TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

JOOEL KORPI

TEACHING PROGRAMMING TO CHILDREN THROUGH GAMES

Master's thesis

Examiner: Prof. Hannu-Matti Järvinen
Examiner and topic approved by the
Faculty Council of the Faculty of
Natural Sciences on
05 Feb 2014

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Science and Engineering
KORPI, JOOEL: Teaching programming to children through games
Master of Science Thesis, 50 pages, 3 Appendix pages
December 2014
Major: Software Engineering
Examiner: Prof. Hannu-Matti Järvinen
Keywords: education, programming, games, gamification

Today, programmable devices are part of everyday life. Children grow up learning how to use them, but do not know how to modify or create them. Traditional classroom teaching methods do not usually focus on the motivational side. On the other hand, children have the innate tendency to play games which motivate on their own. This thesis looks into creating an educational game for programming and analysing its design.

Literacy review gives quite a comprehensive list of game elements found in most games. The techniques used in entertainment games to motivate players can also be used in educational games. Examples are presented to provide a sense of today's educational games focusing in mathematics and programming. The game was analysed by comparing the lists of game elements with the game design. In addition, a specific game design assessment framework designed partly for educational games was used.

In this thesis, a working educational game was created for learning programming. The user testing gave positive feedback which enforced the idea of successful implementation of an educational game. Data about games improving the attitude towards programming was also acquired. Data connected with cognitive improvement was inconclusive. Actual research, based on the improvement of the attitude towards learning programming or cognitive skills, is encouraged.

TIIVISTELMÄ

TAMPEREEN TEKNILLINEN YLIOPISTO
Teknis-luonnontieteellinen koulutusohjelma
KORPI, JOOEL: Ohjelmoinnin opettaminen lapsille pelien avulla
Diplomityö, 50 sivua, 3 liitesivua
Joulukuu 2014
Pääaine: Ohjelmistotuotanto
Tarkastajat: Prof. Hannu-Matti Järvinen
Avainsanat: koulutus, ohjelmointi, pelit, pelillistäminen

Ohjelmoitavat laitteet ovat nykyään osa kaikkien arkipäivää. Lapset oppivat kasvaessaan käyttämään niitä, mutta muokkaaminen ja uuden luominen jäävät oppimatta. Perinteiset opetusmetodit eivät yleensä keskity motivoimaan oppilaita. Toisaalta lapsilla on luontainen tapa pelata pelejä, jotka itsessään ovat motivoivia. Tämä työ perehtyy ohjelmointia opettavan opetuspelin luontiin ja sen suunnitelman analysointiin.

Kirjallisuus tarjoaa kohtuullisen kattavan listan pelielementeistä, jotka löytyvät useimmista peleistä. Viihdepeleissä motivointiin käytettyjä tekniikoita voidaan hyödyntää myös opetuspeleissä. Nykyisten matematiikkaan ja ohjelmointiin perehtyvien opetuspelien taso esitellään esimerkkien kautta. Peliä analysoitiin vertaamalla kirjallisuudesta löydettyjä pelielementtejä ja pelisuunnitelmaa. Lisäksi käytettiin osaksi opetuspelisuunnitelmia varten kehiteltyä analysointikehystä.

Työssä kehitettiin toimiva opetuspeli ohjelmoinnin oppimiseen. Käyttäjätestaus antoi positiivista palautetta, joka vahvistaa päätelmäämme onnistuneesta opetuspelistä. Testaus antoi myös osviittaa pelien myönteisestä vaikutuksesta lasten suhtautumiseen ohjelmoinnin opettelemiseen. Kognitiivisten taitojen parantumisesta ei saatu käytettäviä tuloksia. Tarkempi tutkimus keskittyen ohjelmoinnin oppimiseen liittyvien asenteiden tai kognitiivisten taitojen parantamiseen on suositeltavaa.

PREFACE

This thesis was written for Nokia Corporation at Nokia Student Innovation Lab. Design and implementation of the game was made in co-operation with other Nokia employees, but the rest of this work was done by me.

I would like to thank my supervisor and manager Timo Sorsa for giving insightful comments throughout the process. I would also like to thank my colleague Marion Boberg for discussions concerning psychology and playfulness. Futhermore, I would like to thank everybody who worked with me in this the project, especially the project manager Lauri Ilola. Special thanks to everybody commenting and proof-reading the thesis in these last months.

And finally I am thankful for my friends and family. Without you, I would not be here today.

In Tampere, Finland, on 14th November 2014

Jooel Korpi

TABLE OF CONTENTS

TERMS AND DEFINITIONS

ICT	Information and Communications Technology
SGDA	Serious Game Design Assessment framework
SIL	Nokia Student Innovation Lab
STD	Self-Determination Theory
SVG	Scalable Vector Graphics
UI	User Interface

1. INTRODUCTION

Children have always played multitudes of games such as tag. As they grow older, children switch from free form play to more rule focused games. Card, board and computer games have taken much of people's free time. Why not use the computer games to teach programming like tag helps with motoric skills?

Today, people use computers and other digital devices daily for work and leisure. Children are born into this digital world and learn how to use tablets and web browsers at a very young age. However, they do not learn how programs work and how to create them. In Finland, a new curriculum for elementary school comes into effect in the fall of 2016 and it will include programming [26, p. 6]. For these reasons, learning programming is a contemporary and interesting research field.

This thesis looks into how different levels of game elements could be applied in educational context, especially when learning to program. With this target in mind, an educational game using game elements found in literacy is created. The game will be analysed with an assessment framework and user tested with actual elementary school pupils. The possible usage of gamification in educational context is also discussed.

First in Chapter 2 the focus is on the different levels of game elements and how to classify them. The next chapter looks into pedagogical and psychological theories, focusing mainly on motivation, which enforce the idea of games as an educational medium. After this literacy review, two educational games are presented to get a sense of general level on educational games. Chapter 4 starts with the normal ways of learning programming and continues by presenting the examples of educational games and gamification focusing on programming. In Chapter 5, the focus is on the actual design and implementation of an educational game which was created as a part of the thesis. The analysis of the game design and results from user testing done in a normal elementary school in Finland, are at the end of the chapter. General discussion is in Chapter 6 and the chapter ends with possible further studies. The final chapter, Chapter 7 contains conclusions. Appendix A contains the questionnaires used in the testing.

2. GAMES AND GAMIFICATION

In order to understand how games and gamification could be used in learning programming, they have to be defined. This chapter is meant to give background information about games and different subsets of games. Some of the definitions are not universally fixed, but the definitions given in this chapter will be used for the rest of the thesis.

One attempt to scope different types of games is made by Marczewski and it is displayed in Table 2.1 [19]. Game Thinking means doing something with fun and playfulness in mind. Game Elements are the concrete parts of game that can be distinguished in several different games. For example points, boss fights and leaderboards are game elements. Game Play is how the player interacts with the game world. It is more abstract than game elements and can be very unique to a game. The last one, Just for Fun, means that the game is designed with only its entertainment value in mind.

Table 2.1: Marczewski's levels of game influence. [19]

	Game Thinking	Game Elements	Game Play	Just for fun
Gameful Design	✓	✗	✗	✗
Gamification	✓	✓	✗	✗
Serious Game / Simulation	✓	✓	✓	✗
Game	✓	✓	✓	✓

In this chapter, Marczeski's list is gone through from the bottom up. The first Section 2.1 focuses on the general definition of games. Section 2.2 concentrates on the normal game development with special focus in the design aspects. Next, the educational aspects are added in the form of serious games in Section 2.3. Chapter ends with the definition of gamification in Section 2.4. In this thesis, the focus will be on the levels of game influence which have at least some level of interactivity. For this reason there will not be more discussion about Gameful Design.

The focus is on the digital medium for games and gamification such as video games and websites. Most of the concepts work also with other games like board games, but the examples and the focus in this thesis will be on digital versions.

2.1 Games in general

Everybody has played some kind of game in his life. It could have been basketball in mandatory school sports or solitaire during the quiet hours at the office. However, giving exact definition to a game is more difficult. For example, do you need score in a game? There is score counting in basketball and solitaire, but for example chess just counts winning or losing. Chess has a scoring system involved when played in tournaments, but the basic game does not have it. Next some of the definitions for a game are presented and the definition used in this thesis is selected.

In their book of game design, *Rules of Play: Game Design Fundamentals*, Salen and Zimmerman define game as "A system in which players engage in an artificial conflict, defined by rules, that results in a quantifiable outcome." [29, p.80]. In his book, *The Gamification of Learning and Instruction*, Kapp extends previous definition to the following: "A game is a system in which players engage in an abstract challenge, defined by rules, interactivity, and feedback, that results in a quantifiable outcome often eliciting an emotional reaction." [14, p. 7]. Philosopher Suits gives a shorter definition: "Playing a game is the voluntary attempt to overcome unnecessary obstacles." [30]. One of the problems with this definition is that it defines what playing the game is and not the actual game itself [29, p. 77].

The definition used in this thesis is extended from Zimmerman and Salen's definition by adding voluntarism and the need for meaningful choices: "A game is a system in which players voluntarily engage in an artificial conflict, defined by rules and meaningful choices, that results in a quantifiable outcome." Voluntarism was added because it is usually associated with games and for example McGonigal uses it as one of the four defining traits of a game and Suits used it in his definition of playing a game [20, p. 21]. Meaningful choices give a sense of importance and autonomy. This is partly inspired with Meier's[1] quotation: "A game is a series of interesting decisions." [21]. In a simple game like Snakes, the choices are limited to turning left or right. Nevertheless, it still has a meaningful impact on the game, because turning the wrong way might end the game. This definition rules out at least games of chance like slot machines, because the only meaningful choice player does is to play or not. So this definition does not envelop all games, but should include games which could be useful in educational context.

[1] Designer of Civilization series

Table 2.2: Reasons to play. [28, p. 1-8]

Reason	Explanation
To get a challenge	When a player has faced a challenge and overcame it, he has learned something which enriches him.
To socialize	Social experiences and playing with friends and family is one of the largest motivating factor for playing games.
To have a dynamic solitaire experience	Reading books and watching movies are solitaire, but not dynamic experiences.
To get bragging rights	Players who might not have much to brag about in their ordinary lives can have great boost in self-confidence when they beat a difficult game.
To have an emotional experience	Like in every type of entertainment, players might be wanting an emotional experience.
To fantasize	Many players want to experience more glamorous world and events than the ones they face ordinarily.

2.2 Game elements and game design elements

In his book, *Game Design: theory & practice*, Rouse talks about gameplay and game design [28]. He defines gameplay as the part of games which is not found in other art forms. It is the interactivity between the player and the game world. The player interacts in different ways with his surroundings and his actions impact the game world. Game design is what determines gameplay. In game design it is decided what tools the player has for interaction and what ramification his actions will bring. [28, p. 1-19] Other game designers concur with Rouse. One of these is Crawford who in his book, *Chris Crawford on game design*, says that everything, especially cosmetics, you put into a game should support the gameplay [7, p. 108]. Crawford uses the noun 'interaction' more than gameplay, but uses them as synonyms.

Rouse also collected a list of elements important to know when designing games. Firstly he goes through reasons players want to play which can be seen in the Table 2.2. In the table, there is first the compact reason why people play and in the second column a more verbose explanation for the reason. The second table, Table 2.3, defines the expectations of people when they start to play a game. Players are probably not conscious of these reasons and expectations and there are probably more which are not listed here. This is because in the end, the players do not know what they want, but they know it when they see it.

Table 2.3: What players expect from a game. [28, p. 8-18]

Reason	Explanation
To have a consistent world	Players come to expect a certain result in the game and are frustrated if for no visible reason it does not happen.
To understand the game world's bounds	Player wants to understand what is possible and what is not in the game.
For reasonable solutions to work	After playing for some time, the player feels that he knows game worlds bounds and knows what kind of tactics work. If tactics will not work to a similar problem, the player is left frustrated.
To have a direction	Good games should allow players to do want they want, but also give a goal to direct their playing. Sandbox styled games are an exception where players create their own goals.
To accomplish a task incrementally	Players want subgoals on their way to their ultimate goal so that they know the are on the right track.
To be immersed	Players want to forget that they are playing a game and feel connected to the game world.
To fail	Connects to the reason that players want a challenge. If they do not fail, it does not feel like a game.
To have a fair change	There should be a small theoretical possibility to go through the entire game on the first try. Problems which can only be solved by trial and error should be avoided.
To not need to repeat themselves	A solved puzzle or problem should not be reused unless it is very rewarding to solve or the rewards from solving it are very different.
To not get hopelessly stuck	Player should not be left in a situation where he cannot continue, especially if he himself does not know it.
To do, not to watch	Players want an experience they cannot have through movies or television, so game should not be filled with cut-scenes.

