universität wien

MASTERARBEIT | MASTER'S THESIS

Titel | Title

Enhancing Programming Learnability for Children Through Video Games

verfasst von | submitted by

Nora Volina

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien | Vienna, 2025

Studienkennzahl lt. Studienblatt | Degree programme code as it appears on the student record sheet:	UA 066 921
Studienrichtung lt. Studienblatt | Degree programme as it appears on the student record sheet:	Masterstudium Informatik
Betreut von | Supervisor:	Univ.-Prof. Dipl.-Ing. Dr. Helmut Hlavacs

Acknowledgements

I would like to thank my family and friends whose unwavering support and encouragement have been invaluable throughout the creation of my video game and the writing of this thesis. The acquittances who are now my most trusted peers for always motivating me through their positivity. Thank you to my mentor, Prof. Dr. Helmut Hlavacs, who always listened to my ideas and elevated them to even better outcomes. Every member of my social circle has been a vital pillar of constant feedback.

Abstract

This paper explores the potential of video games as a medium for teaching programming concepts to children. With the new upcoming methods for teaching online and offline, traditional teaching has taken a new path of being digitized. The research studies the effect of video games on the learning process of children and how that can affect the outcome of their perception to new digitized knowledge.

In addition, it investigates the children's problem-solving and logical skills during the gaming stage, which later on translate to an understanding of the concept of what programming is. The result yields a positive outcome to the ability of learning programming while playing a video game. This could indicate that there is a tool that we can further develop to create advanced methods to future learning.

According to an article, there are five ways that video games are good for learnability [Eng23]. While these methods have been experimented with, new methodologies and strategies have been developed. The paper will touch base with some of the methodologies, explaining how they set up the environment, the testing duration, and the end results. Important key aspects that will be talked about in the paper include contextual learning, problem solving and logical skills, motivation, engagement, learner autonomy, and decision making. Game-based programming education helps students motivate themselves and succeed more by creating a platform in which they learn by doing, experiencing, and practicing this method in the learning process [Yil18].

Kurzfassung

In dieser Arbeit wird das Potenzial von Videospielen als Medium für den Programmierunterricht als Konzepte für Kinder untersucht. Mit den neuen Methoden für den Online- und Offline-Unterricht, hat der traditionelle Unterricht einen neuen Weg der Digitalisierung eingeschlagen. Die Forschung untersucht die Wirkung von Videospielen auf den Lernprozess und die Wahrnehmung zu neuem digitalisiertem Wissen von Kindern. Darüber hinaus werden die Problemlösungs- und Logikfähigkeiten der Kinder während des Unterrichts untersucht, welche sich später in einem Verständnis des Programmierkonzepts niederschlagen. Das Ergebnis wirkt sich positiv auf die Fähigkeit, Programmieren beim Spielen eines Videospiels zu lernen, aus. Dies könnte ein Hinweis darauf sein, dass es ein Tool gibt, das wir weiterentwickeln können, um fortschrittliche Methoden für zukünftiges Lernen zu schaffen. Einem Artikel zufolge gibt es fünf Möglichkeiten, wie sich Videospiele positiv auf das Lernen auswirken können. Während mit diesen Methoden experimentiert wurde, wurden neue Methoden und Strategien entwickelt. Diese Arbeit wird sich mit einigen der Methoden befassen, und erklären, wie diese Methoden und Strategien die Umgebung, die Testdauer und die Endergebnisse beeinflussen. Wichtige Schlüsselaspekte, über die in der Arbeit gesprochen wird, sind kontextuelles Lernen, Problemlösungsfähigkeiten und logische Fähigkeiten, Motivation, Engagement, Autonomie der Lernenden und die eigenständige Entscheidungsfähigkeit. Die spielerische Programmierausbildung hilft den Schülern bei der Selbstmotivation und gibt ihnen Erfolgserlebnisse, da sie eine Plattform schafft, auf der sie im Lernprozess Handeln erleben und diese Methoden erarbeiten.

Contents

List of Tables

List of Figures

1 Introduction

The research paper aims to prove games can be a tool for education. Through the power of gamification, adding game mechanics into non-game environments, like a website, online community, learning management system or business' intranet to increase participation. The goal of gamification is to engage with consumers, employees and partners to inspire collaborate, share and interact [WOR24]. There are a few research questions I aim to also prove true.

The focus of the paper and the research is to see how motivated the students will be if they are to study programming through a video game. Using many elements from gamification, I aim to create a mobile game that will get them interested in the game and get them to learn the basics of programming.

Video games have been viewed as entertainment and a leisurely activity for a long time. Being able to capture the attention of users, teaching new skills, but most importantly connecting and creating a community online that can help each other out.

However, now they are seen as a potential learning tool in addition to being used as a form of leisure. In the past few years, researchers have explored new opportunities which will take interactive platforms and use them for an engaging lesson plan. The paper will talk about the unexplored side of games as a medium for learning, especially when it comes to programming languages.

Using games in education has the potential to increase students' motivation and engagement in the learning process, gathering long-lasting practical knowledge [oSE20]. The biggest motivation to achieve high academic marks is having vision of the far future. When starting a new degree, course or even career, we are motivated by what the end result will be since we have been conditioned to always think about our future. Although, when it comes to children they do not live in the same thought process as adults.

For them, they need milestones to celebrate near achievements until they can focus on the bigger picture. This is exactly why video games can offer a reward system to motivate and engage students to learn and be better. In our society, technology has been focal point in growing and improving our daily lives. Being able to create a new kind of technology tool that will start teaching the importance of programming from a young age could further aid in the advancement of scientific knowledge.

Video games have a reward system that offer immediate feedback. This instant gratification combined with a sense of achievement while progressing makes the process more appealing. In addition, they offer a multiplayer environment. Playing in a cooperative mode allows the users to learn from each other. This way, they can share their knowledge and solutions which mimics real-life collaborative projects.

While these ideas work wonderfully in theory, we now have data that support our arguments with positive outcomes. Games like Code Combat, Scartch, RoboCode [For23] successfully have gamified the learning process of code and made it engaging with an interesting story mode. In conclusion, the merge of video games and learning new skills is a fast approaching new method that can be used more frequently in the future.

Children are growing up with this technology and they get to almost become one with it. As part of their natural habitat, regard these systems as children's inclination. They perceive AI systems as part of their everyday environment, and children from different backgrounds may have varying perceptions [Res23].

This could be an advantage because their everyday lives could turn into a game where they are constantly gathering useful data and turning it into a skill. The goal is to create simple yet well formulated instructions that lead the player to trigger one step after another, while getting familiar with the game environment, interactive user interface and following instructions.

The purpose of the study is to show that there is a positive feedback that using video games to teach can be a powerful new tool that fits the modern brain of the children of this generation. For years, video games have been used for entertainment providing countless hours of immersive game play.They offer a form of interactive storytelling that can be deeply engaging, and the fun and enjoyment derived from playing games can lead to hours of sustained attention [oE21].

Being taken into a virtual world to build a new persona, attributes and a story of your own is the motivation behind the escapism from the real world. Codonia, the video game that I have created, is a magical world that comes with its own set of rules. Through game play the children start to understand that every object, action, and event is governed by a set of rules that can be manipulated. A concept that will allow them to think logically, debug their code and learn from their mistakes. The learning phase includes taking problems and breaking them down to solvable solutions.

Although, through the recent years the educational potential of games has been rising. From the above research paper we can open up a discussion that might prove that "researchers and educators are increasingly recognizing that the interactive nature of video games can make them powerful tools for learning [oE21]."

A term that has been created in 1954 is now making a comeback, edutainment. Educational entertainment, meaning that learning and play can co-exist. With that being said, making a new learning tool does come with its challenges. Making sure that the teaching aspect is integrated in the game instead of feeling like an add-on. This is important because for this new tool to succeed the user needs to truly be immersed in the new world and forget they are gaining knowledge.

Throughout my research I was focused on trying to answer my research questions. The first question being "Are the players motivated when learning through a game?" and the second question "Is the self-perceived competence higher with learning through a game?" The answers to these questions would determine if whether the educational game I have created has a positive feedback on learnability for children.

Before I had my experiment with the finished product I had two hypothesis that are related to my research questions.

- Hypothesis 1: Gamification increases the motivation of learners
- Hypothesis 2: Gamification increases the self-perceived competence of learners

In addition to focusing on the outcome of the hypothesis I also aimed in answering the questions in the Table 1.1. The hypothesis is easier to prove since I can have solid data prove if my theories are correct or not. Through a demo experiment and feedback from a group of participants I can tell if the game I created can be further developed for education.

RQ1 aims to prove that games do make the student more motivated to learn. RQ2 aims to prove that the self-confidence of the children increases their ability to solve simple to complex tasks. RQ3 aims to prove that gamification can one day become the primary source of education.

Number	Research Question
RQ1	Are learners more motivated when learning with a game
RQ2	Is the self-perceived competence higher with learning with a game
RQ3	Does gamification have a higher learning outcome than traditional teaching

Table 1.1: Research Questions

The motivation refers to the players wanting to finish the levels and get to the end of the game. The need to know more about the story-line and still be engaged while playing the game. Self-perceived competence is the players own assessment of their ability to perform a task and achieve goals. In simpler words, its the confidence that a player has in knowledge and skills in a specific area or task.

Codonia, the educational game I have created, aims to be a part of edutainment. A game where children can learn programming through playing a video game. The video game includes a story-line for entertainment, a main objective for motivation, characters who guide the player through the levels and user interface that is easy to understand and navigate.

2 Literature Review

With the rise of a new learning potential, video games became a serious study to researchers and educators. Taking a powerful tool and forging it to fit the needs of certain audiences, in particular children. There have been many games that have been created throughout these years which are used to teach problem-solving skills, promote creativity and even help children practice their language skills [oE21]. A new approach to learning in the form of educational games has been adopted in recent years, especially in English language teaching. The educational game learning approach used to teach English to non-native English-speakers who use English as a second or foreign language has recorded great success [Kir02]. Programming languages are similar to learning a new language. There is syntax, ways to formulate functions, variables and so on. This can be taken and applied into any kind of learning.

According to the user experience, the experiment was valued through different categories: Usability, Confidence, Challenge, Satisfaction, Fun, Focused Attention and Relevance. The participants were rather neutral to two out of the three aspects that constitute the focused attention dimension. Although the students agreed that there was something at the beginning of the game that captured their attention , they were divided as to whether they forgot about their surroundings and lost track of time while playing [MDP22].

A research paper written by Stelios Xinogalos and Maya Satratzemi named “The Use of Educational Games in Programming Assignments: SQL Island as a Case Study [XS22]’created an interesting investigation during a compulsory undergraduate course in Web programming. The study involved fifty-six students who had to answer a questionnaire that evaluated the experience of each student before-hand. SQL Island is a game that has a story-line, task, objectives and progression. It starts off telling the player that they are stranded on an island after a plane crash and have to use their SQL skills to survive and get off the island.

There were numerous objectives through SQL queries, such as locating resources, solving puzzles and finding other people on the island. Through the completion of the task, the students had to use commands such as SELECT, WHERE and JOIN. Through this you progress in the story and learn more about the island.

A large body of students found the experience to be positive. Out of the research group, the average game playing was 84 minutes. Two students spend 70 minutes, one student spent 80 minutes, and another spent 180 minutes. Four students were able to finish the game in 2 hours. While there were students who struggled to finish the game, they did so by getting immediate feedback and hints throughout game time. In the end, the study was perceived as short-term learning and the evaluated questionnaire was based on the MEEGA+ model which is used in educational games in computing education. MEEGA+ model is a systematic framework that is used t evaluate the quality of education games. The focus is on player experience and usability.They key aspects focus on the evaluation of the perceived quality of educational games, considering factors like engagement, usability, and learning outcomes [PGvWB17], measurement instrument: ''The model includes a standardized questionnaire and analysis tools to systematically gather and analyze data from students‘ [PGvWB23], application: It can be applied to both digital and non-digital educational games, making it versatile and widely applicable [PGvWB23] and research design:MEEGA+ supports a one-shot post-test research design, allowing for quick and non-intrusive evaluations after game application [PGvWB23].

In a research done in 2022, the co-authors talk about three types of video games that can be used as a case study. These categories were found effective in promoting children's social problem-solving skills. Although, further research has to be done in a more controlled group to see any effects in other fields.

The categories are as followed:

- **Educational Games:** These are primarily produced for educational purposes. They make up half of the video games used in the studies reviewed.
- **Serious Games:** These games, which make up 30 percent of the video games used in the studies, are designed for purposes beyond pure entertainment.
- **Entertainment Games:** These games, which account for 20 percent of the video games used in the studies, are primarily designed for entertainment [oE21].

While these theories have been tested in small group sizes there is also a research that was done in particularly only two youths. They used user experience approaches like the think-aloud protocol and interviews, grounded in the theoretical framework of social semiotics, to analyze the game play videos and discuss the implications for the youths' learning [TL22]. This particular methodology uses a case study approach to examine game play.

- **Data Collection:** The researchers collected data through user experience approaches, such as the think-aloud protocol and interviews.
- **Data Analysis:** The data collected was analyzed using the theoretical framework of social semiotics [TL22].

In the data collection phase the think-aloud protocol was mentioned. This means that the two participants verbalize their thoughts while they're doing a task. This would mean that they say what they're thinking while playing a video game. Meanwhile, the researchers are taking notes about the thought process. The second point talked about social semiotics.

Social semiotics focuses on finding meaning and interpreting context. In our case, it meant to analyze the elements that are in the video game, how they interact, what they do and what is the outcome of interacting with them. This research can also be found in another paper that supports analytical and critical thinking.

The research acknowledges that serious games may be interpreted as a logical and modern continuation of the Neo Platonic concept of serio ludere, defined as a serious and methodologically well-planned analysis of, at first glance, only lighthearted and even superficial examples [Res23]. That seems like a complicated sentence but taking it apart and analyzing the sentence we come to an understanding that we can engage in activities like video games that seem lighthearted and superficial at a first glance but we do so while taking it in a serious and methodical way.

Researchers are suggesting that serious games can be used in a new form of presenting fun environments that teach real life skills. This takes me to my point, where video games can teach you how to code without the user realizing they are learning. A very concrete example is Code Combat, a game that is dedicated to teach coding. This is an innovative HTML5 role-playing game (RPG) that teaches players the basics of programming.

It encourages students to actively learn and apply the fundamental concepts of Python, JavaScript, and C++ [oE21]. Another example is SQL Murder Mystery. This game is for users who love to solve murders but they get to do it while learning SQL commands. The way to learn is by repeated action. This is exactly what they found out when the authors talked about how they found that games have advanced far beyond using simple tutorials or demonstration screens and adopt a range of repeatable and reusable design strategies using visual cues to facilitate learning [PT22]. They also found out that certain design strategies play a big role in how the person learns. Visual cues, repeatable patterns and contextual learning are one of the 7 ways you can capture someones attention to finish a game or quest they have started. A study that focused on Computational Thinking (CT) in primary schools have come back with positive results. The study was conducted on sixty-one students and the final say was that educational programming games can help primary school students develop CT skills or understand fundamental programming concepts‘ [oCiE23]. The question in mind opens up possibilities to explore more areas as to what we can teach through video games? We can take this tool and mold it to fit different skills, scenarios and needs. In a paper written by Katrina Serrano, the whole experience is summed up in one sentence that says when games are intentionally designed to align with educational objectives, they enhance learning experience. [Ser19].

Karl M. Kapp is a professor of Instructional Technology at Bloomsburd University. He is an expert in the field of gamification and instructional Technology. He has written many influential books on gamification and learning and one of those books includes “The Gamification of Learning and Instruction.” The book is a valuable resource for educators, trainers, and learning professionals looking to leverage gamification to improve learning outcomes [Kap12]

- **Introduction to Gamification**:Kapp introduces the concept of gamification and explains its importance in creating engaging learning experiences. [Kap12]
- **Game Elements**: The book discusses various game elements such as points, badges, leaderboards, and how they can be used to enhance learning. [Kap12].
- **Theories and Research**: Insights into the theories behind gamification and includes peer-reviewed research to support the effectiveness of game-based learning [Kap12].

- **Practical Applications**: The book offers practical strategies for designing and implementing gamified learning experiences, including creating a game design document and managing the gamification process [Kap12].
- **Case Studies**: Kapp includes real-world examples and case studies of successful gamification efforts in different settings [Kap12].

There is a lot of insightful material for anyone who wants to know more or start creating video games. This book has been a huge help to me when I first started my research about Codonia. I was able to figure out the genre, what I wanted the result of my game to be, how to make it motivational for children, documentation throughout the creation of the game and how to create a case study.

Through the years of research the games that are being mostly mentioned in his book are:

- **CodeCombat**: CodeCombat is an educational game that teaches programming skills through interactive coding challenges. Players write code to control their characters and solve puzzles, learning programming concepts such as variables, loops, and functions along the way [Kap12].
- **Classcraft**: Classcraft gamifies the classroom experience by turning learning into an epic adventure. Teachers can customize the game to align with their curriculum, providing incentives for participation and engagement while fostering a collaborative learning environment [Kap12].
- **Foldit**: Foldit is a puzzle video game that challenges players to solve complex protein-folding problems. Players manipulate 3D protein structures to achieve the most stable configurations, contributing to scientific research in biochemistry and drug design. The game's competitive elements, leaderboards, and collaborative gameplay motivate players to explore different strategies and solutions, leading to breakthroughs in scientific discovery [Kap12].

On a higher level of education, the paper "A Systematic Literature Review on Teaching and Learning Introductory Programming in Higher Education [MRF19]" has extensive research done for introductory programming courses at university level. They take into account teaching methods and why failure and dropout rates are remaining high.Their paper focuses on three research questions - what skills and background knowledge are important for beginners to learn programming, what difficulties do beginners face when

learning to program, and what challenges do teachers face when teaching introductory programming? The authors took 89 papers from 100 to analyze. The chosen papers had criteria they had to fill. They had to be in English, published in conferences or journals, were between 2010-2016 and had to be on them with the research topic.

The papers that were excluded were if they (i) did not address the research question, (ii) were too short, such as workshop papers, (iii) were published in local conferences and (iv) were written by the same research group with the same data [MRF19]. Through keyword searches they were able to find the material needed and started filtering through a selection process. The chosen papers, which were either field studies and theoretical papers, also went through a quality check.

The criteria for a good quality in a field study measured how well the data collection was formulated, how clear the links between the data and interpretations were carried out and how was the approach to the data analysis was. Meanwhile, the quality criteria for theoretical papers investigated how well the original aim was addressed through analysis, if the knowledge had shown growth throughout the research and if the content had a wide diversity perspective.

The paper goes on to also analyze the educators who are teaching introductory programming at these faculties. In one passage it is written by the authors that the teacher should be a motivator, not just a provider [Har15], and that not many teachers realize that it is also their responsibility to motivate the students and arouse the students' interests [Har15] [MRF19].

This is a passionate topic of mine, as I truly believe that the right teacher can lead an army of successful students. Throughout my years of studies, personally, I have performed much better in classes where the professor is more involved with students. Seeing how excited and motivated a professor is about their subject is the best positive mental attitude they can give to their students.

Especially when learning a subject that requires a lot of patience and reading. The initial years of our academic careers either make it or break it for our aspirations. Having educators who nurture and tend our motivations and passions is the reason why a not so successful student can become one of the best. The quote "be a motivator, not just a provider [Har15]" is a powerful statement.

