



Norwegian University of  
Science and Technology

# Factors affecting teens' academic engagement in computational thinking activities

**Uyen Dan Nguyen**

Master of Science in Informatics

Submission date: June 2018

Supervisor: Maria Letizia Jaccheri, IDI

Co-supervisor: Sofia Papavlasopoulou, IDI

Norwegian University of Science and Technology  
Department of Computer Science



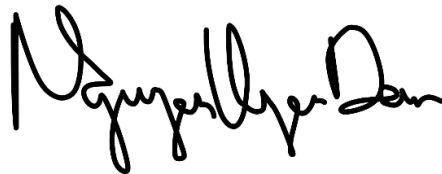
---

### *Acknowledgments*

This Master's thesis was written from August 2017 to June 2018 at the Norwegian University of Science and Technology (NTNU). Although there have been many moments of frustration, doubt, and stress, I had fun and enjoyed my time researching, interacting with others, and creating something for almost a year. First and foremost, I want to thank my dear family and friends for all the support they have given me. Especially the support from my parents who worked so hard to get me to this point has been remarkable. A special shout-out is directed to every person who read my thesis and provided help and proof-reading.

I would also like to thank the employees at NTNU for significant guidance, feedback, contributions, and advice throughout this Master's thesis period, in particular the assistance I received from my supervisors Maria Letizia Jaccheri and Sofia Papavlasopoulou. I will be forever grateful for the chance I got to write something personally meaningful.

Best regards,  
Uyen Dan Nguyen



A handwritten signature in black ink, appearing to read 'Nguyen Dan Uyen', written in a cursive style.

---

# Abstract

In our increasingly technological society, computational thinking is considered vital despite future career choices. Computational thinking can develop and enhance unique thinking skills, and educators all over the world are focused on how they can increase youths' engagement in the field. More insight is required to see what teens enjoy and are engaged by when learning and applying computational thinking, as an increased engagement can make youth more invested and involved in computing.

This study aimed to investigate teens' academic engagement in computational thinking activities by answering the research question: "Which factors can influence teens' engagement in computational thinking activities?" Academic engagement concerns how students are psychologically invested towards learning, understanding, or mastering knowledge and skills that the academic work is promoting. This study looked more into the features of academic engagement and how collaboration can affect teens' academic engagement.

By conducting two workshops called Kodeløypa and Tappetina, data was collected using observations, interviews, artifacts, and questionnaires, and thus used to detect factors that teens find motivating. The result of this study is a mapping of factors that affect teens' academic engagement. Some examples are teamwork and assistance, competence, motivation, perceptions, and learning. These findings can be used to assist educators, lecturers, and teachers in developing engaging and motivating computational thinking activities.

**Keywords:** Computational thinking, academic engagement, motivation, youth

---

# Sammendrag

I det stadige mer teknologiske samfunnet vårt, anses algoritmisk tankegang som viktig uavhengig av fremtidige karrierevalg. Algoritmisk tankegang kan utvikle og forbedre unike tenkeferdigheter, og lærere og undervisere verden rundt fokuserer nå på hvordan de kan øke ungdommens engasjement for dette. Innsikt om hva ungdommer blir motiverte av når de lærer og bruker slik algoritmisk tankegang er nødvendig, siden det kan gjøre dem mer investerte og involverte.

Denne studien har utforsket tenåringers akademiske engasjement i aktiviteter med algoritmisk tankegang ved å svare på forskningsspørsmålet: "Hvilke faktorer kan påvirke tenåringers engasjement i aktiviteter med algoritmisk tankegang?" Akademisk engasjement omhandler den psykologiske investeringen og innsatsen som studenter har for læring, forståelse eller mestring av kunnskap og ferdigheter som det akademiske arbeidet promoterer. Denne studien har sett på engasjementtypene som akademisk engasjement består av og hvordan samarbeid kan påvirke tenåringenes akademiske engasjement.

Gjennom to workshoper kalt Kodeløypa og Tappetina, ble data samlet inn ved hjelp av observasjoner, intervjuer, artefakter og spørreskjemaer, og deretter brukt for å finne faktorer som tenåringene mener er motiverende. Resultatet av studien er en kartlegging av faktorer som påvirker tenåringers akademiske engasjement. Noen eksempler er samarbeid og assistanse, kompetanse, motivasjon, oppfatninger, og læring. Disse funnene kan hjelpe lærere og undervisere i å utvikle engasjerende og motiverende aktiviteter med algoritmisk tankegang som fokus.

**Nøkkelord:** Algoritmisk tankegang, akademisk engasjement, motivasjon, ungdommer

# Table of Contents

<b>Abstract</b>	<b>i</b>
<b>Sammendrag</b>	<b>ii</b>
<b>Table of Contents</b>	<b>vi</b>
<b>List of Tables</b>	<b>ix</b>
<b>List of Figures</b>	<b>xii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	2
1.2 Personal motivation . . . . .	4
1.3 Research inquiry . . . . .	5
1.4 Research process . . . . .	6
1.5 Thesis outline . . . . .	6
<b>2 Background</b>	<b>8</b>
2.1 Learning theories . . . . .	9
2.1.1 Behavioristic learning theory . . . . .	9
2.1.2 Cognitivist learning theory . . . . .	10
2.1.3 Constructivist learning theory . . . . .	11
2.1.4 Constructionist learning theory . . . . .	11
2.2 Computational thinking . . . . .	13
2.2.1 Computational thinking in school . . . . .	14

---

2.2.2	Computational thinking and creativity . . . . .	15
2.2.3	Computational thinking and programming . . . . .	17
2.3	Visual programming environments . . . . .	19
2.3.1	Benefits of visual programming . . . . .	19
2.3.2	Scratch . . . . .	20
2.4	Academic engagement . . . . .	23
2.4.1	Behavioral engagement . . . . .	25
2.4.2	Emotional engagement . . . . .	27
2.4.3	Cognitive engagement . . . . .	28
2.4.4	Agentic engagement . . . . .	29
2.5	Teens' engagement in computational thinking activities . . . . .	30
2.6	Overview of contributions from related theories . . . . .	32
2.7	Overview of contributions from related studies . . . . .	37
<b>3</b>	<b>Methodology</b>	<b>42</b>
3.1	Research approach . . . . .	43
3.2	My position and preconceptions . . . . .	45
3.3	Data collection . . . . .	45
3.3.1	Kodeløypa outline . . . . .	45
3.3.2	Tappetina outline . . . . .	49
3.4	Sampling . . . . .	52
3.4.1	Kodeløypa sampling . . . . .	52
3.4.2	Tappetina sampling . . . . .	53
3.5	Data collection procedure . . . . .	54
3.5.1	Observation . . . . .	54
3.5.2	Interview . . . . .	56
3.5.3	Artifact . . . . .	58
3.5.4	Questionnaire . . . . .	59

---

---

3.6	Data analysis procedure . . . . .	60
3.6.1	Observation . . . . .	60
3.6.2	Interview . . . . .	61
3.6.3	Artifact . . . . .	64
3.6.4	Questionnaire . . . . .	67
3.7	Validity and reliability . . . . .	67
3.7.1	Validity of research design and data collection . . .	68
3.7.2	Validity of analysis . . . . .	70
3.7.3	Reliability . . . . .	70
3.8	Statement of ethics and confidentiality . . . . .	71
<b>4</b>	<b>Results</b>	<b>73</b>
4.1	Observations . . . . .	74
4.2	Interviews . . . . .	78
4.3	Artifacts . . . . .	86
4.4	Questionnaires . . . . .	91
<b>5</b>	<b>Discussion</b>	<b>94</b>
5.1	Teens' position in computational thinking . . . . .	95
5.2	Factors of academic engaging activities . . . . .	96
5.2.1	Teamwork and assistance . . . . .	97
5.2.2	Competence . . . . .	98
5.2.3	Motivation . . . . .	99
5.2.4	Perceptions . . . . .	100
5.2.5	Scratch . . . . .	101
5.2.6	Learning . . . . .	102
5.3	Addressing academic engagement types . . . . .	103
5.3.1	Behavioral engagement . . . . .	104
5.3.2	Emotional engagement . . . . .	105
5.3.3	Cognitive engagement . . . . .	107
5.3.4	Agentic engagement . . . . .	109

---



---

5.4	Implications and recommendations . . . . .	110
5.4.1	Generalizability and applicability . . . . .	110
5.4.2	Limitations . . . . .	113
5.4.3	Future research . . . . .	115
5.5	Closing words . . . . .	115
	<b>Bibliography</b>	<b>117</b>
	<b>Appendix</b>	<b>134</b>

# List of Tables

2.1	Overview of similar research and their contributions to this study . . . . .	33
2.2	Overview of similar research and their contributions to this study (continued) . . . . .	34
2.3	Overview of similar research and their contributions to this study (continued) . . . . .	35
2.4	Overview of similar research and their contributions to this study (continued) . . . . .	36
2.5	Overview of similar studies and their contributions to this study . . . . .	38
2.6	Overview of similar studies and their contributions to this study (continued) . . . . .	39
2.7	Overview of similar studies and their contributions to this study (continued) . . . . .	40
2.8	Overview of studies that address challenging computational thinking concepts for youth . . . . .	41
3.1	Kodeløypa statistics which include participation dates, class types, number of students, and team divisions . . . . .	52
3.2	Kodeløypa statistics which include participation dates, gender ratios, and class grades . . . . .	53
3.3	Tappetina statistics which include the class grades of the participants . . . . .	53

---

3.4	Table used to indicate the levels of help the teens received during the workshops . . . . .	56
3.5	Overview of the focus points used to link themes to engagement types . . . . .	64
4.1	Overview of the purposes of help and frequencies in the workshops . . . . .	77
4.2	Overview of the purposes of help and frequencies in the workshops (continued) . . . . .	78
4.3	Overview of categories and number of codes after coding Tappetina interviews . . . . .	84
4.4	Overview of categories and number of codes after coding Kodeløypa interviews . . . . .	85
D.1	Competence levels used by Dr.Scratch (Moreno-León et al., 2015) . . . . .	152
D.2	Competence levels used by Dr.Scratch (Moreno-León et al., 2015) (continued) . . . . .	153
E.1	Subcategories for collaboration, frequencies and examples	155
E.2	Subcategories for difficulties, frequencies and examples . .	156
E.3	Subcategories for difficulties, frequencies and examples (continued) . . . . .	157
E.4	Subcategories for out of school experience, frequencies and examples . . . . .	158
E.5	Subcategories for out of school experience, frequencies and examples . . . . .	159
E.6	Subcategories for fun, frequencies and examples . . . . .	160
E.7	Subcategories for achievement, frequencies and examples .	161
E.8	Subcategories for Scratch potential, frequencies and examples	162
E.9	Subcategories for help, frequencies and examples . . . . .	163

---

---

E.10	Subcategories for change of perceptions, frequencies and examples . . . . .	164
E.11	Subcategories for Scratch discoveries, frequencies and examples . . . . .	165
E.12	Subcategories for learning, frequencies and examples . . . . .	165
E.13	The intention category, frequencies and examples . . . . .	166
F.1	Subcategories for difficulties, frequencies and examples . . . . .	168
F.2	Subcategories for emotion, frequencies and examples . . . . .	169
F.3	Subcategories for Scratch, frequencies and examples . . . . .	170
F.4	Subcategories for collaboration, frequencies and examples . . . . .	171
F.5	The easy category, frequencies and examples . . . . .	172
F.6	The environmental issues category, frequencies and examples . . . . .	172
F.7	The time issues category, frequencies and examples . . . . .	173
H.1	Frequencies for prior attended visual programming courses before the Kodeløypa workshop . . . . .	176
H.2	Frequencies for Likert scores given during the Kodeløypa workshop . . . . .	177
I.1	Frequencies for prior attended visual programming courses before the Tappetina workshop . . . . .	178
I.2	Frequencies for Likert scores given during the Tappetina workshop . . . . .	179
I.3	Frequencies for Likert scores given during the Tappetina workshop (continued) . . . . .	179
I.4	Frequencies for Likert scores given during the Tappetina workshop (continued) . . . . .	180



# List of Figures

1.1	Main ingredients of thesis . . . . .	7
2.1	An example of how the graphical user interface of offline Scratch can look like . . . . .	21
2.2	Two examples on code block categories of Scratch . . . . .	22
3.1	An illustration of a robot from the Kodeløypa paper tutorial	47
3.2	An example robot from the Kodeløypa workshop . . . . .	47
3.3	An example of a Scratch script in the paper tutorial . . . . .	48
3.4	An example of a game created during the Kodeløypa workshop . . . . .	49
3.5	Tappetina participants working on their storyboard . . . . .	51
3.6	The streamlined codes-to-theory model by Saldaña (2015) which was used in the coding process . . . . .	62
4.1	The ratio of the indicated help the teens received in the workshops . . . . .	75
4.2	Final themes after coding interviews from both workshops	86
4.3	The ratio of Dr.Scratch scores for artifacts . . . . .	87
4.4	The ratio of Dr.Scratch scores for artifacts (continued) . . .	88
4.5	The ratio of Dr.Scratch scores for artifacts (continued) . . .	90
4.6	The number of previous visual programming activities before the teens participated in the Kodeløypa workshop . . .	91
4.7	The number of previous visual programming activities before the teens participated in the Tappetina workshop . . .	92

---

5.1	Placements of themes according to the presented academic engagement features . . . . .	103
5.2	The bullet points represent factors that affect academic engagement . . . . .	111
G.1	Clusters of coding categories made into themes . . . . .	175

# Chapter 1

## Introduction

This chapter describes the motivations, research inquiry, research process, and outline of this study. The former includes the author's personal motivation and thoughts for conducting this study. Section 1.1 and Section 1.2 clarify the motivations, while Section 1.3 introduces the research inquiry of this study. Section 1.4 specifies this study's research process in a brief way, while Section 1.3 outlines the succeeding chapters of this thesis.



