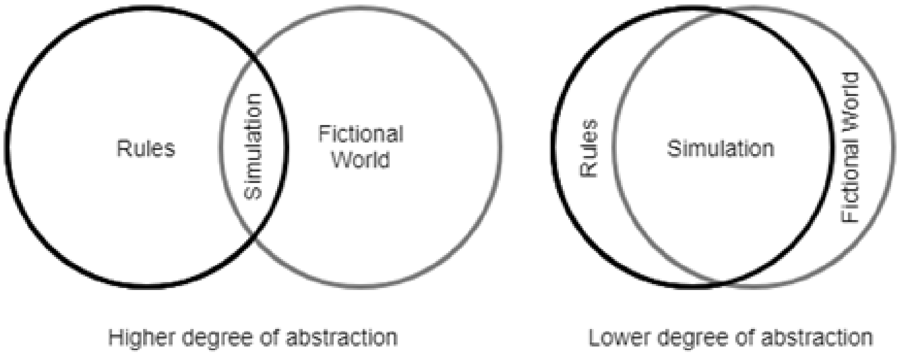


Figure 2.3: Levels of abstraction in video games; adapted from Fernández-Vara (2011).

2.2.4 Game-Based Learning

The idea of using games for learning purposes is not new. Many instructors and, more recently, scholars have hypothesized that games as learning tools have the potential to increase motivation, improve practical reasoning skills and complex problem solving (Dempsey et al., 2002). For this reason, games have been used in education for many years and in many areas. For example, they have been used in primary schools, military training, business and management environments, and to train medical students (Dempsey et al., 2002).

Formally, game-based learning (GBL) is the use of games to facilitate learning (Whitton, 2012), and in formal education, GBL should facilitate the accomplishing of the learning objectives stated by the course programme. Furthermore, GBL does not require a specific type of game, platform, or genre to facilitate learning. Learning can be achieved through commercial off-the-shelf (COTS) games or serious games. Games can also be analogue as well as digital games. When learning is facilitated by digital games, it is known as digital game-based learning (DGBL).

COTS games are games that were designed primarily for entertainment but may have potential for educational purposes (Becker, 2017). When COTS games are used for learning, it is often because they relieve the teachers and instructors of the need to become game designers and programmers. Additionally, COTS games can exhibit characteristics that are desirable for educational environments, such as promoting critical reasoning and problem solving, and different ways of processing information (van Eck, 2009). However, due to their original (entertainment) purpose, the efficacy of a COTS game as a learning tool depends on how well the game fits the learning objectives and how well the learning activities and assessment of knowledge are designed (Becker, 2017; van Eck, 2009).

Serious games are games that are designed for one or more purposes other than entertainment (Becker, 2017), and their main aim is to create or modify the player's behaviour by means of playing (Fernández-Vara, 2014). Serious games can be traditional games (e.g., board games, card games, or sport-type games) or digital games (e.g., console, computer, or mobile games) (Adams & Dormans, 2012), and they do not have a specific genre (Dörner et al., 2016). In other words, serious games can adopt the medium and genre that is adequate for their goals. According to Dörner et al., serious games can be classified according to their objectives (e.g., exergames, advergames, or educational games) or their target audience (e.g., games for professionals, patients, or students).

Due to the aims of this thesis, we will focus on digital educational games. According to de Freitas (2006), (digital) games for learning (or educational games) are "applications using the characteristics of video and computer games to create engaging and immersive learning experiences for delivering specified learning goals, outcomes and experiences" (p. 9).

2.2.4.1 Why Games for Learning?

This section provides a review of theoretical and empirical reasons given by scholars that justify the use of (digital) game-based learning. It is not intended as an exhaustive review, but it shows some of the most common justifications.

One of the most common rationale for the use of games in learning environments is that they can increase students' motivation (Whitton, 2014). Many scholars have suggested that video games possess characteristics that make them intrinsically motivating. For example, Malone and Lepper (Malone & Lepper, 1987) propose that challenge, curiosity, control, and fantasy are factors that make video games motivating. Similarly, de Freitas (2006) found that challenges, exploration opportunities, game realism, and learning control are factors that increases students' motivation. Additionally, other scholars (Kutner & Olson, 2008; Schell, 2014) propose that social factors are fundamental for making games motivating. As mentioned in section 2.1.3, students who are intrinsically motivated perform better in educational environments than less motivated students (R. M. Ryan & Deci, 2000a). Consequently, researchers have hypothesized that games have the potential to motivate the students, and at the same time, achieve learning objectives.

Another rationale is that well-designed video games can embody and effectively use learning theories and pedagogical approaches to support the learning of their contents. For example, Gee (2007) pinpoint thirty-six learning principles that “good” commercial video games have built-in. These learning principles are based on learning theories and pedagogical methods, such as scaffolding, active learning, learning by doing, connectionism, New Literacy Studies, situated cognition, situated learning and communities of practice. Similarly, Whitton (2012) identifies three learning principles used in video games that resemble contemporary learning theories, which are learning by doing, learning with others, and learning by problem-solving. Squire (2011) distinguishes eight characteristics that good games for learning should have based on an analysis of commercial games, such as *Civilization* (1991), *Pirates!* (1987), and *The Sims* (2000). Therefore, many scholars have visualized the potential of videos games as flexible learning tools, which can be adapted to many different contexts.

Chapter 3: Games for Learning Data Structures and Algorithms: A review

This chapter provides a systematic literature review of video games for learning data structures and all type of recursive algorithms. The review aims to answer SRQ 1: *What are the key game elements and pedagogical aspects of video games for learning data structures conceptual knowledge, and how is learning evaluated?* To achieve this objective, the review focuses on three areas: (1) game aspects, (2) pedagogical approaches, and (3) evaluation of the game. Concerning the game aspects, the review extracts the game genre, game elements (e.g., game mechanics, narrative elements, game challenge, levels, etc.), and technology used to develop the game. Regarding the pedagogical approaches, the review extracts and classifies the learning theories, pedagogical approaches, type of knowledge taught, and cognitive processes required to solve the game challenges. Concerning the evaluations, the review extracts and classifies the factors evaluated, experimental design, sample size, instruments, and analysis methods. Each game is analysed individually, and the results are summarized in table format followed by a discussion of the finding. This review highlights open areas of research and identifies game design elements and methodological aspects of the evaluations that can be used, studied, or improved.

3.1 Introduction

Data structures and recursive algorithms are fundamental topics in Computer Science (Sedgewick & Wayne, 2014). Their proper usage ensures the good performance of a computational system. For this reason, the Association for Computing Machinery (ACM) recommends their study at an early stage of the undergraduate program (2013). However, advanced data structures and recursive algorithms are challenging topics because they are abstract and difficult to relate to familiar knowledge (Becker & Beacham, 2000; C. C. Wu et al., 1998). Therefore, visualization tools have been created to facilitate the learning process. However, studies show that visualization tools do not increase learning gains due to the fact that students engage passively with such instruments (Hundhausen et al., 2002). Many researchers suggest that digital serious games may be useful for facilitating learning of data structures and recursive algorithms (Chaffin et al., 2009; Dicheva & Hodge, 2018), because serious games allow students to visualize the data structure in an active way.

This chapter presents the state of the art in digital serious games for learning that teach data structures and recursive algorithms. Specifically, this review aims to: (1) perform a game analysis of each digital game; (2) identify which digital games for learning focus on data structures and recursive algorithms; (3) identify and classify the specific data structures and recursive algorithms covered by those games; (4) analyse the learning theoretical foundations for the games; and (5) assess the studies performed to evaluate the effectiveness of each game.

3.2 Methodology

This review follows the guidelines suggested by Kitchenham and Charters (2004) for carrying out systemic literature reviews in computer science, which were later adapted by Calderón and Ruiz (2015) and Petri and von Wangenheim (2017) for reviewing serious games. The following sections describe the review objectives and protocol.

3.2.1 Research Questions

As mentioned above, the aim of this chapter is to answer SRQ 1. To achieve this objective, SRQ 1 is divided into eight sub-questions. Additionally, to understand the game elements that affect learning, a game analysis of each digital game reviewed is performed. This game analysis is executed with the information reported in the papers.

Game Elements

SRQ 1.1. Which game elements are used to teach and represent the content?

Topic and Pedagogical Aspects

SRQ 1.2. What are the data structures or recursive algorithms covered by the reviewed digital serious games?
SRQ 1.3. What are the learning theories or approaches used by the digital serious games to ensure learning?
SRQ 1.4. Which types of cognitive processes are required to achieve the digital serious games learning objectives?
SRQ 1.5. Which dimensions of knowledge are supported by the digital serious games?

Methodological Aspects

SRQ 1.6. Which factors are evaluated during the experiments?
SRQ 1.7. Which experimental design is used to evaluate the digital serious games?
SRQ 1.8. Which data collection tools are used in the study?
SRQ 1.9. Which data analysis methods are used to analyse the data?

3.2.2 Inclusion and Exclusion Criteria

For this review, we included articles that reported on one or more digital serious games that cover data structures and recursive algorithms. Specifically, we focused on articles written in English, available via digital libraries and published between 1999 and 2019. We did not include articles reporting on gamification or non-digital serious games. Articles lacking information to answer all the research questions were included as long as they could answer some of the questions.

Quality Criteria. We also assessed the quality of the reviewed articles. We only considered articles published in peer-reviewed journals and conference proceedings. Also, we excluded the articles that were not clearly written or possessed serious methodological problems.

3.2.3 Extracted Data and Classification Criteria

To answer the SRQ 1 sub-questions, we developed guidelines to extract and classify the articles' data as enumerated in the following list.

1. **Game analysis**. The game analysis included the name of the game, genre, game engine (or technology used to develop the game), and a description of the game and its game elements, such as the environment, mechanics, story and challenges.
2. **Covered topic**. The covered topic consisted of the name of the data structure or recursive algorithm covered by the game. Data structures were presented without classification in order to be as comprehensive as possible. Finally, recursive algorithms were aggregated and presented under a category named "recursive algorithm."
3. **Learning theory or principle**. For each article, the learning theory or principle that the digital serious games used to facilitate learning was extracted. Learning theories were defined as theories that explain how humans learn (e.g., Situated Learning (Lave, 1991) or Kolb's Experiential Learning Theory (D. A. Kolb, 2014)). Learning principles were defined as constructs, concepts, methodologies, or processes that facilitate learning (e.g., scaffolding (Zydney, 2012), or learning by analogies and metaphors (Duit, 1991)). We only extracted the learning theory/principle if it was reported in the article.
4. **Cognitive process and knowledge dimension**. To classify the cognitive processes that a player must apply to achieve the learning objective and dimension of knowledge delivered by the game, Bloom's revised taxonomy framework (Anderson & Krathwohl, 2001) was used. According to (Anderson & Krathwohl, 2001), there are six cognitive process: (1) *remember*, (2) *understand*, (3) *apply*, (4) *analyse*, (5) *evaluate*, and (6) *create*. Additionally, there are four dimensions of knowledge: *factual*, *conceptual*, *procedural*, and

metacognitive. Usually, the cognitive processes and knowledge dimensions of a learning tool should be reported in the learning objectives section (Biggs & Tang, 2007). However, if an article did not mention them explicitly, we deduced these aspects from the game description, paying attention to the actions (verbs) that the player must perform while playing the game (the learning activities).

5. **Evaluated factors**. Evaluated factors commonly assess the users' behaviours or opinions about the game. To classify these factors, we used the factor classification framework suggested by Petri and von Wangenheim (2017). The framework has ten categories: *learning*, *motivation*, *user experience* (UX), *usefulness*, *usability*, *instructional aspects*, *correctness*, *completeness*, *quality*, and *7S-model features*. Some articles do not explicitly report the evaluated factor. In those cases, we deduced the factors from the article's description of the data collection instruments.
6. **Research design**. Research designs were classified using the classification framework suggested by Petri and von Wangenheim (2017). This framework divides experimental designs into four categories: *ad-hoc*, *non-experimental*, *quasi-experimental*, and *experimental*. The ad-hoc category includes designs that analyse "learner's in-formal comments after they played the game or describing some observations of pilot studies" (Petri & Gresse von Wangenheim, 2017). The non-experimental category consists of systematically defined evaluations that do not follow a strict experimental design. Experimental designs use random assignment to allocate the participants in either the treatment or the control group. In contrast, quasi-experimental designs do not employ the random assignment approach.
7. **Sample size**. We considered only the number of participants who finished the experiments. The classification groups used were 1-20, 21-50, 51-100, and 101 or more.
8. **Instrument**. Data collection tools used in the game evaluations, such as qualitative surveys, tests/questionnaires, interviews, and observations were extracted from the selected articles.
9. **Data analysis methods**. The name of the analysis methods and type of method (quantitative or qualitative) were extracted from each article. Quantitative methods used were classified as either descriptive or inferential statistics.

3.2.4 Search Strategy

Digital libraries reviewed included ACM Digital Library, IEEE Xplore, SpringerLink, SAGE Journals, ScienceDirect, and Scopus. We selected these data sources because they have great influence in the computer science domain. Furthermore, we searched for additional related articles using Google Scholar to consider studies indexed on different journals outside of the mentioned databases.

For each data source, we defined a search string using core concepts and their synonyms. The following key words were used for the construction of each string: educational games, serious games, game-based learning, data structures, recursion, and sorting. Table 3.1 shows the search string used in each database.

Table 3.1: Search Strings.

Data source	**Search String**
ACM Digital Library	("educational games" OR "serious games" OR "game-based learning") AND ("data structures" OR "recursion" OR "sorting")
IEEE Xplore	("educational games" OR "serious games" OR "game-based learning") AND ("data structures" OR "recursion" OR "sorting")
SpringerLink	("educational games" OR "serious games" OR "game-based learning") AND ("data structures" OR "recursion" OR "sorting")
SAGE Journals	for [[All "educational games"] OR [All "serious games"] OR [All "game-based learning"]] AND [[All "data structures"] OR [All "recursion"] OR [All "sorting"]]
ScienceDirect	("educational games" OR "serious games" OR "game-based learning") AND ("data structures" OR "recursion" OR "sorting")
Scopus	("educational games" OR "serious games" OR "game-based learning") AND ("data structures" OR "recursion" OR "sorting")
Google Scholar	("educational games" OR "serious games" OR "game-based learning") AND ("data structures" OR "recursion" OR "sorting") Custom range: 1999-2019

3.2.5 Execution of the review

We performed the systematic literature review between December 2019 and June 2020. The review was executed in three stages. Table 3.2 shows the results (number of articles) of each stage. In the first stage, the initial search, queries were executed in all selected digital libraries and Google Scholar. After executing the queries, 9795 articles were retrieved.

Table 3.2: Execution of the literature review.

	ACM	**IEEE Xplore**	**SpringerLink**	**SAGE**	**Elsevier**	**Scopus**	**Google Scholar**	**Total**
Stage 1. Initial Search	23	32	1053	65	594	78	7950	9795
Stage 2. Brief analysis	23	32	1053	65	594	78	350	2195
Stage 3. Complete analysis	5	15	1	0	0	3	7	31
Final selection	4	11	1	0	0	1	2	19

In the second stage, the title of each article retrieved from the digital journals (including the 350 most relevant articles pulled from Google Scholar) were read. In total, this led to us reviewing 2195 articles in the second stage. When the title did not provide enough information to exclude or include the article, we proceeded to read the article's abstract. All repeated articles were excluded. At the end of the second stage, 31 promising articles were identified.

In the third stage, we proceeded to read the whole article and to extract the data. During this stage, some articles were excluded due to the following reasons: some were not digital games, others were not legibly written, and others were about gamifications or visualization tools. At the end of this stage, we found nineteen articles reporting data on fifteen serious games and two bundles of mini games designed to teach data structures and recursion.

3.3 Data Structure and Recursion Video Games

In this section, an analysis of each game that passed the selection criteria is presented. Games are arranged by the topic that they focus on. The classification categories are array games, stack games, linked list games, tree games, recursion games, coding games, and bundle of mini games.

3.3.1 Array Games

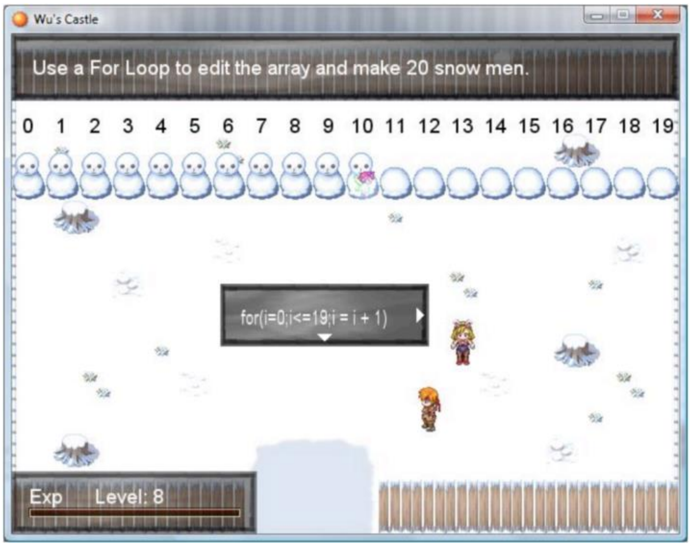
An array is a data structure that stores a sequence of values of the same type identified by an array index or key (Sedgewick & Wayne, 2014). Arrays can have one or more dimensions. Even though arrays are fundamental data structures, only one game focused exclusively on this data structure. In this section, we present an analysis of the one game that focuses on arrays.

3.3.1.1 Wu's Castle

3.3.1.1.1 Description

Wu's Castle (Eagle & Barnes, 2009) is a 2D puzzle-coding game built in RPG Maker XP and teaches arrays and loops to undergraduate students. The characters of the game are snowmen, Tsui, Koi, Machina, and the player's avatar, which is shown as a human. Tsui and Koi are non-player characters who guide the player during the missions, and Machina is a fairy who executes the code provided by the player. Concerning the game design, *Wu's Castle* possesses two game modes. In the first mode, the player can interact with the game by controlling loop parameters such as the start, end, and increment values of the for-loop structure (see Figure 3.1). In the second mode, the player physically walks the avatar through two paths that represent a nested loop (see Figure 3.2).

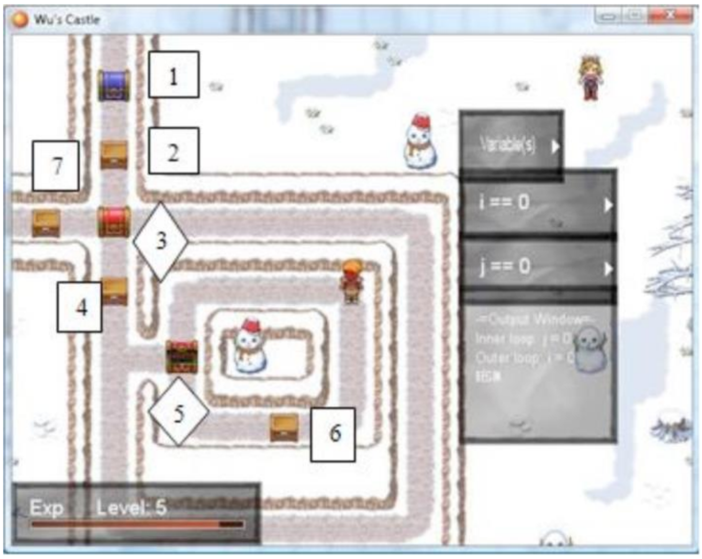
Wu's Castle is divided into three levels. In the first level, the player must create and locate snowmen following the patterns requested by the game. In this level, the game environment represents a one-dimensional array where it is possible to create a snowman. Concerning the game challenge, the player must indicate the position where the snowmen are going to be created by changing the start, end, and increment values of a for-loop. In the second level, the video game teaches the nested for-loop and the nested do-while-loop. Both nested loops are represented by an outer circular path and an inner circular path. Each path possesses several chests that represent executable instructions, and the act of the player touching a chest represents the execution of the instruction. Regarding the game challenge of this level, the player must guide the avatar through the outer and inner path executing the instructions and following the loop's indexes given by the game. Finally, in the third level, the player must locate and create snowmen by assigning the values of the indexes of nested loops.



Use a For Loop to edit the array and make 20 snow men.

```
for(i=0;i<=19;i = i + 1)
```

Figure 3.1: Screenshot of the first level of *Wu's Castle*.



```
i == 0
j == 0
=Output Window=
Inner loop: j = 0
Outer loop: i = 0
```

Figure 3.2: Screenshot of the second level of *Wu's Castle*.

3.3.1.1.2 Pedagogical Approach

Wu's Castle was designed to offer immediate feedback. According to its game designers, this type of feedback allows players to understand their progress and learn from their mistakes. Additionally, the video game uses scaffolding code to help students to complete the game challenges. Scaffolding code is pre-written code provided by the instructor or learning activity to help students get started with a coding task. Finally, from the description of the game, we deduced that the type of knowledge supported by the game is procedural, and it is required to employ the *understand, apply,* and *analyse* cognitive processes to complete the game challenges.

3.3.1.1.3 Evaluation Aspects

Eagle and Barnes reported two evaluations: a pilot (2008) and a full experiment (2009). The pilot study was a quasi-experiment with an experimental group and two control groups. The experiment measured the learning gains, enjoyability, preference, perceived learning, and motivation before and after playing the game. In total, 27 undergraduate students enrolled in an introductory computer science course participated in the study. Participants were divided into three groups: the "Control-NoPretest" group, the "Control-Pretest" group, and the "Game" group. The "Game" group made up of nine students performed the following tasks: First, they answered a pre-test; then, they played *Wu's Castle*; and finally, they answered a post-test and a qualitative survey. The "Control-Pretest" group made up of seven students answered the pre-test. The "Control-NoPretest" group made up of eleven students did not answer the test and did not play the game. Finally, Eagle and Barnes reported the results of the final exam of the course as the final observation of the experiment.

Regarding the data collecting tools, Eagle and Barnes reported the results of a test and a qualitative survey. The test had five questions, which required the students to read, execute, and analyse pseudo code. The qualitative survey had eleven five-point Likert scale questions. To analyse data, Eagle and Barnes used descriptive statistics (percentages and averages) and inferential statistics (a t-test). Results showed that the students who played *Wu's Castle* performed better in the final exam than those students who did not play the game. Additionally, the exposure to the pre-test did not affect the performance in the final exam.

In the second experiment, Eagle and Barnes (2009) evaluated the learning gains, enjoyability, usability, preference, and perceived learning after playing the game. To measure the learning gains, Eagle and Barnes performed a randomized stratified sample with crossover experimental design known as "switching replications." In the experiment, participants performed two activities and completed three tests (a pre-test, mid-test and post-test). The participants were randomly divided into two groups (experimental group and control group). The experimental group performed the following activities: (1) completed a pre-test; (2) performed an experimental activity (play the game); (3) completed a mid-test; (4) performed a control activity (complete a coding assignment); and (5) completed a post-test. The control group completed the tests following the same order as the experimental group, but they performed the control activity first and then the experimental activity. Additionally, at the end of the experiment, the participants completed a qualitative survey about enjoyability, usability, preference, and perceived learning. Concerning the participants of the experiment, 92 computer science students were randomly assigned into two groups of 46 participants, but only 26 students in the experimental group and 29 students in the control group completed all the tests. The tests had five questions, and the qualitative survey had ten five-point Likert-scale questions. Data was analysed using a one-tailed t-test on matched samples and considered significant at the p=0.05 level. Additionally, Eagle and Barnes used descriptive statistics (averages, line chart, and histograms) to analyse the test and the survey results. The results showed that *Wu's Castle* was more effective than a traditional programming assignment about loops, arrays, and nested for-loops problems. Finally, Eagle and Barnes concluded that playing a video game for learning before doing a coding assignment is more effective than just doing the coding assignment.

3.3.1.1.4 Analysis

In relation to the game design, we observe that *Wu's Castle* used the game environment as a model that represents elements of the array, for-loop, and do-while-loop structure. For example, the yard covered with snow that represents an array (Figure 3.1), the inner and outer paths that represent the structure of a nested loop (Figure 3.2), and some chests on the paths that represent the execution of an instruction. Also, we found that the video game includes non-trivial challenges using simple game mechanics, such as (top-down) moving mechanics and prompt mechanics. For example, the player can use the prompt system to modify the indexes of the for-loop. Also, the player can walk through the paths following the inner and outer loop indexes displayed in the second level. The environment and challenges of the game allow the player to learn and practice basic concepts of the array data structure and its manipulation using loops. Overall, *Wu's Castle* possesses non-trivial challenges and appropriately teaches array and loop concepts through the game elements.

Regarding the game genre, we noticed that Eagle and Barnes inaccurately stated that *Wu's Castle* is a role-playing game (RPG). Generally, RPG games focus on the development of the character and have strong storylines (Bateman & Boon, 2005). From the description of the game, we did not find any description regarding the development of the character and the narrative. Also, we deduced from the description that the game is a 2D puzzle game. To prevent such misclassification, we suggest that game designers who place their game within a genre should also report a description and justification of the game genre selected. Genre information can be useful for further research, e.g., it can be used to identify strengths or weaknesses of certain features of the genre to achieve learning objectives, or it can be used to compare features between genres based on the results of their evaluations.

In relation to the pedagogical aspects, we found that Eagle and Barnes did not report a learning theory to explain how the game supports learning. Also, they did not explicitly report the learning objectives of the game. According to several authors (Biggs & Tang, 2007; Mayes & de Freitas, 2004), learning tools designed to be used in formal education environments benefit from clarity around the underlying learning theory and the specific learning objectives, because they facilitate the assessment of the learner and the tool. We suggest that game designers should include and report the learning theory and the learning objectives of the game. This information will be useful in further research because it will help to understand the contents, its scope, and evaluation of the game.

Concerning the game evaluations, we found that Eagle and Barnes carried out two evaluations of the game: a quasi-experiment and an experiment. In total, 82 participants finished the experiments (27 in the quasi-experiment and 55 in the full experiment). We found that Eagle and Barnes analysed data using parametric methods. However, they did not report the methods used to verify the assumptions of parametric data. Additionally, to assume that scores obtained through a test are interval numbers, it is necessary to calibrate the test using Item Response Theory. However, Eagle and Barnes did not report the calibration of the test. Using parametric methods with data that does not meet the parametric assumptions can lead to incorrect results. A solution to this issue is to use a non-parametric method equivalent to the t-test or use descriptive statistical methods for the analysis.

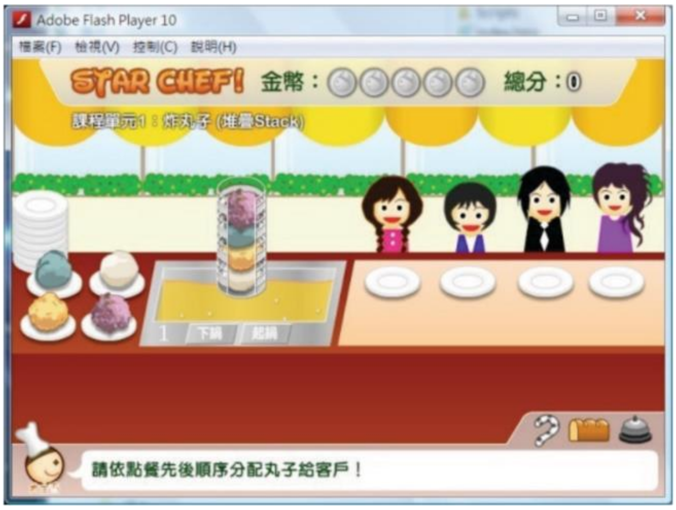
3.3.2 Stack and Queue Games

A stack is a collection of elements that follows the *last-in-first-out* (LIFO) principle (Sedgewick & Wayne, 2014). In other words, the last element stored in the stack will be the first element to be retrieved. A queue is a collection of elements that follows the *first-in-first-out* (FIFO) principle (Sedgewick & Wayne, 2014). This means that the first element stored will be the first element to be retrieved. The stack and queue data structures are simple but important concepts in computer science, and they are ideal for prototypes of games for learning data structures. Consequently, most of the games reviewed focus on these data structures. In this section, we present an analysis of five games that focus on the stack or queue.

3.3.2.1 Star Chef

3.3.2.1.1 Description

Star Chef (Liu et al., 2012) is a 2D casual game that teaches several data structures and algorithms (see Figure 3.3). The game was built using Flash, and it possesses five learning units. Each unit focuses on a specific data structure or algorithm, such as the stack, queue, bubble sort, tree traversal, and binary search. For example, the first level of the game, called the *Fry Meat Balls Game,* introduces the stack principle last-in first-out (LIFO). In this level, the player must deliver meat balls to the customers applying the LIFO rule. Unfortunately, Liu and colleagues did not describe the other levels.



請依點餐先後順序分配丸子給客戶！

Figure 3.3: Screenshot of the first unit of *Star Chef* called "Fry meat balls".

3.3.2.1.2 Pedagogical Approach

From the game description, we deduced that the player most employ the *remember* cognitive process to solve the game challenges. Additionally, the game teaches procedural knowledge. Finally, Liu et al. did not report a learning theory, principle or pedagogical approach used to design the game.

3.3.2.1.3 Evaluation Aspects

Liu et al. (2012) evaluated the acceptance of serious games and aimed to answer the following research question: “What is the difference between the technology acceptance of the proposed computer game and those of the simulation software?” To achieve the study’s aim, Liu and colleagues used Technology Acceptance Model (TAM). According to Davis et al. (1989), technology acceptance can be explained using the TAM that specifies the “causal linkages between two key beliefs: perceived usefulness and perceived ease of use, and users' attitudes, intentions and actual computer adoption behaviour."

Concerning the evaluation design, Liu et al. (2012) executed a quasi-experiment. The study contrasted two learning tools: a serious game and a data structure visualizer. The experimental group used *Star Chef*. On the other hand, the control group used a set of simulation tools from the Java Applet Centre. Furthermore, the visualization tools selected focus on the following topics: stack, queue, bubble sort, tree traversal, and binary search. The experiment took place during an Introduction to Data Structures course at the Lunghwa University of Science and Technology, and it had a duration of eight weeks. In the experiment, students attended traditional classes, but in addition to their classes, students completed a set of self-learning activities where they used one of the two mentioned learning tools. At the end of the course, lecturers administrated the TAM survey to evaluate the acceptance of the tools. One hundred and ten students from two classes participated in the experiment. All participants were majoring in Game Science and possessed computer science backgrounds. Liu and colleagues assigned one class to the control group and the second to the experimental group.

The data collection tool used was a survey based on Davis's TAM (Davis et al., 1989). The survey had twelve questions divided in three categories: usefulness, easiness, and attitude toward use of learning tools. To analyse the data, Liu et al. calculated Cronbach's alpha to verify the internal consistency of the survey. Also, they used descriptive statistics (mean, standard deviation, and standard error), and inferential statistics, hypothesis testing using ANOVA. Liu et al. (2012) analysed the F value of each question individually, and results showed that ten items of the questionnaire presented a statistically significant difference. Finally, the Liu and colleagues concluded that students may prefer computer games over visualizing tools.

3.3.2.1.4 Analysis

Concerning the game design, we found that Liu et al. did not report the game genre and that they omitted a description of most of the game elements, such as the game mechanics, challenges, and winning condition. We deduced that *Star Chef* is casual game based on the brief description of one level and one image provided by the authors of the article reviewed. Furthermore, we did not find any information regarding the development of the game. For example, it is not reported who developed the game or which game engine was used. Without this information, it is not possible to understand the scope of the game, its strengths or weakness. All these elements are necessary to understand which game elements may cause acceptance.

In relation to the pedagogical aspects, we found that Liu et al. did not provide a learning theory or pedagogical approach used by the game to support learning. Additionally, they did not report the learning objectives or learning activities of the game. For this reason, other researchers who read this work cannot know what and how students were learning while playing the game. Furthermore, it is not possible to know the scope of the topics covered by the game.

Regarding the experiments, results showed that Liu et al. executed a quasi-experiment with a large sample size (110 participants). Also, we found that the was executed in an introductory course of data structures at the Lunghwa University of Science and Technology, improving the ecology of the experiment. Concerning the data collection tool, the evaluators designed the survey based on the TAM, which is a validated and calibrated survey (Davis et al., 1989). This may improve the validity of the items. However, it does not mean that the new adaptation of the TAM survey has scale properties. To achieve scale properties, it is necessary to calibrate the tool using Item Response Theory methodologies. Regarding the analysis methods, we found that Liu et al. did not report the analysis that verified the parametric assumptions of data.

3.3.2.2 Stacks and Queues

3.3.2.2.1 Description

Stacks and Queues (Park & Ahmed, 2017) is a casual game that teaches the adding (push/enqueue) and removing (pop/dequeue) algorithms of the stack and queue data structures (see Figure 3.4). The game was developed in Unity 3D. In the game, the player must protect two piles of blocks that represent a stack and a queue from encroaching rockets. To protect the piles, the player must move the piles up or down by pressing the up and down keys; however, the player can only move one structure at the same time. For this reason, the player must switch from one pile to the other by pressing the left and right keys. Additionally, the player must intercept the rockets by shooting (popping/dequeuing) blocks from the piles. To pop or dequeue, the player must press the left button of the mouse, and to push or enqueue, the player must press the right button.

3.3.2.2.2 Pedagogical Approach

Park and Ahmed (2017) justify the use of serious games to teach data structures arguing by that games assist constructivist learning and are “learner-centred.” Additionally, they state that serious games improve motivation, communication, collaboration, computer skills, problem-solving skills, and subject knowledge. According to Park and Ahmed, *Stacks and Queues* is an interesting and interactive learning tool that improves motivation. Finally, we deduced from the game description that the type of knowledge covered by the game is factual. Also, we deduced that players must employ the *remember* cognitive process to solve the challenges.

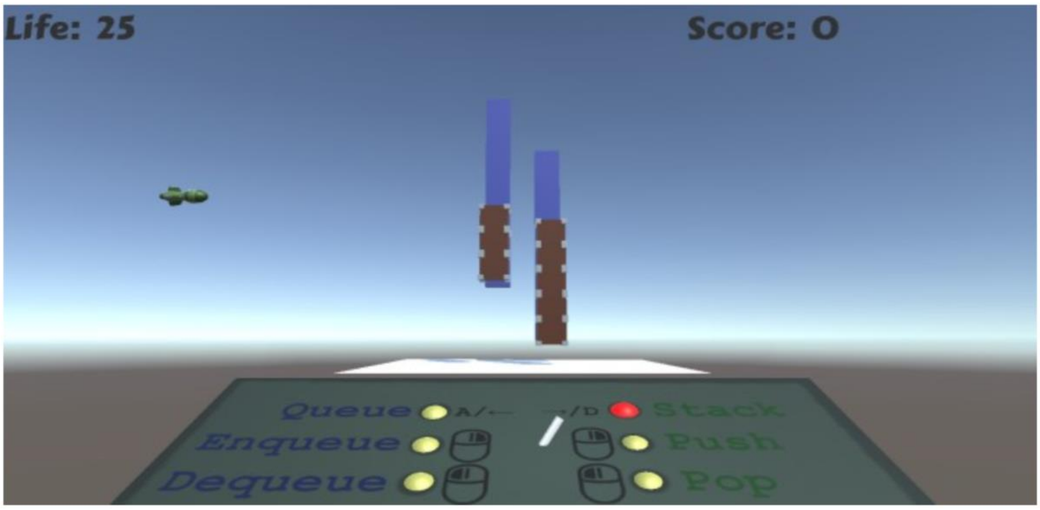


Figure 3.4: Screenshot of *Stacks and Queues*' level.

3.3.2.2.3 Evaluation Aspects

Concerning the evaluation design, Park and Ahmed (2017) carried out a non-experimental study where 32 students from the university of North Carolina at Charlotte participated. The evaluation consisted of a pre-learning activity survey divided into three categories (demographic, learning styles, and gaming expectation), a learning activity using the game, and a qualitative survey about the gaming experience. The pre-learning activity survey consisted of twelve questions. The qualitative survey had five three-point and one five-point Likert scale questions divided into three categories: usability, perceived learning, and preference. To analyse the data, Park and Ahmed used descriptive statistical methods (histograms and percentages). The evaluation results showed that most of the participants thought that the game was interesting. Also, participants perceived that they learned while playing the game and indicated that they would like more serious games in their classes.

3.3.2.2.4 Analysis

Concerning the game design, we found that Park and Ahmed used the popping and dequeuing functions of the stack and queue data structures as shooting game mechanics. This means that during the game, the player must be aware of the pile selected because the pile representing the stack can only shoot from the top, while the pile representing the queue can only shoot from the bottom. This challenge may help the player to memorize and recognize the difference between the two data structures. However, this challenge is simple and limited, and the game does not provide further levels or challenges. Furthermore, the game aesthetics were built using the prototyping resources provided by Unity 3D. For this reason, they are not visually appealing. Such design issues may reduce the player's engagement.

Regarding pedagogical aspects, we noticed that while Park and Ahmed did identify that the game is based on constructivism, they did not explain how the game supports constructivism or how the player learns from the game. Furthermore, they did not report the learning objectives and learning activities of the game. As stated in previous sections, these three aspects facilitate the assessment of the learner and the evaluation of the tool.

In relation to the evaluation, we found that the questions asked in the qualitative survey were general. For example, the first question was "Was this game interesting and interactive?" Half of the participants answered "Yes." However, Park and Ahmed did not report which elements of the game were interesting and interactive. In other question, participants were asked "Did the game help you understand the data structures?" However, the scope of the game was not described in the work reviewed.

3.3.2.3 Stack Em Up

3.3.2.3.1 Description

Stack Em Up (Ismail et al., 2013) is a 2D puzzle game based on the Tower of Hanoi problem (Figure 3.5). The Tower of Hanoi puzzle requires the player to transfer several disks from an initial rod to a final rod following these rules: (1) the player can only move one disk at a time; (2) the player can only take the topmost disk from one rod and transfer it to another rod; (3) the player cannot put a bigger disk over a smaller one. *Stack Em Up* possesses three levels, and each level has a different level of difficulty. The difficulty of each level is defined by the number of disks on the initial rod. For instance, in level 1 (the easiest level), the player must move three disks from the initial rod to the final rod. In level 2, the player must move five disks, and in level 3, seven disks. As part of the game challenge, the player has a limited number of movements per level. For example, in level 1, the player must move the disks in seven movements. To move a disk, the player should drag and drop the disk from one rod to another rod. Every time that the player completes a level, the game unlocks a stack concept. The player can check the unlocked concepts in the stack concept screen (Figure 3.6).

3.3.2.3.2 Pedagogical Approach

Stack Em Up teaches two concepts of the stack data structure: (1) the last-in first-out (LIFO) and (2) the PUSH and POP stack operations. We deduced from the game description that the player must utilize the *apply* cognitive process to solve the game challenges. Additionally, the knowledge covered by the game is conceptual and procedural.

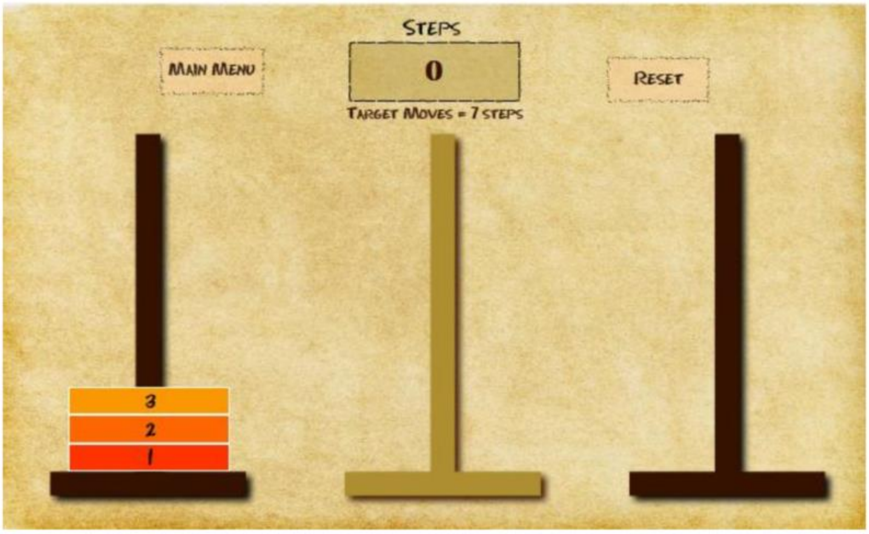
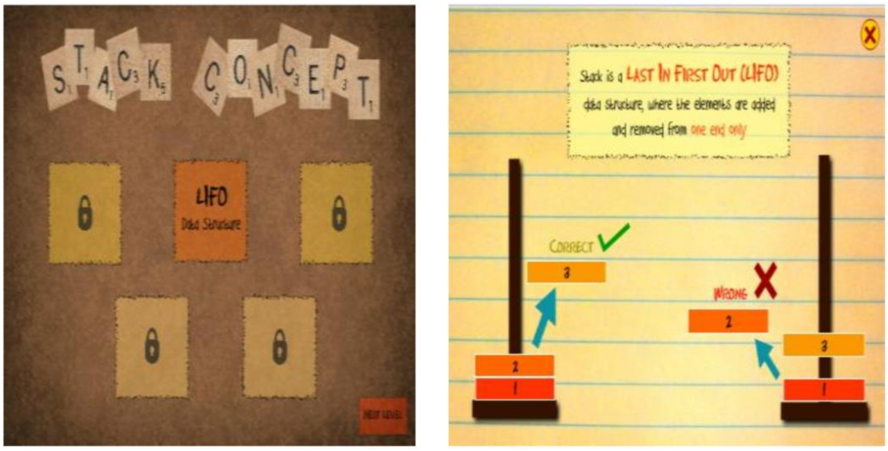


Figure 3.5: Screenshot of the first level of *Stack Em Up*.



Stack is a LAST IN FIRST OUT (LIFO) data structure, where the elements are added and removed from one end only

Figure 3.6: Stack concepts screen of the first level of *Stack Em Up*.

To design the game, Ismail et al. (2013) adopted a constructivist stance. They reported that video games facilitate active learning because they allow the students to build their knowledge through experiencing and exploring the game learning activities. Additionally, they indicated that the design of *Stack Em Up* was guided by the following principles:

1. The game should cleverly present the content in a game-like scenario.
2. The game should include motivational and cognitive elements that facilitate learning.
3. The game should apply the Flow principle; the game activities should be challenging but not extremely difficult.

3.3.2.3.3 Evaluation Aspects

Ismail et al. executed a non-experimental evaluation of the game suitability. They defined suitability as referring to the correct functioning of the game and the fulfilment of the players' expectations. In the study, 15 randomly selected students played the game and then completed a qualitative survey. The survey consisted of nine yes or no questions divided into five categories: performance issues, play issues, calculation issues, graphics issues and functional testing. The data analysis methods used by Ismail et al. were histograms and percentages (descriptive statistics). Finally, results showed that the tool functionality was acceptable and suitable for future testing.

3.3.2.3.4 Analysis

Concerning the game design, we found that the aesthetics were consistent, and players may find them appealing. Also, we found that the reward system may motivate players. However, the game challenges were monotonous. The Hanoi Tower puzzle can be solved applying a simple algorithm that it is easy to discover. Additionally, we noticed that the game lacks narrative elements. The lack of narrative combined with monotonous challenges may decrease the player's engagement.

Concerning the pedagogical aspects, we found that Ismail et al. reported the learning theory and learning principles that support learning. Additionally, we found a clear description of how and what the students will learn while playing *Stack Em Up*. In other words, the authors implicitly stated the learning objectives and activities. This will facilitate future assessment of participants' learning gains and evaluations of the game.

In relation to the game evaluation, results showed that Ismail et al. specified the factors evaluated, and they used adequate statistical methods to analyse data. However, the evaluation was limited, because their evaluation design was non-experimental, and the sample size was small (fifteen participants).

3.3.2.4 Stack Game

3.3.2.4.1 Description

Stack Game (Dicheva et al., 2016; Dicheva & Hodge, 2018) is an isometric action adventure game developed in Unity 3D. The game story is about a robot that is returning home after completing a mission on a foreign in space. During its travel, the robot's spaceship has had a mechanical failure, and for that reason, the robot must return home walking. However, several doors secure the path to return home. To traverse those doors, the robot must unlock them by solving stack-related problems.

Stack Game is divided into three chapters, and each chapter has several levels that correspond to a learning objective. The first chapter (Figure 3.7) has four levels, and its game challenges are based on the Tower of Hanoi puzzle. In this chapter, the player must arrange cubes in a specific order to unlock the door, but he or she can only move the cubes by following the Tower of Hanoi rules.

The game's second chapter (Figure 3.8) possesses six levels. In the first and second levels, the player must convert an arithmetic expression from an infix to a postfix notation. In the third and fourth levels, the player must evaluate a postfix expression. Finally, in the fifth and sixth levels, the player must evaluate an infix expression.

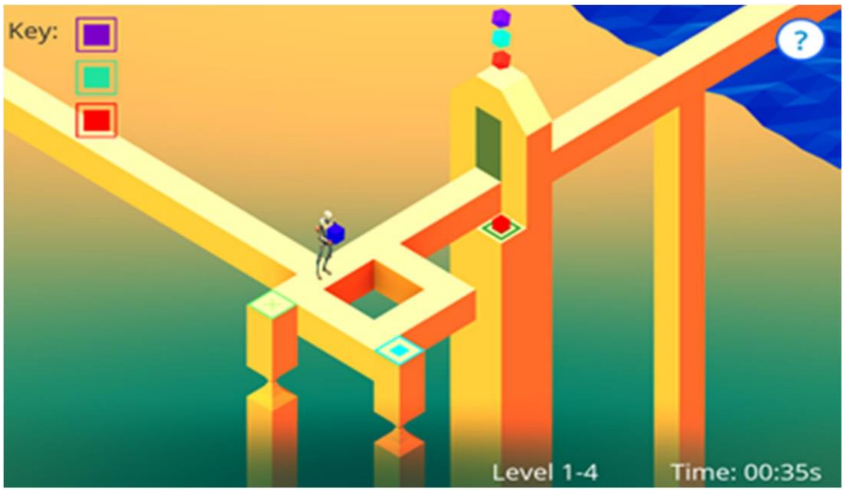


Figure 3.7: Screenshot of Chapter 1 ─ Level 4 of *Stack Game*.

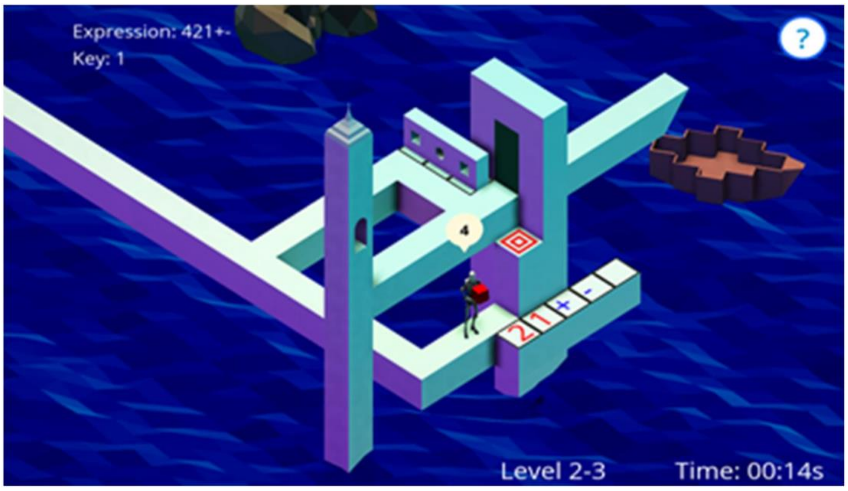
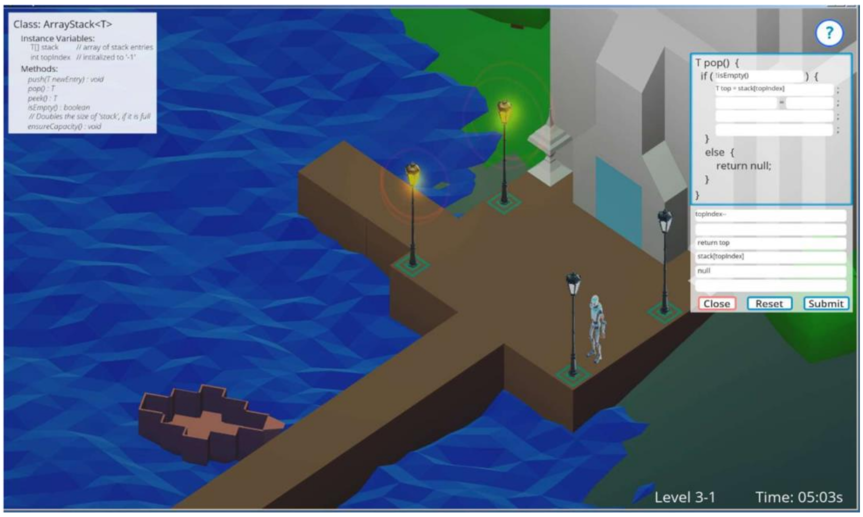


Figure 3.8: Screenshot of Chapter 2 ─ Level 3 of *Stack Game*.

The game's third chapter (Figure 3.9) has four levels. In this chapter, players must code operations of the stack data structure. In the first and second levels, the player must complete the code of the functions through a drag-and-drop system. In the third and fourth levels, the player must program all the functions from scratch.



```
Class: ArrayStack<T>
  Instance Variables:
    T[] stack      // array of stack entries
    int topIndex   // initialized to '-1'
  Methods:
    push(T newEntry) : void
    pop() : T
    peek() : T
    isEmpty() : boolean
    // Doubles the size of 'stack', if it is full
    ensureCapacity() : void
```

```
T pop() {
  if ( !isEmpty() ) {
      T top = stack[topIndex] ;
               =              ;
                              ;
                              ;
  }
  else {
    return null;
  }
}
```

```
topIndex--

return top
stack[topIndex]
null
```

Figure 3.9: Screenshot of Chapter 3 ─ Level 1 of *Stack Game*.

3.3.2.4.2 Pedagogical Aspects

Stack Game teaches fundamental concepts, operations, and applications of the stack data structure. According to Dicheva (Dicheva et al., 2016), the game was designed following a constructivist stance, and it aims at facilitating active learning. Additionally, the game aims to improve players' motivation by adding fun elements to engage players and applying the "learning-by-doing" approach. Furthermore, Dicheva and Hodge (2018) reported that the game addresses three learning objectives:

1. Understand the abstract concept of the stack.
2. Implement the stack data structure and its operations.
3. Apply the stack data structure to solve problems.

We deduced that to achieve these learning objectives, players must employ the *understand*, *apply*, and *analyse* cognitive processes. Additionally, the type of knowledge covered by the game is procedural.

3.3.2.4.3 Evaluation Aspects

Dicheva and Hodge (2018) reported an evaluation of the motivation and learning gains obtained by participants after playing *Stack Game*. The study followed a quasi-experimental design, and it was executed in two sessions. In the first session, participants answered a pre-test, and then they played the first and second chapters of the game. In the second session, participants played the third chapter of the game and then answered a post-test. At the end of the evaluation, participants completed an online qualitative survey. In total, 29 students participated in the experiment.

The pre- and post-tests consisted of questions related to function execution, and to infix and postfix expressions. According to Dicheva and Hodge, both tests were of similar difficulty. This statement suggests that they designed two different tests. The qualitative survey had 16 questions, and it measured perceived usefulness, perceived educational value, preference, clarity, support provided, and enjoyment (Dicheva & Hodge, 2018).

To analyse data obtained from the tests, Dicheva and Hodge reported the usage of descriptive statistical (mean and standard deviation) and inferential statistical methods (t-tests and Cohen's d). To analyse data obtained from the survey, Dicheva and Hodge reported the usage of descriptive statical methods (percentages). Results showed that participants improved their knowledge.

3.3.2.4.4 Analysis

Concerning the game design, we found that *Stack Game* is in an advanced stage of development and presents an interesting narrative, appealing aesthetics, and clever game challenges. The game has a non-trivial story about a robot who is trying to reach his home. The game aesthetics are consistent between all chapters and are appropriate for the game story. The game challenges are engaging and related to a learning objective. The levels' difficulty varies from one level to another, and the type of challenge changes from one chapter to another. For example, challenges of the first chapter are based on the Tower of Hanoi puzzle, while challenges of the third chapter are coding puzzles. All these features may increase and sustain players' engagement and motivation.

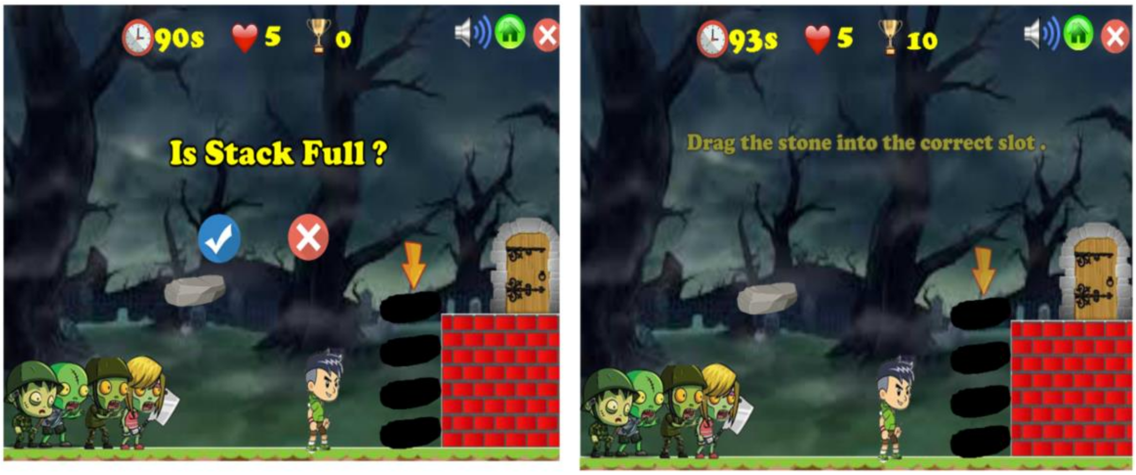
Concerning pedagogical aspects, we observed that Dicheva and Hodge reported the learning theory that explains how the game supports the acquisition of knowledge. Additionally, they listed the learning objectives covered by the game and explained how each chapter teaches each topic covered. In general, the pedagogical aspects were rigorously treated by Dicheva and Hodge. It was possible to understand the scope of the topics covered by the game, and the approach used to teach the topics.

Concerning the evaluation, we found that Dicheva and Hodge analysed data using parametric statistical tests. However, they did not report the verification of the assumptions of parametric tests. In addition, the pre- and post-tests were not calibrated using Item Response Theory to verify their scale properties. For this reason, data collected using these tools is ordinal, and it cannot be analysed using parametric tests. Non-parametric data (e.g., ordinal data) can be analysed using non-parametric tests. Finally, Dicheva and Hodge did not explain the evaluated factors of the qualitative survey. For example, they asked the participants about their motivation. Dicheva and Hodge did not specify which type of motivation was evaluated, for example *outcome-focused* or *process-focussed* (Touré-Tillery & Fishbach, 2014).

3.3.2.5 *Ramle's Stack Game*

3.3.2.5.1 Description

Ramle's Stack Game (Ramle et al., 2019) is a 2D puzzle game that teaches the LIFO (last-in first-out) principle and stack data structure operations (pop, push, peek, isEmpty and isFull; see Figure 3.10). In the game, the player must insert (push) and remove (pop) stones to build and destroy a ladder to evade zombie attacks. Before every insert action, the player must answer whether the stack is full (isFull). If the player answers correctly and there is an empty slot, the player can drag and drop a stone in last empty slot. Similarly, before every remove action, the player must answer whether the stack is empty (isEmpty), and then, the player can drag and drop a bomb to the topmost stone. Additionally, the game requires that players apply the LIFO principle to overcome the game challenge. The game score system increases 10 points every time the player answers correctly and decreases one life every time the player answers incorrectly. Finally, the game was built using Adobe Flash.



Is Stack Full ?

Drag the stone into the correct slot .

Figure 3.10: Screenshot of *Ramle's Stack Game*.

3.3.2.5.2 Pedagogical Approach

Ramle's Stack Game teaches stack operations and the LIFO principle using two pedagogical approaches. First, Ramle et al. suggested that game-based learning may connect abstract concepts with objects from the real world, facilitating the understanding of those concepts. Second, Ramle et al. used the scaffolding approach through question prompts to improve learning gains. Finally, we deduced that players employ the *remember* cognitive process to solve the game challenges, and the game covers factual knowledge.

3.3.2.5.3 Evaluation Aspects

Ramle et al. (2019) performed an evaluation of the game's usability and the learning gains obtained by students after playing the game. The evaluation was a quasi-experiment, and it was divided into three stages. First, participants completed a pre-test to measure the level knowledge before playing the game. Second, participants played the game. Finally, participants completed a post-test and a qualitative survey. In total, 29 students participated in the evaluation. All were students from the first-year Diploma in Information Technology at the University Tun Hussein Onn Malaysia.

Concerning data collection tools, Ramle et al. used a test and a qualitative survey. The test consisted of nine multiple-choice questions about basic stack concepts, stack operations, and implementation. The survey had fourteen five-point Likert scale questions about the user interface, user friendliness, and interactivity. Both tools were informal.

In relation to data analysis, Ramle et al. did not use any statistical method. Instead, they reported the results in a table. One column of the table had the aggregated values of the correct answers of the pre-test, and another column had the aggregated values of the correct answers of the post-test. Finally, to analyse the qualitative survey, Ramle et al. used percentages (descriptive statistics).

3.3.2.5.4 Analysis

Concerning the game design, we found that *Ramle's Stack Game* has appealing aesthetics. However, the game challenges are trivial. For example, the main task of players is just to answer a question (e.g., "Is the stack empty?" or "Is the stack full?") and then, drag and drop stones or bombs. Furthermore, to answer these questions, players do not need to exert any effort. This lack of challenge may decrease the motivation and engagement of players after a few minutes playing the game.

Regarding pedagogical aspects, results showed that Ramle and colleagues reported the learning theory and pedagogical principle used to facilitate learning through the game. However, they failed to apply the theory in the game design. For example, they suggested that scaffolding through question improves the acquisition of procedural knowledge. Yet, the questions formulated are trivial and difficult to relate to stack concepts. Finally, Ramle et al. did not report learning objectives or learning activities. Omitting these aspects makes it difficult to assess the learning gains and the tool.

3.3.3 Linked List Games

According to Sedgewick and Wayne (2014), “a linked list is a recursive data structure that is either empty (null) or a reference to a node having a generic item and a reference to a linked list” (p. 142). In other words, a linked list consists of nodes, each of which has a reference to another node or a null value. The linked list data structure is ideal to introduce the concept of nodes. However, we could only find two games focusing on this topic: *Space Traveller* and *La Petite Fee Cosmo*. In this section, we present an analysis of these two games that focus on the linked list data structure.

3.3.3.1 Space Traveller

3.3.3.1.1 Description

Space Traveller (Zhang et al., 2015) is a 2D action-coding game developed in GameMaker Studio. The main goal of the game is to protect a spaceship from enemies that try to destroy it. To protect the spaceship, the player must collect orbs that increase the length of the spaceship (Figure 3.11). After collecting some orbs and earning some money, players can purchase weapons and boosters that can be attached to the empty orbs of the spaceship. To attach a weapon to an orb, the player must be in a docking station. Additionally, the player must remove any damaged orbs to keep the spaceship alive longer.

The game possesses two gameplay modes: an action gameplay mode and a coding gameplay mode. The action gameplay mode is a vertical-scrolling shooter like *Xevious* (1983). In this mode, players must collect items, and shoot and avoid enemies. The game switches to the coding gameplay mode when the player docks the spaceship in a space station (Figure 3.12). In the space station, the interface changes, and a coding terminal appears. There, players can purchase weapons and add the new weapons on the orbs by filling in some sections with a Java-like pseudocode language.

Space Traveller has four tutorial levels and a final level. In the first level, the game introduces the basic controls. In the second and third levels, the game introduces the insert operation. In the fourth level, the game introduces the remove operation. In the fifth level, the player must perform three insertions in the front, three insertions at a specific location, and delete three orbs.

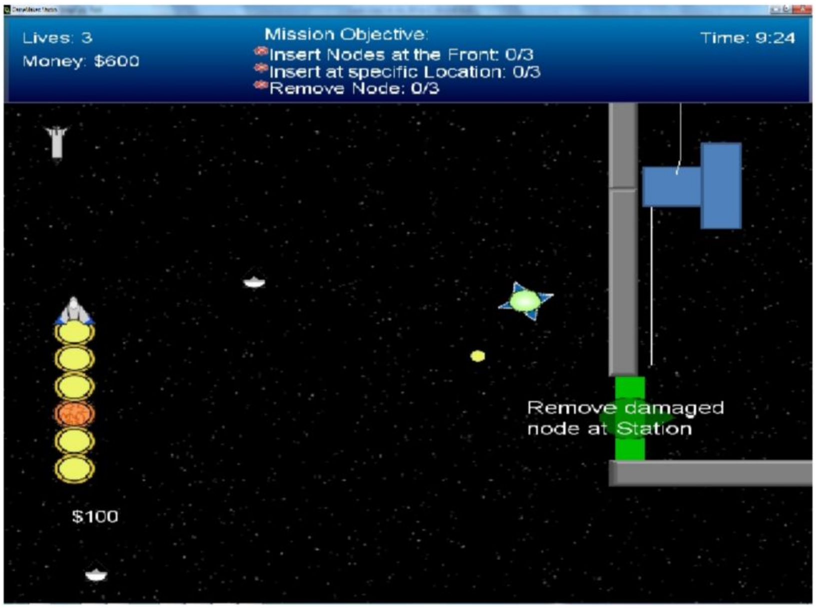
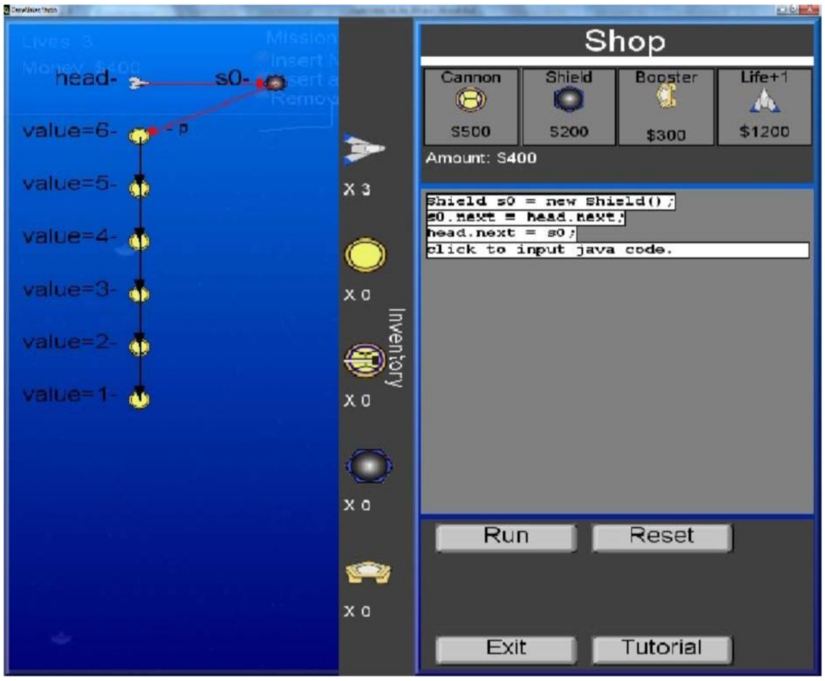

Lives: 3
Money: $600
Mission Objective:
Insert Nodes at the Front: 0/3
Insert at specific Location: 0/3
Remove Node: 0/3
Time: 9:24
Remove damaged node at Station
$100

Figure 3.11: Screenshot of *Space Traveller*'s action gameplay mode.



```
Shield s0 = new Shield();
s0.next = head.next;
head.next = s0;
click to input java code.
```

Figure 3.12: Screenshot of "space station" and the coding gameplay mode.