After going through the game elements, Rouse talks about game design elements that make up a great game. He concurs that there is no definite answer and the list in the book is not complete. The elements he listed were: unique solutions, non-linearity, modeling reality, teaching the player and input/output. *Unique solutions* mean that players should be able to come up with their own solutions to the problems, which even the designer had not figured out. *Non-linearity* means that there should be meaningful decisions to be made, which affect the outcome of the events further in the game. *Modeling reality* means that the designer has to think about what level of reality is really wanted in the game. Reality based games provide worlds which players are instantly familiar with at least to a degree. Still, it also brings expectations from players to be able to for example jump, swim and crouch. Adding these elements would require more resources and might lessen the main play experience.*Teaching the player* is the way that players learn the game. Today manuals are not a good way, because players have strong desire to just start playing the game. The designer should come up with a good and interactive way to introduce the player to the game world. This usually means different kinds of tutorials at the start of the game. Through the game, the player is gradually given new abilities starting with basic movement. *Input/output* means the tools that player uses to physically communicate with the game, for example controller, keyboard and mouse. Rouse thinks that the designer should not be creative with the controls because then player immersion with the game is hindered. Established control systems might not be the best ones, but they are the ones players already know. He feels that each of these design elements deserves serious thought when designing a game. [28, p. 121-145]

2.3 Serious games

One of the subtypes of games is serious games as displayed in Table 2.1. According to Marczeski serious games differ from ordinary games, because they are not developed just for their entertainment value. This concurs with the definition in Michael's and Chen's book, *Serious Games: Games That Educate, Train, and Inform* [22]. Their definition is "A serious game is a game in which education (in its various forms) is the primary goal, rather than entertainment" [22, p. 17]. It is important to realize that serious games are not just "edutainment" which tried to forcibly insert educational elements into games by for example memorizing facts [22, p. 1]. Even though serious games have education in its definition, educational games, at least in school education sense, are only a subgroup of serious games. Serious games are also going to have a bigger role in the classroom. The book contains several surveys. One of the questions was "Do you think serious games will become a standard part of education/training curriculum?" and 95 percent of 63 respondents answered yes [22,

p. 119]. Later in this work, there will be multiple examples of serious games which belong to the educational games subgroup.

Serious games also consist of games for military training or informing local citizens of political situations. For example, U.S. Army has created *America's Army* for boosting recruitment and pre-training. The game tries to present a realistic world, but sacrifices some for sake of entertainment. An example of political serious games is a strategy simulation game, *PeaceMaker*. Game tries to teach and educate Israeli and Palestinian teenagers so that a lasting peace could be achieved. [22, p. 55-57, 209-212]

2.4 Gamification

Gamification is usually defined as using game elements in a non-game context. For example, Oxford dictionary defines gamification as: "The application of typical elements of game playing (e.g. point scoring, competition with others, rules of play) to other areas of activity, typically as an online marketing technique to encourage engagement with a product or service." [1] However, this kind of definition is quite broad, lengthy and can be understood in different ways. A shorter definition is provided by Werbach and Hunter in their book, *For the Win*: "The use of game elements and game-design techniques in non-game contexts." There are also conflicting definitions like Kapp's: "Gamification is using game-based mechanics, aesthetics and game thinking to engage people, motivate action, promote learning, and solve problems." [14, p. 10]. The definition itself is not conflicting with the previous ones, but in his book Kapp uses gamification as an abstract term including for example serious games. The rest of this section tries to clarify the term as it is used in this thesis.

An alternative way to define gamification is to start with what it is not. Gamified system is not a game. It has many similarities with games and uses the same design principles, but the main goal is different. The idea is not to create a standalone game, but instead improve players' engagement for example in a website or in real life activities.

Gamification is a fairly new term and it is believed to have been used in current, broader sense for the first time by Pelling in 2002 [27]. Before that the word meant turning something not a game into a game [32, p. 25]. In 2010, gamification started to gather steam as can be seen in Figure 2.1. The elements of the gamification have been around longer than that and people have used it, but its terminology is still in infancy.

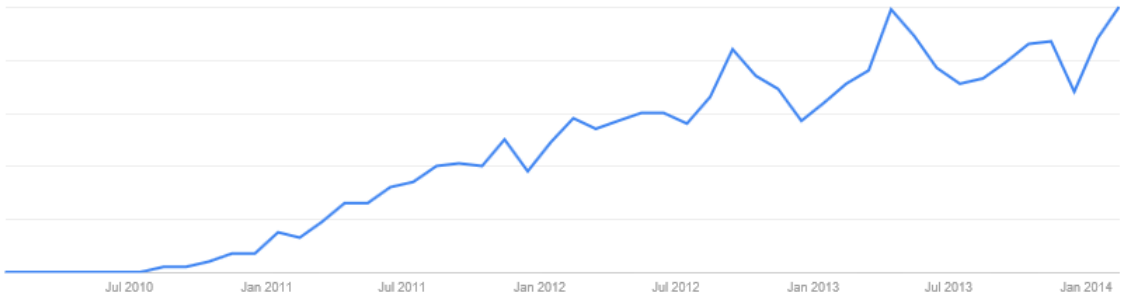


Figure 2.1: Google trend of gamification. [12]

Gamification is in use already in a multitude of places even if the term itself has not been used. For example, Atwood, co-founder of Stack Overflow[2], talked about the usage of gamification on his site in Gamification Summit 2012[3]. They never thought of it as its own concept, they were just "...trying to make things fun that should be fun anyway. Because you learn more when you are having fun.". Stack Overflow's gamification is defined around the concept of reputation. Users rise their reputation by upvoting each other's questions, answers or comments. Through reputation levels users are given more power on the site like deleting posts. The site has the ability to list users according to reputation, but it is not named leaderboard, because founders wanted community to make it such on their own. They just provide the numbers. [3]

[2]Question and answer site for programmers. Website: `http://stackoverflow.com/`.
[3]Annual conference for gamification held in San Francisco

3. PEDAGOGICAL VIEW OF EDUCATIONAL GAMES

This chapter looks in to psychological principles connected to games and play. The existing educational games and usage of games for other educational purposes are also discussed. The chapter begins with psychological theories, focusing on motivation, in Section 3.1. Section 3.2 contains different examples of games meant for educational purposes. Examples range from good design in both game and education sense to bad examples which do not excel in either one. In Section 3.3, the focus is on the games attributes which make them attractive to people and how they could be utilised for educational purposes. Whole Section 3.4 is about Serious Game Design Assessment framework which aims to be a generic model for analysing the design of serious games from the multiple points of view.

3.1 Psychology

3.1.1 Play in children's development

The positive effect of children's play has been an important theme for psychologists like Vygotsky and Piaget [13]. By playing children experience their own world and the world of others. In play, children can imagine being for example parents or firefighters. New neural connections are made while playing and in a way makes the children more intelligent. In addition, cognitive processes used in play are similar to learning. These are motivation, meaning, repetition, self-regulation and abstract thinking. [11, p. 8, 11]

Furthermore, spatial skill improvements gained from playing shooter video games are comparable to the effects of formal courses aimed at the same skills. Spatial skills play a large part when predicting achievement in science, technology, engineering and mathematics. Specific types of video games seem to enhance cognitive skills of different kind. Some of these can also be used in a real-world context. [13]

Vygotsky presented the idea of the zone of proximal development. It is the gap of children's actual development level and the level of development with help. The actual development level is measured by giving the child a variety of tasks with variety also in the degree of difficulty. The level with help is determined by doing similar tasks, but with the help of an adult or peers. Vygotsky proposes that an

essential purpose of learning is to create the zone of proximal development. Meaning that learning should awaken processes in the mind of the child that can be used only with the cooperation of others. These processes can then be internalized and used independently. [31, p. 84-91]

3.1.2 Self-Determination Theory

One of the most influential cognitive theories is the Self-Determination Theory (SDT) created by Deci, Ryan and their collaborators [32]. SDT says and provides empirical support for the proposition that everybody has fundamental psychological needs to be competent, autonomous and related to others. Satisfaction of these needs provides people with autonomous motivation. Thwarting these needs makes people feel that they are pressured to behave in a certain way or makes them demotivated. [9]

Autonomy

Autonomy is the capacity for and desire to experience self-regulation and integrity. Achieving greater autonomy is about internalizing and integrating external regulations over behaviour, learning to effectively control drives and emotions. Also maintaining intrinsic motivation and interest is vital in assimilating new ideas and experiences. More autonomous people exhibit greater engagement, vitality and creativity. In STD the concept of autonomy is sometimes used to refer to a motivational state, other times to an enduring motivational orientation or a fundamental psychological need. Which of these concepts are used depends on the problem at hand. In SDT whether people's motivation is more autonomous or controlled is far more important in making predictions than the overall amount or intensity of motivation. [9]

Competence

Competence, also called mastery, is the ability to effectively deal with outside environment. It could be for example learning to dance, to play an instrument or to solve difficult math problems. [32]

Relatedness

Relatedness involves social connections and the desire to interact with and be involved with other people. It is also manifest in the desire for higher purpose. [32]

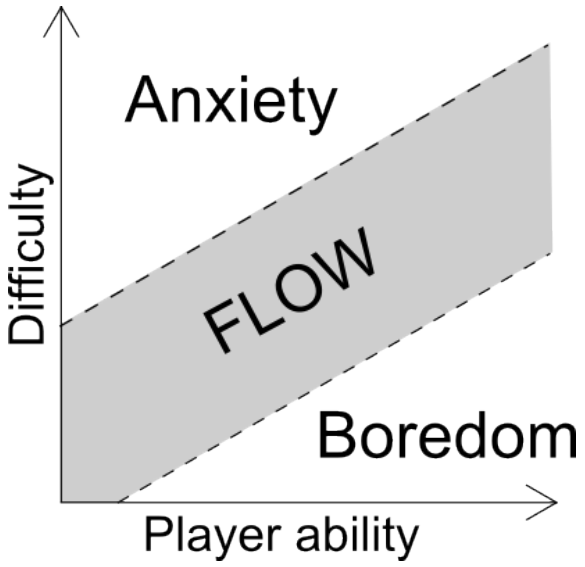


Figure 3.1: Flow is a state of mind between anxiety and boredom. [14, p. 72]

3.1.3 The Theory of Flow

Hopefully everyone has had the feeling of being so entranced in doing a task that time flies and the outside world seems to fade away. That state of mind is called flow. The term was coined by Csíkszentmihályi in 1975 and he defined it as: "the satisfying, exhilarating feeling of creative accomplishment and heightened functioning." [20, p.35]. Figure 3.1 shows how flow is a state situated between anxiety or frustration and boredom [14, p. 71]. Csíkszentmihályi also had more complex version which separated more mind states and it is illustrated in Figure 3.2. The center of the graph is the average of a person's challenges and skills in day to day life. To get to the flow is to develop higher skills and take on more difficult challenges.

The mental state of flow is usually achieved when playing a game or working on a challenging task in work or hobby. Csíkszentmihályi's examples were chess, basketball, rock climbing and dancing. The most important in flow inducing activities is that they are done for pure enjoyment and not for material gain or other external influence. In his opinion, games are obvious sources of flow, because they usually have self-chosen goals, personalized difficulty and continuous feedback. [14; 20]

Designing flow inducing elements to a game is in a way straightforward, because the elements are mostly the same as usually found in a good game. This does not mean that good games are easy to do. It just means that a good game usually already has flow inducing elements. Csíkszentmihályi lists six aspects: an achievable task, concentration, clear goals, feedback, effortless involvement, control over actions. It is hard to the developing team to get into the flow state by playing their own game, so playtesting is required to estimate the flow inducing qualities of a game. [14]

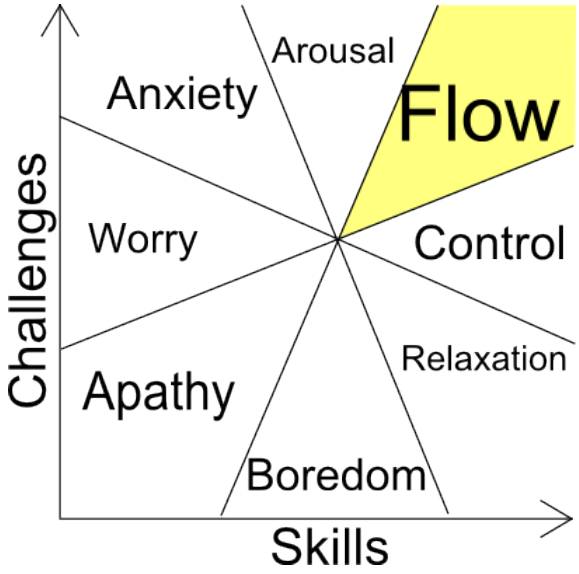


Figure 3.2: Flow is a state of mind when skills and challenges are higher than average. [8]

3.2 Examples of existing educational games

The examples in this section try to present the majority of current educational games. One of the criteria for selection was how easily they are found when searching for educational games in the Internet. There are better and worse educational games, but these ones present the current middle ground of educational games. They are not just question and answer games like the earliest educational games and have a level of gameplay. Still, in both games one fault is clearly evident. Design did not start from scratch to include both game and educational elements. The base game works well without the educational part, which is constructed on top of an existing game idea.