There are teachers who would do the bare minimum of giving the study materials and expect his students to succeed on those alone. Education is a tool that needs to be wielded correctly and in a diverse manner for each student.

Educators as part of their job should be able to understand their students.Give effort to get them to engage in the material, help them create their own study guides and encourage them when needed. By the time most students are in their university years, they are already mentally drained. This is exactly why an interactive curricula can help them kick start their motivation for a successful first semester.

Common challenges for learners are problem-solving, staying motivated, and understanding programming syntax [MRF19]. A common theme to problem-solving is the divide and conquer method. When a student first starts to learn programming they usually struggle in breaking down problems into smaller parts.

This makes managing the tasks more challenging since they can not understand the logical sequence, hence also struggling in debugging. It is very common for students to be frustrated after the initial excitement of learning a new concept has worn off. This affects their motivation to continue research and practice. A large part of practicing programming is learning the syntax and rules of programming. Novice programmers struggle to remember applying rules correctly, which often includes missing semicolons or typo mistakes.

There were papers that were chosen also mentioned the importance of speaking English. One paper even stated in the course they analyzed, fluent English speaking students were more likely than non-fluent ones to be successful [HC15]. This is mostly because English is used a syntax for programming languages. Unless, there is a student who is highly motivated and would be willing to learn specifics, it would be a struggle for them compared to someone who already has a good basis in the English language.

This also correlates to another point it is important to stress that these students also believe that mathematical dexterity is important as well as creativity, even though they state that programming is basically reasoning [ACA$^+$11]. This is related to how mature the students thoughts are, how seriously they are taking their education and the importance of being in a renowned university. The visual programming language that has had the most success is Scratch.

Developed by a Group at the MIT Media Lab, this platform has brought together communities of children to program together. The whole concept includes putting together coding building blocks to make characters come to life. Using their own imagination, the children can create story-driven games and still learn how to program.

They have had such big success that research has explored how Scratch can be integrated into various subjects, such as mathematics, science, and language arts, to enhance interdisciplinary learning [SB23]. Scratch has been widely recognized for its role in promoting computational thinking among elementary and primary school students.

Studies have shown that Scratch helps students develop skills such as abstraction, decomposition, and algorithmic thinking. Studies have reported positive learning outcomes, including increased engagement, motivation, and self-perceived competence among learners who use Scratch [SB23].

No matter the level of proficiency, when I child is done with their project they can upload it unto a community page that can be accessed by other peers. These projects can then be downloaded by others and they can remix. This means they can further develop what the other child did and add even more elements and events to the game.

Through this type of progression the project can become a fully functional game. In addition, if they do not want to use another peers game, they can see the code and re-create it in their own game. Each child has their account where their games are linked to their page, where they can see their progress of learning.

3 Gamification Basics

To grasp how the concept of video games can further aid education into a more advanced era, we need to understand the basics of gamification. What is the motivation behind gamification? Are we trying to get users to aimlessly be addicted to games? Is there any result? Well, gamification aims to increase user engagement. This encourages them to spend more time completing tasks and challenges without noticing the time passing.

According to Karl Kapp gamification is a careful and considered application of game thinking to solving problems and encourage learning using all the elements of games that are appropriate [Kap12]. The motivation is creating fun and addicting elements from game mechanics and applying them to real-world applications.

Gamification is the use of technologies engaged in promoting intrinsic motivations by using diverse characteristics of games in other domains outside the entertainment industry, such as education, marketing, public administration, politics, and health. It is an emerging trend derived from the huge popularity of games and their intrinsic ability for call to action to solve problems or enable learning in different fields and in people's lives [BPLO18].

As you can see, you can add gamification to every aspect that fits our lives. These two definitions are talking about using the same system of turning entertainment into education although they all serve different purposes. The gamification I want to focus on has everything to do with video games.

In other words, take programming and make it gamified. With the same building blocks or component that are used in video games, I can create an educational platform that will not feel like studying. With the power of gamification, any lesson can turn into a competition with yourself or your peers.

Game mechanics are designed to increase user engagement for games that do not necessarily have anything to do with gaming. They enhance the gaming experience for users.

These are the elements that can fit every kind of user personality depending what they aim to do in a game. Each of these items have their own game elements that help with engagement, motivation, and loyalty.

- **Points and Rewards**
- **Challenges and Quests**
- **Feedback**
- **Leaderboards**
- **Storytelling and Narrative**
- **Social Interaction**
- **Customization**
- **Unlockables**

Points and rewards cover badges and achievements that reward the player for completing a challenge. With points, the player can also level up, indicating progress or expertise. This type of reward system helps maintain motivation by providing tangible goals and recognition for achievements.

Challenges and quests are tasks of varying difficulty that players can complete. These challenges offer feedback through the accomplishment of tasks, promoting a sense of progression and achievement as players overcome obstacles.

Feedback can be immediate, offering instant gratification and hints for improvement. It can also be visually represented through progress bars, displaying progress in percentage or indicating the remaining tasks. Immediate feedback helps players adjust their strategies and stay engaged.

Leaderboards contribute to ranking and social comparisons. Rankings encourage players to compete against each other, boosting engagement. Social comparison motivates users to improve their standings on the leaderboard, fostering a competitive spirit.

Storytelling and narratives create context through themes and characters. These elements drive the story progression, keeping users engaged through a compelling narrative. Characters and plotlines enhance the immersive experience and emotional connection to the game.

Social interactions drive collaboration and competition. Players can encourage each other to complete tasks and work together toward a collective goal through collaboration. Competition adds excitement, as players vie against each other, enhancing the overall gaming experience.

Customization lets players create avatars and user profiles, personalizing their experience. This feature allows players to modify their characters to reflect their identity, increasing their emotional investment in the game.

Unlockables offer exclusive content such as rewards, including in-game skins, coins, chests, or unique items. These rewards provide incentives for continued play and exploration, adding an element of surprise and achievement.

Currently, the loyalty of a player in every type of game depends on the developers. Player loyalty is the commitment shown by players in regularly playing a game or participating in a gaming community over a period of time, indicating a long-term emotional connection [Yap19]. In other words, the longer a player is involved in a game, the better for the long-term success of a game. This could also depend on how many rewards and how much attention you are giving to a player.

There are new games coming out every year with similar mechanics and various game designs. Every gaming company has to make sure their player base does not leave them for another game. This is one of the reasons when the holidays come, players get a lot of "freebies" for their game.

A **freebie** is an item that is given free of charge. They come in the form of free character skins, experience points (XP) and even in-game currency. This assures the loyalty between the player and the game.

The best example of player loyalty is the game, Stardew Valley. The developer, Eric Barone, created the game by himself in four years. He taught himself the necessary skills to produce the game's music, art, programming, and design [Con25].

The game released in 2016 and throughout the years only bug-fixes have been added to the game. The game, simplistic in nature, has a story-rich narrative. It starts with the player inheriting a farm and starting a new life in the country side. He has to meet the town folks and finish tasks for them, all the while upgrading his own farm. Through-out the game the player will get closer to the towns people and learn their dark secrets.

There are so many mysteries that the player has to solve. When focusing on the main objectives, Stardew Valley is about 53 hours in length. If you're a gamer that strives to see all aspects of the game, you are likely to spend around 168 Hours to obtain 100 percent completion [How23].

Even though this game has been out for quite some time, there have been no major updates until 2024. The player base were so involved with these characters in the game, they have upheld the community for years. Creating mods to expand the map, added new characters, and even got others to join the game. With the average players being around 70,000 and the peak 236,614 players at the same time [Ste23].

Stardew Valley has several gamification elements that keep the users engaged, these include: player progression and goals, feedback loops, rewards and achievements, social interaction, randomness and discovery, customization and personalization and core loop. Most of the terms we are familiar with, although the term core loop for example is about planting, harvesting and selling crops. The cycle between planting and selling your crops keeps the players invested in their farms.

Feedback loops are systems where the output is fed back into the system as an input. And when it comes to game design, this typically means that a player's success or failure impacts the likelihood of future successes and failures [SYAN20]. There are positive and negative feedback loops.

A positive feedback loop feels like the game that just keeps giving. You are rewarded for doing well during the game with rewards that make you even better. It's like a party for the winners, giving them even more tools to dominate [SYAN20]. In Stardew Valley there is constant positive feedback loops.

If you finish a timed-challenge in time you get money or an item that you need to finish another challenge. Every time the player gets a higher friendship level with an NPC (Non-playable character) they get a gift in their mailbox the next day from that same NPC. Although, in the negative side of the feedback loop the consequences are brutal.

The essence of the game is to build an run-down community center. There are different bundles where you have to get items from each season and you have around 28 days to get each item. The negative effect of the feedback loop are more prominently shown if you miss one deadline, you have to wait a whole year in our case 84 in-game days to continue that quest.

Each season has its own challenges and failing to complete them results in lost progress and even better opportunities.This game has cracked the code of successful gamification and keeping their player base loyal.

The Self-Determination Theory is a prominent theory of human motivation, provides insights into the motivational effects underlying both successful and unsuccessful gamification and the resulting behavioral outcomes [RWJS18]. Gamification works based on a theory called Self-Determination Theory.For this theory to be successful in gamification there are three psychological needs that every player should have: competence, autonomy and relatedness.

There is a spectrum of different types of motivations, starting from amotivation, extrinsic motivation and intrinsic motivation. In amotivation, the player has no motivation. If a player does not understand the basics of a game, his needs for competence is unsatisfied which undermines his autonomy.

This causes him to lose his temper and start to thing the game is boring. Meanwhile, extrinsic motivation, motivation that you get from outside sources, is either external, introjected or identified. This can also be true for co-working or playing with your peers who will stimulate motivation and longer play time which in return raises learnability.

External Regulation: The learner responds to a concrete reward or punishment, but cannot appreciate the value of the learning activity or learning outcomes aside from that reward or punishment. For example, the medical student completes clerkship exercises for a gift card or to avoid expulsion from school, but has no concern about the actual topic or learning [RWJS18].

Introjected Regulation: The learner begins to internalize the value or regulatory process but refuses to build a personal identity from this. For example, the medical student is slightly more motivated and sees the value in completing the clerkship exercises but only because their friends are in the same study session or because the student wants to get a better score than a rival [RWJS18].

Identified Regulation: This refers to the internalization of the external regulation, which has finally become part of the learner's core self and identity. For example, the medical student is motivated to complete the learning exercises because doing so will likely improve their standardized test scores in the future for residency.

The hope for getting into a competitive residency program is aligned with gaining improved knowledge, and admittance into a residency program of the student's choice is important enough for the student to internalize the regulation as part of their identity [RWJS18].

Intrinsic motivation where the player has completed all the tasks, quests, side-challenges because he find true joy and fulfillment from playing the game. From the different types of motivations we want the players to have intrinsic motivation when playing the game. This is because a self-motivated player usually will play longer, learn more efficiently and effectively.

A good combination is a game that motivates you, challenges you and a player who has intrinsic motivation. The best game plays are those story-driven games where the player truly enjoys the journey without complex challenges that get the player stuck in one area for a long period of time. Gamification aims to create smooth transitions between the players type of motivation and their journey through the levels.

Selective, purposeful gamification that aligns with learning goals has the potential to increase learner motivation and engagement and, ultimately, learning. In line with self-determination theory, game design elements can be used to enhance learners' feelings of relatedness, autonomy, and competence to foster learners' intrinsic motivation [RWJS18].

There are three psychological needs of Self-Determination Theory that gamification assists in. **Competence** is the ability to visualize that a player that can beat the game. Goal setting is a great gamification principle since it gives the player that vision of achieving their goal. This sets a positive relationship challenging the player, achieving the goal and then setting a harder challenge and the cycle continues.Goals are theorized to improve performance through three specific mechanisms that can be incorporated into educational activities using game design elements.

First, difficult goals encourage higher expectations, which in turn increase performance

Second, goals from outside sources enhance self-efficacy, or one's belief that he or she is capable of accomplishing a task

Third, completion of a goal leads to a sense of competence, leading to higher satisfaction and spurring intrinsic motivation [RWJS18]

Autonomy has everything to do how a player decides how they want to learn the game. They take responsibility for their own learners control. Learner control is about the choices a player makes within the gamified system. If a game has a lot of achievements, the player can choose which ones to prioritize. This way, they can also figure out their strong and weak points through reflection of their actions while gaming.

In the long-run this creates a need for more optimized game play which in return gets the players more motivated. Most students in school do not perform well since there is a lot of deadline pressure. This has a negative effect on most students which can cause extreme anxiety and stress. Autonomy lets the student self-regulate their emotions depending what they think they can perform better first.

Relatedness is the relation a player feels to another player, mentor, teacher who are active in feedback and discussions during the learning process. This includes being in a supportive environment with a safe surrounding for failure. A safe learning environment offers opportunities for inquiry, reflection, and feedback-seeking behavior [SEC13]. Almost every point or theory that I have written about somehow has a relation to engagement.

Either social engagement where the players are socializing through co-operation or tournaments. Loyal engagement, where the player is engaged with a game and has a trusting relationship where they will not abandon a game for an up-coming one. Self engagement, where the player truly enjoys what they play or learn and have set a certain goal for themselves to reach in order to gain more confidence.

Poorly applied game design elements, however, may undermine these basic psychological needs by the overjustification effect or through negative effects of competition. Educators must, therefore, clearly understand the benefits and pitfalls of gamification in curricular design, take a thoughtful approach when integrating game design elements, and consider the types of learners and overarching learning objectives [RWJS18]. If you do not apply gamification properly can lead to overjustification and the negative effects of competition.

Overjustification effect occurs when you offer too many rewards or incentives to a player who was already interested in completing the task. This can cause them to change their motivation from enjoying the game to now playing it for a complete different reason. That is, adding game design elements to increase extrinsic motivation can have an adverse impact on learners who already started with a strong intrinsic motivation.

Overreliance on external sources of motivation tends to remove any learner control and thereby adversely affects autonomy [RWJS18]. This would remove the self-control they have to choose how they play their game.If baseline interest is high—meaning the activity itself captures and engages participants—then adding extra rewards leads to overjustification and loss of intrinsic motivation.

Essentially, the goal of the game conflicts with the learning goals [RWJS18]. Self-Determination Theory claims that if a player has too much stress from a competitive game, they will perform even more poorly. In the **negative effects of competition** the player will lack the self-competence to finish their challenges in time or even at an optimized manner.

When the competition is synchronous or public and the lack of achievement is prominently displayed, relatedness—particularly to learner peers—can also suffer [RWJS18]. Competitions can be frustrating which will increase the lack of interest of the players. They will start to feel inferior to the tasks which will affect the learning outcomes. This goes the same for those who feel unchallenged. The task being too easy cause the player to be uninterested and put no effort into finishing it.

4 Codonia: The Game

Reading research papers that talk about how to teach programming can be a fascinating topic. The researchers often deep dive into topics that have to do with development.

Algorithmic Development is where the focus is creating new algorithms to solve problems in a more efficient way. Developing new programming languages or improving the ones that exist. Creating a more structured and secure language that is also more powerful. Researchers also often talk about Artificial Intelligence.

Artificial Intelligence who concentrate on developing new models and techniques. Finding a way to make Artificial Intelligence more human-like. Studying how the users interact with the software so its more effective and intuitive with its answers.

These topics are being also studied through different methods. Conducting experiments and conducting case studies. Testing new approaches to validate their effectiveness. Collaborating with other experts in fields like linguistics to innovate a new programming language foundation. The question arises how does researching "how to teach programming?" differentiate from how we actually teach programming in schools?

Looking into the current education of programming, the lessons depend on each institute and their program. There is a debate whether to let students only learn through projects and research their own material or give them a book. This would include reading chapters about the basics of programming and having them to understand the concept before attempting a project. As much as some would argue that practice makes perfect, I would argue that you need a balance between books and skills. There are continuous methods who have been implemented into universities to teach programming to their students. The best example would be active learning, where students learn pair programming techniques. The task includes working together on a coding task and implementing peer instructions, where they explain these concepts to one another.

Another interesting idea is a flipped classroom. This was a method that was also taught during my high school years in the United States. The students take material and learn it at home, while in the classroom they create hands-on activities and have on-going discussions about the lesson. This enabled the students to collaborate and be more active in the classroom. Having prior knowledge to a topic leads to more confident students who are eager to learn more. Now, more recently gamification has entered the curriculum. Coding challenges, leader boards and badges that show the completion percentage of each student.

The paper "Learning and Teaching Programming: A Review and Discussion" by Anthony Robins, Janet Rountree and Nathan Rountree, talks about this exact point. The education and psychological aspect of programming. When you take an expert programmer and put them next to a beginner, how different is their way of thinking? What makes the expert programmer an expert? Novice programmers struggle when it comes to understanding basic programming concepts. One experiment from the paper says that they "[..]compared novice and expert programmers as they worked to understand programs which were either conventionally or unusually structured. Subjects could view the code one line at a time, and a "run" was defined as a sequential pass over a section of code. [RRR03]." So they put two different levels of skill or rather knowledge into the same test to see how they think. It would make sense for the expert programmer to have a better idea what they need to do in certain situations since they have already, in most cases, solved something similar.

Throughout the years, a programmer earns the title an expert through their experience. No amount of school can teach you what you learn from practice. Which is why its important to teach a beginner how to think like an expert while also letting them come up with their own creative solutions. This would give them a head start when seeing complicated problems and have the confidence to solve them.

The results of the experiment were a bit predictable, as far as my opinion goes, since when it came to experts they "[..] tended to read conventional programs in long but infrequent runs, and read unusual programs in short frequent runs [RRR03]." As far as I understood, each program that was familiar to them were run after coming up with a solution. This mean they knew what they had to do and expected a specific result from their coding. Meanwhile, the less familiar problems were run in experiment phases.

They would write a solution, test it out, continue to tweak and write more code until its solved. This was not the same as the novice programmers,meanwhile the novice programmers would read conventional and unconventional programs the same way [RRR03].

Each problem to them was new, unrehearsed, unpracticed. So their go to solutions would either be working in smaller parts, understanding what the outcome is and continuing to solve it. This is the most efficient way, from my perspective, since they are trying out different ideas and seeing real-time results. Another idea would be trying to solve the whole problem in one go. Writing the code for the solution and running it circles until you de-bug the errors. This is not something that is being taught to be practiced. It makes the solution time longer, there is no telling exactly where the problem is so you have to solely rely on output prints and line-breaks.

This is the kind of high level of thinking that needs to be explained to anyone who is new to the field of programming. Especially to children, who are similar to sponges when it comes to gathering knowledge. The authors suggest that experts (even without relevant knowledge structures or plans) had more flexible strategies, and were better able to recognize and respond to novel situations [RRR03]. This would also suggest that the mindset we use while programming should be the opposite to the rules of programming. Even though, when it comes to syntax, definitions and rules of programming there is a strict guideline that needs to be followed, meanwhile problem-solving and tackling a challenge should be flexible.

Our ideas and innovations do not fit into one box, where you create one blueprint and use it repeatedly for every problem. What makes an expert programmer a good programmer is thinking outside the box. Coming up with creative solutions for setbacks, issues or tasks.