3.3.3.1.2 Pedagogical Approach

Space Traveller teaches linked list concepts, and while playing the game, players can practice the following operations: (1) insert at the front; (2) insert at a specific location; and (3) delete a node at a specific location. According to the Zhang et al., the game was built under the following premises: (1) game-based learning (GBL) enhances motivation, enthusiasm, and excitement about programming topics; (2) GBL enhances engagement; and (3) GBL helps students to learn difficult programming concepts. Finally, we deduced that players must employ the *understand*, *apply*, *analyse*, and *create* cognitive processes, and the game teaches procedural knowledge.

3.3.3.1.3 Evaluation Aspects

Zhang et al. (2015) reported a quasi-experiment consisting of a pre-test, a learning activity, a post-test and a qualitative survey. The factors evaluated were the learning effect, motivation, enjoyability, perceived learning, and usefulness after playing the game. In total, 13 students enrolled in the Data Structure course at Winston-Salem State University participated in the experiment.

The pre-test and post-test consisted of three questions, and each question was worth ten points. Questions asked students to complete missing sections of Java-like pseudocode. The survey had nine five-point Likert-scale questions concerning the player's experience after playing the game. To analyse data, Zhang et al. used descriptive statistics. For the tests, they reported the mean, median, standard deviation, variance, and a histogram. For the survey, they reported percentages and a histogram. Finally, Zhang et al. concluded that the game had a positive impact on student learning, but due the limitations of the experiment, it is necessary to perform more evaluations.

3.3.3.1.4 Analysis

Results showed that *Space Traveller* is a game that mixes game mechanics and challenges of the action and coding game genre. This combination may increase players' intrinsic motivation. For example, challenges of the action gameplay mode provide a context that encourages players to upgrade their spaceship. To achieve this, players must switch to the coding gameplay mode. There, players must solve the coding problems to upgrade their spaceship. At the same time, these new upgrades encourage players to try them in the action gameplay mode. Then, with the new upgrades, players will earn more money and find more orbs, allowing them to buy and add more upgrades. This cycle may keep players engaged, and as result, players will practice during a longer time their coding skills.

Concerning the pedagogical aspects, we found that Zhang et al. did not report a learning theory to support acquisition of knowledge. Additionally, they did not explicitly list the learning objectives covered by the game. Omitting these two aspects will cause difficulties in assessing the efficacy of the game's learning activities and the players' learning gains.

Regarding evaluation aspects, our results showed that Zhang et al. did not explain the factors evaluated in the qualitative survey. The omission of these definitions may cause problems during the formulation of the survey’s questions. The consequences could be that the questions do not measure the factors under observation. Furthermore, the pre-test and post-test were very similar to the coding challenges presented in *Space Traveller*. This type of assessment may evaluate memorization instead of an understanding of the concepts covered.

3.3.3.2 La Petite Fee Cosmo

3.3.3.2.1 Description

La Petite Fee Cosmo (Kannappan et al., 2019) is a 2D side-scrolling puzzle game built in Unity 3D that teaches linked list concepts. In the game, players must help the game character (Cosmo) to build and traverse a bridge made of floating platforms. To build a platform, players must remove items from their inventory and place them on an empty space (green squares; Figure 3.13). After building the bridge, some crystals and a portal will appear. Players must collect the crystals and traverse the portal to complete a level. The game bridges represent the linked list and each platform represent a node. Additionally, above each node, there is a glyph and a numeric value that represent the data stored in the node.

La Petite Fee Cosmo possesses three chapters. The first and second chapter have two levels each, and the third level has one. Each chapter has a different type of challenge. For example, in the first chapter, players must collect the largest or smallest value stored in the first or last node of the list. In the second chapter, players must sort the nodes’ values either in ascending or descending order. In the final chapter, players must relocate values of a lower bridge into an upper bridge (Figure 3.14).

3.3.3.2.2 Pedagogical Approach

La Petite Fee Cosmo teaches basic linked list concepts and operations. For example, the game covers topics such as adding, removing, finding, modifying links between nodes, and the head pointer. To teach those concepts, Kannappan et al. reported that *La Petite Fee Cosmo* supports a learning approach known as *productive failure*. According to Kapur (2014), productive failure is a methodology divided into two phases. In the first phase, the learner experiences an unknown problem and tries to solve it. In the second phase, the learner engages with the theory, concepts, and principles necessary to solve it. This approach prepares the learner for a better understanding of the theory and methods used to solve a problem. Furthermore, we deduced that the player must employ the *apply* cognitive process to solve the challenges, and the type of knowledge taught by the game is procedural.

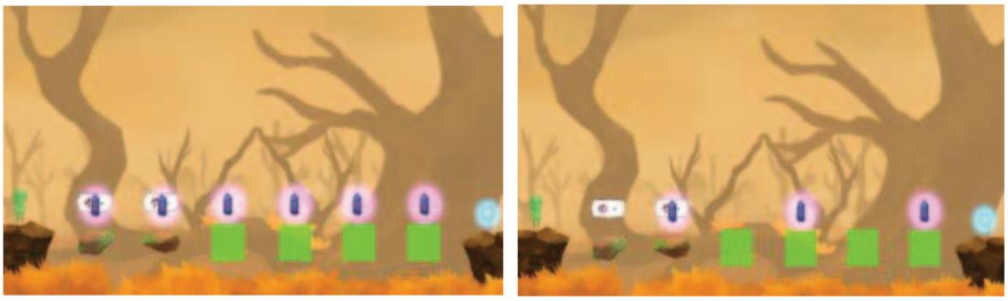


Figure 3.13: Screenshot of *La Petite Fee Cosmo*'s Chapter 2 ─ 1 and Chapter 2 ─ 2.

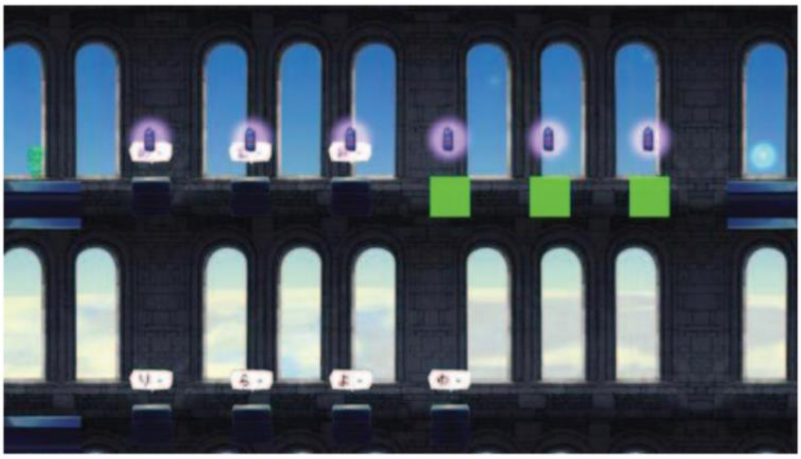


Figure 3.14: Screenshot of *La Petite Fee Cosmo*'s Chapter 3 ─ 1.

3.3.3.2.3 Analysis

Concerning the game design, we found that *La Petite Fee Cosmo* possesses appealing aesthetics. This feature may improve players' engagement. Additionally, the designers selected game mechanics that resemble the linked list's operations. These mechanics allow the player to visualize the execution of the operations and the outcomes of the operations.

Concerning learning aspects, results showed that Kannappan et al. reported a learning theory that explains how players learn while playing the game. However, they did not report learning objectives and learning activities. As stated before, omitting these elements may difficult the assessment of learning gains.

3.3.4 Binary Search Tree and AVL Tree Games

Binary search tree (BST) and Adelson-Velsky and Landis (AVL) tree data structures are modifications of the binary tree data structure. A binary tree can be either a null link or a node with a left and right link, each of which can point to a subtree that is also a binary tree (Sedgewick & Wayne, 2014). A BST is a binary tree with some special characteristics. First, its nodes have a comparable key and a value; and second, the node are arranged according to the BST property (Sedgewick & Wayne, 2014). A comprehensive explanation of BSTs can be found in section 5.2.1. Finally, the AVL tree is like a BST; however, the insert and delete algorithms were designed to keep the tree balanced. In other words, they keep the number of nodes of each side of the tree as equal as possible. In this section, we present an analysis of two games that focus on the BST and AVL tree data structures.

3.3.4.1 Super Mario AVL Tree Game

3.3.4.1.1 Description

Super Mario AVL Tree Game (Wassila & Tahar, 2012) is 2D action-puzzle game based on the aesthetics of *Super Mario Bros.* (1985). The game has five levels, and each level covers a concept or operation of the AVL tree. In the first level, the protagonist Mario must make a bouquet of flowers. Each flower possesses a number, and players must arrange the bouquet by applying the BST property (Figure 3.15). In the second level, Mario must visit the princess, but first, he must balance several sets of platforms arranged as an unbalanced binary tree (Figure 3.15). To balance the platform, the player can perform four types of rotations with the arrow keys. At the end of this level, Mario discovers that the antagonist Bowser has kidnapped the princess. In the third level, the game environment represents an AVL tree (Figure 3.16). Mario must search for a key to enters Bowser’s castle. To find the key, players must apply the BST property. In the fourth level, Mario must build a robot that has the shape of an AVL tree. The pieces used to build the robot have a value, and players must place them on the proper position to keep the tree balanced (Figure 3.17). In the final level, Mario and his robot must fight Bowser. Bowser will try to stop Mario from saving the princess and will attack Mario with firebolts. Mario must dodge the firebolts and rebalance the robot every time that it loses a node (Figure 3.17).

3.3.4.1.2 Pedagogical Approach

Super Mario AVL Tree Game provides an environment where the player can apply concepts and operations of the AVL Tree data structure. Wassila and Tahar hypothesized that if a serious game is attractive, fun, stimulating, and engaging, the player will learn. Concerning the game genre, Wassila and Tahar selected the action genre because action games facilitate the mastery of concepts and skills (Frété, 2002). Furthermore, this genre allows the application of learning techniques such as drill and practice, behaviourist systems, learning by discovery, and learning by doing. Finally, we deduced that the player must employ the *apply* cognitive process, and the game teaches procedural knowledge.

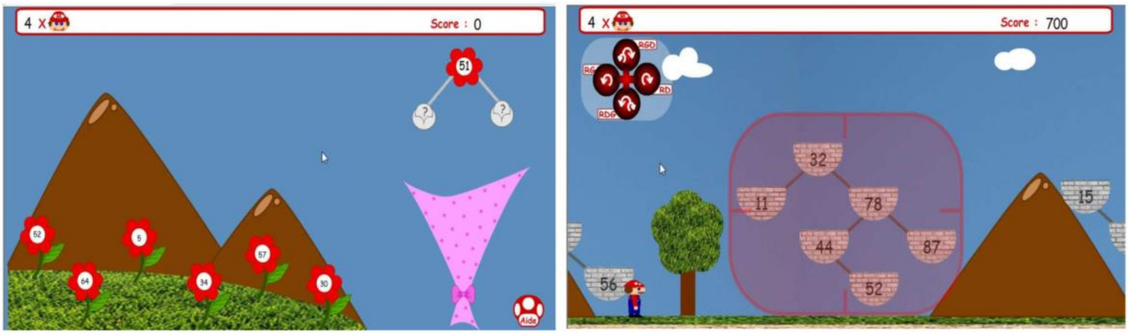


Figure 3.15: Screenshot of the first and second level of *Super Mario AVL Tree Game*.

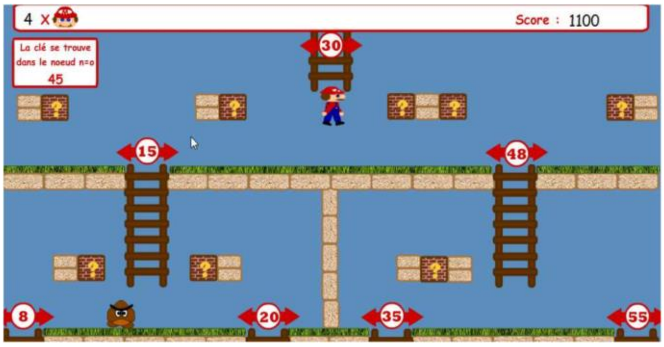


Figure 3.16: Screenshot of the third level of *Super Mario AVL Tree Game*.

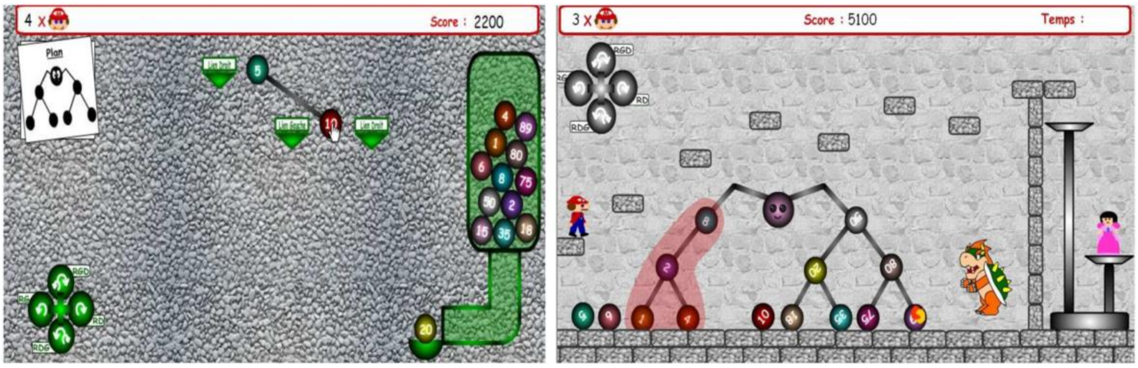


Figure 3.17: Screenshot of the fourth and fifth level of *Super Mario AVL Tree Game*.

3.3.4.1.3 Analysis

Our results showed that *Super Mario AVL Tree Game* combines elements of the action and puzzle genre. For example, in the final level, players must dodge Bowser’s attacks while they rebalance an AVL tree. In the third level, players must find a key using the BST property while avoiding enemies. These combinations may keep the player engaged and motivated, and at the same time, players may spend more time learning about AVL trees.

Another interesting finding is that Wassila and Tahar explained the pedagogical benefits of the action genre. However, based on the game description, three of the five levels (levels 1, 2 and 4) are puzzles, and the researchers paid little attention to this genre in their design. Concerning the pedagogical aspects, Wassila and Tahar did not report a learning theory, the learning objectives, or the learning activities. As stated in the previous discussions of stack games, omitting these aspects may make it difficult to assess the learning gains and the game itself.

3.3.4.2 AVL Trees Game

3.3.4.2.1 Description

AVL Trees Game (Šuníková et al., 2018) is a 2D puzzle game for mobile devices, which provides an environment where students can practice the AVL tree insert algorithm (Figure 3.18). The game does not provide theoretical content, and according to Šuníková et al., *AVL Trees Game* is intended to complement traditional teaching methods.

Concerning the game design, *AVL Trees Game* divides the insert algorithm into several steps and dedicates a level to each step. In addition, each level has several instances (sublevels) that allow players to practice and internalise the steps of the algorithm. In the first level, players must insert a node with a given key in the right position in a BST. In the second level, players must mark the root node of an unbalanced subtree. In the final level, players must insert a node, mark all nodes that need to be rotated, and perform the AVL tree balancing rotations. Additionally, the game has a tutorial that explains the basic rules and game mechanics.

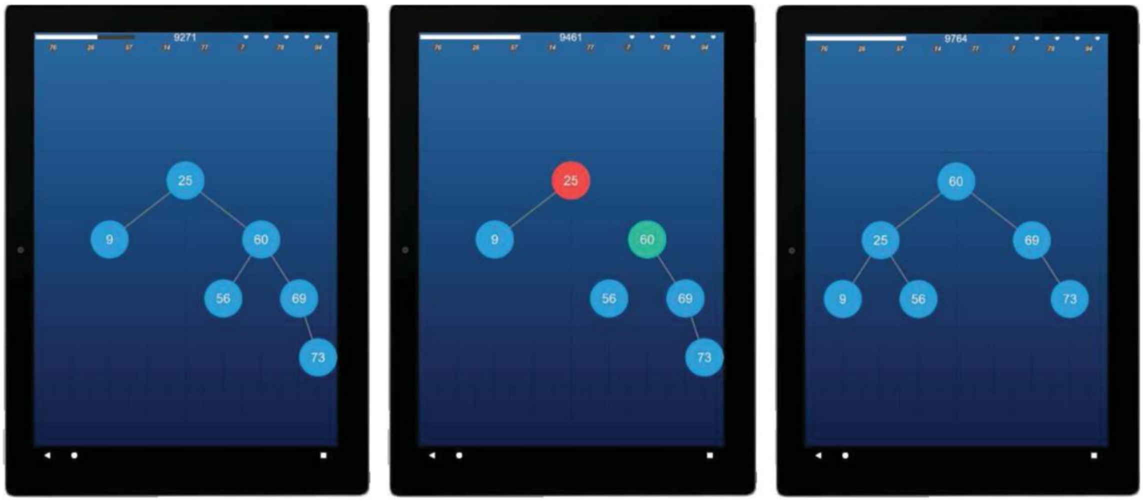


Figure 3.18: Screenshot of the third level of *AVL Trees Game*.

3.3.4.2.2 Pedagogical Approach

According to Šuníková et al. (2018), video games are an effective learning tool because they provide a quick cycle of hypothesis formulation, testing, and revision. This cycle is the result of the immediate feedback provided by the game. Additionally, Šuníková and colleagues suggested that video games may provide a meaningful context for learning activities.

Furthermore, Šuníková et al. explained the importance of the game's levels for the learning process and the players' motivation. First, they stated that the game's levels allow players to acquire knowledge gradually. Knowledge built in previous levels will be necessary to build new knowledge using the information provided in subsequent levels. Second, Šuníková et al. suggested that the game's levels may increase motivation. After completing a level, players will experience a feeling of success that may motivate them to keep playing.

Finally, we deduced that the type of knowledge taught by the game is procedural, and players must employ the *apply* cognitive process to solve the game challenges.

3.3.4.2.3 Evaluation Aspects

Šuníková et al. (2018) performed an "ad-hoc" evaluation of the game where five students enrolled in an Algorithm and Data Structure course at Comenius University participated. The evaluation consisted of three phases. First, the teacher briefly explained the AVL tree theory and the insert operations. Second, the teacher explained the game and the steps of the insert operation. Finally, the participants played the game. During the third phase, the teacher guided those students who required extra help and observed the participants' behaviours towards the game. Additionally, three participants answered a qualitative survey.

Šuníková et al. concluded that the participants engaged actively and enjoyed the game. Additionally, some students showed a competitive behaviour and other students a collaborative behaviour. Finally, the survey results showed that the participants enjoyed the game and perceived that they learned about the insert algorithm.

3.3.4.2.4 Analysis

Results showed that *AVL Trees Game* provides considerable amount of practice in relation to the AVL insert algorithm. The game possesses many sublevels that may engage and motivate players to keep playing. However, we found that *AVL Trees Game* lacks other game elements that may increase the player's intrinsic motivation. According to Malone (1980), a game that increases the player's intrinsic motivation should possess an interesting challenge, encourage players' curiosity, and have fantasy elements. *AVL Trees Game* possesses interesting challenges, but it does not have elements that provoke players' curiosity, and it lacks fantasy elements. For example, the game does not have narrative and the aesthetics resembles a chart.

Concerning pedagogical aspects, we found that Šuníková et al. did not explicitly report the learning objectives. However, they did provide an explanation of how players learn while playing *AVL Trees Game*. From the description, we deduced that Šuníková et al. are using a constructivist theory, such as Kolb's Experiential Learning Theory.

Regarding evaluation aspects, we found that Šuníková et al. did not explain the factors evaluated. Additionally, they did not report a protocol for the observations, and they did not explain the methods used to analyse data. Furthermore, the sample size was small (five participants). Results obtained from this evaluation were not conclusive.

3.3.5 Recursive Algorithms Games

A recursive method is a method that in its definition, calls itself. The method must possess a base case that defines when to stop the execution of the recursion. Recursive calls must deal with subproblems smaller than the original problem, and these subproblems should not overlap (Sedgewick & Wayne, 2014). In this section, we present an analysis of three games that focus on recursive algorithms.

3.3.5.1 Elemental: The Recurrence

3.3.5.1.1 Description

Elemental: The Recurrence (Chaffin et al., 2009) is a coding-puzzle game that teaches the binary tree's depth-first search (DFS) recursive algorithm (Figure 3.19). The game has a code editor and an interpreter that allows the player to write, compile and run C++ like code. To solve the puzzles, players must control their avatar using programs that they must code. *Elemental: The Recurrence* was developed using a game engine called DarkWynter, which is an open-source engine developed in the University of North Carolina at Charlotte (UNCC) using XNA Game Studio 2.0 and C# Code Dom.

Regarding the game design, *Elemental: The Recurrence* possesses three characters: Ele who is the player's avatar, Thoughts who reminds players important aspects of the game objectives, and Cera who is a mentor who gives the instructions relating to the level and the theory covered.

The game has three levels, and in each level, players must solve a programming puzzle. In the first level, players must complete and compile a "hello word" program. This level introduces the game interface, the game mechanics, and explains the plot. In the second level, players must complete the depth-first search algorithm code for the left node from partially finished pre-written code, and in the third level, players must write the whole depth-first search algorithm.

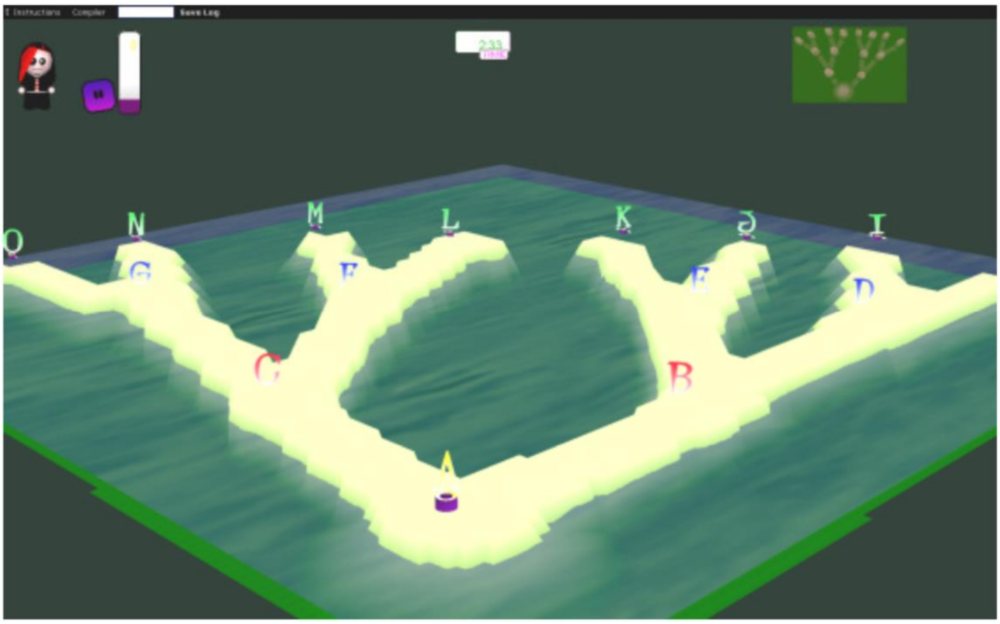


Figure 3.19: Screenshot of the second level of *Elemental: The Recurrence*.

3.3.5.1.2 Pedagogical Approach

Elemental: The Recurrence (Chaffin et al., 2009) employs the scaffolding code pedagogical approach to support learning. As mentioned before (section 3.3.1.1.2), scaffolding code is pre-written code provided by the instructor or learning activity to help students get started. Additionally, the game uses metaphors to facilitate the understanding of abstract concepts. In educational environments, analogies and metaphors are used to create relations between non-intuitive concepts and familiar concepts (Duit, 1991). Finally, the types of knowledge taught by the game are conceptual and procedural, and the player must employ the *understand*, *apply*, and *analyse* cognitive processes.

3.3.5.1.3 Evaluation Aspects

Chaffin et al. (2009) performed an evaluation to measure enjoyability, perceived learning, student's preference, and learning gains after playing the game. The evaluation consisted of two quasi-experiments. In the first evaluation, participants answered a pre-test and a post-test. In the second, participants only answered a post-test. In total, 43 students from the 2008-2009 Data Structures and Algorithms course at UNCC participated in the experiments. Specifically, 16 participated in the pre-test post-test experiment and 27 in the post-test experiment, but only 42 participants completed the qualitative survey.

At the beginning of the first experiment (pre-test and post-test), participants filled a consent form, a demographic survey, and a pre-test about recursion concepts. Then, participants played the game for 40 minutes. Finally, participants answered the post-test and a qualitative survey.

The second evaluation (post-test only) had the same structure as the first evaluation, but participants did not complete the pre-test.

Data collected using the tests was analysed using descriptive and inferential statistic methods. Data collected with the qualitative survey was analysed using descriptive statistic methods. The descriptive methods used were percentages, means, standard deviations and histograms. The inferential methods were t-test and Cohen's d coefficient.

T-test results showed that the difference between the results from the pre-test and post-test of the first evaluation was statistically significant. Additionally, results showed that there was not a statistically significant difference between the scores of the post-test of the first and second evaluation. This finding showed that the pre-test did not affect the results of the post-test.

Concerning the qualitative survey, results showed that most of the participants (81%) enjoyed the game and its mechanics and perceived learning gains. Additionally, most of the participants (74%) stated that they preferred playing games for learning (e.g., *Elemental: The Recurrence*) over programming a traditional coding assignment.

3.3.5.1.4 Analysis

We found that *Elemental: The Recurrence* employs metaphors and scaffolding code to support learning of conceptual and procedural knowledge, and both methods are well integrated into the game elements. For example, the game employs metaphors to teach recursion, and these metaphors are integrated into the game narrative and presented through the dialogues between the player's avatar and the non-playable characters. This approach may increase players' engagement because concepts covered will be taught as part of the game story, and the pace of the game will not be interrupted to teach a new concept. Regarding the scaffolding code, it is integrated in the game challenges and is used to control the levels' degree of difficulty. For example, in the first level, coding challenges provide a large amount of pre-written code for scaffolding, but as players start mastering the algorithm, advanced levels reduce or completely remove the pre-written code. In other words, levels increase in difficult as they decrease in the amount of scaffolding code provided. A proper integration of pedagogical methods in the game elements may increase players' engagement and learning gains.

Our results showed that even though Chaffin et al. reported pedagogical methods, they did not report a learning theory that explains how the player learns through the game. Additionally, they did not explicitly report the learning objectives and the learning activities. Omitting these aspects may make it difficult to assess the learning gains.

Concerning evaluation aspects, we found that Chaffin and colleagues performed an “abbreviated Solomon” and verified that the pre-test did not have any learning effect in the group that answered it. Regarding data analysis, we found two issues. First, Chaffin and colleagues analysed ordinal data as interval data. To treat data collected from a test as interval data, it is necessary to validate and calibrate it using psychometric methods, such as Item Response Theory. Second, they performed t-test and Cohen’s d parametric methods. However, they did not verify the parametric data assumptions. To solve these issues, Chaffin et al. should employ non-parametric methods or just use descriptive statistics. The sample was not large enough to obtain conclusive results. This means that the results obtained by Chaffin et al. may guide subsequent research, but they cannot be used to provide generalizations or solid affirmations about the use of games for teaching data structures.

3.3.5.2 HTML5 Hanoi Tower

3.3.5.2.1 Description

HTML5 Hanoi Tower (Vasić et al., 2014) is an online 2D puzzle game that teaches recursion, stack concepts, and the LIFO principle (Figure 3.20). The video game was developed using HTML5, JavaScript, and SCORM standards. Due to its characteristics, the game can be installed in Learning Management Systems (LMS).

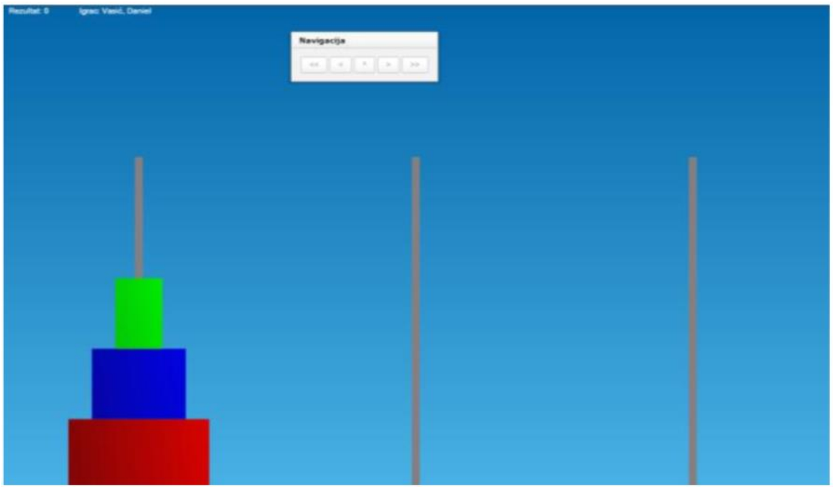


Figure 3.20: Screenshot of the first level of *HTML5 Hanoi Tower*.

HTML5 Hanoi Tower is based on the Tower of Hanoi puzzle, and its final goal is to move all disks from the initial rod to the final rod. To move a disk, the player must click over the source rod and then click over the destination rod. The difficulty of each level is defined by the number of disks on the initial rod. For example, in the first level, players must transfer three disks; in the second level, players must transfer four disks, and in the third level five disks. Additionally, the game possesses a score system: players will get ten points for each correct movement and lose twenty for each incorrect movement.

3.3.5.2.2 Pedagogical Approach

Vasić et al. (2014) justified the use of games based on the premise that serious games may increase the student motivation, provide immediate feedback, and support the development, consolidation and application of skill. Even though Vasić and colleagues did not report learning objectives, we deduced them: (1) apply the recursion to solve the Hanoi Tower puzzle; and (2) apply the LIFO rule to solve the Hanoi Tower puzzle. To solve the game challenges, players must employ the *apply* cognitive process, and type of knowledge taught by the game is procedural.

3.3.5.2.3 Evaluation Aspects

Vasić et al. (2014) performed a non-experimental study to measure learning gains and evaluate gaming experiences. The authors used a qualitative survey to evaluate the user experience and a test to measure the learning gains. The evaluation had a pre-test post-test structure without a control group. Participants were asked to complete a pre-test, then play the game, and finally take a post-test and the survey. The test had nine questions, and the type of questions were multiple and single selection. The evaluators asked to 105 students to participate in the evaluation; however, only seventeen students completed the tests and the survey. For the analysis, Vasić et al. used descriptive statistics (histograms, means and standard deviations) and inferential statistics (T-test and Cohen’s d). Finally, Vasić et al. concluded that students understand recursion after playing the game.

3.3.5.2.4 Analysis

Concerning the game design, we found that *HTML5 Hanoi Tower* was built in a way that facilitates its distribution, for example through its use of SCORM standards. This allows the game to be installed on Learning Management Systems, such as Moodle and Blackboard Learn. This feature facilitates the distribution of the game because currently, most of the higher education institutions have LMSs.

In relation to the game elements, the game needs to be improved. *HTML5 Hanoi Tower* lacks attractive aesthetics, narrative elements, and engaging mechanics and challenges. According to Malone (1980), the lack of these game elements may decrease the player’s intrinsic motivation.

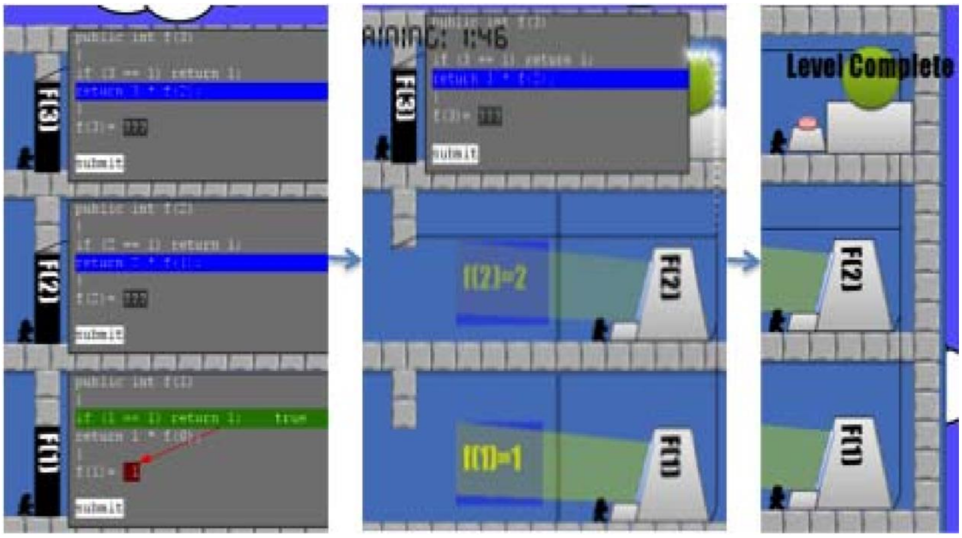
Concerning pedagogical aspects, Vasić et al. stated that students have problems to relate recursion and the Tower of Hanoi puzzle. Also, they stated that the game helps students to discover this relation. However, they did not explain how *HTML5 Hanoi Tower* helps to achieve this objective. Additionally, we found that Vasić and colleagues did not report any learning theory to explain how player acquire knowledge through the game.

Regarding evaluation aspects, we found two issues. First, Vasić and colleagues analysed ordinal data as interval data. Data obtained from a non-calibrated test cannot be used as interval data. Second, Vasić and colleagues utilized parametric methods to analyse data. However, they did not verify the parametric assumptions.

3.3.5.3 Recursive Runner

3.3.5.3.1 Description

Recursive Runner (Zhang et al., 2014) is a 2D puzzle game developed in GameMaker Studio that teaches recursion to college students (Figure 3.21). The game has two levels. The first level covers the Fibonacci recursive algorithm, and the second focuses on the Factorial recursive algorithm. According to the game story, players must stop a reactor from exploding. To achieve this objective, player should interact with the security system and shut the reactor down.



```
public int f(3)
{
if (3 == 1) return 1;
return 3 * f(2);
}
f(3)= ???
submit

public int f(2)
{
if (2 == 1) return 1;
return 2 * f(1);
}
f(2)= ???
submit

public int f(1)
{
if (1 == 1) return 1;   true
return 1 * f(0);
}
f(1)= 1
submit
```

Figure 3.21: Screenshot of the Factorial level of *Recursive Runner*.

To initiate the game, players must enter a number between three and thirty-three. The entered number *n* indicates the *nth* value of the Fibonacci or Factorial sequence, and it determines the number of floors of the game environment and the sequence value that players must find to win the game. At the beginning of the game, players appear on the top floor of the reactor. Each floor has a computer to access the security system, but a gate blocks access to the computer. To open the gate, players must reach the gate and open a coding console. The console will display the Fibonacci or Factorial recursive algorithm and will ask players to enter the *nth* value of the selected sequence. However, the system will not allow players to enter the *nth* value until they find the previous values of the sequence. Therefore, players must descend to the lower floors until finding the base cases and return those values to the upper floors.

3.3.5.3.2 Pedagogical Approach

According to Zhang et al. (2014), *Recursive Runner* aims to provide a fun environment that allows college students to visualize recursive algorithms and increase engagement. The type of knowledge taught by the game is procedural, and the player must employ the *apply* cognitive process to solve the challenges.

3.3.5.3.3 Evaluation Aspects

Zhang and colleagues performed a game evaluation, which consisted of a pre-test post-test quasi-experiment, a qualitative survey, and five informal interviews. Thirty-one college students from the Computer Science School of the Winston-Salen State University participated in the evaluation. The experiment was divided in two sessions. In the first session, participants answered a pre-test and performed a learning activity that included playing *Recursive Runner*. The second session took place few days later. Participants answered the post-test, the qualitative survey, and five of them were interviewed.

Concerning the data collection tools, the pre-test and post-test measured learning gains obtained by students after playing the game. Both tests possessed three questions about the Fibonacci recursive algorithm and three questions about the Factorial recursive algorithm. The qualitative survey had nine five-point Likert-scale questions regarding participants' game experiences: enjoyability, perceived learning, motivation, and preference. The interviews were informal, and Zhang and colleagues did not give details about their protocol, execution, or content.

Data was analysed using descriptive statistic methods (averages, medians, standard deviation, variance, and histograms). Results showed that students performed better in the post-test after playing the game, and participants reported that they enjoyed the game and perceived that they learned.

3.3.5.3.4 Analysis

We found that Zhang et al. did not report a learning theory or pedagogical methods to deliver the educational content. However, we noticed that *Recursive Runner* uses analogies to represent “the call stack” that occurs during the execution of a recursive algorithm. The analogy works as follows: In the game, each floor represents a value of the Factorial or Fibonacci sequence. In each floor, players must calculate and write the corresponding value of the sequence, but they do not have the two previous values of the sequence. Consequently, players must descend to the previous levels until they find the base cases. Then, after finding the base cases, players will be able to perform the calculations to find the current value and ascend to the upper levels until the final value of the sequence. This approach may facilitate the understanding of recursion through a game challenge.

Concerning learning objectives and activities, we found that Zhang et al. did not report these aspects. The omission of this information may cause problems regarding the assessment of the learning gains and evaluating the game. Regarding the evaluation, we found that Zhang used descriptive statistics to analyse data. This approach is simple, but it is appropriate for the data collected. However, they did not explain the factors evaluated in the qualitative survey.

3.3.6 Coding Games

We define coding games as digital games that are controlled through a program previously coded by the player instead of physical controller (e.g., joystick, keyboard, or mouse). These games require that the player possesses advanced programming skills, and their main aim is to improve the player’s coding skill and introduce high-level topics. However, some coding games also allows players to learn and practice data structure concepts. In this section, we present an analysis of two coding games that also cover arrays and tree data structures.

3.3.6.1 Critical Mass

3.3.6.1.1 Description

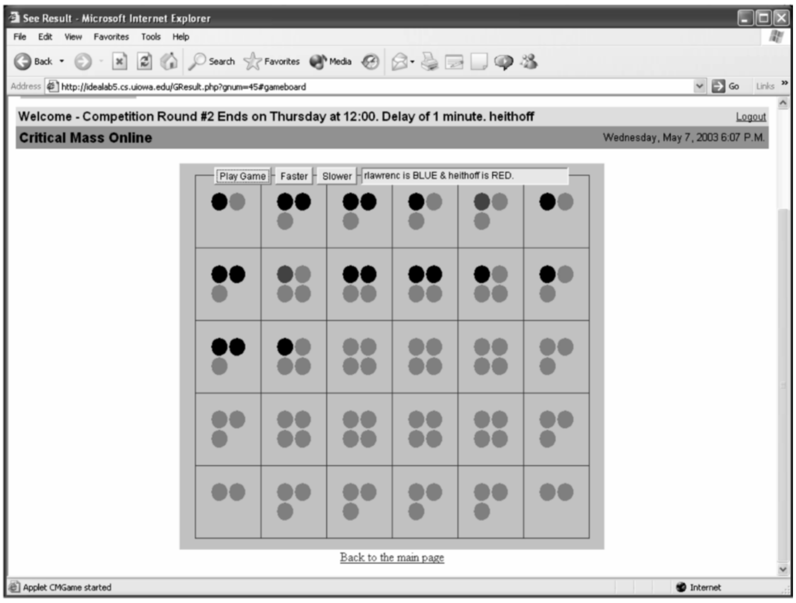
Critical Mass (Lawrence, 2004) is a coding turn-based strategy game designed to be played by artificial intelligence algorithm implemented by college students. The game matches happen in an arena-like website where students can upload their A.I. programs to compete against algorithms provided by instructors and other students. *Critical Mass* was developed in Java and C++ , and the website was built using open-source software: Red Hat Linux server, Apache 1.3, MySQL 3.23, PHP , Java, and HTML.

The rules of *Critical Mass* are the following:

> *Critical Mass* uses a 5x6 board of squares. The object of the game is to remove completely all of the opponent’s pieces from the game board. Each cell may contain

> zero or more pieces. The pieces are of two different colours, one belonging to each player. All pieces in a cell are always of the same colour. The two players alternate moves. To make a move, a player must place a piece of his or her own colour into any cell, which does not contain a piece of the opponent's colour. If the number of pieces in a cell becomes greater than or equal to the number of adjacent cells, then that cell explodes. When a cell explodes, the pieces in that cell are removed, one additional piece is added to each adjacent cell, and all pieces in adjacent cells became the colour of the player whose move caused the explosion. Explosions may cause subsequent explosions. If an explosion causes additional explosions, then the new explosions happen simultaneously, not by means of a wavelike chain reaction. The game ends when, after any move except the first, all pieces on the board are the same colour. The player corresponding to that colour wins. (Lawrence, 2004, p. 2)

To play *Critical Mass*, students must implement a board evaluator, a move generator, and a game tree. According to Lawrence, developing the board evaluator requires that students understand how to use two-dimensional arrays and cost functions based on piece location. The move generator requires students to code functions to determine valid movements. Additionally, this requires an understanding of conditional structures and the game logic. Finally, coding the game tree requires an understanding of trees and recursion. After coding the A.I. programs, students must upload their code to the game website. The website acts as an arena where students can challenge other students' code. If a student challenges another student, the website will compile a program that allows both students' code to play *Critical Mass* against each other. The student who wins the game takes a higher position in the overall rank.



See Result - Microsoft Internet Explorer

Address http://idealab5.cs.uiowa.edu/GResult.php?gnum=45#gameboard

Welcome - Competition Round #2 Ends on Thursday at 12:00. Delay of 1 minute. heithoff

Critical Mass Online

Wednesday, May 7, 2003 6:07 P.M.

rlawrenc is BLUE & heithoff is RED.

Applet CMGame started

Figure 3.22: Students view of a game result using a Java applet.

3.3.6.1.2 Pedagogical Approach

According to Lawrence (2004), competitive programming and tournament-like assignments are a source of motivation and decrease procrastination. In their pursuit to beat other competitors, students must improve and evaluate their coding skills and programs. Consequently, students will practise more, and therefore, they will learn more. Additionally, Lawrence suggested that competition may enhance students' interaction, but pointed out that it may not suit all types of students. Finally, we deduced that the type of knowledge taught by the game is procedural, and players must apply all the six Bloom's taxonomy cognitive processes to solve the game challenges.

3.3.6.1.3 Evaluation Aspects

Lawrence performed an "ad-hoc" evaluation that consisted of a class assignment and a qualitative survey. In the assignment, students had to code a board evaluator, a movement generator, and a game tree. Then, they had to upload their code to the game website and compete against other students' code. Students had three weeks to submit their code. In total, 55 students participated in the assignment, but only 42 completed the assignment. After submitting the assignment, participants answered a qualitative survey. The survey was a five-point Likert-scale questionnaire, and it evaluated topics such as preference and perceived learning. The data collected was analysed using descriptive statistical methods (percentages). Results show that the students liked the competitive coding assignment more than traditional assignments. Finally, the students perceived that they learned.

3.3.6.1.4 Analysis

Our review showed that *Critical Mass* was based on *Chess* and *Go*, and like these games, the *Critical Mass* rules are simple. However, the space of possibility is wide and allows the player to apply different strategies and game styles. Therefore, *Critical Mass* may be easy to learn, but it is a difficult game to master. Additionally, to play and master the game, it is necessary to possess strong programming skills. Players must code the A.I. algorithm that plays the game. These features increase the difficulty of the game and may decrease the motivation of students. It is possible that this explains why thirteen students did not submit their assignment.

Concerning pedagogical aspects, we found that Lawrence did not report a learning theory, the learning objectives, or the learning activities. Omitting these aspects may impede the assessment of knowledge and the evaluation of the game. For example, Lawrence informed that the assessment of the assignment was based on the results obtained in the leader board of the tournament website instead of the knowledge of the student. This could demotivate students who know how to code but did not perform well in the competition.

Regarding the game evaluation, we found that it was an informal evaluation. Lawrence did not explain in detail the evaluated factors or did not describe the survey used to collect data. Additionally, only students who submitted that assignment answered the survey. The opinion of thirteen students who failed the assignment were excluded from the evaluation. Results obtained from this evaluation were not conclusive.

3.3.6.2 Resource Craft

3.3.6.2.1 Description

Resource Craft (Jiau et al., 2009) is a real-time strategy coding game developed to test a metric framework called Simulated Programming Learning Environment (SIMPLE). The SIMPLE framework collects information from game variables and displays them in real time. The framework aims to provide immediate feedback to players allowing them to improve their game strategy. The SIMPLE framework and *Resource Craft* were developed in Python and PyGame.

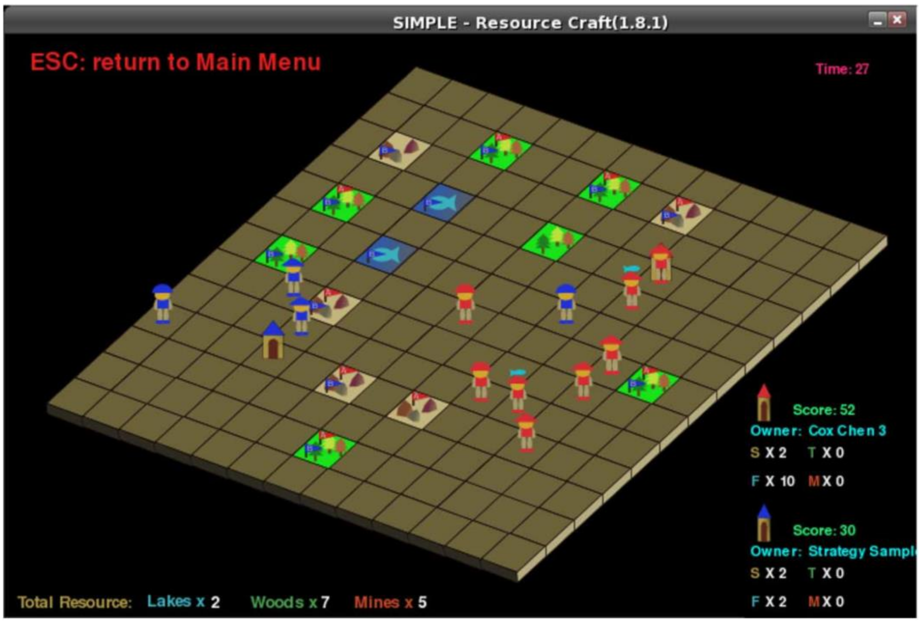


Figure 3.23: Screenshot of Resource Craft.

In *Resource Craft*, the player's objective is to obtain the highest possible score by collecting the largest quantity of resources. To accomplish this, players can control four types of units. The first one is the seeker who can explore the map and find sources of resources (e.g., lakes, forests, or mines). The second unit is the fisherman who can fish from lakes. The third is the "timberjack" who can extract wood from forests. The last type is the miner who can extract minerals from mines. The player starts the game with two seekers, two fishermen and one castle. The castle is where resources are stored and where workers are created. To play the game, players must code a strategy that ensures an efficient exploration of the map, maximises the collection of resources, and minimises the waste of resources and idle workers.

3.3.6.2.2 Pedagogical Approach

Resource Craft covers many topics related to programming, such as conditional statements, loops, arrays, and dictionaries. According to Jiau et al. (2009), the SIMPLE metric and *Resource Craft* enhance self-motivation due to the immediate feedback. The type of knowledge delivered by the game is procedural, and the player must apply all the Bloom's cognitive process to solve the game challenge.

3.3.6.2.3 Evaluation Aspects

Jiau et al. (2009) performed an "ad-hoc" evaluation of the game to assess three aspects of the SIMPLE framework. The three aspects evaluated were (1) the effect of SIMPLE on students' self-motivation; (2) differences between students who used SIMPLE and those who did not; and (3) the effect of SIMPLE on the learning gains. To evaluate learning gains of students who used SIMPLE and compare them against students who did not use the framework, Jiau et al. developed an assignment where students had to code a program to play *Resource Craft*. They divided the participants into two groups. The first group (57 participants) used a version of the game that included SIMPLE. The second group (45 participants) used a version of *Resource Craft* without the framework. To evaluate the self-motivation, students who used SIMPLE answered a six-question qualitative survey. Data collected was analysed using descriptive statistic (percentages). Results from the first evaluation showed that students who used SIMPLE performed better than students who did not. Results from the qualitative survey showed that students who used the framework had fun using the framework and felt that they were learning.

3.3.6.2.4 Analysis

Concerning pedagogical aspects, we found that Jiau et al. did not report a learning theory that explains how players learn from the game. However, they did report learning objectives and learning activities. They stated that the game improves students' self-motivation, but they did not define what they meant by self-motivation. For example, Jiau et al. aimed to examine whether the game improved the students' self-motivation. However, the questions in the survey relate to perceived learning, amount of code programmed, preference, and fun. None of those questions were directly concerned with the player's self-motivation. Omitting the learning theory, pedagogical approach, learning objective, or the evaluated factors from the research may lead to problems during the evaluation of the game and the assessment of the student's knowledge. Due to the lack of theoretical background, the evaluation may not lead to conclusive results.

3.3.7 Bundles of Data Structure Games

Bundles are collections of mini games that focus on more than one data structure or algorithm. The mini games are like arcade games with simple game mechanics and almost no narrative elements. In this section, we present our analysis of two such bundles.

3.3.7.1 Algorithm Games Prototypes

3.3.7.1.1 Description

In their work, Shabanah et al. (2010) propose a concept called "Algorithm Games." Algorithm Games are learning games that teach data structure and algorithm concepts and facilitate the visualization of data structures and algorithms. According to Shabanah and colleagues, Algorithm Games possess the following six characteristics: (1) they can be developed from scratch or can use an existing games; (2) they must be simple, but they must challenge the player; (3) they can be of any genre; (4) they must have elements that increases the player's intrinsic motivation; (5) the game graphics must depict aspects of the selected data structure; (6) finally, the gameplay must simulate the behaviour of the data structure or algorithm depicted by the game.

To facilitate the creation of Algorithm Games, Shabanah et al. implemented two game development tools called Algorithm Game Designer and Serious Algorithm Visualization Game Engine (SAVGEngine). These tools were developed using Microsoft .Net Framework, C#, and Microsoft XNA Game Studio. The Algorithm Game Designer tool is an editor that allows users to create algorithms and game-specific scripts and permits the manipulation of graphic elements. SAVGEngine is a game engine that permits the creation of 2D and 3D environments.

To test the tools' efficacy, Shabanah and colleagues developed three Algorithm Game prototypes named *Binary Search Game*, *Singly Linked List Game*, and *Binary Search Tree Game*. The *Binary Search Game* (Figure 3.24) is an arcade game like *Arkanoid* (1986). In the game, players must hit an array of blocks using a paddle and a ball following the binary search algorithm. *Linked List Game* is a puzzle game that teaches the traversal, insert, and remove algorithms. In this game, players must connect nodes to complete patterns requested by the game. Finally, *Binary Search Tree Game* is a puzzle game that teaches the search, add, and remove algorithms of the BST data structure. The game objective is to build a binary search tree with values provided by the games as fast as possible.

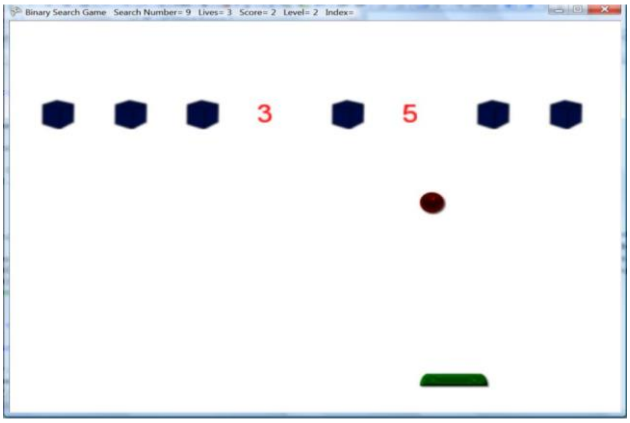

Binary Search Game   Search Number= 9   Lives= 3   Score= 2   Level= 2   Index=
3
5

Figure 3.24: Screenshot of the Binary Search Game prototype.

3.3.7.1.2 Pedagogical Approach

According to Shabanah et al. (2010), video games have several characteristics that allow them to be converted into great learning tools for teaching algorithms. First, digital games support the six types of engagement of visualization tools suggested by Naps et al. (2002). Second, digital games increase intrinsic motivation allowing the player to engage with the learning activity. Third, digital games allow the player to build their knowledge due to the immediate feedback provided by the game. From this perspective, digital games for learning follow a constructivist stance. Finally, we deduced that the type of knowledge taught by the prototypes is procedural, and players must employ the *understand* and *apply* cognitive processes to solve the challenges.

3.3.7.1.3 Analysis

The Algorithm Games concept may be useful when designing serious games that teach data structures and algorithms. For example, the advice given by Shabanah et al. suggests that designing game challenges or environments that resemble the functionality of algorithms or data structure may help to teach those abstract topics. Using familiar knowledge to explain unfamiliar concepts is a popular approach used in science education through analogies. This approach may be useful also in other type of serious games.

Concerning the tools developed (Algorithm Game Designer and SAVGEngine), creating a game engine from scratch might be unnecessary. Game engines are complex software. Coding a "good" engine requires advanced coding skills and considerable time and resources. A better idea may be to code a plugin or a library for an existing game engine, such as Panda 3D or Spring Engine. Both engines were open source and were released before this work was published.

3.3.7.2 ***Data Structure Learning Platform***

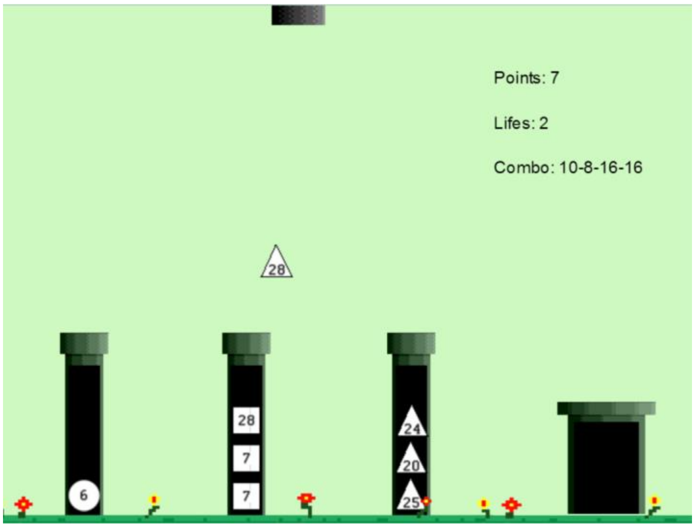
3.3.7.2.1 Description

Data Structure Learning Platform (DSLEP) is an online learning tool that facilitates the learning process of data structures by applying gamification concepts (Costa et al., 2014). According to Costa and colleagues, DSLEP is divided into three modules called the *System Core module*, the *User Interaction module,* and the *Gamification module*. The System Core module contains the data structure topics, and each topic contains one or two learning tasks that consist of two activities (a tutorial and a practice). Specifically, the System Core covers five data structures (arrays, stacks, queues, lists, and trees) and had seven activities (one array activity, two stack activities, one queue activity, one list activity, and two tree activities). The Gamification module contains all the gamification elements used by the learning tool, such as experience points, levels, leader boards, achievements, and user profiles. The User Interface module is the web interface that integrates the System Core and Gamification Modules.

DSLEP is an online learning tool that applies gamification principles. However, we considered that five of the seven activities are video games, and for this reason, we included them in our review. The first game is *Piperray* (Figure 3.25), a puzzle game like *Tetris* (1984) that teaches array concepts. In the game, players must store variables of the same type in its corresponding array. Variables fall from the top side of the screen and are represented by geometrical shapes and a numerical value. The geometrical form of the shape represents the type of the variable. Arrays are represented by pipes located in the bottom side of the screen. The player gets points by arranging the variables in the correct array and by inserting the variables with the values asked by the game.

The second game is the *Tower of Hanoi* game (Figure 3.26). The game is a puzzle game inspired by the Tower of Hanoi problem, and it focuses on the stack data structure. *Tower of Hanoi* game requires that students complete the puzzle with the lowest number of pin changes and in an average time.

The third game called *Asterostacks* (Figure 3.27) is a vertical-scrolling shooter like *Xevious* (1983). To win the game, players must collect circular items with the correct numerical value. Additionally, players can shoot the last collected item to practice the first-in first-out (FIFO) property.

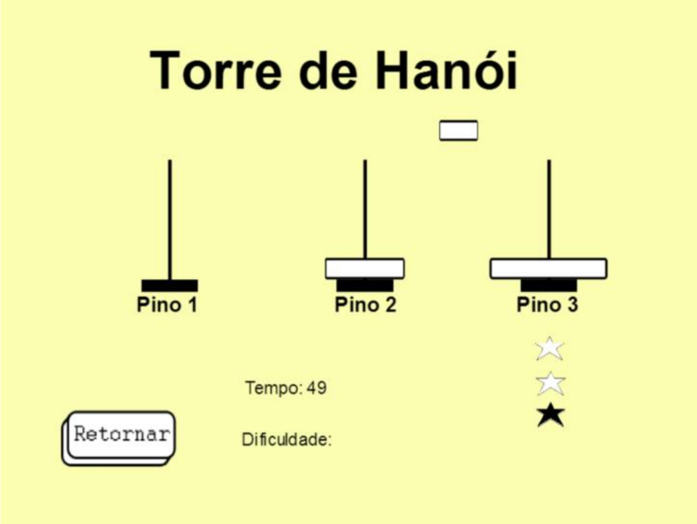


Points: 7

Lifes: 2

Combo: 10-8-16-16

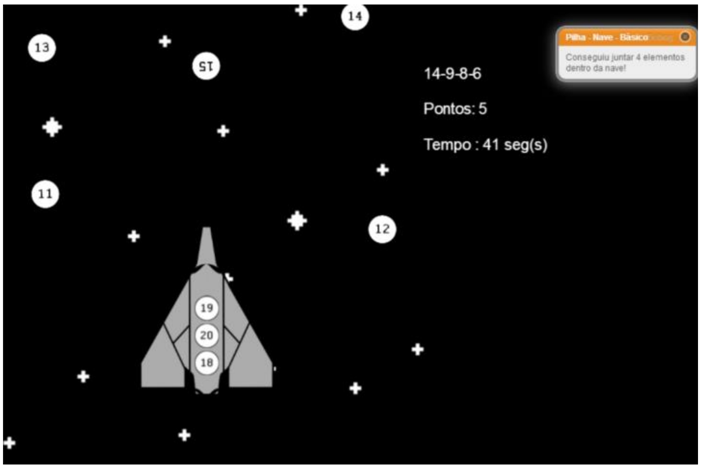
Figure 3.25: Screenshot of Piperray.



Tempo: 49

Dificuldade:

Figure 3.26: Screenshot of Tower of Hanoi game.



Pilha - Nave - Básico

Conseguiu juntar 4 elementos dentro da nave!

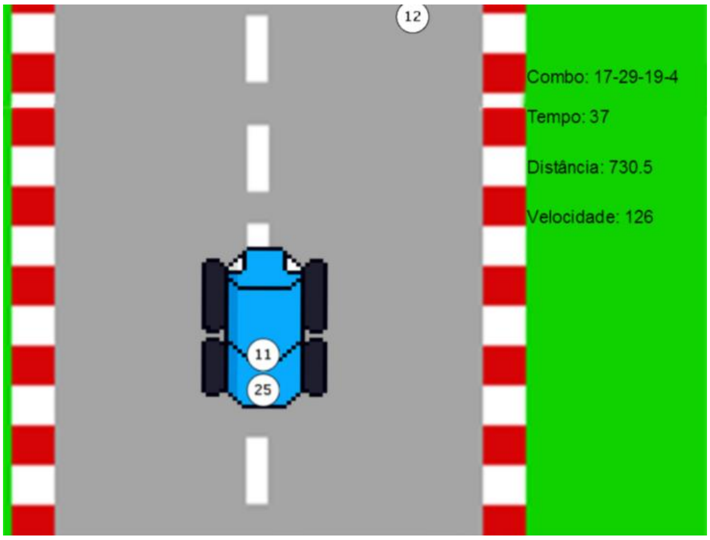
14-9-8-6

Pontos: 5

Tempo : 41 seg(s)

Figure 3.27: Screenshot of Asterostacks.

The fourth game named *Queue Race* (Figure 3.28) is a racing game like *Super Sprint* (1986), and it focuses on the queue data structure. In this game, players must collect and queue circular items that can be used as speed busters. Players can only use the last item collected following the LIFO principle.