3.2.1 Math Lines

Math Lines is a game where the player controls a cannon with the ability to shoot balls with numbers. A string of balls with numbers is moving in a predefined circular path towards the goal. The player's task is to stop the balls from reaching the end by shooting balls towards the moving balls. If the shot ball hits a ball and their numbers sum to ten, then both of them disappear. A group of same numbered balls can be destroyed by shooting one of them. The player can target the cannon with mouse and shoot with the left mouse button. The ball to be shot can be interchanged with the next one using spacebar. A screen capture of the gameplay can be found in Figure 3.3. [24]

The balls are coloured, but not according to their value which makes the game more frustrating and hectic. The gameplay is also quite limited because the player

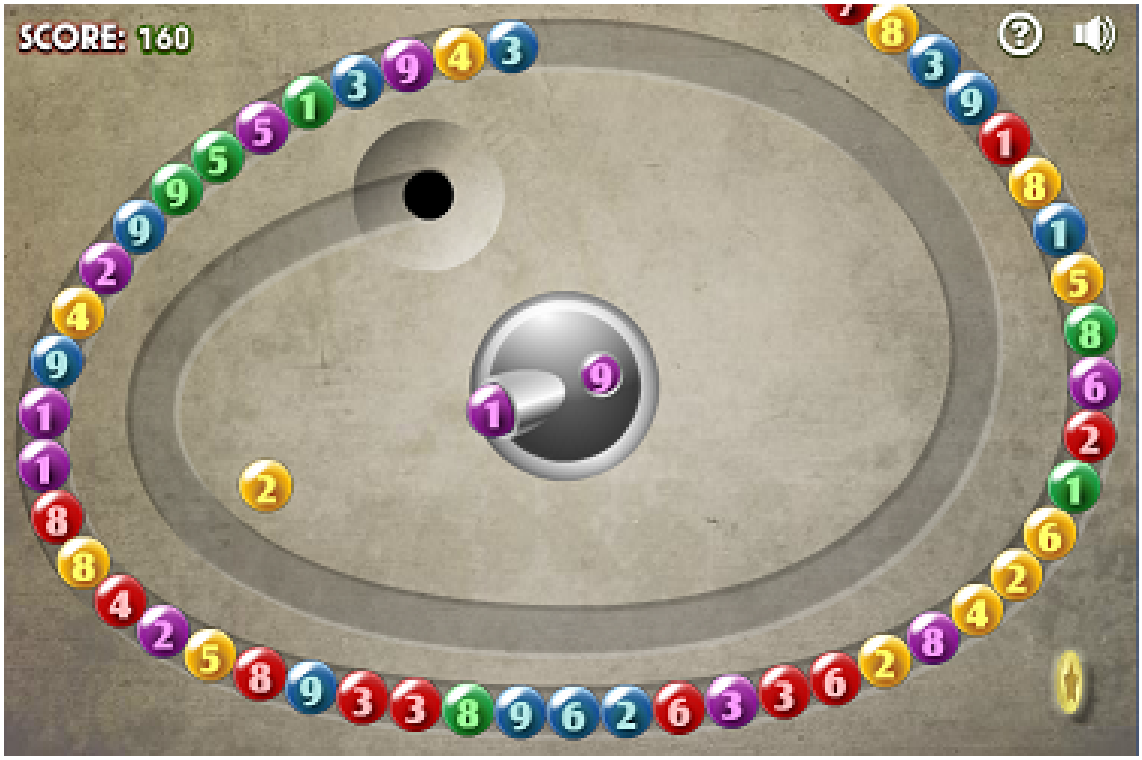


Figure 3.3: Screen capture of Math Lines. [24]

.

can only change which of the two balls to shoot in addition to the basic targeting and shooting. The educational goal of the game is to teach which two numbers sum to ten.

3.2.2 Math Man

Math Man is a Pac-Man clone, but on the contrary to the original, the main goal is to eat the ghosts in the correct order. Eating pellets gives points and the player has to avoid ghosts most of the time. In addition to the basic Pac-Man rules, ghosts have numbers and there are circles with question marks. Eating a question mark circle will display an equation at the bottom of the screen. The player can eat the ghost which has the answer to the equation. Level is passed when all of the ghosts have been eaten. A screen capture of the gameplay can be found in Figure 3.4. [16]

The gameplay of the game is quite simple like original Pac-Man. The equations on the other hand are not always simple and can contain any of the basic arithmetic operations. The best tactic seems to be to estimate the correct answer based on the numbers on the ghosts. The last ghost is always simple because there is no other option. Having only one option takes away the choice from the player which is an important part of the game at the both educational and gameplay point of view.

Figure 3.4: Screen capture of Math Man. [16]

With each level, the number of ghosts and terms in the equations are increased by one. Additionally, numbers are higher and equations have more multiplication and division than easier addition and subtraction operations. However, there is no change in graphics or basic game play mechanics which are quite basic. They might present new mechanics in the later levels, but usually flash games are small and present all of the mechanics in the start. The games educational focus is to improve estimation skills. Using estimation is a preferable tactic when compared with actually calculating the answer. Using estimation in the game could encourage players to use estimation more for example when going to grocery shopping.

3.3 What makes games attractive?

Linehan et al. write about designing educational games and how to merge the goals of education and game design. They feel that similar approach needs to be used when designing educational games and entertainment games. Games not designed this way can focus too heavily on the educational side and lose the interesting gameplay. In order to find out important elements when designing educational games, they compiled a list of features seen in the most successful entertainment games from game design literature. A slightly modified version of the list can be found in Table 3.1. [17]

Table 3.1: Game elements for educational games. [17]

Element	Explanation
Goals	Three types of goals: short, medium and long-term. Player has to do something in order to achieve the goals.
Action	Players are required to take actions or make decisions in order to reach goals.
Feedback	Immediate, appropriate and specific feedback.
Rewards	Rewards are presented using complex system.
Challenges	Complex challenges are faced by gradually learning smaller components.
Mastery	Players are expected to master the smaller components before attempting complex ones.
Decision	When faced with a decision, no option should be obviously correct, while obviously incorrect is acceptable.

Looking at the items in the list it seems that there are several attributes in games which make them attractive. However, there are aspects missing that might be surprising to some people. There is no mention about graphics, music or plot. The same elements are missing from Rouse's list [28]. These are important elements in other kinds media like movies, comics and books. That is exactly why they are not the most important elements in games, because if people want an interesting plot they could read books which have done it for centuries. Games unique aspect is their gameplay, the ability to interact with the game world. Still, elements like graphics and a compelling plot are important when trying to achieve game elements like immersion.

There are multiple benefits of computer games as medium for education compared with traditional ones. Games can be used to 1. teach in a one-to-one manner, 2. adapt to the skill level of each individual, 3. deliver right feedback to players at the right time and 4. motivate players spanning different knowledge and skill levels. [17]

3.4 Serious Game Design Assessment Framework

Mitgutsch and Alvarado developed a Serious Game Design Assessment (SGDA) Framework to analyze serious games. Their motivation for this work was to compensate for the lack of assessment tools. The Framework consists of looking at six separated game elements and then at the holistic relations between them. The

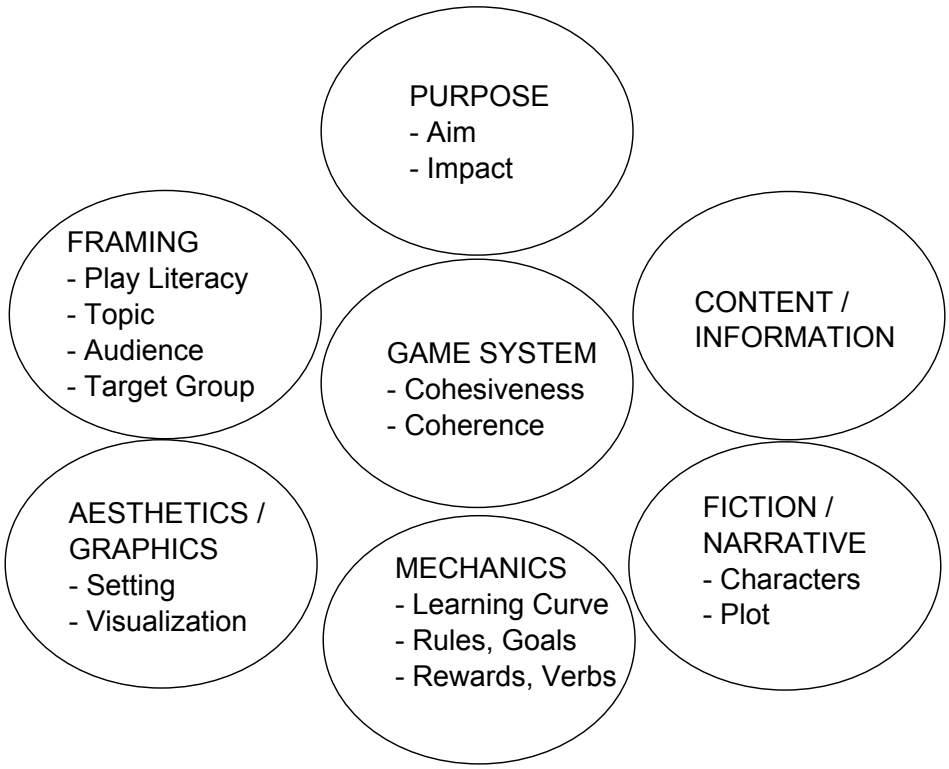


Figure 3.5: Serious Game Design Assessment framework. [23]

elements are listed below and the overview of the framework is visualized in Figure 3.5. [23] *Purpose* in SGDA Framework focuses on games' purpose to impact players. Games always have certain goals and game designers follow their explicit and implicit intentions when designing the game. In normal entertainment games, the purpose is in the game itself and focuses on the gameplay, but educational games are especially designed to have a specific purpose beyond the game. If a serious game does not have an impact on a person's real life, it misses its most important purpose.

In SGDA framework, *content and information* refers to all of the data and words in the game which are visible and approachable to the player. Content could be well presented and correct in accordance to the current knowledge or the exact opposite.

The *fiction and narrative* element of the framework focuses on the fictional space of the game and how it relates to the purpose of the game. Game might not provide an exact story or plot but be more like a fictional sandbox of creativity. In every case, there is something to be analysed.

Mechanics element in the framework is mostly the same as game mechanics in the general game design field. Game mechanics define the rules and methods of interaction in the game world. Most important game mechanics can be translated into *basic verbs*. Verbs are actions that can be performed in the confines of the game. For example in Sweatshop, a tower defence type serious game, the basic verbs are *hiring*, *managing*, and *executing*. In the tower defence game genre, the player's goal

is to stop enemies from reaching the end by building towers along the way. Many games of this genre also require the player to build a maze which has the longest route for the enemies. In Sweatshop, the route is fixed and the player can only add the towers in the optimum places. The basic verbs can be found in the game when the player hires workers as towers, manages different resources and executes levels.

Audiovisual language and the general look and feel is inspected in the *aesthetics and graphics* component. These elements present the game to the players from the start so they play a fundamental role in the game's purpose and its impact on the player.

The *framing* of five key design elements needs to be treated as an additional aspect of the analysis. The focus is in the target group and their play literacy.

Last and pivotal part of the SGDA Framework is the analysis of the *game system*. Cohesiveness and coherence are the focused parts of the analysis. Framework looks at the game system as an integral entity that surrounds the elements explained earlier to shape the gameplay.

4. GAMIFYING PROGRAMMING

The chapter starts in Section 4.1 by looking into how programming is taught today. Section 4.2 continues with the examples of games and gamification focused on teaching programming or skills needed in programming.