## 1.1 Motivation

Digital technology is becoming more common in all aspects of society, and sets new demands for digital skills (Utdanningsdirektoratet, 2016). Regardless of future profession choices, some basic knowledge of technology and its applications are needed in our increasingly technological society. Educators need to prepare students for their future, and a way to do so is to teach them *computational thinking* (Wing, 2006). Despite the lack of a universal definition among researchers, many computational thinking definitions include problem-solving, algorithmic thinking, and recognizing how existing knowledge can be transferred to new situations. Programming as a tool can be used to promote this kind of thinking (Yadav et al., 2016).

The importance of developing digital skills is now reflected in curriculum all over the world (Hubwieser et al., 2015). In the USA, the CS10K initiative by NFS has aimed to employ 10,000 new Computer Science teachers in U.S high schools. Foundations for Advancing Computational Thinking (FACT), Exploring Computer Science (ECS), and AP CS Principles are some courses that target the fostering of computational skills in students (Grover and Pea, 2013; Grover et al., 2014a). Another example is Computing At School, a growing organization for improving the teaching of Computer Science in UK schools (Brown et al., 2013). Norwegian curriculum is also found to target the need for digital skills. In Fall 2016, 146 middle schools rolled out an elective programming course called *Valgfag i Programmering* (Utdanningsdirektoratet, 2016). The goal of this course has been to teach teens how computers and programs work, alternative uses of programming languages, and basic principles in programming and computational thinking.

Various studies look at ways to teach youth Computer Science and computational thinking (Resnick et al., 2009; Repenning et al., 2010; Lee et al.,

2011; Grover et al., 2014a; Voogt et al., 2015; Weintrop and Wilensky, 2015; Garneli et al., 2015; Yadav et al., 2016). Researchers have found that visual programming environments can teach novice users the basics of computing without being overwhelmed by syntaxes of textual-based programming (Resnick et al., 2009; Adams and Webster, 2012; Grover and Pea, 2013; Mannila et al., 2014). Studies by Meerbaum-Salant et al. (2010) and Franklin et al. (2013) focused on the visual programming language Scratch, and the assessment of learning with it. Somehow similarly, Burke and Kafai (2012) investigated the types and frequency of programming concepts utilized in different kinds of Scratch projects.

Researchers have also explored student performances in computational thinking activities with visual programming (Denner et al., 2012; Armoni et al., 2015; Fields et al., 2016). Brennan and Resnick (2012) expressed the need for multiple means of assessing student-created artifacts. Grover et al. (2014a) agreed that there is a need for more objective assessment instruments to illuminate student understanding of computing concepts and computational thinking skills.

Several studies examine how we can engage teens in computational thinking activities (Kelleher et al., 2007; Maloney et al., 2010; Giannakos and Jaccheri, 2013; Giannakos et al., 2013). According to Giannakos et al. (2013), creativity has been pointed out as a positive factor in learning programming and Computer Science, and in agreement with Kelleher et al. (2007), they found that a storytelling approach could increase the interests for programming. While many studies look at how they can broaden participation in computing, not many have looked into specific factors that can increase engagement in computational thinking activities. This study is built on the lack of insight regarding particular characteristics that can increase engagement for such activities. This research brings light to different factors that affect students' engagement for academic work, which contributes to filling a gap of absent research on quality of engagement,

and how to deepen participation in computing (Fields et al., 2014).

This study can benefit educators, teachers, and researchers who want to engage teens in learning computational thinking. The goal should be to motivate them to a great extent, so they want to continue to learn more.

## **1.2 Personal motivation**

This exploration of mine started as an interest in education, learning, and Computer Science. Long before I enrolled as an undergraduate at my university, I was interested in kids and how they learn, mainly through mentoring, being an assistant coach for a basketball team of girls, and teaching at summer school. When I went to primary school, I learned to create images in Paint, write documents in Word, play around with Wordart, and then save my projects on diskettes. The role of technology has however evolved since then, and we now live in a technology-centered society where kids know how to use a lot of different technologies, but not necessarily how they are built up and how they can benefit from it.

Technology will have a more significant impact on us and our community in the future (Lye and Koh, 2014). There will be more engineers, new findings, and improvements in current education. Accordingly, we need more people interested in fields such as Computer Science, computing, and engineering. By motivating children to develop an interest in computational thinking early in education and childhood, more people will be increasingly interested and motivated to continue on this path. To make children enjoy computational thinking tasks, we need to look further into how we can engage and motivate them. Like Koca (2016) expressed, motivation has shown to be an important, if not the most essential, element in academic success.

This thesis was written for people who are interested in learning and teaching technology and Computer Science, and individuals who feel that we need more adaption of curriculum with the pace of our technological society. To people similar to me who went to school and enjoyed creating something digitally, rather than painting, drawing, and knitting, and to those who enjoyed both. This thesis is also available to the curious people who just like to read.

## 1.3 Research inquiry

This study will examine factors that influence teens' engagement in computational thinking activities in order to increase interest and participation in relevant fields. The research question is hence: *Which factors can influence teens' engagement in computational thinking activities?* The specific engagement this study will focus on is academic engagement, which concerns the psychological investment and efforts students have towards learning, understanding, or mastering knowledge, skills, and crafts that academic work is intended to promote (Turner et al., 2014).

To answer the research question, this study will use the popular visual programming environment Scratch and look at factors that encourage teens to participate in computational thinking activities. Furthermore, this research will look at what role collaboration has for teens' engagement. In other words, the research inquiry is as follows:

- **RQ1: Which factors can influence teens' engagement in computational thinking activities?**
  - SQ1.1: What role does collaboration play in teens' engagement?

## **1.4 Research process**

To undertake the research inquiry, the research is based on the constructionist paradigm and on related theories and studies.

To collect data, two out of school workshops called Kodeløypa and Tapetina were studied, which were conducted by researchers at the author's university, Norwegian University of Science and Technology (NTNU) in Trondheim, Norway. The data collection followed a mixed methods approach and included instruments such as observation, interview, artifact, and questionnaire.

To analyze data, data collected with different instruments underwent different processes. Detailed descriptions will be described in Chapter 3.

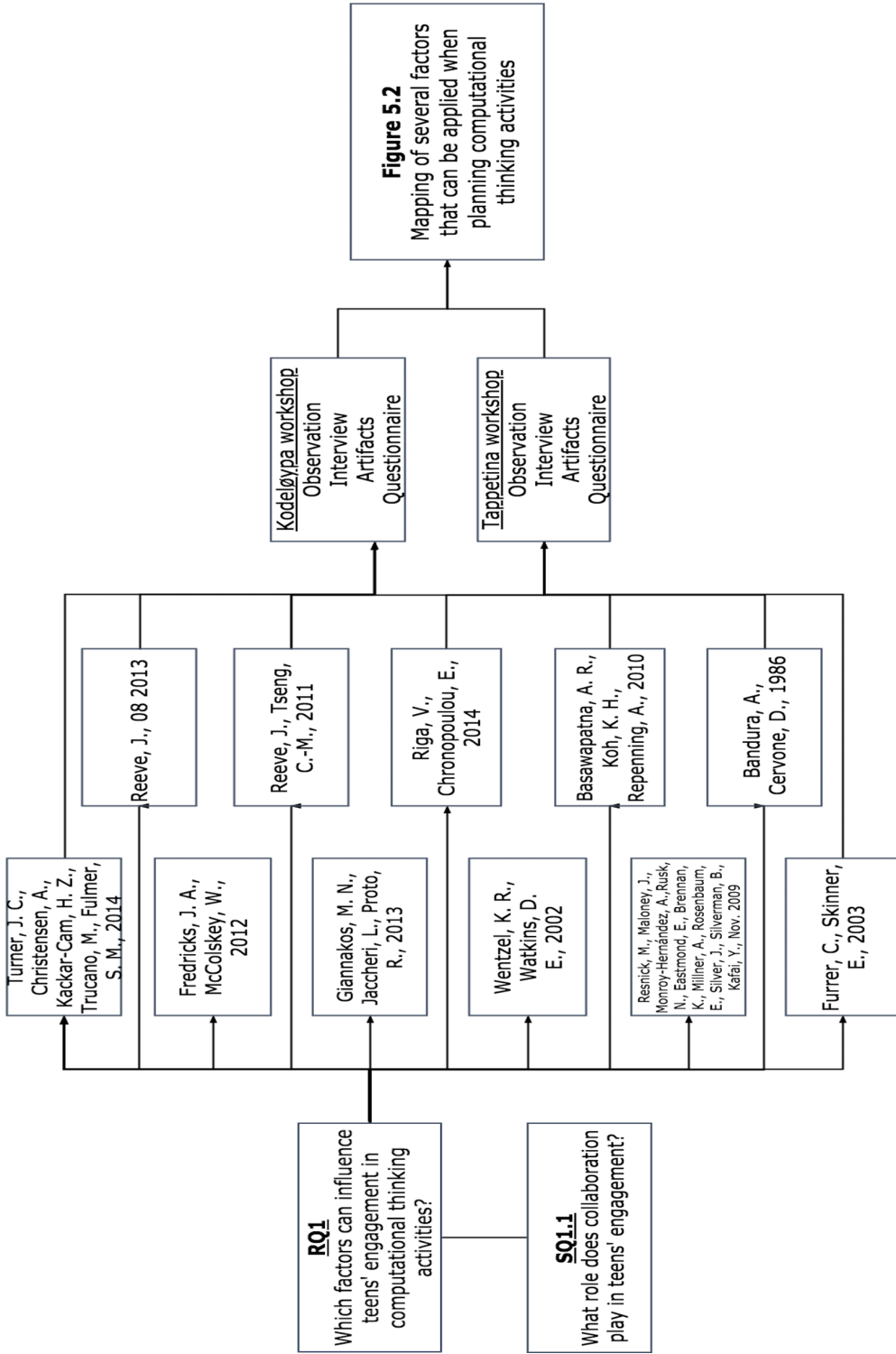
Figure 1.1 illustrates the main ingredients of this thesis and includes the research inquiry, the most relevant research and theories, the conducted investigations in this study, and finally this study's contributions to research.

## **1.5 Thesis outline**

Chapter 2 introduces relevant concepts, theories, and previous research that are applicable for exploring the research inquiry.

Chapter 3 describes the methodology for collecting and analyzing the data in this study. This chapter contains an elaboration of how validity, reliability, ethics, and confidentiality have been ensured in this study.

Chapter 4 presents the results of the data collection, while Chapter 5 finally discusses the results and how they relate to the research inquiry, the generalizability and applicability of the findings, limitations, possible future work, and lastly this study's closing words.



**Figure 1.1:** Main ingredients of thesis

## Background

This chapter outlines theories, concepts, and studies relevant to the objectives of this research, and is a base for succeeding chapters and choices made in this study. It is essential to look at how literature defines and describes different terms in order to answer the research inquiry. The theoretical part is sectioned as follows: Section 2.1 looks at four essential learning theories and presents some of the most distinct differences between them. Section 2.2 reflects on the existing definitions of computational thinking, and reasons for introducing computational thinking to schools. The following subsections introduce programming and creativity as two ways to maintain and improve computational thinking skills. Section 2.3 describes visual programming environments, and how they are considered a more playful and attractive way for novice users to learn computational thinking, Computer Science, and programming. This section also includes a paragraph regarding Scratch, which will be used later in this study. Section 2.4 will explain academic engagement and how students can be motivated to learn. Section 2.5 looks at how related studies have addressed youth's engagement in computational thinking activities. Finally, the last two sections present how research and studies have contributed to this study.

## 2.1 Learning theories

Learning has been defined in various ways by researchers, and although there are common elements in many of them, there is no existing universal agreement. A definition by Shuell, but later interpreted by Schunk in 1991, incorporates a lot of these elements (Ertmer and Newby, 2013):

*"Learning is an enduring change in behavior, or in the capacity to behave in a given fashion, which results from practice or other forms of experience".*

The way we define learning and how we believe learning occurs is therefore crucial for situations where we want to change how people do or know something. Instructional designs with verified instructional strategies and techniques for facilitating learning are called learning theories or learning paradigms. The different learning theories can emphasize factors variously, such as whether and how much the learner contributes to the learning processes, how learning is obtained, and how the environment affects learning. Three popular learning theories are behaviorism, cognitivism, and constructivism. These are considered the most utilized learning paradigms when creating instructional environments (Siemens, 2014). Another widespread learning theory is constructionism, which is commonly used in educational studies and strategies associated with how children learn with computers (Bers et al., 2002; Denner et al., 2012).

### 2.1.1 Behavioristic learning theory

Learning in behavioristic learning theory is facilitated when there are changes in form or frequency of an observable performance, and a learner demonstrates a proper response to a presentation of a specific environmental stim-



ulus (Ertmer and Newby, 2013). The critical elements of behavioristic learning are the stimulus, the response, and how the association between them is made, strengthened, and maintained. The learner is reactive to conditions in the environment, and whenever the learner starts generalizing, behaviorists say that the learner transfers knowledge from one situation to another. Regardless, it is generally agreed that the behavioristic learning theory cannot explain the acquisition of higher level skills or in-depth processing. In other words, behavioristic learning theory is about behavioral change and the passive absorption of knowledge, through repetition and both positive and negative encouragements (Ertmer and Newby, 2013; Lorås, 2017).