Combo: 17-29-19-4

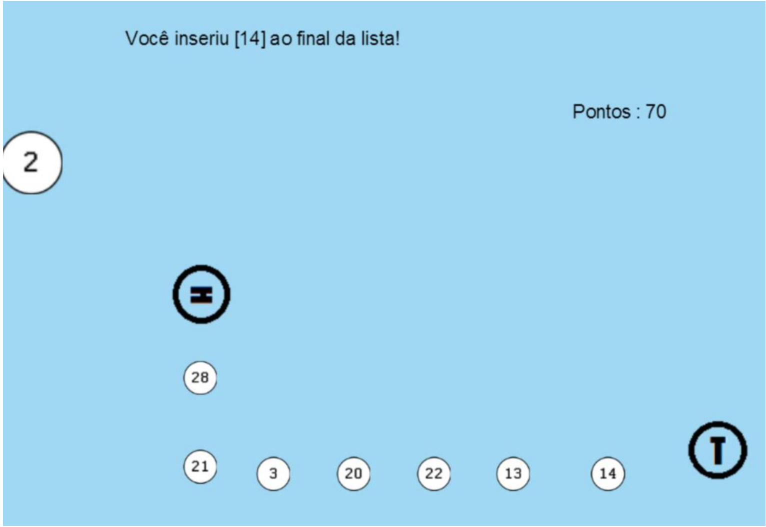
Tempo: 37

Distância: 730.5

Velocidade: 126

Figure 3.28: Screenshot of Queue Race.

The fifth game is a remake of the popular mobile game *Snake* (Figure 3.29), and it focuses on the linked list data structure. In the game, players can eat items through its head or tail, and at the end of the game, players must connect the snake's head and tail to win the game. According to Costa et al. this winning condition is a representation of the circular list.



Você inseriu [14] ao final da lista!

Pontos : 70

Figure 3.29: Screenshot of the Snake game.

3.3.7.2.2 Pedagogical Approach

According to Costa et al. (2014), DSLEP applies gamification principles to support learning. Gamification is defined as the utilization of game elements in scenarios that are not games to improve the participant’s engagement. Additionally, Pink’s motivation theory (2009) is used by the game designers to improve the user’s motivation. This theory suggests that motivation is driven by three elements: autonomy, mastery and purpose. Autonomy refers to the control that a person has over her own decision and actions. Mastery refers to the potential that a person possesses to learn and improve a skill or knowledge. Purpose refers to the meaning that a person gives to a task. Generally, if the task is important, a person will be more motivated. Finally, Costa and colleagues report that DSLEP was designed to apply the Flow theory (Csikszentmihalyi, 1990) to increase learners’ engagement. This theory states that trying to solve a problem that is beyond one's skills is a frustrating task, and on the other hand, solving problems that are quite easy is a boring task. To enhance engagement, an activity should fit the skill of the person who is performing the activity. Finally, we deduced that DSLEP teaches procedural knowledge, and players must employ the *understand* and *apply* cognitive processes.

3.3.7.2.3 Analysis

Our review showed that DSLEP is an online learning platform that teaches several data structures using gamified activities. Most of the activities of the platform are mini games that have appealing aesthetics but quite simple game mechanics. The simplicity of the games is compensated for by the variety of games that the platform provides. However, from a learning perspective, some games are so simple that they may fail to teach the intended content. For example, the challenges of *Snake* intend to represent the functionality of a circular linked list. However, it is not clear how a student will learn circular linked list concepts just by collecting items. Another example is *Piperray*. In this game, players must sort descending items with different shapes in their corresponding pipe, but it is not clear how the student will learn array concepts just by sorting items.

Concerning pedagogical aspects, we found that Costa and colleagues explained in detail the theoretical foundations of how users learn when they use DSLEP. However, they did not report the learning objectives or learning activities. The omission of these aspects may cause problems with assessing learning gains and evaluating the platform’s performance.

3.4 Overall Results and Summaries

3.4.1 Game Elements

In order to answer SRQ 1.1, we identified six game elements used to teach and represent the content. The first category includes the narrative elements, which are the components that tell the game story (e.g., dialogue, animations, text, etc.). The narrative elements can be used to convey concepts and facts related to the learning content. The second category is the game environment (game world), which consists of the elements used to build the virtual place where the events happen (Schell, 2014). The game environments are commonly used to reflect the data structure models.

The third category contains the game mechanics. The game mechanics are functions that allow interaction with the game world and are constrained by the game rules (Sicart, 2008). Game mechanics are invoked by game agents, including players and non-player characters. Usually, game mechanics are employed to represent functions that can be performed in the data structures, such as the insert and remove methods, or the rotations of the AVL trees.

The fourth category is the user interface (UI). According to Adams (2014), the UI is “the collection of presentation elements and control elements that mediate between the player in the real world and the game world” (p. 525). For this analysis, the controllers (e.g., joysticks, mouse, or scripts) and graphical elements (e.g., menus, information cards, maps, or icons) are considered UI elements. These elements provide information about the game world or player avatar but are non-diegetic, i.e., they do not belong to the game world.

The fifth category includes the game challenges. Game challenges are tasks that the player needs to perform to achieve the game’s goals or winning condition. These task must require a certain amount of effort from the player and should not be trivial (Adams, 2014). Examples of challenges are logic, conceptual, memory, time pressure, management, exploration, or physical coordination challenges. In learning games, challenges embody the learning activities.

The sixth category consists of the player avatar. The avatar is “a fictional character in a game with whom the player identifies as the personification of herself with the game world” (Adams, 2014, p. 503). The avatar can be a humanoid character, animal, or vehicle. Due to their versatility, some game avatars have been used to represent data structures.

Table 3.3 presents each game reviewed and the game elements used to teach and represent the data structures covered.

Table 3.3: Game elements used to teach the learning contents.

Game	Narrative Elements	Game Environment	Game Mechanics	Game Challenges	User Interface	Avatar
Wu’s Castle	X	X		X		
Star Chef		X	X			
Stacks and Queues		X	X			
Stack Em Up		X	X	X		
Stack Game		X	X	X	X	
Ramle’s Stack Game		X	X			
Space Traveller		X	X			X
La Petite Fee Cosmo		X		X		
Super Mario AVL Tree Game		X	X	X		
AVL Tree Game		X	X	X		
Elemental: The Recurrence	X	X		X	X	
HTML5 Hanoi Tower		X	X	X		
Recursive Runner		X		X		
Critical Mass					X	
Resource Craft					X	
Prototypes Bundle*		X	X	X		
DSLEP Bundle**		X	X	X	X	X

* The prototype bundle includes the following games: *Binary Search Game*, *Singly Linked List Game*, and *Binary Search Tree Game*.
** This bundle includes the following games: *Piperray*, *Hanoi Tower*, *Asterostacks*, *Queue Race*, and *Snake*.

3.4.2 Topic

With regard to SRQ 1.2, we identified nine data structures (array, 2D array, stack, queue, linked list, dictionary, tree, binary tree and the AVL tree) and six recursive algorithms (Hanoi Tower recursive algorithm, tree traversal, binary search, deep-first search, Fibonacci and Factorial). Eight games were found to focus on a single data structure or algorithm while seven games were found to cover more than one topic. Regarding the bundles, each mini game was found to focus on a single data structure or algorithm. Table 2.4 summarizes information extracted from the articles regarding data structures and recursive algorithms covered in each game or bundle. In the table, the name, associated reference, and data structures or algorithms for each game are presented. As mentioned above, all recursion algorithms were aggregated in a single column. A grey box with an “X” indicates the primary topic covered, while a white box with an “X” indicates a secondary topic by a game.

Table 3.4: Topics covered by the reviewed games.

Game	Array	2D Array	Stack	Queue	Linked List	Dictionary	Tree	Binary Tree	AVL Tree	Recursive Algorithms***
Wu’s Castle	X	X								
Star Chef			X	X						X
Stacks and Queues			X	X						
Stack Em Up			X							
Stack Game			X							
Ramle’s Stack Game			X							
Space Traveller					X					
La Petite Fee Cosmo					X					
Super Mario AVL Tree Game								X	X	
AVL Tree Game									X	
Elemental: The Recurrence								X		X
HTML5 Hanoi Tower			X							X
Recursive Runner										X
Critical Mass		X					X			X
Resource Craft	X					X				
Prototypes Bundle*					X			X		X
DSLEP Bundle**	X		X	X	X					

* The prototype bundle includes the following games: *Binary Search Game*, *Singly Linked List Game*, and *Binary Search Tree Game*.
** This bundle includes the following games: *Piperray*, *Hanoi Tower*, *Asterostacks*, *Queue Race*, and *Snake*.
*** The recursive algorithms are tree traversal, binary search, deep-first search, Fibonacci, and Factorial.

3.4.3 Theoretical Foundations

In relation to the learning theories and principles (SRQ 1.3), eleven games/bundles (65% of the reviewed games) reported one or more learning theories or principles that support learning while playing the game. In total, we found eleven theories/principles: immediate feedback (Gee, 2007), Pink’s Motivation Theory (Pink, 2009), gamification (Darina Dicheva et al., 2015), intrinsic motivation (Touré-Tillery & Fishbach, 2014), motivation, analogies and metaphors (Duit, 1991), productive failure (Kapur, 2014), learning by doing (Bruce & Bloch, 2012), the Flow (Csikszentmihalyi, 1990), scaffolding (Zydney, 2012) and constructivism (Gogus, 2012). Constructivism theories and principles were the most widely used (nine of seventeen). The third column in Table 4 lists the learning theories/principles used by each game.

Concerning cognitive processes (SRQ 1.4), only *Stack Game* was found to explicitly report this aspect. Consequently, we deduced the cognitive processes of the rest of the games based on the descriptions in the articles. It was found that in fourteen of the seventeen games, players must employ the *apply* cognitive process. Additionally, we found that in four of the five minigames of the *DSLEP Bundle* as well as *Star Chef*, *Stacks and Queues*, and *Ramle's Stack Game*, players must employ the *remember* cognitive process. These games are simpler and therefore only require players to remember facts about the relevant data structure or algorithm (e.g., the stack follows the LIFO principle) in order to solve the game challenges. In contrast, *Stack Game*, *Space Traveller* and *Elemental* include coding challenges, which require players to write well-known algorithms (e.g., depth-first-search or the linked list insert and remove algorithms). Therefore, to achieve the objectives for this set of games, players must employ the *understand*, *apply*, and *analyse* cognitive processes. Finally, *Critical Mass* and *Resource Craft* were found to involve the widest range of cognitive processes. In these advanced coding games, players are required to code a program capable of playing the game. This involves creating an original program which the player must then evaluate and optimize, taking into account the results given by the game system. Consequently, the player must use all the cognitive processes listed in Bloom's revised taxonomy.

Concerning the knowledge dimension (SRQ 1.5), results showed that fourteen games deliver procedural knowledge with only three games and four mini games of the *DSLEP Bundle* delivering factual knowledge. The challenges of these games require that the player only remember certain facts or principles of the data structure. However, some games were found to deliver both factual and conceptual knowledge. For example, in certain games, the game story or in-game messages provided players with conceptual and factual information that they could use to solve challenges of the game. Finally, three games were found to deliver factual, conceptual, and procedural knowledge.

Table 2.5 summarizes the learning game genre, theories/principles, dimensions of knowledge and cognitive processes associated with each game. The term NI (not included) is used to note cases where articles did not report a learning theory or principle.

Table 3.5: Summary of the learning aspects and the game genre.

Game	Game Genre	Learning Theory/Principle and Pedagogical Approach *	Type of Knowledge **	Remember	Understand	Apply	Analyse	Evaluate	Create
Wu’s Castle	Puzzle-Coding	Immediate feedback Scaffolding code	P		X	X	X		
Star Chef	Casual	NI	P			X			
Stacks and Queues	Casual	Constructivism	F	X					
Stack Em Up	Puzzle	Constructivism The Flow	C/P			X			
Stack Game	Action-Adventure Coding	Constructivism Learning-by-doing	F/C/P		X	X	X		X
Ramle’s Stack Game	Puzzle	Scaffolding	F	X					
Space Traveller	Action-Coding	NI	P		X	X	X		X
La Petite Fee Cosmo	Puzzle	Productive failure	P			X			
Super Mario AVL Tree Game	Action-Puzzle	NI	P			X			
AVL Tree Game	Puzzle	Constructivism (game-based learning principles) Scaffolding	P			X			
Elemental: The Recurrence	Adventure-Coding	Scaffolding code Analogies and metaphors	F/C/P		X	X	X		X
HTML5 Hanoi Tower	Puzzle	NI	P			X			
Recursive Runner	Puzzle	NI	P			X			
Critical Mass	Coding	Motivation (competition)	P	X	X	X	X	X	X
Resource Craft	Coding	NI	P	X	X	X	X	X	X
Prototypes Bundle	Action	Constructivism Intrinsic motivation	P		X	X			
DSLEP Bundle	Action	Gamification Pink’s motivation theory The Flow	P		X	X			

* NI means “not included”.
** The types of knowledge are Factual (F), Conceptual (C), and Procedural (P).

3.4.4 Evaluation Aspects

Twelve of the seventeen articles reviewed included a game evaluation. All evaluations involved users and intended to measure users' abilities or opinions about the game.

Concerning the evaluated factors (SRQ 1.6), we identified fifteen factors, which are listed in the second column of Table 5. It was found that nearly all studies (eleven of thirteen) evaluated more than one factor. The factor classification framework suggested by Petri and von Wangenheim (2017) was used to classify the factors into seven categories. The factors identified most commonly fell under the learning, UX, and usefulness categories. Studies that evaluated perceived learning were classified under the usability category.

Regarding the research design (SRQ 1.7), seven studies were classified as quasi-experiments, three studies as ad-hoc, three as non-experimental, and only one as experimental. The third column of Table 5 lists the research design used by each game.

Concerning data collection tools (SRQ 1.8), thirteen evaluations were found to use a qualitative survey, seven a test or questionnaire, two an observation method, one an assignment, and one an interview. It is important to note that with the exception of the evaluation of *Star Chef*, which used the TAM scale (Davis et al., 1989), all studies analysed utilized an informal instrument (an instrument that was not validated or calibrated) to evaluate game factors. The fourth column of Table 5 lists the instruments used by each game.

In terms of data analysis methods (SRQ 1.9), twelve evaluations were found to use descriptive statistics (means, variance, standard deviations, histograms, and percentages), while only four were found to use inferential statistics (t-test and ANOVA). Finally, two studies were found to employ informal methods to analyse the qualitative data collected from interviews and observations. The fifth column of Table 2.6 presents the methods used in each evaluation.

Table 3.6: Evaluation aspects.

Game	Evaluated Factors	Evaluation Design (N° participants)	Instruments	Analysis Methods
Wu's Castle	Learning, enjoyability, preference, perceived learning, motivation, usability.	Quasi-experiment (27), Experiment (55)	Test, Qualitative survey	DS: percentages, averages. IS: t-test
Star Chef	Technology acceptance (usefulness, easiness, and attitude towards the tool)	Quasi-experiment (110)	Qualitative survey	DS: mean, standard deviation, and standard error IS: ANOVA
Stacks and Queues	Usability, perceived learning, preference.	Non-experimental (32)	Qualitative survey	DS: histograms, percentages
Stack Em Up	Suitability.	Non-experimental (15)	Qualitative survey	DS: histograms, percentages
Stack Game	Learning, motivation, usefulness, perceived learning, preference, clarity, provided support, enjoyability.	Quasi-experiment (29)	Test, Qualitative survey	DS: mean, standard deviation, and percentages IS: t-test, Cohen's d
Ramle's Stack Game	Learning, usability, user interface, interactivity.	Quasi-experiment (29)	Test, Qualitative survey	DS: percentages, aggregated values.
Space Traveller	Learning, motivation, perceived learning, enjoyability, usefulness.	Quasi-experiment (13)	Test, Qualitative survey	DS: mean, median, standard deviation, variance, histograms, percentages.
La Petite Fee Cosmo	NE			
Super Mario AVL Tree Game	NE			
AVL Tree Game	Enjoyability, engagement	Ad-hoc (5)	Observation, Qualitative survey	Informal analysis
Elemental: The Recurrence	Learning, enjoyability, perceived learning, preference.	Quasi-experiment (42)	Test, Qualitative survey	DS: percentages, means, standard deviation, histograms. IS: t-test, Cohen's d
HTML5 Hanoi Tower	Learning	Non-experimental (17)	Test, Qualitative survey	DS: mean, standard deviation, and histograms
Recursive Runner	Leaning, enjoyability, perceived learning, motivation, preference.	Quasi-experiment (31)	Test, Qualitative survey	DS: average, medians, standard deviation, histograms
Critical Mass	Learning, preference, and perceived learning	Ad-hoc (42)	Assignment, Qualitative survey	DS: percentages
Resource Craft	Learning, self-motivation	Ad-hoc (102)	Qualitative survey	DS: percentages
Prototypes Bundle	NE			
DSLEP Bundle	NE			
NE means "No Evaluation". DS means "descriptive statistics". IS means "inferential statistics".				

3.5 Discussion

Concerning the game elements used to teach and represent the data structures and algorithms, our results show that the game environment, game mechanics, and challenges are the preferred game elements. Regarding the game environment, all games reviewed, excluding the coding games, use the game environment or some of its components to represent the data structures or their fundamental aspects. A reason for this finding may be that data structures are usually represented spatially, and the game environment is a spatial representation of a fictional world. Additionally, the game environment is very flexible and allows 2D, 3D, abstract or realistic representations. This characteristic facilitates the representation of data structures and algorithms through creative scenarios that learners are familiar with. For example, some games represent the data structures through analogical models. *Star Chef* used a stack of meat balls to represent the stack data structure (see Figure 3.3). *La Petite Fee Cosmo* utilized the bridges made of floating platforms to represent the linked list nodes (see Figure 3.13). In the first level of *Super Mario AVL Tree Game*, the BST data structure was represented through a bouquet of flowers (see Figure 3.15). Other games utilized models that were very close to the theoretical models used in textbooks. For example, the *AVL Tree Game* represent BSTs and AVL trees (see Figure 3.18) using the graphical model as academic textbooks (e.g., Aho, 1988; Sedgewick & Wayne, 2014).

In relation to the game mechanics, most of the games used game mechanics to represent data structure functions. These game mechanics interact with the game environment, which, as mentioned above, is employed to represent the data structure model. For example, the insert and remove functions of the stack (push/pop) and queue (queue/dequeue) data structure are used by many games as game mechanics to interact with the game environment. In *Star Chef*, the player must supply meat balls from a stack of food using the “pop” function. In *Stacks and Queues*, the player must shoot cubes from two piles using the pop and dequeue functions. In *Stack Em Up*, the player must push and pop disks from one rod to another to solve the Tower of Hanoi puzzle. Another example is the rotation function of the AVL tree. Both AVL tree games employ it, and the players must solve conceptual puzzles using this game mechanic.

Regarding the game challenges, our results show that all the challenges designed to teach concepts about data structures are logic or navigation challenges. Also, we found that the challenges can be classified in two categories: challenges that are solved using data structure functions (e.g., insert, delete, or rotate), and challenges that can be solved using traditional mechanics (e.g., walk, jump, shooting, or drive). An example of a challenge that uses data structure functions is the Hanoi Tower puzzle, which uses the push and pop functions, and it is found in *Stack Em Up*, *Stack Game,* and *HTML5 Hanoi Tower*. An example of a challenge that uses traditional game mechanics can be found in the third level of *Super Mario AVL Tree*, in which the challenge is to find a key hidden in a set of platforms organized as a BST data structure. To overcome the challenge, the player must search for the key by applying the BST property (see section 5.2.1). This challenge is solved using the running and jumping mechanics.

To summarize, most of the games reviewed reflect the data structure models and their functions through the game environment and game mechanics. The challenges are used to solve logic puzzles or navigate the data structure model. This allows players to understand the arrangement and principles of the data structure model and the functionality of its algorithms.

In relation to topics, our results show that the most common data structure covered in the serious games reviewed was the stack, which appeared in seven of the 17 games reviewed. A reason for this finding may be that the stack is a simple but fundamental data structure, which makes it ideal for fast prototyping and testing of potential uses of learning games in the field. Additionally, stacks may appear more due to the fact that the Association for Computing Machinery (ACM) recommends that the stack be included as an essential topic in undergraduate programs (2013). However, we noticed that apart from the *Stack Game,* all the games that focus on stacks are trivial. For example, three games used the Hanoi Tower puzzle as the main game challenge, while in the other three games, the only game mechanics available were the queue and dequeue operations. As a result, these games may fail to engage the player due to their lack of sophistication. In contrast, *Stack Game* uses the Hanoi Tower puzzle in a clever way. In this game, the player must arrange blocks of different colours in a certain order, following the LIFO principle to unlock doors. Additionally, *Stack Game* offers different challenges, such as puzzles, which require that players evaluate arithmetic infix and postfix expressions as well as execute coding puzzles. We suggest that following *Stack Game*'s example, games that teach data structures should employ data structure properties in a clever and creative way to create more engaging game challenges and game mechanics. The more engaged a player is, the more motivated he or she will be, which helps to facilitate the learning process.

Furthermore, we noticed that most of the reviewed games focus on teaching simple data structures (e.g., array, 2D array, stack, queue, linked list, and dictionary) and recursive algorithms (e.g., Hanoi Tower algorithm, Fibonacci, Factorial, and Binary Search). Only a few games were found to focus on teaching medium-complexity data structures (e.g., binary trees and AVL trees) and algorithms (e.g., tree traversal and depth-first search). This finding suggests that more research on games that teach advanced data structures and recursive algorithms, such as complex trees (e.g., red-black trees or B-trees), graphs, and their associated algorithms is needed.

Concerning theoretical foundations, 65% (eleven of seventeen) of the studies reviewed reported a learning theory or principle. The most common theory observed was constructivism (five of eleven) followed by scaffolding (four of eleven), a concept based on Vygotsky's Proximal Developmental Zone (Podolskiy, 2012). This finding is consistent with the results obtained in other literature reviews. For example, Wu et al. (W. H. Wu et al., 2012) performed a literature review of serious games for learning and likewise found that most of games reviewed reported a constructivist theory. In another literature review focusing on serious games for learning science, Cheng et al. (2015) found that most of the reported learning theories were either constructivist or based on Vygotsky's theories. Similarly, our findings suggest that most of the reviewed works (eight of eleven) explain learning through games as an active process that requires the construction and socialization of knowledge (Vygotsky's theories).

In our review, only one study was found to explicitly report learning objectives. In general, learning objectives facilitate the extraction of cognitive processes and the type of knowledge delivered by a game. Consequently, it was necessary to deduce these aspects from the game description of the rest of the games.

We found that in almost all games reviewed (fifteen of seventeen), it was necessary for the learner to employ the *apply* cognitive process to achieve the learning activities. Additionally, the most common type of knowledge delivered by the games reviewed was found to be procedural (fourteen of seventeen). This finding was not surprising due to the interactive nature of video games. However, games with complex tasks that required higher cognitive processes were identified. For example, in order to succeed in the coding games reviewed, *Critical Mass* and *Resource Craft*, players had to employ the *create* and *evaluate* cognitive processes, the highest cognitive processes of Bloom's taxonomy. This finding confirms previous observations made by game scholars (e.g., (Gee, 2007; Squire, 2011)) who suggest that video games support the acquisition of skills and knowledge that require higher cognitive states. Finally, concerning type of knowledge, it was found that some games used narrative elements to deliver factual and conceptual knowledge (e.g., *Elemental* and *Stack Game*). This indicates that game elements can be used to deliver different types of knowledge.

It is a concern that sixteen of seventeen games reviewed did not explicitly report the learning objectives of the games. Learning objectives are important because they define the level of mastery of a topic that a learner should have at the end of a learning experience (Biggs & Tang, 2007). Furthermore, learning objectives specify the scope of the learning material, tool, or program. It is desirable to define learning objectives using frameworks that systematically describe the complexity of tasks that learners are expected to master (Biggs & Tang, 2007). Normally, these frameworks are hierarchical, with their classification categories possessing an ordinal nature (e.g., SOLO taxonomy (Biggs & Tang, 2007), Bloom’s taxonomy (Anderson & Krathwohl, 2001), etc.). It is also desirable that learning tools, such as serious games for learning, state their learning objectives (Mayes & de Freitas, 2004) during their design stage. By doing this, the designer can align the learning activities to fulfil the objectives and develop accurate assessment tools for the learner and the game itself. The latter aspect is quite important in terms of research which requires proper assessment tools; without a proper assessment, it is not possible to develop good theory about serious games.

Concerning evaluation aspects, we found that in general, researchers were interested in evaluating games’ (1) efficacy to teach data structures and recursion; and (2) affective outcomes. Most of the evaluations conducted were quasi-experiments; in total, thirteen games were evaluated using a quasi-experimental design. In contrast, only one study was evaluated using a full experiment. A reason for this finding may be that quasi-experiments are easier to carry out; researchers may not need to divide the sample into random groups, and they may not need a control activity. Therefore, this type of experiment is easier to design, execute and analyse than a full experiment. However, such results are not as conclusive as those obtained through a full experiment (Kitchenham, 2004).

In terms of data collection tools, we found that except for one instrument, all tools were informal. By informal, we mean instruments that were not validated nor calibrated to behave as a scale. Consequently, data collected using such instruments cannot be evaluated using parametric statistical methods such as t-tests or ANOVA. Excluding one evaluation, all collected data was analysed using descriptive statistics. Of these, three studies employed parametric methods to analyse ordinal data (scores of tests), which is unfortunate as doing so departs from best practice (Field, 2013). Additionally, studies that used qualitative instruments (three of thirteen, e.g., interviews and observations) did not report any protocol describing how the data was collected and analysed. Consequently, results obtained by these methods are not conclusive. Therefore, like other scholars (e.g., (Whitton, 2014)), we suggest that more qualitative or mixed experiments be carried out to properly analyse the nature of learning through digital serious games.

Chapter 4: Game Design for Higher Education Environments

In this chapter, we aim to answer SRQ 2: *How can game mechanics and game models be designed to teach computer science conceptual knowledge effectively using analogies?* To achieve this objective, a review of literature related to design principles for higher education, video games, and video games for learning is presented. At the end of this chapter, a new design framework based on the reviewed ones is provided. We begin the chapter by presenting the alignment principle, which is fundamental for the design of learning experiences, and which is a key component of our framework.

4.1 The Alignment Principle

According to Mayes and de Freitas (2004), a good pedagogical design requires the adoption of a learning theory. When designing learning experiences (e.g., higher education courses, or learning tools), it is necessary to align the learning objectives, learning activities, and assessment to increase learning quality. However, the alignment of these elements must be framed within a learning theory, which provides the fundamental assumptions about the intended learning. Guided by a learning theory, it is possible to select the pedagogical methods that best supports those learning assumptions. Furthermore, adopting a learning theory will help to judge whether the learning and teaching processes will accomplish the desired learning objectives.

A good example of the alignment principle is presented by Biggs and Tang (2007). They propose the term *constructive alignment*, which consists of framing all the learning activities into the constructivist paradigm. Constructive alignment is based on the principle that individuals construct their knowledge. Consequently, to achieve the learning objectives, the learning experiences must employ pedagogical approaches that ensures the active learning; in other words, students must engage in activities that allow them to build their knowledge.

This thesis has adopted a constructivist stance based on the model that individuals build new knowledge based on their experiences and the knowledge that they already possess. For this reason, constructive alignment is a key concept to structuring our design approach. In the next section, a summary of the main aspects of constructive alignment is presented.

4.1.1 Constructive Alignment

Constructive alignment is set of principles to design learning experiences for formal educational environments. According to Biggs and Tang (2007), higher education methods should ensure that all students actively engage with their educational experiences by applying learning approaches that ensure *deep learning*. Deep learning is a type of learning that occurs when students employ the appropriate cognitive activities to solve or engage with a learning task. Additionally, when students apply this type of learning, they focus on the meaning and core aspects of what is being taught and seek out principles to organize and structure the learned information (Jackson, 2012). In higher education, deep learning can be achieved by aligning the intended learning outcomes (ILOs), learning activities, and assessment (Biggs & Tang, 2007).

According to Biggs and Tang (2007), constructive alignment is based on two principles. First, its foundation is the constructivist paradigm, and it therefore considers learning to be an active process. Students must build their knowledge and meaning through the transformation and integration of information with their existing knowledge; educators cannot transfer their knowledge to the students. Educators and learning activities merely facilitate the construction of knowledge. Second, the learning activities and assessment should be aligned with the ILOs. The ILOs expressed as verbs state what students should know after the learning activity. To warranty the achievement of the ILOs, the learning activities should be designed in a way that allows the activation of those verbs. Furthermore, assessment should verify that the learners achieve the ILOs. In other words, assessment verifies that the learners possess the knowledge or skills defined by the ILOs.

In a simplified approach, it is necessary to complete three steps to implement a learning experience following the constructive alignment principles (Biggs & Tang, 2007). First, the ILOs should be defined. The structure of the ILOs should contain a verb, content, context, and standard. The verb indicates the skill that the learner should obtain. Verbs should describe higher and lower levels of cognitive processes. The content indicates the subject that learner should learn. The context and standard dictate how the learner will obtain the skill and the level of understanding. Second, the learning activities should be defined. The learning activities should be designed in a way that learners should employ the ILOs’ verbs when they engage with the activity. This will increase the likelihood of achieving the learning outcomes. Finally, the assessment should be defined. Assessment should tell the evaluator how well a learner has achieved the ILOs, and it should have grading criteria and rubrics that all students understand. Furthermore, assessment should activate the verbs and cognitive processes defined by the ILOs.

4.2 Entertainment Game Design

Researchers have demonstrated that “good” entertainment video games have properly used learning principles that resemble contemporary learning theories (Gee, 2007; Squire, 2011) and that they are intrinsically motivating (Boyle et al., 2012; Malone, 1980; Malone & Lepper, 1987). Such observations suggest that design strategies used by entertainment game designers have the potential also to produce games that support learning and that are intrinsically motivating. For this reason, we present a short review of literature related to game design approaches and game design documents used for entertainment games. The next section reviews two game design methodologies suggested in two classic game design books (Adams, 2014; Salen & Zimmerman, 2004). Then, the concept of game design document is provided.

4.2.1 Iterative Design and Player-Centric Design

Iterative Design is a methodology that focuses on playtesting and prototyping (Salen & Zimmerman, 2004). This methodology suggests that game design decisions should be made based on previous experiences, and that those experiences should be gathered via game testing. In other words, game designers must develop early prototypes focusing on key aspects of the game, such as the game system, the game mechanics, and player interactions. Other elements such as aesthetics (visual art, sound, music, etc.) should be added in later iterations. Once the early prototype is ready, the designer should do play testing, gather and evaluate the results, modify the game system based on the new insights, and the play test again. The cycle will be performed several times until the game designers decide that the game is ready to be delivered (Figure 4.1). Salen and Zimmerman suggest that, as a rule of thumb, a playable prototype of the game must be ready to be tested at the end of the first fifth of the project. The methodology is based on the observation that experiences of the player cannot be easily predicted, and for that reason, early testing with players is necessary in order to increase the chance of success.

The Player-Centric framework proposes that designers should design their games by focusing on the idea of a representative player (Adams, 2014). This design approach has objectives in which the game should succeed: (1) entertain the player and (2) fulfil the player expectations and desires. The methodology requires the designer to perform audience research or imagine an ideal player. The representative player specification archetype must contain information about who the player is, her gaming preferences, her likes and dislikes, and her motivations to play the designer’s game. Based on the representative player, designers then make design decisions intended to fulfil the expectations of that player. This approach will warrant that the game matches the capabilities and limitations of the players (Hodent, 2017).

According to Adams (2014), video game design has three main stages: concept, elaboration, and tuning. During the concept stage, the designer must focus on three main tasks. The first task is to define an audience, who are the representative players. The second task is to define the concept of the game and to understand how the game is going to entertain the audience. The third task is to determine the player’s role inside the game. Once completed, the results of the concept stage should not change during the next phases.

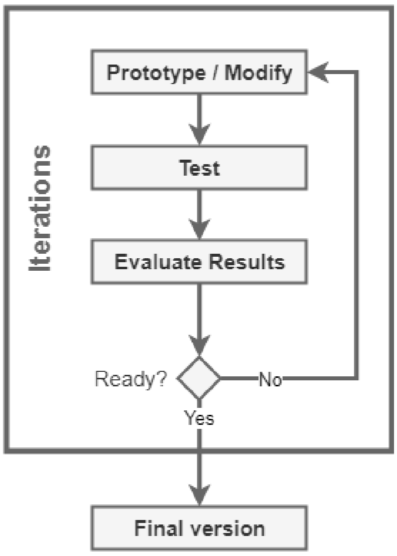


Figure 4.1: Iterative design cycle.

The next stage is the elaboration stage, which shares many similarities with the iterative design approach. This stage is the most time-consuming stage, and it consist of several sub-phases where the designer must define the primary and secondary gameplay modes, core mechanics, the protagonist, the game world, the levels, and the story. Furthermore, the designer must create several prototypes of the game before starting the final development phase. These prototypes should be done quickly without much detail. The main goal is to use them for play testing. The main elements that must be tested are the gameplay, the game system, and the game mechanics. With the information gathered from the tests, the designer should improve the play experience.

The final stage is the tuning stage. During this stage, only minor changes are allowed. This stage is ideal for balancing the game system, the levels, and the game mechanics. Additionally, during this stage, the designer could decide to eliminate sections of the game that are not working properly.

4.2.2 Game Design Documents

A game design document is a proposal, plan, or record of a game, which consists of a description of its concept and main aspects. It has two fundamental purposes:

communication/marketing and guidance. In other words, it tells others what the game is about and ensures consistency between all the members involved in the development of the game in relation to the main concept of the game (T. Ryan, 1999a). At its early stages, the game design document is under constant criticism and feedback. As a result, many design issues can be solved at this stage. However, sometimes, some game ideas can be rejected completely.

The structure of a game design document can vary from one designer to another. However, the aims are similar. According to T. Ryan (T. Ryan, 1999a, 1999b), there are four types of game design document: (1) game concept; (2) game proposal; (3) game functional specifications; and (4) game technical specifications. Each document serves a different purpose and is based on the previous one. Table 4.1 summarizes each document.

Table 4.1: Stages of a game document (T. Ryan, 1999a, 1999b).

Document Name	Description
Game concept	The game concept document communicates the core idea of the game. It is a very brief document, no more than two pages, and its target audience are producers. Typically, this document includes the following sections: introduction, background (optional), description, key features, genre, platforms, and concept art (optional).
Game proposal	The game proposal aims to find funding and resources for a game development project, and it should only be written for promising game concepts. The document includes a revised version of the game concept document and adds five sections related to marketing, costs, technical features and risks, and legal issues. It sections are a revised game concept, market analysis, technical analysis, legal analysis, cost and revenue projections, and concept art.
Functional specification	The functional specification document contains a description of the features and functions of the game. This document aims to guide the team of developers and artists who are going to create the game. The functional specification should express its content from a user perspective. It is a detailed document, and its length varies according to the size of the project. The functional specification document addresses six key areas, and each area contains several game elements. 1. Game mechanics 2. User interface. 3. Art and video 4. Sound and music 5. Story 6. Level requirements
Technical specification	The technical specification explains how to develop the functional specifications. Its target audience are the technical director and lead programmers who oversee the development team. It focuses on the system and technical aspects, such as the platform, operating system, game engine, game object data, data flow, physics, statistics, etc.

A different classification is proposed by Adams (2014). He suggests that there are nine types of game design documents, which are summarized in Table 4.2. Also, he suggests that game designers write these documents due to five reasons: (1) to record design decisions; (2) to transform ideas into specific plans; (3) to communicate the game concepts and guide the development stages of the game; (4) to set contractual obligations; and (5) to find funding.

Table 4.2: Type of game design documents (Adams, 2014).

Document Name	Description
High concept document	This is a summary of the main ideas of a game. It does not go into deep details. Its main purpose is to advertise the game to publishers or producers.
Game treatment document	This is a broad description of the game for people who are interested; however, it is a brief document. Its main purpose is to assist in finding funding to create a thorough design, a prototype, or to develop the game.
Character design document	The purpose of this document is to specify the design of characters that will appear in the game. This document includes the concept art, list of movements, background story, strengths, weaknesses, etc.
World design document	This document contains a general overview of the game world and the things it contains. It also defines the "feel," the aesthetic style, and emotional tone of the game world. Sometimes, it also includes a map of the environment.
User interface design document	This document includes aesthetics, technical, and usability considerations of the user interface (e.g., graphics, menus, audio, text, etc.).
Flowboard	This document is a combination of a storyboard and a workflow, and its main purpose is to structure the game's flow of actions.
Story and level progression document	This document records the story of the game and the level progression. It is a general description of the player's experience. It also defines the approach used to tell the story (e.g., cutscenes, mission briefings, dialogs, etc.).
On-screen text and audio dialog scripts	This document contains a record of all the on-screen text and audio dialogue scripts that may need a translation to other languages.
Game script	This document contains the rules and game mechanics of the game. It should possess enough detail to let the reader play the game in the form of a paper prototype or small simulation; however, it does not include the technical specifications.

4.3 Game-Based Learning Design

In addition to the entertainment game design approaches, it is necessary to review frameworks development for the design of game-based learning experiences. These frameworks provide a deep understanding of how to embed learning theories and learning approaches into the game elements. Additionally, they provide insights into the necessary considerations to apply game-based learning properly.

The review focuses on three frameworks, and each highlights certain considerations of game-based learning. For example, the four-dimensional framework focuses on the context where the game is going to be used and the player's characteristics. The G4LI's framework focuses on all the game elements that enhance learning. The Learning-Mechanics Game-Mechanics framework highlights specific elements of the game. Having three perspectives of how games for learning should be designed increases our possibilities of developing a successful game.

4.3.1 Four-Dimensional Framework

As its name suggests, the four-dimensional framework (Figure 4.2) considers four aspects to evaluate video games and their learning potential (de Freitas & Oliver, 2006). The framework aims to help educators to choose suitable video games and learning strategies for their learning objectives and learning environment. This framework can be used to analyse existing games and to design new educational games and simulations.

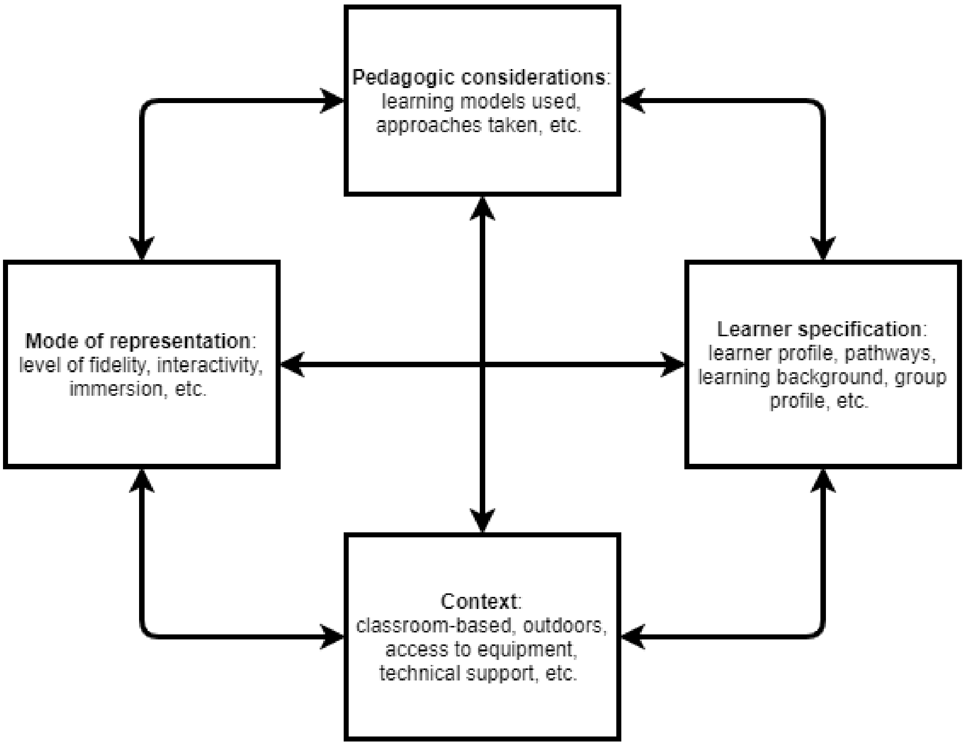

Pedagogic considerations: learning models used, approaches taken, etc.
Mode of representation: level of fidelity, interactivity, immersion, etc.
Learner specification: learner profile, pathways, learning background, group profile, etc.
Context: classroom-based, outdoors, access to equipment, technical support, etc.

Figure 4.2: Four-dimensional framework for evaluating game-based education; adapted from de Freitas and Oliver (2006).

The first dimension refers to the *context*. The context considers the place in which the learners will play and learn. It is used to evaluate historical, political, and economic factors and the availability of specific resources and tools. It is also concerned with the instructor's background and understanding of the technical aspects entailing the use of video games. The second dimension refers to the *learners*. This dimension is used to evaluate the age, knowledge level, learning background, styles, and preferences of the learners. The third dimension focuses on the *internal representational world* of the game. This dimension considers the mode of representation, the interactivity, the levels of immersion, and fidelity used in the game or simulation. This dimension highlights the differences between being immersed within the game and the process of critical reflection that takes place outside the game. Finally, the fourth dimension, *processes of learning*, is used to evaluate methods, theories, models, and frameworks used to support learning practice. Also, it considers whether learning happens in a formal education environment or an informal environment.

Additionally, de Freitas and Oliver (2006) propose a checklist (see Table 4.3), which can be used as a guideline for applying the four-dimensional framework.

Table 4.3: Checklist for evaluating the use of educational games (de Freitas & Oliver, 2006, p. 256).

Context	**Learner specification**	**Pedagogic considerations**	**Mode of representation (tools for use)**
• What is the context for learning? (e.g., school, university, home, a combination of several) • Does the context affect learning? (e.g., level of resources, accessibility, technical support) • How can links be made between context and practice?	• Who is the learner? • What is their background and learning history? • What are the learning styles or preferences? • How can the learner group be best supported? • In what ways are the groups working together (e.g., singly, partially in groups) and what collaborative approaches could support this?	• Which pedagogic models and approaches are being used? • Which pedagogic models and approaches might be the most effective? • What are the curricula objectives? • What are the learning outcomes? • What are the learning activities? • How can the learning activities and outcomes be achieved through existing games? • How can the leaning activities and outcomes be achieved through specially developed software? • How can briefing and debriefing be used to reinforce learning outcomes?	• Which software tools or content would best support the learning activities? • What level of fidelity needs to be used to support learning activities and outcomes? • What level of immersion is needed to support learning outcomes? • What level of realism is needed to achieve learning objectives? • How can links be made between the world of the game and reflection upon learning?

As mentioned above, the four-dimensional framework aims to facilitate the evaluation and design of game-based learning experiences. The framework emphasises the analysis of the context where the learning experience will happen. For example, it analyses the location, learner background, learning styles, curricula, learning objectives, and learning outcomes to evaluate the efficacy that a game-based learning experience has or may have. It is expected that by contextualizing the learning activity, the design process may result in a higher quality game. However, the four-dimensional framework does not address topics of games design. In particular, the framework does not provide guidelines explaining how to transform the learner specifications and the pedagogical considerations into ludic experiences. Also, the framework does not provide a list of game elements that can be used to convey learning (e.g., game model, game mechanics, user interface, or narrative elements). Finally, the four-dimensional framework does not provide examples of game design using the framework. Therefore, a review of frameworks that focuses on game for learning design is provided in the next section.

4.3.2 G4LI's Framework for Game-Based and Playful Learning

The Game for Learning Institute's (G4LI) framework for Game-Based and Playful Learning (GBPL) is a framework created to design and analyse digital game-based learning from a pedagogical and psychological perspective (Plass et al., 2014, 2015). The framework models the pedagogical and psychological pillars of game-based learning, and according to Plass et al., the framework can be adapted to any learning theory. The objective of the framework is to allow designers to incorporate the playful dimension of games into the learning theories used for the game. Furthermore, the framework aims to be simple and general; it is intended to be suitable for any type of digital game of any game genre.

4.3.2.1 A Model of Game-Based Learning

According to Plass et al. (2014, 2015), game-based learning can be modelled using three basic aspects: challenge, response, and feedback. During play, these aspects interact with each other and form a loop (see Figure 4.3). First, the game presents a challenge to the player. Then, the player inputs a response. Then, the game modifies the state of the challenge and returns to the player feedback about the current state, restarting the cycle. Plass et al. suggest that embodiment of the learning theories depends on the way that the challenge, response, and feedback are designed. The quality of the learning experience depends on how these game elements are designed, and to achieve a good learning experience, the design process may require several iterations and testing.

The model suggests that the core building blocks of a game are the incentive system, game mechanics, aesthetics, narrative, and musical score (Plass et al., 2014, 2015). The incentive system is the mechanism that motivates the player and provides feedback to modify the player behaviour. The game mechanics are the activities that the player performs repeatedly in the game. Aesthetics refers to all the visual elements of the game, such as the look and feel, theme, graphic representation of the objects, environments, etc. The narrative elements communicate the story and plot that provides a context to the mechanics and rules of the game. Finally, musical elements include the background music, voices, sound effects, etc.

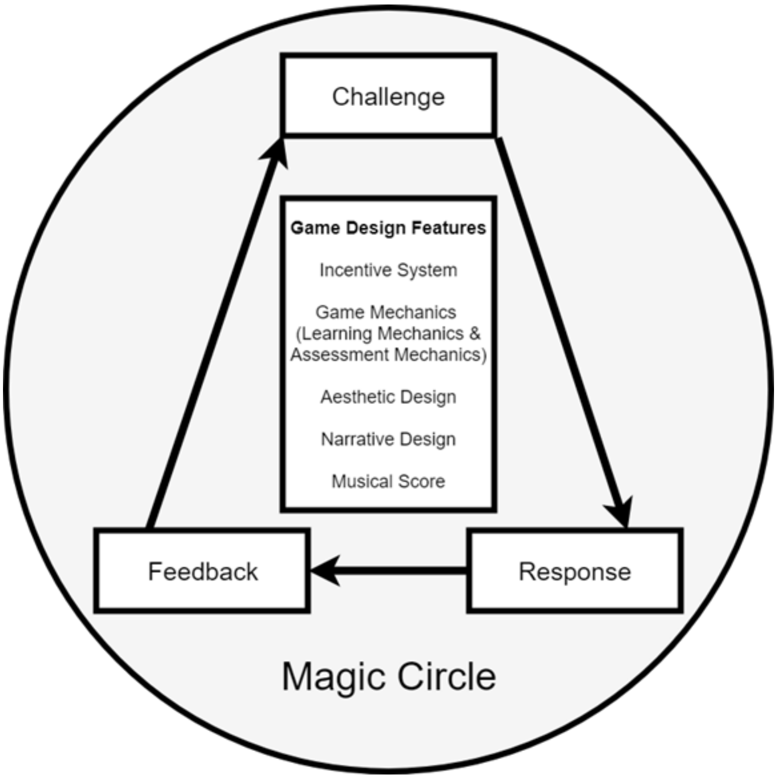


Figure 4.3: Model of game-based learning; adapted from Plass et al. (2015).

4.3.2.2 The Design Framework

Plass et al. (2015) propose that to ensure an efficient game-based learning experience, game designers should consider the four *foundations of game-based learning* when designing the core building elements of a game. The framework focuses on four foundations based on four learning aspects used in game-based learning design. Specifically, these areas are the *affective*, *behavioural*, *cognitive*, and *socio-cultural*.

According to Plass et al. (2015), designers can use different learning theories or approaches to address each *foundation*, and when possible, designers should consider *game-based learning design patterns*. Game-based learning design patterns are "general solution to commonly occurring problems that can guide the design of effective games for learning" (p. 265). However, there are few game patters available. Figure 4.4 presents a graphical representation of the framework.

The *cognitive* foundation addresses the cognitive processing of the learning contents and cognitive load experienced by the learner during gameplay. Some learning principles that can facilitate cognitive representation are situatedness of learning, transfer of learning, scaffolding and feedback, dynamic assessment, informative design, interactive design, and gestures and movements. Additionally, Plass et al. identify three design patterns. First, game mechanics should be aligned with the learning goals. Second, scaffolding should facilitate the integration of important content. Finally, game elements that are not aligned with the learning goals should be eliminated because they can increase the cognitive load.

The *motivational* foundation addresses the motivational aspects of games that enable engagement and the learner’s desire to continue playing. The motivational foundation focuses on the reasons why learners want to play learning games (e.g., interests, desires, goals, etc.) and approaches to enhance learners’ motivation. Designers should consider motivational principles, such as intrinsic and extrinsic motivation, situational and individual interests, and mastery and performance goal orientation.

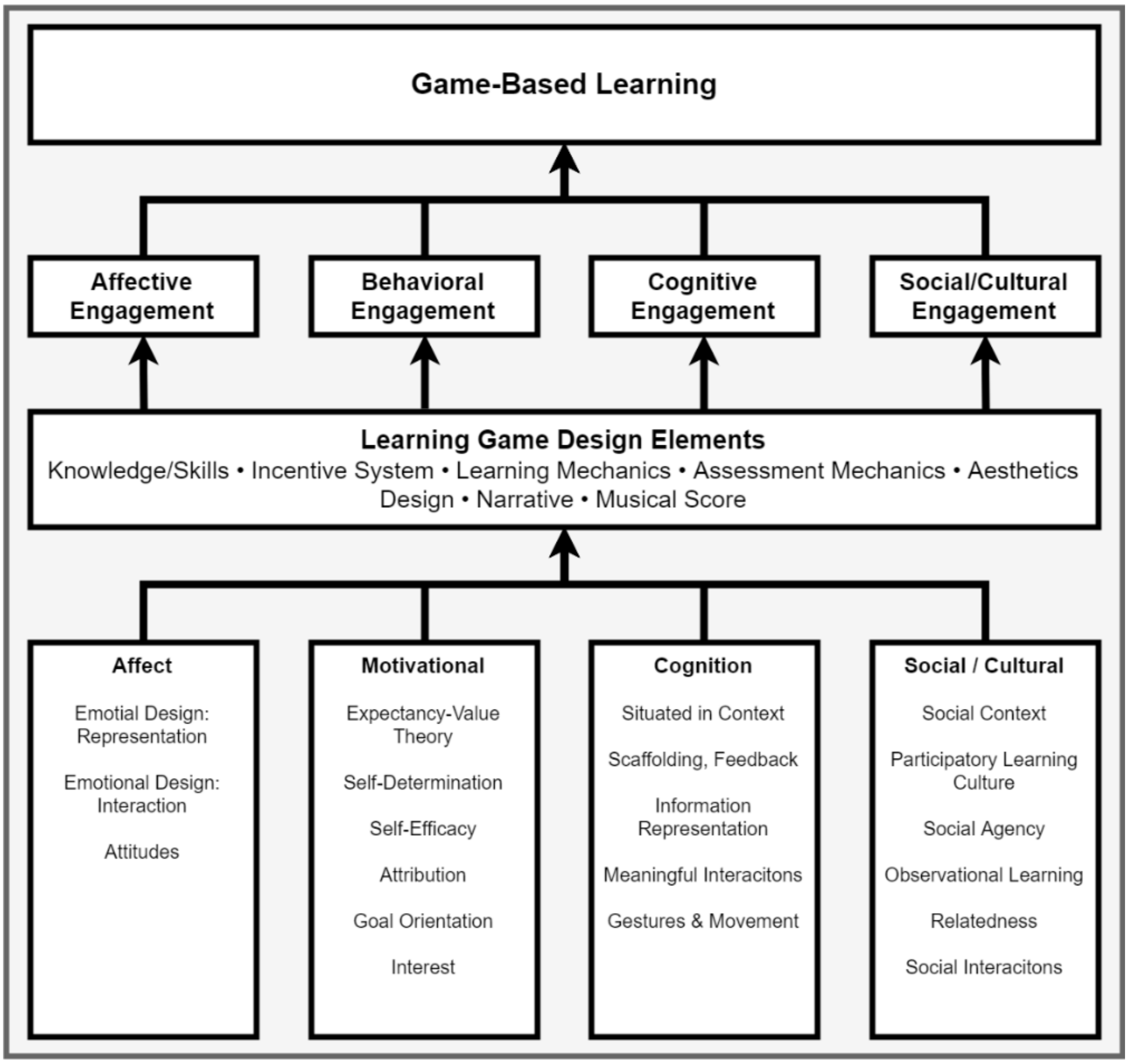


Figure 4.4: Integrated design framework of game-based and playful learning; adapted from Plass et al. (2015).

The *affective* foundation addresses the player’s emotions, attitudes, and beliefs and how game elements can provide positive affective engagement for learning. Additionally, it studies how affect impacts the cognitive, motivational, and social aspects of learning.

The *sociocultural* foundation addresses the aspects of games that facilitate social interactions and the construction and motivation of knowledge. According to Plass et al., this foundation considers the social context, participatory learning, social agency, observational learning, relatedness, and social interactions.

4.3.2.3 Learning Mechanics and Assessment Mechanics

Plass et al. (2011) suggest that game mechanics and rules are key elements of serious games. Through them, designers can regulate players' behaviour and facilitate the understanding of models and concepts that the game tries to transmit. Thanks to this characteristic, games can be used to develop learning experiences. To simplify the design process of games for learning, Plass et al. propose an extension to the previous framework called *Learning Mechanics* and *Assessment Mechanics*.

Learning Mechanics (LMs) are the set of actions that result from the translation of learning theories into instructional strategies (Plass et al., 2011, 2012). Those actions are repeatedly performed by the player during the game. LMs are guidelines or descriptions of the learning processes and contents but not specific game mechanics. LMs constitute an extra level of abstraction between learning theories and game mechanics. Based on the LMs and the learning objectives, designers must select the proper game mechanics to develop the activities required by the LMs. For example, if the LMs require that the learner acquires information about a specific historic place, this could be implemented using exploration mechanics or puzzle mechanics; the designer must decide which game mechanic is more suitable to represent the LMs. All LMs must be grounded in learning science and learning theory.

In order to choose the best game mechanics for a LM, Plass et al. (2011) suggest the following steps:

1. Designers may not introduce excessive amounts of unnecessary cognitive load.
2. Designers may not simplify the game mechanics too much, because this can compromise the transfer of learning.
3. Designers may not introduce unnecessary confounds. In the context of this framework, confounds refer to additional skills and knowledge that are necessary to play the game but that do not relate directly to the learning objectives.

Assessment Mechanics (AMs) are the set of actions repeatedly performed by the player that are measurable and can be utilized to assess the knowledge acquired by the player during the game sessions. According to the authors, the main goal of AMs is to reveal the learning process and learning objectives of the player during gameplay. They could be recorded using a user log or some previously designated variables that must be available to evaluator. Like LMs, AMs constitute an extra level of abstraction that links assessment methodologies and game mechanics; they specify the assessment processes or variables that should be measured, but they do not specify how are they going to be implemented in the game. The designer must select the proper game mechanics that allow assessment.

The G4LI framework for game-based learning and playful learning is a comprehensive framework that focus on learning aspect of games for learning. It provides examples of theories and pedagogical approaches that can be useful for game design. Additionally, the framework provides useful concepts, such as learning mechanics and assessment mechanics, which can aid designer to align learning activities and the ludic activities of the game. However, the framework does not focus on the context where the learning experiences happens. For example, the resources (e.g., economic, and technological) available to implement the learning experience. Additionally, it does not address other game elements that can convey learning, such as the game model, user interface, or narrative elements.

4.3.3 Learning Mechanics and Game Mechanics Model

The Learning Mechanics and Game Mechanics (LM-GM) model bridges the gap between learning theories and game mechanics. It is based on the idea that serious games have their own type of game mechanics (GMs) called *serious game mechanics* (SGMs). According to Arnab et al. (2015, p. 393), SGMs are “the design decision that concretely realises the transition of a learning practice/goal into a mechanical element of gameplay for the sole purpose of play and fun” (see Figure 4.5). In other words, SGMs are the GMs and game elements that allow the player to obtain learning gains through play. Additionally, Arnab et al. suggest that SGMs are based on learning mechanics (LMs), which are “the dynamic operation of learning, that we typically model relying on learning theories and pedagogical principles” (p. 393). Therefore, SGMs are LMs that are embodied in the GMs.

The LM-GM model suggests that efficient game design and analysis require to define clear SGMs. To achieve this, the model proposes the use of tables and diagrams to model the static and interactive relationships between LMs and GMs. The tables are used to define the static aspects of the game such as the game mechanic, the learning mechanic, the implementation and use of the mechanic. The diagrams are used to model the order and interactions of the LMs and GMs inside the game. Additionally, the model provides a non-exhaustive list (Figure 4.6) of LMs and GMs collected from previous research (Arnab et al., 2015). The list aims to guide designers and scholars while designing or analysing SGMs.

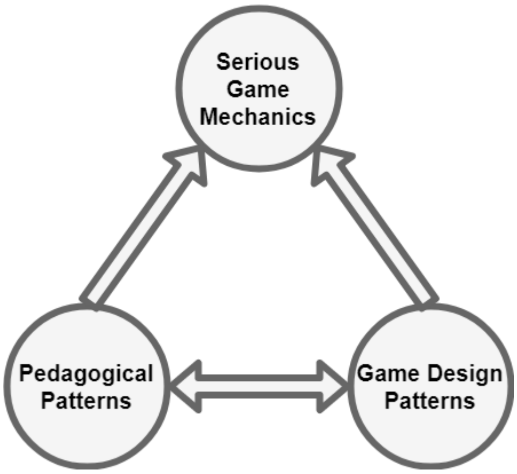


Figure 4.5: The relationship between SGMs and the pedagogical and game design patterns of a game; adapted from Arnab et al. (2015).

The LM-GM model highlights the importance of aligning the game mechanics with learning activities that are based on a learning theory. Aligning the game mechanics with learning theories and learning objectives of the learning experiences is a fundamental principle of instruction design that ensures learning. However, the LM-GM model is very specialized, and it only focuses on game mechanics. The model does not cover other game elements that can convey learning (e.g., narrative elements, user interface, or game model). Also, it does not focus on the context where learning is going to occur.

4.4 The Context-Learning-Game (CLG) Framework

In this section, we aim to answer SRQ 2: *How can game mechanics and game model be designed to teach computer science conceptual knowledge effectively using analogies?* Results of our brief review of game-based learning design frameworks show that the design frameworks reviewed specialize in some aspects of game design, leaving out other important design topics. For example, the four-dimensional framework focuses on the context where the learning experience will happen. The G4LI framework focuses only on the learning aspects of the game; however, it is a very comprehensive framework. The LM-GM framework specializes only on game mechanics. Therefore, we consider that it is necessary to create a design framework for video games for learning that takes the strengths of each framework reviewed and enhanced them with other concepts reviewed in this chapter (e.g., the alignment principle and the entertainment design approaches). By doing so, it is possible to create a comprehensive framework, which allows us to answer SRQ 2.

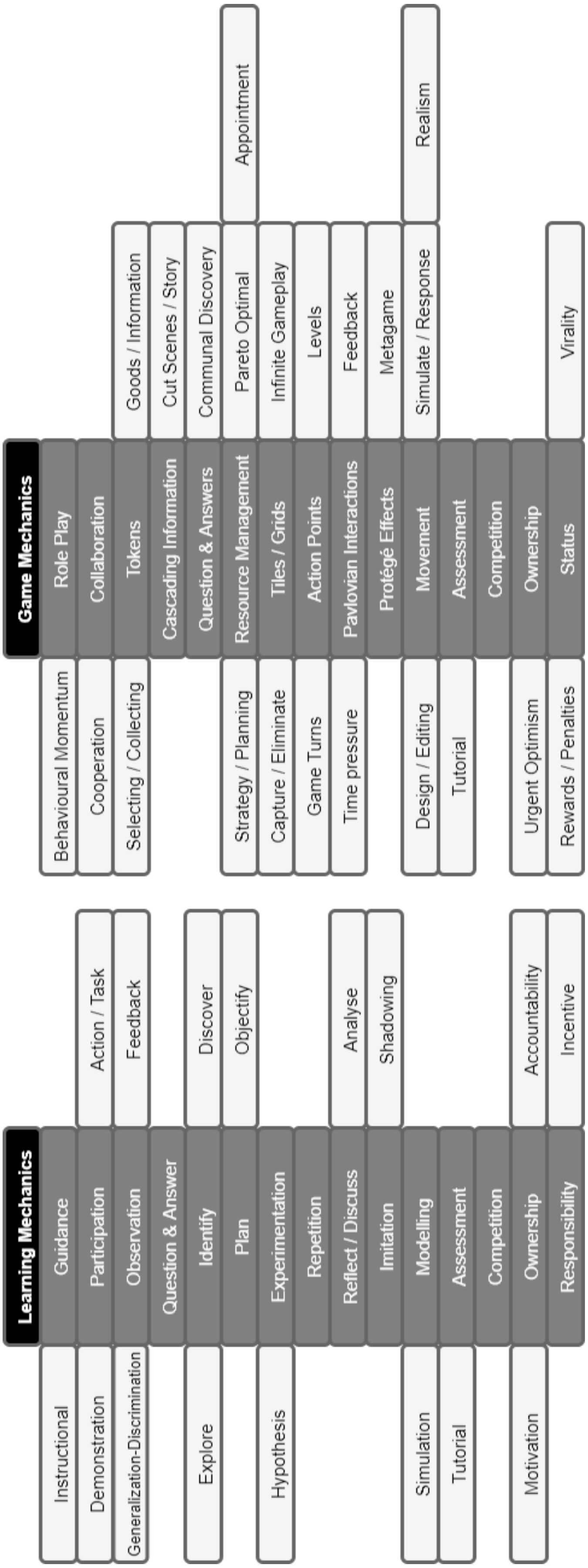


Figure 4.6: List of Learning Mechanics and Game Mechanics; adapted from Arnab et al. (2015).

Our framework integrates the concepts, frameworks, and design approaches reviewed in this chapter, and it is divided into three design stages: context, learning, and game. For this reason, we named it the Context-Learning-Game (CLG) framework. Each design stage of the framework is not exhaustive; so, other designers may include more relevant aspects when it is necessary. Additionally, the CLG framework encourages designers to use tested frameworks and theories to tackle each stage and substages. The aim of the framework is to provide a global perspective of design process and its iterative nature.