4.1 Teaching programming

The current curriculum for elementary school in Finland is from the year 2004. ICT (Information and Communications Technology) skills were mentioned as elements that should be improved within teaching methods. This usually meant that pupils used computers only for writing documents and searching information from the Internet. Teaching more refined ICT skills, usually means the teacher is passionate of the subject and takes extra care to include it to the lessons. New curriculum comes into effect in the fall of 2016 and it will include programming from the first grade forward. Curriculum will include programming appropriate for the age group. Playful experiences should be used as contributing factors for learning. [26, p. 6][25]

4.2 Existing examples

This section presents three different examples of using games as a learning medium focusing on programming. The selected examples are Lightbot, Hakitzu and the Hour of Code. Lightbot is quite popular game which started as a flash game, but has been ported to iOS and Android. Hakitzu is a robot programming game on Android and iOS. One of the reasons for selecting Hakitzu was unusually high amount of graphical quality and 3D elements. Hakitzu also directly teaches to commonly used programming language, JavaScript. The Hour of code was included because it had gone more the gamification style learning than an actual game.

4.2.1 Lightbot

In Lightbot, the player controls a robot whose mission is to light all of the panels in the level. The player controls the robot by giving simple instructions, like move forward, light panel etc., which will then be run one by one. The levels introduce new commands at the start of the level. There is also a maximum of two possible function slots when doing something several times or recursively. Screenshots of the

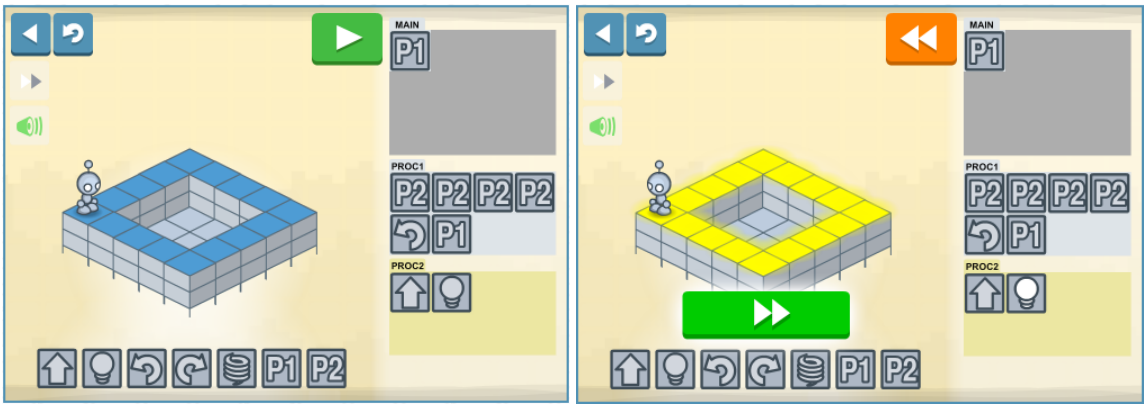


Figure 4.1: Screen capture of Lightbot gameplay. [2]

gameplay are illustrated in the Figure 4.1. The left one is before the code has been run and the one on the right is after running.

The game UI (User Interface) is very easy to use. It is completely mouse driven and the player adds commands by dragging them from the bottom bar to either main or proc slots. Levels usually start with a help screen which can be replayed using the question mark button. All of the other options like reset and sound control is situated in the same view and there is no separate options menu. Keyboard shortcuts would have been a possible addition, but the target audience probably does not need them.

Lightbot is a good example of good learning game for programming. The gameplay is tied directly to the object of learning and it is also interesting. The difficulty level rises by giving more options and requiring more complex codes, but does not necessarily require more code blocks. The rising is also done in small increments so that player has time to get comfortable with earlier tools.

4.2.2 Hakitzu

Hakitzu is a game where the player controls a team of robots using JavaScript and tries to defeat the enemy team. Game contains multiple game modes: single player challenges, Combat and Stealth, and multiplayer mode.

A level starts in a 3D flight to the battle ground. In the setup phase, the player is shown a top-down view of the whole map with robots and obstacles. The player selects a robot to be programmed by tapping on it. Next, the view zooms to the selected robot and shows the code editor. Selecting a line in the editor opens a on-screen keyboard. Depending on the selected difficulty rank, there are buttons which automatically append code or autocompletes by spacebar. Example of the setup view can be seen in Figure 4.2.

When executing the commands, the game shows 3D-animation of the robots fighting. Written code is also shown as floating text and the line being executed is

Table 4.1: Hakitzu coding ranks.

	Beginner	Junior Coder	Coder	Hacker
Buttons for functions	✓	✗	✗	✗
Autocomplete for strings	✓	✓	✗	✗
Autocomplete for functions	✓	✓	✓	✗

shown highlighted. The animation lengths vary between different commands and sometimes the code is obstructed by the animation. Also enemies codes can be glimpsed around them when they are moving, but reading the code is more difficult than with the players own robots. An example of the gameplay can be found in Figure 4.3.

After completing a challenge, the player is given one to three starts and in-game currency based on the competence and difficulty. The player can customise his robots in the chop shop section of the game. Changeable modules are melee, ranged, parts and paints. Modules can be bought with the in-game currency.

Hakitzu has four different coding ranks which are shown in Table 4.1. Each rank has a different level of automatic coding tools. The problem is that the ranks do not really add difficulty as much as they make the coding more tedious. Coding is not about the tools that you use as much as it is about solving the problems. Saying that you are a better programmer if you do not use autocomplete is not really true.

4.2.3 Hour of Code

Code.org® has created a collection of tutorials and activities called the Hour of Code[1] to help people start programming. The parts that are especially interesting concerning this work are targeted to young children. Code.org®'s way of introducing programming starts with introduction videos and then easy coding tasks with block type programming language. Localization is extensive and it allows children around the world to join. The coding tasks involve characters from for example Angry Birds™ in order to make the task more inviting. [6]

After completing the coding session, the user can have a personalized certificate to be send to his email. An example certificate can be found in Figure 4.4. This sort of reward helps to augment the feeling of achievement to the user. The actual certificate is not mentioned earlier on the site, which makes it an unexpected reward.

[1]Hour of Code is a trademark of Code.org

Figure 4.2: Screenshot of Hakitzu setup phase. [15]

Figure 4.3: Screenshot of Hakitzu animation. [15]

Figure 4.4: Certificate for attending the Hour of Code™.

Students are shown the number of the source code lines written, and the actual written code can also be shown. Showing the actual code helps children to move from graphical coding to actually writing code.

Below is a list of game elements, which can be found in the gamified system:

- The characters used in the game are familiar and funny.
- The tasks are written like they are mini plots.
- After each task, the number of lines of code written during the latest task is shown with the total amount of lines of code.
- After finishing the tasks (or pressing "I have finished my Hour of Code" button), the player is given a certificate illustrated in Figure 4.4.
- Leaderboards have both the country and city listed.

5. CASE STUDY

This chapter discusses the case study done alongside with this thesis. Section 5.1 starts with defining the case study and who were involved in it. Section 5.2 continues by looking into the design of the game using different methods described in this work earlier. The actual implementation is described in Section 5.3. Lastly in Section 5.4 the test usage of the game in the real environment is described. Evaluation of the analysis is done in Section 5.5. The results of the user testing are written out in Section 5.6.

5.1 Design Process

The purpose of this project was to use modern web technologies in an educational purpose. Teaching programming to children was chosen, because it is a modern topic and many IT-companies are trying to rise awareness of the importance of programming. For example, different companies keep kids' code school[1].

The game was created inside Nokia[2] with multiple participants. There were people from the Student Innovation Lab (SIL) and Nokia's Hermia office in Tampere, Finland. At SIL, most of the employees are high school student trainees. SIL participated with five coders during the spring, two during the summer and one graphics artist during the whole development. During the semester, they worked approximately 12 hours per week and full-time during the summer. From Hermia there were two persons working mainly in this project. One of the Hermia workers was the primary project manager. The author acted as the project manager for SIL trainees in addition to participating in the design and development of the game.

5.1.1 Idea

The point was to create a game for young children to learn programming related skills. During brainstorming sessions, different styles and environments were proposed. The age of children makes designing harder, because the mindsets of designers and children differ. Furthermore, the game should be equally attractive for both boys and girls. The project manager came up with the idea of bookworms defending

[1]Website: http://koodikoulu.fi/
[2]Website: http://company.nokia.com/

their home tree from termites. The game genre will be tower defence, because it is casual and easy to get into.

Tower defence has been used also previously in educational games, for example a game for diabetes patients called Power Defence [4]. Also reporter Battjes defines a flash game Sweatshop being mechanically a normal tower defence game [5]. Sweatshop is a serious game of manufacturing clothes at a factory which uses human exploitation [18]. Doucet and Srinivasan also used some elements from tower defence games when they created real-time strategy game, *Super Energy Apocalypse*. Their topic was energy economy and sustainable energy. Their game consisted of several different economy buildings in addition to towers (flood lights, turrets and Tesla coil) used to combat zombies which were the enemies in the game. They did not use the mazing aspect of tower defences, and the zombies tried to destroy everything and not just get to the goal. Still, the placement of towers and supporting economy are important aspects of the game which can also be found in the tower defence genre. [10]

5.1.2 Target group

The game is targeted to elementary school pupils from the ages of 7 to 12. Because children develop mentally especially during this age period, the design focused on the first graders and the target group was expanded by adding difficulty. With age also the interesting themes change and the bookworm aesthetic can be less interesting at the higher end of the age spectrum.

Special concerns were about children's literacy skills, and to address this problem mostly symbols were used. Text was only used in specific places where symbols would have been too complex or taken too much space. For a multitude of reasons such as children's age and educational context, the aesthetics of the game had to be thought carefully. One constriction to the design was the minimization of all visual connections to violence.

5.1.3 Game overview

The game was designed to have three different phases. The main phase is a tower defence where the player tries to understand the path the termites will take and place towers to stop them. The resource collection phase switches the player to a creation mode where he will create script for bookworms to execute. The player tries to create the shortest route from the home tree to the resource node and back in order to get the maximum amount of resources. Resources collected in this phase can be used to upgrade towers. When the player tries to upgrade something, the player switches to the third game phase. In the upgrade phase, the player will be

given several objects with slight differences. For example, three balloons from which two is red and one is blue. Players' job is to group objects based on what unites them. In this example, two red balloons would be their own group. Each object is also a balloon so they are part of the same group. This challenge is meant to get the player to think about the similarities of objects. Grouping skill talent would be useful when learning object-oriented programming. Later analysis will also touch this subject.

The tower defence phase is the most complex phase. When the player starts a level, he will see the level layout which consists of squares and the script which the termites will execute. Before placing towers, the player should understand the way termites are going to move. The game has a free-form drawing tool to help the player visualize termite's path. After drawing the path, the player can place towers. Different towers work well against different termites and can have different properties like range and damage. In the first levels, the player will come across only a couple of types of termites and can use only the basic tower. Players should get familiar with the script concept first and tower defence aspects become more complex later.

The game has different worlds, each of which have a different set of levels. Different worlds can specialize in training different programming aspects such as loops. Implemented game has tutorial, normal and randomized worlds. The tutorial world has very easy levels to get familiar with the game. The normal world has a set of incrementally harder levels which use most of the features of the game. The randomized world has a couple of different presets for the randomization engine. Players have all the towers unlocked in these levels.

Progression inside levels is done by using a star crediting system. The first level is always unlocked, but the player has to get at least one star in the level in order to progress to the next level. The maximum number of stars is three and three stars usually require almost perfect execution.

5.2 Analysis of the design

5.2.1 Serious Game Design Assessment Framework

In this subsection, the game is analyzed using Mitgutsch's and Alvarado's Serious Game Design Assessment Framework [23].

Purpose

The purpose of the game is to develop young children's logical thought abilities and inspire them to take on programming on their own. Understanding block type coding and the ability to modify it are more concrete goals at the game.

Content / Information

The game provides in-game data of available resources, completed upgrades and progression through the levels. Players can also get achievements which can be seen in the treasury room.

Fiction / Narrative

Plot: "Players have come to help defend the Great Tree of Knowledge where the kingdom of bookworms is situated. Council of Elders gives the players a group of elite bookworms to command. In order to efficiently place the fortifications players are also given a magical scroll of Code which will divine the plan of the termites. It is the player who has to decipher the code on the scroll and save the bookworms."

Mechanics

The gameplay is divided into three phases. The main phase is the tower defence which uses mechanics similar to other tower defence games. Second phase is about collecting resources. In order to get bookworms collect resources efficiently, the player has to improve the script which bookworms follow. This script is made with similar blocks as in the tower defence phase, but in this phase, the player can also change it. The third phase is the research game which determines how much researching an upgrade will cost resources. The player will be shown different objects from which he should find common properties. The basic verbs in the game are *understanding*, *improving* and *observing*.

Aesthetics / Graphics

The game uses 2D cartoon-styled graphics. Because of the target group being young children, the attacks from towers etc. are also cartoony and try to be as non-violent as possible.