To create a lesson plan that will be relevant to children you need to understand their psychology. Jamie Madigan, an expert on the psychology of video games, spoke about this in his book The Psychology of Video Game Engagement and recently talked about it during a lecture. ‘’[..] adjusting the difficulty of video games is crucial in maintaining player engagement and motivation, particularly for children. As players progress, increasing the challenge helps keep their interest and encourages continuous learning and skill development [Uni24].” The lesson plan needs to be thoroughly written for each step of the difficulty level. It should progress naturally, without feeling like they took a big leap

from learning how to walk. The levels that were played in the demo were simple, easy to understand although they would not uphold in the long run. Because this was supposed to be an introduction, easy concepts were chosen. If I am to continue teaching, I have to define more creative levels.

To create a successful lesson plan, I had to create a reward system. These are the six steps that I am planning to continue implementing into the levels as they progress to harder terms and definitions.

1. **Clear learning objective**: Define skills that the game will teach: variables, loops, conditionals, problem-solving techniques.
2. **Progression System**: A structured progression system where players advance through levels.
3. **Immediate Feedback**: Provide immediate feedback to players' actions to encourage learning and experimentation.
4. **Long-term Feedback**: Have players reflect on their performance and identify areas for improvement.
5. **Social Interaction**: Integrate social features that allow players to collaborate, compete, and share progress with friends.
6. **Rewards and Recognition**: Badges, virtual currencies for tasks, achieving mile-stones, or mastering programming skills.

Each of these steps are counted as mile-stones as to how much the game has progressed itself. Not only will I continue to test the growth of students through future demos, I am also testing out the game ability to keep their attention for a longer period of time. Clear learning objectives are the basis of our learning step. We can imagine an empty Lego plate that will be used to build upon. The learning objectives have to be defined in a simple manner, without adding any unnecessary steps.

Writing a curriculum sounds easy since you are organizing the teaching material, although this can get complex real quick. There is a fine line between explaining and over-explaining and most of the time we tend to over-simplify objectives and creating too many sub-divisions.

Progression systems are mark for the player to see how far they have gone in their quest. I have decided to make this into a levels system. In the main menu, there is a levels menu that takes the player into their locked and unlocked levels. Those levels that are solved, vivid in color, are open and can be re-played as many times as they wish. The locked levels, which are grayed out, have no type of interaction attached to them since the player has not reached it yet. This also encourages them to finish the levels, since there have also been studies that talk about "achievement hunters." These are players who play games solely to one hundred percent the game or gather every single achievement in the game.

A known study has been conducted for Xbox360 achievements and talks about human behavior connected to it. "A gamer with the alias Shipwreck says, "I'm not comparing my score against other people's... It's just a way to track what I've played. Like, that's the best part about the whole achievement system" [CAG10]."The gaming community has changed drastically since players were accountable for their progress. To this day, Steam sends out "Steam end of the Year Review/Recap" which shows your statistics for the whole year. A report about that year's gaming history which includes which games you have played, the genre of the game, how long you have spent in each game and the amount of achievements you have received from those games.

Immediate feedback , which you can see in Figure 4.1 is immensely highlighted in Codonia. For each right or wrong answer you get an explanation. I believe this has a double effect since even if the child loses focus after a while and starts to just click around to pass the levels, their attention might get caught by an explanation. The pop-ups that show up are in bright green or bright red depending on the situation. Designing them this way created an alarm system that flashes in the screen, automatically diverting their attention to the words on the screen.

Immediate consequences feedback returns results or responses as a direct output of the users actions within a game. This consequence can be negative or positive depending on the event that was triggered. These feedbacks can range from visual cues and audio feedbacks to score indicators and haptic feedback.

Haptic Feedback Vibrations or tactile response from a game controller that provides a physical sensation according to the actions in-game.

A more compelling version of immediate feedback is feedback through consequence. As seen in Figure 4.1 the consequence follows after a wrong answer is pressed. This is one of the levels where I talk about in the Game Design section. In this level, there are trolls that are waiting for a command. The command, in our case is answering the questions through a quiz, will tell the troll to either not react or destroy a tower. This is the kind of feedback that makes it thrilling for kids to remember the right answer by repeating the level every time they fail answering the questions correctly, in our case by memorization. Memorization is not recommended in programming.

Figure 4.1: Immediate feedback through consequence

You cannot create a program through memorizing functions to make a software or product. Although, in our level this works wonderfully. The reason why is because they are not learning how to program, they are memorizing terms. The statement on the screen says "GDScript is a scripting language for what?" with the answers being "Godot Engine" , "Unity", "Unreal Engine." By playing this level, they have the right to only make three mistakes, since there are only three towers.

Each wrong answer, a tower collapses. Once they are all gone, the level will reset. Each time the level resets, they get the same statements and this way they memorize the statement and answer.

Long-term feedback is the opposite of immediate feedback. They do not get to see their behavior right after playing a level. This is data that is accumulated throughout the game and would be shown in different scenarios. The first scenario would be after the player has finished a chapter which would be equal to one lesson plan, around ten levels for each lesson plan, they get a detailed summary on their performance.

The words they missed the most, the objectives where they made the most mistakes and a suggestions on which levels they can replay. I believe the children will be able to play and focus in a more calming manner knowing in the end they get feedback that will help them in the future. It takes away one less worry of trying to remember the check-points of their less highly performed levels.

Leader boards are a great way to create social interaction. Between creating a competitive spirit and friendly rivals between players, the interactivity boosts. Although, there is still research to be done as how to make it friendly for everyone since a study shows the researchers learned that leader boards make highly competitive users engage and spend more time on the task, while low competitive users are driven away by the use of leader boards [NH23]. So, even if leader boards are highly popular, it is still a hit and miss.

There is always the option to not want to participate in competitive game play and only enjoy the material you learn. This would also take out the collaboration with friends, since sharing your progress with others and helping each other in the form of cooperative games is a big part of social interaction.

In our case, I believe having quizzes or time-sensitive themed levels would be beneficial. As long as the early challenges have a lot to do with matching definitions and creating instructions, as to not make them too stressful for beginners, it would be a healthy competition for the players. You still want them to be out of their comfort zone trying new methods of learning.

For those who do not like quick paced challenges, the option to get in-game rewards is always available. For every milestone passed, the player earns a badge. Badges can range from the lowest achievement, which would be following instructions correctly the first time you play the game, to the most complex. Leading into using gamification for education, I have created a video game that strives to have a balance between learning and playing.

The video game is created in Godot using GDScript. It is intended to be a mobile game, although it can also be played on a laptop with a touch screen. The design is simple, which ensures children to have a clear understanding of what each elements purpose is. The levels are created in-between cut-scenes and dialogue scenes. Before they start to play the game, there had to be a story-line that would get them interested in continuing.

Giving them a purpose to continue playing, I created the story of a kingdom called Codonia. Codonia was ruled by their queen, Ada, who had been captured by an evil villain, Nullify. The children are introduced to their objective early in the level so they get a basis of what their mission is. While working on the demo, I paid more attention to the different types of elements I can add to each level so it wont feel repetitive.

Although, the lesson was repetitive and they had to learn about the same topic throughout the five levels, they did not seem bored due to the level design. Each level introduced a new way to play, so they were forced to pay attention. In the beginning of each level, there is a dialogue.

The dialogue would explain what they need to do including a piece of information about the characters, as seen in Figure 4.2 where the villain of the game introduces himself to give motive.

Figure 4.2: Introduction to the villain

Once the dialogue was over, they could not go back and re-read the instructions. This was told to the children before they started playing the demo. For the finalized version of the game they will be able to re-read conversations without a limit.

Since I had limited time to prove why my thesis could work, the experiment had to be harder. The children only had one shot in understanding the concept, solving the problems and re-iterating what they had learned. This realization would set for those children who would try to skip a conversation and get perplexed. Next level they performed better.

The game is in 2.5D and some of the levels even being top-down. A 2.5D game is a two-and-a-half dimensional game. This design technique is used to add a bit of depth to a video game. While your character can move through a 3D environment, their only options are going up, down, left, right. The environment is set in layers, so each object in the game has its own position in the world. The game is in different types of islands that is surrounded by water. Starting from the background, which is a blue layer, I was able to create the perception of water in the game. Layer two involved the first type of terrain, which was sand.

TileMap in Godot is a grid of tiles that lets the user create a layout. You can "paint" the tiles onto a grid, which makes it faster to set a foundation, rather than putting tiles one by one.

This makes it easy if you have a large canvas to work with. Each TileMap has to be individually created. This can mean either importing a custom made by the user or downloaded from another creator. A big role plays the canvas size of layout.

I used a horizontal mobile game layout which set my size to 1920x1080 which is FULLHD, 2,073,600 pixels in total. Each tile in the game scene was 64x64, which let me have a high resolution game.

Figure 4.3: Multi-layer concept of 2.5D Game

Codonia has three different layers of tiles. In Figure 4.3 you can see water,sand and rocks. To create an illusion of a 3D game, on top of the water there are individual islands of sand. The sand is the foundation for the real platform that the players will use. The third tile, which was accessible, is the ground. The stacking of different layers gave the perception of depth. I wanted the players to have the feeling that they were in a remote island.

Although, to be able to have the player move around the map, I had to create smaller islands that connect with bridges. One of the levels includes getting answers correct in a quiz to be able to access the bridges. An improvement to my current layout would be to have a complete map layout of all the islands together. This would cut down time on level design since you get to see progress as to where you are on the level map. In addition to make it more real I used environment animations, which I will be talking about later on.

Even though there is a sense of freedom when playing the game, it can not be completed without finishing the objective. A good example is one of the Algorithm levels where character is stuck in a forest with a lot of tress, with no road in sight.

They have to depend on a list of step by step instructions they found on a sign that give the way out. The instructions clearly state how many step they need to take toward a direction to be able to navigate the forest. This sounds interesting until you have to figure out the functionality of the road. To be able to have the character move, you need buttons that will make him move around 100 pixels per step.

This could also mean if a child makes the wrong move, the main character will be off road or off screen. This is why I have added collisions to the TileSet in the TileMap node. In Godot, to be able to have collisions you have to have a Physics Layer.

Physics Layer manages and organizes the layers that are used in collision detection. It lets the game creators layer objects and specify which objects can collide others. Table 4.1 can give a clearer insight with an example which objects collide with what layer.

In order to make the Physics Layer work, you have to understand how collision works in Godot. Everything works in layers, including the characters and objects. If my ground layer is above my character, I will not be able to see my character. This is why each object that is entered on the scene has a hierarchy. For the collision detection to work properly you have choose the layer the body of the character is in.

You name that layer the 'player' layer and continue to name different objects such as 'enemy', 'collectible', 'environment'. The second step to it, is the Collision Masking. We set up the object that can collide and now we set up objects that detect that collision. Best way to describe this is by an example.

Object	Layer	Mask
Player	PlayerLayer	EnemyLayer, PlatformLayer
Enemies	EnemyLayer	PlayerLayer, PlatformLayer, BulletLayer
Platforms	PlatformLayer	PlayerLayer, EnemyLayer
Bullets	BulletLayer	EnemyLayer

Table 4.1: Collision Layers and Masks

As you can see from the Table 4.1, if an object, in our case the player, has the PlayerLayer, he can collide with a EnemyLayer and PlatformLayer. If the game is a shooter, which in most cases games with enemies are, the EnemyLayer can additionally collide with a BulletLayer. This also applies to environments, in our case the TileSets. In Codonia we use give the TileSets a PlatformLayer so the character can not go out of bounds.

The challenge with Codonias' layout is the spacing. Our main character is 64x64 which means the roads have to be twice as wide in each direction so the he does not get stuck. In the Physics Layer you can set custom masking layers, seen in Figure 4.4. This means if a tile is only half road, you set the other half as a masking layer so the character does not go out of that bound. If the roads are too close to each other, it does not give enough space to the players collision to be able to move freely. It would constantly get stuck in-between collisions and that makes it a frustrating level to play.

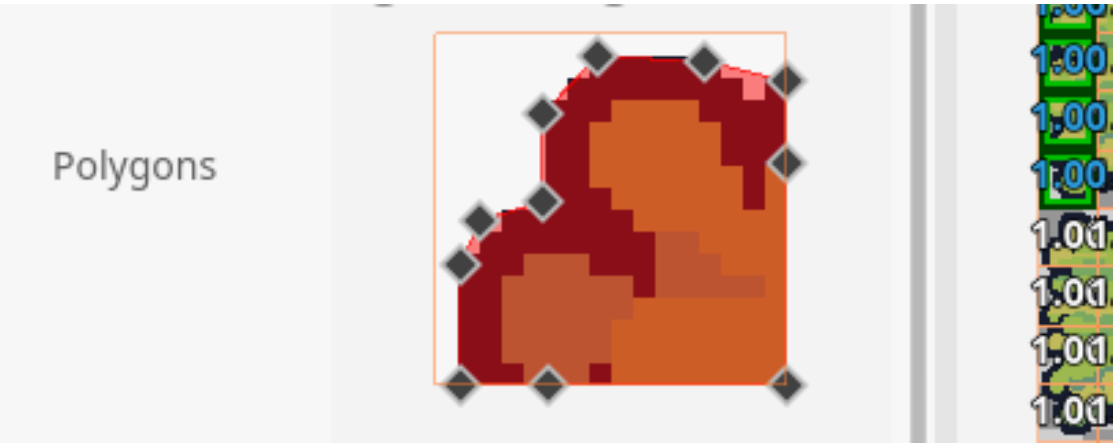

Figure 4.4: Setting the custom boundaries in the PhysicsLayer

Godot has four collision object types: Area2D, RigidBody2D, StaticBody2D and CharacterBody2D. In order to create an object that works correctly you need to know what type of object you are working with. Each of the objects are being used in Codonia, serving their own purpose.

Area2D nodes provide detection and influence. They can detect when objects overlap and can emit signals when bodies enter or exit. The node can also be used to override physics properties, such as gravity or damping, in a defined area [Eng24] this is used throughout the levels to change scenes. The easiest detection you can have is one that triggers an event.

RigidBody2D nodes implement simulated 2D physics. You do not control a RigidBody2D directly, but instead you apply forces to it (gravity, impulses, etc.) and the physics engine calculates the resulting movement [Eng24] are used in controlling the main character. Since we are giving him control to move around the map we need to apply force for him to move.

CharacterBody2D can be used in levels where collision detection is needed but physics do not play much. This lets you create the movement through scripting. This lets you move your character through pixels. For example you can make your main player go in each direction 100 pixels each time. This is useful when you want to create a player that is controlled by amount of movements he makes. Given a list of instructions you need to move the player five times to the right and three times down. The level design needs to have a road with a fixed length that will only fit these parameters.

StaticBody2D node, a static body is one that is not moved by the physics engine. It participates in collision detection, but does not move in response to the collision. They are most often used for objects that are part of the environment or that do not need to have any dynamic behavior [Eng24]. This is attached to all objects that the main player has to interact with without giving it a force to move. If the user needs to collect an item, they will add this property so the item disappears after touching to collect.

Animation node,you can animate anything available in the Inspector, such as Node transforms, sprites, UI elements, particles, visibility and color of materials, and so on. You can also modify values of script variables and even call functions [Eng24].

Animations can be added to object to make it a simple or complex animations. The simplest animations include making an object transform in position, size, or rotation. In my game, I made sure to add waves to the sides of my Tiles to give it the look of moving water. The sand layer of the Tile had white waves moving forward and backward. This is possible through 2D sprite animation. If you download a sprite, it would either come in individual images or as one big image. These images look similar but have small differences in-between creating an animation. The images, otherwise called animation frames, can be put into a AnimatedSprite2D node, which had multiple textures as animation frames, which can be shown in Figure 4.5. Once you press play you will see the images creating a movement, similar to a stop-motion method, although with a more seamless blend.

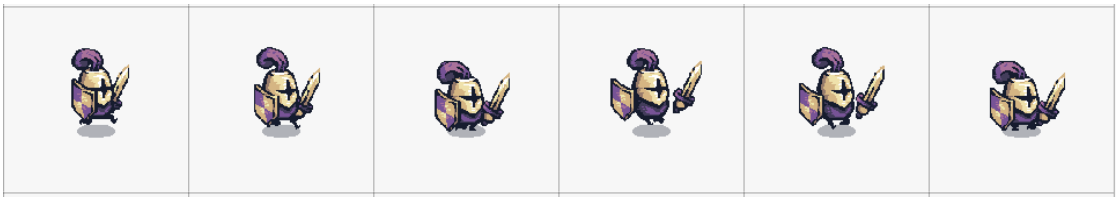

Figure 4.5: Walking knight animated sprite panel

Each animation in the game is made through an AnimatedSprite2D node. The sprites used are by an author named PixelFrog [Pix24]. The asset pack contains sprite packs for enemies, players, NPC, objects and decorations. Each of these sprites have enough images that when you put them together, you get a moving object. The AnimatedSprite2D node takes these images and put them in order. You then have the option to let the animation play from the beginning of the scene once or in loop. In Codonia, I have used a piece of meat that flips constantly as an indication that this is where the players end goal is. The second option for sprites is triggering it through an action.

In addition to the water, the characters and the environment around has been animated. Each scene has a set up with the first three TileMap layers, a ruined castle or tower and decorations. The characters are the most in-depth complex animations. Since there are multiple characters in the game, each of them have a purpose and an animation that defines their personality. The enemies, which are a pack of trolls, have the animation to hit with a stick that is on fire. The animation is eight frames long and it shows the perfect loop of raising their arm, swinging and then bringing it up again. To make it more realistic, their body also moved forward and backward as is to show that they are gathering force before hitting an object.Putting their foot up and down to mimic their body moving. This recurrence is often paired with a ruined tower underneath a fire animation.

The fire animation contains six sprites, images, and it shows a fire building from the bottom and going to the top so it looks like a constant wave. This combination of objects is shown as a result of something that can not be fixed.

The second version of the ruined towers with the enemy hitting it, is shown in a quiz game. The scene has perfectly build towers with an enemy,a troll, next to it. The player needs to answer correctly so the troll does not destroy it. If the answer is wrong, the troll will begin its hitting animation and the tower will transform into a ruined tower. This common theme is shown throughout the game to have a sense of consequence.

There is a clear line between punishing a player for doing something wrong in a game and letting them know there is a consequence to getting an answer wrong. In most games where perfection is required, the game will restart the level if even the smallest mistake is made. This are the time-sensitive case games, where even 0.01 second is between being first place or third place in the leader board.

Educational games should not have that kind of outcome since we want to motivate children to get involved, nevertheless, they need to know that there are repercussions.