### **2.1.2 Cognitivist learning theory**

In the late 1950s, learning theory started to shift over to cognitive sciences, and psychologists and educators started to emphasize on complex cognitive processes, including thinking, problem-solving, and information processing (Ertmer and Newby, 2013). Learning for the cognitivist is related to discrete changes between states of knowledge, as opposed to changes in the probability of a response. This paradigm is more concerned with what learners know and how they know it, instead of what they do. Knowledge is made meaningful for the learner, such that he can organize, code, and relate new information to existing knowledge in the memory. The learner is perceived active in a learning process, and his thoughts, attitudes, and values, along with environmental conditions are essential for facilitating learning. When a learner has understood how to apply knowledge to different situations, transfers have occurred. To conclude, the cognitivist learning theory views knowledge as constructed cognitive structures within the mind of the learner, and that learning actively adds new experiences to existing knowledge (Ertmer and Newby, 2013).

### **2.1.3 Constructivist learning theory**

Constructivism is a learning paradigm which claims that students construct knowledge, rather than merely receive and store knowledge transmitted by the teacher (Ben-Ari, 1998). Learners build personal interpretations of the world based on experiences and interactions, meaning that actual experiences have to be examined for each individual to see if learning has occurred (Ertmer and Newby, 2013). This construction builds recursively on the knowledge that the student already possesses, and both learner and environmental factors are critical to the constructivist. It is thus the interaction between the learner and the environmental factors that create knowledge. Learning must be active by students, and they must construct knowledge through guidance from teachers, and feedback from other students. Classrooms that consider messiness and complexity in real-life learning will be more effective in preparing learners for life-long learning (Siemens, 2014). The focus of constructivism is to create cognitive tools which reflect on the insights and experiences of individuals, and the settings where the tools are used. This paradigm assists learners in actively exploring complex topics, and have a higher chance of making learners think as domain experts would. Constructivism promotes collaboration and social interaction, as this can promote that multiple perspectives are used to solve a specific problem (Ertmer and Newby, 2013; Lorås, 2017).

### **2.1.4 Constructionist learning theory**

Constructionist learning theory believes that learning is to build new knowledge on existing knowledge, regardless of the circumstances of the learning (Papert and Harel, 1991). It supports the constructivist ideas that the learner is an active builder of knowledge and their cognitive tools. According to Kafai and Resnick (1996) and Ackermann (2001), the construction-

ist theory goes beyond constructivism, as it focuses on making artifacts, asserting that learners are engaged and that they understand the meaning of constructing the artifacts. In addition to this, the constructionist learning theory identifies a strong bond between design and learning, and that designing provides a rich context for learning. Bers et al. (2002) present constructionism as built up of four principles. The first is learning by designing meaningful projects to share in the community. The second is to use concrete objects to build and explore the world. The third is to identify powerful ideas that are both personally and epistemologically significant for the learners, and the fourth is to emphasize self-reflection as a part of learners' learning processes. It is common that these principles are present in early childhood education, which makes a constructionist approach suitable for introducing, for instance, technology to kids (Bers et al., 2002).

In his book *Mindstorms*, Papert (1980) introduced an idea of constructionism where projections of inner feelings and ideas are vital for learning. The learning process is an iterative process where learners use the tools and meditations that best support the exploration of what they care most about. It is therefore fundamental for students to recognize why it is essential to learn what they are learning. By being personally involved in situations instead of looking from a distance, the learner will feel a connection, which counts as a powerful mean of gaining understanding (Papert, 1980). A learner in this paradigm will outgrow their current views of the world, and construct a deeper understanding of themselves and their environments. A constructionist is situated, connected, and sensitive to variants in the environment, and enjoys gaining understanding from singular cases, rather than extracting and applying general rules (Kafai and Resnick, 1996). A learner must therefore embrace unknown situations, be relational, and collaborate with others.

## 2.2 Computational thinking

Despite the lack of universal agreement for the term computational thinking (Barr and Stephenson, 2011; Grover and Pea, 2013; Voogt et al., 2015), it was first introduced and made famous by Wing (2006). Wing (2006) defined computational thinking as a fundamental skill for everyone, not just for computer scientists: It involves problem-solving, pattern recognition, algorithmic design, understanding humans, and thinking recursively. Other scientists argued that computational thinking is about accomplishing a defined task with rigorous analysis and procedures, while some expressed that it is the study of the mechanisms of intelligence that can yield practical applications by magnifying human intelligence (Council, 2010). Lu and Fletcher (2009) suggested that computational thinking is *not* about getting humans to think like computers, but rather about developing the full set of mental tools necessary to effectively use computing to solve complex human problems. Similar to Wing (2006), Aho (2012) argued that computational thinking is the thought processes involved in formulating problems so that solutions are represented as computational steps and algorithms. Another interpretation was presented by Yadav et al. (2016), which draws similarities to many of the mentioned definitions. It consists of four essential and concrete concepts of computational thinking, where the first concept is to break down complex problems into more manageable sub-problems (problem composition), and the second concept is to use a sequence of steps to solve problems (algorithmic thinking). The third concept is to review how a solution transfers to similar problems (abstraction), and finally the fourth is to determine if computers can help us solve problems more efficiently (automation).

### **2.2.1 Computational thinking in school**

Lye and Koh (2014) predicted that current students would be working in fields influenced by computing in the future, and thus deal with computing and computational thinking in their everyday life. The lack of universal interpretation and how educators should treat computational thinking as a discipline have however made it difficult for school curriculum to find common ground (Grover and Pea, 2013). Researchers have discovered that computational thinking can develop unique thinking skills, in addition to enhance existing skills in mathematics, engineering, problem-solving, and even design thinking (Barr et al., 2011; Grover and Pea, 2013). Basic knowledge of computational thinking should therefore be mandatory to position children better in a world where computing can be considered as relevant and critical as basic knowledge in science and math (Grover and Pea, 2013). In addition to this, researchers agree that computational thinking concepts should be introduced earlier in education since it is no longer sufficient to wait until students are in college (Wing, 2008; Barr and Stephenson, 2011).

However, many students are already and unknowingly taught a set of computational skills in various disciplines in school (Millians, 2011; Beauchamp, 2016). Projects and frameworks such as Leveraging Thought Leadership for Computational Thinking in PK-12, the CSTA/ISTE framework, and the CS Principles framework are some of the steps towards giving all students the opportunity to learn computational thinking skills (Barr et al., 2011; Voogt et al., 2015; Yadav et al., 2016).

### 2.2.2 Computational thinking and creativity

Creativity has in literature been described in numerous ways, although it is difficult to define and measure. For simplicity, creativity is considered a mental process that involves generation of new ideas or concepts, or new associations between existing ideas or concepts (Jackson et al., 2012). The past few years, educators in the field of Computer Science have started to recognize the importance of incorporating creative processes as one of the big themes of Computer Science (DfE, 2013). Computing has thus shifted into a field that promotes exploration and creation of knowledge, enables innovation, and allows individuals to create personally meaningful artifacts (Mishra et al., 2013).

According to Resnick et al. (2009), a computational thinker sees computation as more than something to consume; it is considered something they can use for design and self-expression. In an ICT2Think project funded by NSF, researchers have proposed that learning computational thinking can be improved when paired with creativity (Shell et al., 2014). According to Shell et al. (2014), *computational creativity* is based on the idea that students who possess computational thinking skills can leverage them to improve their creative thinking skills, and vice versa. Computational creativity can also improve student achievement, learning, self-regulation, and engagement. Supplementary, Voogt et al. (2015) stated that with Computer Science knowledge, imaginative capacities such as innovative thinking and curiosity could lead to more success in using computational thinking. By using computational thinking skills with creative skills, researchers believe that students move from being technology consumers to tool builders (Mishra et al., 2013). This way, the tools can be used to create new forms of expression, in addition to benefit the society.

Researchers consider school as a place that develops and supplies the needs of the society, and schools should therefore produce students with creative thinking skills (Soh, 2017). However, children's creativity is underprioritized in education, and instead of focusing on the development of problem-solving skills, creative thinking, and decision-making abilities, educational systems seem to focus on students' recall and reproduction abilities (Riga and Chronopoulou, 2014). Research implies that schools are not fulfilling the creative potential of students and that highly creative persons often feel isolated, misunderstood, or unappreciated because of the strict structures and restrictions in educational contexts (Hennessey, 2000; Runco et al., 2017). In a study by Yazzie-Mintz (2007), 75% of the students who answered that they had been bored at school said that the material was not interesting enough, and that their engagement within the classroom was affected by the little interesting academic content. While the creativity level in younger students decreases at home rather than in schools, the decrease of creativity in upper school elementary can be explained by the increased focus on standardized tests (Kim, 2011). The high-stakes testing environment has led to the elimination of content areas and activities in school, leaving little room for imagination, problem-solving and creative thinking for the students. Students who do not fulfill their creative needs can become underachievers and express lower levels of educational attainment. Supported by self-determination theorists, Riga and Chronopoulou (2014) stated that only a creative environment can create the conditions for creative thinking to flourish, and that the biggest effect on children's creativity and behavior is related to alternative uses of ideas and materials, use of improvisation, testing hypothetical situations, and the use of problem-solving (Vygotsky, 1990; Deci et al., 1991).

### **2.2.3 Computational thinking and programming**

Several researchers believe that computational thinking is to develop thinking skills in subjects beyond Computer Science and according to Lu and Fletcher (2009), a way to do so is with programming. Researchers have found that introducing computational thinking before programming can spark interest, and avoid problems that undergraduates in introductory Computer Science courses are known to face (Lee et al., 2011; Grover and Pea, 2013; Han et al., 2015; Rich and Hodges, 2017). Considering that computational thinking skills are learned and practiced in education in the future, programming can be used to expose students to computational thinking, work as a way to detect computational competencies, and a measure to expose students to creativity, as programming is considered a creative process (Grover and Pea, 2013; Lye and Koh, 2014).

Programming as a tool can also be used to support the cognitive tasks in computational thinking (Resnick et al., 2009; Barr et al., 2011; Grover and Pea, 2013). Somehow similarly, Papert (1980) reported that children who programmed with the programming language LOGO could develop powerful procedural thinking skills identical to computational thinking skills. The author described that a child would start to think through programming, become an epistemologist, and thus able to transfer the earned skills in non-programming contexts in and out of school. Despite several studies contradicting the findings of Papert's original theories (Mldlan and et al., 1986), researchers have adopted the idea of using programming instruction to teach mental skills and computational thinking (Voogt et al., 2015). According to Resnick and Silverman (2005), programming can also extend what children are capable of creating, design, and invent with a computer. They can gain experience in using and manipulating formal systems, and transfer this to other domains.



### **Computational participation**

According to Kafai and Burke (2013) and Fields et al. (2014), there has been a social shift for learning programming the past few years. While it was more common to learn programming out of individualistic, tool-oriented, and self-centric reasons in the past, applications are progressively developed to design artifacts of true significance for others. Technology and programming have moved into something more sociological and cultural, making it normal to create digital media, and share it with communities or with other people. There is a chance that this shift called *computational participation* is influenced by the increasing interest in computational thinking in educational environments, and the growing need for self-made (DIY) projects among teens. With this in mind, Kafai and Burke (2013) expressed a need for turning computational thinking and programming to more social acts, and "a communal practice steeply grounded in how we think about what students today should learn to become full participants in their respective communities", rather than isolated, individualistic acts. To continue broadening participation in computing successfully, and on a larger scale than done before, new guidelines for designing activities, tools, and communities in educational environments are essential. Furthermore, while it is important to broaden participation, there is also a need for investigating the quality of engagement among teens in order to deepen participation (Fields et al., 2014).

## 2.3 Visual programming environments

Nowadays, programming is packed in different sorts of environments that are useful when learning computational thinking, and one example is visual programming tools (Mannila et al., 2014). These environments are popular and can teach people essential 21st-century skills such as thinking creatively, how to reason systematically, problem-solving skills, and concepts in mathematics and computation (Resnick et al., 2009). With these environments, youth are enabled to construct running programs through simplified programming interfaces. They can consequently learn central Computer Science concepts by using drag and drop blocks of code (Adams and Webster, 2012; Grover and Pea, 2013; Weintrop and Wilensky, 2015; Armoni et al., 2015).