As mentioned above, the CLG framework suggests that the design processes for video games for learning should be divided into three stages: (1) context, (2) learning, and (3) game, and it can be visualized as a pyramid, where each stage serves as the base of the next one. The context is the basis for the leaning aspects, and the learning aspects are the basis to implement the video game for learning (see Figure 4.7). Furthermore, the design process is iterative. For example, once a designer finishes the game stage, he or she should return to the previous stages to revise the assumptions made (e.g., context or learning assumptions). Then, if necessary, the designer should improve/modify those aspects that need to be changed. After that, the game designer should revise the assumptions made in the game stage to determine whether they reflect the context and learning assumptions. Then, he or she can implement an improved version of the video game for learning.

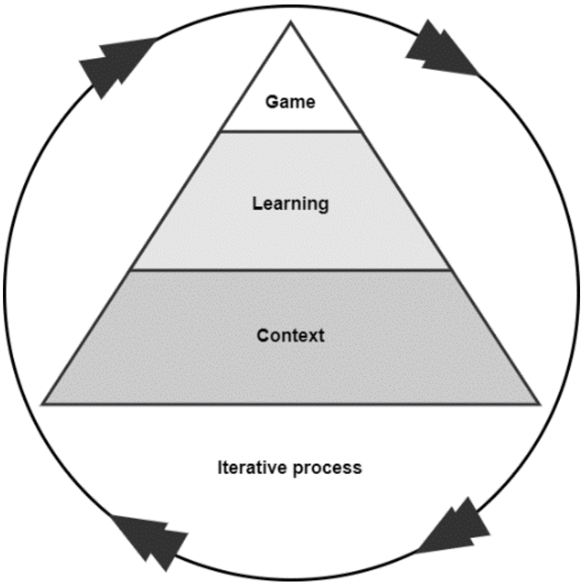


Figure 4.7: The CLG framework, a design framework for video games for learning.

4.4.1 The Context Stage

Generally, designing requires a deep understanding of the addressed problem. Understanding the problem involves analysing and defining the key elements of the context where the designed object is going to be developed and used. By analysing and defining the context, it is possible to identify strengths and constraints related to key aspects of the design process (e.g., resources available, environmental factors, target audience, etc.). Designers must consider the strengths and constraints when making design decisions to find an optimal solution to the addressed problem. Therefore, the objective of this stage is to define the context of the problem (e.g., design a game for learning data structures) to identify strengths and constraints that may affect further stages of the game design process.

The context stage is based on the four-dimensional framework (de Freitas & Oliver, 2006). However, it also includes aspects that apply for entertainment games during their design stage. In this stage, the game designer should analyse the problem, which is to design a video game for learning, and the resources available to solve the problem. Concerning the problem analysis, the designer should analyse the educational environment, the target audience, and the location where the solution is going to be used. Concerning the resources available, the designer should analyse the deadlines, the size of the team and the technical and academic skills of its members, as well as the development tools available, such as computers, software, cameras, microphones, and other equipment. A context analysis should include at least the above-mentioned aspects. However, some contexts will require the consideration of additional aspects. The following list describe the key aspects of the context that designers should contemplate:

1. *Scope and objective*. They define the size and main purpose of the project. For example, commercial games for learning may have a large scope (e.g., they may cover topics exhaustively), and they must be fully completed products. Therefore, their design and development will require a considerable amount of resources. On the other hand, game prototypes, which are useful for research purposes, may have a small scope and may not need a large team. Therefore, they will require fewer resources than a commercial game for learning. Defining the scope and objectives of the project will guide further design decisions.
2. *Educational environment*. It defines the type of learning environment where the learning experiences will take place. For example, the education environment can be formal or informal; it can be primary school, high school, or a university or college; or it can be professional training. Also, in formal education environments, it may be useful to consider the curricula and course programme.
3. *Target audience*. Game scholars and game designers agree that the players of the game are one of the most important aspects to consider when designing a video game. By defining the target audience, the designer aims to understand the expected player/learner who is going to play the game and design the game to suit his or her characteristics, such as demographic

data, academic level, or gaming experience. For example, the designer may define (or collect) information of the target audience, such as the age, the gender, the socio-economic status, the academic background, academic level, and gaming preferences. A useful approach that may help designers understand their target audience and guide their decisions is the creation of *personas* (Cooper et al., 2014), which are definitions of an expected learner/player. Understanding the characteristics of the target audience will help the designer to make decisions concerning the complexity of the contents, the learning approaches, and the video game.

4. *Location*. It defines the place where the learning experience will occur (e.g., a classroom, laboratory, home, outdoor space, etc). This will help designers make decisions regarding the pedagogical approaches and game design aspects. For example, if the game will be used by students in their home without any guidance from their instructors, it may be useful to use pedagogical approaches used in long distance education. Also, it may be useful to design a comprehensive in-game tutorial. Additionally, this aspect should address whether the location will have the necessary resources to support the learning experience. For example, the designer should the computers available in a laboratory, the size of the classroom, the resources available in the learner's house, etc. The designers should verify whether the learners and educators have the devices and knowledge to perform the learning experiences.
5. *Deadlines*. Usually, commercial and research projects have deadlines that must be met. Time restrictions must be considered because they affect many design decisions of the following stages.
6. *Design and development team*. This aspect addresses the human resources available to design and develop the solution. The composition of the team will define the strengths and limitations. These characteristics should be contemplated before starting a project because they will affect further decisions. For example, a team with many graphic designers or 3D modellers can create original art and 3D assets for the game. However, they may need to simplify the game mechanics of the video game. On the other hand, a team composed of coders can develop complex game mechanics for the game. However, they will need to buy 3D assets. Designers should also include potential collaborators, which are not part of the team but can contribute to the project, such as experts or educators.
7. *Budget and development tools*. It considers all the economic resources and tools available to design and develop the learning experiences/game. Development tools include hardware, software, information sources, assets, and equipment to produce multimedia content.

All the described elements should be documented in a short report. This report should provide the designer with an understanding of the context where the game for learning is going to be used and the resources available to design and develop the game. A proper understanding of the context will increase the probabilities of developing a successful learning tool. The designer will have information to choose adequate learning and cognitive theories, pedagogical approaches, and development tools that suit the team's capabilities and budget.

4.4.2 The Learning Stage

The objective of the learning stage is to define the learning aspects of the game, which are the learning theory, the contents, the learning objectives, the pedagogical approaches, and the assessment methodologies. In other words, this stage defines the aspects of the game that differentiate it from an entertainment game.

The learning stage is closely related to the context stage because all the learning aspects are constrained by the context. For example, the designer must choose an appropriate learning theory for the target population; he or she must select the contents and the learning objectives according to the background and academic level of the target audience; and the learning activities must be appropriate for the academic environment and locations defined in the context stage. Therefore, this stage cannot begin without the context analysis.

Furthermore, the learning stage must follow the alignment principle. In other words, the learning objective, the learning activities, and the assessment should be aligned with the learning theory. Consequently, the designer should select the learning theory before selecting the pedagogical approaches and assessment methodology.

The following list describe the main aspects that should be defined in this stage.

1. *Learning theory*. It defines the assumptions of how the target population learn. The assumptions of leaning are fundamental, because they guide the selection of the pedagogical approaches and learning activities. The designer may use one or more learning theories, which provide a systematic and validated approach to understand human learning. However, the selected theories should be used consistently, and they should not contradict each other.
2. *Content and learning objectives*. These aspects are closely related to the context. The content defines the skills or knowledge that the learner should acquire. The learning objectives specify the scope and expected mastery level that the learner should acquire after completing the gaming session successfully. The learning objectives should be stated in a way that they can be measurable. Taxonomies of learning objectives, such as Bloom's taxonomy (Anderson & Krathwohl, 2001) or the SOLO taxonomy (Biggs & Tang, 2007), may provide guidance when writing the learning objectives.

3. *Pedagogical approaches*. The pedagogical approaches define the methods used to teach the knowledge or skills. Usually, the pedagogical approaches are aligned with some learning assumptions. For example, scaffolding is a learning approach based on the Proximal Development Zone theory developed by Vygotsky, which is a constructivist theory (Zydney, 2012). Therefore, the pedagogical approaches should be selected carefully to match the selected learning assumptions. Additionally, due to the interactive nature of video games and complexity of this medium, the designer should consider the cognitive limitations of humans while learning. In this regard, designers should pay special attention to the presentation of learning contents.
4. *Learning activities*. The learning activities are tasks that learners should perform to achieve the learning objectives. They must satisfy the learning assumptions, be aligned with the learning objectives, and apply the pedagogical approaches. Additionally, when drafting the learning activities, the designer should consider that they will be translated into ludic activities.
5. *Assessment*. The assessment is a systematic approach to measure whether the student achieved the learning objectives. The assessment should be aligned with the learning theory, the learning objectives, and the learning activities. It should also possess a similar level of difficulty to the learning activities.

The resulting artifact of this stage is a document that defines each of the listed aspects. This document should be comprehensive and clear. The designer should pay special attention to the learning theory, the learning objectives, and the pedagogical approaches. These aspects are fundamental because they define the factors that should be evaluated to understand whether the video game for learning was successful.

An example of how to define the learning theory is given in section 2.1.1, section 2.1.2., and section 2.1.3. An example of how to define the contents is given in section 5.2.2.2. The example of the learning objectives is presented in section 5.2.2.3. An example of the definition of learning activities is presented in table 5.2.

4.4.3 The Game Stage

The game stage is based on the Iterative Design (Salen & Zimmerman, 2004) and Player-Centric Design (Adams, 2014) approaches. However, this stage is constrained by the contexts and the learning requirements defined in the previous stages. Its main objective is to develop a playable prototype of the game. To achieve this objective, this stage is divided into three sub-stages: game design, development, and testing. Each sub-stage produces its own artifacts. In the design sub-stage, the game designer must define the main elements of the video game, and he or she must portray these ideas in a game design document. In the development sub-stage, the designer and the development team must create a prototype of the game, which is the artifact of these stage. In the last sub-stage, testing, the designer must evaluate the game. The game testing may evaluate many aspects of the game (e.g., engagement, user experiences, etc.). However, the designer should pay special attention to the learning aspects (e.g., learning assumptions and learning objectives), which are the core aspect of a game for learning.

4.4.3.1 The Game Design Sub-Stage

The main objective of this stage is to define the game elements (e.g., game mechanics, game challenges, game story, aesthetics, user interface, game world, etc). To achieve this objective, the game designer may use game design documents (see section 4.2.2). Game design documents are an easy and economic method to define, visualize, and modify the main concepts of the game. It also allows the designer to check for flaws and to make quick corrections. While drafting the game design document, the designer may do small research and prototyping tasks in relation to concepts that may be included in the document.

The game design sub-stage is constrained by the context and the learning aspects. The game designer should align the game elements with the target audience, the resources available, the learning theories, the pedagogical approaches, the learning objectives, and the learning activities. Also, the game design should consider the entertainment aspects of a game.

Concerning the learning aspects, the designer should translate the learning activities into ludic activities, using the game elements as building blocks. It is desirable that the game design document matches the learning objective, the learning activity, and the ludic activity. Additionally, the designer may make use of design frameworks for video games for learning (e.g., G4LI’s framework, GM-LM framework, or Game Object Model framework (Amory, 2007)).

Concerning the entertainment aspects, the designer should consider the factors that makes a video game fun, such as challenge, fantasy, curiosity, control, competition, and social features. These factors increase the intrinsic motivation and engagement of the player, which are desirable characteristics for learning. Like the learning aspects, the designer should use the game elements as the building blocks. Additionally, he or she may review studies related to the elements that make video games fun (e.g., Boyle et al., 2012; Kutner & Olson, 2008; Malone, 1980; Malone & Lepper, 1987; Whitton, 2007).

At the end of this sub-stage, the designer should have a game design document. This document should translate the learning aspect into ludic learning activities. Also, it should specify the workflow of the game and the elements that engage the player. Two examples of the game document are available in Appendix 1 and Appendix 2. These game documents are organized by the levels of the game and show the contents, learning objectives, and learning activities covered by each level. Concerning the game elements, they present the game challenges, the structure of the game environment, and the game story.

4.4.3.2 The Development Sub-Stage

Once the game elements were specified, the designer and the development team should implement the first prototype of the game. The prototype should be a small playable version of the game. It does not need to be a complete version of the game and does not require polished aesthetics. However, it should reflect the core concept, game mechanics, and the other elements that convey learning.

The main objective of the game prototypes is testing. According to the Iterative Design methodology, fun and engagement are difficult to predict and design. In addition, the learning aspects constrain the implementation of fun. Therefore, it necessary to test possible the playability, the fun elements, the engagement, and the learning outcomes as soon as possible. Therefore, a cheap and quick-to-develop tool is necessary before creating the final version of the game.

It is desirable to implement a digital playable version of the game rather than using a paper-based prototype. Paper-based prototypes have limitations, and they cannot portray the complex aspects of a video game model. For example, a paper-based prototype may not be able to portray the fundamental concepts of a video game that possesses a game world that behaves like a data structure.

There are many tools available to develop digital game prototypes. Many game engines, such as Unity, Unreal Game Engine, or Godot, have free version of their licenses. These game engines provide plugins (e.g., visual scripting tool or in-game 3D modelling tools) and generic assets (e.g., scripts, 3D and 2D models, sound effects, game music, etc.) that allow fast prototyping. Additionally, there are asset stores that allow the purchasing of assets and tools that increase the development productivity. Due to the availability of game engines and asset stores, developing digital game prototypes is less demanding than some years ago when these technologies where not available.

At the end of the development stage, the designer and development team should have a playable prototype of the game. It does not need to be a final version, but it should portray the main elements of the game including the learning aspects. The prototype will be used in testing substage, and it will serve as a standing point for further versions of the game prototypes and the final version of the game.

4.4.3.3 The Testing Sub-Stage

The only way to know whether the video game for learning is effective (e.g., increases knowledge), engaging, and fun is through testing. Therefore, once the prototype is ready, it should be validated and tested.

The validation should verify whether the contents and teaching methods are delivered by the game properly. Experts in education and the learning subject should participate in this stage. We consider that an ideal methodology to perform the validation is through the Delphi method. “The Delphi method is an iterative process used to collect and distil the judgments of experts using a series of questionnaires interspersed with feedback.” (Skulmoski et al., 2007, p. 2). This is useful to stablish consensus among experts in a specific topic.

The game testing should evaluate learning efficacy and factors related to the use of the video game, such as usability, user experience, usefulness, and instructional aspects. Also, it should be performed on a sample with the same characteristics of the target population. There are several methodologies that may be useful to collect information about the game. For example, the designer can use qualitative methodologies, such as observations, in-depth interviews, workshops, or focus groups. If the designer chooses qualitative methodologies, he or she should define a detailed protocol. Furthermore, the designer can adopt a quantitative approach, such as randomized experiments using statistically validated tests.

At the end of this stage, the designer should develop a document reporting the method employed to collect data, the results of the experiment, an analysis of the data, and a discussion of the results. This report may point out strengths of the game as well as gaming and learning issues that must be corrected. Additionally, it may detect and document unexpected player behaviours. An example of a game evaluation is available in Appendix 3.

Based on the results of the evaluation, the designer may consider it necessary to return to previous stages to correct the issues found during the testing stage. Usually, this iteration will occur several times until an advanced version of the prototype can be used to develop the final version of the game. Figure 4.8 shows the CLG framework's workflow and the documents generated in each stage.

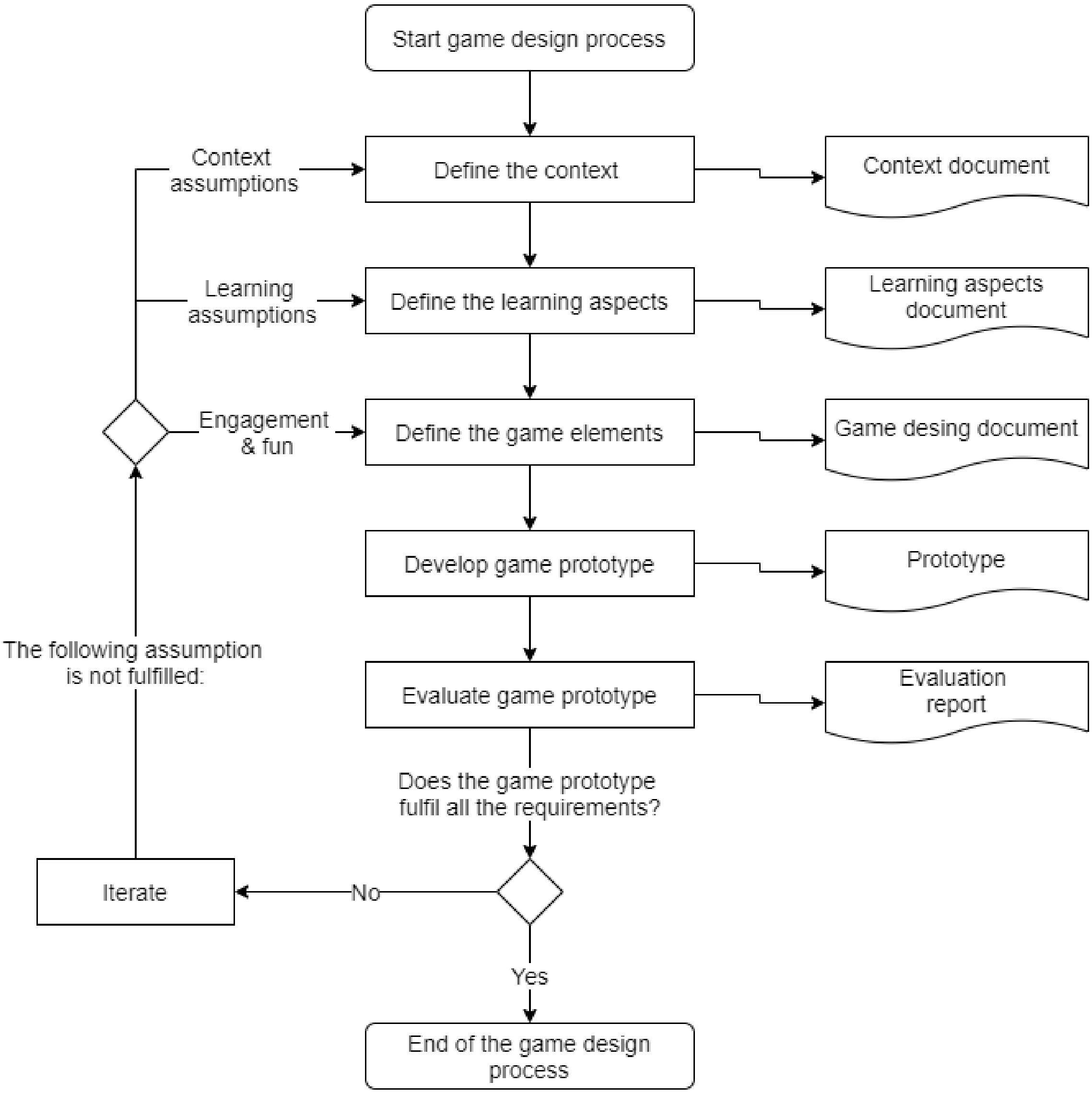


Figure 4.8 The CLG framework workflow.

Next chapter provides a description of how we used the CLG framework to design and develop a video game for learning computer science conceptual knowledge that uses analogies, game mechanics, and the game model to teach the content.

Chapter 5: DS-Hacker

In Chapter 1, we stated that the aim of the thesis is to study the efficacy of video games for learning that use analogies embedded in their game mechanics and game model. Specifically, this objective is addressed by two research questions:

SRQ 4. Are video games for learning that use analogies to teach conceptual knowledge more or less effective and motivating than other, more traditional, digital approaches?
SRQ 5. Does the realism level of video games for learning that use analogies affect the learning gains and motivation?

To answer these research questions, it is necessary to build a video game for learning that teaches conceptual knowledge of BSTs using analogies from scratch. Additionally, the game should possess two versions. The first version should have 3D realistic graphics and physics. The second version should have 2D abstract graphics and simplified physics. To reduce the external factors that can affect the results of our evaluations, both versions should cover the same contents and has the same learning objectives and learning activities (goals, challenges, and game mechanics).

In the next section, the design and development process are presented. In section 5.2, the context and learning aspects of the games are provided. Section 5.3 presents a description of the final versions of DS-Hacker. Finally, section 5.4 provides a discussion of the game and the expected outcomes.

5.1 Design and Development Process

In this section, the design and development process of the 2D and 3D versions of DS-Hacker are described. This section covers two versions chronologically int the other in which they were developed. Thus, the 3D version process is presented first and then the 2D version.

5.1.1 DS-Hacker 3D

The design and development process of DS-Hacker 3D was organized using the design framework presented in Chapter 4. The framework suggests that the context and learning aspects of the game should be defined before starting the design and development stage. Therefore, our design process began with defining the context where the learning experience would occur. The aspects covered in the context stage were the learning environment (e.g., education environments where the game can be used), target population (e.g., demographic assumptions, background, and academic level), resources (e.g., human, academic, and technological), and locations (e.g., possible locations where the game can be played). These aspects were recorded in the first game design document (see Appendix 1).

In the next stage, we defined the learning aspects of the game, starting with the assumptions of how people learn. For this effect, we used Kolb's experiential learning theory (KELT), which suggests that learning is a holistic process where people build and re-build their knowledge based on their current and previous experiences (see Section 3.1.2). Once the learning assumptions were defined, the pedagogical approaches were specified. Specifically, we selected analogies. As mentioned in Section 3.1.4, analogies facilitate the construction of new knowledge using familiar knowledge. This approach is aligned with KELT, and it is possible to use it in game-based learning experiences. Furthermore, motivation and cognitive load theories were reviewed to increase engagement and intrinsic motivation, and to facilitate the presentation of the learning content.

While we were defining the learning theories and pedagogical approaches, we defined the content and the learning objectives that the game was going to cover. The content and learning objectives were based on the guidelines for undergraduate degree programs developed by the Association for Computing Machinery (ACM; 2013). The scope of the learning objectives just covered introductory concepts and few algorithms. Finally, the learning activities were drafted in a way that they could be translated into ludic activities. They were aligned with the learning theory, pedagogical approaches, and learning objectives. Like the context, the learning aspects were recorded in the game design document.

In the design and development stage, we translated the learning activities to game mechanics, goals, and challenges. Additionally, we created a science fiction game environment and story, which intend to engage and motivate the player. To achieve these aims, we followed our design framework, which is based on the *Iterative Design* and *Player-Centric Design* approaches. As mentioned in Chapter 4, Iterative Design suggests that the first prototype should be developed as soon as possible to be tested it. Using the feedback of the testing sessions, the design assumptions and prototype should be modified and tested again until reaching the final version. Player-Centric Design suggests that the design process should focus on a player representation. The player representation should consider the player's capabilities and limitations. For these reasons, the design and development process of *DS-Hacker 3D* had three major iterations where the game experienced several modifications.

Before starting to draft the game document, we performed a review of game engines and of the assets (e.g., sound effects, particle effects, scripts, and 3D models) available for the engines.

The rationale of this review was to find a development environment that would facilitate rapid prototyping and offer high asset quality. The game engines reviewed and tested were Godot, Unreal Engine, and Unity. After the testing, Unity was selected due to four reasons. First, it has a free licensing option for games that does not generate $100,000 or more per year, which is convenient for non-profit academic projects. Second, the Unity engine has a comprehensive library of functions for 3D game development, and it is well documented. Third, the scripting language of the Unity engine is C#, which the author (who is also the developer) proficient in. Fourth, Unity has an online asset store that possesses good quality assets, including free ones. Additionally, Unity's Asset Store has assets that can be used to develop of different genres and topics. This characteristic increases the range of game design decisions that we could make as designers (e.g., selection of the genre, game mechanics, narrative, aesthetics, look and feel, etc.).

After selecting the game engine and reviewing the assets available, the first draft of the game design document was written (see Appendix 1). The game design document defined the learning and gaming aspects. Regarding the learning aspects, learning was organized by game levels. Each level covered a set of learning objectives, and the learning activities of each objective were translated into level goals, challenges, and game mechanics. The challenges and learning content were communicated through the dialogue mechanics. The challenges consist of puzzles and navigation challenges through the game environment that reflects the structure of binary search trees (BST), which is one of the topics covered by the game. Furthermore, the levels, goals, challenges, and game mechanics were designed to be aligned with the learning objectives. The first draft covered six learning objectives and had seven levels.

Concerning the entertainment aspects, the game design document specified the game genre, theme, aesthetics, game environment, narrative elements (game story and dialogues), and game mechanics. The genre, theme, and aesthetics were chosen based on the assets available in the Unity's Asset Store. For example, the science fiction (sci-fi) theme and aesthetics were chosen due to the good quality sci-fi assets available in the Asset Store.

After finishing the game design document, the first prototyping stage was initiated. The first prototype (see Digital Appendix) possessed six levels and covered five learning objectives. A game tutorial was included using in-game text to explain the game mechanics and controllers. This prototype was available only in English. After finishing it, two professors of the Algorithms and Data Structures course at Trinity College Dublin (TCD) tested and provided feedback concerning the content, data structure representation, user interface (UI), and playability.

Based on this feedback, a second version of the game was developed (see Digital Appendix). This version did not require changes to the game design document. All changes to the game were implemented on the basis of the first prototype. The elements that were changed were the UI (e.g., the navigation map, dialogues, and menus) and the game environment, which did not depict the BST model precisely. Furthermore, the second version was translated to Spanish.

To verify the effectiveness of the second prototype, a pilot experiment (Appendix 3) was performed. The experiment evaluated learning gains, perceived learning, clarity of the contents, intrinsic motivation, usability, user experience, usefulness, and presentation of the instructions. Thirty-two engineering students at the University of Costa Rica (UCR) completed the experiment. During the experiment, the researchers guided the participants, answered their questions, and observed the participants' and game's performance. The experiment's results and observations showed that students were motivated and obtained learning gains. However, it also unveiled pedagogical and playability issues that drastically reduced the learning gains.

To fix the issues, we revisited all the design stages and drafted a new game design document (see Appendix 2). Concerning the context, the target audience and the technological resources were modified. In our initial design, we were expecting that students possess a high video game proficiency. However, the pilot experiment results showed that some students struggled with the controls. Additionally, we were expecting laboratories with computers with higher computational power than were ultimately available. This caused performance issues while running the game, which resulted in user experience issues.

Concerning learning aspects, three major modifications were performed: (1) the contents were reduced (e.g., search and in-order tree traverse were excluded), and the learning objectives (LOs) were modified and aligned with the contents; (2) a deeper explanation of the Binary Tree data structure was included; and (3) explanatory images were added to facilitate the understanding of the contents.

In relation to the game, four modifications were implemented: (1) a tutorial level, which allows participants to learn the game mechanics, was included; (2) level goals were divided into sets of small numbers of game challenges; (3) and in-game signages were included. Additionally, the game was compiled with a lower graphic quality to avoid performance issues with computers with lower grade specifications. The third version of the game had an English and Spanish version. This prototype was evaluated with students from University of Costa Rica (UCR) and The National University of Colombia (UNAL; see Chapter 7).

5.1.2 DS-Hacker 2D

The 2D version of *DS-Hacker* was designed and implemented after the evaluation of the 3D version. The design and implementation processes were simpler and faster than the 3D version's process because it did not require the definition of the context, learning aspects, and many sections of the game design document; it was only necessary to modify some dialogues.

The implementation process started with a review of 2D game engines and assets available. The game engines reviewed and tested were RPG Maker MV, GameMaker Studio, and Unity. At the end of the testing stage, RPG Maker MV was selected due to four reasons. First, it is an easy-to-use game engine. The learning curve is gentle, and there is plenty online documentation. Also, it possible to create games without coding. Additionally, the scripting language used by the engine is JavaScript, which is a language that the author knows. Second, RPG Maker MV is relatively inexpensive. Its license cost $79.99, and it comes with a comprehensive pack of assets (e.g., 2D sprite sheets, user interfaces, sounds, basic scripts, etc.). Third, there are many free and paid packages of assets, and the cost of the assets is also relatively low. Finally, there are available many plugins and script libraries that decrease the development time.

After selecting the game engine, the implementation of the game began. First, the game environment of the 3D version was translated to a 2D version. The structure of the mazes and chambers were the same, but they were flattened due to the higher degree of abstraction. The levels, game mechanics, goals, and challenges were implemented for in the 2D environment. All these game elements were the same as the 3D. Finally, the dialogue system and UI was implemented. The 2D version is only in Spanish. Finally, a quick usability test was performed with two students from University of Costa Rica, and a final evaluation was performed with students from UNAL (see Chapter 7).

5.2 DS-Hacker 3D and 2D

5.2.1 Context

Regarding the learning environment, *DS-Hacker* was designed to be used in higher education environments, such as universities or coding schools. Specifically, it targets engineering, computer science, data science, or statistics schools that have programming courses. The game's content was selected based on the guidelines for undergraduate degree programs (ACM; 2013). Therefore, it fits the curricula of most of the introductory courses on programming or data structures.

The location aspects are closely related to the learning environment. The game was designed to be used in classes or laboratories. However, it does not require any guidance from a facilitator or the course instructor to be played. Therefore, the game can also be used as homework.

In relation to the resources, the game was designed for computers with medium processing capacity (e.g., Intel i3 and 8 GB RAM). Consequently, laboratories and students should possess relatively new computers. Due to the courses targeted, it is expected that all laboratories and most of the students have computers with this computational capacity.

Concerning the target population, the game was designed for bachelor's degree students who are taking introductory computer science courses. The students' expected age range is between 18 and 22 years old. They should possess basic knowledge about computers. Also, it is expected that students possess basic knowledge of video games and that they have played at least casual games.

5.2.2 Learning Aspects

5.2.2.1 Theoretical Stance and Pedagogical Approach

The learning aspects of the game was designed from a constructivist stance, which consider that the learner builds his or her knowledge linking new information and familiar experiences. Specifically, assumptions of how people learn were taken from KELT and Kolb's experiential learning cycle (see Chapter 3). Consequently, the game's learning experiences emulate this cycle and links new information with familiar knowledge and experiences. To achieve this objective, the game uses analogies.

Analogies (see section 2.1.4) are an effective approach to teach abstract and non-intuitive knowledge. Many studies have pointed that a considerable number of college students find difficult computer science knowledge because it is very abstract (Danielsiek et al., 2012; Kaczmarczyk et al., 2010; Qian & Lehman, 2017). Considering this, analogies can be effectively used to teach computer science knowledge. Consequently, we decided to make use of them. *DS-Hacker* was designed to facilitate computer science conceptual knowledge through analogies.

5.2.2.2 Content

As mentioned in section 1.1, data structures and algorithms are essential for the development of efficient software (Sedgewick & Wayne, 2014). While a deep understanding of data structures is considered foundational knowledge, studies suggest that misconceptions of basic data structures are common (Danielsiek et al., 2012; Zingaro et al., 2018). Data structures and algorithmic concepts are abstract and difficult to relate to previous knowledge. From a constructivist point of view, it is important that new experiences and information link to previous knowledge in order to create new knowledge (Gogus, 2012). Furthermore, it is necessary that learners engage actively in the learning processes. Traditional approaches such as master class are not enough. Educators should provide experiences and environments where the students can construct knowledge through reflection, critical thinking, and their familiar knowledge (Gogus, 2012). Consequently, a well-designed video game can be learning tools that meet these characteristics, and they can be used to teach data structures and algorithms.

Due to resource constrains, it was not possible to design and implement a video game for learning that covers many data structures and its algorithms. For this reason, our game only focuses on fundamental concepts of Binary Tree (BT) and Binary Search Trees (BST) data structure. BST is a foundational data structure, and it is studied in computer science introductory courses. Its learning difficulty is easy; however, it is more advanced than the queue, stack, or linked list. Additionally, in our review of literature about digital game for learning, we found only one game that specifically focuses on BST data structures.

5.2.2.2.1 Binary Search Tree

According to Sedgewick and Wayne (2014), BSTs are BTs organized in a specific order. A BT is a structure made up of objects called nodes. Nodes contain links that can be null or refer to other nodes. BT's nodes have two links, and each node can be null or point to another node. Furthermore, each node can be pointed to just by one single node. A BST possesses two additional elements called the comparable key and the associated value and satisfies the property that "the key in any node is larger than the keys in all nodes in that node's left subtree and smaller than the keys in all nodes in that node's right subtree" (Sedgewick & Wayne, 2014).

5.2.2.3 Learning objectives

An important aspect of learning tool design is to define the learning objectives. In Chapter 4, we mentioned that all learning experiences must align the learning activities and assessment with their LOs in order to maximize the learning gains. By doing so, students' learning efforts will be optimized, assessment will be fair, and it will facilitate the evaluation of the learning experience's efficacy.

Concerning the game LOs, they cover two topics: BT and BST. As showed above, BSTs are a special case of BTs. For this reason, it is necessary to include BT concepts to teach BST conceptual knowledge. Table 5.1 presents the intended learning objectives.

Table 5.1 Subjects and learning objectives covered by the game.

Subject	Learning Objectives (LOs)
Binary Tree (BT)	LO1. The student should describe the concept of BT. LO2. The student should describe the basic elements that compose a BT node. LO3. The student should identify BTs. LO4. The student should identify the basic elements of a BT.
Binary Search Tree (BST)	LO5. The student should explain the BST property. LO6. The student should identify a BSTs. LO7. The student should solve problems using the BST property. LO8. The student should determine whether the BST property is fulfilled.

5.2.3 Game Description

As mentioned before, *DS-Hacker* possesses two versions that vary in their levels of realism (e.g., graphics and physics). Both versions are for PC. The first version is *DS-Hacker 3D* (Figure 5.1), a third person action-adventure game developed with Unity. Its controllers are the keyboard and mouse, and both controllers are required to play the game. The aesthetics are realistic, and it possesses three-dimensional graphics. Its physics are accurate and can simulate realistic 3D movement.

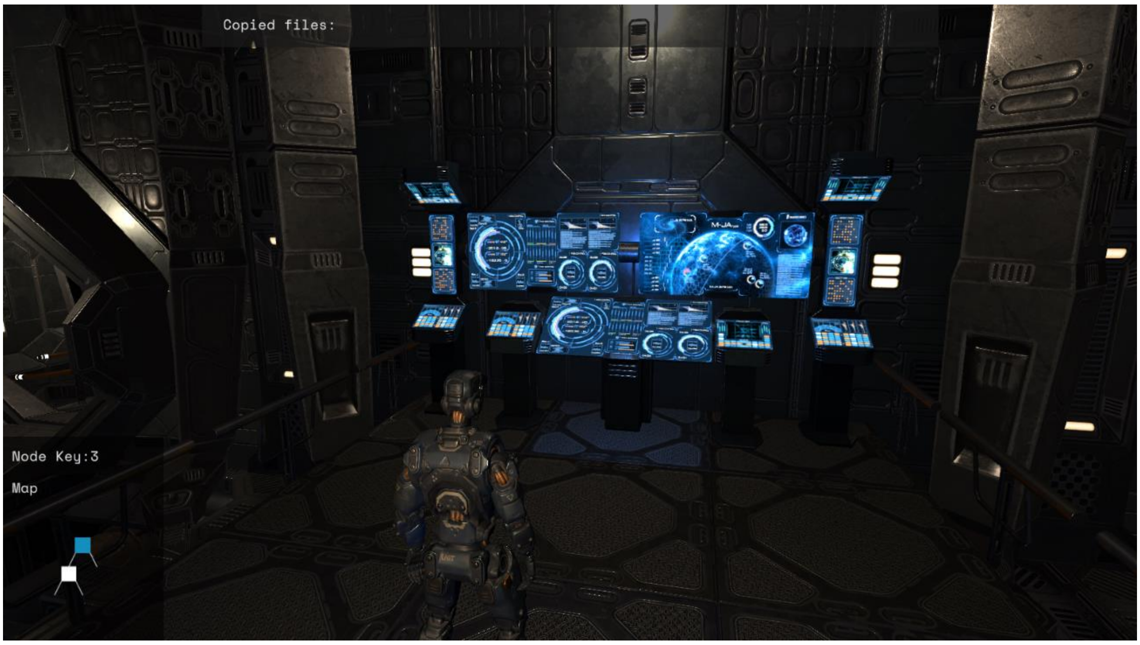


Figure 5.1: Screenshot of *DS-Hacker 3D* ─ Level 1.

The second version is *DS-Hacker 2D* (Figure 5.2), a top-down action-adventure game developed with RPG Maker MV. Its main controller is the mouse; however, the keyboard can be used to move the avatar, but it is optional. The aesthetics are relatively abstract, and it has two-dimensional graphics. The physics are abstract, and the game only represents two-dimensional movements over a flat surface.

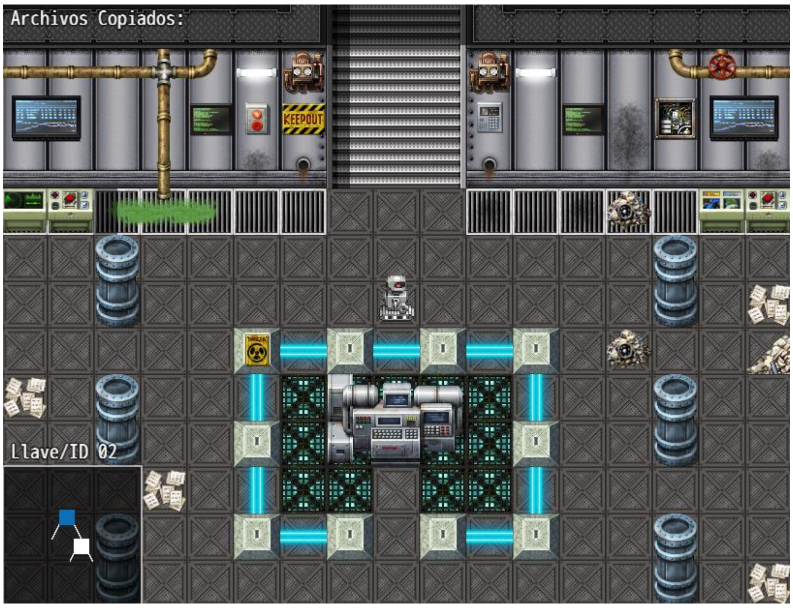
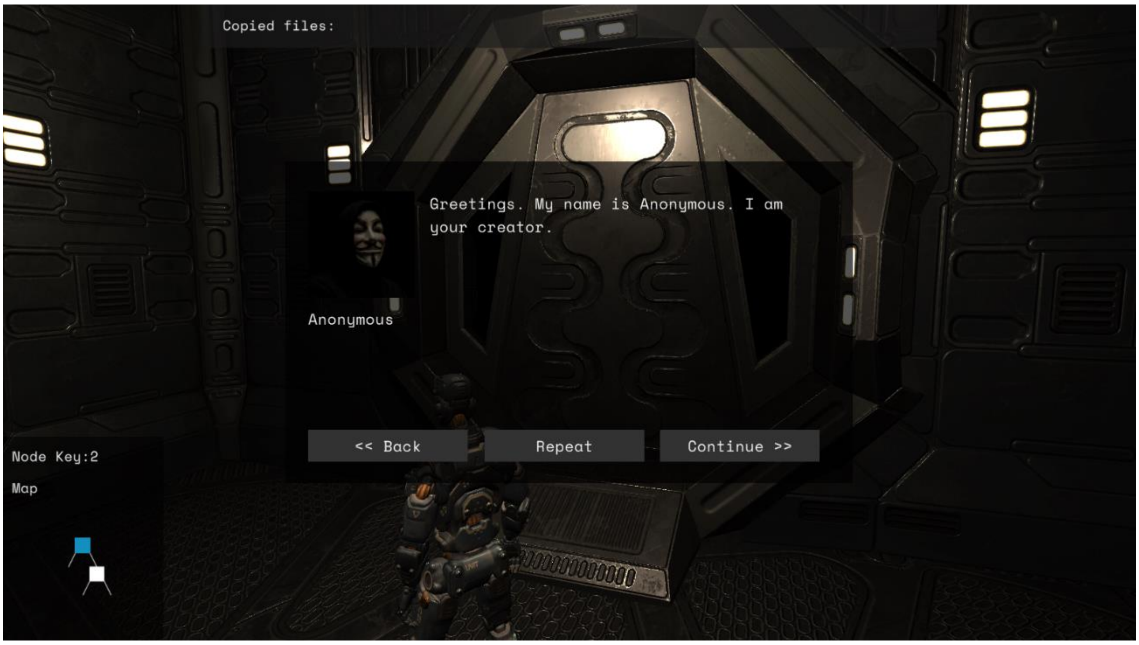


Figure 5.2: Screenshot of *DS-Hacker 2D* ─ Level 1.

The rationale for adapting the action-adventure genre is its flexibility for teaching purposes as well as its popularity. The genre combines game elements from the action genre and the adventure genre, for example by borrowing the story-driven and conceptual puzzle elements from the adventure genre, and the physical challenges from the action genre (Rollings & Adams, 2003). The characteristics of the action-adventure genre have the potential to teach conceptual and procedural knowledge. For example, it is possible to introduce and explain concepts and theory through the game story using narrative elements. The physical challenges and conceptual puzzles can be used to create scenarios where the player can practice the concepts taught by the story. Finally, the action-adventure genre is a well-known genre among teenagers and young adults, and it is considered even more popular than the action and adventure genres themselves (Adams, 2014). This characteristic may decrease the learning curve of the game mechanics to potential players who are already familiar with the genre. Concerning our game, *DS-Hacker* (3D and 2D) is a story-driven game, and its challenges are cognitive puzzles. Its navigation requires advanced mechanical skills, and its implementation supports shooting mechanics. These game mechanics are not available in the current version of the game, but they will be included in later iterations. Due to these characteristics, we classified *DS-Hacker* (3D and 2D) as an action-adventure game.

The theme of the *DS-Hacker* (3D and 2D) game is cyberpunk science fiction, and its story is based on the Hero's Journey structure (Campbell, 1991), which is a narrative progression structure used by many myths and classic tales. The game story takes place in a distant future where a corrupt corporation is harming the balance of society. In the game, the player takes on the role of a robotic hacker who was created by activists to hack the computational systems of corrupt corporations. The robot must traverse several mazes composed of interlinked chambers and extract information stored in each chamber of the computational systems. However, before starting the quest, the robot must be trained and calibrated by its creator, Anonymous (Figure 5.3), a non-player character who guides the player through the game levels. These events start the hero's transformation. We expect that the story and narrative aspects arouse players' curiosity and engagement. Additionally, the story is presented through appealing graphics, music, sound effects, and a futuristic game environment, creating an atmosphere aligned with the game story and intended to increase the players' immersion and fantasy and hence also their intrinsic motivation.



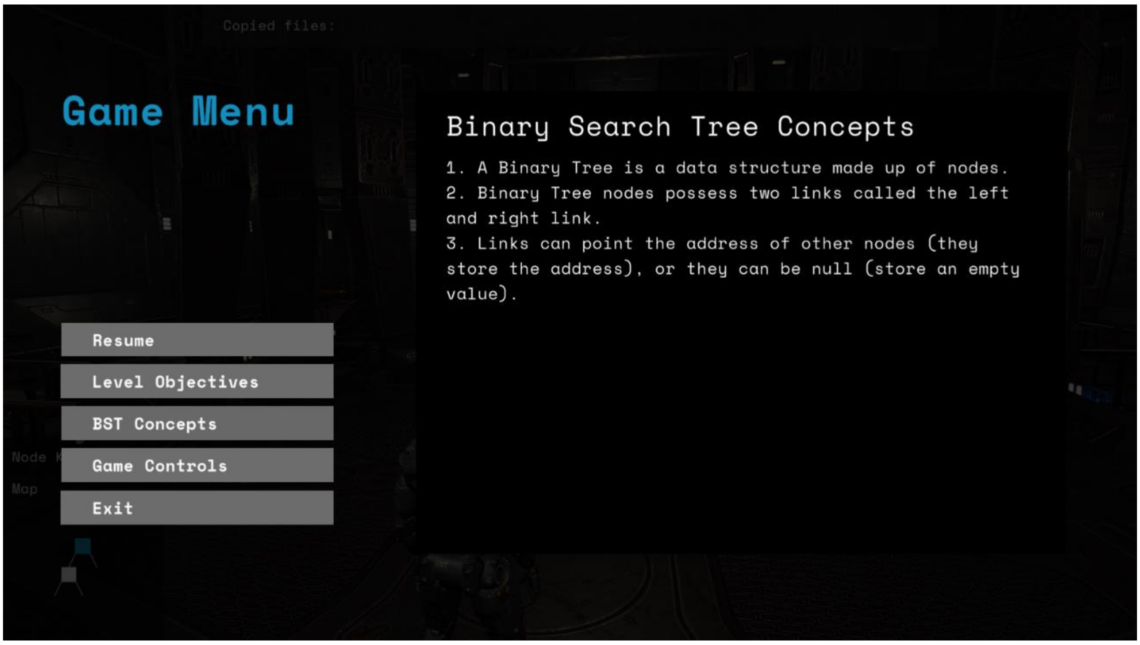
Greetings. My name is Anonymous. I am your creator.

Figure 5.3: Screenshot of *DS-Hacker 3D* ─ Level 1: Anonymous introducing himself to the robotic hacker.

Besides their engagement function, the narrative elements are also a fundamental tool to teach the factual and conceptual knowledge. The game story is told through a monologue given by Anonymous, who appears at the beginning of each level or when the player has completed a challenge. Additionally, through the monologues, Anonymous introduces the challenges and missions of each level and the necessary BT and BST concepts to accomplish them. To facilitate the understanding of the BT and BST concepts, Anonymous uses analogies between the game environment and the BT and BST data structures. The purpose of these analogies is to enable the creation of new knowledge through familiar knowledge.

The game environment of *DS-Hacker* (2D and 3D) reflects the structure of BTs or BSTs. According to the game plot, a robotic hacker controlled by the player must enter and traverse mazes called "*Data Systems*." In the game world, *Data Systems* are places where corporations hide and protect their information. The structure of these systems follows the same organization as well-known data structures. For the game learning objectives, the *Data Systems* reflect the structure of BTs and BSTs, and many elements of the game environment represent important elements of these data structures. In this regard, mazes that represent a BST consist of rooms that represent the BST nodes. The rooms are organized following the BST property, and they possess portals, an ID, and a central computer. Additionally, rooms have ornamental features aligned the theme of the game, and a structure that allows navigation and exploration. The portals of each room represent the links that point to other nodes or null values. The room ID represents the BST comparable key, and the information stored in the central computer represents the node's associated values.

In addition to the environment and narrative elements, *DS-Hacker* (2D and 3D) uses the UI to reinforce the BT and BST concepts. For example, the map of the game environment is a visual representation of a BST. The map is part of the game's heads-up display (HUD), and it is based on the BT and BST representation used in textbooks (Aho, 1988; Sedgewick & Wayne, 2014). Another example is the pause menu. One of its options displays cards with the concepts taught in the level. The player can pause the game and check the definitions to solve the tasks. In *DS-Hacker*, the UI is utilized to reduce the cognitive load of the player by offering reminders regarding BT and BST concepts. Figure 5.4 presents the concepts provided in the second level of *DS-Hacker 3D*.



Binary Search Tree Concepts

1. A Binary Tree is a data structure made up of nodes.
2. Binary Tree nodes possess two links called the left and right link.
3. Links can point the address of other nodes (they store the address), or they can be null (store an empty value).

Figure 5.4: Screenshot of *DS-Hacker 3D* ─ Level 2: BT and BST concepts in the main menu.

Currently, *DS-Hacker* (2D and 3D) has four levels. In level one, the player is introduced to the game controllers, UI, environment, and story of the game. This level does not possess any learning objective related to the BT or BST data structure, and the level of difficulty is low. All challenges are related to environment exploration. The purpose of this level is to allow players who do not have any experience with video games or the action-adventure game genre to get used to the game controls and to develop the necessary navigation skills. Additionally, it allows the players to get familiar with the arrangement of the game environment in case they find it difficult to navigate.

Table 5.2: Levels, topics, intended learning objectives (LOs), and learning activities (LAs) of *DS-Hacker*.

Level	Topics	LOs	Learning Activities
Level 1	No topic	No LOs	No learning activities.
Level 2	Binary Tree node	LO1. The student should understand the concept of BT. LO2. The student should define the basic elements that compose a BT node.	LA1. Read and listen the BT definition. LA2. Read and listen the node definition and its basic components. LA3. Read and listen the link definition (reference/pointers). LA4. Relate the portals (links) with the concept of reference/pointers. LA5. Relate the chambers of the game environment with the BT node structure and components.
Level 3	Binary Tree structure	LO3. The student should identify BTs. LO4. The student should identify the BT's components	LA6. Read and listen the definition of the left and right child. LA7. Read and listen the definition of the parent node. LA8. Read and listen the definition of a sub-tree. LA9. Identify the left and right child of the BT represented by the game environment. LA10. Identify the root node of the BT represented by the game environment.
Level 4	Binary Search Tree	LO5. The student should explain the BST property. LO6. The student should identify the BSTs. LO7. The student should solve problems using the BST property. LO8. The student should determine whether the BST property is unfulfilled.	LA11. Read and listen the definition of the basic components of a BST. LA12. Read and listen the definition of the BST property. LA13. Apply the BST property to search for specific nodes

In the second level, the player is introduced to the concept of BTs and their fundamental components. In this level, Anonymous explains the BT data structure, its nodes, and node's components. Then, Anonymous instructs the player to identify the elements of the game environment that represent the BT components. In the third level, the player is introduced to the BT structure and the names of its parts. Then, Anonymous instructs the player to find the left and right child of the chamber, which represents a BT node. In the fourth level, the player learns about the BST data structure, its unique components, and the BST property. In this level, Anonymous requires that the player extract the information stored in certain nodes. To achieve this objective, the player should apply the BST property. Table 5.2 summarize the topics covered by each level, the learning objectives, and the learning activities (game mechanics and challenges) that the player must perform during a gaming session.

Finally, Table 5.3 summarizes the aspects that are the same in both versions of the game and the aspects that differ between versions.

Table 5.3: Summary of aspects of the game that are the same or that differ.

Aspects that are the same	Aspects that differ between versions
• Context o Target audience • Learning Aspects o Learning theory o Pedagogical approach o Content o Learning objectives o Learning activities • Game aspects o Game genre o User interface and content presentation o Music and sound effects o Narrative elements (story and dialogues) o Levels and challenges	• Context o Computer specs • Game aspects o Graphical representations o Game physics o Game environment (game world) o Game mechanics o Control inputs

5.3 Discussion

DS-Hacker is a digital game for learning that intends to teach BTs and BSTs by applying four principles. First, the game aims to link new information with familiar knowledge. According to the constructivist paradigm, this principle is crucial for creating new knowledge. To achieve this objective, the game takes advantage of a well-known game genre (action-adventure) to provide environments and game mechanics that are familiar to learners. The game model is utilized as an analogical representation of the structure of the BT and BST. Furthermore, to facilitate the understanding of the relation of the two domains (game elements and data structure concepts), *DS-Hacker* utilizes analogies embedded in the narrative elements of the game. As shown above, analogies are useful for teaching abstract concepts, such as data structures, because they illustrate the relation between two symmetrical domains. We expect that by using these three approaches, the learners may create links that will facilitate the understanding of the BST concepts and the creation of new knowledge.

Second, *DS-Hacker* aims to promote active learning. To achieve this, it was designed to offer a learning environment that embodies the Kolb’s experiential learning cycle. For example, at the beginning of each level, the game offers conceptual information to the player. The player should use this information to plan a strategy to solve the missions. Then, the player should execute the strategy that will provide new experiences. Then, the player should consider and rationalize these new experiences, restarting the cycle. We anticipate that DS-Hacker ensures active learning by means of this principle.

Third, *DS-Hacker* was designed such that the learning activities (game mechanics and challenges) are aligned with the learning objectives. This characteristic ensures that learners will engage in activities that will facilitate the acquisition of learning gains. Additionally, it will facilitate the assessment of the game and the creation of the knowledge assessment tool.

Finally, *DS-Hacker* was designed to promote engagement and intrinsic motivation, which are related to higher quality learning gains. According to Malone and Lepper (1987), fantasy, curiosity, challenges, and control are fundamental ways to increase intrinsic motivation. *DS-Hacker* employs all these elements. The game offers an appealing story, theme, and aesthetics that invite players to immerse, explore and empathize with the game world and characters. Additionally, the game offers the players several levels and non-trivial challenges. Finally, the game provides a tutorial that allows the player to learn and practice the game mechanics. This provides the player with the necessary skills to control the avatar and solve the challenges. Furthermore, the player can decide to accomplish the challenges or explore the game environment. We expect that these characteristics will increase players intrinsic motivation and engagement.

In the next chapter, we will describe the BST conceptual knowledge assessment tools that we used to measure learning during the evaluations of DS-Hacker (2D and 3D).

Chapter 6: Assessment of Knowledge

A fundamental aspect in education and education research is the assessment of knowledge learning. Good assessment allows researchers and instructors to compare students' performance across learning experiences, which in turn makes it possible to assess the efficacy of the pedagogical approaches used by the learning experience. This raises SRQ 3: *How can conceptual knowledge learning be measured accurately?* To answer this question, we paid special attention to the validity of measure, which is the evidence that justifies the use and interpretations of the scores of an assessment.

In this thesis, assessment is used to measure students' knowledge before and after completing a game-based learning experience. Specifically, it is used for experimental purposes; it is used to evaluate the efficacy of *DS-Hacker*. For this reason, the assessment tool should allow us to assess a large number of participants simultaneously and consistently. Additionally, the tool should allow us to revise the answers in a fast and economic manner and to apply inferential statistical methods, which will allow us to make predictions based on our data.

Initially, a "ready-made" validated assessment tool was sought, and a number of data structure concept inventories (CIs) were found (e.g., Danielsiek et al., 2012; Farghally et al., 2017; Karpierz & Wolfman, 2014; Porter et al., 2019). CIs are validated assessment instruments that focus on the misconceptions of precise topics within a domain, and they are easy to use for researchers and instructors (Tew & Guzdial, 2011).

Concerning the CIs that we found, there were multiple problems. First, the CIs related to BSTs were not available, meaning that the designers of the CIs did not publish the items or tests. Therefore, it was not possible to use them for our evaluations. Second, their scope was broader than the required for the experiments, i.e., the CI developed by Porter et al. (2018, 2019) covers six data structures: linked list, BT, BST, stack, list, and set. Additionally, CIs may exclude many basic concepts. For our purpose, these concepts are important to know in order to understand whether our game is effective. Consequently, we decided to develop an assessment tool specifically to measure knowledge changes after a learning experience. Furthermore, the development process had to be easy and fast.

This chapter presents the methodology followed to develop and validate a tool to measure BST conceptual knowledge. The methodology is divided into three stages: First, the test specifications were formulated. Second, the items were drafted and assessed by experts. Finally, statistical validation of the items was performed through the Rasch Model (RM). Our results show that the instrument produces precise and reliable scores with interval data properties, which can be used to generate valid statistical inferences.

In the next section, the importance of validity is explained. Then, a review of literature related to assessment validation is presented. After that, the methodology and results are presented. Finally, the discussion and limitations of our tool is provided.

6.1 Rationale of Validity

The validity of measures in education is important because it constitutes the evidence that justifies the use and interpretations of the scores of an assessment (American Educational Research Association, 2014; Bond & Fox, 2015). Valid measurements allow students' learning outcomes to be compared in a consistent manner across pedagogical approaches, instructors, institutions, and curricula (Taylor et al., 2014). Many works in computer science education (CSE) and game-based learning do not adopt best validation practices or do not adopt strictly correct use of assessment tools. For example, Randolph et al. (2008) observed that it is common practice in CSE to measure knowledge gains through attitude and perception items, and only one of 65 articles surveyed reported using a validated assessment tool. Sanders et al. (Sanders et al., 2019) found that most of the works that applied inferential statistical analysis did not verify the numeric properties of the data obtained with their instruments, which is assumed by the statistical tests. Concerning game-based learning, Petri and Gresse von Wangenheim (2017) found that many studies did not use validated instruments and in some cases those data collected with those instruments were analysed with inferential statistics. Additionally, the results of our literature review (Chapter 2) show that most of the evaluations of video games for learning data structures did not use validated assessment tools. Such findings show a lack of rigorous analysis. Assessment validation improves research quality by justifying the inferences made with the assessment scores.

6.2 Background

In the field of CSE, initial efforts were made by Deek et al. (1999) who presented a set of tools to measure programming and problem-solving skills. Validity was assessed through interviews with experts and through statistical analysis. Almstrum et al. (2006) presented a review of literature related to the history and characteristics of CIs, and they also proposed a comprehensive methodology to develop a CI for discrete mathematics.

Later, Tew and Guzdial (2010, 2011) reported the development process of an instrument to measure knowledge of fundamental concepts of introductory computer science courses. Topics such as variables, logical operators, selection statements, loops, arrays, function parameters and return values, recursion, and object-oriented programming were covered. The items were language independent. Validity was assessed by a panel of experts in computer science, through student interviews, as well as statistical analysis using Item Response Theory.

Similar to the previous work, Caceffo et al. (2016) reported the implementation of a CI for introductory programming. The evaluated concepts were selected using different approaches. First, an analysis of previous exams and selection of tentative misconceptions were performed. Then, there were interviews with five experienced instructors to review the tentative misconceptions. After this stage, Caceffo et al. drafted a bank of items, covering topics such as function parameters, variables, identifiers, recursion, iteration (loops), pointers, Boolean expressions, and syntax. Items were written using a C-like language.

In relation to data structures, Danielsiek et al. (2012) presented the initial stages of the development process of a CI for algorithms and data structures. Specifically, they identified concepts that generate misconceptions and developed the pilot items. Concepts were selected by analysing previous exams and open-ended questions developed by Danielsiek et al. and administrated to first-year students. Subjects covered were Heaps and BSTs, divide-and-conquer algorithms, dynamic-programming, and invariants. The items were administered to university students, who were asked to discuss possible answers to the items. Statistical validity was not reported.

Karpierz and Wolfman (2014) reported the development process of BST and Hash Table items for a CI based on student misconceptions. Their methodology consisted of three stages: (1) identification of key concepts, (2) identification of misconceptions, and (3) formulation, testing, and validation of multiple-choice CI questions. Their data sources included interviews, exam statistics, exam analysis, project analysis, think-aloud interviews, and a CI pilot. Their results show that students had misconceptions related to BST duplicated keys, BST structure, and the hash table resize algorithm.

Farghally et al. (2017) described the development and validation process of a CI for algorithm analysis topics. Content selection and validation were executed using the Delphi process with a panel of experienced instructors. Fifteen concepts and seventeen misconceptions concerning mathematical foundations, growth rates, running time analysis, upper and lower bounds, and recursive analysis were selected. Then, a 10-item test was drafted addressing the concepts and misconceptions. Items were administered to 294 students. Cronbach’s-alpha coefficient (0.84) suggested consistency among the items.

Porter et al. (2019) reported a comprehensive development and validation process of a Basic Data Structure CI. Content and misconceptions were carefully validated (Porter et al., 2018; Tew & Guzdial, 2010). Seven data structures (Set, Stack, Linked List, Binary Tree, and BST) were selected and evaluated through six learning outcomes. The structure of the items was judged by experts, and statistical validity was analysed with Item Response Theory (Porter et al., 2019).

This review shows the published options of valid instruments to measure data structures knowledge. These instruments, mostly CIs, have been demonstrated to be effective and thoroughly validated. The methodology of the works reviewed focus primarily on the identification of misconceptions and validation of the items' content. However, only few works (e.g., Porter et al., 2019) pay attention to the statistical validation. Furthermore, the contents' scope of the CIs reported cover more topics than our project's learning objectives. For this reason, we decided to develop an instrument for educational and scientific experimental research on BST topics.

6.3 Methodology

Our methodology is based on the Standards for Educational and Psychological Testing (2014), which provide the guidelines and principles to develop educational and psychological tests. However, we adapted the guidelines to develop an assessment tool using multiple-choice questions. The main stages of the methodology are: (1) development of the test specifications; (2) development and validation of the items; and (3) testing and statistical validation of the items. Each stage possesses more than one sub-stage. In the following, we detail the three stages.

6.3.1 Test Specifications

The test specifications state the requirements that the test must meet after the development process. These specifications consider five aspects. The first aspect is the general considerations, which are closely related to the context of *DS-Hacker* (see section 5.2.1). These considerations express the test purpose, intended users, intended use, construct/content domain, and examinee population. The test purpose is to measure BT and BST knowledge gains; the intended users are researchers and educators; the intended use is to facilitate assessment of knowledge before and after playing *DS-Hacker* 3D and 2D; the domain is BT and BST fundamental knowledge; and the examinee population consists of first- and second-year university students enrolled in programming courses.

The second aspect is the content specification, which is closely related to the learning aspects of the game (see section 5.2.2). The content addresses the topic and learning objectives (LOs). The topic and LOs are the same as those covered by *DS-Hacker*, and they are explained in section 5.2.2. These specifications were based on the curricula suggestions (Association for Computing Machinery (ACM) Joint Task Force on Computing Curricula & IEEE Computer Society, 2013), previous studies about difficulties learning computer science (Qian & Lehman, 2017; Tew & Guzdial, 2011), and “well-known” algorithm textbooks (Aho, 1988; Sedgewick & Wayne, 2014). Additionally, Bloom’s taxonomy (Anderson & Krathwohl, 2001) was employed to draft eight LOs, which are aligned with our game for learning.

The third aspect is the format of the items and scoring system. Multiple-choice questions were selected because they are easier and faster to validate, administer, and evaluate than other item formats. A dichotomous scoring model (1: success and 0: failure) was used to simplify the evaluation of the scores.

The fourth aspect is the psychometric specifications of the model that is going to be used to calibrate the items statistically. The Rasch Model (more details in section 6.3.3) was selected because it generates invariant measures among different groups (Bond & Fox, 2015).

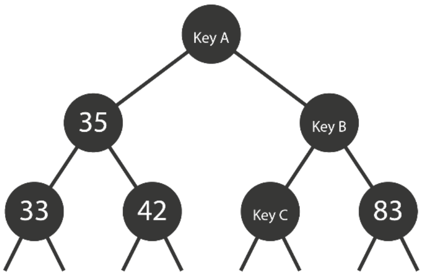
Finally, the fifth aspect is the administration considerations. The test was intended to be applied online; therefore, it required a computer or mobile device with Internet access. The test does not have a time restriction.