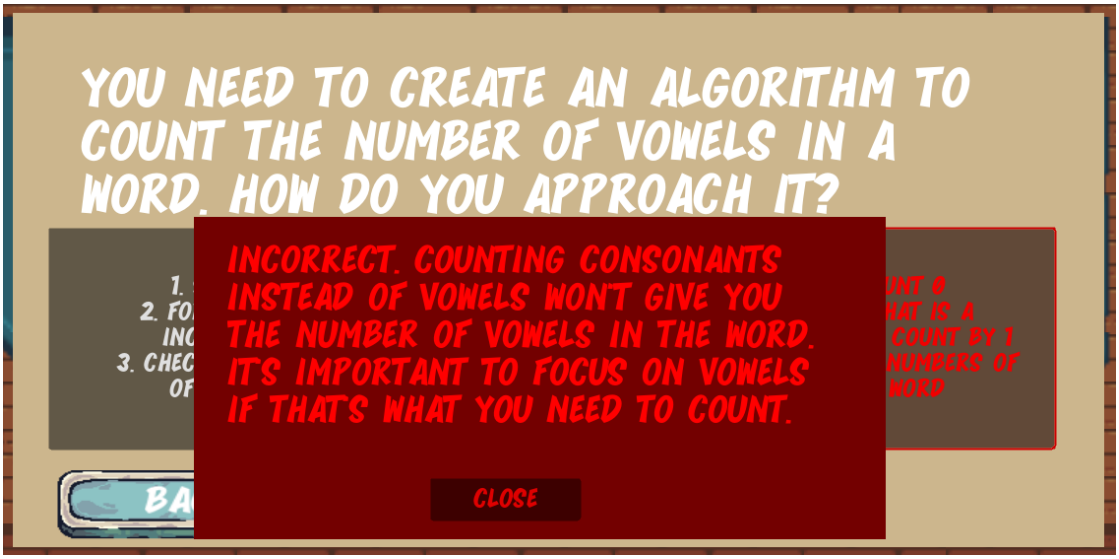

Figure 4.6: Showing the explanation for a wrongly chosen answer

A seen in Figure 4.6 this is part of a feedback with a negative review. It shows the player that that they can not guess answers while trying to finish the level. It gives them a sense of responsibility for the towers. Through narrative I am trying to connect the players to the characters on the screen. If something were to happen to the towers or a character in the game, I want the player to understand the consequences of their actions. This in a way forces the players to take the game seriously in order to not concurrently restart the level. This negative feedback also need to have a positive effect on their psyche.

As we already discussed, in gamification positive reinforcement is crucial. The psychology of every player is different although the main component that makes them keep playing is the thought of finishing the game. One way to make sure that the students who are playing these educational games do not get discouraged enough to leave the game, is to give them explanations.

Even if they choose a wrong answer, destroy a tower or cause a negative impact, there needs to be an explanation as to why the answer is wrong. This gives them purpose to continue since now they understand what they did wrong and next time will focus on how to fix it.

If we take restarting level as an example, we can use the same game mechanic but in a positive manner. If they get three answers wrong they get to try again. In a positive connotation they feel as though they did not fail but as though they are getting a second or third chance to have a perfect score. The already ruined towers serve as a sense of purpose. The players also have the chance to fix the towers, as a way to feel like they are progressing in the story. The more questions they get right and continue to play the game, they get to re-build the town. Areas that were already destroyed become focal points to fix by becoming stronger in the game, meanwhile more knowledgeable in real life.

The anti-effect of building up the towers is also letting them get destroyed by the enemy. In some levels, throughout the story, the main character has arrived to areas that are yet to be destroyed. The level is populated by enemies that are waiting for a command to ruin the objects. This command is solely depended on the player. If the player can figure out a way to communicate with the enemy through GDScript and tell them to not go through with the action.

This puts responsibility on the player, thinking through decisions before they give an answer. Each wrong answer gets rid of a tower, a right answer leaves the tower as it is. I find these levels captivating because if you put an adult in a similar situation with much harder questions, they will have the same reaction. It is in the human nature to be nervous to not make mistakes. We are programmed to get everything right on the first try and that makes these kinds of levels thrilling.

There are four main character animations. Each animation is walking in four directions. The animations are created separately and called in GDScript.

In order to create these, you need to have all sides of the character shown walking those directions. Each animation is created in 6 frames. The more frames you have, the seamless your transitions are. I had already chosen the game style that I though fit best a bright and vibrant world. Those are the TileSet and decorations I chose.

Although, the main character and enemies also came in the same pack. The pack includes a knight and his movements include a sword being swung when he walks around. Even though, that was not in my initial plan, once I created the animated sprite it turned out hilarious.

He can walk left and right normally with his sword next to him, although the up and down animations have a step and a swing incorporated into his routine. So whenever he walks, he looks like he is chopping trees or even fighting air. I could have found another similar sprite, although this added a lot of personality into our main player. When you look into the story-line, he went from a village into a portal which transported him into a kingdom realm.

Realistically, if anyone would transform into a suit of armor they would walk around swinging their sword around. In addition, this is a children's game and it made the walking a lot more charming since it looks like the character is wobbling around the screen trying to carry the weight of the sword.

Decorations are non-moving 2D objects. They are usually rocks, flowers, mushrooms, trees, signs, anything that will add liveliness to an environment. For example, one of the levels has the player in a forest. The forest contains a lot of trees and mushrooms. To add suspense, I added skulls and bones near signs where they should not go to. This add a touch of metaphorical dimension to e theoretical solution. Which means, it enriches the environment for a better story-driven game. Why are the skulls in that position? Why does the sign have an "X" across it? It creates a mystery that lingers in the back of the players mind that would be solved later on.

I used decorations throughout my levels at the same time making sure the game file was not too high. Everything you add to the scene adds to the game file size, sometimes making the output outrages for simple game design. Animated decorations, such as trees, were my favorite to use in the game. Creating a forest filled with trees that are moving back and forth made the game so interesting to look at.

Trying to solve a problem that is complex requires techniques of dividing and conquering. Algorithms create well-defined instructions that guide a user to finish the task in a timely and effective manner. These set of instructions can be applied to a video game when asking a player to finish an action, trigger an event or even finish a puzzle.

The goal is to create simple yet well formulated instructions that lead the player to solve one step after another, while getting familiar with the game environment, interactive user interface and following instructions. Algorithms in Codonia are taught by mentioning the term, giving part of a definition and then teaching the rest through game play.

They are introduced to each phase of what an algorithm is, how it works, what is right and wrong with individual tasks. What is more important is to make the children feel superior when they are learning.

If they do not understand the concept of the lesson but they enjoy the game, then the game is not fulfilling its purpose. There are many games that are for entertainment purposes without making it educational. This is the epidemic that we have today, where children spend hours on mindless games with little to no positive impact to their knowledge base. Even though these games are captivating there is no real lesson to be learned. In the end, these games are for leisurely relaxation for a short period of time.

Figure 4.7: Following step-by-step instructions

The player learns about algorithms as seen in Figure 4.7 through tasks such as navigating mazes and sequencing, following instructions, and choosing the right way to construct an algorithm through a quiz. Loops enable execution of well-structured repetitive tasks which we call code blocks. Granting the player to automate a process, optimize an algorithm or even manipulate certain data. The demographic for Codonia are ages 7 to 14. This is a perfect age for children who can start early with understanding programming, which would be ages 7 to 10, and children who are already interested in video games and want to learn more, which would be ages 11 to 14.

This is the perfect age to get into programming. School-age children have currently used mobile phones and laptops to find important information for supporting the learning process or for the sake of entertainment such as playing games. However, most children only know how to use it without knowing how the program runs the devices inside [SYAN20].

This paper also talks about how advanced our children are with their technical skills as seen in Figure 4.8, yet they lack the theoretical skills and practice of educational technology. They can use the newest devices and technologies, learn how to make videos, edit images, create animations for the fun of it. They are literally born in the technology era, where new advances are shown every single year. The technology companies are making more advanced devices each year with new features. This is their future. The children are born into this, they grow up with this and only a handful actually start to learn its basics from a young age. Schools have started adding computer programming from an earlier age, starting from pre-school.

Figure 4.8: Student playing Codonia Demo on mobile

The level for early age is at level of 0 (Age range of 2-7 years including drag-and-drop or simpler, teaching planning/sequence only, & requiring no abstraction) and Level 1 (Age range of 5-10 years, including drag-and-drop, & requiring no abstraction) [SYAN20]. In a school program, that is talked in this paper, children start from the age of 2 to interact with programming concepts. There are levels where they learn from and continue to go

higher each year as they get older and graduate from one level to the next. This sounds like a great concept, since they are growing up in parallel with programming concepts that students in their high school or college learn about. Having a strong basis of programming will give them the upper hand through their professional academic career.

Psychomotor learning includes evolving their physical skills and ability to coordinate cognitive functions. The skill that you learn from playing games can later help in real-life tasks that require, agility, quick reflexes and precision.

There are the key aspects that focus on psychomotor learning.

Fine Motor Skills: These skills involve the precise movements of small muscles, such as those in the fingers and hands, which are often enhanced by activities requiring dexterity and control, like using a game controller.

Gross Motor Skills: These skills involve larger movements of the body's muscles, which can be developed through motion-based games that require physical activity.

Hand-Eye Coordination: This is the ability to use the eyes to guide hand movements effectively. Video games often require players to react quickly to visual stimuli, improving their coordination between what they see and how they respond with their hands.

Reaction Time: Video games can help improve the speed at which a person reacts to visual and auditory signals, enhancing overall responsiveness.

They already have a good foundation and it lets each child figure out if that is something they want to continue learning to turn into a career. Whether they would continue learning programming as they get older or not, they still get passive skills. There are benefits to learning how to solve problems, no matter how simple or complex they are. You learn the process of organization, optimization and how to divide and conquer.

From another research paper, it was stated that coding not only improved children's ability to solve problems, also improved executive functions (spending time for planning, solving standardized planning tasks, and inhibiting prepotent responses). In addition, the introduction of coding from an early age can support communication, collaboration, and creativity in the classroom [BGGAT19].

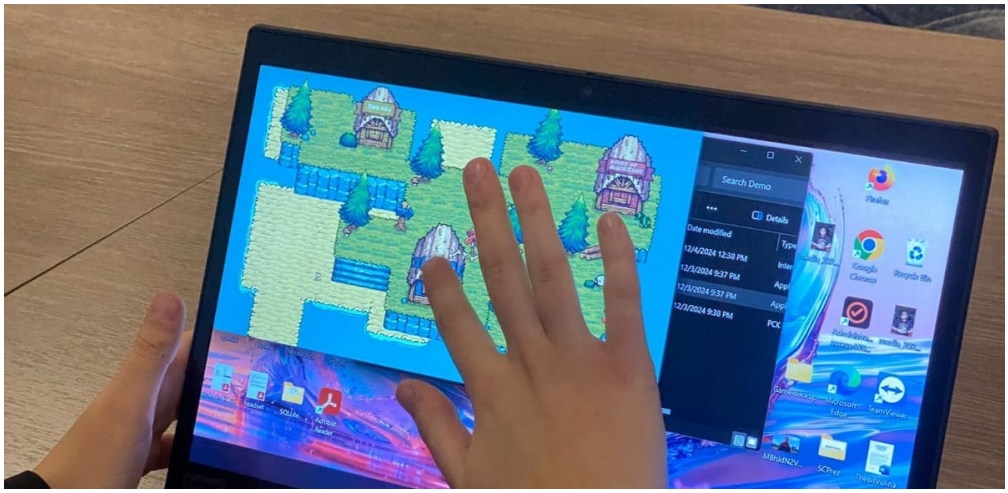

Figure 4.9: Student playing Codonia Demo on touchscreen laptop

Figure 4.9 shows the versatility of playing Codonia through different technology. The importance of collaboration is a life skill that will be used through group projects throughout their educational career. The children get to experience creativity, error-handling and bugs in a safe space with that encourages making mistakes to learn from them.

This is beneficial to their psychology since we are creating a safe environment where they understand to learn is to fail as many times as possible until you succeed. Codonia let the students make mistakes while also teaching them at the same time with no repercussions.

Having prior experience in teaching, I was able to communicate with a local primary school and chose 23 children at random. Some of the children had no experience with gaming while others were interested in pursuing a computer science career. A classroom was set up with laptops and mobile devices. There were four groups of 5 and one group of 3 children. Each group took around 15 minutes to play the demo and around 10 minutes to answer the questionnaire.

I thought it'd be best to keep an organized and controlled environment with a smaller number of participants at a time so I can be available if any questions arose. For the study, I used mobile devices and laptops. Some children were set to play on a mobile device and others were set to play on a laptop. There were laptops with touch screens and others who played with a mouse.

This way I was able to look at how they interact with each device and took notes on what kind of changes I have to make to the game to fit different criteria. There were five groups of children who came one after another into a room with an already set device two to three seats away from one another.

Even though there was no right or wrong way to play the game, I did not want them to be influenced by one another. In total there were 23 children aged 7 to 14. The demo took around 5 to 10 minutes to play, depending on the age of the child. The younger ones took around 15 to 20 minutes to finish the levels.

Each child was instructed to enjoy the game, with no distractions and an understanding that this was solely for a research purpose so their performance and feedback had to be genuine. No further instructions were given, since they had to gather all the knowledge through the in-game explanations.

Most of the children had silent reactions to the game with loud comments when they either got stuck on a question and had to restart the level, or a victory comment when they passed a level.Once they were done with the demo, I sent them a feedback form with 20 questions. The answers were anonymous since I wanted to make sure they gave honest answers. The feedback forms were a combination of multiple-choice answers, short answers and a rating system.

5 Evaluation

In order to evaluate the effectiveness, a qualitative and quantitative study was carried out that included twenty-three participants. I focused on their feedback on what makes the game more enjoyable to play while keeping the learning elements. The questionnaire focused on the enjoyment level, learning, game play, motivation, difficulty, excitement, and interest.

A **Binomial test** is a statistical test that determines if the proportion of success in a binary outcome. This outcome is compared to the hypothesized proportion. In this study there are only two outcomes that can be possible, either a success or a failure. These are the steps that determine a binomial test.

1. **Define the Hypothesis**:
    - Null Hypothesis (H0): The proportion of successes is equal to the hypothesized value (p0).
    - Alternative Hypothesis (H1): The proportion of successes is not equal to the hypothesized value (p0).
2. **Collect the Data**:
    - Count the number of successes (x) and the number of trials (n).
3. **Calculate the Test Statistic**:
    - The test statistic in a binomial test is based on the binomial distribution.
4. **Determine the p-Value**:
    - The p-value is the probability of obtaining a result as extreme or more extreme than the observed value under the null hypothesis. It is calculated using the binomial probability formula or statistical software.

5. **Compare the p-Value to the Significance Level (α):**
    - If the p-value is less than the significance level (commonly $\alpha = 0.05$), reject the null hypothesis. Otherwise, fail to reject the null hypothesis.

Since the questions were multiple-choice answers, I focused on making a binomial test using Python. I created a script that counted "strongly agree" with "agree" as positive feedback and "strongly disagree" with "disagree" as negative feedback.

To ensure I have the correct numbers, I also printed out individual responses. The focus is on having one scale, either they agree or disagree with my game, completely eliminating neutralization.

Question	Positive Responses	Negative Responses
I feel confident about understanding Algorithms after playing the game	25	0
I can explain what an Algorithm is in my own words	25	0
I feel confident about understanding Loops after playing the game	23	2
I can explain what a Loop is in my own words	19	6
I found the game to be engaging	25	0
The game did not keep my interest while playing	4	21
I will likely play the game if more levels come out	22	3
I felt excited while playing the game	24	1

Table 5.1: Overall Positive vs. Negative Responses

As seen from the above Table 5.1, I got positive feedback. From all the questions that included a multiple-choice answer, I combined the answers individually and generated the table. In the form, I also added a double-negative question, "The game did not keep my interest while playing", as a way to invert the scale.

This way, I ensured that the children paid attention to the questionnaire. Since it was important for me to get high-precision answers, the reliability measurement ensured that the responses were consistent across all different types of questions. This helps identify whether those who respond truly understand the topic of the question and the question itself. Another important reason is reducing biased responses.

If everything is positively phrased, the children will go through each question without a second thought. Since making sure that my video game is effective is more important than getting positive responses, I had to make sure the reflection of the contents in it is worth the response.

The first few questions were about understanding the concept they learned. If they recognized the word, understood the concept of it, and could describe it in their own words. The majority of the levels focused on Algorithms and Loops. The same concepts were reiterated throughout the levels.

From the Table 5.1, the only conclusion was to make a binomial test in Python. The first idea was to gather together the positive answers as one positive answer and gather negative answers as one negative answer. This would mean that from the 23 participants if you gather all the positive answers in the above table you get 160 positive results.

In the same way if you gather all the negative answers from the above table you get 40 negative results. This posed an issue since to be able to make a binomial test you cannot have more positive answers than participants because it makes more sense to have one positive answer per participant.

That being said, I decided write a script that will take the answer for each answer individually and do a test for each. This way we can count exactly which part of the demo failed and which was successful. I took the table which was created from the feedback answers and added it to Python. This is the Python code for performing a one-sided binomial test. Binomial tests are very straight-forward. There are only three step to the process.

```
from scipy.stats import binomtest

# Data from the table
positive_responses = [25, 25, 23, 19, 25, 4, 22, 24]
negative_responses = [0, 0, 2, 6, 0, 21, 3, 1]

# Perform a one-sided binomial test for each question
for i in range(len(positive_responses)):
    total_responses = positive_responses[i] + negative_responses[i]
    p_value = binomtest(positive_responses[i], n=total_responses, p=0.5,
    alternative='greater').pvalue
    print(f"Question {i+1}: p-value = {p_value:.4f}")
```

1. **Hypothesis**:
    - **Null Hypothesis**: The proportion of positive answers is 0.5, which is half of the answers.
    - **Alternative Hypothesis**: The number of responses is not half, indicating a deviation from the neutral ground.
2. **Decision**:
    - Making a decision based on the results of the test.
3. **Explanation**:
    - Providing an explanation for the decision based on the test results.

We want to test a Null Hypothesis, we are testing if our experiment failed or succeeded. It seems as a ruthless scale, although it helps in the long-run to create a successful project.

Q	P-value	Explanation
1	0.0000	Strong evidence that the proportion of positive responses is greater than 0.5.
2	0.0000	Strong evidence that the proportion of positive responses is greater than 0.5.
3	0.0000	Strong evidence that the proportion of positive responses is greater than 0.5.
4	0.0073	Evidence that the proportion of positive responses is greater than 0.5.
5	0.0000	Strong evidence that the proportion of positive responses is greater than 0.5.
6	0.9999	This suggests that there is strong evidence against the proportion being greater than 0.5, which aligns with the negative framing of this question.
7	0.0001	Strong evidence that the proportion of positive responses is greater than 0.5.
8	0.0000	Strong evidence that the proportion of positive responses is greater than 0.5.

Table 5.2: Binomial Test Results and Explanation

In order to understand the answers we got, I created a table with the P-value and what it means. If you look at the Table 5.2, each statement concludes a positive connotation. Looking through the answers, the only prominent answer I can pay attention to is Question 6 with a P-value of 0.999.

This answer could not be further away than the 50 percent we are looking for. All the evidence that we see from it suggests that is the most negative answer. Now, lets look at the Table 5.1 and read out question number 6.

The negatively written question "The game did not keep my interest while playing the game" also concludes a positive answer. This is very important because after taking a look at the Table 5.2, each question falls into the positive category, confirming that each statement has been approved. The Binomial test results also confirm the research questions.

The feedback form was created to gather information about the user experience and the quality of the game. This allowed the participants to write about which part of the game they enjoyed more, giving me insight into the more exciting parts of the game. Since the children need to be invested in the game to continue playing, it is significant to understand their behavior, attitude, and motivation towards it. I gained a clear understanding of why one level is more fun than the other, why some game elements frustrated the users, and why other elements made it easier to navigate through levels.