Framing

The game focuses on one to sixth grade elementary pupils and especially on the youngest ones at grade one. This is one of the reasons why the game mechanics will be very easy to learn and the difficulty comes from understanding more complex

scripts. Players should find replaying a level interesting, but playing exactly the same level might encourage players to skip the script and just see what happens. For this reason, a level generator was developed. The generator can create similar levels, but still different enough that player has to read the script.

Game System

The purpose of the game is to improve logical and analytical thought processes. Additionally, basic programming elements are introduced. Game mechanics works towards the purpose by utilising analytical challenges. The main activities *understanding*, *improving*, and *observing* have analytical aspects. Linking between mechanics and fiction is the least coherent part in the game. The battle plan mechanic could be linked to the swarm intelligence of termites. Also resource collecting can be linked to the swarm intelligence. The research minigame cannot be linked to the fiction, but it is coherent to other elements in the SGDA framework.

Framing, aesthetics, and fiction work well together in author's opinion. Children like cartoon-styled characters and speaking bookworms are acceptable to child's mind.

5.2.2 Rouse's list of game elements

In the following, there is a comparison of the design with Rouse's lists of game elements. The basis for this analysis is that these elements should be in a good game or there should be a good reason of not having them. First there are six reasons people have for playing games and then 11 things that players expect of a game. Summary of findings can be found in Table 5.1

Reason 1: To get a challenge

Tutorial levels and early normal levels can be beaten with badly placed towers. However, other levels will give fair amount of challenge, especially if player want full three stars.

Reason 2: To socialize

There are not really any socialization features in the game. For example, chat functionality, a friend system, and leaderboards could have implemented, but were not because of limited resources. These social elements would not have necessarily directly improved the main focus which is learning programming. For this reason, the resources were focused on other aspects of the game. The game is not meant to be played for long periods of time and it is more about self-improvement than trying to compete with friends.

Table 5.1: Summary of comparison to Rouse's lists.

Reason	Included	Partly	Not found
To get a challenge	✓		
To socialize			✓
To have a dynamic solitaire experience	✓		
To get bragging rights	✓		
To have an emotional experience		✓	
To fantasize	✓		
Expectation			
To have a consistent world	✓		
To understand the game world's bounds	✓		
For reasonable solutions to work	✓		
To have a direction	✓		
To accomplish a task incrementally	✓		
To be immersed		✓	
To fail		✓	
To have a fair change	✓		
To not need to repeat themselves		✓	
To not get hopelessly stuck	✓		
To do, not to watch	✓		

Reason 3: To have a dynamic solitaire experience

Players always make their own choices which affect how what path termites use and how far they get. This is also completely single player game, so the dynamic solitaire experience is achieved.

Reason 4: To get bragging rights

Players can use beating of difficult levels as bragging right. When, playtested with children they constantly asked from each other, how far the others had got in the game.

Reason 5: To have an emotional experience

The game will not probably rise very strong emotions. Children might sympathise with the bookworms whose home is attacked by termites.

Reason 6: To fantasize

The world is quite different from ordinary lives. Still, talking animals are a common theme in children books so the world is probably somewhat familiar to the players.

Expectation 1: To have a consistent world

The only place where things happen differently is when either termites or towers change. However, each of different termites and towers have their own unique look, so there should not be any confusion.

Expectation 2: To understand the game world's bounds

The game is not very complex so the bounds should be easy to understand. The player can draw anything he wants, but it does not affect the game. The towers can be only placed on squares and there is limited amount of them. After releasing the termites, the player cannot do anything expect restart the level or speed up the animation.

Expectation 3: For reasonable solutions to work

The most successful tactic in the game is to make the termites move to the longest route. Usually, this means that at least one of the towers will be on termites' path. Different towers work well against different termites, but that does not usually affect the correct placement tactic.

Expectation 4: To have a direction

The direction in the game is achieved by unlocking new levels to play and in the end trying to master the random generated ones.

Expectation 5: To accomplish a task incrementally

The levels get incrementally harder and in the end, the player should be able to beat any level which uses familiar code blocks. Players can test their abilities in the random generated ones.

Expectation 6: To be immersed

The game can be played in a full screen mode, which would remove the possible distractions like browser tool bars or other windows. There is no change in aesthetics during the game which could break the immersion. The game is a good representation of the immersion that a 2D tower defence game can have. Games with faster pace, 3D world or more engaging narrative could have better immersion if implemented correctly.

Expectation 7: To fail

This is probably not one of the important features in a children's game. There are failure states in the game, but the tutorial levels are usually beaten if the player just places one tower somewhere on the map. On normal levels, the difficulty is higher, but it is quite easy to beat a level with one star.

Expectation 8: To have a fair change

The game is very fair. It is mostly a puzzle type game and all the information is given to the player beforehand. A try and error approach should be needed only if the player does not want to read and process all the information.

Expectation 9: To not need to repeat themselves

This expectation is not completely granted in the game. One of the goals was to create a game where players would see the code becoming incrementally more complex. Complexity comes from the addition of code blocks and variation in termites and towers. This will inherently add some level of repetition, but still no two levels are exactly the same.

Expectation 10: To not get hopelessly stuck

The only way a player cannot get forward is if he cannot get a single star on a level. All of the levels created were tested on people who did not create them and the testers did not find a level where one star would be almost impossible to get. Getting three stars is a very difficult challenge in some levels.

Expectation 11: To do, not to watch

There are only two places where the player is not able to actually do anything else but watch or read. First of these is the tutorial window in tutorial levels. Still, the player can easily skip them, if he knows how to play already. The other place is the animation of termites moving. The actual outcome is calculated when the player

presses the play button. For this reason, the player cannot do anything but speed up the animation.

5.2.3 List of important education game elements

This section tries to find the elements, listed by Linehan et al. for a good educational game, in the game. The section continues with discussion about how well the game uses the benefits of computer games as a medium for education. The discussion is done by comparing the game with the benefits found in literacy. The summary of the analysis can be found in the Table 5.2. [17]

Table 5.2: Summary of comparison to Linehan et al. list

Element	Included	Partly	Not found
Goals		✓	
Action	✓		
Feedback	✓		
Rewards	✓		
Challenges	✓		
Mastery	✓		
Decision	✓		
Benefit			
Teach in one-to-one manner			✓
Adapt to the skill level of each individual			✓
Deliver right feedback to players at the right time		✓	
Motivate players spanning different knowledge and skill levels	✓		

Element 1: Three types of goals: short, medium and long-term

Short term goals are different in each phase of the game. In the tower defence phase, the goal is to destroy the termites before they reach the tree. In the resource gathering phase, the player tries to collect as many resources as possible. In the research phase, the player tries to find categories for the object as well as possible. Of medium goals, there is only one which is to get through a world. There is really no long-term goal for the player. A long term educational goal is to get better

understanding about scripting and improve analytical skills. However, this goal probably does not convey to the players.

Element 2: Player has to do something in order to achieve the goals

The goals are quite easy to achieve at least at a minimum level to keep a low threshold for people to start playing. Still, the player has to do something to get through. In the tower defence phase, the player has to put towers on the map or he will lose. Reading the script is not mandatory, because the player can luckily put the towers in right places. In the resource gathering phase, the player has some rudimentary script which will need to be updated and in the research phase, the player gets nothing automatically.

Element 3: Immediate, appropriate and specific feedback

The player will get immediate feedback when trying to put a tower on top of another tower or something else completely against the rules. The player will not get any feedback, if the tower is in the right place in order to beat the level. This kind of feedback is only given after pressing ready, and replaying the level will give a slightly different challenge.

Element 4: Rewards are presented using complex system

Rewards are given on two different levels. Smaller rewards are about getting one to three stars per level depending on how well it was completed. Players are also given achievements for completing different challenges. An example challenge could be to beat all of the levels in a particular world with three stars.

Element 5: Complex challenges are faced by gradually learning smaller components

Challenges depend on the levels created and of the worlds in which they are. In the initial worlds created by the development team, there is gradual complexity by adding more complex code blocks and structures. For example, first levels have only individual blocks and later levels have also while loops.

Element 6: Players are expected to master the smaller components before attempting complex ones

First time players should start with the tutorial world. There they are presented new code blocks one by one and have a couple of levels to test each. After completing the tutorial, they can move to the normal world where they are given increasingly

difficult levels, but they have to beat them in order. After achieving some level of mastery, they can move to randomly generated levels. In randomized levels all types of termites can appear and the player has access to all of the towers. These levels will require both skills in reading the script and placing towers strategically.

Element 7: When faced with a decision, no option should be obviously correct, while obviously incorrect is acceptable

For the placement of towers, the understanding of the script is necessary. Even placing the tower next to the tree is not always the best place, because the range of the tower is quite low. Usually incorrect places are, for example, the corners of the map where the range is very limited, but they could be right places for specific levels.

Benefit 1: Teach in one-to-one manner

There is not really any one-to-one teaching options. The player can choose which world they want to play, but tutorial and normal worlds levels have to be beaten in order. A teacher could unlock the levels before hand and then tell the player which levels to play and in which order.

Benefit 2: Adapt to the skill level of each individual

There is no automatic adaptation designed into the game. Players can adapt by themselves by not trying to get all three stars and focusing only getting one star.

Benefit 3: Deliver right feedback to players at the right time

Delivering feedback is one very important educational benefit of games, because even in classroom teachers cannot be always present and give feedback. One educationally beneficial feedback in this game is how well the player has understood the termites' path by reading the script. The feedback is delivered to the player using the drawing feature. The player first draws the path he thinks termites will follow and after releasing the termites can evaluate how well the drawn path matched the actual path. An improvement could be to automatically compare the paths for the player, but in order to use less development resources this was skipped.

Benefit 4: Motivate players spanning different knowledge and skill levels

The game tries to use this aspect by having incrementally rising difficulty in levels. Because the target group is the grades one to six, this aspect was very important.

5.3 Implementation

5.3.1 Used technologies and development process

One of the purposes of this project was to use modern HTML5 technologies. For this reason the main coding language was JavaScript. Actual HTML part in the project was minimum and contained mainly the canvas which was manipulated with JavaScript. During development mostly open source tools and libraries were used.

Unit testing was done using QUnit[3], because it was easy to set up and light enough for the purposes of the project. Git[4] was selected as the version control system, mainly for its familiarity. During the project there were some network problems and Git being distributed system helped when distributing code between separate teams. The game was designed to work with different resolutions and SVG(Scalable Vector Graphics) graphics were used for fixing possible scaling problems. For the same reason, PaperJS[5] was selected as the graphics library.

Development was carried out using agile principles. Project team had the sprints of five weeks and the backlog in a project management website. The sprints were long because the students worked only three and a half days in two weeks. One sprint consisted of around nine work days.

Students at SIL were from two different classes and worked on different days. For this reason there was three separate coding teams. SIL people were split to two groups by classes and people in Hermia were in their own group. The older group was allocated the random generation part and the younger group worked on the visualization of the script and free-form drawing. Hermia team worked on many other parts of the game particularly elements connected to the graphics library. The author worked mostly on the more generic parts of the game such as storage and base mechanics such as winning conditions.

At the end of the spring, the resource allocation changed. One of the team members in Hermia was mostly allocated to another projects and team at SIL was also scaled down to one team with two members. During the summer trainees at SIL worked full-time which helped to compensate. The author started to work on other parts of the game to get it ready for testing.

[3]JavaScript unit testing framework. Website: http://qunitjs.com/
[4]Distributed version control. Website: http://git-scm.com/
[5]Vector graphics scripting framework. Website: http://paperjs.org/

5.3.2 Implemented part

The actual development of the first part of the game, tower defence part, went approximately as it was designed. However, because of changes in resource allocation the other phases of the game, which were research and resource collection phases, were not implemented. For this reason, any kind of upgrading was removed from the tower defence game. The towers had fixed attributes which helped when designing levels. At the beginning, the plan was to create only random generated levels, but tutorial levels and a set of levels designed to be harder were added.

One of the principles the designers had during creation the game was that it should have as little of text as possible. The ability to read should not be necessary when playing the game. This can be seen in the main screen illustrated in Figure 5.1. Buttons for play, setting and about have been illustrated as icons. Here you can also see the cartoonish and playful aesthetics the game tries to achieve. After clicking the play button, the player is moved to world selection screen shown in Figure 5.2. The cabinet is the same as the one in the middle of start screen. Every world has a unique icon, but also title to tell what kind of world it is. The one on the left outside the cabinet is the tutorial world and the upper one in the cabinet is a normal world consisting of 15 levels with increasing difficulty. The last world is collection of four levels of different kind of random generation. Selecting one of the worlds will open the cabinet and give a zoomed in view of levels as shown in Figure 5.3.