### 2.3.1 Benefits of visual programming

According to Repenning et al. (2010), effective computational thinking tools for school children have *low floor* and *high ceiling*. These concepts have existed since the time of LOGO programming and are considered a guiding principle for creating programming languages for children (Papert, 1980; Resnick et al., 2009). Low floor implies that it should be relatively easy for a novice user to cross the threshold of creating working programs, while high ceiling suggests that programming environments should be extensive and powerful enough to satisfy the needs of advanced programmers. Programming environments should also scaffold flow, enable transfer, support equity, and be systematic and sustainable (Repenning et al., 2010; Grover and Pea, 2013). Several visual programming environments such as Scratch, Alice, Snap!, and Blockly fit these requirements to a certain degree (Weintrop and Wilensky, 2015). These platforms target a younger audience by introducing animations and graphics: By turning variables and

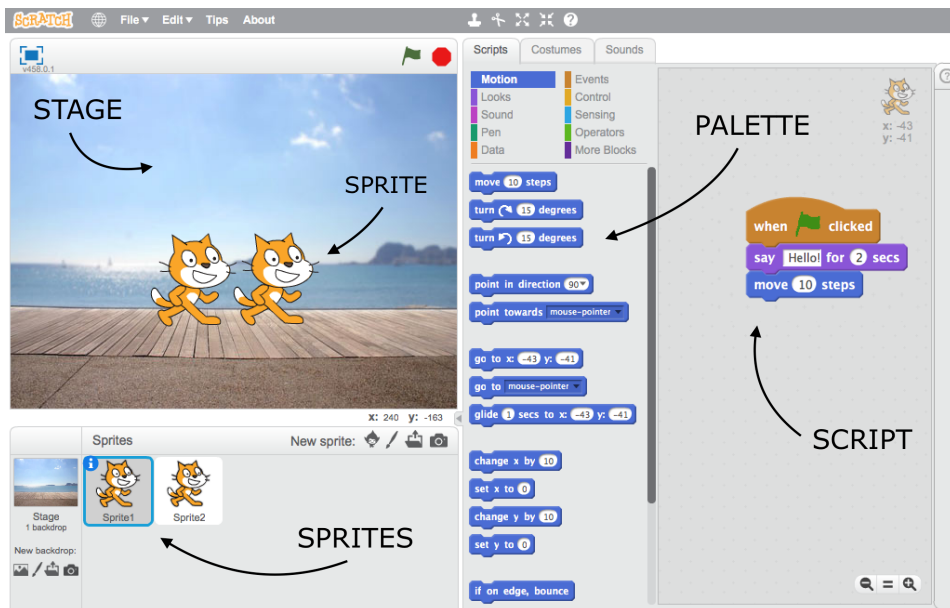
commands into concrete objects users can see and manipulate, it allows users to connect certain code blocks to specific manipulations. Concrete objects can also help users understand the concepts of programming easier, thus support computational thinking and enable that they learn the basics instead of memorizing text-based code (Kalelioğlu and Gülbahar, 2014; Lye and Koh, 2014; Han et al., 2015). Although Kelleher et al. (2007) found little to no differences in student performance for block-based and text-based programming, visual programming environments can be viewed as more beneficial and attractive for novice learners by lowering floors, and becoming more playful, emerging, and meaningful to users (Maloney et al., 2010; Resnick et al., 2009)

### **2.3.2 Scratch**

Being unsatisfied by existing programming languages, Scratch was created by the Lifelong Kindergarten group at the MIT Media Laboratory. The goal of the environment is to nurture a new generation of creative, systematic thinkers comfortable with using programming to express their ideas (Resnick et al., 2009). The environment is available in 50 languages and counting, and Scratch offers a highly interactive environment where it is possible to experiment incrementally and iteratively, online and offline. In 2009 and 2010, the creators of Scratch reported that more than 1,500 new projects were uploaded to the site every day, meaning more than one new project every minute (Resnick et al., 2009; Maloney et al., 2010). In addition to this, Scratch had 90,000 active monthly users as of July 2014, up from 20,000 only two years prior to that, making Scratch by far the most extensive online programming community for youth (Fields et al., 2014).

*Scripts* in Scratch projects are programs built up by drag and drop code blocks. The code blocks represent program concepts such as expressions, conditions, statements, and variables (Meerbaum-Salant et al., 2010; Ar-

moni et al., 2015). A green flag in the editor can be clicked to execute scripts. How the offline editor looks like can be seen in Figure 2.1. The editor consists of four distinct areas, and on the very left of the editor is the current *stage*, which is the last clicked background. In the area below, Scratch users can click on existing stages and *sprites*, which are figures in Scratch. Very right in the image is the area for scripts. Code blocks from the palette are dragged and dropped into this area. The clicked code category in the palette is motion, which naturally contains the code blocks related to motion. In Figure 2.2a and Figure 2.2b, the code blocks for events and sensing categories are presented. The environment gives immediate visual feedback through the behaviors of sprites when scripts are executed. All sprites and stages can have scripts.



**Figure 2.1:** An example of how the graphical user interface of offline Scratch can look like



(a) Events category

(b) Control category

**Figure 2.2:** Two examples on code block categories of Scratch

According to the creators, Scratch offers three core principles different from other programming environments: It is more tinkerable, more meaningful, and more social. It supports diversity and personalization, and make children more eager to learn concepts related to their projects. Additional to this, Scratch offers a large community whereas people can support,

collaborate, remix, and critique others' works. With the tool, users can continue to experiment with new forms of self-expression, and produce a diverse range of projects while deepening their understanding of a core set of computational ideas. With its wide walls, people with different interests and learning styles can benefit from the environment (Resnick et al., 2009).

## 2.4 Academic engagement

Academic engagement is a concept that has gained much traction as a possible measure for increasing academic motivation and achievement (Fredricks et al., 2004). Academic engagement is related to how students are psychologically invested in or use effort towards learning, understanding, or mastering knowledge, skills, and crafts that academic work is intended to promote. Research has found that students will be more engaged when the academic learning is valued and meaningful to them (Turner et al., 2014). As previously mentioned, academic engagement will be the focus of this study. Opportunities for academic engagement are found in the adaption of needs between a teacher and student, and when students think and make decisions. Researchers have described academic engagement as a multi-construct of the following features or engagement types: *Behavioral engagement*, *emotional engagement*, *cognitive engagement*, and finally *agentic engagement* (Fredricks et al., 2004; Reeve and Tseng, 2011; Fredricks and McColskey, 2012; Reeve, 2013). The engagement types will be explained further in the next subsections.

The following needs have also been found to be motivational for students' academic engagement: Autonomy, competence, relatedness, and meaningfulness. Autonomy includes to be self-initiating and self-regulating of one's actions, and to behave according to personal interests and values (Deci et al., 1991; Turner et al., 2014). It is related to student outcomes, greater

interests in lessons, and reports of enjoyment in academic work. Autonomy increases when students are given enough time, reflection, feedback, exploration, and acknowledgments of their perspectives. In a field study, Deci et al. (1981) found that students who are more exposed to controlling-promoting techniques rather than autonomy-promoting ones from their teachers, are more likely to be disinterested in schoolwork, have preferences for easy rather than challenging tasks, as well as a desire to please the teacher rather than working on their interests and curiosity. Competence is an important requirement for meeting one's goals and involves understanding how to attain various external and internal outcomes, as well as being effective in performing the requisite actions (Deci et al., 1991). The perceptions of competence are proven to contribute to students' achievements (Boggiano et al., 1992). Self-efficacy is a theory in competence motivation theory and defined as the belief that one can successfully organize and perform a particular task. It evolves when students find a correlation between positive outcomes and their efforts. The primary arena for the development of competence is school, and by providing students with appropriately challenging tasks, scaffolding, and informational feedback, competence can increase. Relatedness is also known as belongingness and is the need to establish close, secure and satisfying relationships with others (Deci et al., 1991). The relations between teachers and other students make unique contributions to emotional engagement, thus able to lead to increased motivation. Wentzel and Watkins (2002) argued that the processes between peers can promote positive academic outcomes at school, and that task engagement increases through positive feedback and encouragement in collaborative learning contexts. Approaches to increase belongingness are for instance to model and encourage mutual respect and productive collaborations. Whenever students are working productively in cooperative groups, they are more likely to participate, to develop positive attitudes towards others and the content, and to exert more effort.

Meaningfulness involves developing interests in or appreciation for the associated content, and experiencing its authentic application (Turner et al., 2014). Students' perceptions of meaningfulness are related to their interests and values (Grover et al., 2014b), and by helping students to understand the relationship between the relevance of school tasks and their personal goals, teachers can foster engagement. Meaningful learning can be characterized by building on students' prior knowledge, providing arenas for complex thinking, and possibilities to participate in extended conversations that build shared understanding.

### **2.4.1 Behavioral engagement**

Behavioral engagement includes participation, involvement, effort, and attention, and it can range from merely doing academic work, or participating in higher order student councils (Fredricks et al., 2004). Student behavioral engagement is a vital condition which supports academic achievement, and many studies reveal that behavioral disengagement is a risk factor for concurrent and later low academic achievements (Gregory et al., 2014). Wentzel and Watkins (2002) argued that learning is linked to the social contexts where children learn, and that peer tutoring improves children's self-esteem and attitudes towards school. Improvements of self-esteem and attitudes towards school are especially noticeable in situations where less competent students are provided with opportunities of active participation, and elaborate explanations and encouragements from a more experienced partner. The challenging part for a more advanced partner would be to provide positive feedback to motivate the less advanced partner, to bridge the gap between current skill and desired skill. However, Juvonen et al. (2012) found that students' levels of engagement influence their peers' academic performance and involvement, and that the interactions and support students receive from their peers are essential, particularly in challenging



situations. Wentzel and Watkins (2002) reported that students tend to turn to others for information and ways to cope with uncertainty and ambiguity, while Riga and Chronopoulou (2014) described that activities with effective group dynamics, communication, and emotional interactions can result in enjoyment, confidence, and free expression.

In contrast, Furrer and Skinner (2003) stated that children who feel disconnected from their key peers find it difficult to become involved in academic work, and are easily bored and frustrated. While feelings of relatedness to partners can awake enthusiasm and increase willingness to participate, children who show low relatedness are found to have lower enthusiasm and decrements of motivation over time. Specifically larger groups for collaboration have been pointed out to affect student motivation and behavioral engagement in learning negatively (Wentzel and Watkins, 2002; Downer et al., 2007).

The relationship between a teacher and a student are also found to affect student behavioral engagement (Skinner and Belmont, 1993; Furrer and Skinner, 2003; Reeve, 2012; Pianta et al., 2012; Gregory et al., 2014). According to Furrer and Skinner (2003), students who feel appreciated by their teachers are more likely to report positive feelings towards the academic work. On the other side, teachers tend to respond more positively to children who report high behavioral engagement, while they respond to passive children with more neglect, coercion, and inconsistency (Skinner and Belmont, 1993). Researchers have also found that teachers who also improve their feedback interactions with students, get more skilled at actively facilitating student involvement, and provide students opportunities to use higher level thinking skills result in a positive increase of students' behavioral engagement (Pianta et al., 2012). In other words, an adequate level of complexity for challenges and exciting tasks are considered positive teacher-student interactions, and can consequently increase the behavioral engagement (Gregory et al., 2014).

### **2.4.2 Emotional engagement**

Emotional engagement includes reactions, expressions of interest, and positive effects towards academic work, and can range from only liking something to valuing it strongly (Fredricks et al., 2004). It plays a vital role in promoting students' performance and well-being, and research suggests that emotionally disengaged students often begin to disengage behaviorally and cognitively as well (Park et al., 2012). Caraway et al. (2003) found that teens who feel confident about their competences are more engaged in various aspects of school and thus more self-efficient with their tasks.

Another influence on students' emotional engagement is how they relate to tasks and persons around them. Students' inner motivational resources have been discovered to decide their level of academic engagement (Pianta et al., 2012). Park et al. (2012) have found that students are more engaged when they perceive that the context is supportive of their psychological needs for autonomy, competence, and relatedness. The perceived opportunities for relatedness are further found to be more associated with engagement for higher achieving students than for lower achieving students, and supports that a high relatedness can minimize feelings of boredom, pressure, and frustrations (Furrer and Skinner, 2003). This suggests that students who are at risk for emotional disengagement and underachievement are sensitive and responsive to opportunities that fulfill their needs to be autonomous, emotionally connected, and competent learners. In addition to this, researchers have stated that education should match the needs of the society, which includes generation of innovative ideas and unblocking of creativity (Riga and Chronopoulou, 2014). Activities which allow students to explore and be playful are also found to increase students' willingness for exposure to unfamiliar situations. Students should therefore be provided enough space and time for creative ideas to avoid discouragement of self-expression and decision-making.

### **2.4.3 Cognitive engagement**

Cognitive engagement includes the investment, strategy use, and willingness towards academic work, and it can range from memorizing something to facilitating own strategies for further insight (Fredricks et al., 2004). Students who make academic work personally meaningful are found to be cognitively engaged (Appleton et al., 2008). On the other side, teachers who make connections between academic activities and students' values obvious, can make distressed and demotivated teens more engaged by increasing their value of learning and meeting their personal goals (Caraway et al., 2003). Learning environments which are designed to enhance students' self-efficacy and task value have also been found to make teens more engaged in using cognitive and metacognitive strategies (Bircan and Sungur, 2016).

Researchers imply that students are more self-efficacious and invested when activities are perceived as useful, entertaining and important to them (Bircan and Sungur, 2016). Bandura and Cervone (1986) found that students' perceptions of self-efficacy influence the level of challenging tasks they choose, how much effort they are willing to expend, and how long time they are willing to use in order to solve them. In addition to this, research has found that students who set goals for their academic activities have a higher willingness towards achieving their goals, and are more likely of feeling less distressed in challenging situations (Bandura and Cervone, 1986; Caraway et al., 2003). Many students also feel a temporary satisfaction and an increased motivation by setting new future goals when they fulfill difficult tasks. However, Caraway et al. (2003) reported that students are found to be most motivated and involved when their standard of challenging tasks and self-belief of capabilities match.

#### **2.4.4 Agentic engagement**

Agentic engagement includes making contributions to instruction and learning activities, and it can range from asking questions to expressing own wants and needs (Reeve and Tseng, 2011). According to Reeve (2013), agentic engagement is a way for students to achieve greater achievement and motivational support. It is a proposed addition to the most common multidimensional construct of academic progress, which consists of behaviors, cognitive, and emotional features. Reeve (2013) also believes that student contributions can make the learning experience more personally valued by students, as they can contribute to the flow of instruction they receive and consequently enhance their learning. These student contributions include offering inputs, expressing preferences, communicating thoughts and needs, and recommending a goal or objective that they can pursue (Reeve and Tseng, 2011). Agentic engagement is also viewed to benefit self-efficacy, personal goals and values, individual interests, and a mastery goal orientation.

Reeve (2013) reported that autonomy-supportive teachers can further result in agentially engaging students. These students expect that teachers create motivationally enriching classroom conditions for them, such as opportunities to stimulate autonomy, competence, and relatedness in social interactions. Teachers' autonomy support can also benefit students in enhancing their motivation, learning, and well-being. The effect works both ways, as agentic engagement can contribute to changes in teachers' ways of motivating from something that exists within teachers, to something that appears during teacher-student interactions.