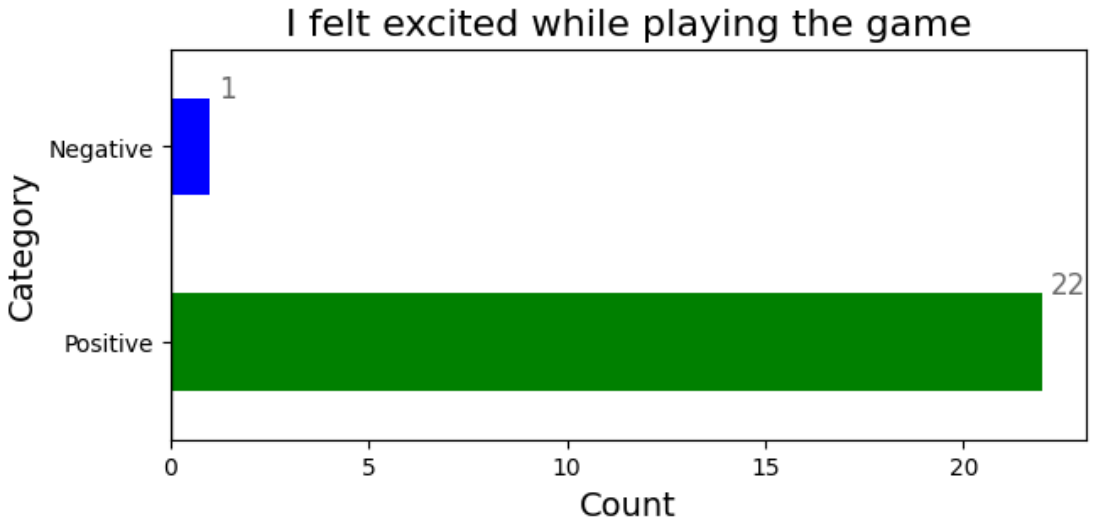

Figure 5.1: I felt excited while playing the game

As seen in the above Figure 5.1, from the 23 children that have played the game, the highest number is on the positive side. This gives me awareness that the product I created met the expectations of my research.

Each level the children went through included dialogue that provided more insight and explanation for the story. The dialogue always mentioned an update on what currently happening and then gives instructions on what the player needs to do. They never gave hints or details to show how to solve a level. The excitement can be measured how interactive they are doing game play. This could continue through demo trials where in each trial new levels are tested.

This also opens up the possibility of testing the same children year round and seeing how much they have advanced through the year. In the meantime, the excitement level was measured how into the game they were.

In the first wave with the first five candidates, the introductory went too in depth. In the surface, it should have been myself introducing my thesis and the objective of the day. Although, not being sure how they would react I also went through the levels. There was initial nervousness of how the game will be accepted. Once they started playing, it became clear that I did not in fact have to introduce the game to them. They were able to read through, interact and figure out the story themselves. There were students who wanted to sit and play next to their friends except that was not allowed. The room where the demo was held was treated as an exam room. Since, in the end that is exactly what the research is about. Testing if Codonia could be successful enough to enhance learnability in children.

Through wave two and three, the ambiance had relaxed and I was more focused on talking about Codonia as in general as video game rather than what they need to do so the kids could figure out the other game aspects themselves.

The sense of mystery as to what they will be playing and the process of figuring out its a game, got them extremely excited. They paid attention to all the text, read through the answers and took their time answering. It was evident those who played on a computer finished the game quicker since the text design was more prominent in the bigger screens. Those on mobile were able to play the levels quicker since the game design was made for a hand-held device. By the end of the fifth wave, we had gotten similar interactions and the same level of enthusiasm from most of the students.

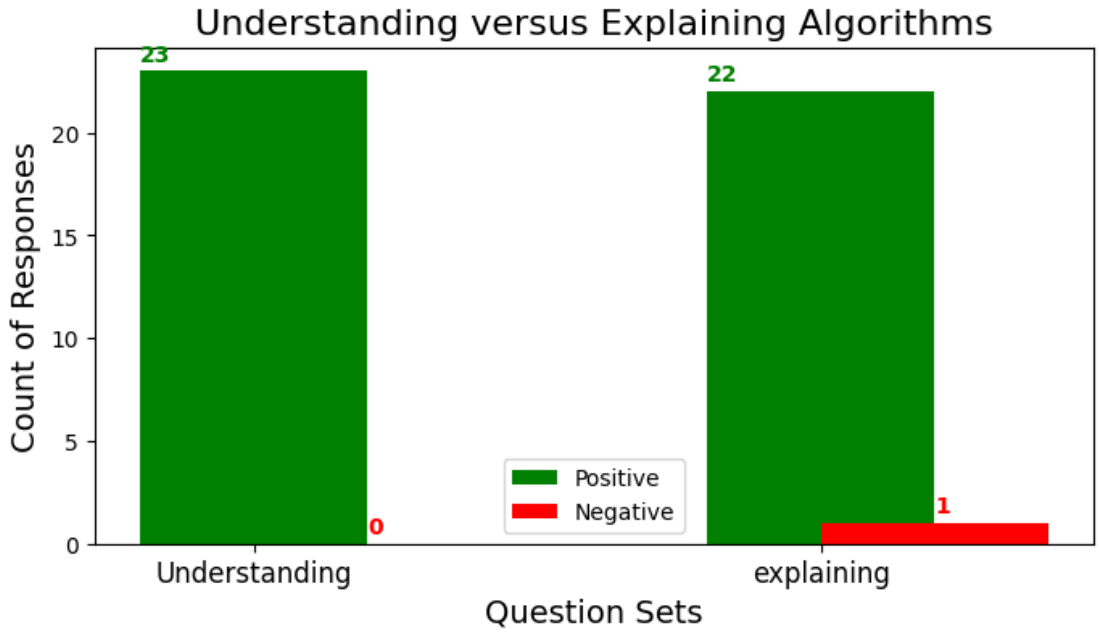

Figure 5.2: Understanding versus explaining Algorithms

The first and second questions in the feedback form are related to Algorithms, seen in Figure 5.2. The first question, " feel confident about understanding Algorithms after playing the game", serves a different purpose than the second question, "I feel confident explaining what an Algorithm is." Understanding a term versus explaining what it means is measured at different knowledge levels.

One can understand a concept but not be able to explain it in their own words. Even if the children are not able to explain it just by playing the demo, as long as they capture the essence of what an Algorithm is, they have learned something new.

The **cognitive process** in which a person comprehends how concept works, could potentially also show signs that they are good at pattern recognition, they have use the knowledge in different aspects and they can interlink ideas.

Pattern recognition is about recognizing structures, patterns or regularities within data. This plays a big role in trying to identify common principles or reasons of a concept.

This means that you can see the step-by-step processes that make up a big picture. Once you understand how step one works, you can continue to decipher each step until you see

a common pattern. This pattern is the blueprint that can be applied to another similar problem. With the new awareness you can start applying the knowledge to different aspects. So, the user or player has the ability to be flexible in problem-solving situations. An example would be understanding what an algorithm is and trying to decipher exactly what turns it into an instruction. You start to interlink context clues until it turns into a pattern and you come to the conclusion that you can apply algorithms to everyday life.

The second statement, ''I can explain what an Algorithm is in my own words'', dives deeper than just understanding what they have played. Once the children played the game, we went through the motion of asking them general information about the game and the lesson. I questioned them about Algorithms and let them create an example together. From each group, I got a lot of interesting examples, as they all tried to make steps on how they get ready in the morning. They took a concept that was new and tried to make it into an example that everyone can grasp.

Breaking down each definition that everyone said in their own words and making them more simplistic in the form of step-by-step examples.Another indication that they did understand what the lesson was, included giving them a false answer and letting them break down each statement until they figured out why the concept was wrong. They had to look at a list of two different algorithms and in the end they chose the right one after dissecting each statement.

This teaches them the method of divide and conquer which is used in many problem-solving tasks. If a task is too big to be solved in one go, you set it into smaller portions and start to solve them in parts until the big picture is complete.

The third and fourth statements are related to understanding and explaining Loops. The same logic would go into this lesson as well, since we want to make sure they understand the concept even if they cannot explain it with their own ideas yet. As seen from the Figure 5.3, a high number understood what the lesson is and could give an explanation.

The complexity of loops is not in the definition itself, but the way it needs to be taught. They are more abstract since they do not have a straightforward answer as to how to use them. Loops seemed to be a harder concept to grasp. Not in the sense that they did not understand it, but it was harder for them to explain why we use it. This would largely be due to the ineffective interaction with loops during game play.

Algorithms were easy to create in a level but Loops require more practice. From the Figure 5.3 , I can pinpoint exactly which levels need to be changed design wise. The game, should have a clearer visual explanation with added interactivity to the elements.

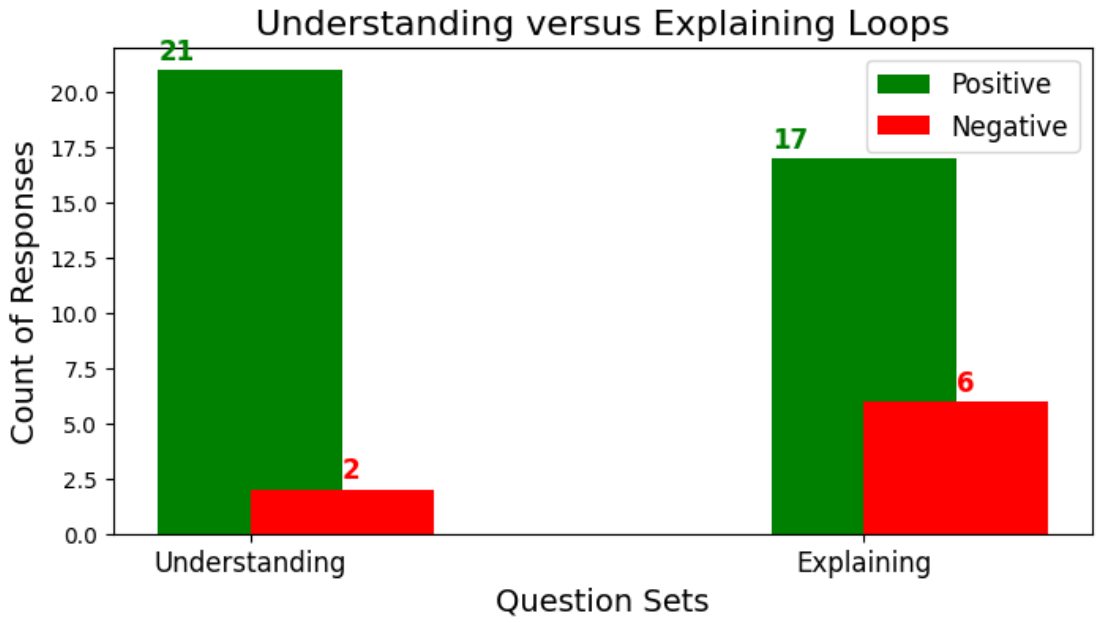

Figure 5.3: Understanding versus explaining Loops

Not all lessons can be created the same, using the same level design. The children already proved they have a strong pattern recognition skill, which means there have to be constant new ways of teaching different methods so they do not fall into a repetitive, boring pattern. One way I could have made Loops more understandable was to let the children create their own loop. The main character in the game would have to walk to the right five times and the children would have to program it themselves using coding blocks. Creating a level with real-time feedback and view-ability would turn the game into an experiment. It would let the players create their own building blocks and tweak their answers to reach the correct answer.

A **sandbox element** where they can drag and drop commands to create a pattern and let that pattern repeat in an optimized solution. Another idea would be re-playing Algorithm levels but now using new knowledge and re-trying old solutions with less effort. Constant changes to their coding blocks would mean immediate feedback with visual clues and tips as to how they can find the right solution.

As long as the response is in a positive manner and encouraging to player to keep trying, the motivation and engagement levels will stay the same. This would also including raising the difficulty of the levels the more they play. Starting off with simple loops and going to more complex solutions once they start to get comfortable with experimentation. This would also let their creative side go wild. Since there are no bounds in video game imaginary, there should be no bound to what they can create themselves. Programming should be fun and approachable for everybody.

This was the data collected throughout the focus groups. There were five groups with around five children each. The questionnaire focused on these main points:

- **Enjoyment Level**: How much fun did they have while playing the game.
- **Learning**: How much did they learn from playing the game.
- **Game Play**: Did the user interface, controls, and overall design of the game satisfy their needs.
- **Motivation**: How motivated were they depending on the storyline, learning, or levels to finish all the levels.
- **Difficulty**: Was the game too easy or too hard for them to learn.
- **Excitement**: How excited they felt while playing the game or finishing a certain level.
- **Interest**: Would they play the game again if more levels came out or a new update was available.

When it comes to keeping the children engaged, everyone agreed that the game kept their interest while playing. Even though one child did not maintain their enthusiasm throughout, as seen in the Table 5.1, they were motivated enough to finish the game. It's not only one aspect of the game that made the game engaging, the research says that it's a combination of the levels, design, learning, and game play as seen from the Table 6.4. There were specific levels that were more desired than others. These levels were the ones that were more confusing to figure out what is going on.Throughout the focus group sessions, we had three educators on standby who took notes depending on the children's reactions.

There is an indication that suggests the children received the game very well. From being confused to focusing on the game, their attention had shifted in wanting to make a good impression. There were audible noises for successful or failed levels. This was a sign that they were invested in passing the levels. Students who failed a level were disappointed when the screen went blue, thinking they had failed the level. In the next few seconds, the level would restart and they would get their enthusiasm back to re-do the level. The second or third time around you could see they started paying more attention to the answers and how they performed the last time.

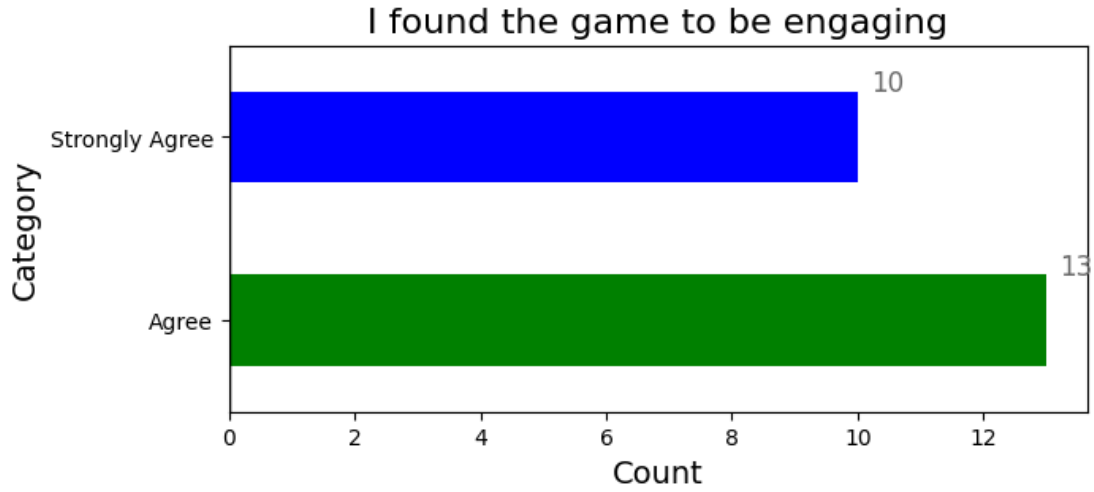

Figure 5.4: Engagement throughout the game play

From Figure 5.4 you can see increased levels of engagement which can be categories into three different conclusions.

Engagement is key: Having subjects that participate with high enthusiasm is important to get good feedback. If they are committed, they can absorb more programming concepts, which makes them go faster through the lesson plans. This is also a good indication that the game is doing its job correctly with the right audience.

Constant positive reinforcement: Keeping their interest is as easy as giving them constant feedback with a positive attitude. Telling them they are right, with an explanation as to why they are right. When they get a question wrong, they get an explanation as to why their answer is not the right one. Without giving harsh criticism, they are getting clarification on different subjects.

Improved learning outcomes: An interactive education game improves the outcome of the learning experiences to make it more pleasant. Less stressful environment, without any repercussion that will let the child grow through their mistakes without feeling like they are constantly failing.

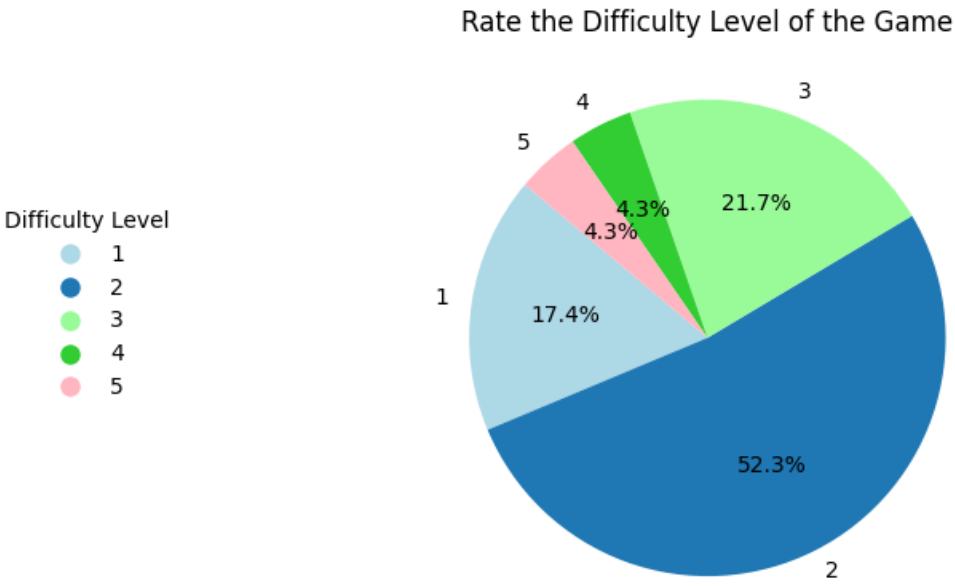

Figure 5.5: Game difficulty rating

The difficulty of the game was rated on a scale of one to five displayed in Figure 5.5, with one being very easy and five being very difficult. From the levels that were played, more than half said that the game was easy, which sparked the desire to ask for harder levels. The short answer reviews I got from the form also contained requests to raise the game difficulty. As the levels progress, the questions will get harder and trickier.

Different strategies game developers use to adjust difficulty, such as dynamic difficulty adjustment (DDA), which involves real-time changes to game elements based on player performance. This ensures that each player has a tailored experience that is both challenging and enjoyable [Uni24].

Since more than half of the respondents set the difficulty level to a 2, it means they found the game easy. While this is acceptable for a new game, it might not keep their interest over time.

The game seems straightforward, with little to no challenges. The key take away is to have adjustable difficulty settings. For each level passed, the difficulty changes and the same levels can become harder to solve. This strives for the players to complete all levels in every difficulty.

Jamie Madigan through his extensive research has written in his books about the psychology and motivation between video game levels. It's emphasized that well-designed video games can have significant educational benefits.

By gradually increasing difficulty, these games can teach problem-solving, critical thinking, and perseverance. This mirrors educational theories that suggest learning is most effective when students are continually challenged just beyond their current capabilities [Uni24].