6.3.2 Items Development and Validation

The item development stage was divided into three substages: (1) drafting the items; (2) validating the content and quality; and (3) correction or elimination of items. Concerning the drafting substage, all the items were written in English and Spanish by the author. The items were based on the LOs and the textbooks’ contents. Furthermore, the item drafting followed best practices to build multiple-choice questions suggested by Haladyna et al. (2002) and Moreno et al. (2004). The structure of the items consists of a stem, a correct answer, and three distractors. Table 6.1 presents two examples: The first example assesses BT knowledge, and the second one assesses BST and includes an image. At the end of this substage, twenty-three items were drafted, and on average, each LO was verified by two or three items.

Regarding the item validation, a survey was conducted with six experts. The experts were six Computer Science professors from the University of Costa Rica with several years of experience teaching Programming or Data Structures and Algorithms courses. The survey verified the drafting quality of each item and alignment with the LOs. Additionally, the survey included one open-ended question asking for feedback to improve each item (see Appendix 4.1).

Table 6.1: Examples of the items and their structure.

Example 1	Binary Tree
Stem	Select the option that better defines the concept of Binary Tree.
Correct answer	• A Binary Tree is a null link or a node with a left link and a right link, each reference to subtrees that are themselves binary trees.
Distractors	• A Binary Tree is a data structure made up of nodes that possess multiple links that point to other nodes, and they cannot be null. • A Binary Tree is a data structure that possesses two nodes with a left and right link that point to objects that do not belong to the data structure, and they can be null. • A Binary Tree is a data structure implemented using a bidimensional array that stores null values or the index value of another array's elements.
Example 2	Binary Search Tree
Stem	Consider the following Binary Search Tree.  Select the option that correctly completes the values of the comparable keys of the previous Binary Search Tree.
Correct answer	• Key A = 50; Key B = 77; Key C = 60
Distractors	• Key A = 36; Key B = 43; Key C = 37 • Key A = 20; Key B = 37; Key C = 44 • Key A = 91; Key B = 82; Key C = 78

Finally, items that were advised to be removed or rewritten in the open question were rewritten or eliminated. At the end of this substage, eighteen items (see Appendix 4.2) were redrafted considering the feedback provided by the experts.

6.3.3 Statistical Validation

The validation consists of two sub-stages: (1) testing the items; and (2) performing a statistical analysis of the results to select the items that fit the model of measurement.

6.3.3.1 Pilot Testing

The sample was a group of Computer Science undergraduate students of all academic years at the National University of Costa Rica and the University of Costa Rica. Students were recruited by email and participation was voluntary; no incentives were given. The invitation contained a link to a custom-made online survey application created using React. Additionally, this application collected the number of window changes and the time taken to answer each question to control consistency of the respondents' behaviour. The answers were stored on Cloud Firestore in JSON format. In total, 220 undergraduate students completed the questionnaire online.

6.3.3.2 Statistical Validity

The Rasch Model (RM) is a mathematical framework to obtain interval data of non-observable psychological traits (Bond & Fox, 2015). The RM allows researchers to make meaningful, precise, and reliable inferences of respondents' performance (Cronbach & Meehl, 1955). Specifically, the RM describes the probability of success on an item (P_{ij}) based on the difference between the respondent's ability (ϑ_i) and the item's difficulty (δ_j). The level of ability is defined by the respondent pattern of answers (Bond & Fox, 2015). If X is an n × p data matrix with n individuals in the rows (i = 1, …, n) and p items in the columns (j = 1, …, p), for dichotomously scored items, the RM is defined as:

$$P\left(X_{ij} = 1\right) = \frac{1}{1 - e^{(\vartheta_i - \delta_j)}}$$

To fit the items to the model and visualize the results, the R 4.0.1 compiler, R Studio, and R package *eRm* were utilized. Furthermore, the RM requires a set of assumptions on the data to ensure the validity of interpretations made using the scores. These assumptions are *unidimensionality*, *local independence*, *invariance*, and *monotony*.

Unidimensionality guarantees that all the items measure just the intended ability. Dimensional reduction techniques were applied to determine whether this condition was satisfied, specifically, Principal Components Analysis (PCA) and Exploratory Factor Analysis (EFA) using the tetrachoric correlation matrix. To perform the PCA, the R package *Gifi* and *FactoMineR* were used to calculate the components and *factoextra* was used to plot the results. Concerning EFA, the following criteria was evaluated: (1) Acceleration Factor (AF); (2) Optimal Coordinates (OC); and (3) Parallel Analysis (PA). The R package *nFactors* was used to obtain the AF and OP, and the package *psych* was used to perform the PA. Also, fit with respect to the RM was verified using the standardized infit MSQ-index, which is another indicator of unidimensionality violations (Bond & Fox, 2015). The R package *eRm* was employed to obtain the infit MSQ-index.

Local independence means that the success probability of an item is independent of the success probability of another item when the knowledge estimate is controlled. By verifying the unidimensionality, local independence can be confirmed (Martínez Arias et al., 2006).

Invariance implies that the item parameters must be approximately the same on any sub-group split (Mair, 2018). The Andersen's Likelihood Ratio (LR) Test method was used to check this condition with the Wald test to identify the problematic items when the assumption was violated (Mair, 2018). To perform the LR test, data was divided into two categories using the participants' total score. The *eRm* package was used to perform and plot the LR and Wald tests.

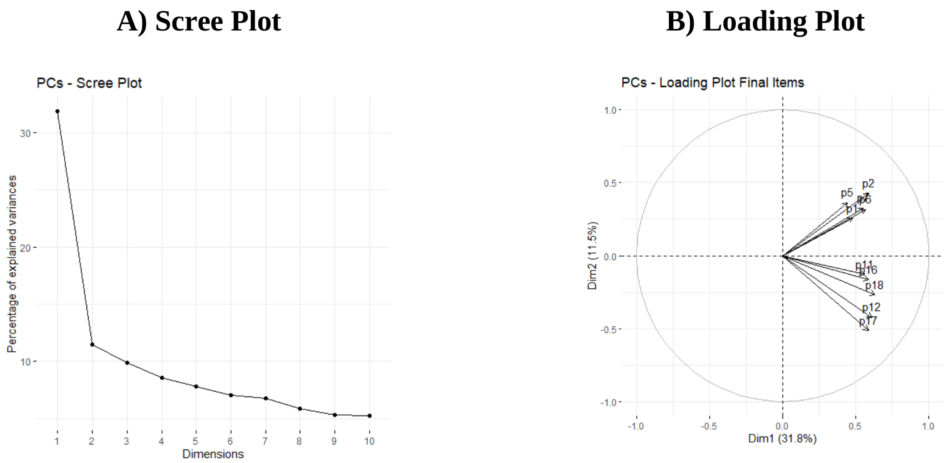
Finally, monotony implies that the expected item score should be a monotonically increasing function of the latent score (Nielsen, 2018). The Item Characteristic Curve (ICC) plot is used to visualize this feature (Bond & Fox, 2015). The ICC was plotted using the *eRm* package.

Studies often report the Cronbach's alpha as a reliability statistic. In this context, reliability is understood as the consistency among repeated measures. In practice, an alpha between 0.8 and 0.9 is intended; values between 0.7 and 0.8 are acceptable, and values over 0.9 suggest redundancy among the items (Cronbach & Meehl, 1955). To calculate the Cronbach's alpha, the R package *ltm* was used.

6.4 Results of the Statistical Validation

Initially, some items did not fit the RM, and some assumptions were not achieved. Therefore, an iterative depuration of the scale was followed by checking the model fit measures and assumptions and eliminating the misfitting items. Best practices suggest that this process should be done one item at a time (Mair, 2018). After this process, eight items were discarded, resulting in a 10-item scale (see Appendix 4.3), which fits the RM and fulfils the RM assumptions.

Regarding the PCA, results show that the first and second Eigenvalues are 3.18 and 1.15, respectively. This suggests that the first Principal Component (PC) explains 31.84% of the variance, and the second PC explains 11.48% of the variance. The Scree plot (Figure 6.1 A) shows the percentage of explained variance (Eigenvalues) of the Principal Components (PCs). The plot shows a drop in the PCs after the first PC, denoting unidimensionality. However, the second PC explains a significant amount of the variance, suggesting that it may be considered. The PCs loadings plot (Figure 6.1 B) depicts the relation between the items and the first two PCs, and it shows two clusters of vectors. Additionally, the plot shows that one cluster consists of the items about binary trees, and the second cluster consists of the items about binary search trees. However, all vectors are located along the right side of the x-axis, denoting a strong relationship with the first PC.



A) Scree Plot

B) Loading Plot

Figure 6.1: (A) Principal Components Scree plot. (B) Principal Components loading plot of the final 10-items version scale.

Concerning the EFA, results shows that the first two Eigenvalues of the tetrachoric correlation matrix, 4.74 and 11.7, suggest that existence of two factors. However, results of the AF, OP, and PA suggest the existence of only one factor (Figure 6.2).

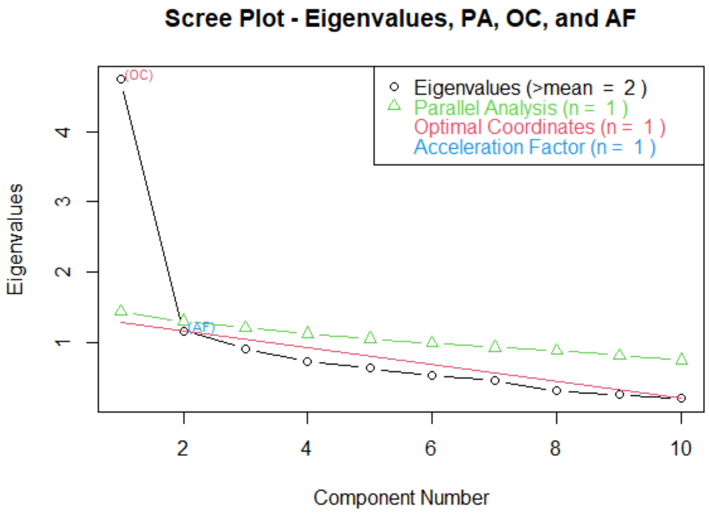


Figure 6.2: Scree plot of the eigenvalues, PA, OC, and AF of the Exploratory Factor Analysis

On the items' fit to the RM, the location of the standardized infit mean square (t) of each item are shown on the x-axis of the Figure 6.3. The tolerance thresholds are demarked by the vertical lines at -2 and 2. Values below -2 denotes over-fit of items, while values above 2 denotes misfit. The 10 items of the scale are between these boundaries, suggesting unidimensionality. Based on the PCA, EFA, and item's fit to the RM, we conclude that the unidimensionality assumption is met.

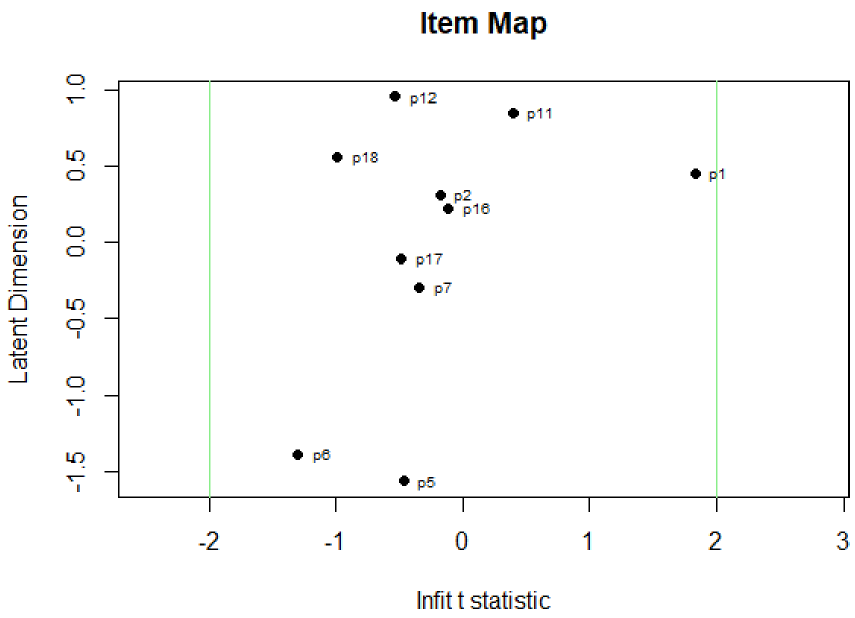


Figure 6.3: Rasch model fit of the final 10-items version scale.

To verify the invariance assumption, the median of the total score was used to split the answers in two samples. Then, the Andersen's Likelihood Ratio test was applied, looking for invariance among the subgroups. Results show that the difference between groups was not statistically significant (*LR-value* = 8.18, *Chi-square df* = 9, *p-value* = 0.52), suggesting that the invariance assumption was met.

The fitted model for each item is displayed by the Item Characteristic Curve (ICC). Figure 6.4 shows that all curves are monotonically increasing functions of the latent dimension of conceptual knowledge (standardized scores). It is possible to describe the probability of solving each item with the estimated scores of the respondents using the scale.

With respect to reliability, the Cronbach's alpha is 0.76. In practice, values over 0.7 are acceptable for research purposes, while values between 0.8 and 0.9 are required for high-stakes evaluations. The Cronbach's alpha of the knowledge scale is acceptable for research and close to the standard required for high-stakes evaluations.

The distribution of the scores is shown with each item's location along the same latent dimension in the person-item map in Figure 6.5. This allows the comparison of each item's difficulty with respect to the frequency of participant achievement. The person-item map shows that the difficulty of the test is medium-low.

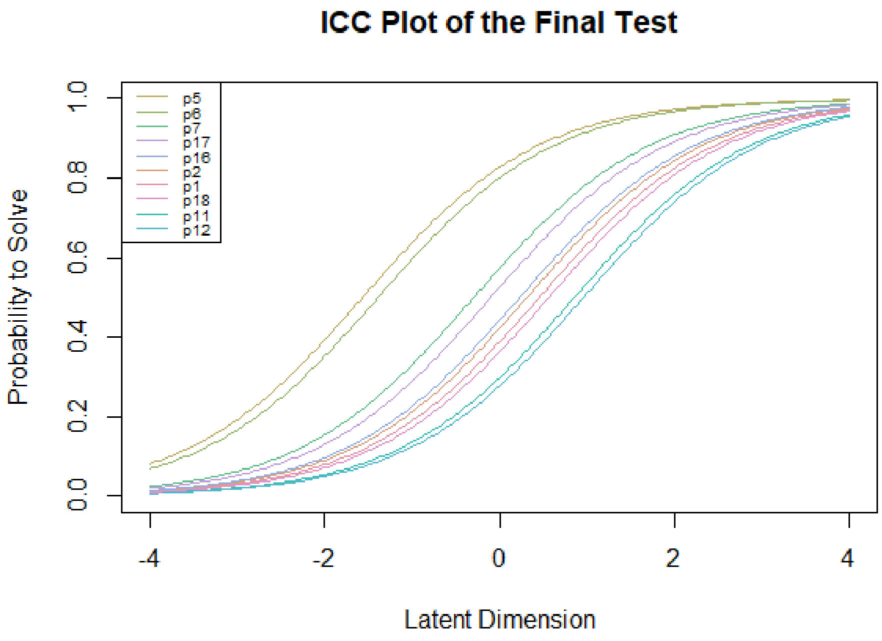


Figure 6.4: Item Characteristic Curves of the final 10-items test.

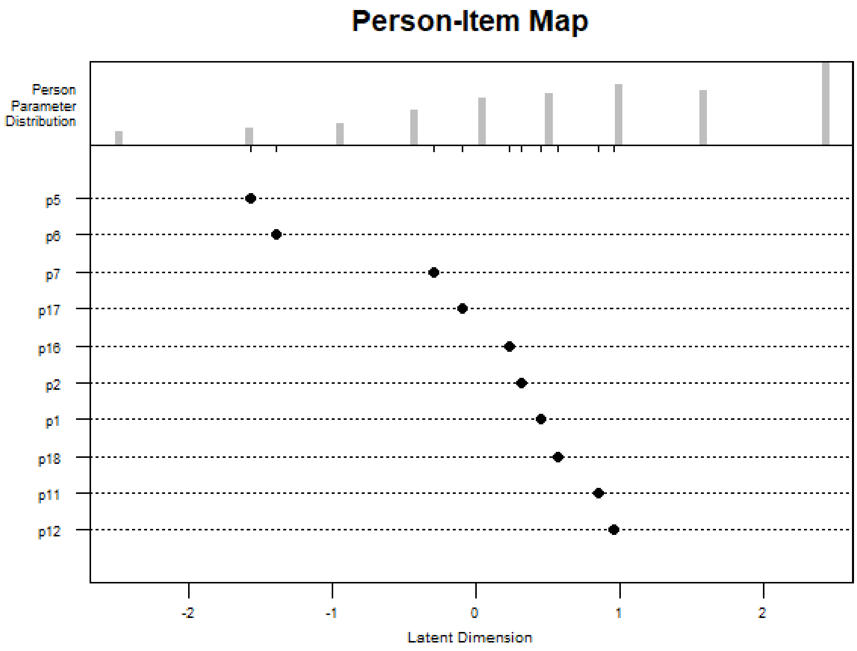


Figure 6.5: Person-Item Map with the final 10-items test.

6.5 Conclusion and Limitations

This chapter has presented the methodology and results for creating an assessment tool that allows valid inferences about learners' conceptual knowledge about BSTs. For this purpose, test content was selected and validated to meet the *DS-Hacker*'s LOs. Internal consistency was corroborated by the unidimensionality of the contents and calibrating the RM. This evidence was necessary to use the evaluation scores as continuous variables for statistical purposes properly.

Studies indicate that a considerable number of investigations conducted statistical analysis without considering the type of data obtained by their instruments. This can lead to misinterpretation of the results obtained during the analysis, decreasing the validity of the inferences made by those researchers. Validating the research instruments improves the precision and reliability of the conclusions inferred from the analysis; conclusions using valid scores are better justified. Results show that our assessment tool generates interval data.

Related to our scale, even though the contents are basic, all items have considerable variance, and they distinguish novice students from advanced students who are supposed to have greater knowledge about data structures. The scale covers all the LOs of *DS-Hacker* and allows us to understand which objectives are properly addressed by the learning activities embedded in the game and those that require modification. One limitation of the scale is a gap between -1.5 and -0.5 on the standardized score of the knowledge level (see ICC in Figure 6.4). Learners with a level of knowledge between those values of the scale are measured with less precision. Additionally, third- and fourth-year undergraduate students, who are not ideal subjects, were included in the sample to calibrate the items.

Chapter 7: Evaluation

In this chapter, we assess the efficacy of *DS-Hacker* 3D and 2D as a learning tool for higher education environment. We have structured this task into two evaluations. The first evaluation (section 7.1) assesses and compares *DS-Hacker* 3D to a "traditional" digital learning experience. The second experiment (section 7.2) assesses and compares *DS-Hacker* 3D to *DS-Hacker* 2D, searching for variations caused by the different levels of realism. Furthermore, both evaluations assess two factors: learning gains and intrinsic motivation. Also, these evaluations will be the final validation stage of the CLG framework. Positive results will indicate that the framework has the potential produce good video games for learning effectively. These will be discussed in section 7.3.

7.1 Evaluation 1: Learning and Motivational Effects

The first evaluation studies the learning and motivation gains obtained through digital game-based learning. The evaluation aims to answer SRQ 4: *Are video games for learning that use analogies to teach conceptual knowledge more or less effective and motivating than other traditional digital approaches?* For this purpose, *DS-Hacker 3D*, is evaluated and compared to a "traditional" digital learning experience. Furthermore, SRQ 4 is divided into three precise questions that consider the target population and traditional digital learning experience evaluated in the experiment. The questions are:

SRQ 4.1. Do university students learn conceptual knowledge about BSTs using *DS-Hacker 3D*?
SRQ 4.2. Do students learning with *DS-Hacker 3D* obtain higher scores than students working with a traditional digital learning experience?
SRQ 4.3. Do students learning with *DS-Hacker 3D* obtain higher scores in an intrinsic motivational test than students working with a traditional digital learning experience?

It was hypothesised that *DS-Hacker 3D* will obtain positive results. Students were going to increase their learning gains regarding BST conceptual knowledge after playing the game, and the students' intrinsic motivation was going to be high. Also, it was hypothesised that students who play *DS-Hacker 3D* were going to obtain higher scores in a conceptual knowledge test than students who complete a traditional digital learning experience. Finally, it was hypothesised that students who play *DS-Hacker 3D* were going to obtain higher scores in an intrinsic motivation test than students who complete a traditional digital learning experience.

To confirm or reject the hypotheses, an evaluation was performed during August and September of 2020 at University of Costa Rica (UCR). Due to the Covid-19 pandemic, UCR was placed on lockdown, and classes were taught online. For this reason, the experiment was performed virtually, and all materials and tests were distributed and completed using a custom-made web application. The evaluation was conducted using the final version of *DS-Hacker 3D*, which was described in Chapter 5.

7.1.1 Materials and Methods

7.1.1.1 Participants

Students from the bachelor programme of the Industrial Engineering Department at University of Costa Rica were selected to participate in the evaluation. The sample met the characteristics of the game's target population, which is students who have basic programming skills. In total, 54 students participated in the experiment and completed all the surveys and tests. Participants were randomly assigned to a control or an experimental group. The control group had 28 participants (50.9%), and the experimental group had 27 participants (49.1%). Regarding their demographic background, 33 participants (61.1%) were female, and 21 participants (38.9%) were male.

7.1.1.2 Materials and Data Collection Tools

To measure Binary Search Tree conceptual knowledge changes, a custom-made test about BST conceptual knowledge was used (see Appendix 4.3). The test had 10 multiple choice questions (items) with three distractors and one correct answer. The test allowed the researchers to distinguish between students who understand and can apply BST conceptual knowledge from those students who cannot. The test was aligned with *DS-Hacker 3D* content and intended learning objectives (ILOs) and based on the books Data Structure and Algorithms by Aho (1988) and Algorithms by Sedgewick and Wayne (2014), which are two of the primary textbooks in relation to data structures. The content validity was verified by two undergraduate data structure and algorithms professors from Trinity College Dublin (TCD) and six programming and data structure and algorithms professors from UCR. The consistency of the test was assessed with psychometric techniques within Item Response Theory and the Rasch model with a sample of 220 computer science bachelor students from University of Costa Rica. A full description of the test can be found in Chapter 6.

To evaluate participants' intrinsic motivation, we used a Spanish translation of the Intrinsic Motivation Inventory (IMI) survey (R. M. Ryan, 1982). The IMI is a well-known survey used to assess six factors: enjoyment, perceived competence, effort, usefulness, felt pressure and tension, and perceived choice, where enjoyment is considered the self-reported measure of intrinsic motivation (McAuley et al., 1989). The other factors have a close correlation with the enjoyment factor result. Studies (e.g., McAuley et al., 1989; Monteiro et al., 2015) have proven the validity and consistency of the IMI. For the evaluation, the IMI survey was used to assess participants' intrinsic motivation after completing the experimental and control activities.

Demographic data concerning gender, gaming experience, career, academic year, and programming skills was collected. Data collected was used to identify factors that may affect the test and IMI scores results.

Concerning the tools used in the control activity, two video tutorials[1] developed by the School of Mathematics at the Open State University of Costa Rica (UNED) were used. These tutorials covered all the ILOs taught by *DS-Hacker 3D* and were available in the official YouTube channel of the school. One of the videos was about fundamental concepts of the binary tree data structure, and it had a duration of six minutes and twenty-six seconds. The second video was about fundamental concepts of the BST data structure, and it had a duration of twelve minutes and twenty-nine seconds.

Due to the quarantine caused by the global pandemic, it was necessary to build a web application to perform the evaluation. The web application was developed using React and Firebase technologies. Furthermore, it was designed to guide the participants during the experiment, present the surveys, and store the collected data. Additionally, the web application provided the materials for the learning activities.

7.1.1.3 Research Design

An independent repeated measures experimental design was programmed. Students were assigned randomly to one of two learning activities. The experiment was guided by the web application, and all surveys and tests were completed and submitted through the application. First, a consent form and an agreement checkbox were presented to the students. Then, students who agreed to participate were randomly assigned by the web application to one of the two learning activities. Next, the web application presented the general instructions of the experiment. Then, the demographic survey and its instructions were presented to the students. Once all the questions of the survey were answered, participants could proceed to the next section. Then, the pre-test and its instructions were presented. Once all the questions were marked, the students could continue to the next section. Next, the instructions and materials for the learning activities were provided by the web application. Participants were asked to complete the activities before they could continue to the next section of the experiment. Then,

[1] The videos tutorials are available in https://youtu.be/2K8CmQs1jl8 and https://youtu.be/Bh61AvHAf90.

the post-test followed by the IMI scale were provided to the students. Once all the questions were marked, participants could submit all the surveys and tests. Data were stored in a Firebase database. The consent form, experiment instructions, and demographic survey used during the experiment are provided in Appendix 5.1.

7.1.1.4 Data Analysis Methods

Data analysis was performed using R language version 4.01 (*See Things Now*), R Studio, and Microsoft Excel. Initial data cleaning was implemented using Microsoft Excel. All inconsistent data, such as repeated rows, were removed using the Excel's filters. Plotting, descriptive, and inferential statistics analysis were implemented in R Studio.

During the initial exploration of data collected, boxplots and descriptive statistics summaries were used to visualize and analyse data patterns. Data visualization was implemented using R Package *graphics* and *ggplot2*. The descriptive analysis was implemented using the R Package *psych*.

We verified that data were normally distributed and homoscedastic. Concerning the normal distribution assumption, it was verified using the skew and kurtosis value, histograms, Normal Probability-Probability (PP) plots, and Shapiro-Wilk tests. The skew and kurtosis value were obtained using the R Package *psych*. The Shapiro-Wilk test was performed using the R Package *nortest*, and the histograms and Normal PP Plots were drawn using the R Package *ggplot2* and *qqplotr* . Homoscedasticity was verified using the Levene's tests available in the R Package *car*.

Data collected that meet the parametric conditions were analysed using T-tests. Data that did not meet the assumptions were analysed using the Mann-Whitney U test. The R Package *rstatix* was used to perform the inferential statistics, including the Cohen's d, and the Wilcoxon effect size.

7.1.2 Results

7.1.2.1 Descriptive Analysis Results

Results of the analysis of the pre-test show that the median of the scores obtained by the control and experimental group are the same (*Median* = 5). The value of the control group's mean (*M* = 4.29) and the experimental group's mean (*M* = 4.85) are close to each other. Similarly, the value of the control group's standard deviation (*SD* = 2.12) and the experimental standard deviation (*SD* = 1.99) are close to each other. However, the interquartile range (*IQR*) values are different. The control group's *IQR* value (*IQR* = 4) is larger than the experimental group's *IQR* value

(*IQR* = 2). Consequently, the experimental group has a more compact distribution than the control group. Figure 7.1.A shows a boxplot of the pre-test results.

Results of the analysis of the post-test show that the value of the control group median and mean values (*Median* = 8, *M* = 7.8) is slightly smaller than the experimental group's median and mean values (*Median* = 9, *M* = 8.89), respectively. The control group's standard deviation (*SD* = 1.96) is larger than the experimental group's standard deviation (*SD* = 1.013). Consequently, the experimental group has a more compact distribution than the control group. Figure 7.1.B shows the boxplot of the post-test results.

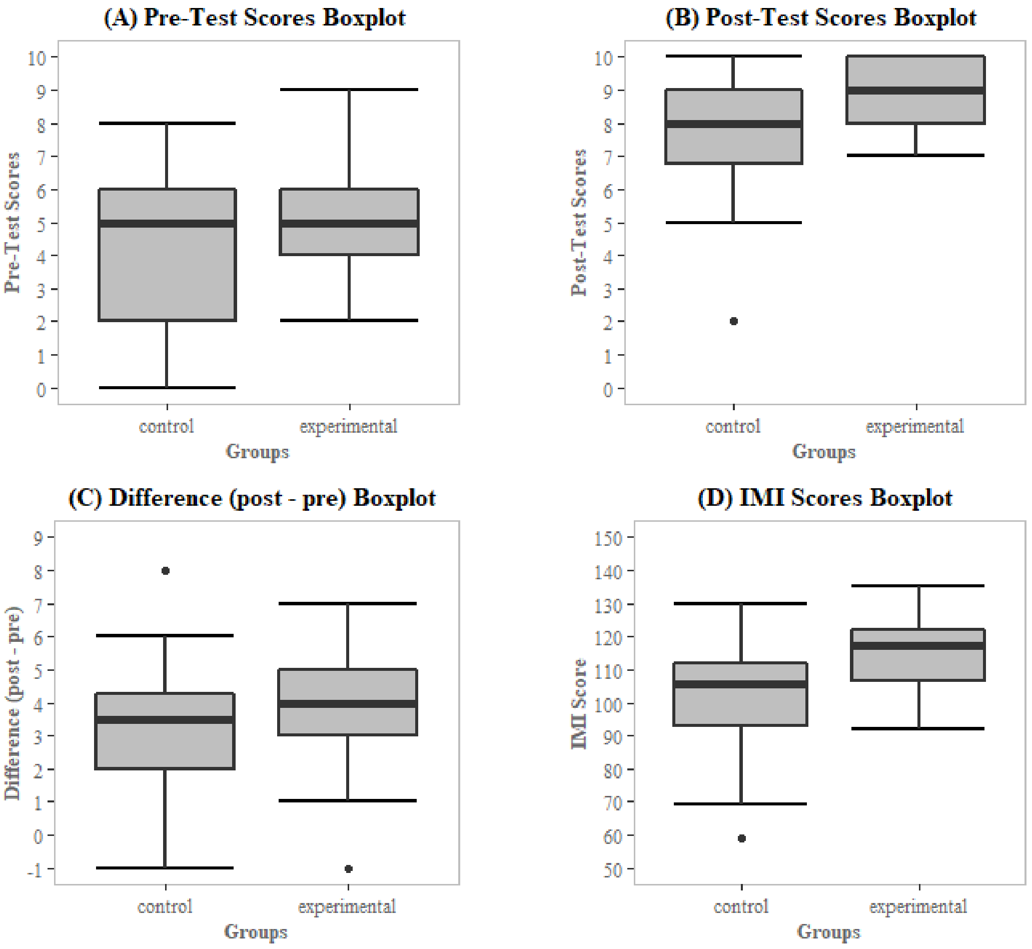


Figure 7.1: Boxplots of the (A) pre-test, (B) post-test, (C) difference between the pre- and post-test, and (D) IMI scores.

Results of the analysis of differences between the post-test scores and the pre-test scores (post-test scores minus pre-test scores) show that the control group's median and mean values (*Median* = 3.5, *M* = 3.54) are slightly smaller than the experimental group's median and mean values (*Median* = 4, *M* = 4.04). The value of the control group's standard deviation (*SD* = 2.43) is larger than the experimental group's standard deviation (*SD* = 1.77). Figure 7.1.C shows the boxplot of the differences between the post-test scores and the pre-test scores.

Results of the IMI scores' analysis show that the values of the control group's median and mean (*Median* = 105.5, *M* = 101.21) are smaller than the experimental group's median and mean (*Median* = 117, *M* = 115.59). The value of the standard deviation of the control group (*SD* = 16.63) is larger than the standard deviation of the experimental (*SD* = 11.86). Figure 7.1.D shows the boxplot of the IMI scores.

Table 7.1 and 7.2 present a comprehensive summary of the descriptive analysis of the pre-test, post-test, and IMI scores, and the differences between the post-test and pre-test scores. The tables include the minimum and maximum values, interquartile values, interquartile range, median, mean, variance and standard deviation.

Table 7.1: Descriptive statistics of the pre-test and post-test scores of the control and experimental groups.

	Pre-Test Control	Post-Test Control	Pre-Test Experimental	Post-Test Experimental
Number of values	28	28	27	27
Min	0	2	2	7
1st Quartile	2	6.75	4	8
Median	5	8	5	9
Mean	4.27	7.82	4.85	8.89
3rd Quartile	6	9	6	10
Max	8	10	9	10
Interquartile Range (IQR)	4	2.25	2	2
Variance	4.51	3.86	3.98	1.03
Standard Deviation	2.12	1.96	1.99	1.01

Table 7.2: Descriptive statistics of the differences between the post-test and pre-test and the IMI scores of the control and experimental group.

	Difference Control	Difference Experimental	IMI Score Control	IMI Score Experimental
Number of values	28	27	28	27
Min	−1	−1	59	92
1st Quartile	2	3	92.75	106.5
Median	3.5	4	105.5	117
Mean	3.54	4.04	101.21	115.59
3rd Quartile	4.25	5	112	122
Max	8	7	130	135
Interquartile Range	2.25	2	19.25	15.5
Variance	5.89	3.11	276.55	140.71
Standard Deviation	2.43	1.77	16.63	11.86

7.1.2.2 Verification of the Parametric Assumptions

To apply parametric methods, it is necessary to verify four assumptions.

1. Measurements must be interval data.
2. Data must be normally distributed.
3. Homogeneity of variance is required. The variance should be the same throughout the data.

4. Independence of data. Data obtained from one participant should not affect data obtained from other participants (Field, 2013).

Data obtained in this evaluation fulfil the first and fourth assumption. Measures were obtained from tests and surveys that have proven their psychometric and scale properties. Due to the characteristics of the experiment, participants' behaviour did not affect the results of other participants. Consequently, only the second and third assumptions need to be verified analytically.

7.1.2.2.1 Normally Distributed Data

Concerning the pre-test scores of the control group, the skewness value ($S = -0.093$) is close to zero suggesting that data are symmetrically distributed. The kurtosis value ($K = -0.98$) is close to negative one indicating that data distribution is flat and light-tailed. The Shapiro-Wilk test results ($W = 0.95$, *p-value* = 0.24) show that the null hypothesis cannot be rejected (*p-value* > 0.05). In other words, we cannot reject that the data is normally distributed. The histogram (Figure 7.2 A) shows that most of the observations are normally distributed excluding values between one and two of the x-axis. The Normal PP Plot (Figure 7.2 B) shows that many observations fall outside of the "ideal" diagonal. Based on the results, we conclude that the pre-test scores are normally distributed.

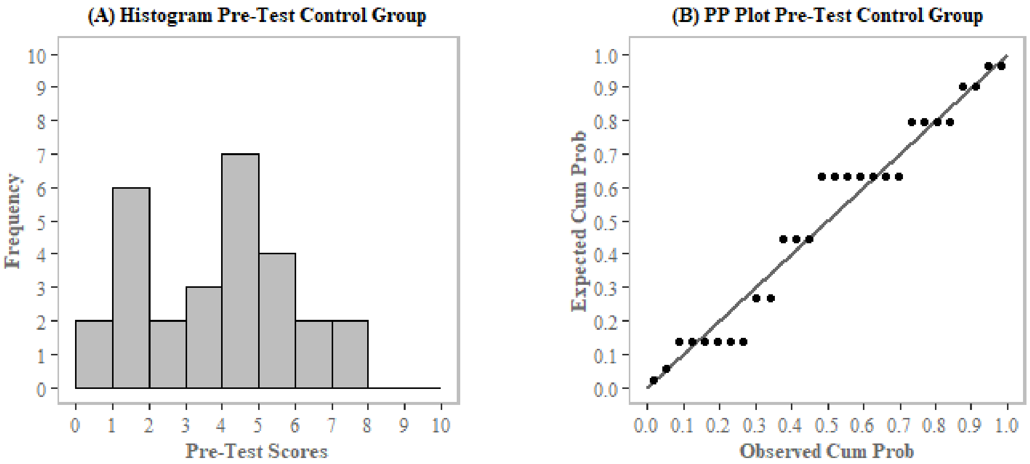


Figure 7.2: (A) Histogram of the pre-test scores obtained by the control group; (B) Normal Probability-Probability Plot of the pre-test score obtained by the control group.

Regarding the pre-test scores of the experimental group, the skewness value ($S = 0.42$) indicates that the data distribution is slightly positive skewed; the scores pile up to the left side of the distribution slightly. The kurtosis value ($K = -0.58$) indicates that the data distribution is flat and light-tailed. The Shapiro-Wilk test results ($W = 0.94$, *p-value* = 0.11) show that the null hypothesis cannot be rejected (*p-value* > 0.05). The histogram (Figure 7.3 A) shows that the data distribution resembles the normal distribution. The Normal PP Plot (Figure 7.3 B) shows that many observations fall outside of the "ideal" diagonal. Based on the results, we conclude that pre-test scores of the experimental group are normally distributed.

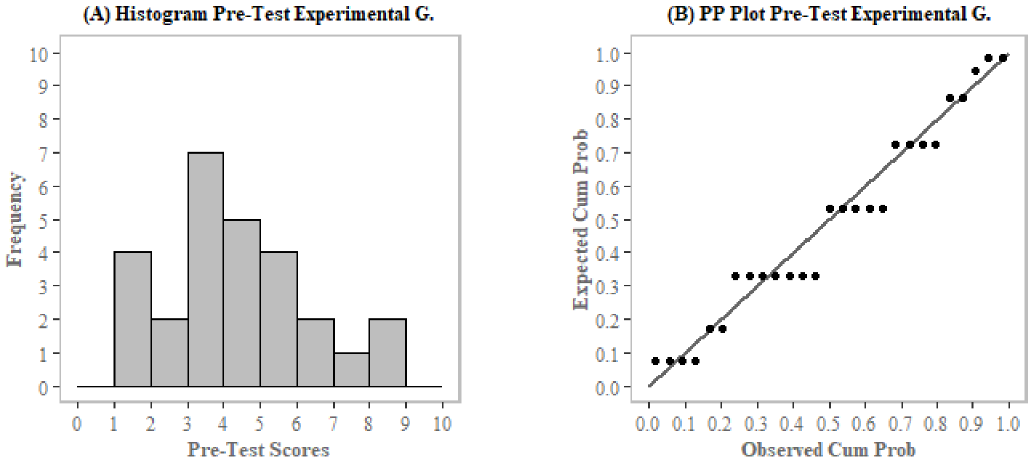


Figure 7.3: (A) Histogram of the pre-test scores obtained by the experimental group; (B) Normal Probability-Probability Plot of the pre-test score obtained by the experimental group.

In relation to the post-test scores of the control group, the skewness value (S = −0.92) suggests that the data distribution is negative skewed; the scores are piled up to right side of the distribution. The kurtosis value (K = 0.53) indicates that data distribution is pointy and heavy-tailed. The Shapiro-Wilk test results (W = 0.89, *p-value* = 0.007) show that the null hypothesis is rejected (*p-value* < 0.05). We cannot accept that the data are normally distributed. The histogram (Figure 7.4 A) shows that the data is completely skewed to the right side of the chart. The Normal PP Plot (Figure 7.4 B) shows that many observations fall outside of the "ideal" diagonal. According to the results, we conclude that the post-test scores of the control group are not normally distributed.

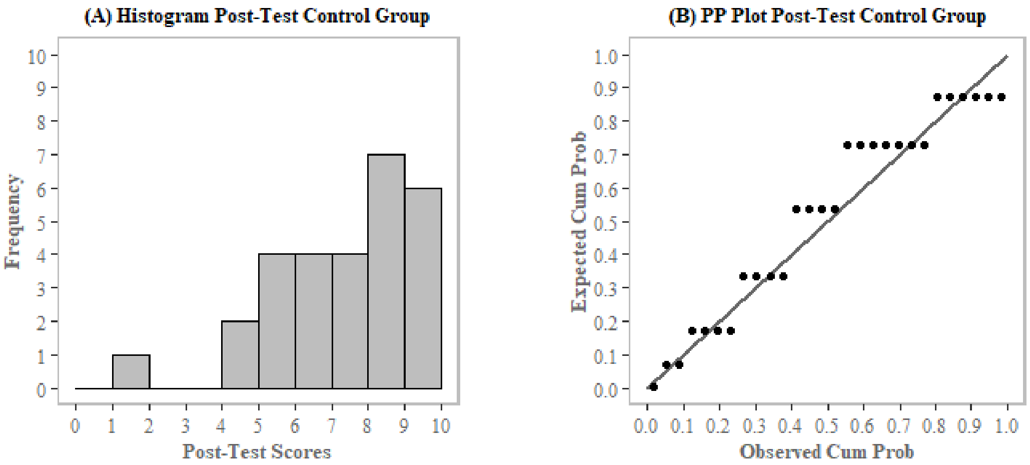


Figure 7.4: (A) Histogram of the post-test scores obtained by the control group; (B) Normal Probability-Probability Plot of the post-test score obtained by the control group.

Concerning the post-test scores of the experimental group, the skewness value (S = −0.43) indicates that the data distribution is negative skewed, suggesting that scores are slightly piled up to the right side of the distribution. The kurtosis value (K = −1.04) indicates that the data distribution is flat and light-tailed. The Shapiro-Wilk test results (W = 0. 86, *p-value* = 0.001) show that the null hypothesis is rejected (*p-value* < 0.05). We cannot accept that the data is normally distributed. The histogram (Figure 7.5 A) shows that the data are completed skewed to the right side of the plot. The Normal PP Plot (Figure 7.5 B) shows that many observations fall outside the "ideal" diagonal, and observations are distributed across four values of the y-axis. Based on the results, we conclude that the data are not normally distributed.

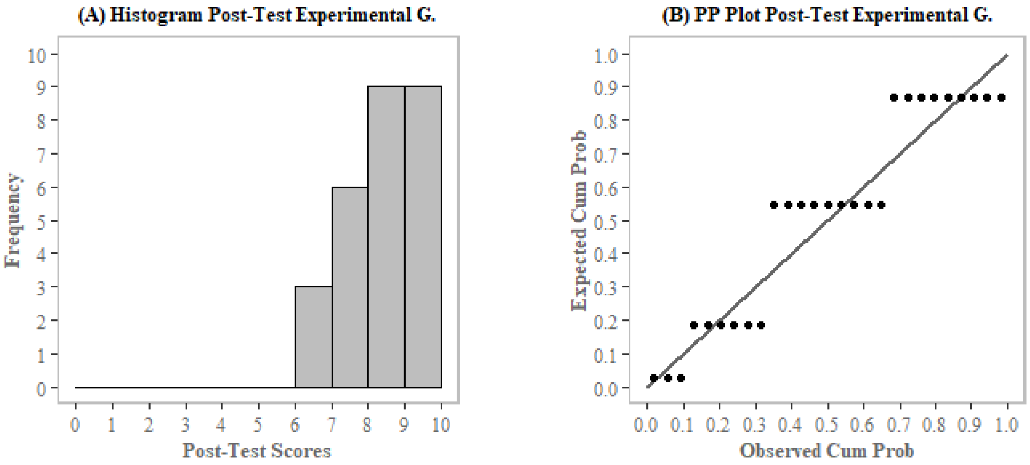


Figure 7.5: (A) Histogram of the post-test scores obtained by the experimental group; (B) Normal Probability-Probability Plot of the post-test score obtained by the experimental group.

Regarding the difference of scores between the post-test and pre-test of the control group, the skewness value (S = 0.42) indicates that the data distribution is moderately positive skewed, suggesting that the scores are slightly pile up to the left side of the distribution. The kurtosis value (K = −0.56) indicates that the data distribution is flat and light-tailed. The Shapiro-Wilk test results (W = 0. 93, *p-value* = 0.065) show that the null hypothesis cannot be rejected (*p-value* > 0.05). The histogram (Figure 7.6. A) shows that the data are skewed to the left. The Normal PP Plot (Figure 7.6. B) shows that many observations fall outside the "ideal" diagonal. Based on the results, the data is assumed to be normally distributed. However, we are aware that these results do not provide conclusive evidence that the data are normally distributed, and that the *p-value* of the Shapiro-Wilk test is very close to the rejection range.

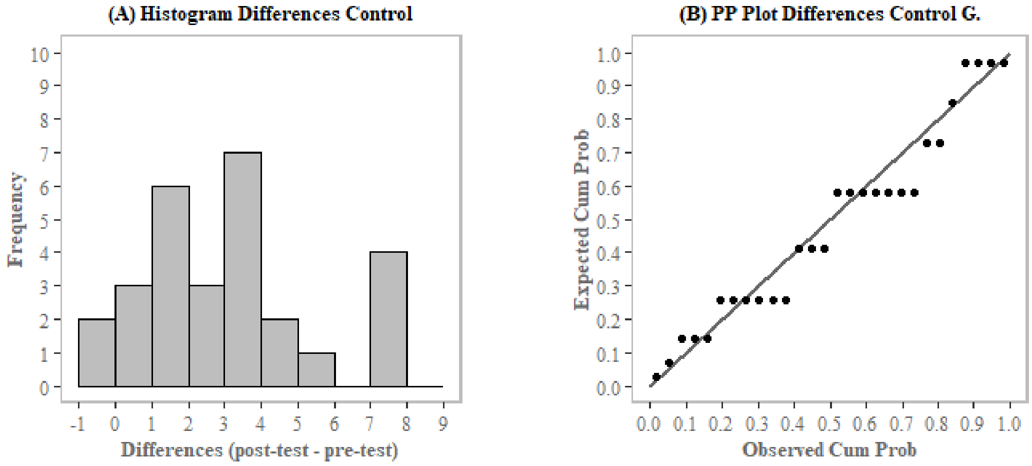


Figure 7.6: (A) Histogram of the difference of scores of the post-test and pre-test obtained by the control group; (B) Normal Probability-Probability Plot of the difference of scores obtained by the control group.

In relation to the difference of scores between the post-test and pre-test of the experimental group, the skewness value (S = −0.78) indicates that the data distribution is positive skewed, suggesting that the scores are piled up to the right side of the distribution. The kurtosis value (K = 0.49) indicates that the data distribution is pointy and heavy-tailed. The Shapiro-Wilk test results (W = 0.93, *p-value* = 0.064) show that the null hypothesis cannot be rejected (*p-value* > 0.05). The histogram (Figure 7.7 A) shows that the data distribution resembles the normal distribution, and it is moderately skewed to the right side. The Normal PP Plot (Figure 7.7 B) shows that many observations fall outside the "ideal" diagonal. Based on the results, the data are assumed to be normally distributed. However, we are aware that these results are not conclusive, and that the *p-value* of the Shapiro-Wilk test is close to the rejection range.

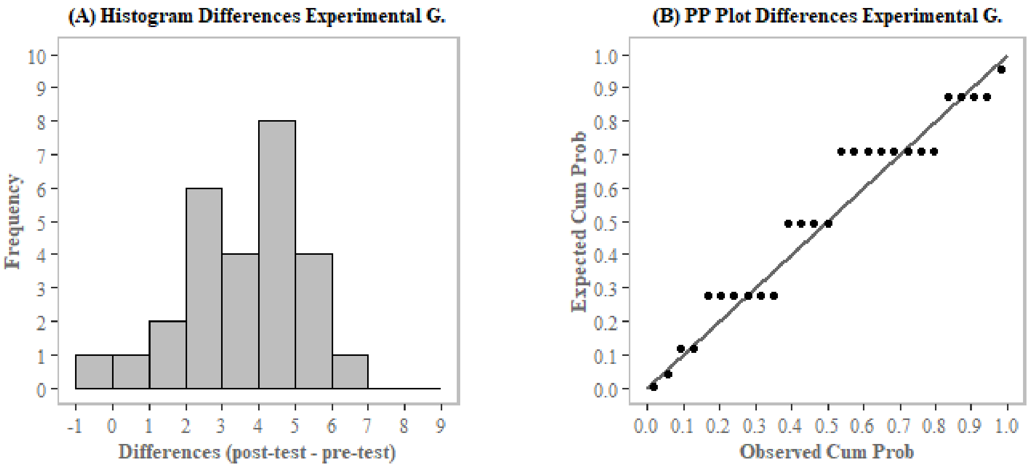


Figure 7.7: (A) Histogram of the difference of scores of the post-test and pre-test obtained by the experimental group; (B) Normal Probability-Probability Plot of the difference of scores obtained by the experimental group.

Concerning the IMI scores of the control group, the skewness value ($S = -0.7$) indicates that the data distribution is negative skewed; the scores are piled up to the right side the distribution. The kurtosis value ($K = -0.06$) indicates that the data distribution shape is close to the normal distribution. The Shapiro-Wilk test results ($W = 0.96$, *p-value* = 0.27) show that the null hypothesis cannot be rejected (*p-value* > 0.05). The histogram (Figure 7.8 A) shows that the data are skewed to the right side. The Normal PP Plot (Figure 7.8 B) shows that there are some observations that are not aligned with the "ideal" line, but they are close to the line. Based on the results, the data are assumed to be normally distributed.

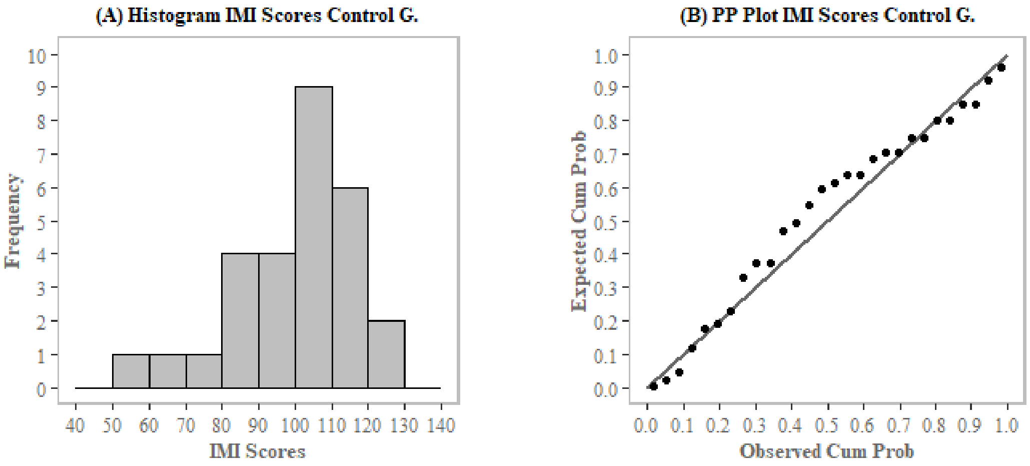


Figure 7.8: (A) Histogram of the IMI scores obtained by the control group; (B) Normal Probability-Probability Plot of the IMI scores obtained by the control group.

Regarding the IMI scores of the experimental group, the skewness value ($S = -0.\ 11$) indicates that the data are symmetrically distributed. The kurtosis value ($K = -0.89$) indicates that the data are flat and light-tailed. The Shapiro-Wilk test results ($W = 0.97$, *p-value* = 0.63) show that the null hypothesis cannot be rejected (*p-value* > 0.05). The histogram (Figure 7.9 A) shows that the data seem normally distributed. The Normal PP Plot (Figure 7.9 B) shows that there are some observations that are not aligned with the “ideal” line, but they are close to the line. Consequently, the data are assumed to be normally distributed.

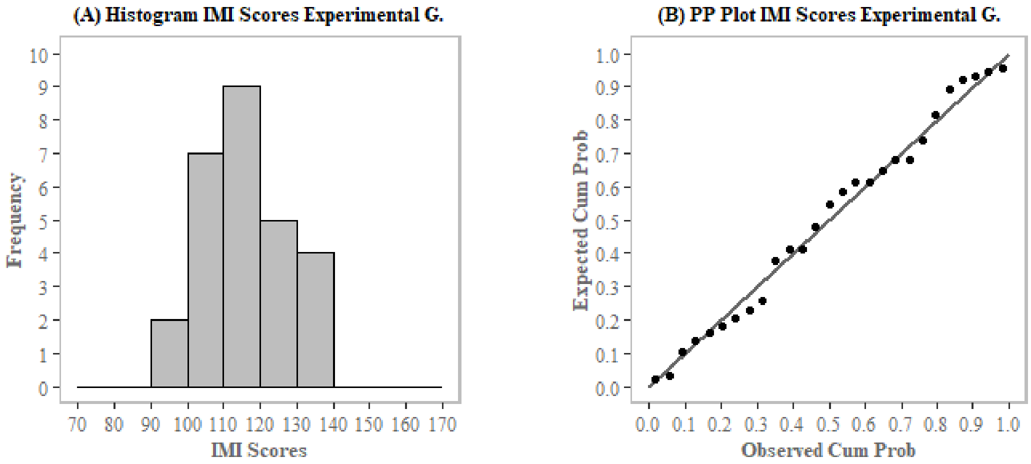


Figure 7.9: (A) Histogram of the IMI scores obtained by the experimental group; (B) Normal Probability-Probability Plot of the IMI scores obtained by the experimental group.

7.1.2.2.2 Homogeneity of Variance

For the scores obtained in the pre-test, the variances were equal for the control and experimental groups, *F*(1, 53) = 0.2 and the *p-value* = 0.66. For the scores obtained in the post-test, the variances were significantly different in both groups, *F*(1, 53) = 8.43, *p-value* = 0.01. For the difference between the post-test and pre-test scores, the variances were equal for the control and experimental group, *F*(1, 53) = 2.24, *p-value* = 0.14. Finally, for the IMI scores, the variances were equal for both groups, *F*(1, 53) = 1.38, *p-value* =0.25. Table 7.5 summarizes the results of the Levene’s tests.

Table 7.3: Levene’s test results

	Levene Statistic	df1	df2	p-value
Pre-test scores	0.2	1	53	0.66
Post-test scores	8.43	1	53	0.01
Differences	2.24	1	53	0.14
IMI scores	1.38	1	53	0.25

7.1.2.3 Inferential Analysis Results

Results of the t-test performed on the pre-test scores suggest that the difference between the control and experimental group was not significant, *t*(53) = −1.02, *p-value* = 0.31 (*p-value* > 0.05). Results of a Mann-Whitney U test suggest that the differences between the control and experimental group was statistically significant, *W* = 259.5, *p-value* = 0.04. Results of the t-test performed on the differences of scores between the post-test and pre-test scores show that the difference between the control and experimental group was not significant, *t*(53) = −0.87, *p-value* = 0.39. Results of the t-test performed on the IMI scores indicates that the difference between the control and experimental group was statistically significant, *t*(53) = −3.68, *p-value* = 0.001. Table 7.6 summarizes the results of the inferential analysis.

Table 7.4: Results of the inferential analysis between groups.

	Statistic	df	p-value	95% confidence interval	Effect size
Pre-test scores	−1.02	53	0.31	[−1.68, 0.55]	−0.28
Post-test scores	259.5		0.04		0.28
Differences	−0.87	53	0.39	[−1.65, 0.65]	−0.24
IMI scores	−3.68	53	0.001	[−22.22, −6.54]	−1

7.1.2.4 Results Analysis

Results of the analysis of the pre-test scores suggest that the control and experimental group had the same level of knowledge regarding BSTs before doing the learning activity. The mean and standard deviation values of the experimental group (*M* = 4.85, *SD* = 1.99) were close the mean and standard deviation values of the control group (*M* = 4.29, *SD* = 2.12), respectively. The median values of both groups were the same (*Median* = 5). The *p-value* of the t-test shows that there is no difference between the experimental and control groups, indicating that variations happened by chance. However, results suggest that there is a difference between the interquartile range of the experimental group ([4, 6]) and the control group ([2, 6]). These values indicate that a larger portion of the participants of the control group had lower scores than the participants of the experimental group. Even then, these results confirm that on average, both groups started the evaluation with a similar level of knowledge regarding BSTs. Additionally, the results confirm that most of the participants had basic knowledge of computer science, and that the knowledge test has a medium level of difficulty.

Results of the analysis of the post-test scores show that both groups, the experimental and control, learned by using their respective learning tools, i.e., the game and the video tutorials. However, the results suggest that the experimental group obtained significantly better results. The median of the control group increased from 5 to 8 points, and the median of the experimental increased group from 5 to 9, one point higher than the control group. Similarly, the mean of the control (*M* = 7.82) and experimental (*M* = 8.89) group increased. The results show less variation of the scores obtained by the experimental group than the scores obtained by the control group, suggesting that the game has more consistent learning outcomes. The standard deviation of the experimental group (*SD* = 1.01) is smaller than the standard deviation of the control group (*SD* = 1.96). Finally, results of a Mann-Whitney U test indicate that the difference between the control and experimental group is statistically significant (*p-value* =

0.04), suggesting that the experimental group performed better than the control group. However, the *p-value* is close to the acceptance range. Consequently, these results should not be taken as conclusive. To summarize, results indicate that both learning tools, the game and video tutorials, are efficient means to teach BST factual and conceptual knowledge.

As shown above, results of the post-test analysis show that the experimental group performed better than the control group. However, the quartile range values of the pre-test analysis show that there was a slight difference between the groups in the form of a larger number of lower scores in the control group. This difference might affect the results obtained in the post-test analysis. To verify this supposition, the differences between the post-test and the pre-test of the experimental and control groups were analysed. These differences tell us how much participants learned from the activities. It is expected that participants who achieved a high score in the pre-test obtained lower learning gains because they already possessed the BST knowledge taught by the learning tools. In contrast, participants who obtained a lower score in the pre-test are expected to obtain higher learning gains. Therefore, due to the slightly lower performance on the pre-test scores of the control group, higher learning gains are expected from this group.

Results of a t-test of the difference ($t(53) = -0.87$, *p-value* = 0.39) show that there are no statistically significant differences of learning gains between the experimental group and the control group. However, the mean and median of the experimental group (*M* = 4.04, *Median* = 4) are slightly higher than the mean and median of the control group (*M* = 3.54, *Median* = 3.5). The standard deviation of the experimental group (*SD* = 2.43) is lower than the standard deviation of the control group (*SD* = 1.77).

Results of the analysis of the IMI scores show that participants who played the game were more intrinsically motivated than the participants who watched the video tutorials. Results of the descriptive analysis show that on average, there is a difference of 14 points between the experimental (M = 115.59) and control (M = 101.21) group. The interquartile range of the experimental group ([106.5, 122]) is higher and more compact than the interquartile range of the control group ([92.75, 112]). Furthermore, results of the t-test show that this difference is statistically significant (*p-value* = 0.001).

7.1.3 Discussion

In this section, we answer SRQ 4 by answering its three sub-questions. First, we address the questions concerning knowledge (SRQ 4.1 and 4.2). Then, we discuss the question about intrinsic motivation (SRQ 4.3). At the end of this section, we summarize the points given and answer SRQ 4. The discussion and answers are based on the evaluation’s results and theory.

Concerning SRQ 4.1, our results proved that *DS-Hacker 3D* is an effective learning tool to teach conceptual knowledge about BSTs. The game showed that it can provide an environment where students can actively learn and build their knowledge. It was demonstrated that the analogies embedded in the game mechanics transmitted the BST concepts accurately. Also, it was demonstrated that the analogical game model provided an environment where the students could test and practice the concepts transmitted by the game mechanics. This was evidenced by the fact that the students who played the game increased their scores in the post-test significantly, and the scores among all students was consistently high.

Regarding SRQ 4.2, our results showed that *DS-Hacker 3D* is slightly more effective at teaching conceptual knowledge than the two video tutorials. Like the students who played the game, students who watched the video tutorials increased their score in the post-test. However, on average, students who played the game obtained higher learning gains, and their gains were more consistent than the control group's learning gains. These findings suggest that well-designed video games that uses analogies embedded in their game model and game mechanics have the potential to be at least as effective as "good" video tutorials to teach conceptual knowledge.

In Chapter 1, we discussed about the difficulties of learning computer science knowledge. These difficulties may be caused by students' previous knowledge, communication flaws, task complexity, and inappropriate learning strategies (Kempa, 1991). Our findings suggest that *DS-Hacker 3D* could mitigate these learning difficulties. Video games allow the construction of analogical models that can make use of students' understanding of video games to teach the abstract topics efficiently. Also, video game levels allow to split the content and learning task into manageable sizes that fit students' capabilities. We conclude that video games that use analogies, like *DS-Hacker 3D,* can teach conceptual knowledge effectively and can be used in formal education environments, such as higher education, to teach introductory scientific conceptual knowledge, increasing the amount of effective learning tools.

The positive results of the knowledge evaluation can be attributed to the design strategy applied. *DS-Hacker 3D* was designed to ensure that the learning activities were aligned with the ILOs. According to Biggs (2007), a good pedagogical design for higher education environments must ensure that less proactive and motivated students should achieve the same learning objectives as the engaged and motivated students. To guarantee this aim, educators must choose or design learning activities that require active engagement and proactivity of all students, including those students who are less proficient, engaged, or motivated. These activities must be aligned with the learning objectives and also be in concordance with a learning theory. Furthermore, Mayes and de Freitas (2004) extend this idea and suggest that learning tools must apply this principle during their design stage. In other words, learning tools must have clear learning objectives and learning activities that support the achievement of those objectives. In this regard, *DS-Hacker 3D* was designed taking in consideration the alignment of learning objectives and learning activities. The learning objectives were based on (Aho, 1988) and (Sedgewick & Wayne, 2014), two well-known books for teaching data structures in higher education environments. The learning activities were designed together with the game elements and embedded in the game elements, such story, challenges, environment, and user interface (UI). For example, the player must listen to and read analogies, facts and terminology related to the BST data structure. This information is embedded in the game story. Also, the game requires that the player traverse the game environment that is structured as a BST. To achieve this, the player must understand the maps provided by the UI. These maps are visual representations of BSTs. Additionally, to solve the puzzles (challenges), the player must apply fundamental BST concepts, giving the opportunity to recall and apply the concepts learned. These game features ensure that all players achieve the learning objectives while they are playing. This is supported by the knowledge evaluation. Results of the analysis of the pre- and post-test scores demonstrate that participants who played *DS-Hacker 3D* achieved the learning objectives through the game learning activities, suggesting that the game was efficient and that the design approach was effective.

Finally, our findings differ from other studies that found that gamified activities, including video games, are better suited for procedural knowledge, and non-gamified activities for factual and procedural knowledge (Perini et al., 2018). Rather, our results suggest that digital game-based learning has the same potential to teach factual and conceptual knowledge as non-gamified activities. As mentioned, the scope of the learning objectives and type of knowledge are decided during the design phase. Similarly, the genre and game elements that are going to be used to achieve the learning objectives and communicate the type of knowledge are chosen and designed during the design phase. Consequently, the creativity of the game designers determines the efficacy of the game as a learning tool.