Figure 5.1: Start screen.

In the tutorial world, every level begins with a small piece of the story and introduction of new gameplay mechanics in that level. The start of first tutorial level can be found in Figure 5.4. This is a major situation where the idea of minimum

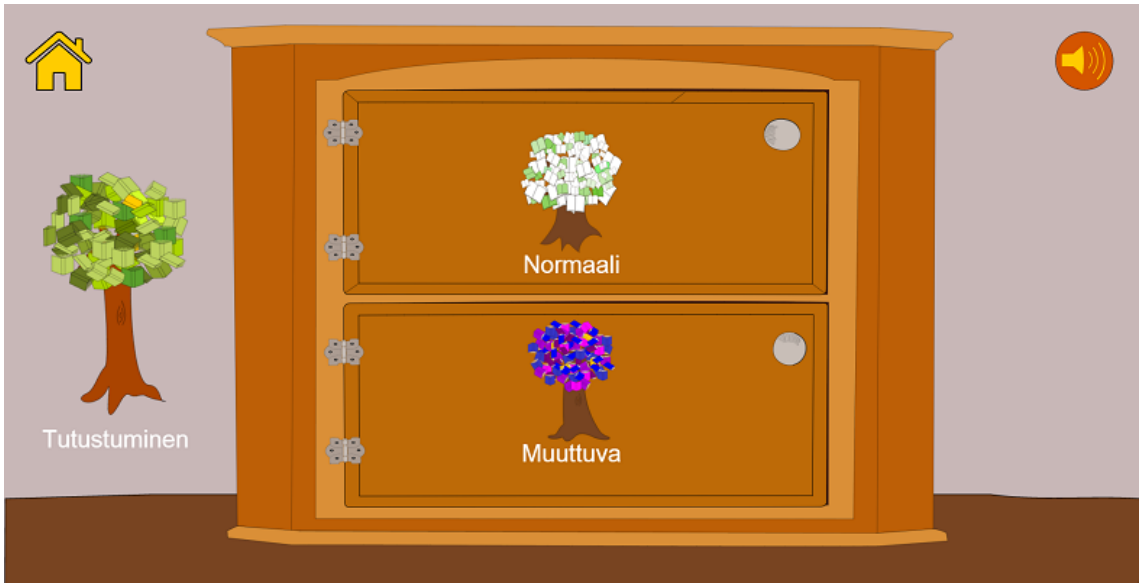


Figure 5.2: World selection screen.

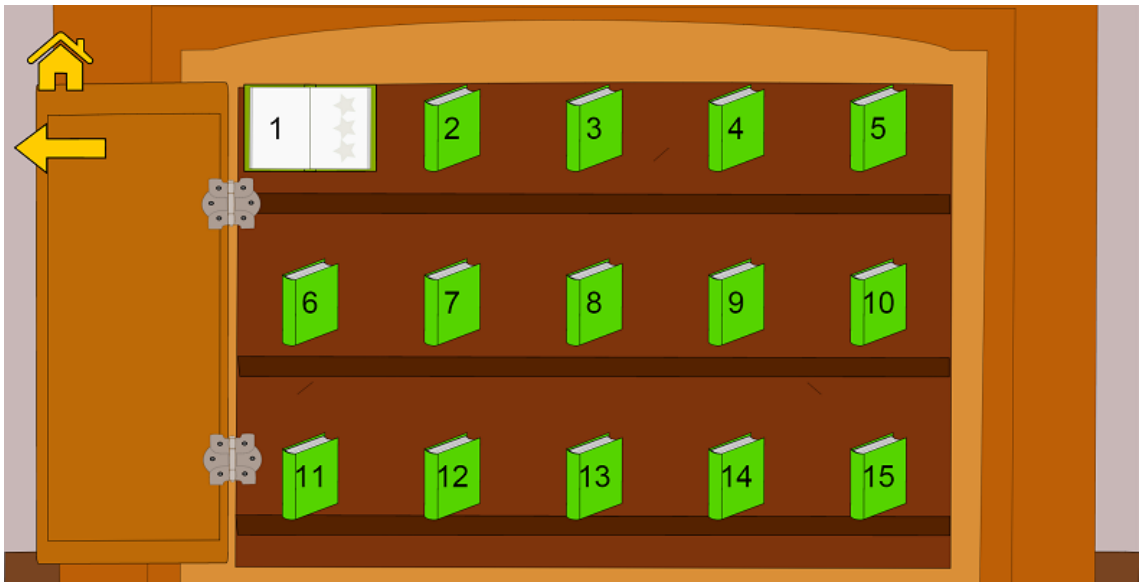


Figure 5.3: Level selection screen.

text did not work. A spoken version of the text was an option that the team did not have the resources to create. Another option, telling the story through pictures would have taken too long to create and would have decreased the performance of the game.

In the tutorial world after the help window and in the other worlds right away, the player is shown a level similar to the one in Figure 5.5. Buttons for the generic control of the game are on the left side of the screen. From the top, buttons are: home, to level selection, release termites, draw on/off, clear, zoom in and zoom out. The player cannot make any changes to tower placement after releasing the termites. Buttons for different panels are on the upper right corner. First button

Figure 5.4: Start of first tutorial level.

brings back the help window, which is visible on tutorial levels. Next one opens info sheet about towers and termites which can be seen in Figure 5.6. The upper right button in Figure 5.5 swaps between script and tower listing. In the figure, the script is selected and it can be seen at the right side of the level. In scripts' upper left corner is the type and quantity of termites. The first block of the actual script tells where the termites are going to dig up and where they are going. In this level, termites are digging up at the location of A2 and moving to the right. The next two blocks represent moving forward. When the player is inspecting the script which termites will try to follow, he can draw the path on top of the level using red colour.

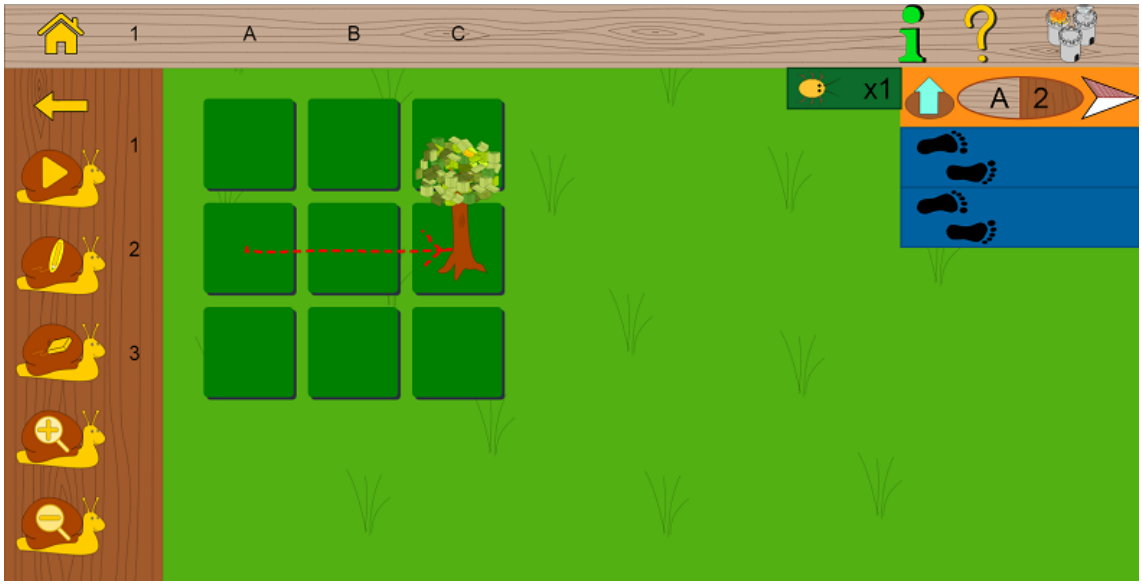


Figure 5.5: Drawing termites path before placing towers.

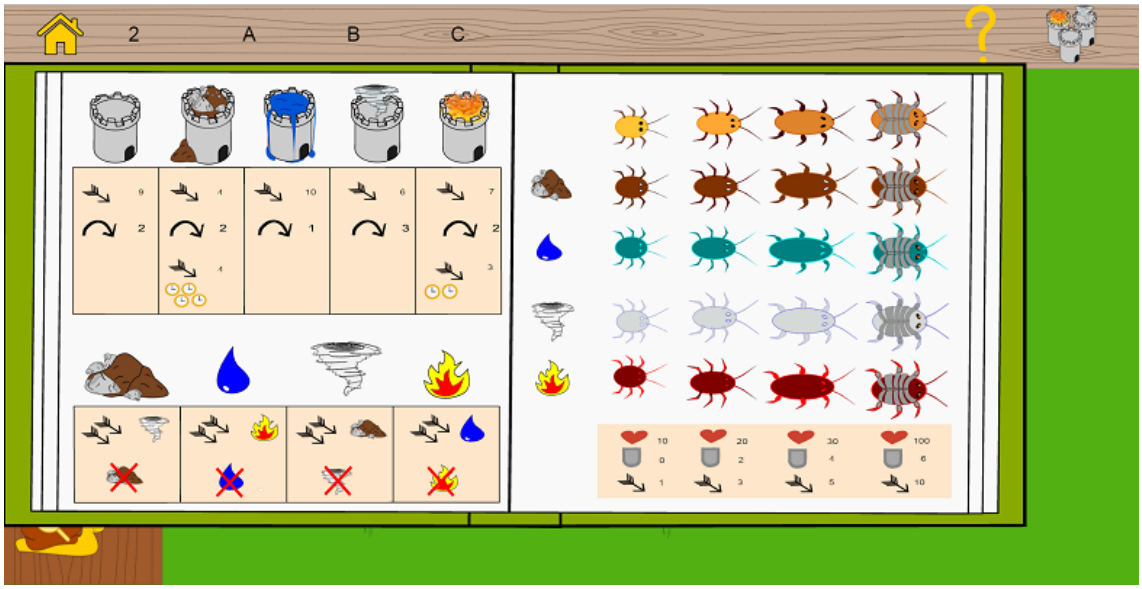


Figure 5.6: Info sheet about towers and termites.

After the player has drawn the path, he can place the towers. In Figure 5.7, the player has placed a tower to the bottom right corner. At this level, the player had only one tower to be placed and the type of the tower was also limited to one. On later levels, players will have multiple towers to be placed and there will be five different types of towers. After termites have been released, the buttons on the left have changed. The play button is still visible, because the image was taken when the game was paused, but normally it would be a generic pause icon. The next button is for restarting the level, because the player cannot change the outcome after releasing the termites. He can reset the level quickly if he sees an obvious error in his tower placement. The last one speeds up the animation.

After all the termites have reached the tree or died, the player is shown an ending screen as shown in the Figure 5.8. The player is told if he beat the level or not. Also the previous result in this level is shown. The level can be beating by one, two or three stars, and the number is depending on the amount of termites getting to the tree. At the bottom of the screen are buttons for restarting the level, going back to the level selection and moving on to the next level. The next level button is not shown if the player has not beaten the level.

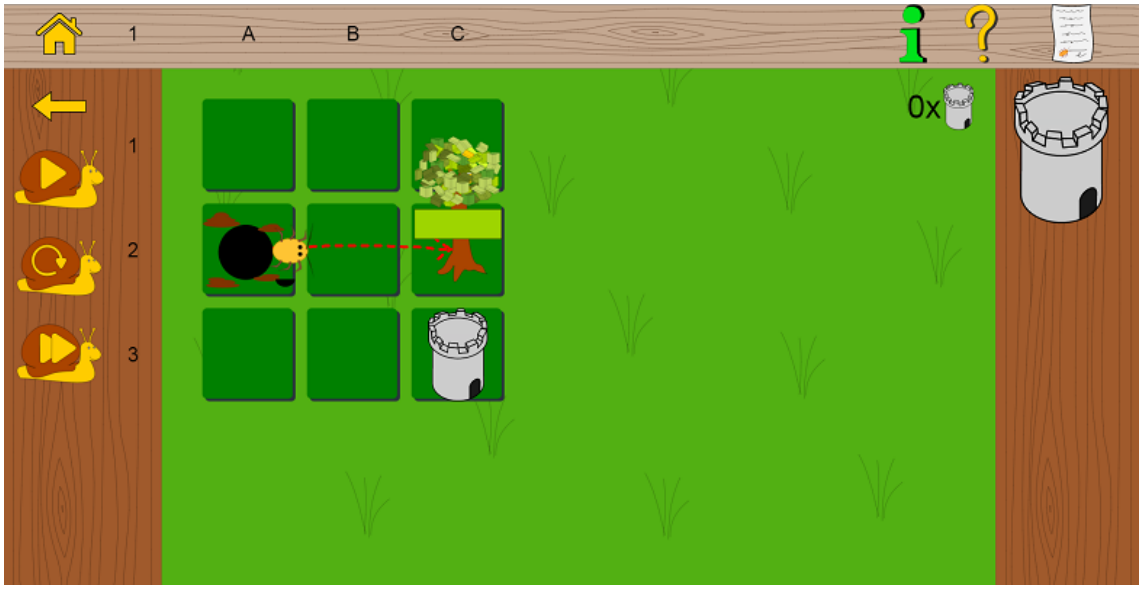


Figure 5.7: Towers have been placed and termites have started attacking.