## **2.5 Teens' engagement in computational thinking activities**

While studies have had in goal to address challenging computational thinking concepts (Maloney et al., 2008; Denner et al., 2012; Franklin et al., 2013; Werner et al., 2014; Fields et al., 2016), other studies have looked into how youth can be more engaged in the field of Computer Science, and thus computational thinking. Maloney et al. (2008) found that activities with similarities to subjects that support kids in being creative and expressive such as arts, science, and computer class can make youth engage more in learning and creating their own artifacts. In a study by Giannakos et al. (2013), the authors investigated whether involvement with creative artifacts with Scratch, Arduinos and recycled materials at a ReMida center in Norway could increase students' engagement in Computer Science (Giannakos et al., 2013). The findings from the study reveal that students enjoy the creative parts of the workshops, and that many of them change their perspectives on Computer Science. In another study, Giannakos and Jaccheri (2013) have found that creating stories is attractive for students, and that students' control over their actions influence how they perceive the usefulness of the activities. Other studies such as Kelleher et al. (2007) and Burke and Kafai (2012) have reported that a storytelling approach can introduce youth to Computer Science concepts, in addition to attract and motivate children to learn more about the discipline. Overall, research has discovered that exciting and enjoyable tasks in programming contexts will lead to more interactive teens, and ultimately result in increased interest for Computer Science (Lykke et al., 2015).

Basawapatna et al. (2010) believe that game design and creation is an excellent way to introduce students to programming and computational thinking, and that it can enable learning among both experienced and inexperienced

students, in addition to motivate, engage, and educate students about Computer Science (Repenning et al., 2010; Werner et al., 2014). According to Basawapatna et al. (2010), creating and designing games are proven to be beneficial for peer-to-peer interaction in middle school environments, and adopting the creation of games increases class engagement and interests simultaneously for middle schoolers. Basawapatna et al. (2010) also discovered that computational thinking gains are more significant when participants create projects regarding some topic from their majors, as they can simulate informative real-life situations meaningful to themselves (Repenning et al., 2010). Çakır et al. (2017) found that female students can meaningfully engage with game-design tasks when they use programming resources in creative ways. Adams and Webster (2012) reported that project-based learning can force learners to move from the first level of knowing something, through the second level of comprehension, and lastly the third level where the students are able to apply the knowledge to new situations, hence showing a deeper level of understanding and increased engagement.

Collaborative learning is found to lower attrition and improve performance and critical thinking for Computer Science learners. In a game-based learning activity, research showed that students with less gaming experience seem to be at a disadvantage without collaboration (Buffum et al., 2016). When novice learners have experienced partners, they can learn from their partners' experiences. On the other side, having partners with more experience can also lead to decrements of participation for novice learners if their partners become dominant. Researchers have also disclosed that feedback can make teens open to new ideas and viewpoints in computational thinking activities (An, 2016). Teens appreciate the comments they receive and fix their programming projects based on the feedback. This way they become more active learners, which seems to affect their engagement. In contrast, students who receive critique from their peers are more prone to being disengaged.

Lykke et al. (2015) stated that teens with joyful and lively interactions with their teachers participate more in activities by asking questions and commenting. The findings revealed that a guiding role for teachers is essential and that teens consequently show more attention, involvement, and interest in the activities. Research has also found that proper teacher education in large extent and in-depth is one of the most vital factors for the success of Computer Science education (Hubwieser et al., 2015). Additionally, it has been reported to be one of the hardest goals to achieve for educators.

A study by Grover et al. (2014b) revealed that misperceptions of what computer scientists do affect students' interest for Computer Science, and after attending a mini-course in Computer Science, youth express more positive adjectives about the field, increased insight of the uses of it, and in general more positive attitudes towards learning more. Exposure to activities that show that there is more to the Computer Science discipline than building, fixing, and studying computers can therefore be helpful for teens' engagement.

## **2.6 Overview of contributions from related theories**

Tables 2.1, 2.2, 2.3, and 2.4 summarize the most relevant research and theories from the conducted literature review. Although many studies have been mentioned, only some are relevant for the rest of this study. These studies are referred to in Chapter 5.

## 2.6 Overview of contributions from related theories

---

Study	Concept	Contributions to this study
Mannila et al. (2014)	Constructionist learning theory, visual programming	Four principles of constructionism, visual programming useful for learning computational thinking
Papert (1980)	Constructionist learning theory, computational thinking, visual programming	Projections of inner feelings and ideas are vital, tools are used in the learning process to support exploration of meaningful things, programming to support cognitive thinking
Grover and Pea (2013)	Computational thinking	Computational thinking can develop unique thinking skills, enhance mathematical, engineering, and design thinking, basic knowledge in computational thinking is as relevant and critical as basic knowledge in science and math
Riga and Chronopoulou (2014)	Computational thinking and creativity	Only creative environments can make creative thinking flourish, improvisation, testing, problem-solving, and alternative uses influence children's creativity and behavior, effective group dynamics result in confidence and enjoyment
Resnick et al. (2009)	Computational thinking, creativity, programming	Computing is used for design and self-expression, programming is a tool to support the cognitive tasks in computational thinking, Scratch

**Table 2.1:** Overview of similar research and their contributions to this study

---



<b>Study</b>	<b>Concept</b>	<b>Contributions to this study</b>
Resnick and Silverman (2005)	Computational thinking, programming	Programming can extend what children are capable of creating
Kafai and Burke (2013)	Computational participation	Reasons for learning programming are changing, there is a need for making computational thinking and programming social acts
Fields et al. (2014)	Computational participation	There is a need for investigating quality of engagement to broaden participation
Weintrop and Wilensky (2015)	Visual programming	A list of popular visual programming environments that fulfills requirements for programming languages for children
Kelleher et al. (2007)	Visual programming	No differences in performance with textual-based and visual programming, visual programming to learn computational thinking concepts, self-exploration
Grover et al. (2014b)	Academic engagement	Disinterest is influenced by misperceptions
Fredricks et al. (2004)	Academic engagement	Academic engagement to increase motivation and achievement, definitions for behavioral, emotional, and cognitive engagement

**Table 2.2:** Overview of similar research and their contributions to this study  
(continued)

---

## 2.6 Overview of contributions from related theories

---

Study	Concept	Contributions to this study
Turner et al. (2014)	Academic engagement	Students are engaged when academic learning is valued and meaningful, motivational factors
Reeve and Tseng (2011)	Academic engagement, engagement	Definition of engagement types, personalization of learning experiences to make academic achievements become more personal
Reeve (2013)	Academic engagement, engagement	Definition of academic engagement, teacher-student interactions are essential, teachers should provide students motivational conditions
Deci et al. (1981)	Autonomy	Autonomy as a motivational factor
Boggiano et al. (1992)	Competence	Student perceptions of own competence affect their achievements
Wentzel and Watkins (2002)	Academic engagement	Collaborations in activities affect academic engagement
Gregory et al. (2014)	Academic engagement, behavioral engagement	Behavioral disengagement is a risk, teachers affect behavioral engagement, feedback from teachers is essential
Juvonen et al. (2012)	Academic engagement	Collaborations are important for engagement, performance, and involvement

**Table 2.3:** Overview of similar research and their contributions to this study (continued)

---

<b>Study</b>	<b>Concept</b>	<b>Contributions to this study</b>
Furrer and Skinner (2003)	Academic engagement	Interactions with peers and teachers affect academic engagement
Skinner and Belmont (1993)	Academic engagement	Interactions between teacher and student affect academic engagement
Pianta et al. (2012)	Academic engagement	Teachers can influence students' academic engagement
Park et al. (2012)	Academic engagement, emotional engagement	Emotionally disengaged students often begin to disengage behaviorally and cognitively, emotional engagement is increased when psychological needs are met
Caraway et al. (2003)	Academic engagement, emotional engagement	Competence confidence increases emotional engagement, higher willingness to achieve goals when they are set, balances between challenging tasks and perceived competence, fear of failure can lead to worse performances
Bandura and Cervone (1986)	Academic engagement, cognitive engagement	Useful, entertaining, and important tasks increase cognitive engagement, students are less likely to feel distressed in challenging situations when they set goals, competences influence academic engagement

**Table 2.4:** Overview of similar research and their contributions to this study  
(continued)

---

## **2.7 Overview of contributions from related studies**

The following tables 2.5, 2.6, and 2.7 summarize related studies from the conducted literature review. These tables include the context, research goals, and findings that are relevant to this study. Additionally, Table 2.8 has an overview of how several studies have addressed challenging computational concepts for youth. These studies are relevant for Chapter 5.

<b>Study</b>	<b>Context</b>	<b>Research goal</b>	<b>Contributions to this study</b>
Giannakos et al. (2013)	Out of school workshop	Explore the relationship between engagement and creativity	Positive outcomes of programming with creativity, learning is a social process
Giannakos and Jaccheri (2013)	Out of school program	Provide validated knowledge between collaboration and control throughout creative programming activities	Creativity facilitates learning and engagement, collaboration improves the value of activities, children's control over their actions influences willingness and usefulness of activities
Kelleher et al. (2007)	After school program	Detect motivating reasons to learn programming	Self-expression and self-exploration are found motivating, benefits of visual programming, survey can be used to collect participants' experiences
Burke and Kafai (2012)	In school	Explore the use of basic programming concepts in a writing workshop context	Collaboration and personal expression are found to be successful for educational projects, use of storyboards to help participants design their games, storytelling is interesting to teens

**Table 2.5:** Overview of similar studies and their contributions to this study

## 2.7 Overview of contributions from related studies

Study	Context	Research goal	Contributions to this study
Lykke et al. (2015)	University	Compare the differences of motivation with three learning designs	Interesting and enjoyable tasks increase motivation, the interactions between teacher and student are important, students who feel incompetent fear for failure
Basawapatna et al. (2010)	In school	Investigate the use of game creation in Computer Science education	Games enable the emerging benefit of peer-to-peer interaction and peer learning in middle school, game creation increases engagement, personal values paired with games
Repenning et al. (2010)	In school	Game design to motivate, engage, and educate students about Computer Science	Game design to introduce Computer Science to teens, stimulation of real-life situations is engaging
An (2016)	In school	Investigate how students design educational computer games and its impact on learning	Creation of games increases class engagement, feedback from peers for new ideas and viewpoints

**Table 2.6:** Overview of similar studies and their contributions to this study (continued)

Study	Context	Research goal	Contributions to this study
Çakır et al. (2017)	Out of school workshop	Explore the impact game-design workshop have on girls' attitudes towards computing	Creative uses of resources are meaningfully engaging, game creation as a fun computational thinking activity
Adams and Webster (2012)	Outreach program	Identify computational thinking differences in learning in programming projects	Different learning and motivational outcomes with different genres of programming projects, deeper levels of understanding can be engaging
Buffum et al. (2016)	In school	Investigate the nature of collaboration in game-based learning environment	Collaborations improve performance and critical thinking, novices learn from experienced partners with game design
Hubwieser et al. (2015)	N/A	Extract useful information of Computer Science education	Teacher education influences Computer Science education
Grover et al. (2014b)	In school	Address misperceptions of Computer Science among students	"Correct" perceptions make students more interested and aware of the possibilities within the discipline

**Table 2.7:** Overview of similar studies and their contributions to this study (continued)

---

## 2.7 Overview of contributions from related studies

Study	Context	Research goal	Contributions to this study
Fields et al. (2016)	Scratch summer camp	Awareness of novices' pathways of learning	Way to measure <i>demonstrated</i> computational thinking skills
Werner et al. (2014)	Game programming class	Measure students' understanding of computational thinking concepts	How computational thinking looks like in middle school, game-like approaches are engaging
Denner et al. (2012)	After-school program	Identify how children can be engaged in computational thinking	Usefulness of pairs, programming promotes thinking and problem solving, novice users avoid creating complex functionality
Franklin et al. (2013)	Scratch summer camp	Assessment of learning	The frequencies and types of help during programming projects to reflect <i>demonstrated</i> competence, Hairball to analyze Scratch projects
Maloney et al. (2008)	After school center	Motivations and implications of after school programming	Creative and personal expression with visual programming

**Table 2.8:** Overview of studies that address challenging computational thinking concepts for youth



# Chapter 3

## Methodology

The last chapter looked at theories and findings from related studies. Chapter 3 will provide elements of the methodology used to explore the objective of this study. Two workshops based on constructionism called Kodeløypa and Tappetina were conducted to answer the research inquiry. This chapter proceeds as follows: Section 3.1 describes the research approach chosen for this study, which is a mixed methods approach. Section 3.2 explains the author's position and preconceptions, while Section 3.3 clarifies the settings for where data was collected. Section 3.4 gives an overview of this study's sample, and Section 3.5 explains the data collection procedures of this study. Since there were only a few differences in analyzing the data, Section 3.6 describes how data was analyzed for both workshops if the opposite is not stated. Finally, the steps taken to ensure the validity and reliability of this study are presented. Lastly, an ethical statement is clarified. The procedures in this study were conducted in collaboration with a Ph.D. student and by the author herself. The former will be referred to as "we", while the latter as "I".

## 3.1 Research approach

According to Johnson et al. (2007), qualitative research is usually inductive, discovering, and exploratory, and in this theory and hypothesis generating approach, the researcher is the main "apparatus" for the data collection. On the other side, quantitative research tends to focus on deduction, confirmation, testing of theories and hypotheses, prediction, standardized data collection, and statistical analyses. Both qualitative and quantitative research have been found to be suitable for this study.

The objective of this study is to explore factors that influence the academic engagement teens have in computational thinking activities with programming. While a qualitative method such as interview has the opportunity to catch insight into teens' beliefs, experiences, and emotions for such coding activities, a quantitative method such as questionnaire has the ability to provide statistics with scoring systems such as Likert (Allen and Seaman, 2007). To use different techniques, methods, and approaches popular for either qualitative or quantitative research, this research study followed a mixed methods research approach. This way, I have been able to draw strengths and minimize the weaknesses known for either of the research methods to answer the research question (Johnson et al., 2007; Creswell, 2014). Mixed research can provide stronger evidence for a conclusion through convergence and corroboration of findings, and adds insight and understanding into situations where that might be missed when only a single method is used. While this approach is more time consuming, expensive, and hard for a single researcher to conduct alone, an effective combination of the two methods provides more complete knowledge to inform theory and practice (Johnson et al., 2007).