Regarding SRQ 4.3, our results showed that participants who played *DS-Hacker 3D* were more intrinsically motivated than participants who watched the video tutorials. Intrinsic motivation is a desirable feature for learning experience because it enhances the students' will to keep engaged learning a topic (Touré-Tillery & Fishbach, 2014). *DS-Hacker 3D* was demonstrated to be superior in this aspect compared to the video tutorials. Results showed that almost all the students felt enjoyment while playing, and at the same time, they felt that what they were doing was useful and relaxing. This finding is particularly positive, because it differs from studies that suggested that a considerable proportion of adults in higher education are not motivated by video games (Whitton, 2014). We conclude that well-designed video games are intrinsically motivating learning tool that can be used in formal education.

A reason for these results is that *DS-Hacker 3D* was designed by taking into consideration game aspects that are considered intrinsically motivating. First, studies related to video games, motivation, and engagement suggest that narrative elements, such as the plot and character story, boost the player's immersion, curiosity, and engagement (Calleja, 2007; Malone, 1980; Whitton, 2014; Yee, 2006). In this regard, *DS-Hacker 3D*'s story is based on the Hero's Journey narrative archetype, a classic narrative structure that can be used to create engaging stories (Campbell, 1991). Additionally, the game story is loaded with fantasy elements that increase the players immersion and maintain the players' intrinsic motivation at a high level.

Besides the narrative and fantasy elements, *DS-Hacker* includes challenges and goals that boost the players' motivation. According to several studies (Boyle et al., 2012; Calleja, 2007; Malone, 1980; Whitton, 2014), achievement of challenges and task-related goals, and progression are key techniques to increase players' motivation. Additionally, the challenges and tasks should be well matched to the players' skills or knowledge (Csikszentmihalyi, 1990). In this regard, *DS-Hacker* provides several levels, and each level has tasks and cognitive puzzles related to the BST concepts. The difficulty increases every level; the first level is the easiest, and the fourth level is the most difficult. Furthermore, at the beginning of every level, the game provides the BST concepts that players need to solve the level's tasks and challenges, ensuring that all players have the means to accomplish the tasks and puzzles given. Therefore, *DS-Hacker* is designed to provide players with challenges and tasks that are well matched for the players' skill and that allow the progression through the game levels.

7.1.4 Limitations

Despite our efforts to be as thorough as possible during the evaluation and analysis, this study has some limitations. First, we are aware that the sample size is small for an evaluation related to human psychological features. Variation in human psychological constructs such as learning and intrinsic motivation is large. For this reason, it is necessary to obtain larger samples to increase the reliability of the measurements and the power of the statistical methods (Parsons et al., 2019). Additionally, our evaluation was performed over a population with very similar characteristics, i.e., students from the same year in the same school. Therefore, a replication of this evaluation with a population with different characteristics may provide different results. Another limitation is that we assumed that the Spanish translation of the IMI survey has the same psychometric properties as the English version. Based on this assumption, we performed an inferential statistical analysis using parametric methods. However, best practice requires the verification of the psychometric properties of the Spanish version of the survey. Finally, due to the characteristics of our methodology, our findings do no explain *how* people were learning; our observations only confirm *that* people were learning. To fill this gap, it is necessary to perform a qualitative study using appropriate methodologies for those purposes.

7.2 Evaluation 2: Representational Effects of Games for Learning

The second experiment evaluates the effect of realism on learning gains and motivation. This evaluation aims to answer SRQ 5: *Does the realism level of video games for learning that use analogies affects the learning gains and motivation?* To achieve this goal, two version of *DS-Hacker* with different levels of realism are evaluated. Furthermore, SRQ 5 is divided into three specific questions that consider the target population and traditional digital learning experience evaluated in the experiment. The questions are:

SRQ 5.1. Do university students learn conceptual knowledge about BST using *DS-Hacker 2D* and *DS-Hacker 3D*?
SRQ 5.2. Are there learning differences between learning with *DS-Hacker 2D* and *DS-Hacker 3D*?
SRQ 5.3. Are there intrinsic motivation differences between learning with *DS-Hacker 2D* and *DS-Hacker 3D*?

In Chapter 5, we explained that *DS-Hacker* has two versions that differ in their level of realism. Specifically, the game aspects that differ are the graphics and the physics. One version has realistic 3D graphics and physics. Due to these aspects, this version has more complex game mechanics and a more complex environment (game world and levels). The second version of the game has 2D graphics, and the physics are abstracted. Consequently, this version has simpler game mechanics and environment. Besides these differences, both versions of the game have the same ILOs, learning activities, game story, and game challenges.

It is hypothesised that results from the evaluation will confirm that *DS-Hacker 2D* and *DS-Hacker 3D* are effective learning tools. Students will increase their learning gains about BST conceptual knowledge after playing the games. Also, it is hypothesised that there will be learning differences between both games. Students who play *DS-Hacker 2D* will obtain higher scores in a conceptual knowledge test than *DS-Hacker 3D*. This hypothesis is based on Mayer's Theory of Multimedia Learning that suggest that learning is more effective when interesting but extraneous material is excluded from the learning experience (Mayer & Moreno, 2003). Regarding intrinsic motivation, it is hypothesised that there will be motivation differences between both games. Students who play *DS-Hacker 3D* will obtain higher scores in an intrinsic motivation test than students who played the 2D version. We expect that 3D graphics and a richer environment will increase the fantasy and players' exploration engagement (Malone, 1980).

To confirm or reject our hypotheses, an evaluation was performed during October and November of 2020 at National University of Colombia (UNAL). Due to the Covid-19 pandemic, UNAL was placed on lockdown, and classes were taught online. For this reason, the experiment was performed virtually, and all materials and tests were distributed and completed using a custom-made web application. The evaluation was conducted using the third version of *DS-Hacker 3D* and the first version of *DS-Hacker 2D* (see section 5.2).

7.2.1 Materials and Methods

7.2.1.1 Participants

Participants were students enrolled in the Object-Oriented Programming course at the National University of Colombia. This population matched the characteristics of the game's target population, students who possess basic programming knowledge and skills. In total, 53 students completed all the surveys and tests required for our evaluation. Participants were randomly assigned to the 2D or 3D group. The 2D group had 25 participants (47.2%), and the 3D group had 28 participants (52.8%). Regarding their demographic background, 9 participants (17%) were female, and 44 participants (83%) were male.

7.2.1.2 Materials and Data Collection Tools

For this evaluation, the materials and data collections used in Evaluation 1 were utilized again (see Appendix 5.1 and 5.2). To measure BST knowledge, we used the custom-made BST knowledge test discussed in section 6.4 (also in Appendix 4.3). To measure participants' intrinsic motivation after using the 2D and 3D versions of the game, we used the Intrinsic Motivation Inventory (IMI; see Appendix 5.1.6) survey (R. M. Ryan, 1982). Demographic data was collected using the custom-made demographic survey discussed in section 7.1.1.2. Concerning the materials used for learning activities, we used the 2D and 3D versions of *DS-Hacker*, which differ only in the level of abstraction of their graphics and physics. Both games have the same story (learning content) and challenges (learning activities). Finally, due to the quarantine caused by the pandemic, classes and laboratories were run online. For this reason, it was necessary to build a website to perform the experiments. The website used to execute this evaluation was implemented based on the website of the first evaluation covered in section 7.1 and used same the web technologies.

7.2.1.3 ***Research Design***

This evaluation followed the same research design as the first evaluation. The experiment was structured as an independent measures experimental design where students were assigned to one of the two available learning activities: 2D game and 3D game. The experiment was driven by the web application, and all surveys and tests were completed through the web application. The experiment was structured as follows: The consent form was presented to the students. Then, students who agreed to participate were randomly assigned to one of the two learning activities. The random assignation was performed automatically by the web application. Then, the demographic survey and pre-test were provided to the participants. Then, the instructions and materials for the learning activities were provided to the students. Finally, the post-test and IMI survey were provided to the students. To submit the survey and tests, all questions had to be answered. The completed surveys and tests were stored in an external database in Firestore.

7.2.1.4 ***Data Analysis Methods***

Data analysis was performed using R language version 4.01 (*See Things Now*), R Studio, and Microsoft Excel. Initial data cleaning was implemented using Microsoft Excel. All inconsistent data, such as repeated rows, were removed using the Excel's filters. Plotting, descriptive, and inferential statistics analysis were implemented in R Studio.

During the initial exploration of data collected, boxplots and descriptive statistics summaries were used to visualize and analyse data patterns. Data visualization was implemented using R Package *graphics* and *ggplot2*. The descriptive analysis was implemented using the R Package *psych*.

We verified that data were normally distributed and homoscedastic. Concerning the normal distribution assumption, it was verified using the skew and kurtosis value, histograms, Normal Probability-Probability (PP) plots, and Shapiro-Wilk tests. The skew and kurtosis value were obtained using the R Package *psych*. The Shapiro-Wilk test was performed using the R Package *nortest*, and the histograms and Normal PP Plots were drawn using the R Package *ggplot2* and *qqplotr* . Homoscedasticity was verified using the Levene's tests available in the R Package *car*.

Data that did not meet the parametric assumptions were analysed using the Mann-Whitney U test and the Wilcox effect size. The R Package *rstatix* was used for this purpose.

7.2.2 Results

7.2.2.1 Descriptive Analysis Results

Results of the analysis of the pre-test show that the median of the scores obtained by the 2D and 3D group are the same (*Median* = 5). The value of the 2D group's mean (*M* = 4.4) and the 3D group's mean (*M* = 4.93) are close to each other. The value of 2D group's standard deviation (*SD* = 2.33) and the 3D group's standard deviation (*SD* = 1.84) are close to each other. The 2D group's *IQR* value (*IQR* = 4) is larger than the 3D group's *IQR* value (*IQR* = 3). However, as presented in Figure 7.10.A, the interquartile range (*IQR*) values of both groups are close to each other.

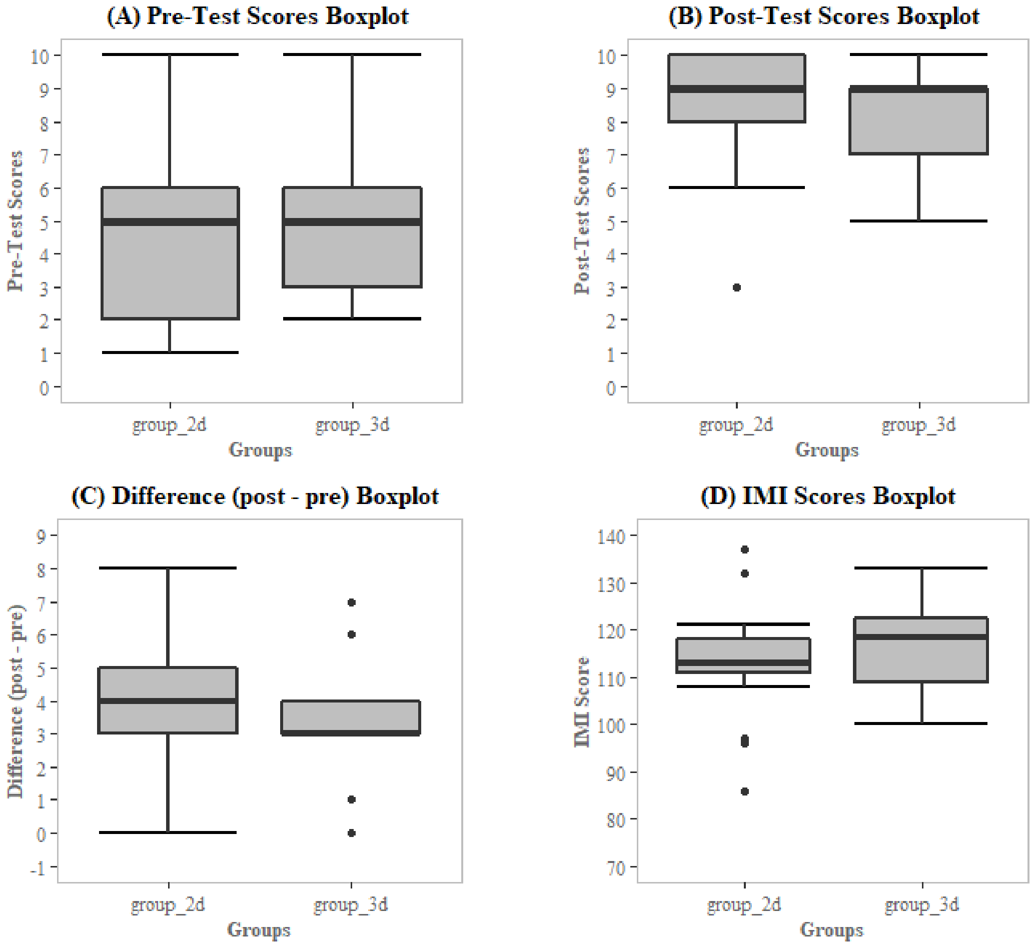


Figure 7.10: Boxplots of the (A) pre-test, (B) post-test, (C) difference between the pre- and post-test, and (D) IMI scores of the second evaluation.

Results of the post-test show that the median values of the 2D and 3D group are the same (*Median* = 9). The 2D group's mean (*M* = 8.56) is slightly larger than the 3D group's mean (*M* = 8.32). The 2D group's standard deviation (*SD* = 1.76) is slightly larger than the 3D group's standard deviation (*SD* = 1.47). The first and third quartile of the 2D group (Q_1 = 8, Q_3 = 10) are larger than the first and third quartiles of the 3D group (Q_1 = 7, Q_3 = 9). Figure 7.10.B shows the boxplot of the post-test results.

Results of the analysis of differences between the post-test scores and the pre-test scores (post-test scores minus pre-test scores) show that the 2D group's median and mean values (*Median* = 4, *M* = 4.16) are slightly larger than the 3D group's median and mean values (*Median* = 3, *M* = 3.39). The value of the 2D group's standard deviation (*SD* = 1.86) is slightly smaller than the 3D group's standard deviation (*SD* = 1.95). Figure 7.10.C shows the boxplot of the differences between the post-test scores and the pre-test scores.

Results of the IMI scores' analysis show that the median and mean of the 2D group (*Median* = 113, *M* = 113.40) is smaller than the median and the mean of the 3D group (*Median* = 118.5, *M* = 116.57). The standard deviation of the 2D group (*SD* = 10.3) is larger than the standard deviation of the 3D group (*SD* = 8.39). However, the 2D group's *IQR* value (*IQR* = 7) is smaller than the 3D group's *IQR* value (*IQR* = 13.5). Figure 7.1.D shows the boxplot of the IMI scores.

Table 7.5: Descriptive statistics of the pre-test and post-test scores of the control and experimental groups.

	Pre-Test 2D Group	Post-Test 2D Group	Pre-Test 3D Group	Post-Test 3D Group
Number of values	25	25	28	28
Min	1	3	2	5
1st Quartile	2	8	3	7
Median	5	9	5	9
Mean	4.4	8.56	4.93	8.32
3rd Quartile	6	10	6	9
Max	10	10	10	10
Interquartile Range (IQR)	4	2	3	2
Variance	5.42	3.09	3.40	2.15
Standard Deviation	2.33	1.76	1.84	1.47

Table 7.6: Descriptive statistics of the differences between the post-test and pre-test and the IMI scores of the control and experimental group.

	Difference 2D Group	Difference 3D Group	IMI Score 2D Group	IMI Score 3D Group
Number of values	25	28	25	28
Min	0	0	86	100
1st Quartile	3	3	111	109
Median	4	3	113	118.5
Mean	4.16	3.39	113.4	116.57
3rd Quartile	5	4	118	122.5
Max	8	7	137	133
Interquartile Range	2	1	7	13.5
Variance	3.47	3.80	106.09	70.33
Standard Deviation	1.86	1.95	10.3	8.39

Table 7.5 and 7.6 present a comprehensive summary of the descriptive analysis of the pre-test, post-test, and IMI scores, and the differences between the post-test and pre-test scores. The

tables include the minimum and max values, interquartile values, interquartile range, median, mean, variance and standard deviation.

7.2.2.2 Verification of the Parametric Assumptions

To apply parametric methods, it is necessary to verity four assumptions (see Section 7.1.2.2). Data obtained in this evaluation fulfil the first and fourth assumption. Measures were obtained from tests and surveys that have proven their psychometric and scale properties. Due to the characteristics of the experiment, participants' behaviour did not affect the results of other participants. Consequently, only the second and third assumptions must be verified.

7.2.2.2.1 Normally Distributed Data

Regarding the pre-test scores of the 2D group, the skewness value (S = 0.41) suggests that the data distribution is slightly positive skewed, suggesting that the scores are piled up to the left side of the distribution. The kurtosis value (K = −0.62) suggests that the data distribution is slightly flat and light-tailed. The Shapiro-Wilk test results (W = 0.95, *p-value* = 0.22) show that the null hypothesis cannot be rejected (*p-value* > 0.05). In other words, we cannot reject that the data is normally distributed. The histogram (Figure 7.11.A) supports the skewness value and shows that many observations are piled in the left side of the distribution. The Normal PP Plot (Figure 7.11 B) shows that many observations fall outside of the "ideal" diagonal. Based on the Shapiro-Wilk test, we conclude that the pre-test scores are normally distributed.

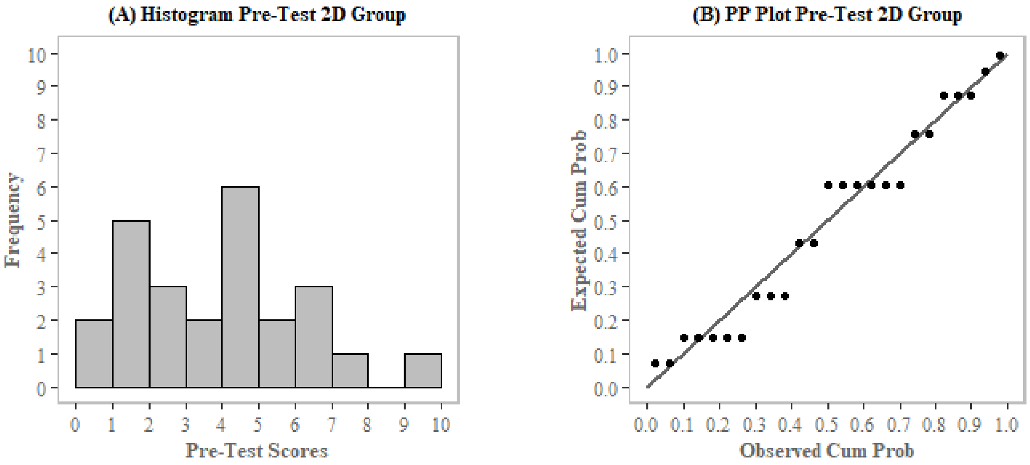


Figure 7.11: (A) Histogram of the pre-test scores obtained by the 2D group; (B) Normal Probability-Probability Plot of the pre-test score obtained by the 2D group.

In relation to the pre-test scores of the 3D group, the skewness value (S = 0.37) indicates that the data distribution is slightly positive skewed; the scores are slightly piled up to the left side of the distribution. However, the value is close to zero. The kurtosis value (K = 0) indicates that the data distribution is not flat or pointy. The Shapiro-Wilk test results (W = 0.91, *p-value* = 0.03) show that the null hypothesis is rejected (*p-value* < 0.05). We cannot accept that data are normally distributed. The histogram (Figure 7.12.A) shows that the data distribution is not normally distribution. The Normal PP Plot (Figure 7.12 B) shows that many observations fall outside of the "ideal" diagonal. Based on the results, we conclude that pre-test scores of the 3D group are not normally distributed.

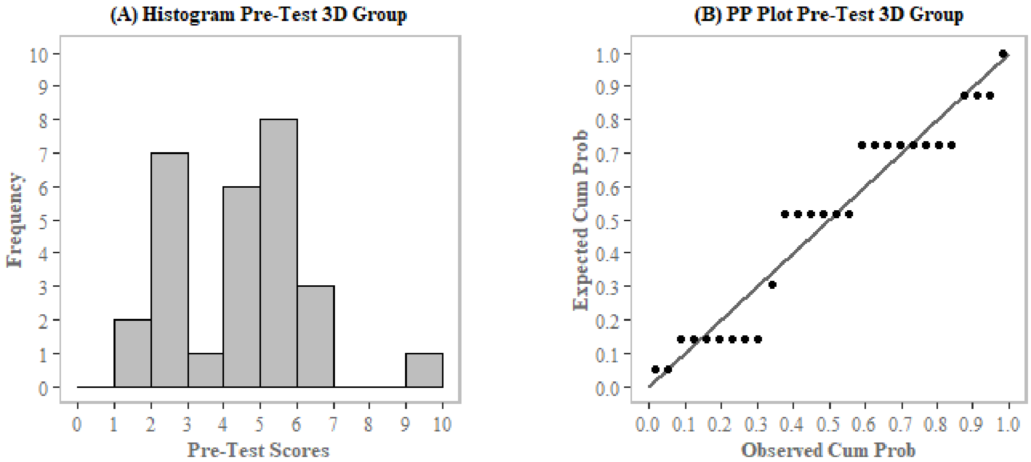


Figure 7.12: (A) Histogram of the pre-test scores obtained by the 3D group; (B) Normal Probability-Probability Plot of the pre-test score obtained by the 3D group.

Concerning the post-test scores of the 2D group, the skewness value (S = −1.42) suggests that the data distribution is negative skewed, indicating that the scores are piled up to the right side (see Figure 7.13 A). The kurtosis value (K = 1.72) indicates that the data distribution is pointy and heavy-tailed. The Shapiro-Wilk test results (W = 0.79, *p-value* = 0.0001) show that the null hypothesis is rejected (*p-value* < 0.05). We cannot accept that the data are normally distributed. The Normal PP Plot (Figure 7.13.B) shows that many observations fall outside of the "ideal" diagonal. According to the results, we conclude that the post-test scores of the 2D group are not normally distributed.

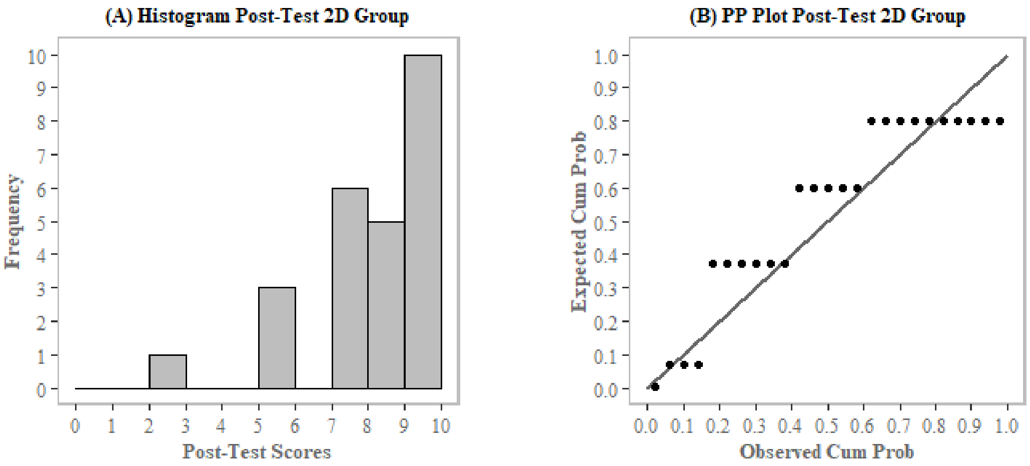


Figure 7.13: (A) Histogram of the post-test scores obtained by the 2D group; (B) Normal Probability-Probability Plot of the post-test score obtained by the 2D group.

Regarding the post-test scores of the 3D group, the skewness value (S = −0.75) suggests that the data distribution is negative skewed; the scores are piled up to the right side of the distribution (see Figure 7.14 A). The kurtosis value (K = −0.77) indicates that data distribution is flat and light-tailed. The Shapiro-Wilk test results (W = 0.84, *p-value* = 0.0005) show that the null hypothesis is rejected (*p-value* < 0.05). We cannot accept that the data is normally distributed. The Normal PP Plot (Figure 7.14 B) shows that many observations fall outside the “ideal” diagonal. Based on the results, we conclude that the post-test scores are not normally distributed.

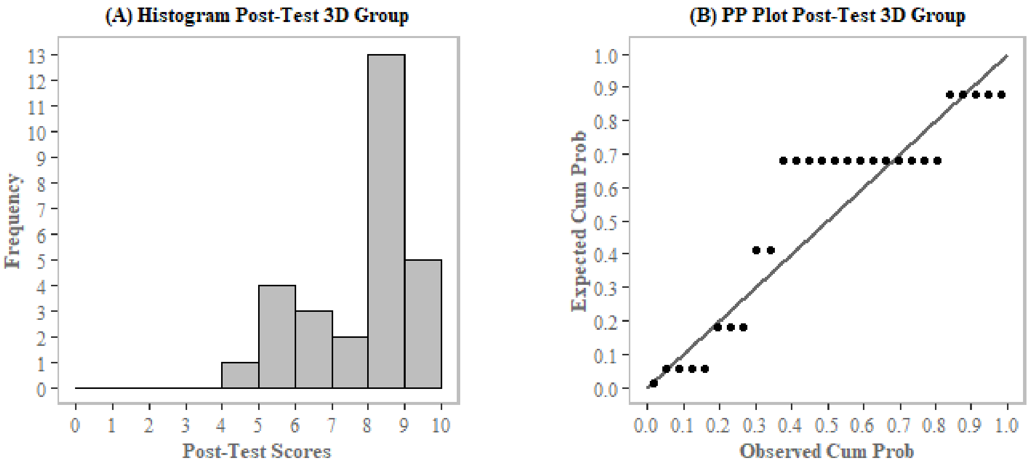


Figure 7.14: (A) Histogram of the post-test scores obtained by the 3D group; (B) Normal Probability-Probability Plot of the post-test score obtained by the 3D group.

In relation to the difference of scores between the post-test and pre-test of the 2D group, the skewness value (S = −0.08) suggests that the data are normally distributed (see Figure 7.15 A). The kurtosis value (K = −0.27) indicates that the data distribution is slightly flat and light-tailed. The Shapiro-Wilk test results (W = 0. 97, *p-value* = 0.68) show that the null hypothesis cannot be rejected (*p-value* > 0.05). The Normal PP Plot (Figure 7.15 B) shows that many observations fall outside the "ideal" diagonal. Based on the results, the data is assumed to be normally distributed.

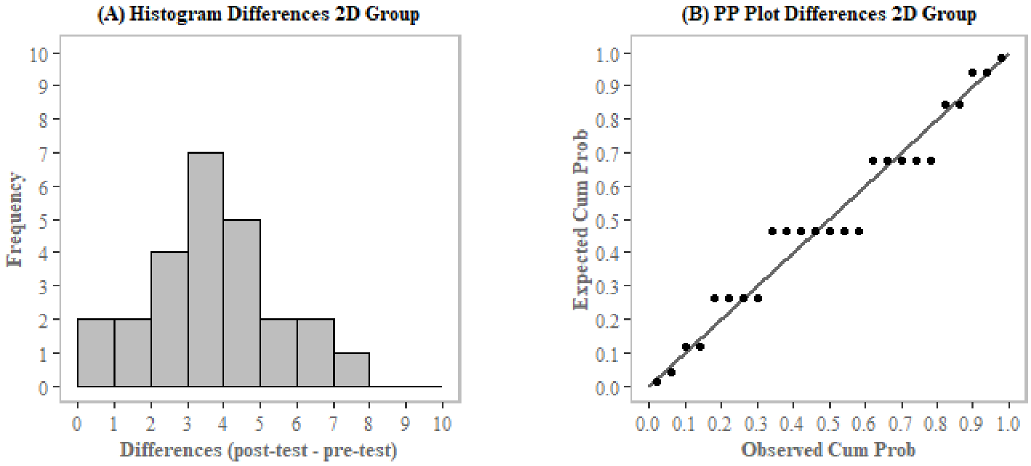


Figure 7.15: (A) Histogram of the difference of scores of the post-test and pre-test obtained by the 2D group; (B) Normal Probability-Probability Plot of the difference of scores obtained by the 2D group.

Concerning the difference of scores between the post-test and pre-test of the experimental group, the skewness value (S = 0.16) indicates that the data distribution is slightly positive skewed; the scores are piled up to the left side. The kurtosis value (K = −0.73) indicates that the data distribution is flat and light-tailed. The Shapiro-Wilk test results (W = 0.91, *p-value* = 0.02) show that the null hypothesis is rejected (*p-value* < 0.05). We cannot accept that scores are normally distributed. The histogram (Figure 7.16.A) shows that the data distribution is moderately skewed on the left side. The Normal PP Plot (Figure 7.16.B) shows that many observations fall outside the "ideal" diagonal. Based on the results, we conclude that the scores are not normally distributed.

Regarding the IMI scores of the 2D group, the skewness value (S = −0.34) suggests that the data distribution is slightly negative skewed, indicating that the scores are slightly piled up to the right side of the distribution (see Figure 7.17 A). The kurtosis value (K = 1.16) indicates that the data distribution is pointy and heavy-tailed. The Shapiro-Wilk test results (W = 0.91, *p-value* = 0.04) show that the null hypothesis is rejected (*p-value* < 0.05). We cannot accept that the scores are normally distributed; however, the p-value is very close to the threshold limit. The Normal PP Plot (Figure 7.17 B) shows that most of the observations are not aligned with the "ideal" line. Based on the results and the histogram, we conclude that the data are not normally distributed.

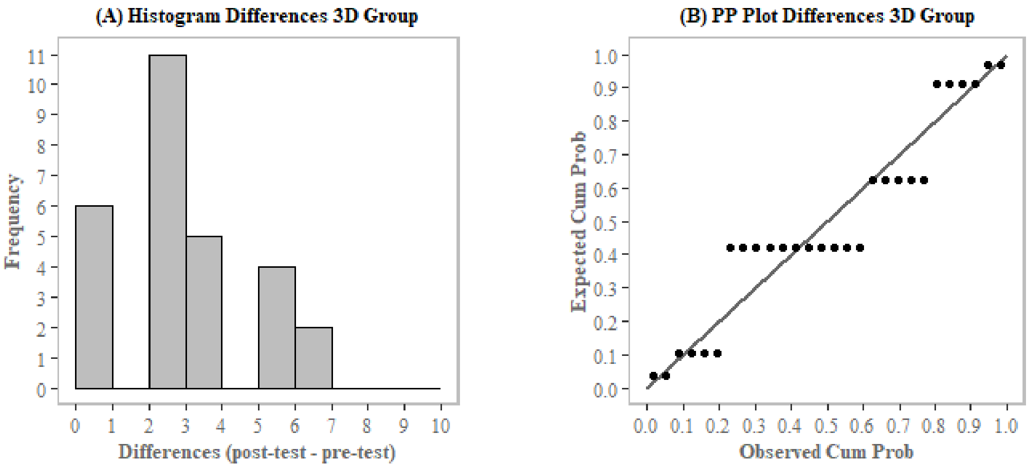


Figure 7.16: (A) Histogram of the difference of scores of the post-test and pre-test obtained by the 3D group; (B) Normal Probability-Probability Plot of the difference of scores obtained by the 3D group.

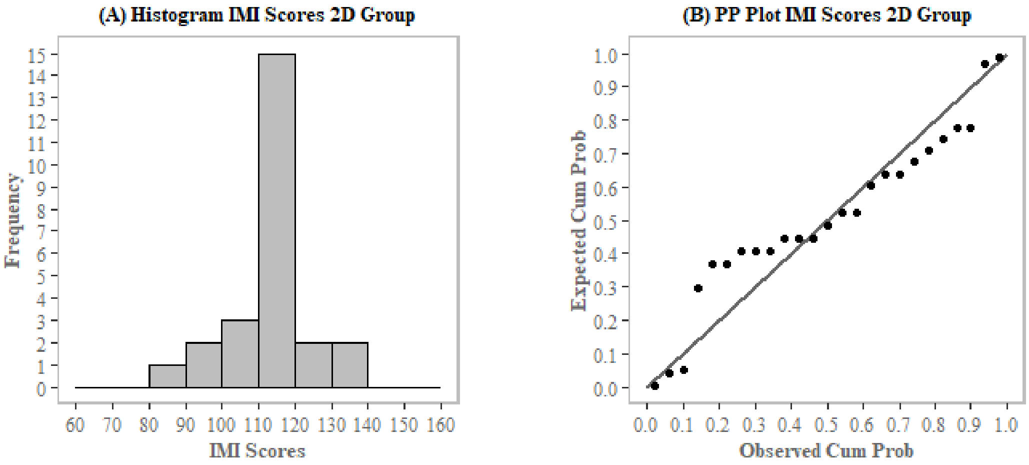


Figure 7.17: (A) Histogram of the IMI scores obtained by the 2D group; (B) Normal Probability-Probability Plot of the IMI scores obtained by the 2D group.

Concerning the IMI scores of the 3D group, the skewness value (S = −0.13) suggests that the data distribution is slightly negative skewed. The kurtosis value (K = −1.02) indicates that the data are flat and light-tailed. The Shapiro-Wilk test results (W = 0.97, *p-value* = 0.68) show that the null hypothesis cannot be rejected (*p-value* > 0.05). The histogram (Figure 7.18 A) shows that data seem normally distributed. The Normal PP Plot (Figure 7.18 B) shows that there are some observations that are not aligned with the "ideal" line, but they are close to the line. Consequently, we conclude that data is normally distributed.

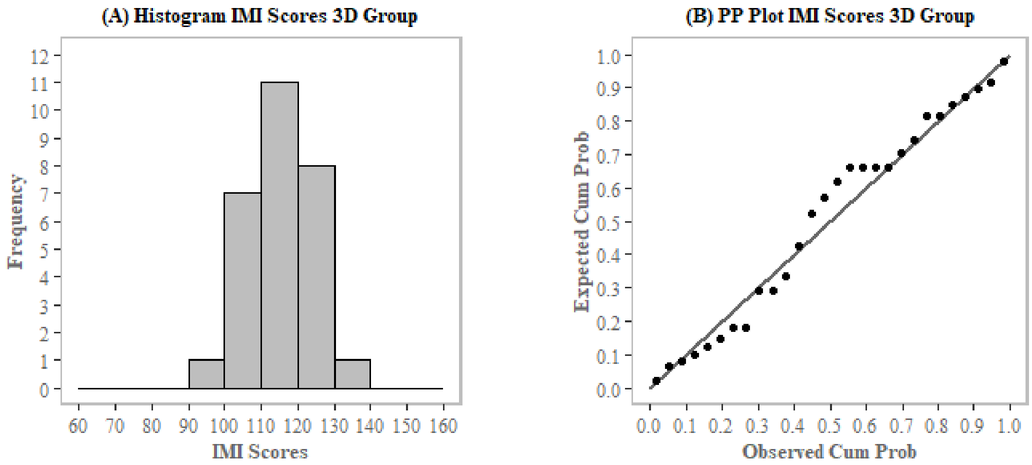


Figure 7.18: (A) Histogram of the IMI scores obtained by the 3D group; (B) Normal Probability-Probability Plot of the IMI scores obtained by the 3D group.

7.2.2.2.2 Homogeneity of Variance

For the pre-test, post-test, differences between post- and pre-test, and IMI scores, the variances were equal for the 2D and 3D group. Table 7.7 summarizes the results of the Levene's tests.

Table 7.7: Levene's test results

	Levene Statistic	df1	df2	p-value
Pre-test scores	1.61	1	51	0.21
Post-test scores	0.35	1	51	0.56
Differences	0.01	1	51	0.93
IMI scores	0.01	1	51	0.94

*7.2.2.3 **Inferential Analysis Results***

Results of the data's distribution analysis show that most of the scores are not normally distributed. For this reason, the Mann-Whitney U test, a nonparametric method, was utilized to perform the inferential analysis.

Regarding the within groups analysis, results of the paired Mann-Whitney U test performed on the 2D group scores found that the difference between the pre-test and post-test scores is statistically significant, *W* = 300, *p-value* = 0.00 (*p-value* < 0.05). The Wilcox effect size is 0.72, suggesting a large effect. Results of the paired Mann-Whitney U test performed on the 3D group found that the difference between the pre-test and post-test scores is statistically significant, *W* = 351, *p-value* = 0.00 (*p-value* < 0.05). The Wilcox effect size is 0.71, suggesting a large effect. Table 7.8 summarize the results of the inferential analysis within groups.

Table 7.8: Results of the within groups inferential analysis and effect size.

	Size	Statistic	p-value	Effect size
2D Group	25	300	0.00002	0.72
3D Group	28	351	0.00001	0.71

Concerning the between groups analysis, results of the Mann-Whitney U test performed on the pre-test scores suggest that the difference between the 2D and 3D groups was not significant, *W* = 290, *p-value* = 0.28. Results of the Mann-Whitney U test performed on the post-test scores suggest that the difference between the groups was not significant, *W* = 400, *p-value* = 0.39. Results of the Mann-Whitney U test performed on the differences between the pre- and post-test (post-test minus pre-test) suggest that the difference between groups was not significant, *W* = 440, *p-value* = 0.11. Results of the Mann-Whitney U test performed on the IMI scores suggest that the difference between the groups was not significant, *W* = 281, *p-value* = 0.22. Table 7.9 summarizes the results of the inferential analysis.

Table 7.9: Results of the between groups inferential analysis and the effect size.

	2D size	3D size	Statistic	p-value	Effect size
Pre-test	25	28	290	0.28	0.15
Post-test	25	28	400	0.36	0.13
Differences	25	28	440	0.11	0.22
IMI	25	28	281	0.22	0.17

7.2.2.4 Results Analysis

Results of the analysis of the pre-test scores suggest that the 2D and 3D group had the same level of knowledge regarding BSTs before playing the games. The mean and standard deviation values of the 2D group (*M* = 4.40, *SD* = 2.33) were close the mean and standard deviation values of the 3D group (*M* = 4.93, *SD* = 1.84), respectively, and the median values of both groups were the same (*Median* = 5). The first and third quartile of the 2D group (Q_1 = 2, Q_3 = 6) and 3D group (Q_1 = 3, Q_3 = 6) and the boxplot of the pre-test scores (Figure 7.10.A) suggest that the scores' ranges were similar in both groups. Results of the Mann-Whitney U test shows that the difference between the groups is not statistically significant, suggesting that it was caused due to random variation. Based on this evidence, we conclude that the 2D and 3D groups came from the same population. Additionally, the results show that most of the participants had basic knowledge of computer science.

Results of the post-test scores analysis suggest that participants learned with both versions of the game. The median of both groups increased from 5 to 9. The mean of the 2D group increased from 4.40 to 8.56, and the mean of the 3D group increased from 4.93 to 8.32. Results of the paired Mann-Whitney test show that the difference between the pre-test and post-test is statistically significant for both groups. However, the boxplot (Figure 7.10.B) shows that the 2D group obtained more consistent and tighter scores. For example, 50% of the participants of the 2D group obtained 9 or more correct answers. Meanwhile, 75% of the participants of the 3D group obtained 9 or less but more than 5 correct answers. Results of the Mann-Whitney U test indicates that the difference between the post-tests of the 2D and 3D group is not significant. This evidence suggests that both versions are effective learning tools.

Results of the Mann-Whitney U test performed on the differences between the pre-test and post-test scores (*W* = 440, *p-value* = 0.11) show that the difference between groups is not statistically significant. However, the mean and median of the 2D group (*M* = 4.16, *Median* = 4) are slightly higher than the mean and median of the 3D group (*M* = 3.39, *Median* = 3). Also, the boxplot (Figure 7.10.C) shows that the 50% of the participants of the 2D group increased more than 4 points in the post-test. On the other hand, excluding four participants, all the participants of the 3D group increased 3 or 4 points in the post-test. These results suggest that the 2D version of the game is slightly more effective than the 3D version in teaching BST conceptual knowledge.

Results of the analysis of the IMI scores show that participants who played the 3D version of the game were slightly more intrinsically motivated than the participants who played the 2D version. The mean and median of the 3D group (*M* = 116.57, *Median* = 118.5) were slightly higher than the mean and median of the 2D group (*M* = 113.4, *Median* = 113). The boxplot (Figure 7.10.D) shows that a larger number of participants perceived that the 3D version was more intrinsically motivating than the 2D version. However, results from the inference analysis (*W* = 281, *p-value* = 0.22) shows that this difference is not statistically significant. Results of the statistical analysis suggest that both versions are intrinsically motivating.

7.2.3 Discussion

In this section, we answer SRQ 5: *Does the realism level of video games for learning that use analogies affects the learning gains and motivation?* To achieve this, we divided SRQ 5 into three sub-questions: SRQ 5.1, SRQ 5.2, and SRQ 5.3.

Concerning SRQ 5.1, results of the descriptive and inferential analysis show that both versions of the game are effective learning tools. Post-test scores increased significantly after playing the games. These findings suggest that regardless of the level of realism, the game model and game environment of both versions of *DS-Hacker* are effective means to represent BTs and BSTs. Concerning the pedagogical methods, analogies embedded in the narrative elements of the game (e.g., dialogues and user interface) were demonstrated to be an effective approach to facilitate the learning process of conceptual knowledge in both versions.

Regarding the teaching efficacy comparison (SRQ 5.2), the 2D version is slightly more efficient in facilitating conceptual knowledge acquisition than the 3D version. Results of the inferential analysis show that score differences between learning gains of the 3D and 2D group are not statistically significant. Similarly, results of the inferential analysis of the post-test results of the 3D and 2D group are not statistically significant. However, results of the descriptive analysis (e.g., boxplots, means, and quartile results) show slight differences, favouring the 2D version. These findings suggest that a simpler graphical representation and simpler game physics may increase learning gains. According to the cognitive load theory, learning tools and instructional methods should present the information in a way that maximizes the cognitive resources available. To achieve this, instructors and instructional designers should minimize external sources of cognitive load. In this regard, video games may possess elements that increase external cognitive load (e.g., background music, goals, challenges, narrative elements, etc).

As mentioned in section 5.2.3 (see Table 5.3), *DS-Hacker* 3D and 2D share the same contents and learning aspects. Similarly, most of the game elements are the same in both versions. The sole game elements that were different were the graphical representations, which also impacted the game physics, game mechanics and controllers' inputs. All these game elements were simplified in the 2D version. Specifically, the 2D version has less and simpler navigation mechanics that the 3D version. Also, the 2D version has less control inputs than the 3D version. These features of the 2D version make it easier to play than the 3D version, allowing players to focus more on the game narrative and challenges, which convey the learning contents and allow practice. Consequently, the 2D version allows players to focus more on the learning contents than the 3D version.

We conclude that the level of realism does not significantly affect the learning efficacy of video games that use analogies. In both version of the game, the game model and game environment represent the same aspects of the BST data structure. Even if the graphical representation of each version is different, players understand the game model and the analogies between the game model and environment and the BST data structure. However, analogies may increase a game's efficacy when external causes of cognitive load are reduced (e.g., abstracting game mechanics and graphical representations).

Furthermore, our results agree with other studies that evaluated the effect of 2D and 3D graphical representations in learning (Ak & Kutlu, 2017; Kaimara et al., 2020; Schrader & Bastiaens, 2012; Zaharias et al., 2017). These studies suggest that 2D and 3D graphical representations are effective for learning purposes; however, 2D versions are slightly better than 3D.

Concerning the IMI scores (SRQ 5.3), results of the inferential analysis indicate that there are no statistically significant differences between the two versions. However, results of the descriptive analysis show that the 3D version produces higher and lower IMI scores than the 2D version. In other words, the 3D version presents a larger range of variability than the 2D version. As mentioned above, *DS-Hacker* 2D and 3D are the same except for their graphical representations and game physics. Therefore, the cause of the variations in the IMI scores can be attributed to these two game aspects.

Concerning *DS-Hacker 3D*, the 3D graphical representation allows a designer to create rich game environments. A “well-crafted” 3D game environment has the potential to increase the players’ curiosity to explore it and improve the players’ fantasy feeling. According to Malone and Lepper (1987), curiosity and fantasy are fundamental factors that enhance intrinsic motivation. However, 3D graphical representations may possess complex navigation mechanics and a larger number of control inputs. In this regard, *DS-Hacker 3D* is a PC 3D action-adventure game. This genre possesses complex navigation mechanics and requires the use of the keyboard and mouse at the same time to move the avatar properly. These features may cause frustration to players who are not familiar to the game genre, decreasing their intrinsic motivation. These two features of *DS-Hacker* 3D may explain the cause of the larger variability of IMI scores.

Regarding the *DS-Hacker 2D*, results of the descriptive analysis show that on average, the IMI scores of participants who played the 2D version are lower than the IMI scores of participants who played the 3D version. However, the 2D IMI scores present less variation and a more consistent and compact range than the 3D scores. Contrary to the 3D version, *DS-Hacker 2D* has a lower level of realism (e.g., 2D graphical representations, game environment, and simple game physics). Consequently, it also has simpler navigation game mechanics and fewer control inputs, e.g., it can be played only with the mouse. A simple game world that can be explored quickly reduces the players’ curiosity and consequently, their intrinsic motivation. However, players who are not familiar with action-adventure game mechanics and/or video games in general may find this option more adequate for their gaming skills, reducing frustration.

Finally, from a game developer’s perspective, there are differences that may be important to consider. First, developing a 2D video requires less technical skills. Modern 2D games engines (e.g., RPG Maker MV) are easy to learn and allow the creation of games without coding. Second, game development processes for 2D video games are faster than those for 3D video games. In this regard, the development process of *DS-Hacker 2D* took just a few weeks (approximately 4); meanwhile, *DS-Hacker 3D* took a few months (approximately 4). Third, 2D assets (e.g., sprite sheets, background images, etc) are easier, cheaper, and faster to create than 3D assets (e.g., 3D models, visual effects, game mechanics scripts, etc.). Fourth, 2D action-adventure games are easier to learn than 3D games of the same genre. This may be beneficial for students who are not familiar with video games. Finally, 2D video games requires less computation power than 3D games. This may be beneficial in educational environments that have low-spec (even outdated) equipment. Considering that 2D and 3D video games have the similar potential to teach conceptual knowledge and intrinsically motivate players, we conclude that 2D video games are better than 3D video games for learning environments.

7.2.4 Limitations

This study presents similar limitations to the previous evaluation. First, the sample size is small for an evaluation related to human psychological features. As mentioned in section 7.1.4, it is necessary to obtain larger samples to increase the reliability of the measurements and the power of the statistical methods (Parsons et al., 2019). Additionally, it is necessary to perform a replication of this evaluation with a population with different characteristics (e.g., different universities, programmes, geographical locations, cultures, etc.). Also, like in the first evaluation, we used a Spanish translation of the IMI survey, which has not been validated. Finally, due to the characteristics of our methodology, our findings do no explain *how* levels of realism affect learning and motivation; our observations only confirm *that* there are small learning and motivational differences between the two levels of realism evaluated. To fill this gap, it is necessary to perform a qualitative study using appropriate methodologies for those purposes.

7.3 CLG Framework Discussion

In this section, we discuss the validation and results of using the CLG framework presented in Chapter 4.

Regarding the rationale for the creation of the new framework, the CLG framework aims to mitigate the lack of comprehensive frameworks that consider and integrate the context, the learning aspects, and the game elements. Design frameworks, such as the four-dimensional framework, the G4LI framework, and the LM-GM framework, are specialized in that they focus on specific sections of the game design process and leave out important aspects of game design or learning design. For example, the four-dimensional framework focuses on the context where the learning experience will happen and the learner characteristics. However, it does not address game design aspects, such as the game concept, the narrative elements, or the game world. The CLG framework aims to fill these gaps by two means. First, it integrates learning design principles with game design principles for entertainment games and serious games for learning. Second, it organizes the game design process as an iterative process and divided it into three stages: (1) context; (2) learning; and (3) game. Each stage has its objectives and produces one or more artifacts that defines the requirements of the next stage.

In the context stage, the designer should define the context where the game is going to be used (e.g., learning environment, location, target audience, etc.) and developed (e.g., economical resources, deadlines, development team, development tools available, etc.). The designer should create a document with the characteristics (e.g., strengths and constrains) of the main aspects of context. In the learning stage, the designer should define the learning aspects of the game (e.g., the learning theory, pedagogical approach, contents, learning objective, learning activities, and assessment) following the alignment principle. Additionally, while defining the learning aspects, the designer must consider information gathered during the context stage. At the end of this stage, the designer should write a document with a description of the aspects.

Concerning the game stage, it is divided into three sub-stages: (3.1) game design; (3.2) development; and (3.3) testing. In the game design sub-stage, the designer should define the game elements (e.g., genre, game mechanics, game story, game world, aesthetics, etc.). Additionally, the designer should ensure that the game elements are appropriate for the context and that learning aspects are embedded into the game elements. At the end of this sub-stage, the designer should write a game design document, describing the game elements. In the development sub-stage, the designer and development team should develop a playable prototype of the game. The prototype should present the core elements of the game, including all the elements that convey learning. The prototype's main objective is testing purposes. In the testing sub-stage, the designer should test the prototype with a sample of the target population. At the end of this sub-stage, the designer should write a document with the results of the analysis and the discussion of the evaluations. The designer should use the results to decide whether it is necessary to iterate through the design process again.

Regarding the CLG framework validation, we validated it through the design, development, and testing of *DS-Hacker*, an action-adventure game for learning the BST data structure based on analogies embedded into the game mechanics and game model. Results of using the CLG framework were positive, and after three iterations, we obtained a satisfactory game prototype. Additionally, results of the evaluations of the final versions of *DS-Hacker* (3D and 2D) showed that they were effective and intrinsically motivating. We found three reasons why the CLG framework produced positive results.

First, the continuous testing and the iterative design characteristic of the CLG framework increased the probability of obtaining positive results. Regarding *DS-Hacker,* it was necessary to test and to iterate three times through the CLG framework workflow to develop a successful prototype. The first prototype was tested and validated by college professors. The professors' feedback showed that the first prototype had problems with the representations of the BST data structure. Therefore, we proceeded to correct the game document and to develop a second prototype. The second prototype was tested through a pilot experiment. Results of this experiment showed that the second prototype had problems relate to the context and learning assumptions. Therefore, it was necessary to correct the context document, learning aspects, and the game document. After these corrections, we developed and tested the third and final prototype of *DS-Hacker*. Results of the final evaluations were positive. Positive results would not be possible without testing and correction of the assumptions made in the previous stages. As suggested by the Iterative Design methodology (Salen & Zimmerman, 2004), player behaviour is difficult to predict. Additionally, it is difficult to consider all the factors (e.g., context factors and learning factors) that may affect the results. Therefore, an approach that allows testing and correction is desirable for game design of serious games for learning.

The second reason is defining the learning aspects following the alignment principle. The CLG framework suggests that the designer should adopt a learning theory to define the learning assumptions of how people learn. After identifying the learning assumptions, the designer should use them to guide the definition of the learning objectives and the selection of the pedagogical approaches, the learning activities, and the assessment methods. In other words, the designer should ensure that the learning aspects and learning assumptions are aligned. By doing so, the designer obtains three benefits. First, it increases the probability that the students achieve the learning objectives. Well-established learning theories have been tested by scholars and practitioners meticulously. The learning assumptions suggested by those theories have higher chances of providing positive results. Therefore, choosing pedagogical approaches and designing learning activities aligned with a good learning theory increase the probability of developing a successful learning tool. In this regard, *DS-Hacker* follows the learning assumptions suggested by Kolb's experiential learning theory. Its main pedagogical approach is the analogy, which facilitates the theoretical assumptions. The game's learning activities use the pedagogical approach, and they follow the learning assumptions. Second, it facilitates game design decisions. Game elements (e.g., the game mechanics, the game story, the game challenges, etc.) should embody the learning theory through the correct inclusion of the pedagogical approach. Additionally, the game elements should be the building blocks of the learning activities, which take the form of ludic activities. Consequently, the learning aspects guide the game design stage and keep the game coherent with the overall learning objective of the game for learning. In this regard, *DS-Hacker*'s game model and game mechanics embody the pedagogical approach (e.g., analogies), and the learning objective and learning activities guide the organization of *DS-Hacker*'s levels and challenges. Third, defining the learning aspects aligned with a learning theory facilitates the evaluation of the learning tool's efficacy and its learning assumptions easily. Usually, learning theories provide a list of well-defined principles that explain how people learn. During the evaluation of the learning tool, the designer can use these principles to verify whether the implementation of the learning activities follows the theory, facilitating the definition of factors evaluated.

The third reason why the CLG framework produced positive results was that it considers the context. The context considerations include use factors and development factors. The use factors define where, how, and who will use the learning. The development factors define the resources available to develop the learning tool. The context considerations provide a proper understanding of the strengths and constraints that the environment impose on the project. Therefore, the designer can define the scope of the learning tool appropriately.

Concerning the limitations of the CLG framework, its main weakness is that it has only been validated with one game, *DS-Hacker*. In other words, the framework has proved to be suitable to develop an action-adventure third-person PC game that targets undergraduate students who need to learn programming or computer sciences conceptual knowledge. We designed the CLG framework to be generic. The objective was that designers could use it to design games of various genres that focus on different areas of knowledge and that use different pedagogical approaches. However, currently, we cannot say that the framework will work in projects with different characteristics (e.g., a casual video game for learning high-school chemistry based on behavioural rewards). It is therefore necessary to validate the CLG framework in different contexts, using different learning aspects, and game genres. Additionally, if researchers and developers decide to adopt the CLG framework, further use of the framework will show whether the framework will be as useful as we think.

Another caveat of the CLG framework is that it addresses learning aspects from an academic stance. In other words, the framework requires the definition of a set of learning aspects (e.g., learning theory, clear learning goals, pedagogical approaches, assessments methodologies, etc.) which are common in formal education. However, an informal game for learning may not require all these elements (e.g., an assessment method). Therefore, defining all the learning aspects may in some situations be an unnecessary and overwhelming task that will not provide too many benefits to the game.

Another limitation of the CLG framework is that it relies on constant prototyping and testing. Prototyping requires additional resources, and it is time-consuming. Similarly, testing is time-consuming, and the designer requires a representative sample of the target population, which can be hard to recruit. Additionally, testing may require research methods and data analysis knowledge. Therefore, this approach may not be ideal for projects with short deadlines or limited resources.

Chapter 8: Conclusion

This chapter summarizes the thesis achievements, contributions, and discusses future research directions. The chapter is organized as follows.

8.1 Summary and Achievements

The thesis examined the field of video games for learning. Specifically, it studied video games that use analogies embedded in the game rules and game mechanics to achieve learning. The rationale is that analogies embedded in games elements, such as the game rules, narrative elements, and game mechanics, can relate familiar knowledge and abstract information to build new knowledge in an active way, which is a basic principle of constructivism. Furthermore, games can increase players' intrinsic motivation, which is related with optimal learning performance.

The thesis was structured to answer the overarching research question (***Can analogies embedded into the game rules and game mechanics effectively teach computer science conceptual knowledge?***), which was divided into five sub research questions (see section 1.2). The sub research questions were answered through the thesis and are summarized in the following paragraphs.

The first sub research question, SRQ 1, addressed matters concerning the game elements and pedagogical aspects of video games used by designers to create successful learning games. Additionally, it addressed matters regarding the methodologies used by other researchers to evaluate the efficacy of the games. SRQ 1 was answered through a systematic literature review presented in Chapter 3. The review informed us regarding the use of serious games for learning in the field of computer science topics, specifically in the field of data structures. Findings of the review were classified into four areas. The first area focused on game elements used by the games reviewed to convey the learning contents. Findings suggests that the game environment, the game challenges, and game mechanics convey learning in most of the games reviewed. The second area focused on the topics covered by the games. In this regard, our findings suggests that the most popular data structure was the stack, which was followed by the linked list and the queue data structures. The third area focused on the learning theories and pedagogical methods used by the game designers. Our findings show that the constructivist theories and pedagogical approaches are the most used by the games reviewed. Additionally, most of the games taught procedural knowledge and required *apply* cognitive processes solve the game challenges. The fourth area focused on the methodological aspects employed to evaluate the reviewed games.

Our findings showed methodological issues in the methodologies reviewed. First, most of the evaluations did not use validated measurement instruments. Second, evaluations that employed parametric methods did not verify the data assumptions required by the statistical methods. These two issues affect the inferences made in those studies. Additionally, only few studies performed qualitative evaluation of the game, and they were informal evaluations.

The second sub research question, SRQ 2, addressed design matters for video games for learning. SRQ 2 was answered in Chapter 4, which presented a brief literature review and culminated in the presentation of a new design framework called the Context-Learning-Game (CLG) framework. The CLG framework is based on the alignment principle as well as design frameworks for entertainment game and games for learning. The CLG framework suggests that the design process for games for learning is iterative and has three stages that cover key design aspects. The first stage covers the specific context where the learning experience will happen (e.g., resources, learner background, education environment, etc.). The second stage covers the learning aspect of the game (e.g., learning theory, pedagogical methods, learning objectives, etc.). The third stage focuses on the game design, prototyping and testing. The CLG framework was validated by designing and prototyping a video game for learning called *DS-Hacker* that teaches BST data structure conceptual knowledge and which was presented in Chapter 5.

The third sub research question, SRQ 3, focused on the assessment, which is a key aspect to evaluate the efficacy of our game for learning. SQR 3 was answered in Chapter 6. The chapter presented assessment validity concept and a brief review of literature related to validated assessment in computer science education. Then, we presented the methodology and results of the validation process of an assessment tool to measure learning gains about BST conceptual knowledge. Our assessment tool was successfully validated. Additionally, it has scale properties. In other words, measures produced by our assessment tool can be interpreted as interval data.

The fourth sub research question, SRQ 4, was answered in Chapter 7 (section 7.1). The section presented an evaluation of *DS-Hacker 3D* and a “traditional” digital learning experience (video tutorials). The evaluations assessed the learning gains and intrinsic motivation of participants in a higher education environment. Concerning the learning gains, results show that our game was slightly more effective than the video tutorials. Regarding the intrinsic motivation, the game was more motivating than the video tutorials.

Finally, the last sub research question, SRQ 5, was answered in Chapter 7 (section 7.2). The section presented an evaluation of *DS-Hacker* 3D and 2D. The evaluation assessed the participants’ learning gains and intrinsic motivations. Concerning the learning gains, results showed that both versions were effective. However, the 2D version was slightly more effective than the 3D version. Concerning the intrinsic motivation, results show that both games are motivating. However, there are differences between the two versions. The 3D version increased players' curiosity and fantasy feeling. However, it required high navigation skills from the player. On the other hand, the 2D was easier to play than the 3D version, making it convenient for beginner players.

To summarize, our findings show that video games for learning can teach abstract conceptual knowledge with an efficacy that is comparable to other "traditional" digital methods. However, regarding intrinsic motivation, they are more motivating than "traditional" digital methods. Regarding the pedagogical approach, analogies embedded in the game rules and game mechanics are effective and facilitate the acquisition of new information. Furthermore, the level of realism of video games can slightly affect learning gains. However, this is caused by game elements that increase the external cognitive load.

8.2 Contributions

The major contributions of this thesis are:

- **Systematic literature review** related to video games for learning data structures and recursion, which classifies the games reviewed based on their topics, learning aspects, and evaluation. This analytical tool can be used by designer and researchers to get insights about how video games that focus on data structures and recursion convey learning and the methods used to evaluate them.
- **The CLG framework**. This framework can be used by designers of video games for learning based on analogies for formal education environments. The framework provides guidelines and a workflow for designing and prototyping games for learning considering the context, learning aspects, and game design. Also, the framework allows designers to integrate other specialized frameworks in the design process.
- ***DS-Hacker***. The video game can be used in formal and informal educational environments to introduce BST conceptual knowledge. The game has two versions. The first version has 3D graphics. The second version has 2D graphics and can be used in education environments that has computer with low computational capacity. Furthermore, the 3D version is available in two languages (English and Spanish).
- **BST assessment tool**. The assessment tool can be used in research or education assessment. The tool measures BST conceptual knowledge, and it was fully validated. Measures

produced by our assessment tool has scale properties, allowing researchers and educators to analyse data using parametric statistical methods.

The following list presents the bibliography of peer review publications:

- Alberto Rojas-Salazar and Mads Haahr, DS-Hacker: Teaching Data Structures and Algorithms Through Analogical Representations in the Game Environment and Game Mechanics, *Digital Games Research Association (DiGRA)*, TBD, 2-6 June 2020, edited by Frans Mäyrä et al , 2020, pp 1 – 2
- Alberto Rojas-Salazar, Paula Ramírez-Alfaro and Mads Haahr, Learning Binary Search Trees through Serious Games, *First International Computer Programming Education Conference (ICPEC 2020)*, ESMAD, Vila do Conde, Portugal, 25-26 June 2020, edited by Ricardo Queirós, Filipe Portela, Mário Pinto, and Alberto Simões , ACM, 2020, pp22:1 - 22:7
- Alberto Rojas-Salazar and Mads Haahr, Learning Binary Search Trees through Serious Games based on Analogies, *Foundations of Digital Games (FDG 2020)*, Malta, 15-18 September 2020, edited by Antonios Liapis, Georgios N. Yannakakis, Penny Kyburz and Vanessa Volz , ACM, 2020
- Alberto Rojas-Salazar and Mads Haahr, Theoretical Foundations and Evaluations of Serious Games for Learning Data Structures and Recursion: A Review, *Joint Conference on Serious Games (JCSG)*, TBD, TBD, edited by Minhua Eunice Ma et al , Springer Verlag, 2020, pp1 – 2

8.3 Future Work

This thesis has shown that well-designed video games for learning based on analogies can teach conceptual knowledge effectively in higher education environments. This was proved through the development of a game design framework, a video game for learning that focuses on data structures, and two quantitative evaluations. However, due to the characteristics/limitations of this thesis, there are opportunities for further research that can contribute to field of game-based learning.

Qualitative evaluations. The evaluations presented in this thesis are quantitative; they only explain whether the game is effective and motivating. However, in order to understand how students learn using games and how students understand analogies in video games, it is necessary to perform qualitative evaluations. Qualitative research in this field is scarce (see section 4.4), and it may lead to better design strategies and use of video games in education environments.

Replication of the evaluations. The sample selected for the evaluations consisted of students from the University of Costa Rica and the National University of Colombia. However, it is advantageous to evaluate *DS-Hacker* using different samples from different locations, for example, undergraduate students from Mexico or Spain. Additionally, the game is translated in English. Therefore, it is also desirable to evaluate the game with English speaker students. Ideally, these replications should be performed using larger samples.

Games for learning complex data structures. As showed in Chapter 3, video games for learning data structures are scarce, and most of the games reviewed focus on simple data structures (e.g., stack, queue, linked list, etc). Video games for learning complex data structures (e.g., advanced trees and graphs) do not exists or have not been reported. This field has potential for being studied.