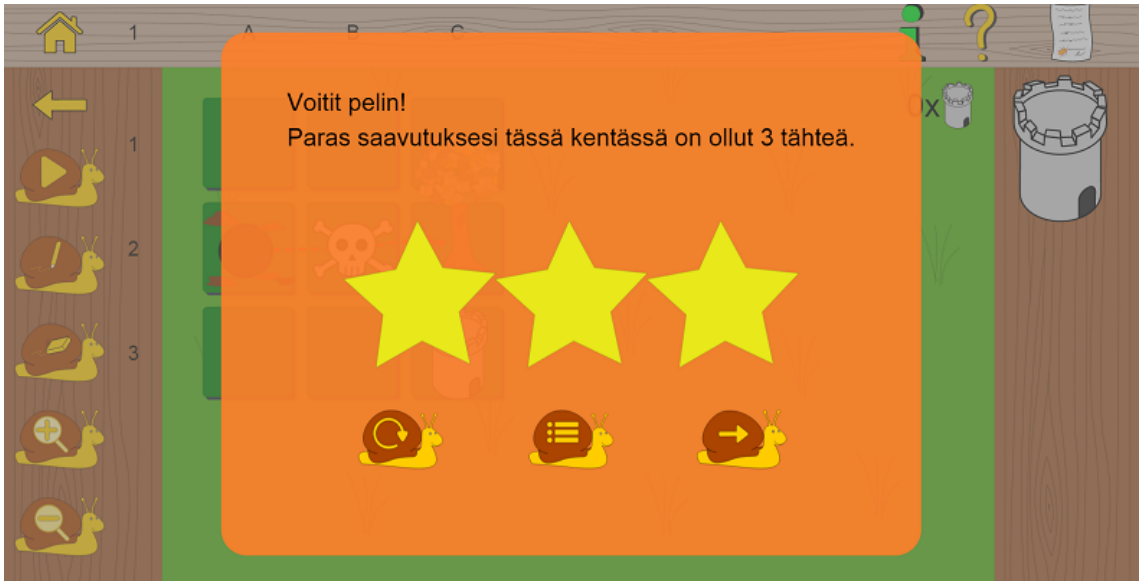


Figure 5.8: Victory screen.

5.4 Conducting the user testing

5.4.1 Purpose of the user testing

The main reason for the user testing was to get generic feedback about Bookworms game. Another reason was the evaluation the game against another game with the same purpose of teaching programming.

The testing session was also a great opportunity to ask more generic questions from pupils. Questions were created to check the attitude towards coding and to test how well the children understand the basic programming aspects like executing

commands literally. The attitude is important because "Coding is only for boys" attitude drives girls away from computer science fields. If they get the chance to try activities similar to programming at early age, they might be less inclined to develop a negative attitude towards programming.

The testing sessions also had another education game to evaluate the goodness of Bookworms. The chosen comparison game was LightBot.

5.4.2 Test group

The actual user testing was conducted in an elementary school in Akaa, Finland. Test group consisted of 17 fourth graders which generally means 10 years old in the Finnish school system. Nine of the pupils were girls and eight boys.

5.4.3 Test period

The user testing was conducted in September 2014 and the session duration was 75 minutes.

5.4.4 Questionnaire

The questionnaire was created to be very simple so that it could be answered by elementary school pupils in about five to ten minutes. An earlier idea was to do multiple sessions with the same group and compare the first and last sessions to get data about how the game affected the pupils. With limited resources, only one testing session could be used. Still, getting even a faint feeling if the games improved children skills or motivation was interesting. For this reason, there were two questionnaires in one session. One in the very beginning of the session and one after playing the games. Both questionnaires contained statements with the Likert scale[6] and open questions. The last questionnaire asked additionally about games and generic thoughts about playing educational games. The questionnaires can be found in Appendix A, but they are in Finnish.

5.4.5 Test session

The test session composed of several phases. The first one was the general introduction of people and the reason for testing. Then the children got a questionnaire to test their starting point and to be able monitor change after playing the game. Before each game, there was a short introduction to the game.

The first game was Bookworms game, the one discussed in this thesis. There were some network connection problems which took some time of the actual playing.

[6]Scale where respondent express their agreement or disagreement to several statements.

After around 20 minutes of playing, the game was switched to LightBot in order to compare Bookworms game with other coding games. LightBot was also played for around 20 minutes.

The second questionnaire was given with enough time to calmly fill it. The extra time was used in general discussion and thanking the testers.

5.5 Evaluation of design

This evaluation focuses on the design of the game. Because of the reallocation of resources during the development, there were not enough resources to implement all of the phases. For this reason small modifications had to be made to the tower defence phase like removing upgrades from towers. These modifications did not affect critical parts of the game and that is why the evaluation of design can also be used as the evaluation of the implementation for the parts which were implemented.

In the previous chapters, different game assessment frameworks and features that can be found usually in games were introduced. In Section 5.2 was the analysis of the game using those metrics. Game fared quite well when analysed using Serious Game Design Assessment Framework. Most of the Rouse's game elements could be found in the game. For most of the missing ones, the reason for not including them was the age group or small size of the game. Educational game elements found by Linehan et al. were slightly less well included in the design. For example, teaching in one-to-one manner, adaptation to the skill level of each individual and, to some extend, the complexity of rewards, were not included in the design.

5.6 Results

All of the Likert scale answers can be found in Tables 5.3 and 5.4. In the first questionnaire almost all pupils, 94%, did not know whenever or not learning programming is easy. After testing the games answers moved away from not knowing and only two pupils answered not knowing again. Most moved to the positive side, meaning that 71% answered agree or strongly agree.

Before playing the games, pupils disagreed with the statement, "Coding is only for boys", by a percentage of 88. After the game, the percentage had risen to 94 percent. There was not much difference between genders, girls disagreed slightly more.

After the Likert scale questions, there were two logical tests for following the orders exactly. The first task was to draw the turtle in a new position in a 3x3 map after executing the orders. The commands where either turn left, turn right or move forward. The second task asked whether or not the termite got to the tree by executing the orders. It was only a yes or no question and for this reason any

Table 5.3: Results before playing the games. First is the sum and then in brackets first boys and then girls. (n=17)

Statements	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
Learning programming is easy	0 (0/0)	0 (0/0)	16 (8/8)	0 (0/0)	1 (0/1)
Programming is only for boys	8 (4/4)	7 (3/4)	1 (0/1)	1 (1/0)	0 (0/0)
School could have more activities concerning programming	0 (0/0)	0 (0/0)	4 (1/3)	8 (4/4)	5 (3/2)
Games would motivate to learn more	0 (0/0)	0 (0/0)	5 (3/2)	2 (2/0)	10 (3/7)

Table 5.4: Results after playing the games. First is the sum and then in brackets first boys and then girls. (n=17)

Statements	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
Learning programming is easy	0 (0/0)	3 (2/1)	2 (1/1)	3 (3/0)	9 (2/7)
Programming is only for boys	11 (4/7)	5 (3/2)	0 (0/0)	0 (0/0)	1 (1/0)
School could have more activities concerning programming	1 (0/1)	0 (0/0)	0 (0/0)	6 (3/3)	10 (5/5)
Games would motivate to learn more	0 (0/0)	0 (0/0)	3 (2/1)	4 (2/2)	10 (4/6)

drawing that children made did not impact the results. The results can be found in Table 5.5. However, the drawings showed that the children got to the right answer with wrong logical thinking. Even if they did not draw erroneously, it was still a 50 % chance to get the right answer by guessing. Hence this question gave only inconclusive data.

At the end of the second questionnaire, the children were asked which was more interesting game and why. Bookworms got 76 percent of the votes as shown in Figure 5.9. Reasons for choosing bookworms was very disperse. One chose it,

Table 5.5: Ratio of right answers to the logical thinking tasks.

Task	Before	After
Draw the turtle	6 %	24 %
Did the termite get to the tree	82 %	76 %

because it was the easier one. Another child chose Bookworms, because it was harder than LightBot. Other generic adjectives like fun and nice were also used.

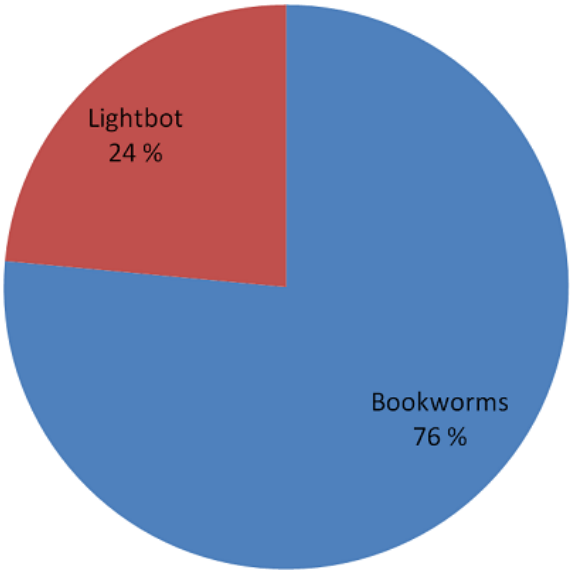


Figure 5.9: LightBot vs Bookworms.

6. DISCUSSION

In this chapter, is discussion about the implications of the evaluation of the game design and the results from classroom testing.

6.1 Creating a compelling educational game for programming

In Chapter 5, the design of the game was analysed using different analytical frameworks presented in background study part of the work. The actual evaluation based on the frameworks was presented in Section 5.5. Based on the evaluation, the game fared well. Normal game elements were found and they were used in a compelling way. Some of the educational aspects were missing, which might lessen its effectiveness in a classroom. Still, it might have more compelling aspects in getting children to play it on their own time.

In Section 5.6, the findings got from normal fourth grade classroom were listed. During the testing, children tried two games: the one created as part of this thesis and LightBot. LightBot was selected for being featured in the Hour Of Code [2]. In the questionnaire, Bookworms game got 76% of the votes. Of course this was only one test and the designers might have presented their game in a better light or given more help for it. Nevertheless, it still shows that somewhat compelling game for children was created.

There was one important aspect of the game which was not tested well enough. The first goal was to create a game for 7- to 12-years-old children. Because of the large age scale, the focus was on the first graders. Still, the testing was done on fourth graders because existing contact and limited resources. Some children who tested the game thought it to be difficult when to others it was somewhat easy. Although the general goal of 7- to 12-years-old was achieved, but there is no data about how well the game would work on the first graders.

6.2 Learning programming through games

Almost all games try to get players play more of them. For this objective in mind they use compelling stories, create wonderful worlds and create intuitive and interactive gameplay. All of this is done to get players more motivated and linked to the activity. Using these same techniques it should be possible to create educational games. In Chapter 2, these techniques and elements used in games were connected and an educational game, Bookworm, was created based on them.

Two educational games, Bookworm and LightBot were tested in a classroom environment in a time frame of 75 minutes. The results in Section 5.6 were promising at least when thinking about children's preconception about programming. In the beginning, children did not really know if learning programming is easy or not, but after playing the games they were thinking that it is easy. Of course programming has its own difficult parts and it requires hard work from time to time like every discipline. However, it is easier to be motivated during these difficult parts if you know that you can do it. Also after playing the games, almost all the children were opposing the idea that programming would be just for boys.

The questionnaires had tasks which required some understanding about following the commands literally. One was drawing a turtle to its new location after executing the commands and the second one was yes or no question about the termite reaching the tree. There was some improvement in the first task after playing the games, which was expected, because both games included aspects of executing commands. Still, the amount of correct answers was less than quarter of the subjects. Correct answers dropped in the second task, but not in any significant way. It had a very high success percentage, but scientific value was very small. Because it was a yes or no question, any additional drawing was discarded when transferring data to the digital format. Nevertheless, when analysing the drawings, it could be seen that the children had stumbled upon the right answer by mistake. And the answers had to be examined very sceptically, because of the yes or no nature of it. There was a 50% chance to get the right answer by guessing and then over 75% success rate does not look so great.