According to Hanson et al. (2005), designing a mixed methods study involves a number of steps, many which are similar to those taken in tradi-

tional research methods. The first step is to decide a theoretical lens that refers to a philosophical basis, or a paradigm for the study and the methodological choices. The educational strategy of computer game programming with children and much research on how and what children learn when they work on computers are based on the constructionist paradigm (Denner et al., 2012). This study therefore supports the constructionism learning paradigm and acknowledges that when people are actively engaged in making meaningful products through designing, exploring, discussing, and reflecting for and with other people, learning is facilitated. This study has thus provided students with greater opportunities to construct own artifacts and allowed the participants to construct new relationships with knowledge in the process, rather than embedding "lessons" directly in the learning process (Kafai, 2006). The second step in mixed methods design is to decide how data collection will be implemented and prioritized. Both workshops' data collection was weighted equally, but the order of the research instruments was adapted to fit the nature of breaks and program in each workshop. The reasoning for why the data collection was weighted equally was merely based on the uncertainty of data we would collect during the workshops, and how important the different kinds would be to the research inquiry. The third step in designing a mixed methods study is to decide at what point data analysis and integration will occur. For this study, qualitative and quantitative data were first analyzed separately for each workshop. This was done because we wanted to avoid that the results of the qualitative data collection got affected by the quantitative data, and vice versa when analyzing. Secondly, all the results from Kodeløypa were compared to detect relations between them. This step was also done for the results from Tappetina. After this, the results from both workshops were examined together to draw comparisons and contrasts between them.

## **3.2 My position and preconceptions**

As mentioned earlier, I have always been interested in assisting kids in improving themselves, regardless if they needed help on the basketball court or with mathematics. Four summers in a row, I was paired with different teachers with the aim of teaching math to first and second graders. My personal experiences with teaching have given me some knowledge on how to approach learning with younger students. In addition to this, I also consider myself interested in learning technology and Computer Science. However, since my main goal of this study has been to research, I have purposely avoided interactions with the participants more than described in this chapter. I have only met the participants during their short stay at either workshops, and I did not assist them in any way.

## **3.3 Data collection**

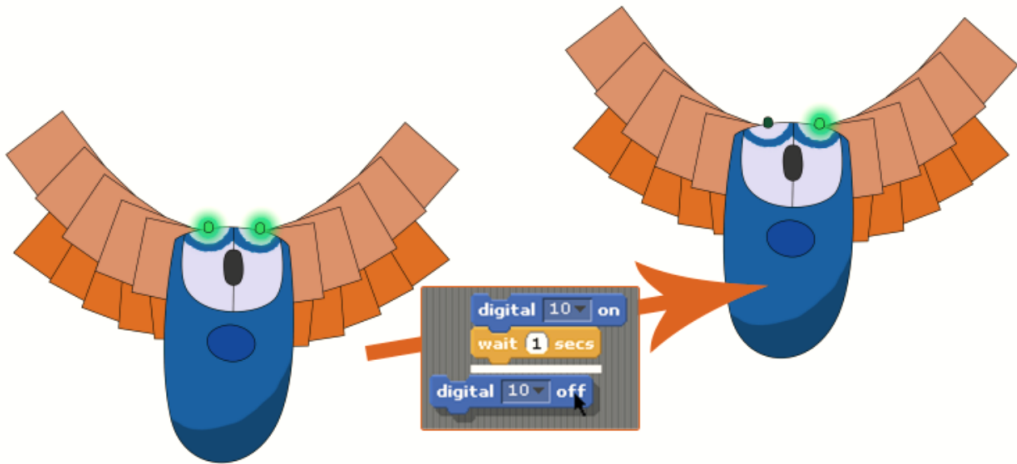
### **3.3.1 Kodeløypa outline**

In six different workshops offered by The Norwegian University of Science and Technology (NTNU) each year, primary and secondary school students are introduced to various science disciplines from Physics, Chemistry, Mathematics, Biology, Energy, and Computer Science, with the goal of boosting the kids' interest for science (Papavlasopoulou et al., 2015). The workshop for Computer Science, Kodeløypa, is built on the ideas that interactions between young students and artifacts in creative activities are vital, in addition to learning by making (Papavlasopoulou et al., 2017). Kodeløypa is an out of school activity conducted informally, and the students attend the workshop for five hours in total with their respective school classes. The workshop is designed such that students without programming experience

can participate, and at each workshop, two to three student assistants are available to assist the students. Programming concepts are not introduced to the students before they start programming, but rather introduced by assistants when and if needed.

Before the workshop, teachers are sent consent letters so that students can collect signatures from their parents. This letter contains information about the study and a notice that data will be collected during the workshop. If a parent signs this letter, they confirm that researchers can collect data regarding their child. In this letter, we inform the parents that participation in the study is voluntary, that all results will be confidential, and withdrawal from the participation will not affect their kids' grades in school. The consent letter is found in Appendix A.

In the first part of the Kodeløypa workshop, participants interact with digital robots in small teams. By using the extension Scratch 4 Arduino, the teams can control the robots using code blocks in Scratch. Each team is placed next to a computer connected to a robot with physical and aesthetical characteristics, and the student assistants provide them a paper tutorial and worksheet each to answer questions regarding the robots. The paper tutorial contains examples which describe how participants can interact with the robots. A snippet from the paper tutorial can be found below in Figure 3.1, while the belonging robot can be seen in Figure 3.2. In this robot part of the workshop, the participants are supposed to create several loops to make the robots move, make the lights on the robots turn on and off, and make the robots react to different factors, for instance turning a light on when a sensor detects that it is below a certain threshold. The first part usually lasts around 45 to 90 minutes before the students have a break for about 15 minutes.

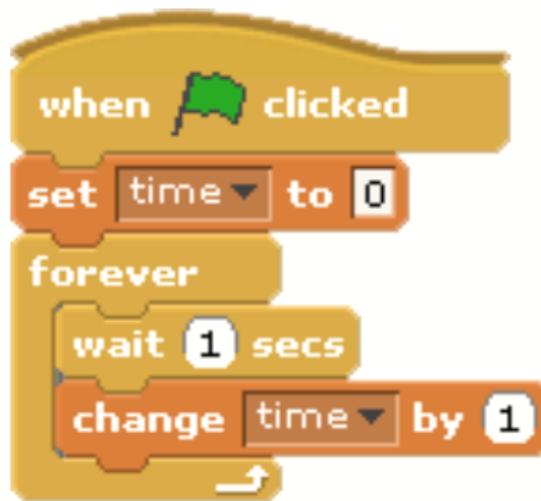


**Figure 3.1:** An illustration of a robot from the Kodeløypa paper tutorial

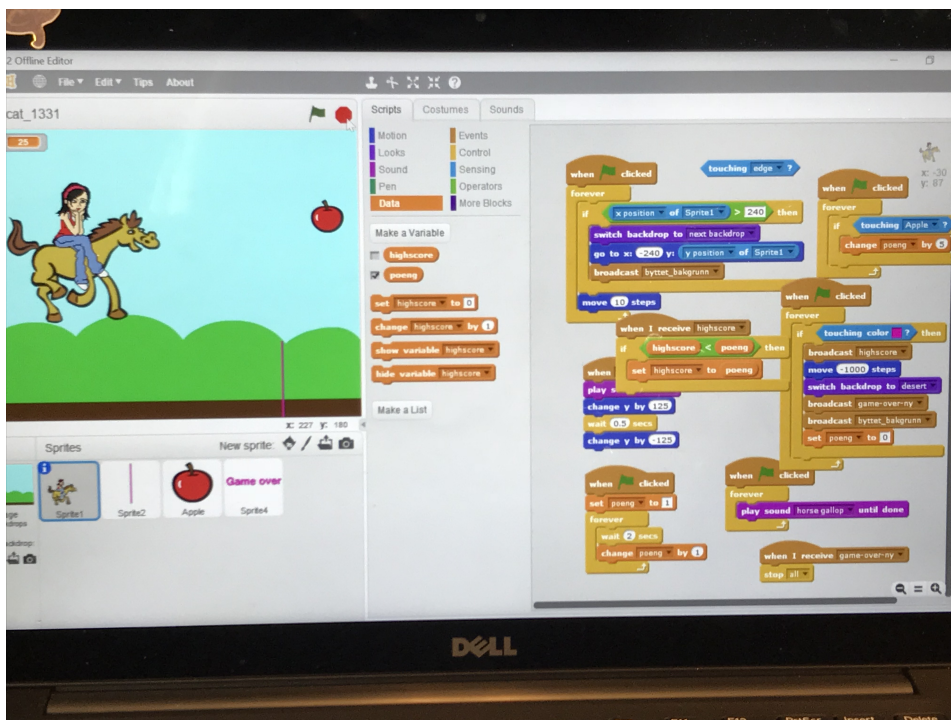


**Figure 3.2:** An example robot from the Kodeløypa workshop

In the second part of the workshop, the same teams are supposed to create their own game in Scratch. If the student assistants think that the game is achievable in terms of time and complexity, the teams are free to create any game they want. The students are given a new paper tutorial with examples on what they can add into their game, including how they can make characters move, react to touching something, jump, and adding points and time functionality. A programming example from this paper tutorial is available in Figure 3.3. In the game creation part, the teams are first told to concentrate on the idea of the game, reach a consensus, and create a draft storyboard before they start to code. When a student assistant approves a team's game idea, the team can start to program their game. Also here, the student assistants are available to help the teams. The students create and test their games iteratively during the process of developing their games, and in the end, the teens can play games made by their classmates. The second part of the workshop usually lasts for three hours, excluding a long lunch break. A script that was created during Kodeløypa can be found in Figure 3.4.



**Figure 3.3:** An example of a Scratch script in the paper tutorial



**Figure 3.4:** An example of a game created during the Kodeløypa workshop

### 3.3.2 Tappetina outline

In a two-day workshop at a library in Trondheim, Norway, girls between the age of 12 and 15 were invited to code and be creative during Fall break. The workshop was in collaboration with the library, and employees from NTNU who were involved in Kodeløypa. Also prior to this workshop, consent forms were sent out to the participants' parents. The workshop was conducted for approximately 9.5 hours including lunch, and the focus of this workshop was to introduce girls to the field of Computer Science, with a character called Tappetina from the book *Little Doormaid*. Tappetina is a rather ordinary woman with children and everyday problems, but she is also Doory Mentor, a superhero who helps people to succeed with their



technology projects (Jaccheri, 2017). Tappetina spends time being self-conscious, while the superhero part of her is strong and clear-headed, as well as ambitious and visionary. Jaccheri (2017) plays with language to create a playful and visual experience of people and events, with characters who do powerful things. The story appeals especially to girls who are known to be a minority in the field of Computer Science compared to boys (Kelleher et al., 2007; Repenning et al., 2010; Grover and Pea, 2013; Giannakos et al., 2013; Papavlasopoulou, 2016; Thanapornsanguth and Holbert, 2017). The girls would during their stay create a storyboard on cardboards, and a game related to Tappetina saving the world from an environmental issue. In this workshop, two student assistants were available for assistance if needed.

The first day consisted of the girls getting familiarized with Scratch and the basics of it. The girls could bring own laptops from home, but they were also able to use available laptops provided by NTNU or the library. In the beginning, the girls did exercises in Scratch individually, and around midday, Jaccheri and a student at NTNU presented the story of Tappetina to the participants. After lunch, the student assistants divided the girls into three teams and presented the concepts of storyboards, and why creating one was important. The student assistants also demonstrated an example game in Scratch to show the girls what could be done and started a discussion about environmental issues. After this, the girls were told to create a storyboard for the game they wanted to create. One of the creative aspects of the Tappetina workshop was to create the storyboard in teams: The girls could choose between various materials for instance scissors, pens, pencils, colors, and cardboards to design them. Figure 3.5 shows how the girls were working on their cardboards.



**Figure 3.5:** Tappetina participants working on their storyboard

On the second day, the teams continued to work on their storyboards. After this, each team presented their product and game concepts in front of fellow participants and student assistants. The remaining time of the workshop was used to create the games in Scratch, and at the end of the day, the girls held a presentation about their games and played the other teams' games.

## 3.4 Sampling

### 3.4.1 Kodeløypa sampling

From September to November 2017, Kodeløypa was conducted on Fridays for seven weeks in total. Six classes from Trondheim and around the area signed up to be participants in the workshop, and three of these classes were technology classes that the students chose as electives. Depending on the number of students that arrived at the workshop, and the number of signed consent forms, the assistants divided the students into small teams of dyads or triads. Each team was given a team animal in the form of a sticker to separate them from the others. We had in total a number of 106 participants in Kodeløypa, and 69 of these were boys, while 37 were girls. In Kodeløypa, we had in total 14 teams of dyads and 26 teams of triads. Two of the visiting schools were small, hence the reason why they sent students from three different grades. An overview of Kodeløypa demographics can be seen in Table 3.1 and Table 3.2.