Framework validation. The CLG framework was only validated with an action-adventure game for learning BST conceptual knowledge based on analogies. However, it is necessary to validate the framework with different game genres, learning approaches, topics, audiences, and learning environments.

Bibliography

Adams, E. (2014). *Fundamentals of Game Design* (3rd ed.). New Riders Publishing.

Adams, E., & Dormans, J. (2012). *Game Mechanics: Advanced Game Design* (1st ed.). New Riders Publishing.

Aho, A. V. (1988). *Estructuras de datos y algoritmos*. Addison-Wesley.

Ak, O., & Kutlu, B. (2017). Comparing 2D and 3D game-based learning environments in terms of learning gains and student perceptions. *British Journal of Educational Technology, 48*(1), 129–144. https://doi.org/10.1111/bjet.12346

Alexey Pajitnov & Vladimir Pokhilko. (1984). *Tetris* [Electronika 60].

Almstrum, V. L., Henderson, P. B., Harvey, V., Heeren, C., Marion, W., Riedesel, C., Soh, L.-K., & Tew, A. E. (2006). Concept inventories in computer science for the topic discrete mathematics. In *Working group reports on ITiCSE on Innovation and technology in computer science education* (pp. 132–145). Association for Computing Machinery. https://doi-org.elib.tcd.ie/10.1145/1189215.1189182

American Educational Research Association. (2014). *Standards for educational and psychological testing*. American Educational Research Association.

Amory, A. (2007). Game object model version II: a theoretical framework for educational game development. *Educational Technology Research and Development*, *55*(1), 51–77. https://doi.org/10.1007/s11423-006-9001-x

Anderson, L. W., & Krathwohl, D. R. (2001). *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives*. Longman; /z-wcorg/.

Arnab, S., Lim, T., Carvalho, M. B., Bellotti, F., de Freitas, S., Louchart, S., Suttie, N., Berta, R., & De Gloria, A. (2015). Mapping learning and game mechanics for serious games analysis. *British Journal of Educational Technology*, *46*(2), 391–411. eft.

Association for Computing Machinery (ACM) Joint Task Force on Computing Curricula & IEEE Computer Society. (2013). *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. ACM.

Atari Games. (1986). *Super Sprint* [Arcade]. Atari Games.

Atkinson, R. C., & Shiffrin, R. M. (1968). Human Memory: A Proposed System and its Control Processes. In K. W. Spence & J. T. Spence (Eds.), *Psychology of Learning and Motivation* (Vol. 2, pp. 89–195). Academic Press. https://doi.org/10.1016/S0079-7421(08)60422-3

Aubusson, P. J., Harrison, A. G., & Ritchie, S. M. (2006). Metaphor and Analogy: Serious thought in science education. In *Metaphor and Analogy in Science Education* (pp. 1–9). Springer.

Baddeley, A. D., & Hitch, G. (1974). Working Memory. In G. H. Bower (Ed.), *Psychology of Learning and Motivation* (Vol. 8, pp. 47–89). Academic Press. https://doi.org/10.1016/S0079-7421(08)60452-1

Bateman, C., & Boon, R. (2005). *21st Century Game Design (Game Development Series)*. Charles River Media, Inc.

Bates, B. (2016). *Learning Theories Simplified*. Sage Publications Ltd.

Becker, K. (2017). *Choosing and Using Digital Games in the Classroom*. Springer Nature.

Becker, K., & Beacham, M. (2000). A Tool for Teaching Advanced Data Structures to Computer Science Students: An Overview of the BDP System. *Proceedings of the Second Annual CCSC on Computing in Small Colleges Northwestern Conference*, 65–71. http://dl.acm.org/citation.cfm?id=369274.369319

Biggs, J., & Tang, C. (2007). *Teaching for Quality Learning at University* (3rd Edition). Open University Press.

Bond, T. G., & Fox, C. M. (2015). *Applying the Rach Model: Fundamental Measurement in the Human Science* (Third Edition). Routledge.

Boyle, E. A., Connolly, T. M., Hainey, T., & Boyle, J. M. (2012). Engagement in digital entertainment games: A systematic review. *Computers in Human Behavior*, *28*(3), 771–780. https://doi.org/10.1016/j.chb.2011.11.020

Bruce, B. C., & Bloch, N. (2012). Learning by Doing. In N. M. Seel (Ed.), *Encyclopedia of the Sciences of Learning* (pp. 1821–1824). Springer US. https://doi.org/10.1007/978-1-4419-1428-6_544

Bueno, D. (2018). *Neurociencia para educadores: Todo lo que los educadores siempre han querido saber sobre el cerebro de sus alumnos y nunca nadie se ha atrevido a explicárselo de manera comprensible y útil* (2nd ed.). Octaedro.

Caceffo, R., Wolfman, S., Booth, K. S., & Azevedo, R. (2016). Developing a Computer Science Concept Inventory for Introductory Programming. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (pp. 364–369). Association for Computing Machinery. https://doi.org/10.1145/2839509.2844559

Caillois, R. (1961). *Man, play, and games*. Free Press of Glencoe; /z-wcorg/.

Calderón, A., & Ruiz, M. (2015). A systematic literature review on serious games evaluation: An application to software project management. *Computers & Education*, *87*, 396–422. https://doi.org/10.1016/j.compedu.2015.07.011

Calleja, G. (2007). Digital Game Involvement: A Conceptual Model. *Games and Culture*, *2*(3), 236–260. https://doi.org/10.1177/1555412007306206

Campbell, J. (1991). *The hero's journey*. Harpercollins.

Chaffin, A., Doran, K., Hicks, D., & Barnes, T. (2009). Experimental Evaluation of Teaching Recursion in a Video Game. *Proceedings of the 2009 ACM SIGGRAPH Symposium on Video Games*, 79–86. https://doi.org/10.1145/1581073.1581086

Cheng, M.-T., Chen, J.-H., Chu, S.-J., & Chen, S.-Y. (2015). The use of serious games in science education: A review of selected empirical research from 2002 to 2013. *Journal of Computers in Education*, *2*(3), 353–375. https://doi.org/10.1007/s40692-015-0039-9

Cooper, A., Reimann, R., Cronin, D., & Noessel, C. (2014). *About Face: The Essentials of Interaction Design*. Wiley Publishing.

Costa, E. B., Toda, A. M., Mesquita, M. A. A., Matsunaga, F. T., & Brancher, J. D. (2014). Interactive Data Structure Learning Platform. In B. Murgante, S. Misra, A. M. A. C. Rocha, C. Torre, J. G. Rocha, M. I. Falcão, D. Taniar, B. O. Apduhan, & O. Gervasi (Eds.), *Computational Science and Its Applications – ICCSA 2014* (pp. 186–196). Springer International Publishing.

Crawford, C. (1984). *The Art of Computer Game Design*. McGraw-Hill.

Cronbach, L. J., & Meehl, P. E. (1955). Construct validity in psychological tests. *Psychological Bulletin*, *52*(4), 281–302. https://doi.org/10.1037/h0040957

Csikszentmihalyi, M. (1990). *Flow: The Psychology of Optimal Experience*. Harper & Row.

Danielsiek, H., Paul, W., & Vahrenhold, J. (2012). Detecting and understanding students' misconceptions related to algorithms and data structures. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education* (pp. 21–26). Association for Computing Machinery. https://doi-org.elib.tcd.ie/10.1145/2157136.2157148

Darina Dicheva, Christo Dichev, Gennady Agre, & Galia Angelova. (2015). Gamification in Education: A Systematic Mapping Study. *Journal of Educational Technology & Society*, *18*(3), 75–88. JSTOR.

Davis, F. D., Bagozzi, R. P., & Warshaw, P. R. (1989). User Acceptance of Computer Technology: A Comparison of Two Theoretical Models. *Management Science*, *35*(8), 982–1003. JSTOR.

de Freitas, S. I. (2006). *Learning in immersive worlds: A review of game-based learning*. Joint Information Systems Committee.

de Freitas, S. I., & Oliver, M. (2006). How can exploratory learning with games and simulations within the curriculum be most effectively evaluated? *Virtual Learning*, *46*(3), 249–264.

Deci, E. L., & Ryan, R. M. (1985a). Cognitive Evaluation Theory. In E. L. Deci & R. M. Ryan (Eds.), *Intrinsic Motivation and Self-Determination in Human Behavior* (pp. 43–85). Springer US. https://doi.org/10.1007/978-1-4899-2271-7_3

Deci, E. L., & Ryan, R. M. (1985b). Toward an Organismic Integration Theory. In E. L. Deci & R. M. Ryan (Eds.), *Intrinsic Motivation and Self-Determination in Human Behavior* (pp. 113–148). Springer US. https://doi.org/10.1007/978-1-4899-2271-7_5

Deek, F. P., Hiltz, S. R., Kimmel, H., & Rotter, N. (1999). Cognitive Assessment of Students' Problem Solving and Program Development Skills. *Journal of Engineering Education*, *88*(3), 317–326. https://doi.org/10.1002/j.2168-9830.1999.tb00453.x

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 & Gaming*, *33*(2), 157–168. https://doi.org/10.1177/1046878102332003

Dicheva, D., & Hodge, A. (2018). Active Learning Through Game Play in a Data Structures Course. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 834–839. https://doi.org/10.1145/3159450.3159605

Dicheva, D., Hodge, A., Dichev, C., & Irwin, K. (2016). On the design of an educational game for a Data Structures course. *2016 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, 14–17. https://doi.org/10.1109/TALE.2016.7851763

Donnelly, C. M., & McDaniel, M. A. (1993). Use of analogy in learning scientific concepts. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *19*(4), 975–987. https://doi.org/10.1037/0278-7393.19.4.975

Dörner, R., Göbel, S., Effelsberg, W., & Wiemeyer, J. (2016). Introduction. In R. Dörner, S. Göbel, W. Effelsberg, & J. Wiemeyer (Eds.), *Serious Games: Foundations, Concepts and Practice* (pp. 1–34). Springer International Publishing. https://doi.org/10.1007/978-3-319-40612-1_1

Doukakis, D., Tsaganou, G., & Grigoriadou, M. (2007). Using animated interactive analogies in teaching basic programming concepts and structures. *Proceedings of the Informatics Education Europe II Conference IEEII 2007*.

Duit, R. (1991). On the role of analogies and metaphors in learning science. *Science Education*, *75*(6), 649–672.

Eagle, M., & Barnes, T. (2009). Experimental Evaluation of an Educational Game for Improved Learning in Introductory Computing. *Proceedings of the 40th ACM Technical Symposium on Computer Science Education*, 321–325. https://doi.org/10.1145/1508865.1508980

Eagle, M., & Barnes, T. (2008). Wu's Castle: Teaching Arrays and Loops in a Game. *Proceedings of the 13th Annual Conference on Innovation and Technology in Computer Science Education*, 245–249. https://doi.org/10.1145/1384271.1384337

Farghally, M. F., Koh, K. H., Ernst, J. V., & Shaffer, C. A. (2017). Towards a Concept Inventory for Algorithm Analysis Topics. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 207–212). Association for Computing Machinery. https://doi-org.elib.tcd.ie/10.1145/3017680.3017756

Fernández-Vara, C. (2014). *Introduction to Game Analysis* (1st ed.). Routledge.

Fernández-Vara, C. (2011). From ‘open mailbox’ to context mechanics: Shifting levels of abstraction in adventure games. *Proceedings of the 6th International Conference on Foundations of Digital Games*, 131–138.

Field, A. (2013). *Discovering Statistics Using IBM SPSS Statistics* (4th ed.). Sage Publications Ltd.

Fleury, A. E. (1991). Parameter passing: The rules the students construct. In *SIGCSE Bull.* (Vol. 23, Issue 1, pp. 283–286). Association for Computing Machinery.

Frasca, G. (2003). Simulation versus Narrative: Introduction to Ludology. In M. J. P. Wolf & B. Perron (Eds.), *The Video Game Theory Reader*. Routledge.

Frété, C. (2002). *The potential of video games for education* [DESS SATF dissertation]. Geneve University.

Gee, J. P. (2007). *What Video Games Have to Teach Us about Learning and Literacy* (2nd ed.). Palgrave Macmillan.

Glynn, S. M. (1994). Teaching Science with Analogies: A Strategy for Teachers and Textbook Authors. Reading Research Report No. 15. *ERIC*.

Gogus, A. (2012). Constructivist Learning. In N. M. Seel (Ed.), *Encyclopedia of the Sciences of Learning* (pp. 783–786). Springer US. https://doi.org/10.1007/978-1-4419-1428-6_4049

Haladyna, T. M., Downing, S. M., & Rodriguez, M. C. (2002). A Review of Multiple-Choice Item-Writing Guidelines for Classroom Assessment. *Applied Measurement in Education*, *15*(3), 309–334.

Harteveld, C. (2011). *Triadic Game Design: Balancing Reality, Meaning and Play*. Springer Publishing Company, Incorporated.

Hasbro. (1959). *Risk*.

Hays, R. T., & Singer, M. J. (1989). Simulation Fidelity as an Organizing Concept. In R. T. Hays & M. J. Singer (Eds.), *Simulation Fidelity in Training System Design: Bridging the Gap Between Reality and Training* (pp. 47–75). Springer New York. https://doi.org/10.1007/978-1-4612-3564-4_3

Hodent, C. (2017). *The Gamer’s Brain: How Neuroscience and UX Can Impact Video Game Design*. CRC Press.

Howe, A. C. (1993). A Vygotskian Perspective on Teaching for Conceptual Change. *The Proceedings of the Third International Seminar on Misconceptions and Educational Strategies in Science and Mathematics*.

Huizinga, J. (1955). *Homo ludens: A study of the play-element in culture*. /z-wcorg/.

Hundhausen, C. D., DOUGLAS, S. A., & STASKO, J. T. (2002). A Meta-Study of Algorithm Visualization Effectiveness. *Journal of Visual Languages & Computing*, *13*(3), 259–290. https://doi.org/10.1006/jvlc.2002.0237

Hunicke, R., Leblanc, M., & Zubek, R. (2004). *MDA: A Formal Approach to Game Design and Game Research* (Vol. 1).

IEEE Standard Glossary of Modeling and Simulation Terminology. (1989). *IEEE Std 610.3-1989*, 1–0. https://doi.org/10.1109/IEEESTD.1989.94599

Indurkhya, B. (1991). On the role of interpretive analogy in learning. *New Generation Computing*, *8*(4), 385–402. https://doi.org/10.1007/BF03037095

Ismail, M., Ghafar, N., & Diah, N. (2013). *Realization of Conceptual Knowledge Through Educational Game*. CGAT 2013, the 6th Annual International Conference on Computer Games, Multimedia and Allied Technologies. https://doi.org/10.5176/2251-1679_CGAT13.06

Jackson, M. (2012). Deep Approaches to Learning. In N. M. Seel (Ed.), *Encyclopedia of the Sciences of Learning* (pp. 913–913). Springer US. https://doi.org/10.1007/978-1-4419-1428-6_3766

Jiau, H. C., Chen, J. C., & Ssu, K. (2009). Enhancing Self-Motivation in Learning Programming Using Game-Based Simulation and Metrics. *IEEE Transactions on Education*, *52*(4), 555–562. https://doi.org/10.1109/TE.2008.2010983

Johnstone, A. H. (1991). Why is science difficult to learn? Things are seldom what they seem. *Journal of Computer Assisted Learning*, *7*(2), 75–83. https://doi.org/10.1111/j.1365-2729.1991.tb00230.x

Juul, J. (2005). *Half-Real: Video Games Between Real Rules and Fictional Worlds*. The MIT Press.

Kaczmarczyk, L. C., Petrick, E. R., East, J. P., & Herman, G. L. (2010). Identifying student misconceptions of programming. In *Proceedings of the 41st ACM technical symposium on Computer science education* (pp. 107–111). Association for Computing Machinery. https://doi-org.elib.tcd.ie/10.1145/1734263.1734299

Kaimara, P., Fokides, E., Plerou, A., Atsikpasi, P., & Deliyannis, I. (2020). Serious Games Effect Analysis On Player’s Characteristics. *International Journal of Smart Education and Urban Society (IJSEUS)*, *11*(1), 75–91. https://doi.org/10.4018/IJSEUS.2020010106

Kannappan, V. T., Fernando, O. N. N., Chattopadhyay, A., Tan, X., Hong, J. Y. J., Seah, H. S., & Lye, H. E. (2019). La Petite Fee Cosmo: Learning Data Structures Through Game-Based Learning. *2019 International Conference on Cyberworlds (CW)*, 207–210. https://doi.org/10.1109/CW.2019.00041

Kapur, M. (2014). Productive Failure in Learning Math. *Cognitive Science*, *38*(5), 1008–1022. https://doi.org/10.1111/cogs.12107

Karpierz, K., & Wolfman, S. A. (2014). Misconceptions and concept inventory questions for binary search trees and hash tables. In *Proceedings of the 45th ACM technical* *symposium on Computer science education* (pp. 109–114). Association for Computing Machinery. https://doi-org.elib.tcd.ie/10.1145/2538862.2538902

Kempa, R. (1991). Students' learning difficulties in science. Causes and possible remedies. *Enseñanza de Las Ciencias. Revista de Investigación y Experiencias Didáticas*, *9*(2), 119–128.

Kitchenham, B. (2004). *Procedures for Performing Systematic Reviews. Technical Report TR/SE-0401*. Keele University and NICTA.

Kolb, A. Y., & Kolb, D. A. (2012). Experiential Theory Learning. In N. M. Seel (Ed.), *Encyclopedia of the Sciences of Learning* (pp. 1215–1219). Springer US. https://doi.org/10.1007/978-1-4419-1428-6_4049

Kolb, D. A. (2014). *Experiential Learning: Experience as the source of learning and development* (2nd ed.). Pearson.

Kutner, L., & Olson, C. (2008). *Grand theft childhood: The surprising truth about violent video games and what parents can do.* (p. 260). Simon & Schuster.

Lave, J. (1991). *Situated learning: Legitimate peripheral participation*. Cambridge [England] ; New York : Cambridge University Press, 1991. https://search.library.wisc.edu/catalog/999668157902121

Lawrence, R. (2004). Teaching data structures using competitive games. *IEEE Transactions on Education*, *47*(4), 459–466. https://doi.org/10.1109/TE.2004.825053

Lazar, J., Feng, J. H., & Hochheiser, H. (2017). *Research Methods in Human-Computer Interaction* (Second). Morgan Kaufmann Publishers.

Lee, R. W., Tse, A. C., & Wong, T. W. (2019). Application of Analogy in Learning Badminton Among Older Adults: Implications for Rehabilitation. *Motor Control*, *23*(3), 384–397. https://doi.org/10.1123/mc.2017-0037

Liu, T., Chu, Y., & Tan, T. (2012). Using computer games in a computer course to improve learning. *Proceedings of IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE) 2012*, W2C-16-W2C-19. https://doi.org/10.1109/TALE.2012.6360301

Mair, P. (2018). Item Response Theory. In P. Mair (Ed.), *Modern Psychometrics with R* (pp. 95–159). Springer International Publishing. https://doi.org/10.1007/978-3-319-93177-7_4

Malone, T. W. (1980). What makes things fun to learn? Heuristics for designing instructional computer games. *Proceedings of the 3rd ACM SIGSMALL Symposium and the First SIGPC Symposium on Small Systems*, 162–169.

Malone, T. W., & Lepper, M. R. (1987). Making Learning Fun: A Taxonomy of Intrinsic Motivations for Learning. In R. E. Snow & M. J. Farr (Eds.), *Aptitude, Learning and Instruction III: Conative and Affective Process Analyses*. Erlbaum.

Martínez Arias, M. R., Hernández Lloreda, M. V., & Hernández Lloreda, M. J. (2006). *Psicometría*. Alianza Editorial.

Matthews, R., & Hua Liu, C. (2012). Play and Its Role in Learning. In N. M. Seel (Ed.), *Encyclopedia of the Sciences of Learning* (pp. 2647–2650). Springer US. https://doi.org/10.1007/978-1-4419-1428-6_1658

Maxis. (2000). *The Sims*. Electronic Arts.

Mayer, R. E., & Moreno, R. (2003). Nine Ways to Reduce Cognitive Load in Multimedia Learning. *Educational Psychologist*, *38*(1), 43–52. https://doi.org/10.1207/S15326985EP3801_6

Mayes, T., & de Freitas, S. (2004). *Review of e-learning theories, frameworks and models*. Joint Information Systems Committee.

McAuley, E., Duncan, T., & Tammen, V. V. (1989). Psychometric properties of the Intrinsic Motivation Inventory in a competitive sport setting: A confirmatory factor analysis. *Research Quarterly for Exercise and Sport*, *60*(1), 48–58. https://doi.org/10.1080/02701367.1989.10607413

McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y. B.-D., 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. In *Working group reports from ITiCSE on Innovation and technology in computer science education* (pp. 125–180). Association for Computing Machinery. https://doi-org.elib.tcd.ie/10.1145/572133.572137

Mellor, D. H. (1968). Models and Analogies in Science: Duhem versus Campbell? *Isis*, *59*(3), 282–290. JSTOR.

MicroProse. (1987). *Pirates!* [Commodore 64]. MicroProse.

MicroProse. (1991). *Civilization* [MS-DOS]. MicroProse.

Monteiro, V., Mata, L., & Peixoto, F. (2015). Intrinsic Motivation Inventory: Psychometric Properties in the Context of First Language and Mathematics Learning. *Psicologia: Reflexão e Crítica*, *28*, 434–443.

Moreno, R., Martínez, R., & Muñiz, J. (2004). Directrices para la construcción de ítems de elección múltiple. *Psicothema*, *16*(3), 490–497.

Namco. (1983). *Xevious* [Arcade]. Namco.

Naps, T. L., Rößling, G., Almstrum, V., Dann, W., Fleischer, R., Hundhausen, C., Korhonen, A., Malmi, L., McNally, M., Rodger, S., & Velázquez-Iturbide, J. Á. (2002). Exploring the Role of Visualization and Engagement in Computer Science Education. *SIGCSE Bull.*, *35*(2), 131–152. https://doi.org/10.1145/782941.782998

Nielsen, T. (2018). *The intrinsic and extrinsic motivation subscales of the Motivated Strategies for Learning Questionnaire: A Rasch-based construct validity study*. *5*, 1–19. https://doi.org/10.1080/2331186X.2018.1504485

Nintendo. (1985). *Super Mario Bros.* [Nintendo Entertainment System]. Nintendo.

Park, B., & Ahmed, D. T. (2017). Abstracting Learning Methods for Stack and Queue Data Structures in Video Games. *2017 International Conference on Computational Science and Computational Intelligence (CSCI)*, 1051–1054. https://doi.org/10.1109/CSCI.2017.183

Parsons, S., Kruijt, A.-W., & Fox, E. (2019). Psychological Science Needs a Standard Practice of Reporting the Reliability of Cognitive-Behavioral Measurements. In *Advances in Methods and Practices in Psychological Science* (Vol. 2, Issue 4, pp. 378–395).

Perini, S., Luglietti, R., Margoudi, M., Oliveira, M., & Taisch, M. (2018). Learning and motivational effects of digital game-based learning (DGBL) for manufacturing education –The Life Cycle Assessment (LCA) game. *Computers in Industry*, *102*, 40–49. https://doi.org/10.1016/j.compind.2018.08.005

Petri, G., & Gresse von Wangenheim, C. (2017). How Games for Computing Education Are Evaluated? A Systematic Literature Review. *Comput. Educ.*, *107*(C), 68–90. https://doi.org/10.1016/j.compedu.2017.01.004

Phillips, D. C. (2012). Behaviorism and Behaviorist Learning Theories. In N. M. Seel (Ed.), *Encyclopedia of the Sciences of Learning* (pp. 438–442). Springer US. https://doi.org/10.1007/978-1-4419-1428-6_750

Pink, D. H. (2009). *Drive: The surprising truth about what motivates us*. /z-wcorg/.

Plass, J. L., Homer, B. D., Hayward, E. O., Frye, J., Huang, T.-T., Biles, M., Stein, M., & Perlin, K. (2012). The Effect of Learning Mechanics Design on Learning Outcomes in a Computer-Based Geometry Game. In S. Göbel, W. Müller, B. Urban, & J. Wiemeyer (Eds.), *E-Learning and Games for Training, Education, Health and Sports* (pp. 65–71). Springer Berlin Heidelberg.

Plass, J. L., Homer, B. D., & Kinzer, C. K. (2014). *Playful Learning: An Integrated Design Framework [White Paper]* (pp. 1–30) [White Paper]. Games For Learning Institute, New York University. https://www.researchgate.net/profile/Jan_Plass/publication/272815274_Playful_Learning_An_Integrated_Design_Framework/links/54ef369a0cf25f74d721c0fe/Playful-Learning-An-Integrated-Design-Framework.pdf?origin=publication_detail

Plass, J. L., Homer, B. D., & Kinzer, C. K. (2015). Foundations of Game-Based Learning. *Educational Psychologist*, *50*(4), 258–283. https://doi.org/10.1080/00461520.2015.1122533

Plass, J. L., Homer, B. D., Kinzer, C. K., Frye, J. M., & Perlin, K. (2011). *Learning Mechanics and Assessment Mechanics for Games for Learning [White Paper]* (pp. 1–19) [White Paper]. Games For Learning Institute, New York University. https://www.researchgate.net/profile/Jan_Plass/publication/272815253_Learning_Mechanics_and_Assessment_Mechanics_for_Games_for_Learning/links/54ef353a0cf2432ba656273d/Learning-Mechanics-and-Assessment-Mechanics-for-Games-for-Learning.pdf?origin=publication_detail

Podolefsky, N. S., & Finkelstein, N. D. (2006). Use of analogy in learning physics: The role of representations. *Phys. Rev. ST Phys. Educ. Res.*, *2*(2), 020101. https://doi.org/10.1103/PhysRevSTPER.2.020101

Podolskiy, A. I. (2012). Zone of Proximal Development. In N. M. Seel (Ed.), *Encyclopedia of the Sciences of Learning* (pp. 3485–3487). Springer US. https://doi.org/10.1007/978-1-4419-1428-6_316

Porter, L., Zingaro, D., Lee, C., Taylor, C., Webb, K. C., & Clancy, M. (2018). Developing Course-Level Learning Goals for Basic Data Structures in CS2. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (pp. 858–863). Association for Computing Machinery. https://doi-org.elib.tcd.ie/10.1145/3159450.3159457

Porter, L., Zingaro, D., Liao, S. N., Taylor, C., Webb, K. C., Lee, C., & Clancy, M. (2019). BDSI: A Validated Concept Inventory for Basic Data Structures. *Proceedings of the 2019 ACM Conference on International Computing Education Research*, 111–119. https://doi.org/10.1145/3291279.3339404

Qian, Y., & Lehman, J. (2017). Students' Misconceptions and Other Difficulties in Introductory Programming: A Literature Review. In *ACM Trans. Comput. Educ.* (Vol. 18, Issue 1, p. Article 1). Association for Computing Machinery.

Ramle, R., Rosli, D. I., Nathan, S. S., & Berahim, M. (2019). Digital Game Based Learning of Stack Data Structure Using Question Prompts. *International Journal of Interactive Mobile Technologies*, *13*(7), 90–102.

Randolph, J. J., Julnes, G., Sutinen, E., & Lehman, S. (2008). A Methodological Review of Computer Science Education Research. *Journal of Information Technology Education: Research*, *7*(1), 135–162.

Rieber, L. P. (1996). Seriously considering play: Designing interactive learning environments based on the blending of microworlds, simulations, and games. *Educational Technology Research and Development*, *44*(2), 43–58. https://doi.org/10.1007/BF02300540

Rojas-Salazar, A., & Haahr, M. (2020a). DS-Hacker: Teaching Data Structures and Algorithms Through Analogical Representations in the Game Environment and Game Challenges. *DiGRA '20 — Proceedings of the 2020 DiGRA International Conference: Play*

Everywhere. http://www.digra.org/wp-content/uploads/digital-library/DiGRA_2020_paper_141.pdf

Rojas-Salazar, A., & Haahr, M. (2020b). Learning Binary Search Trees through Serious Games Based on Analogies. *International Conference on the Foundations of Digital Games*. https://doi.org/10.1145/3402942.3402999

Rojas-Salazar, A., & Haahr, M. (2020c). Theoretical Foundations and Evaluations of Serious Games for Learning Data Structures and Recursion: A Review. In M. Ma, B. Fletcher, S. Göbel, J. Baalsrud Hauge, & T. Marsh (Eds.), *Serious Games* (pp. 135–149). Springer International Publishing.

Rojas-Salazar, A., Ramírez-Alfaro, P., & Haahr, M. (2020). Learning Binary Search Trees Through Serious Games. In R. Queirós, F. Portela, M. Pinto, & A. Simões (Eds.), *First International Computer Programming Education Conference (ICPEC 2020)* (Vol. 81, p. 22:1-22:7). Schloss Dagstuhl–Leibniz-Zentrum für Informatik. https://doi.org/10.4230/OASIcs.ICPEC.2020.22

Rollings, A., & Adams, E. (2003). *Andrew Rollings and Ernest Adams on Game Design*. New Riders Games.

Ryan, R. M. (1982). Control and information in the intrapersonal sphere: An extension of cognitive evaluation theory. *Journal of Personality and Social Psychology*, *43*(3), 450–461. https://doi.org/10.1037/0022-3514.43.3.450

Ryan, R. M., & Deci, E. L. (2000a). Intrinsic and Extrinsic Motivations: Classic Definitions and New Directions. *Contemporary Educational Psychology*, *25*(1), 54–67. https://doi.org/10.1006/ceps.1999.1020

Ryan, R. M., & Deci, E. L. (2000b). Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *American Psychologist*, *55*(1), 68–78. https://doi.org/10.1037/0003-066X.55.1.68

Ryan, T. (1999a, December 17). *The Anatomy of a Design Document, Part 1: Documentation Guidelines for the Game Concept and Proposal*. Gamasutra. http://www.gamasutra.com/view/feature/131791/the_anatomy_of_a_design_document_.php

Ryan, T. (1999b, December 17). *The Anatomy of a Design Document, Part 2: Documentation Guidelines for the Functional and Technical Specifications*. Gamasutra. https://www.gamasutra.com/view/feature/131818/the_anatomy_of_a_design_document_.php

Salen, K., & Zimmerman, E. (2004). *Rule of play: Game design fundamentals*. The MIT Press.

Sanders, K., Sheard, J., Becker, B. A., Eckerdal, A., & HamoudaSimon, S. (2019). Inferential Statistics in Computing Education Research: A Methodological Review. In *Proceedings of the 2019 ACM Conference on International Computing Education Research* (pp.

177–185). Association for Computing Machinery. https://doi-org.elib.tcd.ie/10.1145/3291279.3339408

Schell, J. (2014). *The Art of Game Design: A Book of Lenses* (2nd ed.). A. K. Peters, Ltd.

Schrader, C., & Bastiaens, T. J. (2012). The influence of virtual presence: Effects on experienced cognitive load and learning outcomes in educational computer games. *Computers in Human Behavior*, *28*(2), 648–658. https://doi.org/10.1016/j.chb.2011.11.011

Sedgewick, R., & Wayne, K. (2014). *Algorithms* (4th ed.). Addison-Wesley.

Seel, N. M., & Blumschein, P. (2009). Modelling and Simulation in Learning and Instruction: A Theoretical Perspective. In P. Blumschein, W. Hung, & J. Strobel (Eds.), *Model-Based Approaches to Learning: Using systems models and simulations to improve understanding and problem solving in complex domains* (Vol. 4, pp. 3–15). Sense Publishers.

Shabanah, S. S., Chen, J. X., Wechsler, H., Carr, D., & Wegman, E. (2010). Designing Computer Games to Teach Algorithms. *2010 Seventh International Conference on Information Technology: New Generations*, 1119–1126. https://doi.org/10.1109/ITNG.2010.78

Sicart, M. (2008). Defining Game Mechanics. *Game Studies*, *8*(2). http://gamestudies.org/0802/articles/sicart

Sirkiä, T., & Sorva, J. (2012). Exploring Programming Misconceptions: An Analysis of Student Mistakes in Visual Program Simulation Exercises. *Proceedings of the 12th Koli Calling International Conference on Computing Education Research*, 19–28. https://doi.org/10.1145/2401796.2401799

Skulmoski, G. J., Hartman, F. T., & Krahn, J. (2007). The Delphi Method for Graduate Research. *Journal of Information Technology Education: Research*, *6*(1), 1–21.

Sleeman, D., Putnam, R. T., Baxter, J., & Kuspa, L. (1986). Pascal and High School Students: A Study of Errors. *Journal of Educational Computing Research*, *2*(1), 5–23. https://doi.org/10.2190/2XPP-LTYH-98NQ-BU77

Squire, K. (2011). *Video Games and Learning: Teaching and Participatory Culture in the Digital Age*. Teachers College Press.

Šuníková, D., Kubincová, Z., & Byrtus, M. (2018). A Mobile Game to Teach AVL Trees. *2018 16th International Conference on Emerging ELearning Technologies and Applications (ICETA)*, 541–544. https://doi.org/10.1109/ICETA.2018.8572263

Svinicki, M. D., & Vogler, J. S. (2012). Motivation and Learning: Modern Theories. In N. M. Seel (Ed.), *Encyclopedia of the Sciences of Learning* (pp. 2336–2339). Springer US. https://doi.org/10.1007/978-1-4419-1428-6_392

Sweller, J. (2011). *CHAPTER TWO - Cognitive Load Theory* (J. P. Mestre & B. H. Ross, Eds.; Vol. 55, pp. 37–76). Academic Press. https://doi.org/10.1016/B978-0-12-387691-1.00002-8

Sweller, J. (2012). Cognitive Load Theory. In N. M. Seel (Ed.), *Encyclopedia of the Sciences of Learning* (pp. 601–605). Springer US. https://doi.org/10.1007/978-1-4419-1428-6_446

Taito. (1986). *Arkanoid* [Arcade]. Taito.

Taylor, C., Zingaro, D., Porter, L., Webb, K. C., Lee, C. B., & Clancy, M. (2014). Computer science concept inventories: Past and future. *Computer Science Education*, *24*(4), 253–276. https://doi.org/10.1080/08993408.2014.970779

Tew, A. E., & Guzdial, M. (2010). Developing a validated assessment of fundamental CS1 concepts. In *Proceedings of the 41st ACM technical symposium on Computer science education* (pp. 97–101). Association for Computing Machinery. https://doi.org/10.1145/1734263.1734297

Tew, A. E., & Guzdial, M. (2011). The FCS1: A language independent assessment of CS1 knowledge. In *Proceedings of the 42nd ACM technical symposium on Computer science education* (pp. 111–116). Association for Computing Machinery. https://doi-org.elib.tcd.ie/10.1145/1953163.1953200

Touré-Tillery, M., & Fishbach, A. (2014). How to Measure Motivation: A Guide for the Experimental Social Psychologist. *Social and Personality Psychology Compass*, *8*. https://doi.org/10.1111/spc3.12110

Treagust, D., Duit, R., & Nieswandt, M. (2000). Sources of students difficulties in learning Chemistry. *Educación Química*, *11*, 228–235. https://doi.org/10.22201/fq.18708404e.2000.2.66458

van Eck, R. (2009). A Guide to Integrating COTS Games into Your Classroom. In *Handbook of Research on Effective Electronic Gaming in Education* (pp. 179–199). IGI Global. https://doi.org/10.4018/978-1-59904-808-6.ch011

Vasić, D., Brajković, E., & Volaric, T. (2014). *Experimental evaluation of teaching recursion with HTML5 game*. 6th International Conference on e-Education, ICeE 2014, At Mostar, Bosnia and Herzegovina, Volume: No.1. https://doi.org/10.13140/2.1.1669.2481

Wassila, D., & Tahar, B. (2012). Using serious game to simplify algorithm learning. *International Conference on Education and E-Learning Innovations*, 1–5. https://doi.org/10.1109/ICEELI.2012.6360569

Whitton, N. (2007). *An investigation into the potential of collaborative computer game-based learning in Higher Education* [PhD Thesis]. Napier University.

Whitton, N. (2012). Games-Based Learning. In N. M. Seel (Ed.), *Encyclopedia of the Sciences of Learning* (pp. 1337–1340). Springer US. https://doi.org/10.1007/978-1-4419-1428-6_437

Whitton, N. (2014). *Digital Games and Learning: Research and Theory*. Routledge.

Wu, C. C., Dale, N. B., & Bethel, L. J. (1998). Conceptual Models and Cognitive Learning Styles in Teaching Recursion. *SIGCSE Bull.*, *30*(1), 292–296. https://doi.org/10.1145/274790.274315

Wu, W. H., Chiou, W. B., Kao, H. Y., Alex Hu, C. H., & Huang, S. H. (2012). Re-exploring game-assisted learning research: The perspective of learning theoretical bases. *Computers & Education*, *59*(4), 1153–1161. https://doi.org/10.1016/j.compedu.2012.05.003

Yee, N. (2006). Motivations for Play in Online Games. *Journal of CyberPsychology and Behavior*, *9*(6), 772–775. https://doi.org/10.1089/cpb.2006.9.772

Yin, R. K. (2018). *Case Study Research and Applications: Desing and Methods* (6th ed.). Sage Publications, Inc.

Zaharias, P., Chatzeparaskevaidou, I., & Karaoli, F. (2017). Learning Geography Through Serious Games: The Effects of 2-Dimensional and 3-Dimensional Games on Learning Effectiveness, Motivation to Learn and User Experience. *International Journal of Gaming and Computer-Mediated Simulations (IJGCMS)*, *9*(1), 28–44. https://doi.org/10.4018/IJGCMS.2017010102

Zhang, J., Atay, M., Caldwell, E. R., & Jones, E. J. (2015). Reinforcing student understanding of linked list operations in a game. *2015 IEEE Frontiers in Education Conference (FIE)*, 1–7. https://doi.org/10.1109/FIE.2015.7344132

Zhang, J., Atay, M., Smith, E., Caldwell, E. R., & Jones, E. J. (2014). Using a game-like module to reinforce student understanding of recursion. *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*, 1–7. https://doi.org/10.1109/FIE.2014.7044093

Zingaro, D., Taylor, C., Porter, L., Clancy, M., Lee, C., Liao, S. N., & Webb, K. C. (2018). Identifying Student Difficulties with Basic Data Structures. In *Proceedings of the 2018 ACM Conference on International Computing Education Research* (pp. 169–177). Association for Computing Machinery. https://doi-org.elib.tcd.ie/10.1145/3230977.3231005

Zydney, J. M. (2012). Scaffolding. In N. M. Seel (Ed.), *Encyclopedia of the Sciences of Learning* (pp. 2913–2916). Springer US. https://doi.org/10.1007/978-1-4419-1428-6_1103

Appendix 1: Game Design Document ─ First Iteration

1.1 Context

- Educational Environment: Undergraduate level (formal education)
- Target Audience
  - Academic background: Computer science, engineering, or natural science student students who are taking programming courses.
  - Academic level: Undergraduate.
  - Age: 18-22 years old
  - Gaming experience: Medium. Most of the students have played at least casual games (e.g., mobile games).
- Resources
  - Hardware: Laboratories with computers capable to run 3D video games.
  - Content expert: one content expert
  - Developer and designer: only one designer/developer
  - Software: Unity, Adobe Illustrator, Adobe Photoshop, and Adobe Audition.
  - Assets: Assets available in the Unity Asset Store.
- Location: The game can be used at classroom, laboratories, or home.

1.2 Learning Aspect

- Learning Theory: Kolb's Experiential Learning Theory (section 2.1.2)
- Content: Binary Search Trees
- Learning Objectives

Topic	Learning Objectives
BST, its components and structure.	LO1. List the components of a BST node. LO2. Recognize the basic elements of a BST node (parent, left child and right child, key/ID, and data). LO3. Distinguish the root node definition. LO4. Recall the BST property definition. LO5. Distinguish a BST from a BT. LO6. Apply the BST property through a BST.
BST search algorithm	LO7. Formulate the BST Search algorithm using the BST Property concept. LO8. Recall the BST Search algorithm. LO9. Distinguish the conditions of the BST algorithms pseudocode. LO10. Apply the BST Search algorithm.
In-order tree walk algorithm	LO11. Distinguish the order of steps of the In-Order algorithm. LO12. Recall the In-Order algorithm. LO13. Apply the In-Order algorithm.

- Pedagogical Approaches: Analogies
- Learning Activities: They are game challenges. They are explained in the game design section.
- Assessment: Conceptual knowledge test (see Appendix 4 and 5).

1.3 Game Design Document

Level	**Level 01. BST Nodes**
Gameplay	The player starts in the "root chamber" (Root Node) of the corporation's data system (data structure). The player must explore all the chambers (nodes) of the system. For this, the player must use the portals to teleport the character to the other chambers. Additionally, the player must copy the information stored in the computers of all the chambers.
Game level objectives	1. Introduce the following components of the game: a. The game controls b. The environment c. The story (context) of the game d. The player character (the robot) e. The NPC (Anonymous) f. The copy and teleport mechanics g. The hint system h. The mini map i. The files copied in the inventory 2. Explain Binary Trees and its nodes. 3. The level should be easy.
Challenges	1. Travel to all the rooms. 2. Copy the information from all the computers.
Binary Search Tree	[2|1|3]
Dialogues Scripts	**[Introductory Dialogue]** 1. **[NPC]** Greetings. My name is Anonymous. I will help you during your missions. I am going to tell you what I know about this place. 2. **[NPC]** You have entered the XC-01 Corporation's data system. This corporation is corrupt, and it is creating an imbalance in our society. We must hack its system and make its information public. 3. **[NPC]** Data systems are places where corporations hide their critical information. Corporations use different data structures configurations to build their systems. 4. **[NPC]** This system has chambers with Comparable Keys that distinguish each chamber from the others. Keys are like IDs that help to organize the data. So, probably you are inside a Symbol-Table implementation. 5. **[NPC]** Each chamber stores critical information called Associated Values. It is possible to access the information through the main computer located in the centre of each room. 6. **[NPC]** Your first mission is to extract the information stored in the main computer of each room in this data system. **[Final Dialogue]** 1. **[NPC]** Good job. We have all the information that we need. 2. **[NPC]** It seems that this data system is based on a data structure called Binary Search Tree. 3. **[NPC]** A Binary Search Tree is organized as a Binary Tree. A

	Binary Tree is a data structure of linked objects called nodes (the chambers). 4. **[NPC]** We can define a binary tree as either a null link or a node with a left link and a right link, and each link references to subtrees that are themselves binary search trees. 5. **[NPC]** Binary Search Tree nodes possess 2 links that point to other child nodes (the portals), a Comparable Key (the chamber ID) and an Associated Value (the data stored in the main computer). 6. **[NPC]** In Binary Search Trees, the Left Link points to the Left Child node. The Right Link points to the Right Child node. If there is no node to point, links will point to Null. That means that they point to an empty value (an empty chamber). 7. **[NPC]** In Binary Search Trees, the Root Node (the first chamber) is the only node that does not have another node pointing to it.

Level	**Level 02. BST structure and its property**
Gameplay	At the beginning of the level, the NPC (Anonymous) will explain the BST property. Then, Anonymous will tell the player the mission. After that, the player must use the BST property to travel through the BST avoiding the rooms that are secured by alarms. Additionally, the player must copy the data of the rooms in a specific order.
Game level objectives	1. Introduce the BST property. 2. The difficulty of the level should be medium.
Challenges	1. Copy the information from the rooms 2, 3, 4, and 5 in that order specific order. 2. Do not activate the alarms.
Binary Search Tree	[5|2|7|1|3|6|8|4]
Dialogues Scripts	**[Introductory Dialogue]** 1. **[NPC]** This is the second data system. Remember, these data systems are organized as Binary Search Trees. 2. **[NPC]** I am going to tell you how Binary Search Tree nodes are organized. First, all comparable keys must be unique. 3. **[NPC]** The Left Child Key is always LESS than the Parent Key. The Right Child Key is always GREATER than the Parent's Key. 4. **[NPC]** This is the main characteristic of the Binary Search Tree data structure. 5. **[NPC]** In this mission, we need to extract the information of Room 2, 3, 4, and 5. We need the data in that specific order. 6. **[NPC]** This system stores very important information, and the XC-01 Corporation has located alarms in all the rooms. 7. **[NPC]** Fortunately, I have disabled the alarms of the rooms that we need to hack. I only disabled the alarms of the rooms 2, 3, 4 and 5. 8. **[NPC]** You must not enter to the other rooms. If you do it, you will activate the alarms, and everything will be over for us. 9. **[NPC]** Remember how the BSTs are arranged. That will help you to move from one chamber to another. Good luck. **[Final Dialogue]** 1. [NPC] It seems that you are understanding how the Binary Search Trees are structured. Good job.

Level	**Level 03. BST Search algorithm. Part 1: Discover the algorithm**
Gameplay	The mission begins with a dialogue between Anonymous and the player. Anonymous explains that they require to extract the data of a specific room, but unfortunately, the security system turned on extra alarms. Then, he explains that he disabled the security of the rooms that lead to the room. Then, Anonymous ask the player to follow this optimal path. This optimal path is the same that the BST Search algorithm use. The player must follow this optimal path. Other rooms outside of the optimal path are secured by guard or alarms that will detect the presence of the player. If the player is caught, he or she will lose, and the level will restart.
Game Level objectives	1. Introduce the BST Search algorithm.
Challenges	1. Find the optimal path to the room. 2. Modify the data stored in room ID 4.
Binary Search Tree	[7|2|9|1|5|8|10|3|6|4]
Dialogues Scripts	**[Introductory Dialogue]** 1. **[NPC]** Now, we must extract the data of Chamber 4, but the corporation's guards increase their security measures. 2. **[NPC]** They activated the portals' alarms. Every time that you jump from one chamber to another, the alarms will be activated. 3. **[NPC]** The good news is that I managed to disable the alarms for 4 more jumps. That will be enough jumps to reach the Chamber 4. 4. **[NPC]** You should remember how Binary Search Trees are organized. Remember: 5. **[NPC]** The Left Child Key is always LESS than the Parent ID. The Right Child Key is always GREATER than the Parent Key. 6. **[NPC]** This information will help you to reduce the number of jumps. Good luck. **[Final Dialogue]** 1. **[NPC]** Now, you understand the importance of Keys. They help users to navigate through this maze of chambers. Good job.

Level	**Level 04. BST Search algorithm. Part 2. Explanation of the algorithm.**
Gameplay	Same gameplay as the previous level.
Game Level objectives	1. Explain the BST Search algorithm.
Challenges	1. Find the optimal path to the room. 2. Modify the data stored in room ID 6.
Binary Search Tree	[4|2|9|1|3|7|10|5|8|6]
Dialogues Scripts	**[Introductory Dialogue]** 1. **[NPC]** In this mission, you must extract the data of the Chamber 6. The portals' alarms are still enabled, and like our previous mission, I managed to disable the alarms for 4 jumps. 2. **[NPC]** We must minimize the number of jumps. This situation remains me the Binary Search Tree Get algorithm. 3. **[NPC]** The Get algorithm uses an efficient strategy to search for a Key and return the data stored in the node with that key.

	4. **[NPC]** The formal description of the algorithm looks like this:
	`Get (node, key)`
	`if (node == null)`
	`  return null`
	`if key < node.key`
	`  return Get (node.left, key)`
	`else if`
	`  return Get (node.right, key)`
	`else`
	`  return node.value`
	5. **[NPC]** Or translated to our purposes:
	`If the Chamber is empty:`
	`  Return empty (no data extraction)`
	`If KEY is Greater than the Chamber´s KEY`
	`  Go to the Left Chamber`
	`Else if Key Lesser than Chamber’s KEY`
	`  Go to the Right Chamber`
	`Else`
	`  Extract the information`
	6. **[NPC]** There are 2 possible outcomes. If you find the key, you return the associated value (extract the information). Otherwise, we have a search miss and return null (find an empty room).
	7. **[NPC]** It is the same that you did in the previous mission.
	[Final Dialogue]
	1. **[NPC]** Well done!

Level	**Level 05. BST Search algorithm. Part 3: Return Null.**
Gameplay	Same gameplay as the previous level.
Game Level objectives	1. Explain the BST Search algorithm.
Challenges	1. Find out that there is no data (return null).
Binary Search Tree	[5|2|9|1|4|7|6|8]

Dialogues Scripts	**[Introductory Dialogue]**
	1. **[NPC]** In this mission, you must extract the data of the Chamber 3. The portals’ alarms are enabled, and you can jump through the portals 3 times.
	2. **[NPC]** We must minimize the number of jumps. So, you should remember the Get algorithm.
	3. **[NPC]** Remember:
	`If the Chamber is empty:`
	`  Return empty (no data extraction)`
	`If KEY is Greater than the Chamber´s KEY`
	`  Go to the Left Chamber`
	`Else if Key Lesser than Chamber’s KEY`
	`  Go to the Right Chamber`
	`Else`
	`  Extract the information`
	4. **[NPC]** Also, you must remember that after executing the algorithm, you can obtain 2 possible outcomes. Good luck.
	[Final Dialogue]

	1. **[NPC]** This is not what we were expecting. It seems that there is no chamber with the key number 3. The chamber is empty (null). 2. **[NPC]** Probably they destroyed the chamber before we could reach it.

Level	**Level 06. In-Order tree walk algorithm (Part 1)**
Gameplay	At the beginning of the level, Anonymous is going to explain the In-Order tree walk algorithm and the mission. Then, the player must extract all the data of the BST following the In-Order tree walk algorithm before all the alarms restart. If the player does not follow the order of the algorithm, the player will not have enough time to complete the mission. During the mission, Anonymous is going to guide the player.
Game Level objectives	1. Introduce the In-Order tree walk algorithm. 2. Introduce the NPC hint system. 3. The level should be easy.
Challenges	1. Copy all data following the In-Order tree walk algorithm 2. Do not activate the alarms
Binary Search Tree	[4|2|6|1|3|5|8|7]

Dialogues Scripts	**[Introductory Dialogue]**
	1. **[NPC]** We need to extract the data stored in each room of this data system.
	2. **[NPC]** The security system noticed our presence and activated shutting down protocol. They are disabling all computers. We need to extract the data before they shut down all the computers.
	3. **[NPC]** The shutting down protocol follow an In-Order tree walk pattern (or algorithm). To extract the data on time, you need to follow the same pattern (or algorithm).
	4. **[NPC]** In-Order tree walk pattern is a recursive algorithm. This means that it calls itself. Formally, it looks like this:
	5. **[NPC]** In-Order-walk (Room)
	`If the room is not empty`
	`In-Order-walk (Left Room)`
	`Copy the data`
	`In-Order-walk (Right Room)`
	6. **[NPC]** It looks complicated, but it is not. I will explain it now:
	1. If there is a left portal: travel through it and then execute instruction (1). If not, go to instruction (2).
	2. Copy the data. Go to the instruction (3).
	3. If there are a right portal, you should travel through it, and the return to instruction (1). If not, go to instruction (4).
	4. Return to the previous room. If you were in a Left room, go to instruction (2). Else, if you were in a right room, go to instruction (4).
	7. **[NPC]** This algorithm will execute the actions (copy) in an ascendant order.
	8. **[NPC]** To extract the data on time, you must follow this pattern (algorithm). Chamber 1's computer will be shut down in 60 seconds. And after that one, all the other alarm will be deactivated one by one every 35 seconds.
	9. **[NPC]** Do not worry, I will help you during the mission, but you must keep moving.

	10. **[NPC]** Your mission is to copy the data of all the computers. **[Final Dialogue]** 1. **[NPC]** Good job. Recursive algorithms are not easy.

Appendix 2: Game Design Document ─ Second Iteration

2.1 Context

A full description of the context is available in section 5.2.1.

- Educational Environment: Undergraduate level (formal education)
- Target Audience
    - Academic background: Computer science, engineering, or natural science student students who are taking programming courses.
    - Academic level: Undergraduate.
    - Age: 18-22 years old
    - Gaming experience: Low. Most of the students do not have experience with video games (e.g., mobile games).
- Resources
    - Hardware: Laboratories with computers capable to run 3D video games.
    - Content expert: one content expert
    - Developer and designer: only one designer/developer
    - Software: Unity, Adobe Illustrator, Adobe Photoshop, and Adobe Audition.
    - Assets: Assets available in the Unity Asset Store.
- Location: The game can be used at classroom, laboratories, or home.

2.2 Learning Aspect

- Learning Theory: Kolb's Experiential Learning Theory (section 2.1.2)
- Content: Binary Search Trees
- Learning Objectives (section 5.2.2)
- Pedagogical Approaches: Analogies (section 2.1.4)
- Learning Activities: They are game challenges. They are explained in the game design section. (section 5.2.3)
- Assessment: Conceptual knowledge test (see Appendix 4 and 5).

2.3 Game Design Document

2.3.1 Level 01 ─ Introduction

2.3.1.1 Learning Objectives

This level does not have learning objectives. This level introduces the game controls and the game story.

2.3.1.2 Learning Activities

This level does not have learning activities. This level introduces the game controls and the game story.

2.3.1.3 Description

The level introduces to the player the game mechanics: game controls, activate interactive elements, extract data, and portal usage.

2.3.1.4 BST Structure

2 | 3

2.3.1.5 Script

- Dialogue 01 (Introduction and game controls)
    1. [NPC] Greetings. My name is Anonymous. I am your creator.
    2. [NPC] Your purpose is to help me to extract information from corrupt corporations that harm the balance of society.
    3. [NPC] I have just finished the installation of your software, but we must calibrate the system that controls your movement and stability. So, you must practice your basic movements.
    4. [NPC] I have created a replica of the environment that we are going to face. Here, you can train your movements.
    5. [NPC] I will give you small tasks that you should complete, and slowly, your system will begin to calibrate itself.
    6. [NPC] Your first task is to reach the central room of this node.
    7. [IMG] Image of the game controls
- Dialogue 02 (Exploration – Left side)
    1. [NPC] It seems that you can move properly, but you must practice even more.
    2. [NPC] Your next task is to go to the left room of the node, and then, you should go down to the lower level of that room. I hope your robotic body resist the impact.
- Dialogue 03 (Activate the elevators and more exploration)
    1. [NPC] Great! Your body resisted the impact. I thought that I will need to repair your after that jump.

2. [NPC] In front of you, there is an empty room. That means that the portal of this room is disabled. Soon, we will see one that is working.
3. [NPC] Now, you must learn to use the elevators. To activate them, you should step over the platform and press the start button (press the E key).
4. [NPC] Your next task is to go to the right room and descend to the lower level.

- Dialogue 04
  1. [NPC] Your movements are improving.
  2. [NPC] In front of you, there is a working portal. Portals teleport you to other nodes. All nodes look the same, but do not get confused; once you pass through a portal, you are in another node.
  3. [NPC] Your next task is to traverse the portal.
- Dialogue 05
  1. [NPC] As you can see, once you traverse the portal, there is no portal to return to the previous node, but don’t worry. I can teleport you to the previous node, but you should request to return (press the R key).
  2. [NPC] Your next task is to return to the previous node.
- Dialogue 06
  1. [NPC] When you request to return to the previous node, I will locate you in the entrances of the node.
  2. [NPC] Your final task is to extract the information of this node. To achieve this task, you should go to main computer located in the main room of the node.
  3. [NPC] When you get close to the computer, you can extract the data.
- Dialogue 07
  1. [NPC] You have successfully extracted the data.
  2. [NPC] On the top section of your screen, you can see the Key (ID) of the node where we extracted the data.
  3. [NPC] You must not mistake the Keys (IDs) for the data contained in the node. They are different.
  4. [NPC] Good job. We have finished the basic training. Your systems have been calibrated successfully.
  5. [NPC] The following tasks will help us to verify whether your memory system stores the correct information related to some data structures. Later, we will calibrate your defence systems.

2.3.1.6 Level Objectives

1. Go to the central room of the node

2. Go to the lower level of the left room
3. Go to the lower level of the right toom
4. Cross the right portal
5. Return to the previous node
6. Extract the data of the node 2

2.3.1.7 BT Concepts

This level does not cover BST concepts.

2.3.2 Level 02 ─ Binary Tree Node

2.3.2.1 Learning Objectives

1. Understand the concept of Binary Tree. (Bloom's Taxonomy - Understand)
2. Define the basic elements that compose a Binary Tree (BT) node. (Bloom's Taxonomy – Understand)

2.3.2.2 Learning Activities

1. Read and listen the BT definition.
2. Read and listen the BT node definition and its basic components.
3. Read and listen the link definition.
4. Relate the portals with the concept of references/pointers.
5. Relate the chambers of the game environment with the BT node structure and components.

2.3.2.3 Description

The level 2 introduces the BT concept and explains the basic elements that compose a BT node (left and right link).

2.3.2.4 BST Structure

3 | 1 | 2

2.3.2.5 Script

- Dialogue 01 – Introduction, story and the BT node
  1. [NPC] Previously, I told you that I created you to infiltrate in the Data Systems of corrupt corporations.

2. [NPC] Data Systems are places where corporations hide and protect their critical information. Those systems arrange their structures the same way that some data structures that facilitate the search operation.
3. [NPC] This place where we are right now is a recreation of a Data System arranged as a data structure called Binary Tree. We will use this place to verify your memory system.
4. [NPC] Binary Trees are data structures made up of linked objects called nodes. In the case of this Data System, the whole set of rooms is equivalent to a node.
5. [NPC] To link the nodes, data structures use references or pointers. A reference can store the address of a node, or it can store a null value; in other words, it stores an empty value.
6. [NPC] Also, Binary Trees nodes possess two links to other nodes known as the left and right link.
7. [NPC] **BT image**. This image is a representation of a Binary Tree with two nodes.
8. [NPC] **BT image**. The square and the two lines represent a Binary Tree node.
9. [NPC] **BT image**. The lines represent the left link…
10. [NPC] **BT image**. … and the right link.
11. [NPC] **BT image**. The links can point to another node…
12. [NPC] **BT image**. … or they can be null.
13. [NPC] In this training, you are going to recognize this Data System. This will help you to understand the Binary Tree structure and its node.
14. [NPC] Your first task is to find the left link of this node.

- Dialogue 02
  1. [NPC] You have found the left link. In this Data System, links are presented as portals. Portals have the address to access other nodes, and when you traverse them, they will teleport you to those nodes.
  2. [NPC] Your next task is to traverse the portal.
- Dialogue 03
  1. [NPC] Now, you are in a different node.
  2. [NPC] Your next task is to find the right link.
- Dialogue 04
  1. [NPC] You have found the right link.
  2. [NPC] As you can see, the portal is disabled, and it takes you to an empty room. In this case, this portal does not point to another node. It stores a null value.
  3. [NPC] We have finished this validation.

2.3.2.6 Level Objectives

1. Find the left link
2. Traverse the left link (portal)
3. Find the right link

2.3.2.7 BST Concepts

1. A Binary Tree is a data structure made up of nodes.
2. Binary Tree nodes possess two links called the left and right link.
3. Links can point the address of other nodes (they store the address), or they can be null (store an empty value).

2.3.3 Level 03 ─ BT Structure

2.3.3.1 Learning Objectives

1. Identify Binary Trees.
2. Identify the basic elements of a Binary Tree.

2.3.3.2 Learning Activities

1. Read and listen the definition of the left and right child.
2. Read and listen the definition of the parent node.
3. Read and listen the definition of a sub-tree.
4. Relate the definition of the Binary Tree and its elements with the game environment.
5. Identify the root node represented by the game environment.

2.3.3.3 Description

Level 3 introduces the basic elements that compound the structure of a Binary Tree.

2.3.3.4 BTS Structure

2 | 1 | 4 | 3 | 5

2.3.3.5 Script

- Dialogue 01
  1. [NPC] Now, we are going to verify the information related to the basic concepts of the Binary Tree.
  2. [NPC] As I told you, a Binary Tree is a data structure made of linked objects called nodes. Binary Tree’s nodes possess two links called the left and right link.

3. [NPC] The left link points to the left child node, and the right link points to the right child node. Add an image.
4. [NPC] Every node is pointed by just one other node, which is called its parent. Add an image.
5. [NPC] The only node that does not have a parent is the root node. Add an image.
6. [NPC] Finally, we can say that a Binary Tree is null link or a node with a left link and a right link, each of them references to subtrees that are themselves binary trees.
7. [NPC] Image of a Binary Tree. This image represents a null Binary Tree and a Binary Tree with a node that is parent of two other nodes. Both nodes also are subtrees.
8. [NPC] Your first task is to find the right child of this node.

- Dialogue 02
  1. [NPC] You have found the right child.
  2. [NPC] Your next task is to find the left child of this node.
- Dialogue 03
  1. [NPC] You are in the left child of the previous node.
  2. [NPC] Your final task is to find the root node of this Binary Tree.
- Dialogue 04
  1. [NPC] We have finished to verify all the information related to Binary Trees. It seems that your memory system has installed this information successfully.

2.3.3.6 Level objectives

1. Find the right child.
2. Find the left child.
3. Find the root node.

2.3.3.7 BST Concepts

1. A Binary Tree is a data structure made of linked objects called nodes.
2. Binary Tree’s nodes possess two links: the left and right link.
3. The left link points to the left child node, and the right child node.
4. Every node is pointed by just one other node, which is called its parent.
5. The only node that does not have a parent is the root node.

2.3.4 Level 04 ─ The BST data structure and its property

2.3.4.1 Learning Objectives

1. Explain the Binary Search Tree (BST) property.

2. Identify the Binary Search Tree.
3. Order the comparable keys of a BST using the BST property.
4. Determine whether the Binary Search Tree property is unfulfilled.

2.3.4.2 Learning Activities

1. Read and listen the definition of the basic components of a Binary Search Tree node.
2. Read and listen the definition of the Binary Search Tree property.
3. Apply the Binary Search property to search for specific nodes.

2.3.4.3 Description

Level 4 explains the elements of the BST node (the comparable key and the associated value) and the BST property. Also, the level challenge requires that the player apply the BST property to solve the puzzles.