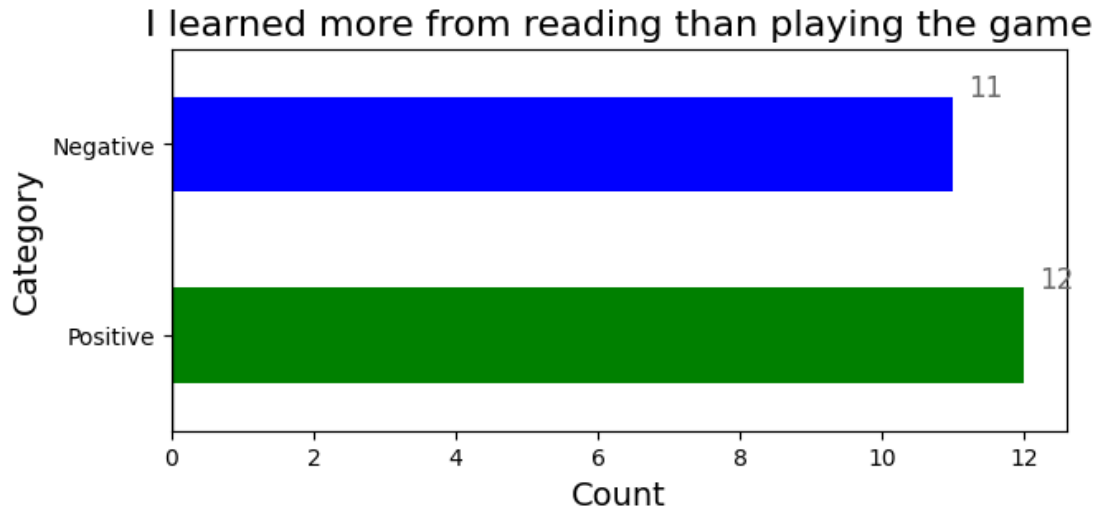

Figure 5.6: Learning vs. playing balance

They read the instructions and then put the learning into practice. As seen in Figure 5.6, the results show they gained more knowledge from playing the game. Even if this number is only significant by one person, that means there is room for adding more narrative into a game to make the balance better.

This would partially be due to them learning from a new platform. I attempted to create a balance between game play and reading. As important as it was to lay a foundation of new knowledge, I also had to ensure it did not feel like a lesson. The results are close which means they are both equally effective when it comes to the game.

The results are similar which indicated that both methods have balanced effectiveness. They learned a new subject through narrative as much as they learned from the practical elements of the game.

This can lead to the conclusion that a combined approach, that makes a perfect mix of game play with reading practices. The hybrid approach can benefit the students through exciting user interface components and captivating story. Furthermore, this can open up new topics for further research about the understanding of children learning preference.

The result and insight from this new knowledge can further aid us into figuring out what motivates them. How to make an even more engaging platform for educators to incorporate into their curricula. Further studies about the type of games, game mechanics and learning environments yield the highest learning result. I can conclude that from the Figure 5.6 there is a need to study individualized learning approaches.

Preference of learning through game play beat learning by reading by one person. The one person who was the indicator for this success might actually enjoy video games more than reading. Since learning by playing was favored by only one vote, it could also show that this person leans more towards gaming than reading.

This is a topic I will talk about later on, how video games are subjective depending if a person is more game-oriented or study-oriented. In addition, this also opens door for further research to understand the preferences of children which will help educators in the future to create a more effective learning experience.

According to Figure 5.7 RQ3, does gamification have a higher learning outcome than traditional teaching , 70 percent of students agree that the game teaches more than traditional teaching, meanwhile 30 percent strongly agree. We can confidently say that this research question has been accepted.

I attempted to create a balance between game play and reading. As important as it was to lay a foundation of new knowledge, I also had to ensure it did not feel like a lesson. I decided to include the lesson plan between conversations and tutorials.

They read the instructions and then put the learning into practice. As seen in Figure 5.7, it clearly states that the bigger number of students would rather learn from a game than the traditional way. If these research experiments continue, we gather more data, figure out a better way to enlighten the youth, traditional teaching would turn to a halt.

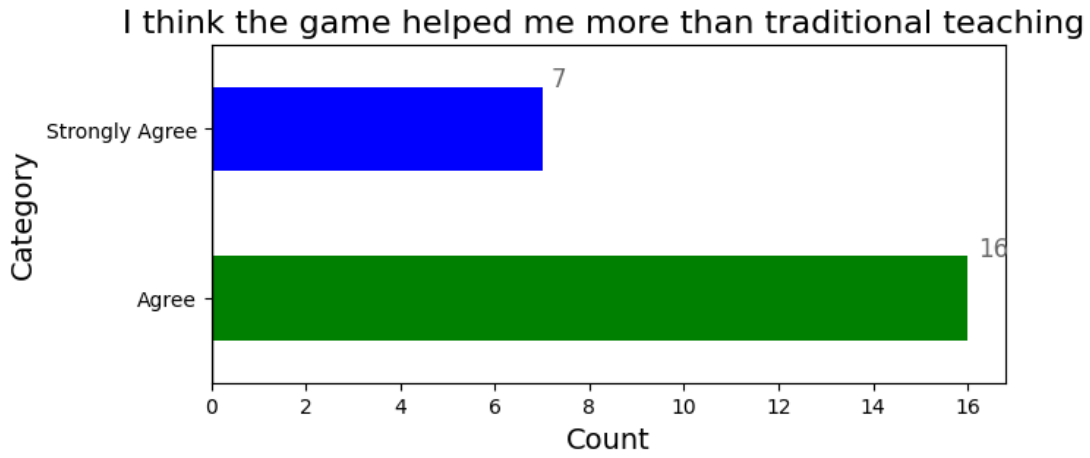

Figure 5.7: Game based learning vs. Traditional learning

In most cases, you learn more from entertaining videos or conversations than you do reading books. Unless a subject is highly engaging to the individual, there is little motivation to read the book. This applies to studies as well, most interactive lesson plans, including university, are online videos on YouTube.

Most computer science students, starting with me, has paid attention to a visually explained lesson rather than the material that was given to us in class. Without even thinking about what you are looking at, as long as there are visuals that pop-up, our brain will pick up the imagery and store in our memories. This is the positive passive effects of using video games to teach.

There will always be a way for the children to remember because it will be fun for them. They will not realize how much they have learned until they test themselves. This is why adding quizzes after each chapter is important to their progress.

Traditional learning, where students sit and listen to a teacher for an hour, is slowly starting to lose its touch. With the new generations starting to be over-stimulated through electronics, their attention span is getting smaller.

It is a small number of students who can still pay attention for longer than 20 minutes during a lesson and still be able to tell you what they understood. Either teachers would start to catch up to these new times and learn the new ways or make way for younger educators to take the stage.

There is a lot to be said about traditional teaching, starting from how long ago it was created. Those same procedures of raising the new generations have been used and its starting to decline. The new generations do not accept anything less than a school curriculum that will force them to get engaged. This brings me to the topic of the age of the educators.

Younger educators have the capacity to make a classroom more interactive and fun. This includes having games as part of the lesson plan. The lesson plans do not include electronic video games but real-life games that get the students active and running on their feet. By doing the same activities that they would on a device, the teachers can include elements from gamification in real life. Using different kinds of arts and crafts they make visual cues for the students to learn.

I would argue that this is also a part of video games enhancing learnability. As long as its an activity that does not require you to sit, listen, write and repeat, you are part of the new system of education. Gaming should be used to continue to master standards, or problem solving standards, that have already been taught. This gives students motivation to work while also encouraging and developing their problem solving skills [Rug13].

From a research done by Dana Ruggiero about using games to teach in classrooms [Rug13], most teachers had positive feedbacks. Some even agreed that games would be a collaboration with students to help them learn different concepts with the help of their peers [Rug13].

Students are able to play games by using their knowledge from concepts gained in the classroom, and as one respondent said, “Games create learning that enhancesunderstanding of key concepts [Rug13].”

6 Discussion

Looking at the responses, I can confidently say that video games do enhance learnability. Once a balance has been found between teaching them and also making it fun, there is not a single lesson or topic that can not be taught. Seeing how these children have reacted throughout the game play and their enthusiasm after finishing the demo, is enough motivation and proof that we need to continue researching how to make games more education oriented.

This approach can let players have a hands-on chance to take educational content and learn through experimentation. Games that are educational only get boring pretty quickly, since the children are constantly being fed new information without the fun aspect. Most of the school subjects like math, history, language classes have turned into game development, this also means that different learning subjects can become more appealing and approachable.

Number	Research Question	Conclusion
RQ1	Are learners more motivated when learning with a game	Yes
RQ2	Is the self-perceived competence higher with learning with a game	Yes
RQ3	Does gamification have a higher learning outcome than traditional teaching	Yes

Table 6.1: Research Questions

These are the questions I successfully answered in the research paper. There is concrete data for each answer that proves that gamification does work. As can be seen in Table 6.1, RQ1 answers the question whether their motivation is boosted from video games instead of learning the traditional way. Understanding the feed back from this data can be beneficial to the educators who can then design motivational educational tools.

Looking into the possibilities of educational video games we can tell that it can be adaptive. Applying variants of educational themes to the design of the game, developers can cater to many a wide range of edutainment. A term that has been created in 1954 is now making a comeback, edutainment. Educational entertainment, meaning that learning and play can co-exist.

With that being said, making a new learning tool does come with its challenges. Making sure that the teaching aspect is integrated in the game instead of feeling like an add-on. This is important because for this new tool to succeed the user needs to truly be immersed in the new world and forget they are gaining knowledge.

RQ2 focuses on the competence or rather the confident of students while playing and finishing a game. The research wants to know if they believe in their abilities when learning through a game. This is crucial for a students success since it will influence their willingness to try harder challenger therefore learn more complex material.

RQ3 seeks to determine if game elements that are introduced to a student would lead to a higher learning outcome rather than traditional teaching. Through digitization we can measure their learning outcomes, knowledge retention, problem-solving skills and their critical skills. This lets the researchers measure the effectiveness which can later on be used to build even education tools with an improved overall learning experience.

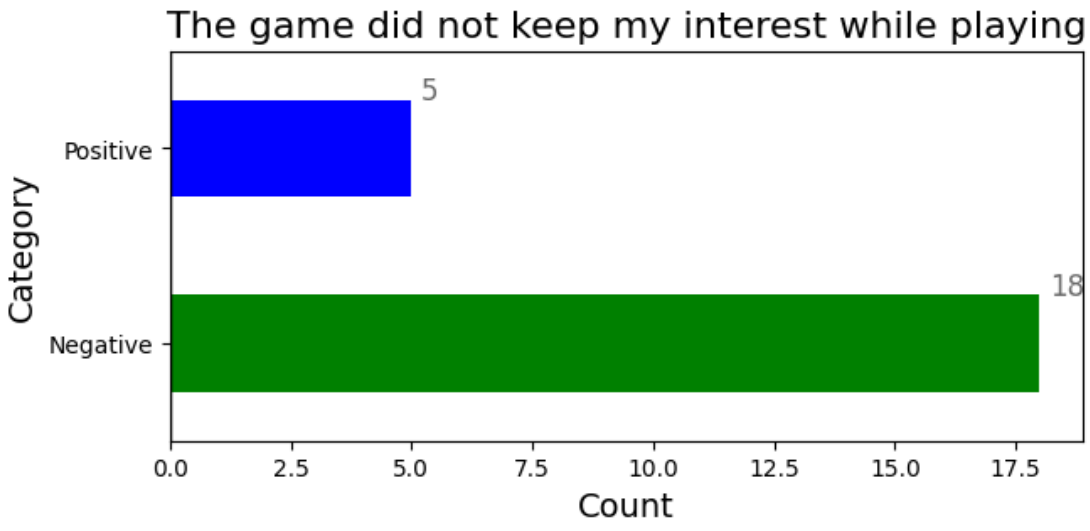

Figure 6.1: The game did not keep my interest while playing

As shown in the Figure 6.1, there is a high number of negative answers to the question"The game did not keep my interest while playing." This question, part of the inverted response, shows that while they played the demo the children were interested in the conversation, game play, and education.

An open-ended question that was posed to the kids was "What suggestions do you have to improve the game?" This yielded a lot of creative answers which I was able to put into categories. The study shows that leading result is wanting more levels. This is very motivating to see, as a creator of the game especially, that a product I have created is being sought after. As a researcher, I am more interested in knowing what exactly makes them want more levels. Which parts of the game play made the children want to play more? What did they like most about the game? Did they solely like the game more because the graphics were fun to look at or was it also because they saw a challenge in the levels? It's important to pay attention to the answers that suggest increasing the quality, difficulty, and vanity of the game.

The same way the game teaches the children that Algorithms have a clear beginning, middle and end, my game has to have the same. Creating a well-rounded product with all the elements that make-up a good game. Fixing the bugs, collisions and some character movements would make a huge difference in the demo experience. The main bugs I have in my game have to do with colliders.

Colliders or borders in my case are used so the player does not leave the map. When a player moves around the map there is only a certain point where they can to, which is the edge of the road, so the movement feels realistic. In the process of creating the colliders, I had to custom set the points where the player can not go. If you are not precise with the points that added in the PhysicsLayer, which I have talked in length about in Chapter 4: Codonia The Game, the player can maneuver out of the bounds and into the water. This removes them from the playable area and they can walk around freely. The only issue is, once they leave the bounds, it is hard to go back in the right to pass the level, so they are stuck. This is where a reset button would be most helpful. From watching the children play, I found the biggest problem areas in colliders that I will fix before continuing adding new levels.

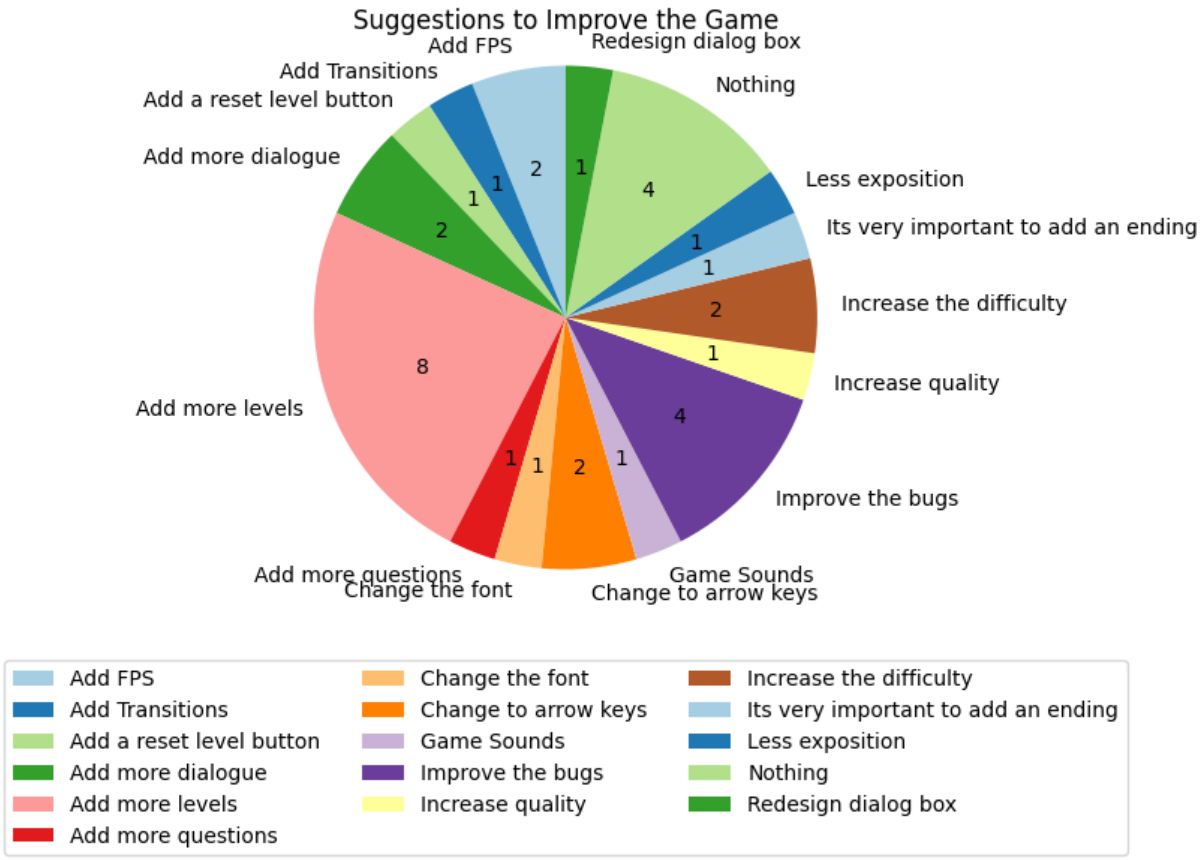

Figure 6.2: Suggestions to improve the game

In addition, the buttons could do a re-work, since it was mentioned in the "Improve the bugs" category featured in Figure 6.2. In the levels where there is dialogue, I have added an invisible big button that lets you click anywhere in the conversation to move it along. This seemed like an aesthetically pleasing solution, although not all children understood what they needed to do. Some started clicking around the map to continue and accidentally clicked multiple times on the invisible button which made them miss a few lines of conversation.

Through user testing I was able to figure out how the children think and what I can do to make it easier for them to understand the user interactions so they can solely focus on the learning. This would also increase the overall quality of the game since it would become a smoother experience.

Even though, they were able to figure out what they needed to do, they still felt they missed a part of the story. I am planning to implement the option to read through the conversation as many times as they click on whichever character is currently speaking.

The second issue with the buttons is in the last level. The last level contains houses with floating scrolls. Each scroll has a question that needs to be answered and once done you have to click the "Back" button. For some reason, even though the whole button is clickable, only the right side of the button functions properly. This is probably due to some kind of layering mistake, which I also mention in Chapter 4: Codonia The Game.

Most of the responses have to do with the design of the game. Adding sounds, transitions, more animations, and re-designing some elements. This gives me an idea as to what elements are missing from the game, so the next demo I create will be more polished and give a clearer impression what the game will be. The buttons in the game are named "left", "right", "up", "down" since I want the children to be familiar with these terms so I can use these terms later on when it comes to actually programming. Getting familiar with the controls so they can write them out in harder levels.

Although, two responses asked to change to arrow keys which has put me into a dilemma. I understand putting arrow keys would be much easier than just reading the solution, since the brain does not have to read what the instruction is. If you see an image of an arrow going up your brain registers it quicker and you have a faster response time.

Although, this is not a competitive game so how quickly you response to an answer or use the controls has no reward. Either the children who wrote the review struggled immensely when they needed to move the character throughout the screen or they simply did not want to remember which button is located in what area on the screen.

In any case, this is an important factor to consider which will require coming up with various solutions to fit different student needs and run tests. This way I will be able to come up with a user interface that would fit the broader audience of children who want to learn programming through playing a game.

Figure 6.3: Moving through the invisible forest

The buttons are rectangles in a square format. Two buttons are on top and the other two are below. If the level of the game required the player to move around then two of the buttons would be on the left and the other two on the right. This ensured that when you are playing on a hand-held device you have easy access to the buttons 4.8. From the Figure 6.3 you can see that the "down" button is paired with the "right" button and the same way "up" button is paired with the "left" button.

This was designed intentionally since our brains connect these commands more easily together.The other opinion I have on this response is that I can add more variants to the controls. Let's say I add an options on the settings tab where they can choose if they want arrows or words on their controls. If they choose words they get the keep the game as is, which would make it much easier later on for them to start programming. If they choose arrows, the design of the game shifts.

Now, whenever the words "left", "right", "up", "down" show on the screen even if its in text it would be in the arrow forms. For example "Click the $\rightarrow$ to continue to the next level." This would make an new mix of written and symbolic language. Imagery is processed much quicker in our brains since we relate it to a word or an action.

I presume this would change the whole gaming experience. If i were to add the arrow keys, their response time would be shorter which means I could add competitive levels. Timed quizzes they can play against their friends.

Having a leader board will add a new edge to the game. This way they will have a new type of motivation to keep playing. Although, as seen in Figure 6.4, There is enough motivation from the gameplay for them to keep being interested in the game.