Participation date	Elective	No. of Students	Dyad teams	Triad teams
29.09.2017	Yes	18	3	4
06.10.2017	No	14	1	4
20.10.2017	Yes	19	2	5
27.10.2017	No	21	0	7
03.11.2017	No	21	3	5
10.11.2017	Yes	13	5	1

**Table 3.1:** Kodeløypa statistics which include participation dates, class types, number of students, and team divisions

Participation date	Girls	Boys	8th graders	9th graders	10th graders
29.09.2017	2	16	0	18	0
06.10.2017	8	6	0	0	14
20.10.2017	7	12	0	3	16
27.10.2017	7	14	0	0	21
03.11.2017	10	11	0	13	8
10.11.2017	3	10	2	3	8

**Table 3.2:** Kodeløypa statistics which include participation dates, gender ratios, and class grades

### 3.4.2 Tappetina sampling

Eight girls attended the Tappetina workshop in October 2017. Of the participating girls, two of the girls already knew each other because they were sisters, but the others did not know each other beforehand. The ages ranged from 9 to 14, and the different grades that the girls belonged to can be seen in Table 3.3. We had in total three teams: Two of these were triads, and the last one a dyad. Also here, each team was given an animal to distinct one team from the others.

Grade	No. of participants
5th	1
7th	2
8th	3
9th	1
10th	1

**Table 3.3:** Tappetina statistics which include the class grades of the participants

## **3.5 Data collection procedure**

For the sake of collecting data from the workshops, we picked out various instruments that were suitable to answer the research inquiry. These were observation, interview, artifact, and lastly questionnaire. Why we chose these and how we used them will be explained further in the next subsections. If not stated explicitly, the process of using these instruments for data collection was used in the same way for both workshops. The reason why we conducted them the same way was to preserve consistency of the collected data.

### **3.5.1 Observation**

According to Oates (2006), observation is a data generation method to find out what people actually do, rather than what they report that they do when questioned. In our case, we decided to implement a systematic observation approach, meaning that we could collect a volume of data fairly easily. With observations, we were able to study overt behavior, but at the same time, this could affect behaviors by the participants if they reacted to being observed. However, by conducting observations, we collected detailed, highly descriptive field notes that could be compared with data collected with other instruments (Mays and Pope, 1995). By picking up behavior that the participants might be aware or unaware of, we could as researchers get a more accurate reflection of reality, which is considered more objective than self-report of behavior (Sandelowski, 2000).

While observing, the aim was to write notes of group dynamics and significant incidents such as one team member taking over and keeping the control of the computer for a longer duration of time, or if team members participated in the collaborative aspects of creating games.

Three researchers had the task of observing teams with consent letters, and if participants asked them any questions, they were told to address the student assistants. To secure informative field notes, we decided before the workshops that the researchers would observe maximum 12 participants with consent forms at each workshop.

While this study most importantly aims to look at factors that influence the engagement among teens in computational thinking activities, observations were also used to catch up signs of computational thinking skills or performance. This information was able to give us further knowledge about the specific sample. It is however important to declare that we did not get a clear picture of what participants actually knew and understood without further testing, but we got an *indication* of their conducted performance. During the observations, the researchers tracked the number of times and types of help the different teams asked for during the game creation phases. The utilized help levels were presented by Franklin et al. (2013), and guided the researchers in identifying indications of performance: If a question was related to a concept taught before, either in the first or second part of the workshop, the researcher would write this down. For the rest of this thesis, participants' performance and competence in computational thinking will therefore imply the indicated performance. The help levels can be found in Table 3.4.

While the focus of observation was set during the game creation phase in Kodeløypa, observations started from when the participants began to create their storyboards to the games were finished in Tappetina. We did this to investigate whether the collaborative learning was dependent on the type of activities the teams were doing. In Kodeløypa, we observed in total 47 students during the six workshops, while we observed eight out of eight participants in the Tappetina workshop.

#	Explanation
0	<b>Validation:</b> Students want confirmation, not information
1	<b>Where:</b> Only needed help navigating the Scratch gui
2	<b>What:</b> Only needed a reminder of the name of the concept
3	<b>How:</b> Given name of concept, still needed help to complete task
4	<b>Reteach:</b> Had to reteach the entire lesson (concept and execution)

**Table 3.4:** Table used to indicate the levels of help the teens received during the workshops

### 3.5.2 Interview

By adding another data collection instrument such as interview, possible biases from observations can be confirmed or eliminated. Systematic observations assume that overt behavior can be broken down into easily observable and categorizable phenomena, and thus oversimplify a situation (Oates, 2006). According to Polkinghorne (2005), the purpose of interviews is to gain a full and detailed account from an informant who has experienced the study. To gain insight into the academic engagement aspect of the research inquiry, we implemented a semi-structured interview style, to allow both flow and room for unprepared issues or topics in the conversation. While this interview style makes it hard to generalize to a larger group in the society, it allows collection and exploration of personal accounts and feelings from the interviewees (Oates, 2006). This way, we could collect different perspectives on the study experience.

Because memories are prone to bias and error, interviews are recommended to be paired with field notes or to be recorded with either video or audio. In this case, we used audio recordings to have a complete record of everything that was said during the interviews. Each interview started with an introduction to the purpose of the study, as well as relevant information

such as the possibility of withdrawing from the study without any questions asked. We made sure that the interviews did not exceed more than ten minutes, and that we conducted them at appropriate times during the game creation phase. To avoid interfering more than necessary during the game creation phase, the researchers had a goal to interview at least one member per observed team.

Due to the differences in workshop programs, the interview questions for the workshops were not identical. However, in both workshops, we wanted to investigate how the participants felt about their game creation experience, what they found easy or hard, frustrations and impressions, and lastly opinions on the collaborative learning. Since the Tappetina workshop was set for two days and had a clear division of planning and execution of games, we included additional interview questions about their challenges during the sketching and game creation part. These questions would be able to catch whether participants could detect differences between planning a game on paper with storyboards, and creating it with Scratch. The Tappetina workshop also focused on environmental issues, so we wanted the girls to answer a question about how they felt about them. The guidelines and questions from Kodeløypa can be found in Appendix B, while the ones from Tappetina can be found in Appendix C.

Since the participants belonged to Norwegian schools, we conducted the interviews in Norwegian, except for one Friday when an international school visited Kodeløypa. After each workshop, the audio recordings were transcribed into English to fit the language of this thesis. By transcribing, the contents of the data material became more familiar, and we provided the interviewees more privacy. In addition to this, written accounts can also be considered easier to work with (Lorås, 2017). To avoid subjective interpretations, there was a focus on only writing down the participants said when transcribing. At the end of each workshop, each interviewee had a corresponding document with their transcribed interview.



A total number of 44 interviews were conducted in Kodeløypa, while we interviewed seven out of eight participants in the Tappetina workshop. The reasoning for the latter was the fact that one of the sisters was younger than the others participants, and felt uncomfortable and doubtful without her sister. With that in mind, the researchers felt that best option would be not to interview her although her parents signed her consent form.

### 3.5.3 Artifact

The studies in Table 2.8 are a few of many studies that have investigated and analyzed contents of Scratch projects to detect computational thinking competence. *Artifacts* have therefore been considered sufficient for assessing computational thinking concepts and practice, but like Franklin et al. (2013) stated, it is important to acknowledge that we can only get a clue of what was done, instead of what was actually understood or learned by the participants without further testing. In this study, the researchers saved different versions of games as artifacts to provide a better picture of the sample, and to see the levels of computational thinking the participants were able to reach in the workshops.

We implemented a summative approach for the game versions, and contents in these artifacts were identified and quantified to investigate the usages of different computational thinking concepts. Versions of the games were saved throughout the workshops, leaving us with a relatively large volume of artifacts in total. In other studies, this has been done to make the assessment even richer (Resnick et al., 2009; Brennan and Resnick, 2012).

In Kodeløypa, we saved in total 139 game versions, while in Tappetina, we had 13 game versions. After removing duplicate versions and empty Scratch projects, we had 130 game versions from Kodeløypa, and 12 from Tappetina, giving us a total of 142 game versions.

### 3.5.4 Questionnaire

Questionnaires are easy measures to gather surface information, and can be used to gather information from a representative sample of a defined population, and to generalize to a wider population (Rattray and Jones, 2007). They can be used to explore self-reports of knowledge, attitudes, emotions, cognition, intention, or behavior. To easily and efficiently collect data regarding demographics, previous knowledge and experiences with programming, and participant opinions on various statements, questionnaires were used in this study.

The two workshops had different questionnaires, as the workshop programs and activities differed. However, both of the questionnaires had an identical part where participants stated their age, gender, their belonging class grade, how many activities with programming they had participated in before the workshop, and former experiences with programming. For the second questionnaire part, we implemented a Likert scale for each possible statement, which are ordinal scales to measure agreement and disagreement (Allen and Seaman, 2007). In this study, seven points were used to reach the upper limits of the scale's reliability, where 1 = strongly disagree, and 7 = strongly agree.

In Kodeløypa, we gave the participants questionnaires after the workshop. We did this to avoid interfering with the activity the participants were involved in. This questionnaire emphasized on participants' perceived collaborative learning during the workshop, and Kodeløypa produced in total 104 questionnaires. In the Tappetina workshop, we handed out questionnaires before the game creation phase. By doing so, we could collect what the participants felt before the game creation, and possibly see how their minds changed after creating games by interpreting the interviews. This questionnaire focused on several aspects, including participant intentions with attending programming activities and their thoughts about program-

ming. For all observed participants, the researchers would write down the animal and member indication. This way, we could connect data regarding the same participant from different data collection instruments (Oates, 2006).

## **3.6 Data analysis procedure**

### **3.6.1 Observation**

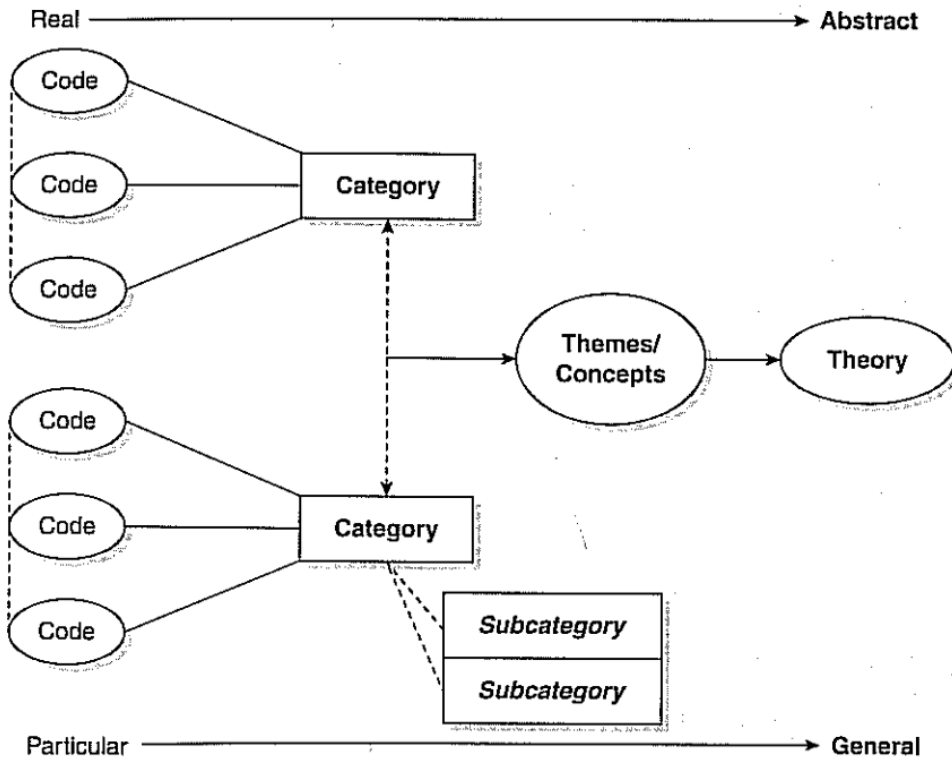
As mentioned earlier, the results of observations were field notes regarding for instance behaviors, assistance provided to the teams, and the group dynamics. The first thing we did in order to analyze the field notes was to create a spreadsheet for Kodeløypa and another one for Tappetina. This way, we could keep the field notes that belonged to one workshop in the corresponding context. In these spreadsheets, we tracked the frequencies of the help levels proposed by Franklin et al. (2013) for each team. The researchers also used field notes that said anything about the performance level of computational thinking in the analysis. This data provided us more knowledge about what the teens needed help with in general. In addition to this, we documented other important statements that supported or contradicted the research inquiry. When we finished the data analyses of field notes, we combined the results from Tappetina with the results from Kodeløypa. By doing these steps, we gained more insight into factors teens found engaging, and an idea of how the teams felt towards the computational thinking activities.

### 3.6.2 Interview

Interviews produce qualitative data, and qualitative data analysis requires interpretations by researchers. The technique we chose was thematic analysis, which involves encoding and "explicit codes" (Boyatzis, 1998; Attride-Stirling, 2001). These codes are meant to capture and represent the reviewed data's primary content and essence, and a code is described by Saldaña (2015) as follows:

*"(A code is) a word or short phrase which symbolically assigns a summative, salient, essence-capturing, and/or evocative attribute for a portion of language-based or visual data."*

After transcribing the interviews, the transcriptions were reviewed multiple times to look for common themes (Creswell et al., 2007). The coding process followed the streamlined codes-to-theory model by Saldaña (2015) as illustrated in Figure 3.6. The process of analyzing interviews was first done separately for Kodeløypa and Tappetina to keep the codes in the workshop context they belonged to, and to isolate the results from the workshops before drawing comparisons and contrasts.