The user testing seems to have improved the motivation of learning programming. The time frame was so small that it clouds the true results. Some children could have thought that after playing fun games they should answer that programming is easy. The data was inconclusive about the logical thinking. This was partly because of the small sample, but even more affected the quality of the tasks. The task where the pupil had to draw the turtle after executing the orders was very difficult and it should have been split into smaller tasks. The second task about logical thinking asked if the termite reaches the tree or not. The yes or no nature of the question already significantly lessened the value of the task and it had the same problem as the turtle task.

6.3 Gamification in learning programming

In this thesis, the focus was more on traditional computer games, but gamification was also defined. Gamification uses mostly the same theory as normal games expect when gameplay is concerned, which is not present in gamification.

Compared to games, gamification could be included more easily in normal school routine. One of the ideas taken from games could be the down to top perspective. Today grades are usually marked by starting from the highest grade and any mistakes during the course will drop the grade. More motivating way would be to start from the lowest grade and the grade will rise during the course.

In New York, a public school called Quest to Learn[1] has taken the idea of gamification of school to the next level. Their idea was to create a school which in itself is a game. The school opened in the fall of 2009 after two years of planning from a team of educators and professional game developers. Not every school needs to be based completely on games, but there could be some very interesting ideas, which could be applied in normal schools. [20, p. 127-132]

6.4 Further studies

An extended length research could be interesting. One possibility could be to do the testing for a full week where the first test would be on the start of the week and the end questionnaire at the beginning of the next week. Children would play the game at first in school, but mostly voluntarily at home. With this kind of research data could be gathered about how many children would play this educational game at home where they probably have a multitude of games and activities also available. Several of the test group mentioned that they were going to play the game at home, but there is no information about how many really did. Of course additional research subjects are also required for good quantitative research, but qualitative research could be done with just more time.

Because children's development is fast during the age period of 7- to 12-years-old, it would be good to test the game with different grades. For resourcing reasons, the testing in this thesis was only on fourth graders even though the target was first graders.

The stereotype of coding being only for boys could be researched more because it was very clearly not the stereotype in this test group. However, it was only a single class and small amount of programming education in the form of the Hour of Code could already have changed the attitude of children.

Analysis of the game was done by the author who was also designing and implementing the game. A small interesting study would be for somebody else to use the same lists of game elements and analysis frameworks to analyse the game. This would give more information about the game and how subjective the used tools are.

[1] Website: `http://q2l.org/`

7. CONCLUSIONS

In this thesis, the goal was to find the elements that make a game compelling and use this information to create an educational game for learning programming. The focus was on the children in the elementary school, because programming is going to be included in the Finnish curriculum for elementary school in the fall of 2016. Another goal was the creation of an education themed application using modern web technologies. For this reason, the implementation used HTML5 technologies and JavaScript as a programming language.

The work started with a basic background study on game elements and several articles and books on the topic were easily found. Less studies were found concerning the educational aspect of games and almost nothing on learning programming through playing games. In addition several educational games were inspected. Information, gathered from literacy and existing games, was used in the creation and analysis of an educational game. After implementing the game, it was tested with fourth graders in a normal elementary school in Akaa, Finland.

Finding the elements and creation the game were successful achieved. These same methods could be used to create and analyse other educational games. The results from field testing showed that at least attitudes towards programming could be influenced towards positivism using games. There was not really any improvement in analytical thinking that could be observed. Due to limited resources, the study was more like user testing, but it could be recreated with small modifications to the questionnaires in other studies to get more relevant data.

This thesis can be used as a starting point for somebody trying to delve into making educational games for programming or more generally gamifying the way programming is learned. The literacy part of the thesis can be useful for game designers in general. Educators can try to embrace game thinking in their lessons and broaden their thinking about educational games.

REFERENCES

[1] "gamification: definition of gamification in oxford dictionary (british & world english)," http://www.oxforddictionaries.com/definition/english/gamification, retrieved on April 04, 2014.

[2] "Lightbot," http://light-bot.com/hocflash.html, retrieved on March 03, 2014.

[3] J. Atwood, "Stack overflow & stack exchange: Programming programmers," http://fora.tv/2012/06/21/Stack_Overflow__Stack_Exchange_Programming_Programmers, retrieved on February 10, 2014.

[4] E. Bassilious, A. DeChamplain, I. McCabe, M. Stephan, B. Kapralos, F. Mahmud, and A. Dubrowski, "Power defense: A video game for improving diabetes numeracy," in *Games Innovation Conference (IGIC), 2011 IEEE International*, 2011, pp. 124–125.

[5] N. H. Battjes, "Sweatshop: A bizarre tower defense game with a message," http://indie-games-ichiban.wonderhowto.com/inspiration/sweatshop-bizarre-tower-defense-game-with-message-0128812/, retrieved on February 2, 2014.

[6] Code.org, "The hour of code," http://code.org/, retrieved on February 2, 2014.

[7] C. Crawford, *Chris Crawford on game design*. New Riders Publishing, 2003.

[8] M. Csíkszentmiályi, "Flow, the secret to happiness," http://video-subtitle.tedcdn.com/talk/podcast/2004/None/MihalyCsikszentmihalyi_2004-480p-en.mp4, retrieved on July 8, 2014.

[9] E. L. Deci and R. M. Ryan, "Motivation, personality, and development within embedded social contexts: An overview of self-determination theory," in *Oxford handbook of human motivation*, 2012, pp. 85–107.

[10] L. Doucet and V. Srinivasan, "Designing entertaining educational games using procedural rhetoric: A case study," in *Proceedings of the 5th ACM SIGGRAPH Symposium on Video Games*, ser. Sandbox '10. New York, NY, USA: ACM, 2010, pp. 5–10. [Online]. Available: http://doi.acm.org/10.1145/1836135.1836136

[11] J. Goldstein, "Play in children's development, health and well-being," http://www.ornes.nl/wp-content/uploads/2010/08/Play-in-children-s-development-health-and-well-being-feb-2012.pdf, retrieved on October 30, 2014.

[12] Google Trends, "Google trends - gamification," http://www.google.com/trends/explore?hl=en-US#q=gamification&date=1/2010+50m&cmpt=q, retrieved on February 14, 2014.

[13] I. Granic, A. Lobel, and R. C. M. E. Engels, "The benefits of playing video games," *American Psychologist*, vol. 69, no. 1, pp. 66–78, 2014, iD: 2013-42122-001.

[14] K. M. Kapp, *The Gamification of Learning and Instruction.* Pfeiffer, 2012.

[15] Kuato Studios, "Hakitzu elite: Robot hackers," https://play.google.com/store/apps/details?id=com.kuatostudios.hakitzu, note = Retrieved on March 16, 2014,.

[16] Learning Games For Kids, "Math man," http://www.learninggamesforkids.com/math_games/random-math/math-man.html, retrieved on February 22, 2014.

[17] C. Linehan, B. Kirman, S. Lawson, and G. Chan, "Practical, appropriate, empirically-validated guidelines for designing educational games," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '11. New York, NY, USA: ACM, 2011, pp. 1979–1988. [Online]. Available: http://doi.acm.org/10.1145/1978942.1979229

[18] Littleloud, "Sweatshop," http://www.playsweatshop.com/, retrieved on February 2, 2014.

[19] A. Marczewski, *Gamification: A Simple Introduction & a Bit More.* self-published, 2013.

[20] J. McGonigal, *Reality is Broken.* Vintage, 2012.

[21] S. Meier, "Interesting decisions," http://gdcvault.com/play/1015756/Interesting, retrieved on May 27, 2014.

[22] D. Michael and S. Chen, *Serious Games: Games That Educate, Train, and Inform.* Thomson Course Technology PTR, 2006.

[23] K. Mitgutsch and N. Alvarado, "Purposeful by design?: A serious game design assessment framework," in *Proceedings of the International Conference on the Foundations of Digital Games*, ser. FDG '12. New York, NY, USA: ACM, 2012, pp. 121–128. [Online]. Available: http://doi.acm.org/10.1145/2282338.2282364

[24] Novel Games Limited, "Math lines," http://www.learninggamesforkids.com/math_games/addition/math-lines-10.html, retrieved on February 22, 2014.

[25] Opetushallitus, "Perusopetuksen opetussuunnitelman perusteet 2004," http://www.oph.fi/download/139848_pops_web.pdf, retrieved on October 26, 2014.

[26] Opetushallitus, "Perusopetuksen opetussuunnitelman perusteet: Opetus vuosiluokilla 1-2. luonnos 19.9.2014," http://www.oph.fi/download/160360_opsluonnos_perusopetus_vuosiluokat_1_2_19092014.pdf, retrieved on October 26, 2014.

[27] N. Pelling, "The short prehistory of gamification," http://nanodome.wordpress.com/2011/08/09/the-short-prehistory-of-gamification/, retrieved on February 14, 2014.

[28] R. Rouse and S. Ogden, *Game design: theory & practice*, ser. Wordware game developer's library. Wordware Publishing, Inc., 2001.

[29] K. Salen and E. Zimmerman, *Rules of Play: Game Design Fundamentals.* The MIT Press, 2004.

[30] B. Suits, *The Grasshopper: Games, Life and Utopia.* Broadview Press, 2005.

[31] L. S. Vygotsky, *Mind in Society: The Development of Higher Psychological Processes.* Harvard University Press, 1978.

[32] K. Werbach and D. Hunter, *For The Win.* Wharton Digital Press, 2012.

APPENDIX A. QUESTIONNAIRES

Koulu 1 Ennen pelien testaamista ID:

Nimi_____

Tyttö / Poika (ympyröi valintasi)

Oletko ohjelmoinut: Kyllä / Ei (ympyröi valintasi)

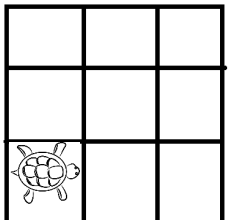
Vastaa oman mielipiteesi mukaan alla oleviin väitteisiin.

	Täysin eri mieltä	Eri mieltä	En osaa sanoa	Samaa mieltä	Täysin samaa mieltä
Ohjelmoinnin opettelu on helppoa					
Ohjelmointi on ainoastaan pojille					
Koulussa voisi olla enemmän ohjelmointiin liittyvää opetusta					
Pelit motivoisivat oppimaan paremmin					

Ohjelmointi on:

Piirrä kilpikonna uuteen sijaintiinsa käskyjen suorittamisen jälkeen:

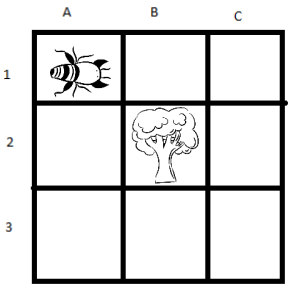
1. Liiku eteenpäin
2. Käänny vasemmalle
3. Liiku eteenpäin



Pääseekö termiitti puulle?

1. Liiku eteenpäin
2. Liiku eteenpäin
3. Käänny oikealle

Vastaus: _____



Koulu 1 Pelien testaamisen jälkeen ID:

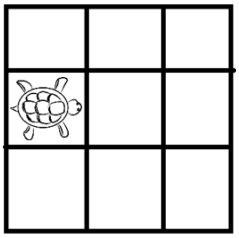
Vastaa oman mielipiteesi mukaan alla oleviin väitteisiin.

	Täysin eri mieltä	Eri mieltä	En osaa sanoa	Samaa mieltä	Täysin samaa mieltä
Ohjelmoinnin opettelu on helppoa					
Ohjelmointi on ainoastaan pojille					
Koulussa voisi olla enemmän ohjelmointiin liittyvää opetusta					
Pelit motivoisivat oppimaan paremmin					

Ohjelmointi on:

Piirrä kilpikonna uuteen sijaintiinsa käskyjen suorittamisen jälkeen:

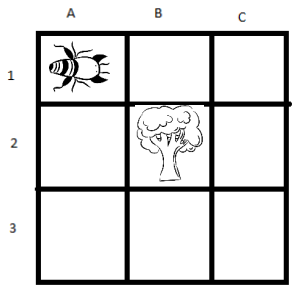
1. Liiku eteenpäin
2. Käänny oikealle
3. Liiku eteenpäin



Pääseekö termiitti puulle?

1. Liiku eteenpäin
2. Käänny oikealle
3. Liiku eteenpäin

Vastaus: _____



JATKUU TOISELLA PUOLELLA ->

Koulu 1 Pelien testaamisen jälkeen ID:

Kumpi oli parempi peli ja miksi? (ympyröi) Bookworms / Lightbot

Vapaita kommentteja pelien käyttämisestä ohjelmoinnin opettamisessa:

Kiitos osallistumistasi!