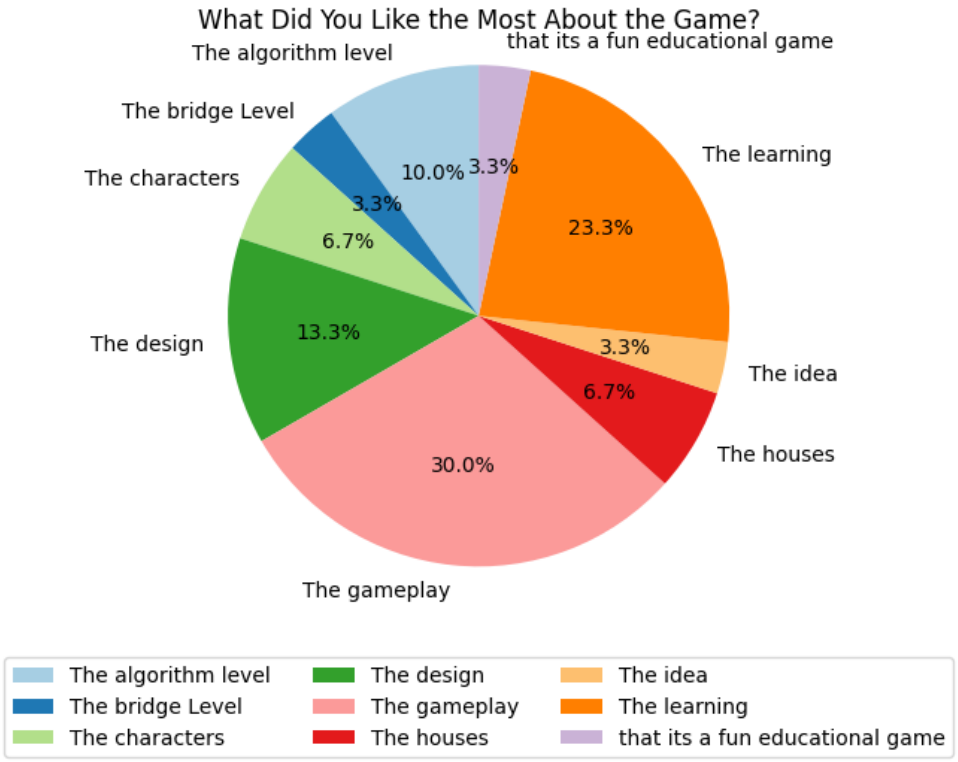

Figure 6.4: What the children liked most about the game

The levels had some kind of mystery that the children had to solve. The game included a level where there is a road which is blocked by some trees. You cannot see the road clearly as a player since you have to follow a sign with instructions. There is a notice board that says "You must follow the instructions, complete the Algorithm and you will be free." This introduces them to two words - instructions and Algorithm. After receiving instructions they have to follow the algorithm step by step and find the way out. The instructions tell the user how many steps they need to take in each direction.

An example would be “5 times right, 2 times down, 3 times left” until they hit the collision that takes them to the next scene. This stage had the children counting out loud to make sure they do not miss any steps. Understandably, in the beginning they want to pay as much attention as possible to not make a mistake. Even a simple level that gives them the answers, they take it as seriously as taking an exam. This behavior would change later on as they got more comfortable making mistakes.

The fun part of experiments lies in how creative you can be with your material. So far, the criteria is to make it enjoyable, educational, and difficult enough so they do not lose motivation, adaptable levels for specific preferences and so on. One element I would focus more when it comes to the game play is the difficulty.

This takes time to create a precise lesson plan. It should not only explain what a term is and then have the children say it back to you in the form of fill in the blanks. In Codonia, the dialogue would say the term a couple of times and give a simple explanation, as seen in Figure 6.5, that incorporates it into the story.

Figure 6.5: Dialogue giving clues what to do

The other level that was most liked involved a series of invisible bridges and a pawn. You had to click the troll to start the level. There were a series of four questions with multiple choice answers.

Each correct answer would reveal a bridge that connected platforms. They had to get all the questions right to be able to end the quiz and the controls would appear. The statements in the quizzes were simple yet they re-iterated a lesson they had previously read about while adding more details. If one level explained Algorithms as "step-by-step instructions", in the level it would be "step-by-step instructions with a beginning, middle and end." By the time they finished and arrived to the last levels they would have the full definition with a few examples and the ability to recognize a right versus wrong algorithm.

Before starting every level there was an introduction to the person who is speaking, who they are, and what their intentions are. For example, in the first scene, which is a cut-scene played like a movie, you can see the player dressed in normal clothes walk through a portal. This triggers the second scene that opens up a dialogue.

The player can see a ruined castle and a few trolls hitting watch towers. The towers are on fire and the dialogue starts. You click through and find out that you went through a portal and are now in another universe. In the end, you are instructed to move closer to the ruined castle. The controls appear on the screen with a sentence advising you on the next move. In here, the children are already getting a feel as to how the game will be. We set a foundation of an early-stage story with a mystery. They do not know how they got to the place or what is going on. Only by meeting each **NPC** (Non-playable character), are they able to get clues as to what their next step is.

This is also a reason as to why the design of the game is third most liked element of my game. They see the outcome of an event that they were never a part of and have to work backwards to figure out what is going on and how to fix it. Moving on to the third scene, we see more ruined watch towers only this time there is a new character on scene.

He introduces himself in a prideful manner, telling the children he is the best villain and he has captures the queen of the land they just landed on.

With his speech finishing he fled the scene leaving our character and a few enemies alone. The instructions for this level are to get the answers right in a multiple choice quiz which might prompt the enemies to not attack the towers. This is where they learn what Godot is, what scripting language its using, and finally they get to see how to print out the words "Hey, stop!" in GDScript. This new scripting language has been a roller-coaster to learn.

Since its being constantly updated, libraries or functions that have worked five months ago, do not work anymore. In these last 1.5 years, I have had to update my scripts various times. Each update, deprecated a statement and I had to find a new way to re-create a function that was already implemented. Although, GDScript is amusing to learn and should be used to teach programming in schools.

Throughout the levels there is a clear indication of what needs to be done, which buttons to press and where to go. The last level gives no context. The scene is set up between three houses, in which, only two of them are playable. The houses are named "Town Hall" ,"Beginner Algorithm" and "Beginner Loop."

Making two out of three houses interactive served a purpose to see how curious the children would get. By the end of the game, they had to be intrigued enough to start tapping random areas of the scene in order to explore other aspects of the game. This level gives no explanation what they need to do.

Counter from every other scene, where they got clues, it was time they led their own investigation. I believe this sets precedence on their brains will start to think. Their curious nature needs to be a part of their play style in order to let Codonia become a success. Children need to constantly ask the most important question.

That question is "why?" Why is this level designed this way? Why is this answer the way that it is? The mystery of the storyline together with the curiosity that comes with their age should be the perfect formula for an educational game to succeed.

They need to press the houses and figure out how to play the next level. Each house has three questions, with statements. You either have to answer a true or false question, find the right script between two scripts or find the correct definition for each term. Whether they choose the right answer or not, an explanation will pop up with an example. The explanations will either tell you why this is the correct answer or why it cannot be.

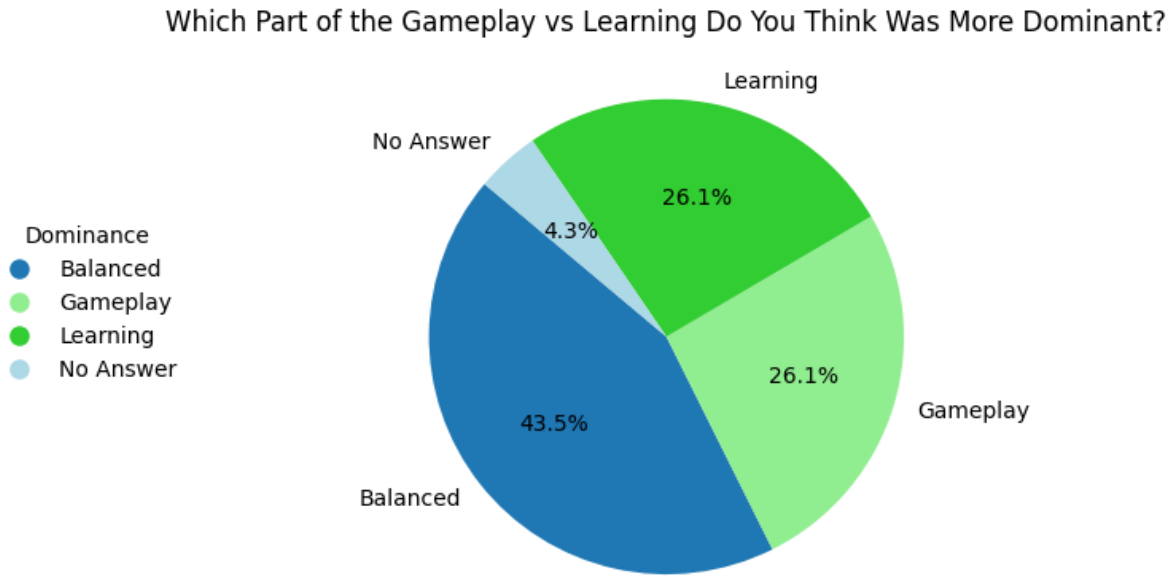

Figure 6.6: Results between learning versus game play

The point of the study is to see if we can further develop games and make them more educational for children, although, that does not mean that they will always see it that way. Video games are subjective. It will not always apply to every child's needs and it will have to be re-worked for individual personalities. Clearly, we can see that from the Figure 6.6 we had ten children think the game had a good balance between what they learned and the game. This means that they were not overwhelmed by or the other. Each level stood out equally for them and it went to their favor to enjoy it.

The other two parties were divided split in the middle, since one of the children did not have a clear answer. There is no data as to why they chose not to answer and preferably once the game has been updated with the suggestions from Figure 6.2, where I mention what the game lacks, I can do a second experiment with the same children.

To revisit the primary split parties mentioned previously. The results for learning versus playing is split in equal. For those who chose to answer game play might have resonated with the game more. This would mean there is a higher chance they enjoy the entertainment aspect of a video game. As much attention that they paid to the learning process, they were captivated by the design and characters.

Meanwhile, those who chose learning would fall into two types of categories. Those who value learning more rather than paying attention to the game details and those who might have felt overwhelmed with the amount of new topics they had to read about.

Looking at the results in Figure 5.5, we can make an educated guess and say that the children were not overwhelmed by how hard the game is. The rating on difficulty out of five is a two. This means there was not a point in the lesson that they felt like they could not finish a level. This also leads to my second point, the six candidates valued more the learning process rather than how the game looked or the conversation between the characters throughout the game play.

So the following experiment would be based on how to balance out the game entirely? We cannot make every level the same without it becoming repetitive. The best option in this situation is to have levels that suit both parties. Create a roller-coaster effect throughout the learning process. The lessons that will be hard to solve, should follow up with an easier level and a new addition to the story-line. This way, they are constantly being stimulated but not overwhelmed.

Category	Count
Strongly Agree	10
Agree	13

Table 6.2: Responses to the statement "I found the game to be engaging."

From the Table 6.2 I can conclude that hypothesis 1, Gamification increases the motivation of learners, that is also the first research question I aimed to prove, does gamification increases the motivation of learners, is approved. From 23 of the players everyone has agreed that their engagement throughout the game play was positively accepted. They were intrigued by the story-line, game play and design of the game.

Since they found the game to be engaging, we can also conclude that they were excited to play it. This is a great outcome for me, since Codonia can now be used research for further development. Through continuous updates and new lessons, I can make the video game a play-ground for myself to be able to answer more research questions.

Frustration is one of the main reasons people quit a task or a game. The games with the lowest ratings on gaming platforms are due to players giving up before they can learn how to play the game.

This is either due to the instructions being too hard to understand or the interaction is hard to learn. Usually when a game is hard to learn there is an indication that there is a lot of mechanics and lore to explore.

If a game has a lot of lore, or as we call it story-line, and it is organized in an optimized manner, the game will be extremely enjoyable. This lets the player get lost in the story, the characters and their lives. It's already been proven that users that are interested in the narrative of a game also have higher chances of learning more. This means the more engaging your game is the longer they will play, get rewarded for their loyalty and in return the player base will grow. In the end, these hard to understand mechanics turn out to have the best story involved games.

Although, I have to be realistic when it comes to a video game for a mobile game. I do not believe I have the privilege of adding a huge story-line when the game is meant to teach. There is a limited amount of narrative you can add to a game before it turns from an interactive story-driven game into a traditional teaching method, where they read more and have less practice. The research aims to teach programming through interactivity, not through text.

If the game had more text than practice, it would just be interactive reading and that borderlines reading a book. Different strategies game developers use to adjust difficulty, such as dynamic difficulty adjustment (DDA), which involves real-time changes to game elements based on player performance. This ensures that each player has a tailored experience that is both challenging and enjoyable [Uni24] .

This can also be connected to adding different styles of levels. A monotone product results in monotone game play. Creating levels that require the player to think in a more elevated state for each solution keeps them playing longer. As mentioned in a previous paragraph, there needs to be a clear structure of level designs to fit specific needs of learning.

From a previous Figure 5.5 I can also conclude that most children did want harder levels. Challenging them to make learning fun because this way they have to figure out for themselves what the answer is. In the same time, this also makes it into sort of a puzzle since they are constantly looking for answers and putting pieces of code together to make it work. The over-justification effect is the net negative effect on engagement and motivation from an over-reliance on external motivating regulations [vRZ17].

Poorly designed gamification that invokes the over-justification effect often stems from a frustration over poor learner engagement, when, in actuality, the poor learner engagement may be related to the underlying instructional design [RWJS18]. There is an negative effect on those games that have been poorly gamified. In fact, some of the most famous companies have tried gamification with no successful result. These projects include Microsoft Ribbon Hero, Volkswagen Fun Theory, and Google News Badges.

Microsoft Ribbon Hero aimed to teach Microsoft Office's Ribbon interface through gamification. They had many issues including technical problems, where the users had compatibility issues with the Office versions and that lead to frustration. Those who were already familiar with the interface were not interested in it, which lowered the appeal. Finally, the project needed constant updates and that became unsustainable through its lifetime.

Volkswagen Fun Theory aimed to change peoples attitude for the better through everyday activities. This stopped when the fun elements were more often temporary and did not have long-term engagement. Maintaining the campaigns required a lot of resources so it was difficult to maintain and it actually had limited impact. There were no lasting behaviors that turned into meaningful impacts.

Google News Badge wanted to encourage users to read more articles. This did not work for long since the users did not find the rewards motivating enough. There were privacy concerts with the users that did not want to share their reading habits. Most importantly, the implementation was poor so the badges did not align with the users real-world goals or interests.

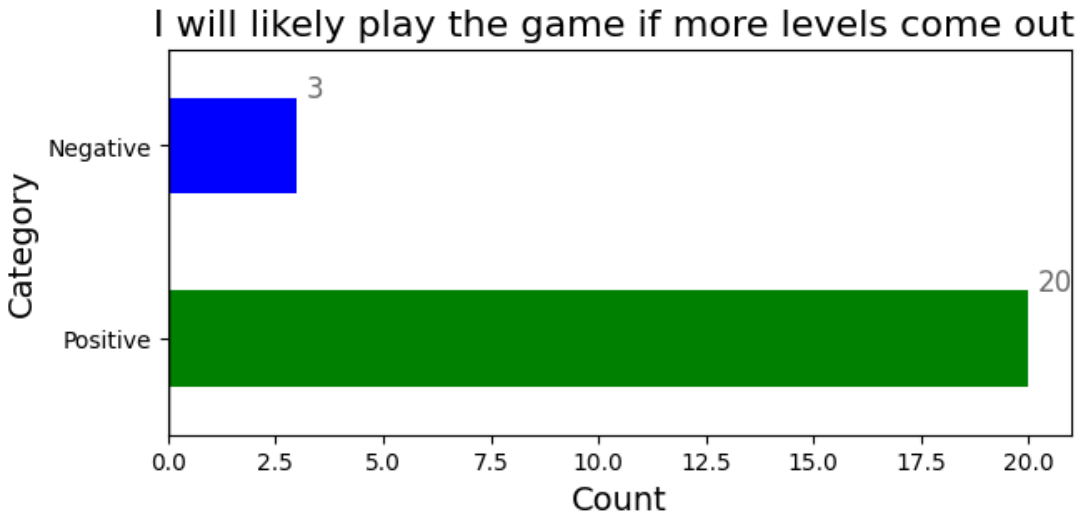

Figure 6.7: Results for students who would likely play the game again

From the Table 1.1 RQ2 talks about the self-perceived competence a player or student has when it comes to gamification. In other words, how confident a student or player feels after playing the game. It focuses on if they understood the concept and if they feel comfortable enough to continue learning. From the above Figure 6.7 there is a high number of confident students who would continue to play the game. This is the first point that I truly believe supports RQ2 that the students self-perceived competence is higher through gamification. Not only does this prove that they enjoyed playing the game, it also means they saw something in the game that they want to play it again. Either being the design, narrative or learning aspect, one part of the gamification objective has gotten them interested to return. According to a study that talks about retention benchmarks for mobile games has set categories as to why people return to mobile games. Day 1 retention measures the percentage of users who return to the game on the first day after installing it. Understanding this metric is a fundamental part of analyzing mobile game retention benchmarks [Gup24]. There is a 30 day retention period although since I am the early stages of my game I can focus on day one.

Day one retention has everything to do with first time experiences. From the time you start the game, the experience, the game play, the design and the ending of the game. From the two important impressions I can say that I have made a good attempt at catching the students attention.

- **Onboarding Experience**: A smooth and engaging onboarding process can significantly impact day 1 retention. Clear tutorials, easy-to-understand gameplay mechanics, and immediate rewards or achievements can hook players early on [Gup24].
- **First Impressions**: The initial gameplay experience should be enjoyable and rewarding. Players should quickly understand the game's core mechanics and feel motivated to continue playing [Gup24].

The onboarding experience in Codonia has clear instructions, narrative that gives hints for the levels and game mechanics that are easy to understand. This feeds well with the first point of onboarding experience. Since the players who played the demo for Codonia were confident enough to finish the game in less than 10 minutes.

They understood and were able to give accurate answers about the lesson plans. Finally, they were interested to know when new levels will be coming out. First impressions were important for Codonia, since if they did not like it from the start there was nothing that would drive them to continue playing the game. The point was to give them an enjoyable experience where they felt like they were learning.

This has been proven true since the first impression of the game has been overall positive according to Figure 6.4 where the students who played the demo liked the game design, the level design and the characters.