2.3.4.4 BST Structure

6 | 2 | 8 | 1 | 4 | 7 | 9 | 3 | 5

2.3.4.5 Script

- Dialogue 01
  1. [NPC] Now, we are going to verify the data stored on your memory system related to Binary Search Trees.
  2. [NPC] The Binary Search Trees are based on the structure of the Binary Tree.
  3. [NPC] Like the Binary Tree, the Binary Search Tree has nodes with two links (the left and right link) that point to other nodes (the left child and the right child).
  4. [NPC] Additionally, the Binary Search Tree node has two more elements: the comparable key and the associated value.
  5. [NPC] The comparable key is used to organize the Binary Search Tree structure. It must store an ordinal value, such as int numbers or alphabet letters. Also, the comparable key of each node must be unique.
  6. [NPC] Data Systems nodes have comparable key. You can find them in the bottom-left corner of your vision system. **Add an image** (arrow).
  7. [NPC] The associated value is the data stored in the node. In Data Systems, the data is stored in the computer located in the centre of the node.
  8. [NPC] As I told you previously, the comparable keys are used to organize the nodes of the Binary Search Tree, and they must satisfy the following property:

9. [NPC] The key in any node is larger than the keys in all nodes in that node’s left subtree and smaller than the keys in all nodes in that node’s right subtree.
10. [NPC] **Image**. This image represents a Binary Search Tree.
11. [NPC] **Image**. The keys are located inside the node, and the associated values next to the node.
12. [NPC] **Image**. All the keys on the left side of node 4 are smaller than that node’s key.
13. [NPC] **Image**. All the keys on the right side of the node 4 are greater than that node’s key
14. [NPC] **Image**. This Binary Tree satisfies the Binary Search Tree property.
15. [NPC] This Data System is based on a Binary Search Tree. All the information that we verified will help you to accomplish the next tasks.
16. [NPC] Your first task is to find node 5 and extract the associated value stored in that node.

- Dialogue 02
  1. [NPC] Now, you must return to the root node.
- Dialogue 03
  1. [NPC] Your next task is to find the node 7 and extract the information stored in that node.
- Dialogue 04
  1. [NPC] We have verified the data stored on your memory system. Our next task is to calibrate your defence system.

2.3.4.6 Level Objectives

1. Find the node 5 and extract the information stored in that node.
2. Return to the root node.
3. Find the node 7 and extract the information stored in that node.

2.3.4.7 BST Concepts

1. The comparable key is used to organize the Binary Search Tree structure, and it must store an ordinal value.
2. Comparable keys must be unique.
3. The associated value is the information stored in the node.
4. The key in any node is larger than the keys in all nodes in that node’s left subtree and smaller than the keys in all nodes in the node’s right subtree.

Appendix 3: Pilot Experiment

DS-Hacker 3D was developed using an iterative design methodology, which requires testing the prototypes. Game testing provides feedback regarding the game design and game development, such as coding bugs, usability and user interface issues, and balancing the levels' difficulty. Additionally, game testing of games for learning provides feedback regarding the implementation of the learning aspects, such as efficacy of the pedagogical approaches. Testing allows designers and developers to fix the game and to provide a better gaming experience to players.

The first version of *DS-Hacker 3D* was tested by two professors of the data structure and algorithms course at Trinity College Dublin. This evaluation was informal. However, the feedback given by the professors allowed us to recognise user interface issues and data structure representation problems that could mislead students. After fixing all the issues, the second version of *DS-Hacker 3D* was created.

To evaluate the efficacy of the second version of *DS-Hacker 3D*, a pilot experiment was performed. The pilot experiment evaluated the efficacy of the game and compared it with other "traditional" methodologies. The experiment took place during the 2019-2020 Summer Term in University of Costa Rica. This pilot experiment aims to study the effect of DS-Hacker on the participants' learning gains and motivation. Also, it aims to verify the game usability, user experience, usefulness, and instructional aspects. Specifically, it aimed to answer the following research questions:

RQ1. Do students learning with *DS-Hacker 3D* obtain higher scores than students working with "traditional" learning approaches?
RQ2. Do university students perceive that they are learning while playing the game? Were the concepts taught in a clear way?
RQ3. Were the game and its components working correctly? (e.g., usability, user experience, and clear instructions)
RQ4. Do university students feel motivated to play the game?

3.1 Materials and Methods

In this section, we explain the methods and materials used to perform the experiment. We describe in detail the participants, materials used for the control activity, the knowledge test, the demographic survey, and the game experience survey. Furthermore, we explain the experiment design and statistical software and packages used for the analysis.

3.1.1 Participants

Thirty-two students participated in the pilot experiment. However, we excluded five participants from the analysis because they did not complete all the evaluation. Therefore, we performed the data analysis using data from twenty-seven participants only. The first group (G1) had thirteen participants, and the second group (G2) had fourteen. Regarding the participants' academic background, eleven students were from the Computer Science School; ten students were from the Industrial Engineering School; and five students from the Electrical Engineering School and Mathematics School.

3.1.2 Materials and Data Collection Tools

The second version of *DS-Hacker 3D* was used as the experimental activity. Concerning the pedagogical elements, the second version aims to teach BST data structure, its property, and search algorithm. Concerning the game elements, the game challenges are difficult because the game provide minimal amount of information. Players should discover the solutions of each level applying the BST concepts. In total, the second version has six level, and it does not have an introductory level to teach novice player the controls.

The control activity consisted of reading a written summary of the BST concepts and three video tutorials. The summary was a Spanish translation of the book Algorithms (Sedgewick & Wayne, 2014). The first video tutorial[2] was about BST structure and characteristics, and the second[3] was about the search and insert operations (the insert operation was not evaluated). The third video[4] tutorial was a general summary about the BST basic concepts and operations.

The pre-test, mid-test and post-tests were designed to assess the learning gains. The tests have 23 questions and cover the first four levels of cognition (remember, understand, apply, and analyse) of the revised version of the Bloom's taxonomy (Anderson & Krathwohl, 2001). The questions were multiple-choice with only one correct answer. Furthermore, the questions verify factual, conceptual, and procedural knowledge. Additionally, a demographic survey and a qualitative survey were performed. The qualitative survey evaluated the game usability, user experience, usefulness, and instructional aspects. All surveys and tests were developed using Google Forms.

3.1.3 Experiment Design

[2] https://youtu.be/Bh61AvHAf90

[3] https://youtu.be/DVKDQcJOqy8

[4] https://youtu.be/mTMrszfrNtI

The pilot experiment follows a “switching replications” experimental design. Participants were randomly divided into two groups, G1 and G2. Then, participants were asked to complete two activities (the treatment and the control activity) and answer three tests (pre-test, mid-test, and post-test). These activities and tests were organized as follows. First, all participants answered the pre-test. Then, G1 performed the treatment activity, and G2 performed the control activity. Next, all participants answered the mid-test. Then, G1 and G2 switched activities; G1 performed the control activity and G2 the treatment activity. Finally, all participants take the post-test. According to Eagle and Barnes (2009), switching replications design may decrease social threats to validity, since it allows all participants equal access to the treatment activity. However, the learning effect due to the overexposure to the tests may lead to a testing threat.

In addition to the formal evaluation, the professor who administered the evaluation performed informal observations regarding playing and learning difficulties. Additionally, she performed brief informal interviews asking to the participants about their playing and learning experience.

3.1.4 Data Analysis Methods

Data analysis was performed using R language version 4.01 (See Things Now), R Studio, and Microsoft Excel. Initial data cleaning was implemented using Microsoft Excel. All inconsistent data, such as repeated or incomplete rows, were removed using the Excel’s filters. Plotting, descriptive, and inferential statistics analysis were implemented in R Studio.

During the initial exploration of data collected, boxplots and descriptive statistics summaries were used to visualize and analyse data patterns. Data visualization was implemented using R Package *graphics* and *ggplot2*. The descriptive analysis was implemented using the R Package *psych*.

We verified that data were normally distributed and homoscedastic. Concerning the normal distribution assumption, it was verified using the skew and kurtosis value, histograms, Normal Probability-Probability (PP) plots, and Shapiro-Wilk tests. The skew and kurtosis value were obtained using the R Package *psych*. The Shapiro-Wilk test was performed using the R Package *nortest*, and the histograms and Normal PP Plots were drawn using the R Package *ggplot2* and *qqplotr* . The homoscedastic was verified using the Levene’s tests available in the R Package *car*.

Data collected does not meet the parametric assumptions; it is ordinal data. Consequently, the Mann-Whitney U test was used for the inferential analysis. The R Package *rstatix* was used to perform the inferential statistics and the Wilcoxon effect size.

3.2 Results

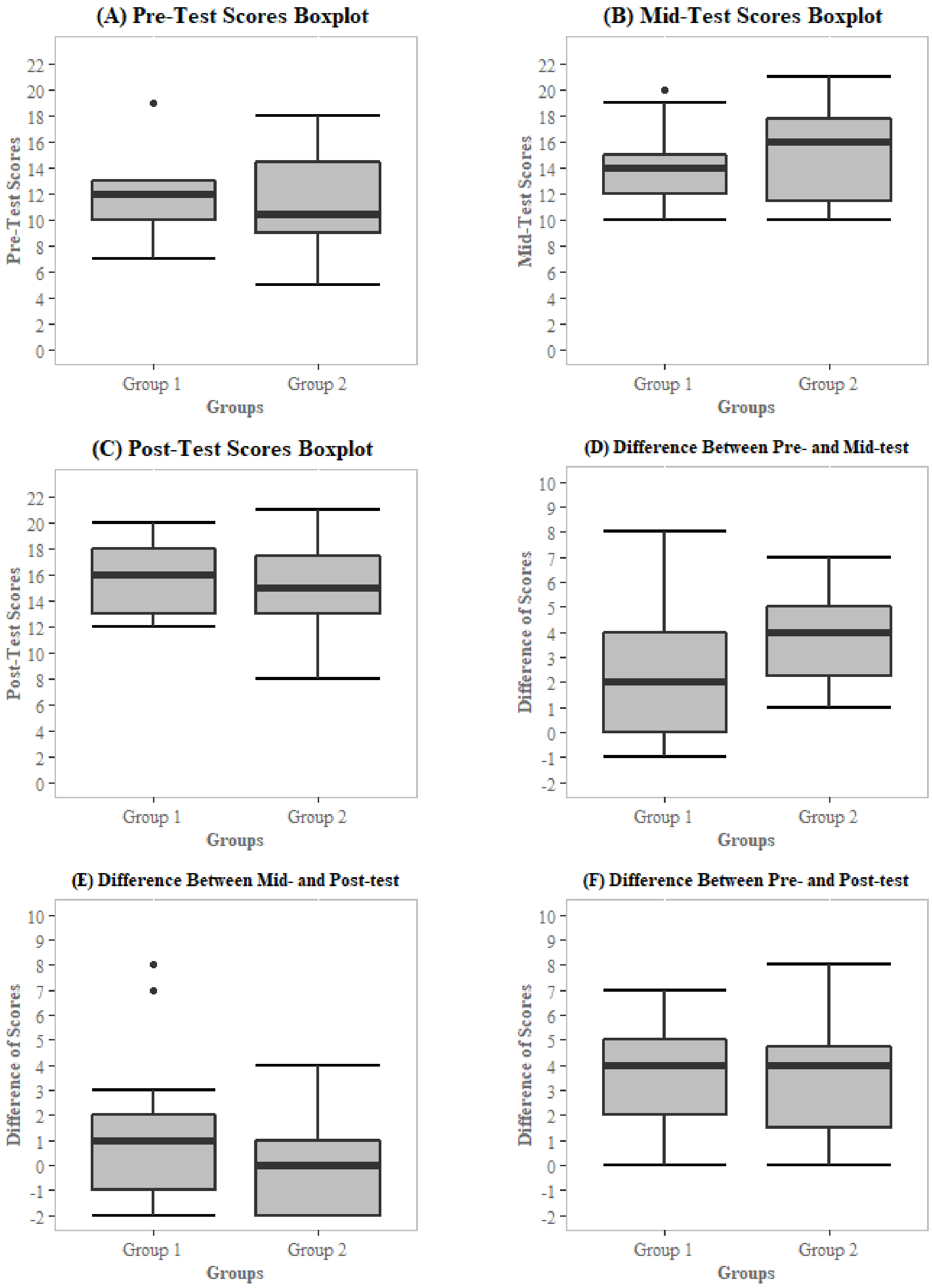
3.2.1 Descriptive Analysis

In the first test (pre-test), on average, Group 1's participants obtained 12.23 (*SD* = 3.39) points and Group 2's participants 11.29 (*SD* = 3.83). After the first round of activities, the Group 2's participants performed better in the mid-test than the Group 1's participants. On average, Group 1's participants (who played the game) increased 2 points (*M* = 14.23, *SD* = 2.95). Meanwhile, Group 2's participants (who watched the video tutorials) increased 3,79 points (*M* = 15.07, *SD* = 3.45). After switching and completing the second round of activities, the G1's participants performed better in the post-test than the G2's participants. Group 1's participants increased 1,69 points (*M* = 15.92, *SD* = 2.96); Group 2's participants decrease -0,36 points (*M* = 14.719, *SD* = 3.45). The complete list of results of the descriptive analysis are presented in Appendix Table 1.

Regarding the learning gains (difference of scores between tests), after the first round of activities, on average, Group 1's scores increased 2 (*SD* = 2.74), and Group 2 increased 3.79 (*SD* = 2.01). After the second round of activities, on average, scores of Group 1 increased 1.69 (*SD* = 2.98), and scores of Group 2 decreased -0.36 (*SD* = 1.91). The complete list of results of the descriptive analysis of the differences of scores between tests are presented in Appendix Table 2. Additionally, Appendix Figure 1 shows the boxplots of the test and the differences of scores between tests (learning gains after experiencing the learning activities).

Appendix Table 1: Descriptive statistics of the pre-test, mid-test, and post-test scores of the Group 1 and Group 2.

	Pre-Test Group 1	Mid-Test Group 1	Post-Test Group 1	Pre-Test Group 2	Pre-Test Group 2	Pre-Test Group 2
Number of values	13	13	13	14	14	14
Min	7	10	12	5	10	8
1st Quartile	10	12	13	9	11.5	13
Median	12	14	16	10.5	16	15
Mean	12.23	14.23	15.92	11.29	15.07	14.71
3rd Quartile	13	15	18	14.5	17.75	17.5
Max	19	20	20	18	21	21
Interquartile Range (IQR)	3	3	5	5.5	6.25	4.5
Variance	11.53	8.69	7.24	14.68	11.92	13.91
Standard Deviation	3.39	2.95	2.69	3.83	3.45	3.73



Appendix Figure 1: Boxplots of the (A) pre-test scores, (B) mid-test scores, (C) post-test scores, (D) difference between the pre- and mid-test, (E) difference between the mid- and post-test and (F) difference between the pre- and post-test.

Appendix Table 2: Descriptive statistics of the difference of scores between the pre- and mid-test, mid- and post-test, and pre- and post-test of the Group 1 and Group 2.

	Difference Mid - Pre Group 1	Difference Post - Mid Group 1	Difference Post - Pre Group 1	Difference Mid - Pre Group 2	Difference Post - Mid Group 2	Difference Post - Pre Group 2
Number of values	13	13	13	14	14	14
Min	-1	-2	0	1	-3	0
1st Quartile	0	-1	2	2.25	-2	1.5
Median	2	1	4	4	-0.5	4
Mean	2	1.69	3.69	3.79	-0.36	3.43
3rd Quartile	4	2	5	5	0.75	4.75
Max	8	8	7	7	4	8
Interquartile Range (IQR)	4	3	3	2.75	2.75	3.25
Variance	7.5	8.9	5.4	4.03	3.63	5.34
Standard Deviation	2.74	2.98	2.32	2.01	1.91	2.31

3.2.2 Inferential Analysis

Regarding the within groups analysis, the paired Mann-Whitney U test performed on the differences between the pre-test and mid-test scores of Group 1 suggests that the differences are statistically significant, W = 58.5, p-value = 0.03 (p-value < 0.05). The Wilcox effect size is 0.38, suggesting a moderate effect. The paired Mann-Whitney U test performed on the differences between the pre-test and mid-test scores of Group 2 show that the differences are statistically significant, W = 105, p-value = 0.001 (p-value < 0.05). The Wilcox effect size is 0.0.48, suggesting a moderate effect. Table 7.8 summarize the results of the inferential analysis within groups.

The paired Mann-Whitney U test performed on the differences between the mid-test and post-test scores of Group 1 shows that the differences are not statistically significant, W = 19, p-value = 0.07 (p-value > 0.05). The Wilcox effect size is 0.28, suggesting a low effect. Similarly, the paired Mann-Whitney U test performed on the differences between the mid-test and post-test scores of Group 2 shows that the differences are not statistically significant, W = 43, p-value = 0.39 (p-value > 0.05). The Wilcox effect size is 0.05, suggesting a low effect. Appendix Table 3 summarizes the results of the inferential analysis within groups.

Appendix Table 3: Inferential analysis within groups.

	Size	Statistic	p-value	Effect size
Group 1: Pre- and Mid-test	13	58.5	0.03	0.38
Group 1: Mid- and Post-test	13	19	0.07	0.28
Group 2: Pre- and Mid-test	14	105	0.001	0.48
Group 2: Mid- and Post-test	14	43	0.39	0.05

Concerning the between groups analysis, the Mann-Whitney U test performed on the pre-test scores suggests that the differences between Group 1 and Group 2 are not statistically significant, *W* = 100, *p-value* = 0.28. Similarly, the Mann-Whitney U tests performed on the mid-test and post-test scores suggest that the differences between the groups are not statistically significant, *W* = 76.5, *p-value* = 0.49 and *W* = 107, *p-value* = 0.45, respectively.

Regarding the learning gains, the Mann-Whitney U test performed on the learning gains after the first learning activity (mid-test minus pre-test) suggests that the differences between Group 1 and Group 2 are not statistically significant, *W* = 51, *p-value* = 0.05. The p-value is over the threshold, suggesting that it is very close to be statistically significant. After the second round of learning activities, the Mann-Whitney U test performed on the learning gains (post-test minus mid-test) suggests that the differences between Group 1 and Group 2 are statistically significant, *W* = 132, *p-value* = 0.04, suggesting that Group 1 learned more using the "traditional" methods than Group 2 (using the game).

Appendix Table 4: Inferential analysis between groups.

	Group 1 size	Group 2 size	Statistic	p-value	Effect size
Pre-test	13	14	110	0.38	0.17
Mid-test	13	14	76.5	0.49	0.14
Post-test	13	14	107	0.45	0.15
Diff pre- and mid-test	13	14	51	0.05	0.38
Diff mid- and post-test	13	14	132	0.04	0.39

3.3 Qualitative Survey

The qualitative survey has 19 four-point Likert-scale questions divided into three categories. The first category (Q1-Q6) assesses the participant's perception about learning and the means used by the game (environment, story, and challenges) to convey the BST concepts. The second category (Q7-Q15) assesses the usability. The third category (Q16-Q19) assesses the enjoyability. We only present the results of the 27 participants who completed all the tests. Table 4 presents the percentage of the positive answers ("Strongly agree" and "Moderately agree") of the qualitative survey.

Appendix Table 5: Qualitative Survey.

Questions	Agree %	Questions	Agree %
Q1. The game help me to understand BST structures.	77.78	Q11. The game tutorial was useful and clear.	81.48
Q2. The game help me to understand the nodes' structure.	81.48	Q12. The voice and way of talking of the NPC were clear.	81.48
Q3. The game help me to understand the search algorithm.	59.26	Q13. Game missions were clear.	81.48

Q4. I could relate concepts presented in the game story with the BST concepts.	88.89	Q14. The game GUI was easy to understand and intuitive.	85.19
Q5. I could relate the game environment with the BST structure.	85.19	Q15. The game menu has useful options.	81.48
Q6. The game allows me to practice the previously learned BST concepts.	85.19	Q16. I enjoyed playing DS-Hacker	70.37
Q7. The game was easy to learn.	85.19	Q17. I like the way that BST concepts were presented during the game.	81.48
Q8. The game controls were easy to learn.	66.67	Q18. I think that video games increase my motivation towards computer science topics.	85.19
Q9. The game controls respond smoothly.	48.15	Q19. I would like more serious games to be used to teach data structures.	81.48
Q10. The map was easy to understand.	77.78		

3.4 Discussion

Concerning RQ1, after the first learning experience, results show that Group 1 (playing *DS-Hacker*) obtained lower learning gains than Group 2. Also, after the second learning experiences, results show that Group 1 (using “traditional” learning approaches) obtained a higher learning gains that Group 2. This finding suggests that *DS-Hacker* is less effective than the traditional methods selected. There are two reasons that may cause these results. First, the “traditional” learning experiences were very comprehensive, consisting of three video tutorials and one written summary of the topic. Second, the game had issues regarding the way it convey the contents, performance in old computers, and level design. However, results show that students improved their scores in the knowledge test after playing the game. This shows that DS-Hacker is teaching the contents, but it needs to be improved.

Regarding RQ2, in general, results show that most of the participants perceive that they learned while playing the game. However, results also show that a considerable number of participants did not understand the search algorithm. There are two reasons that may cause these results. First, the presentation of the content was based on text explanation and pseudocode. This may confuse students who are not advance coders. Second, the challenges of the levels that covered this topic were difficult for the participants. Some of these levels also employed time pressure as part of the challenge, which is not a good approach while learning new topics. Overall, participants believe that games are effective learning method. This is a good result because they did not reject it as learning tool.

Concerning RQ3, most of the game components were working correctly. However, based on the qualitative survey results and the informal observations made by the researcher who administered the evaluation, we found many issues related to the way that the learning content was conveyed, performance issues, and gameplay issues (e.g., challenges difficulty and how instructions were presented). Regarding the learning issues, we noticed that participants needed an introduction to binary tree data structure before learning binary search tree concepts. Also, we noticed that the participants felt overwhelmed by the number of learning objectives and the pace they were presented. Concerning the performance issues, we noticed that computers on laboratories are outdated computers or do not have enough computation resources to run complex 3D games. This caused that the frames per second were low and that the controls did not respond quickly. Also, we noticed that many of the participants were not used to computer game controls (keyboard and mouse). This caused navigation difficulties while playing the game. Finally, concerning the gameplay issues, we noticed that the user interface was not clear. For example, participants did not understand what to do after receiving the level instructions, or they did not understand the meaning of the world map. Also, some participants could not solve the final challenges because the time pressure was very difficult. This also caused learning issues.

Concerning RQ4, most of the participants believe that DS-Hacker 3D was a fun game and that they would like more game for learning. There is an acceptance of video games in learning environments by young adults. This finding is important because other researchers (e.g., Whitton, 2014) have found that adults do not always have a positive attitude towards video games in learning environments.

3.5 Conclusion

In this Appendix, we presented the pilot experiment of the second version of DS-Hacker 3D. The chapter presented the methodology, results, and a brief discussion of the results. Our key findings are:

- *DS-Hacker 3D* is an effective learning tool. However, it needs to be improved to be at the same level of other “traditional” methods.
- Students perceive that they learn while playing *DS-Hacker 3D*.
- Students are motivated to play *DS-Hacker 3D* and video games for learning.

Concerning the game issues found, we conclude that a new version of the game is needed. This version should include the following features:

- New contents. The game should focus on binary trees concepts.
- Reduced list of learning objectives. The game should reduce the number learning objectives. To ensure learning, it is important to focus on a reduced amount of contentment.
- An introductory level. Many potential players are not familiar with PC games or the action-adventure genre. An introductory level aims to facilitate the acquisition of the gaming skills necessary to play DS-Hacker 3D.
- New user interface. It is necessary to provide better cues to the player while they are performing the challenges.
- Decrease the difficulty of challenges. Some challenges use time pressure, which increase the cognitive load and decrease learning gains. This aspect should be removed from all levels of the prototype.
- Graphical quality options. The game should be compiled to support at least two graphical quality options (e.g., high, and low quality). This will allow less powerful computers to run the game without framing issues.

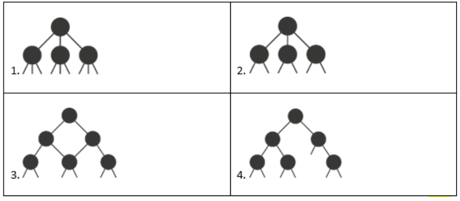
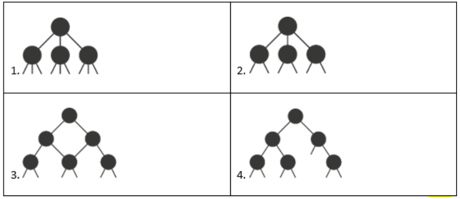
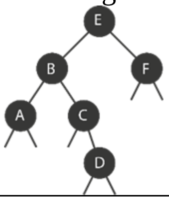
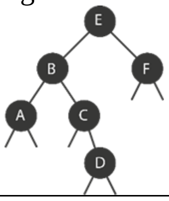
Appendix 4: BST Assessment Tool

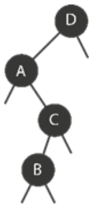
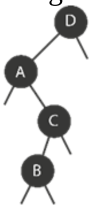
4.1 Survey

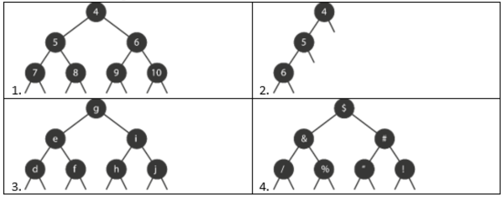
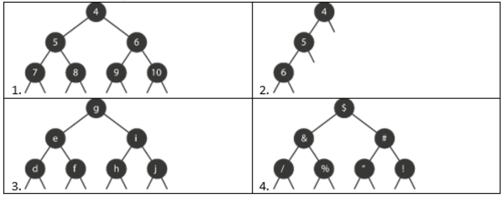
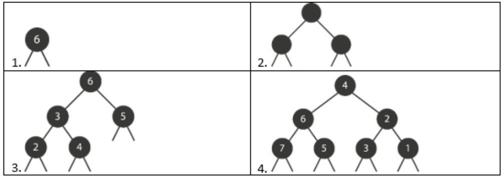
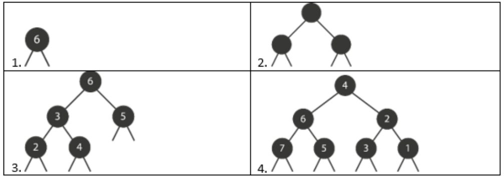
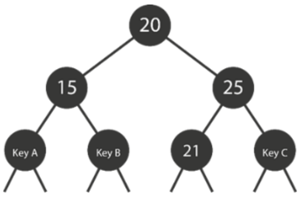
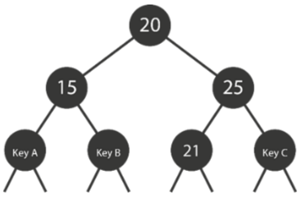
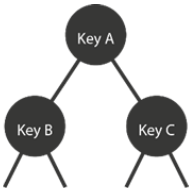
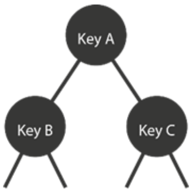
English Version	Versión en español
• Does the content is aligned with the learning objectives? • Is the question relevant to the learning process of binary search trees? • Is the idea of the question clear and readable? • Is it possible to answer the question without looking the answers? • Are the answers provided reasonable answers? • Are the answers homogeneous regarding length and drafting style? • Are the answers independent of each other? • Are the questions drafted affirmatively?	• El contenido de la pregunta se ajusta al objetivo de aprendizaje. • El contenido de la pregunta es relevante para el proceso de aprendizaje del Árbol Binario de Búsqueda. • La idea central de la pregunta es clara. • Es posible contestar la pregunta sin necesidad de ver las respuestas. • Las respuestas son razonables. • Las respuestas son homogéneas en longitud y estilo de redacción. • Las respuestas son independientes una de la otra. • Las respuestas están escritas de manera positiva.

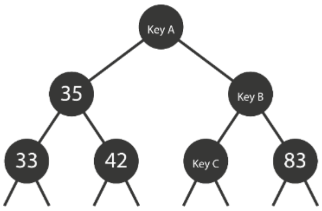
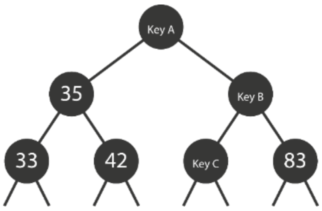
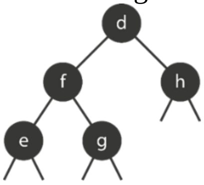
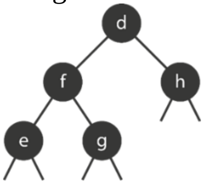
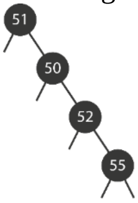
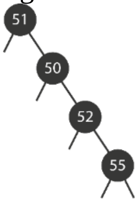
4.2 First Version

English Version	Versión en español
1. Select the option that better defines the concept of Binary Tree. a. A Binary Tree is a data structure made up of nodes that possess multiple links that point to other nodes, and they cannot be null. b. A Binary Tree is a data structure that possesses two nodes with a left and right link that point to objects that do not belong to the data structure, and they can be null. c. A Binary Tree is a data structure implemented using a bidimensional array that stores null values or the index value of another array's elements. d. A Binary Tree is a null link or a node with a left link and a right link, each reference to subtrees that are themselves binary trees. (Correct answer)	1. Seleccione la opción que mejor define el concepto de Árbol Binario. a. Un Árbol Binario es una estructura de datos formada por nodos con múltiples enlaces que apuntan a otros nodos y nunca pueden ser nulos. b. Un Árbol Binario es una estructura de datos formados por únicamente dos nodos con un enlace izquierdo y otro derecho que apuntan a objetos fuera de la estructura o a un valor nulo. c. Un Árbol Binario es una estructura de datos formada por una matriz bidimensional la cual almacena valores nulos o los índices hacia otros elementos de la matriz. d. Un Árbol Binario es un enlace que apunta a un valor nulo o a un nodo con un enlace derecho y otro izquierdo, estos a su vez referencian un subárbol que cumple la definición del Árbol Binario. (Respuesta)
2. Select the number of nodes of a Binary Tree that can point to another node of the tree. a. 0 b. 1 (Correct answer)	2. ¿Por cuántos nodos puede ser apuntado un nodo de un Árbol Binario? a. 0 b. 1 (Respuesta) c. 2

c. 2 d. 3	d. 3
3. Select the maximum number of nodes that can be pointed to by a Binary Tree node. a. 0 b. 1 c. 2 (Correct answer) d. 3	3. ¿Cuál es la cantidad máxima de nodos a la que puede apuntar un nodo de un Árbol Binario? a. 0 b. 1 c. 2 (Respuesta) d. 3
4. What are the names of the links of a Binary Tree node? a. First link and second link b. Link 1 and link 2 c. Left link and right link (Correct answer) d. Child 1 link and child 2 link.	4. ¿Cómo se les conoce a los enlaces de un nodo de un Árbol Binario? a. Primer enlace y segundo enlace b. Enlace 1 y enlace 2 c. Enlace izquierdo y enlace derecho (Respuesta) d. Enlace hijo 1 y enlace hijo 2
5. Select the option that best describes the parent node of a Binary Tree. a. The parent node is the node that points to another node. (Correct answer) b. The parent node is the node that is pointed to by another node. c. The parent node is the node that points an object outside of the data structure. d. The parent node is the rightmost node of the Binary Tree.	5. Seleccione la opción que mejor describe a un nodo padre de un Árbol Binario. a. El nodo padre es el nodo que apunta a otro nodo. (Respuesta) b. El nodo padre es el nodo que es apuntado por otro nodo. c. El nodo padre es el nodo que apunta a un objeto fuera de la estructura de datos. d. El nodo padre es el nodo más a la derecha del Árbol Binario.
6. Select the option that best describes the root node of a Binary Tree. a. The root node is the only node that does not possess a parent node. (Correct answer) b. The root node is the node with the lesser comparable key's value. c. The root node does not possess links. d. The root node is the only node that possess more than one parent node.	6. Seleccione la opción que mejor describe al nodo raíz. a. El nodo raíz no posee un nodo padre. (Respuesta) b. El nodo raíz posee la llave con el valor más bajo. c. El nodo raíz no posee ningún enlace. d. El nodo raíz posee un único nodo padre.
7. Consider the following trees.  Select the tree that is a Binary Tree. a. 1 b. 2 c. 3 d. 4 (Correct Answer)	7. Considere los siguientes árboles.  Seleccione el árbol que es Árbol Binario. a. 1 b. 2 c. 3 d. 4 (Respuesta)
8. Consider the following tree. 	8. Considere el siguiente Árbol Binario. 

Select the Binary Tree’s root node. a. A b. D c. E (Correct Answer) d. F	Seleccione el nodo raíz del Árbol Binario anterior. a. A b. D c. E (Respuesta) d. F
9. Consider the following tree.  Select the Binary Tree’s root node. a. A b. B c. C d. D (Correct Answer)	9. Considere el siguiente Árbol Binario.  Seleccione el nodo raíz del Árbol Binario anterior. a. A b. B c. C d. D (Respuesta)
10. Select the option that best describes the Binary Search Property. a. The comparable key in any node is larger than the keys in all nodes in that node’s left subtree and smaller than the keys in all nodes in that node’s right subtree. (Correct Answer) b. The comparable key in any node is smaller than the keys in all nodes in that node’s left subtree and larger than the keys in all nodes in that node’s right subtree. c. The comparable key in any node is larger than the number of nodes of that node’s left subtree and smaller than the number of nodes of that node’s right subtree. d. The comparable key in any node is smaller than the number of nodes of that node’s left subtree and larger than the number of nodes of that node’s right subtree.	10. Seleccione la opción que describe la propiedad del Árbol Binario de Búsqueda. a. La llave de cualquier nodo N es mayor que la llave de cualquier otro nodo que pertenece al subárbol izquierdo del nodo N y menor que la llave de cualquier otro nodo que pertenece al subárbol derecho del nodo N. (Respuesta) b. La llave de cualquier nodo N es menor que la llave de cualquier otro nodo que pertenece al subárbol izquierdo del nodo N y mayor que la llave de cualquier otro nodo que pertenece al subárbol derecho del nodo N. c. La llave de cualquier nodo N es mayor que la cantidad de nodos del subárbol izquierdo del nodo N y es menor que la cantidad de nodos del subárbol derecho del nodo N. d. La llave de cualquier nodo N es menor que la cantidad de nodos del subárbol izquierdo del nodo N y es mayor que la cantidad de nodos del subárbol derecho del nodo N.
11. Select the option that best describes the purpose of the comparable key of a Binary Search Tree node. a. The comparable key keeps the number of child nodes of a node in a Binary Search Tree. b. The comparable key is used to organize the node in the Binary Search Tree. (Correct Answer) c. The comparable key is utilized to categorize the type of node of each node in a Binary Search Tree.	11. ¿Para qué se utiliza la llave comparable de los nodos del Árbol Binario de Búsqueda? a. Las llaves comparables se utilizan para contabilizar los nodos hijo de los nodos del Árbol Binario de Búsqueda. b. Las llaves comparables se utilizan para organizar los nodos del Árbol Binario de Búsqueda. (Respuesta) c. Las llaves comparables se utilizan para categorizar el tipo de nodo de un Árbol Binario de Búsqueda. d. Las llaves comparables se utilizan para

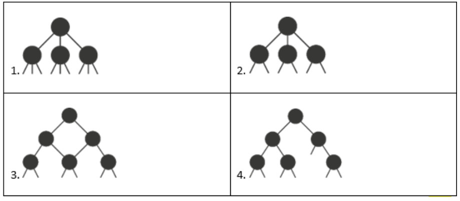
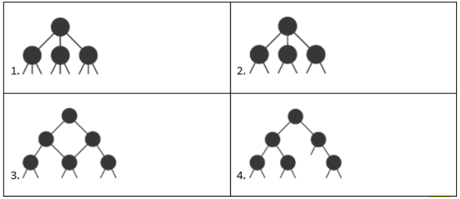
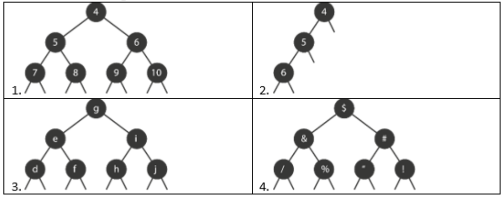
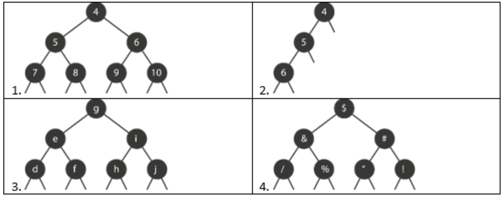
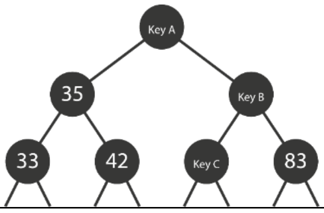
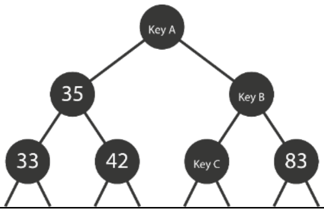
d. The comparable key stores the level of depth of the node in the Binary Search Tree.	contabilizar el nivel en el que se encuentra el nodo dentro del Árbol Binario de Búsqueda.
12. Consider the following trees.  4 5 6 7 8 9 10 1. 4 5 6 2. g e i d f h j 3. $ & # / % * ! 4. Select the tree that is a Binary Search Tree. a. 1 b. 2 c. 3 (Correct Answer) d. 4	12. Considere los siguientes Árboles.  4 5 6 7 8 9 10 1. 4 5 6 2. g e i d f h j 3. $ & # / % * ! 4. Seleccione el árbol que es Árbol Binario de Búsqueda. a. 1 b. 2 c. 3 (Respuesta) d. 4
13. Consider the following trees.  6 1. 2. 6 3 5 2 4 3. 4 6 2 7 5 3 1 4. Select the tree that is a Binary Search Tree. a. 1 (Correct Answer) b. 2 c. 3 d. 4	13. Considere los siguientes Árboles.  6 1. 2. 6 3 5 2 4 3. 4 6 2 7 5 3 1 4. Seleccione el árbol que es Árbol Binario de Búsqueda. a. 1 (Respuesta) b. 2 c. 3 d. 4
14. Consider the following Binary Search Tree.  Select the option that correctly completes the values of the comparable keys of the previous Binary Search Tree. a. Key A = 11; Key B = 12; Key C = 13 b. Key A = 10; Key B = 16; Key C = 99 (Correct Answer) c. Key A = 27; Key B = 28; Key C = 29 d. Key A = 26; Key B = 22; Key C = 14	14. Considere el siguiente Árbol Binario de Búsqueda.  Seleccione la opción que posee los valores de las llaves que completan correctamente el Árbol Binario de Búsqueda anterior. a. Llave A = 11; Llave B = 12; Llave C = 13 b. Llave A = 10; Llave B = 16; Llave C = 99 (Respuesta) c. Llave A = 27; Llave B = 28; Llave C = 29 d. Llave A = 26; Llave B = 22; Llave C = 14
15. Consider the following Binary Search Tree.  Select the option that correctly completes the	15. Considere el siguiente Árbol Binario de Búsqueda.  Seleccione la opción que posee los valores de

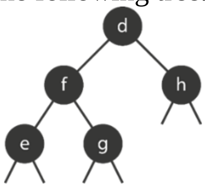
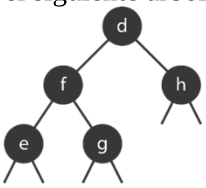
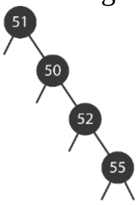
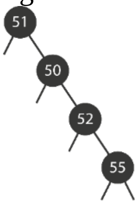
values of the comparable keys of the previous Binary Search Tree. a. Key A = 1; Key B = 2; Key C = 3 b. Key A = 16; Key B = 14; Key C = 15 c. Key A = 9; Key B = 8; Key C = 7 d. Key A = 11; Key B = 10; Key C = 12 (Correct Answer)	las llaves que completan correctamente el Árbol Binario de Búsqueda anterior. a. Llave A = 1; Llave B = 2; Llave C = 3 b. Llave A = 16; Llave B = 14; Llave C = 15 c. Llave A = 9; Llave B = 8; Llave C = 7 d. Llave A = 11; Llave B = 10; Llave C = 12 (Respuesta)
16. Consider the following Binary Search Tree.  Select the option that correctly completes the values of the comparable keys of the previous Binary Search Tree. a. Key A = 50; Key B = 77; Key C = 60 (Correct Answer) b. Key A = 36; Key B = 43; Key C = 37 c. Key A = 20; Key B = 37; Key C = 44 d. Key A = 91; Key B = 82; Key C = 78	16. Considere el siguiente Árbol Binario de Búsqueda.  Seleccione la opción que posee los valores de las llaves que completan correctamente el Árbol Binario de Búsqueda anterior. a. Llave A = 50; Llave B = 77; Llave C = 60 (Respuesta) b. Llave A = 36; Llave B = 43; Llave C = 37 c. Llave A = 20; Llave B = 37; Llave C = 44 d. Llave A = 91; Llave B = 82; Llave C = 78
17. Consider the following tree.  Why does the previous tree violate the Binary Search Tree definition? a. Because the keys of the tree use nominal values instead of ordinal values. b. Because the keys most be integers. c. Because the keys infringe the Binary Search Tree property. (Correct Answer) d. Because the keys are alphabet letters.	17. Considere el siguiente árbol.  ¿Por qué motivo el árbol anterior viola la definición de Árbol Binarios de Búsqueda? a. Porque las llaves comparables del árbol utilizan valores nominales en vez de ordinales. b. Porque las llaves comparables del árbol deben ser números. c. Porque las llaves comparables del árbol violan la propiedad del Árbol Binario de Búsqueda. (Respuesta) d. Porque las llaves comparables del árbol utilizan letras del alfabeto.
18. Consider the following tree.  Why does the previous tree violate the Binary Search Tree property? a. Because the tree does not have any left	18. Considere el siguiente árbol.  ¿Por qué motivo el árbol anterior viola la propiedad de los Árboles Binarios de Búsqueda?

child. b. Because the node 55 is larger than the node 51. c. Because the root node most be the node with the largest key value. d. Because the node 50 is smaller than its parent node. (Correct Answer)	a. Porque el árbol no posee hijos izquierdos. b. Porque el nodo 55 es mayor que el nodo 51. c. Porque el nodo raíz debería ser el nodo con el valor mayor. d. Porque el nodo 50 es menor que su nodo padre. (Respuesta)

4.3 Final Version

English Version	Versión en español
1. Select the option that better defines the concept of Binary Tree. e. A Binary Tree is a data structure made up of nodes that possess multiple links that point to other nodes, and they cannot be null. f. A Binary Tree is a data structure that possesses two nodes with a left and right link that point to objects that do not belong to the data structure, and they can be null. g. A Binary Tree is a data structure implemented using a bidimensional array that stores null values or the index value of another array's elements. h. A Binary Tree is a null link or a node with a left link and a right link, each reference to subtrees that are themselves binary trees. (Correct answer)	1. Seleccione la opción que mejor define el concepto de Árbol Binario. e. Un Árbol Binario es una estructura de datos formada por nodos con múltiples enlaces que apuntan a otros nodos y nunca pueden ser nulos. f. Un Árbol Binario es una estructura de datos formados por únicamente dos nodos con un enlace izquierdo y otro derecho que apuntan a objetos fuera de la estructura o a un valor nulo. g. Un Árbol Binario es una estructura de datos formada por una matriz bidimensional la cual almacena valores nulos o los índices hacia otros elementos de la matriz. h. Un Árbol Binario es un enlace que apunta a un valor nulo o a un nodo con un enlace derecho y otro izquierdo, estos a su vez referencian un subárbol que cumple la definición del Árbol Binario. (Respuesta)
2. Select the number of nodes of a Binary Tree that can point to another node of the tree. e. 0 f. 1 (Correct answer) g. 2 h. 3	2. ¿Por cuántos nodos puede ser apuntado un nodo de un Árbol Binario? e. 0 f. 1 (Respuesta) g. 2 h. 3
5. Select the option that best describes the parent node of a Binary Tree. e. The parent node is the node that points to another node. (Correct answer) f. The parent node is the node that is pointed to by another node. g. The parent node is the node that points an object outside of the data structure. h. The parent node is the rightmost node of the Binary Tree.	5. Seleccione la opción que mejor describe a un nodo padre de un Árbol Binario. e. El nodo padre es el nodo que apunta a otro nodo. (Respuesta) f. El nodo padre es el nodo que es apuntado por otro nodo. g. El nodo padre es el nodo que apunta a un objeto fuera de la estructura de datos. h. El nodo padre es el nodo más a la derecha del Árbol Binario.
6. Select the option that best describes the root node of a Binary Tree. e. The root node is the only node that does	6. Seleccione la opción que mejor describe al nodo raíz. e. El nodo raíz no posee un nodo padre.

not possess a parent node. (Correct answer) f. The root node is the node with the lesser comparable key's value. g. The root node does not possess links. h. The root node is the only node that possess more than one parent node.	(Respuesta) f. El nodo raíz posee la llave con el valor más bajo. g. El nodo raíz no posee ningún enlace. h. El nodo raíz posee un único nodo padre.
7. Consider the following trees.  Select the tree that is a Binary Tree. e. 1 f. 2 g. 3 h. 4 (Correct Answer)	7. Considere los siguientes árboles.  Seleccione el árbol que es Árbol Binario. e. 1 f. 2 g. 3 h. 4 (Respuesta)
11. Select the option that best describes the purpose of the comparable key of a Binary Search Tree node. e. The comparable key keeps the number of child nodes of a node in a Binary Search Tree. f. The comparable key is used to organize the node in the Binary Search Tree. (Correct Answer) g. The comparable key is utilized to categorize the type of node of each node in a Binary Search Tree. h. The comparable key stores the level of depth of the node in the Binary Search Tree.	11. ¿Para qué se utiliza la llave comparable de los nodos del Árbol Binario de Búsqueda? e. Las llaves comparables se utilizan para contabilizar los nodos hijo de los nodos del Árbol Binario de Búsqueda. f. Las llaves comparables se utilizan para organizar los nodos del Árbol Binario de Búsqueda. (Respuesta) g. Las llaves comparables se utilizan para categorizar el tipo de nodo de un Árbol Binario de Búsqueda. h. Las llaves comparables se utilizan para contabilizar el nivel en el que se encuentra el nodo dentro del Árbol Binario de Búsqueda.
12. Consider the following trees.  1. 4 5 6 7 8 9 10 2. 4 5 6 3. g e i d f h j 4. $ & # / % * ! Select the tree that is a Binary Search Tree. e. 1 f. 2 g. 3 (Correct Answer) h. 4	12. Considere los siguientes Árboles.  1. 4 5 6 7 8 9 10 2. 4 5 6 3. g e i d f h j 4. $ & # / % * ! Seleccione el árbol que es Árbol Binario de Búsqueda. e. 1 f. 2 g. 3 (Respuesta) h. 4
16. Consider the following Binary Search Tree. 	16. Considere el siguiente Árbol Binario de Búsqueda. 

Select the option that correctly completes the values of the comparable keys of the previous Binary Search Tree. e. Key A = 50; Key B = 77; Key C = 60 (Correct Answer) f. Key A = 36; Key B = 43; Key C = 37 g. Key A = 20; Key B = 37; Key C = 44 h. Key A = 91; Key B = 82; Key C = 78	Seleccione la opción que posee los valores de las llaves que completan correctamente el Árbol Binario de Búsqueda anterior. e. Llave A = 50; Llave B = 77; Llave C = 60 (Respuesta) f. Llave A = 36; Llave B = 43; Llave C = 37 g. Llave A = 20; Llave B = 37; Llave C = 44 h. Llave A = 91; Llave B = 82; Llave C = 78
17. Consider the following tree.  Why does the previous tree violate the Binary Search Tree definition? e. Because the keys of the tree use nominal values instead of ordinal values. f. Because the keys most be integers. g. Because the keys infringe the Binary Search Tree property. (Correct Answer) h. Because the keys are alphabet letters.	17. Considere el siguiente árbol.  ¿Por qué motivo el árbol anterior viola la definición de Árbol Binarios de Búsqueda? e. Porque las llaves comparables del árbol utilizan valores nominales en vez de ordinales. f. Porque las llaves comparables del árbol deben ser números. g. Porque las llaves comparables del árbol violan la propiedad del Árbol Binario de Búsqueda. (Respuesta) h. Porque las llaves comparables del árbol utilizan letras del alfabeto.
18. Consider the following tree.  Why does the previous tree violate the Binary Search Tree property? e. Because the tree does not have any left child. f. Because the node 55 is larger than the node 51. g. Because the root node most be the node with the largest key value. h. Because the node 50 is smaller than its parent node. (Correct Answer)	18. Considere el siguiente árbol.  ¿Por qué motivo el árbol anterior viola la propiedad de los Árboles Binarios de Búsqueda? e. Porque el árbol no posee hijos izquierdos. f. Porque el nodo 55 es mayor que el nodo 51. g. Porque el nodo raíz debería ser el nodo con el valor mayor. h. Porque el nodo 50 es menor que su nodo padre. (Respuesta)

Appendix 5: Materials of the Evaluations (Spanish Version)

5.1 First Evaluation

5.1.1 Consent Form

FORMULARIO DE CONSENTIMIENTO INFORMADO

Evaluación de un Juego para la Enseñanza de Árboles Binarios de Búsqueda

Investigadores: Alberto Rojas Salazar y Paula Ramírez Alfaro

Usted ha sido invitado a participar en un estudio llevada a cabo por Alberto Rojas Salazar y Paula Ramírez Alfaro. Esta evaluación tiene el objetivo de evaluar los efectos educativos y motivacionales de un video juego (disponible únicamente para PC) para la enseñanza de Árboles Binarios de Búsqueda (ABB) llamado DS-Hacker.

Durante la evaluación se le solicitará que llene una encuesta demográfica, una prueba inicial sobre los ABBs, una actividad educativa sobre los ABBs, una prueba final y una evaluación de las tareas realizadas. Con respecto a la encuesta demográfica, no se le solicitará información personal o datos que puedan conducir a su identificación. Se estima que la duración total del estudio es de una hora o menos.

La evaluación se llevará a cabo por medio de este sitio web que lo irá guiando durante el estudio. La información recolectada será almacenada a una base de datos en Firebase y la únicas que personas que poseen acceso a los datos serán los investigadores. Además, debido a que la información es anónima, no es posible retirarla del experimento una vez que se hayan enviado los formularios.

Los datos y resultados de la evaluación serán publicados en ponencias y/o artículos académicos. Los datos publicados serán agregados de forma anónima y mostrados en categorías.

Su participación es voluntaria y puede retirarse de la evaluación en cualquier momento sin ninguna consecuencia.

Si tiene alguna duda sobre esta investigación puedo referirse a Alberto Rojas Salazar (correo electrónico rojassaa@tcd.ie)

Declaración:

- Tengo 18 años o más y soy competente para dar mi consentimiento.
- He leído, o me han leído, el documento que proporciona información sobre esta investigación y este formulario de consentimiento. He tenido la oportunidad de hacer preguntas y todas mis preguntas han sido respondidas satisfactoriamente y he entendido la descripción de la evaluación que se me ha proporcionado.
- Entiendo que esta evaluación es completa anónima, por lo que no es posible retirar mis datos una vez enviada la encuesta.
- Estoy de acuerdo en que mis datos se utilizan con fines científicos y no tengo ninguna objeción de que los datos de las pruebas se publiquen en publicaciones científicas de manera que no revelen mi identidad.
- Entiendo que puedo negarme a responder cualquier pregunta y que puedo retirarme en cualquier momento.
- Acepto libre y voluntariamente ser parte de esta evaluación, sin perjuicio de mis derechos legales y éticos.

5.1.2 General Instructions

Instrucciones Generales de la Evaluación

La siguiente evaluación consta de cinco secciones:

1. Encuesta demográfica [2 minutos ─ 6 preguntas].

2. Primera prueba sobre Árboles Binarios de Búsqueda (ABB) [10 minutos ─ 10 preguntas].

3. Actividad educativa sobre ABBs. [15-30 minutos]. Para la actividad deberá descargar el material adicional del siguiente sitio:

https://www.dropbox.com/s/z2lje7qnxrw0ngg/DS-Hacker%20v1.1%20Exp.zip?dl=1

Debido al tamaño del archivo (300 MB), se recomienda empezar a descargarlo desde ahora. El archivo se encuentra comprimido (zip) y cifrado. La contraseña se le brindará más en las siguientes etapas.

Si no posee un programa para descomprimir archivos, se recomienda el software libre 7Zip, que puede descargar del siguiente enlace:

https://www.7-zip.org/download.html

4. Última prueba sobre Árboles Binarios de Búsqueda (ABB) [10 minutos ─ 10 preguntas].

5. Evaluación de la actividad de aprendizaje [10 minutos ─ 22 preguntas].

Siga el orden establecido y evite presionar el botón para “devolverse” del navegador.

Al finalizar la evaluación, el sitio web le entregará un código de 12 caracteres que deberá guardar y presentar al profesor(a) para comprar su participación en el experimento.

Si posee alguna pregunta durante alguna de las actividades puede comunicarse con la profesora Paula Ramírez Alfaro.

5.1.3 Demographic Survey

Encuesta demográfica

1. ¿Cuál es su edad?
2. ¿Cuál es su género?
3. ¿Cuántas horas juega videojuegos a la semana?
4. ¿A cuál escuela pertenece?
5. ¿Cuál es al año académico que está cursando?
6. ¿Cuáles son sus conocimientos de programación?

5.1.4 DS-Hacker 3D Learning Activity Instructions

Actividad de Aprendizaje: Jugar DS-Hacker

A continuación, deberá jugar DS-Hacker, un videojuego de acción-aventura que consta de 4 niveles. Tenga en cuenta lo siguiente:

1. Debe jugar los 4 niveles y finalizarlos.

2. Se estima que la sesión de juego puede durar 15-30 minutos.

3. Como se mencionó al inicio de la evaluación, debe descargar el juego del siguiente enlace:

https://www.dropbox.com/s/z2lje7qnxrw0ngg/DS-Hacker%20v1.1%20Exp.zip?dl=1

Luego, debe descomprimir el archivo (.zip). Si no posee un programa para descomprimirlo, puede descargar el programa 7-Zip por medio del siguiente enlace:

https://www.7-zip.org/download.html

La contraseña para descomprimir el archivo es:

DS-Hacker2020**

4. Para iniciar el juego, debe abrir el archivo DS-Hacker.exe. Luego, aparecerá un menú donde debe seleccionar el idioma y luego presionar el botón "Jugar". Para controla el avatar del juego, utilice el teclado y el ratón.

5. Ponga atención a la narración y a los desafíos del juego. Ellos contienen el contenido educativo necesario para responder la siguiente prueba.

6. Al finalizar el cuarto nivel, el juego le presentará un código que debe utilizar para continuar con la evaluación.

5.1.5 Control Learning Activity Instructions

Actividad de Aprendizaje: Ver Video Tutoriales sobre Árboles Binarios de Búsqueda

A continuación, deberá ver por completo los siguientes video tutoriales.

Una vez finalizados los dos videos, el botón Continuar se activará.

Conceptos básicos y terminología de árboles binarios

https://www.youtube.com/watch?v=2K8CmQs1jl8

Árboles binarios de búsqueda

https://www.youtube.com/watch?v=Bh61AvHAf90&t

El contenido de estos video tutoriales es necesario para responder la próxima prueba de conocimiento.

5.1.6 Intrinsic Motivation Inventory (Spanish Version)

Cuestionario de Motivación Intrínseca

7. Mientras estaba trabajando en la actividad, yo pensaba sobre cuánto disfrutaba de ella.

8. No me sentí en nervioso ningún momento mientras realizaba la actividad.
9. Sentí que fue mi decisión realizar la actividad.
10. Creo que son bastante bueno realizando la actividad.
11. Encontré la actividad muy interesante.
12. Me sentí tenso al realizar la actividad.
13. Creo que me desempeñé muy bien en esta actividad comparado a otros estudiantes.
14. Realizar la actividad fue divertido.
15. Me sentí relajado al realizar la actividad.
16. Disfruté mucho realizando la actividad.
17. No tuve otra opción más que realizar la actividad.
18. Estoy satisfecho con mi desempeño en la actividad.
19. Me sentí ansioso al realizar la actividad.
20. Pensé que la actividad fue aburrida.
21. Sentí que realizaba lo que quería hacer mientras trabajaba en la actividad.
22. Me sentí realmente capacitado para la actividad.
23. Pensé que la actividad fue bastante interesante.
24. Me sentí presionado al realizar la actividad.
25. Sentí que era mi obligación realizar la actividad.
26. Describiría la actividad como algo muy agradable.
27. Realicé la actividad porque no tenía otra opción.
28. Después de trabajar por un momento en la actividad, me sentí bastante competente (en la actividad).

5.2 Second Evaluation

Some of the materials of the first and second evaluation are the same. For this reason, in this section, we only include those that are different.

5.2.1 Consent Form

Evaluación de Dos Juegos para la Enseñanza de Árboles Binarios de Búsqueda

Investigadores: Alberto Rojas Salazar

Usted ha sido invitado a participar en un estudio llevada a cabo por Alberto Rojas Salazar. Esta evaluación tiene el objetivo de evaluar los efectos educativos y motivacionales de dos video juegos (disponible únicamente para PC) para la enseñanza de Árboles Binarios de

Búsqueda (ABB) llamado DS-Hacker y DS-Hacker 2D.

Durante la evaluación se le solicitará que llene una encuesta demográfica, una prueba inicial sobre los ABBs, una actividad educativa (jugar algún juego) sobre los ABBs, una prueba final y una evaluación de las tareas realizadas. Con respecto a la encuesta demográfica, no se le solicitará información personal o datos que puedan conducir a su identificación. Se estima que la duración total del estudio es de una hora o menos.

La evaluación se llevará a cabo por medio de este sitio web que lo irá guiando durante el estudio. La información recolectada será almacenada a una base de datos en Firebase y la únicas que personas que poseen acceso a los datos serán los investigadores. Además, debido a que la información es anónima, no es posible retirarla del experimento una vez que se hayan enviado los formularios.

Los datos y resultados de la evaluación serán publicados en ponencias y/o artículos académicos. Los datos publicados serán agregados de forma anónima y mostrados en categorías.

Su participación es voluntaria y puede retirarse de la evaluación en cualquier momento sin ninguna consecuencia.

Si tiene alguna duda sobre esta investigación puedo referirse a Alberto Rojas Salazar (correo electrónico rojassaa@tcd.ie).

Declaración:

- Tengo 18 años o más y soy competente para dar mi consentimiento.
- He leído, o me han leído, el documento que proporciona información sobre esta investigación y este formulario de consentimiento. He tenido la oportunidad de hacer preguntas y todas mis preguntas han sido respondidas satisfactoriamente y he entendido la descripción de la evaluación que se me ha proporcionado.
- Entiendo que esta evaluación es completa anónima, por lo que no es posible retirar mis datos una vez enviada la encuesta.
- Estoy de acuerdo en que mis datos se utilizan con fines científicos y no tengo ninguna objeción de que los datos de las pruebas se publiquen en publicaciones científicas de manera que no revelen mi identidad.
- Entiendo que puedo negarme a responder cualquier pregunta y que puedo retirarme en cualquier momento.
- Acepto libre y voluntariamente ser parte de esta evaluación, sin perjuicio de mis derechos legales y éticos.

5.2.2 General Instructions

Instrucciones Generales de la Evaluación

El siguiente experimento consta de cinco secciones:

1. Encuesta demográfica [2 minutos ─ 6 preguntas].

2. Primera prueba sobre Árboles Binarios de Búsqueda (ABB) [10 minutos ─ 10 preguntas].

3. Actividad educativa sobre ABBs. [15-30 minutos]. Para la actividad deberá descargar el material adicional de los siguientes enlaces:

DS-Hacker:

https://www.dropbox.com/s/z2lje7qnxrw0ngg/DS-Hacker%20v1.1%20Exp.zip?dl=1

DS-Hacker 2D:

https://www.dropbox.com/s/31btnl6osrq1phr/DS-Hacker%202D_v1.3.zip?dl=1

Debido al tamaño de los archivos (300 y 100 MB), se recomienda empezar a descargarlo desde ahora. El archivo se encuentra comprimido (zip) y cifrado. La contraseña se le brindará en las siguientes etapas dependiendo del juego que se le asigne.

Si no posee un programa para descomprimir archivos, se recomienda el software libre 7Zip, que puede descargar del siguiente enlace:

https://www.7-zip.org/download.html

4. Última prueba sobre Árboles Binarios de Búsqueda (ABB) [10 minutos ─ 10 preguntas].

5. Evaluación de la actividad de aprendizaje [10 minutos ─ 22 preguntas].

Siga el orden establecido y evite presionar el botón para “devolverse” del navegador.

Al finalizar la evaluación, el sitio web le entregará un código de 12 caracteres que deberá guardar y presentar al profesor(a) para comprobar su participación en el experimento.

5.2.3 DS-Hacker 2D Learning Activity Instructions

Actividad de Aprendizaje: Jugar DS-Hacker 2D

A continuación, deberá jugar DS-Hacker 2D, un videojuego de aventura que consta de 4 niveles. Tenga en cuenta lo siguiente:

1. Debe jugar los 4 niveles y finalizarlos.

2. Se estima que la sesión de juego puede durar 15-25 minutos.

3. Como se mencionó al inicio de la evaluación, debe descargar el juego del siguiente enlace:

https://www.dropbox.com/s/31btnl6osrq1phr/DS-Hacker%202D_v1.3.zip?dl=1

Luego de descargar el archivo debe descomprimirlo. Si no posee un programa para hacerlo, puede descargar 7-Zip utilizando el siguiente enlace:

https://www.7-zip.org/download.html

La contraseña para descomprimir el archivo es:

DS-Hacker2D++2020

4. Para iniciar el juego, debe abrir el archivo Game.exe. Luego, aparecerá un menú donde debe presionar el botón “Jugar”. Para controla el avatar del juego, utilice el teclado y el ratón. Puede revisar la configuración del teclado en el menú. Seleccione Opciones → Configurar Teclado.

5. Ponga atención a la narración y a los desafíos del juego. Ellos contienen el contenido educativo necesario para responder la siguiente prueba.

6. Al finalizar el cuarto nivel, el juego le presentará un código que debe utilizar para continuar con la evaluación.

Appendix 6: Digital Appendix

Digital appendix: https://1drv.ms/f/s!AAbdgU5ExJt6gV8