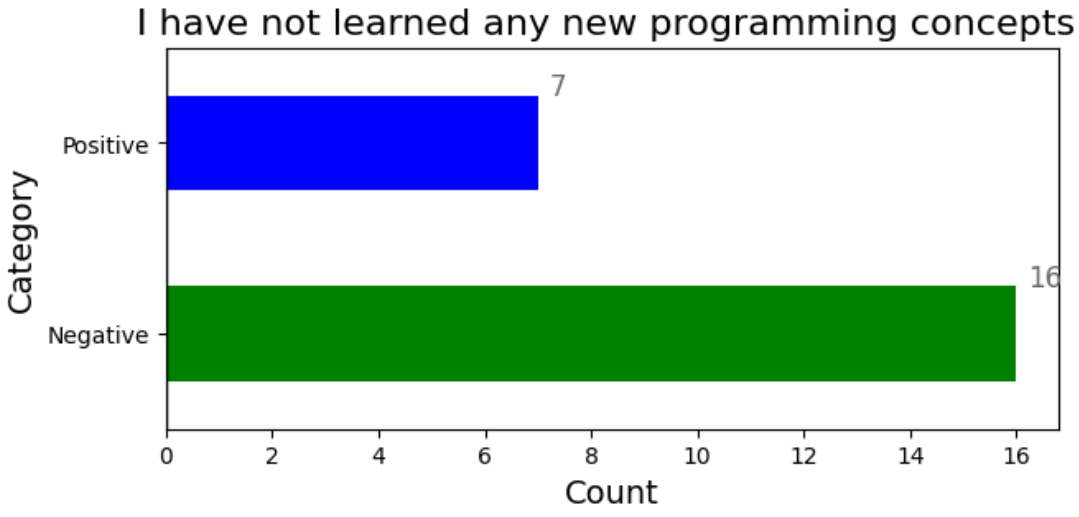

Figure 6.8: Results for students who have learned a new concept

The second point that I believe supports RQ2 is Figure 6.8. The statement that is negatively phrased "I have not learned any new programming concepts" has come up with a lot of negative responses. This means that 16 students have learned a new concept. Learning new ideals usually means that the students were paying attention while playing the game.

This also fits well with the Figure 6.2 where a high number of students were engaged with the game. Understanding new concepts gives new self-found confidence to most students no matter the age or year of study. Self-confidence among students is significantly influenced by their learning experiences. When students grasp new concepts and achieve academic success, their self-confidence grows.

This positive self-image enables them to handle challenges more effectively, take risks, and persist in the face of difficulties [Lon24]. Since there are also other research papers who support this argument I can safely say that RQ2 has been approved. Continuing their education through a system that encourages gamification can actually have a higher result in self-perceived confidence.

Students with high self-confidence are more likely to set ambitious goals, believe in their abilities, and strive for excellence [Lon24]. This is particularly true for students who have taken interest in a subject before they learned it through their curriculum. Students who know more about a subject have a higher confidence that drives them to learn more. Codonia can be a stepping stone for children who want to learn programming or continue in the field of computer science. They can have a basis of programming and what they can create with it. Through the experiment, I had introduced a statement to see if there is a chance for the children to get interested in programming through game play.

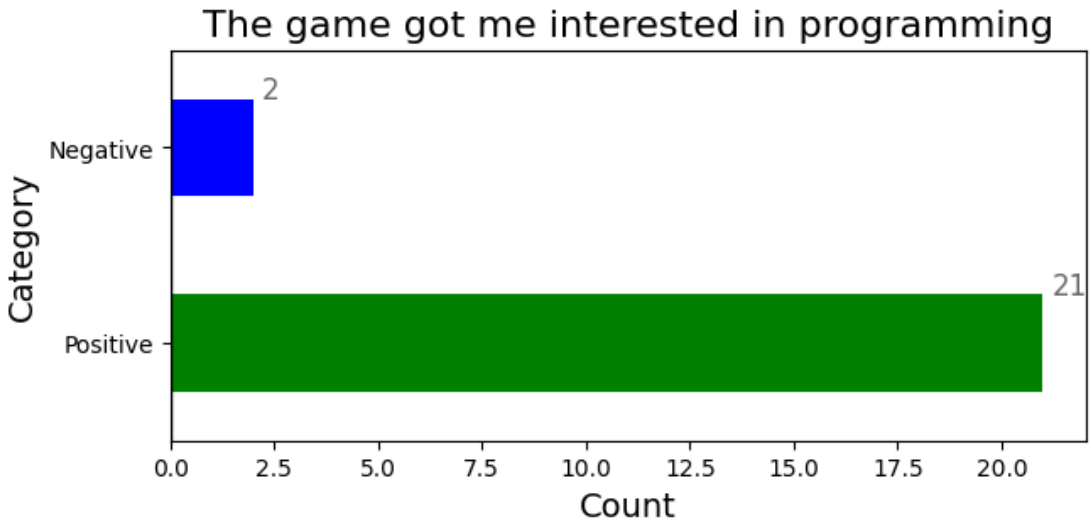

Figure 6.9: Results for students who got interested in programming

From the Figure 6.9, where 21 out of 23 students who played the game, we can confirm that this game sparked an interest in programming. Interestingly enough, working on Codonia and seeing the results has given me fresh motivation to keep working on the game. The hardest challenges I see for the future of my video game is constantly upgrading, analyzing and fixing it by myself. Codonia has the potential to become a large platform with many lessons. This can be achieved by creating new lesson plans in a certain category. For example, if we have five types of different game levels, you can start to put the lesson plans in those game designs. This creates a sense of unity in the design and look of the game, at the same time it eliminates the worry of constantly trying to design a creative level. The categories would include drag and drop, writing code, finishing the definition, matching objectives and so on.

After defining the categories, each lesson plan could go into a different one that fits it best. Some levels are harder so they might need more practicing with drag and drop, while others would be easier so you can only have a conversation with a character. This all depends on the outlook of how hard we want the levels to be.The best solution I have found is to have an easy level followed by a hard one and vice versa. This creates a roller-coaster effect where they can not predict exactly how hard or easy the next level is going to be. Although, for the start of the levels the levels will be easy for a more enjoyable experience while they learn about the game mechanics.

Having only played video games that are primarily for personal computers, I do not have a lot of experience how a device should feel when gaming on it for a long period of time. Taking the time to research user interaction and experience will be a big part of continuing developing Codonia. Since phones come in different sizes there has to be some kind of rule book or research that talks about the ergonomics of video games.

Button sizes and placements are the biggest factors to make the game a success. If the user has clear understanding where the button is, what to press next, and read clearly the next steps, the experience will be more enjoyable. User experience through game design has been a large conversation through the feed back. Without the worry of understanding the user interface, the player will be able to dedicate more time to reading and playing the game.

In **console gaming**, in our case mobiles, you are working with limited equipment capacities and explicit particulars. Obviously, this is more constraining than the PC's movable abilities. Be that as it may, it likewise makes it simpler for originators. It is building a game inside specific parameters and doesn't make up for contrasts in video cards or processors [PVV$^+$20].Consoles have points of interest over PCs, and they are anything but difficult to utilize, don't require redesigns, a large number of our companions additionally have consoles. That is the reason it is anything but difficult to play with companions, they are commonly less expensive and gratitude to remote controllers we can have an increasingly dynamic encounter [PVV$^+$20].

PC gaming objective is to structure extraordinary ongoing interaction yet need it to look incredible and run easily as well. The video card, the sound card, the processor, and a lot more pc segments can differ considerably from the low-end tech gamer to the very good quality tech side of the crowd. The game needs to play well over the entire range [PVV$^+$20].

There are always advantages and disadvantages to creating a mobile game. Having a transportable hand-held device where you can play a game anywhere is a huge advantage. You are able to pass the time and still learn in your free time. Although, that can be enjoyed for a few hours until the device has to be recharged. So far, there are positive results that Codonia can become a game that children can play in any location they wish to.

Creating a video game in a new platform with little documentation and limited tutorials while trying to turn it into a learnable software has been the most daunting task. Godot is still counted as a new software and is constantly being updated. Libraries keep changing, the program itself changes its UI ever few months. If you learn how to create a certain idea, after a few months that solution is deprecated. It would be easier if I worked in a more stable environment with less changes, although there might be a chance the possibilities would not be the same. I have chosen to teach GDScript through Godot and this has become an interesting learning curve for me as well.

My current limitation is the small time frame I have to constantly learn Godot and add new features to the game. Since the idea for Codonia was formulated as an idea for my thesis, I did not have much time to fully immerse myself into the Godot universe. Studying only relevant topics that enabled to turn my ideas into levels. When it comes to the study itself, I believe I need to keep researching what makes my game balanced in the topic of learnability versus game play. This would also include finding out if the ideas I have are possible to re-create in Godot. Continuing to keep making demos, adding new levels, new features and testing it on new groups of children is currently the only scenario I can think of to continue my study.

There is no time frame as to when the study will be complete enough for the game to release. I anticipate it would take six months to thoroughly complete a second demo. This would include adding new features that I have not had the time to learn or include in the levels. One important aspect I have to think about is the main platform this game will be played in. Mobile game characteristics are different to playing with keyboard and mouse. So, the ergonomics contribute significantly to the experience of the game. Having a comfortable hand-held device ensures intuitive interactions, longer play time and enhances the immersion of how far a child will dive into the story.

7 Conclusion

In conclusion, video games have grown from mere entertainment to being powerful tools in education. This revolutionary method can be used to teach children programming from an early age. The study explored a new research area where video games are used for instructional purposes. Video games provide constant feedback and use a reward system that encourages children to continue learning. Video games like Code Combat and RoboCode have successfully gamified education, making learning more interactive. The merge of video games and learning new skills is a fast-approaching method that can be used more frequently in the future.

It's important to continue introducing programming into the everyday lives of children who are interested in this field. The study proves that you can gain programming knowledge while playing. The main objective of the research was to demonstrate that video games can be used to learn programming, encouraging positive results from the design concept and creating an immersive environment that can teach debugging, problem-solving, and logical thinking.The feedback from the children showed that they were highly motivated, encouraged, and ready for new, harder levels. The key takeaway of the thesis is that video games can be a powerful tool for education. By integrating lessons with a combination of storytelling, interactivity, immediate feedback, and reward systems, we create an environment that is safe, fun, and informative.

With children growing up in a world increasingly dominated by technology, it's more important than ever to teach them the correct way to use it. Incorporating programming into their daily routine through video games ensures the effectiveness of education at a high standard. Children will be excited by learning new concepts, sharing them with their friends, and developing new logic and problem-solving skills. This gives them passive life skills, where they learn how to communicate with others in group projects or debugging without complete frustration that will help in their future educational career.

The study proved that digital evolution in the educational system is not just about traditional teaching methods but also leveraging new tools to engage students. Video games have the power to become a learning tool that combines entertainment and learning to create the ideal environment for teaching programming. Video games can evolve from the purpose of pure entertainment into an educational platform. The first research question, are learners more motivated when learning with a game, has been proven right through the figures that I have showed positive data. The second research question, Is the self-perceived competence higher with learning with a game, has also been proven from the data that shows that students would play the game again if there are new levels coming. The third research question, does gamification have a higher learning outcome than traditional teaching has yet to be determined. There are two data figures one which has a high acceptance that they would rather learn from a game rather than a textbook.

Through the stages of effective learning—which include fail-safe environments, motivation, learner automation, and so on [Eng23] children are able to feel safe to fail until they succeed.From these stages, we can see results because they cover the most important topics for teaching such as fail-safe environment where children can experiment fearlessly, knowing that failure is a stepping stone to success. Video games provide a safe space where trial and error lead to mastery. Motivation through immediate feedback where gamified learning offers instant feedback. Whether it's solving puzzles, debugging code, or completing quests, players receive real-time responses that keep them motivated. Engagement through narrative in which video games immerse players in captivating stories. As they progress through game worlds, children forget they're learning—they're simply part of an exciting adventure. The digitized era, where technology permeates every aspect of our lives, sees video games stand out as powerful educational tools. The American Psychological Association (APA) conducted a comprehensive review of research on video game effects, concluding in a paper that highlighted the positive impact of video games on learning, health, and social skills [Kir02]. Video games can be turned into a friendly tool that will continue to nourish and raise children with higher level knowledge in programming, collaboration, and problem-solving skills.

Codonia provides an immersive and interactive platform for children to learn programming. By integrating learning into the game play, it makes the process of learning programming concepts engaging and enjoyable for children.

Bibliography

[ACA+11] A. P. Ambrósio, F. M. Costa, L. Almeida, A. Franco, and J. Macedo. Identifying cognitive abilities to improve cs1 outcome. In *Proceedings of the Frontiers in Education Conference (FIE)*, pages F3G–1–F3G–7, 2011.

[BGGAT19] M. U. Bers, C. González-González, and M. B. Armas-Torres. Coding as a playground: Promoting positive learning experiences in childhood classrooms. *Computers & Education*, 138:130–145, 2019.

[BPLO18] Rafael Brito, Luciana Pinochet, Everton Lopes, and Marcelo Oliveira. Development of a gamification characteristics measurement scale for mobile application users. *Internext*, 13(1):1–16, 2018.

[CAG10] CAGcast. Episode 209, 2010. Podcast, August 19. Retrieved August 22, 2010 from `http://podcast.com/show/4388/`.

[Con25] ConcernedApe. History of stardew valley. `https://www.stardewvalley.net/press/`, 2025. Accessed: 03 January 2025.

[Eng23] Cambridge English. How do video games provide effective learning | cambridge english, 2023. (Accessed on 04/26/2024).

[Eng24] Godot Engine. Introduction to physics. `https://docs.godotengine.org/en/stable/tutorials/physics/physics_introduction.html`, 2024. Accessed: 2024-12-27.

[For23] ForeverGeek. 14 video games that teach you coding, 2023. (Accessed on 04/25/2024).

[Gup24] Sakshi Gupta. Mobile game retention benchmarks: An industry breakdown. *Nudge Blog*, July 2024.

[Har15] B. Hartanto. Enhancing the student engagement in an introductory programming: A holistic approach in improving the student grade in the informatics department of the university of surabaya. In R. Intan, C. H. Chi, H. Palit, and L. Santoso, editors, *Intelligence in the Era of Big Data. ICSIIT 2015 (Communications in Computer and Information Science)*, volume 516, pages 493–504, Berlin, Germany, 2015. Springer.

[HC15] D. Horton and M. Craig. Drop fail pass continue: Persistence in cs1 and beyond in traditional and inverted delivery. In *Proceedings of the SIGCSE*, pages 235–240, 2015.

[How23] HowLongToBeat. How long is stardew valley? `https://howlongtobeat.com/game/34716`, 2023. (Accessed on 03 January 2025).

[Kap12] Karl M. Kapp. *The Gamification of Learning and Instruction: Game-Based Methods and Strategies for Training and Education.* Pfeiffer, 2012.

[Kir02] John Kirriemuir. Video gaming, education and digital learning technologies: Relevance and opportunities. *D-Lib Magazine*, 2002. (Accessed on 04/26/2024).

[Lon24] Rafiq Ahmad Lone. Self-confidence among students and its impact on their academic performance: A systematic review. *International Journal of Creative Research Thoughts (IJCRT)*, pages 1–10, 2024.

[MDP22] MDPI. The use of educational games in programming assignments: Sql island as a case study, 2022. (Accessed on 04/25/2024).

[MRF19] Rodrigo Pessoa Medeiros, Geber Lisboa Ramalho, and Taciana Pontual Falcão. A systematic literature review on teaching and learning introductory programming in higher education. *IEEE Transactions on Education*, 62(2):77–90, 2019.

[NH23] Kibbeum Na and Kwanghee Han. How leaderboard positions shape our motivation: the impact of competence satisfaction and competence frustration on motivation in a gamified crowdsourcing task. *Internet Research*, 33(7):1–18, 2023.

[oCiE23] Journal of Computers in Education. Studying the effects of educational games

on cultivating computational thinking skills to primary school students: a systematic literature review. *Journal of Computers in Education*, 2023. (Accessed on 04/26/2024).

[oE21] Harvard Graduate School of Education. Can video games help kids learn? | harvard graduate school of education, 2021. (Accessed on 04/25/2024).

[oSE20] International Journal of STEM Education. Game-based learning in computer science education: a scoping literature review | international journal of stem education | full text, 2020. (Accessed on 04/25/2024).

[PGvWB17] Giani Petri, Christiane Gresse von Wangenheim, and Adriano Ferreti Borgatto. Meega+, systematic model to evaluate educational games. *Encyclopedia of Computer Graphics and Games*, 2017.

[PGvWB23] Giani Petri, Christiane Gresse von Wangenheim, and Adriano Ferreti Borgatto. Meega+: A model for evaluating educational games, 2023. Accessed: 2024-12-26.

[Pix24] PixelFrog. Tiny swords asset pack. `https://pixelfrog-assets.itch.io/tiny-swords`, 2024. Accessed: 2024-12-27.

[PT22] Lev Poretski and Anthony Tang. Press a to jump: Design strategies for video game learnability. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, 2022. (Accessed on 04/27/2024).

[PVV+20] Jatin Pakhrani, Ashin Varghese, Shubham Vyas, Himanshu Purohit, and C.H. Patil. Comparative study of pc and gaming console for video games. *International Journal of Engineering Research & Technology (IJERT)*, 8(05):1–4, 2020.

[Res23] ResearchGate. Informing age-appropriate ai: Examining principles and practices of ai for children, 2023. (Accessed on 04/27/2024).

[RRR03] Anthony Robins, Janet Rountree, and Nathan Rountree. Learning and teaching programming: A review and discussion. *Computer Science Education*, 13(2):137–172, 2003.

[Rug13] Dana Ruggiero. Video games in the classroom. In *FDG 2013*, pages 39–49. SpringerLink, 2013.

[RWJS18] Chrystal Rutledge, Anne Walsh, David Johnson, and David Sideris. Gamification and self-determination theory. `https://selfdeterminationtheory.org/wp-content/uploads/2020/10/2018_RutledgeWalshEtAl_Gamification.pdf`, 2018. (Accessed on 03 January 2025).

[SB23] William H. Stewart and Kwanwoo Baek. Analyzing computational thinking studies in scratch programming: A review of elementary education literature. *International Journal of Computer Science Education in Schools*, 6(1):35–58, 2023.

[SEC13] Daniel J Schumacher, Robert Englander, and Carol Carraccio. Developing the master learner: Applying learning theory to the learner, the teacher, and the learning environment. *Academic Medicine*, 88:1635–1645, 2013.

[Ser19] Katrina Serrano. The effect of digital game-based learning on student learning: A literature review, 2019. (Accessed on 04/26/2024).

[Ste23] SteamDB. Stardew valley charts. `https://steamdb.info/app/413150/charts/`, 2023. (Accessed on 03 January 2025).

[SYAN20] Reza Edwin Sulistyaningtyas, Prasetyo Yuliantoro, Dian Astiyani, and Chomisah Nugraheni. A literature review of coding for early childhood. *EUDL*, 10:2311709, 2020.

[TL22] Weimin Toh and Fei Victor Lim. Learning in digital play: a dual case study of video gamers' independent play. *Research and Practice in Technology Enhanced Learning*, 2022. (Accessed on 04/25/2024).

[Uni24] Lawrence Technological University. Importance of game difficulty, 2024. Accessed: 19-December-2024.

[vRZ17] R. van Roy and B. Zaman. Why gamification fails in education—and how to make it successful. introducing 9 gamification heuristics based on self-determination theory. In *Serious Games and Edutainment Applications. Vol II*, pages 485–509. Springer, Cham, Switzerland, 2017.

[WOR24] BI WORLDWIDE. What is gamification?, 2024. Accessed: 2025-01-03.

[XS22] Stelios Xinogalos and Maya Satratzemi. The use of educational games in programming assignments: Sql island as a case study. *Applied Sciences*, 2022.

[Yap19] Bernard Yunkang Yap. What make gamers loyal to game publishers: Examining brand loyalty in the video game industry, 2019.

[Yil18] Ebru Yeniman Yildirim. The positive effect of digital game based programming on students. In Terence Andre, editor, *Advances in Human Factors in Training, Education, and Learning Sciences*, Cham, 2018. Springer International Publishing.