**Figure 3.6:** The streamlined codes-to-theory model by Saldaña (2015) which was used in the coding process

The first cycle of coding was to read transcriptions and generate codes through descriptive, InVivo, and initial coding iteratively (Saldaña, 2015). We picked a manual approach, and the codes were written down on individual post-it notes. This step was conducted by the author and the Ph.D. student individually, securing that interpretations were not affected by the other researcher, and that we felt close familiarity and ownership to the data. After repeating this step until no more codes could be produced and we reached a certain satisfaction, the result was a reduction of a large volume of data. The codes were then discussed between the author and the Ph.D. student so that we could agree with the proposed codes. The succeeding steps were also conducted with the Ph.D. student for the same reasons,

as recommended by Attride-Stirling (2001). The next cycle was to create categories of codes, which means to cluster together codes according to similarity and regularity (Saldaña, 2015). All the existing codes were placed on several large tables and investigated in order to be grouped together. Categories were then created to fit the clusters, and continuously examined in case they were not satisfactory enough. Codes placed in categories were also continuously reviewed and moved into more suitable categories during this cycle. If codes in a category were many and distinguishable, we created subcategories. When we reached a certain satisfaction, had placed all the codes, and we could not produce or change any categories, the next step was to create coding themes. This step was somehow similar to creating categories out of clustering codes and conducted by the author herself. In this step, I looked for repetitive patterns of actions and consistencies in the documented data, so categories from both workshops were used to create themes. Here the categories and subcategories were reconsidered repeatedly and moved into themes until the placements were satisfactory. The very last step in analyzing the interviews was to see how the produced themes could relate to the research inquiry. The themes were therefore viewed in the light of the concepts and research presented in earlier chapters.

The following table presents the utilized focus points to link coding themes to individual engagement types presented in Section 2.4.

<b>Engagement type</b>	<b>Focus points</b>
<b>Behavioral</b>	Participation, involvement, effort, to try hard, to work hard, attention
<b>Emotional</b>	Reactions, expressions of interest, positive effects, enjoy learning new things, curious, fun
<b>Cognitive</b>	Investment, strategy use, reflection, own experiences and examples, connections
<b>Agentic</b>	Making contributions to instruction and learning activities, ask questions, preferences and opinions

**Table 3.5:** Overview of the focus points used to link themes to engagement types

### 3.6.3 Artifact

For the artifacts, I used a variant of content analysis, which is often used to analyze text data in health studies (Hsieh and Shannon, 2005). Content analysis can further be used to count occurrences in text and to interpret meaning from the content of textual data. Generally, it will provide basic insights about how words are used, and will often need to be paired with other methods to explore more in-depth meanings of the data.

Because of the high number of game versions, I had to choose a way to score them. After investigating how other studies had assessed learning outcomes (Repenning et al., 2010; Meerbaum-Salant et al., 2010; Brennan and Resnick, 2012; Adams and Webster, 2012; Burke and Kafai, 2012; Denner et al., 2012; Grover et al., 2014a; Armoni et al., 2015; Moreno-León et al., 2015, 2017), an approach used for assessing computational thinking in other programming projects was preferable, leaving us with the framework by Brennan and Resnick (2012), and the two automatic analy-

sis tools Hairball and Dr.Scratch. Hairball is a Scratch static analysis tool developed to determine scores of Scratch projects, concerning flaws and Computer Science concepts (Franklin et al., 2013). The tool is based on Python scripts and runs from the command-line, thus little user-friendly to use and present data. Dr.Scratch is a web-based Scratch analysis tool and is found to be positively correlated to expert evaluations (Moreno-León et al., 2017). It is intuitive and recommended to support the assessment of various computational thinking skills (Moreno-León et al., 2015). The volume of game versions made it more beneficial and less time consuming to choose a tool that was reasonably easy to use, but also provided the author with sufficient ratings of numerous computational thinking concepts. Thus, Dr.Scratch was chosen to check if the artifacts were programmed correctly, if they contained mistakes, and to score the projects. When analyzing a project in Dr.Scratch, the tool rated the project with points ranging from 0 to 21. A low score would consequently assume that the creator was a novice, and accordingly show basic information about the most critical improvements to the code. If the score was high, the tool returned more information about the project in a feedback report about computational skills and programming habits. How Dr.Scratch points Scratch projects can be found in Appendix D.

All 130 game versions from Kodeløypa and 12 game versions from Tapetina were uploaded manually to Dr.Scratch, and when the tool was able to analyze a Scratch project, a performance evaluation of the computational thinking concepts abstraction, problem decomposition, logical thinking, synchronization, parallelism, algorithmic notions of flow control, user interactivity, and data representation was returned. Each artifact had scores for each subcategory of performance, which were written down in a spreadsheet dedicated for artifacts results. When uploading the game versions to Dr.Scratch, 50 of them failed to be analyzed. What I did then was to look at the types of errors Dr.Scratch returned. I ultimately had two instances



of errors, namely `KeyError` and a `Dr.Scratch` error saying that the project could not be analyzed without any further feedback. 31 of the 50 game versions returned `KeyErrors`, and while investigating these versions in the Scratch editor, I found that they had variables on the screen. Two found examples were for instance counters for time and high score. When hiding these variables from screen, the artifacts still had the same functionality, and a majority of the game versions were successfully analyzed when re-uploaded to `Dr.Scratch`. This error was not caused by bad code or faulty logic, it was simply due to `Dr.Scratch` and the tool's inability to analyze Scratch projects with variables on screen. However, game versions that still failed to analyze after removing variables from the screen, returned the same `Dr.Scratch` error that I encountered earlier. After investigating the remainder of artifacts, I ultimately discovered that the teams forgot to connect event code blocks to motion or control blocks. Some of the teams had placed the event block "When start flag is clicked" without any connected code blocks in a sprite or stage, while in other game versions, teams did not connect their control or motion blocks to any event block. After fixing these problems manually in each remaining Scratch project, the game versions were uploaded to `Dr.Scratch` once more, and all of them were finally successfully analyzed. The explanation for the second error could be that game versions were saved at fixed times, optimally several times during the game creation phase. The researchers might have saved versions of the games while a team was trying to edit or implement new functionality, meaning that a separation of artifacts that intentionally contained "bad" code, and those that unintentionally had it was challenging. Nonetheless, I could look at each team's last version of game, and see if I could detect the bad code in this version. If so, this was a valid reason to lower the respective points given by `Dr.Scratch` for concepts like parallelism.

### **3.6.4 Questionnaire**

Before handing out the questionnaires, a spreadsheet for each workshop was created. For each questionnaire the researchers received, cells with participation date, gender, age, and the team the participant was a member of were filled out. Scores for the second part of each questionnaire were also documented. If a participant answered 2 (disagree) on the Likert score, the same number would be written down to the corresponding statement in the spreadsheet. If a participant chose more than one score, the lowest score was chosen as their answer. When all data was filled out, frequencies and percentages for each score from 1 to 7 could be calculated. This way, average scores and unusual answers could be detected.

## **3.7 Validity and reliability**

Validity and reliability are conceptualized as trustworthiness, rigor, and quality of a research project (Golafshani, 2003). The terms are used to reflect the multiple ways of establishing truth by eliminating bias, utilizing triangulation, and increasing the researcher's truthfulness. Validity in a piece of research means that the researchers have used an appropriate process, the findings do indeed come from the data, and the data answers the research questions (Oates, 2006). Since this study has a mixed methods approach, it is therefore essential that the study has a strong qualitative and quantitative validity (Johnson and Christensen, 2008).

Yin (1981) identifies three different types of validity: Construct validity, internal validity, and external validity. Although there is not an easy, single way to determine what construct validity is, we can say for simplicity that it concerns how well the operational measures performed, and how well it measures what we intended to measure (Robson and McCartan, 2016;

Lorås, 2017). Internal validity involves ending up with credible results during analysis, and can be divided into three subcategories: Content validity, criterion-related validity, and construct validity (Roberts et al., 2006). Finally, external validity concerns how the findings are secured and can be generalized during the research design. Robson and McCartan (2016) proposed description, interpretation, and theory as three threats to the validity of a research study. In other words, researchers should aim to minimize the inaccuracy or incompleteness of data, draw conclusions from lessons learned during the involvement, be able to explain how they concluded to their interpretation, and lastly understand the studied phenomena, and other alternative explanations to the conclusions. The next subsections will be dedicated to inspect the validity and reliability of this study.

### **3.7.1 Validity of research design and data collection**

The research approach for this study was based on established theory on mixed methods research, as several sources were used to design the approach to fit the research inquiry. This consequently acted as an audit trail and secured an external validity of the study through transparency (Robson and McCartan, 2016).

Because of the short time span of the workshops and few interactions between respondents and researchers, it was not easy for us to gain trust from the participants, which optimally could reduce the threat of respondent bias. To assure participants that they were allowed to speak and answer frankly, the researchers repeatedly reminded them that they could express anything they thought of without fearing any consequences. We could not secure truthfulness and openness of the answers in a larger extent than this.

With content validity in mind, statements in the questionnaires and the interviews were inspected and agreed upon by at least two researchers. Peer-reviews were conducted to ensure that data about relevant concepts would be collected, and that words were unambiguous, specific, and objective (Oates, 2006). When conducting the questionnaires and interviews, we collected raw data, but the credibility of said data was dependent on the participants. However, the age gaps between the participants and the researchers were not that evident, possibly contributing to a strengthened trust between the two parties.

For observations, we had a dedicated team of researchers, and each researcher was in charge of observing one to two teams. Even though the researchers had in mind to stay objective and follow the observation guidelines, humans are imperfect, which can result in differences in field notes. In addition to this, not all situations were caught, due to unexpected situations such as phone calls and conflicting plans for the researchers. This flaw could consequently lead to decreased descriptive validity. How a researcher observes and interprets is also dependent on the researcher's personal experiences, and might lead to a decrease of interpretive validity. Thus to deal with bias from participants and researchers, verbatim quotations from the people in right context, data and methodological triangulation, and reflexivity were used to encounter these common problems to research projects (Oates, 2006; Robson and McCartan, 2016). The process of triangulation is considered a prevalent practice to provide corroborating evidence, as we can rely on multiple forms of evidence instead of one single source of data (Creswell and Miller, 2000). An example from this study was interviews and observations, which could lead to an increased criterion-related validity.

### **3.7.2 Validity of analysis**

According to Creswell and Miller (2000), procedures for validity include strategies used by researchers to establish the credibility of their study. The findings of any study should be found trustworthy and believable in a way that they reflect participants', researchers', and readers' experiences with the phenomena, in addition to how well they are transferable to other situations (Corbin et al., 2014; Lorås, 2017). However, it is also important to note that the explanation my theory provides is only one of many possible interpretations of the data.

During the analysis, some measures were taken to increase the validity of analysis such as peer reviews with a Ph.D. student to ensure reflexivity (Creswell and Miller, 2000). Peer-reviews were conducted to avoid that personal beliefs, personal experiences, and biases were taken too much into account when analyzing the data, and to ensure the credibility in the views of the participants. In research projects, there will however always be some extent of researcher bias. In order to be transparent and neutralize my own opinions, I have described my opinions and preconceptions in Section 3.2. Another measure used for increasing validity of analysis was to use negative case analyses, which are cases that contrast and contradict the main findings of a study (Robson and McCartan, 2016). This kind of data was considered and included in the analysis process.

### **3.7.3 Reliability**

Reliability is about the trustworthiness of the procedures and data generated, and how consistent the results of a study would be if researchers reconstructed the process under similar conditions (Roberts et al., 2006; Johnson and Christensen, 2008; Robson and McCartan, 2016). To secure reliability, transparency about the research process and personal beliefs is

important. While conducting the study, the researchers had in mind to collect as much data as possible from participating teens, and we were therefore dependent on the participants from the workshops. Detailed notes on decision-making and conducted steps were written down throughout the process, and many of the choices in this study have been explained and justified to increase the reliability. Furthermore, transcriptions, artifacts, field notes, and audio tapes have been kept and secured to confirm the findings of this study. The author's personal position and potential preconceptions have also been elaborated earlier in this chapter, which again can increase the reliability of this study.

### **3.8 Statement of ethics and confidentiality**

The researchers made sure that parent and participant permissions to participate in this research projects were secured before they conducted the workshops. Signed permissions allowed the researchers to collect sensitive and personal data about the participants through interviews, observations, and audio recordings. We maintained participant confidentiality, avoided collecting names, and data was coded and reported in group format, rather than individually. The researchers also stored the consent forms in a secured office that only a few people have access to. The collected data will only be available for use in possible future comparative analyses led by researchers associated with the Norwegian University of Science and Technology (NTNU).

The most important thing in this study concerning ethics is to secure the privacy of the participants. The personal security of the participants who contributed was especially important, and the researchers had in mind that identities should and would not be compromised (Robson and McCartan, 2016). The study was notified to the Norwegian Centre for Research Data

(NSD) in order to collect data about individuals ethically, and it followed NSD's general guidelines for data collection. Although we did not store any real names of the participants and gave each participant an alias, their privacy could still be at risk. During the data collection phase, some safety measures were done to protect the data and anonymity of the participants. We wrote field notes by hand and documented them digitally on the computer, and interviews were audiotaped without names and later transcribed digitally. Handwritten notes were at all times stored in a secure and locked place, which only a small group of people had access to. The digital notes and the audio tapes were stored on the cloud, and only accessible with the correct passwords.

Usually, observed participants volunteered to be interviewed, but occasionally some did not want to be interviewed at all. The researchers informed participants that they only had to answer the questions they wanted to, and that they could end the interview at any time. We avoided the unwilling participants as much as possible, but if no one volunteered, individual team members were asked directly if he or she wanted to be interviewed. By having a combination of interviewees who volunteered themselves and interviewees who were asked, we also ended up having a diverse participant pool. Seldom times there was simply not enough time to interview all team members, and the remaining members were informed about this so no one would feel left out.

# Chapter 4

## Results

The last chapter described where, how, and from whom the data was collected in this study. Furthermore, it explained the research approach and steps taken in order to analyze data and secure the validity and reliability of the study and its findings. This chapter presents the results of the study, before the findings are discussed in Chapter 5. If not stated otherwise, the results from both Tappetina and Kodeløypa are presented together. Overall there were not any significant differences in the results from each workshop.

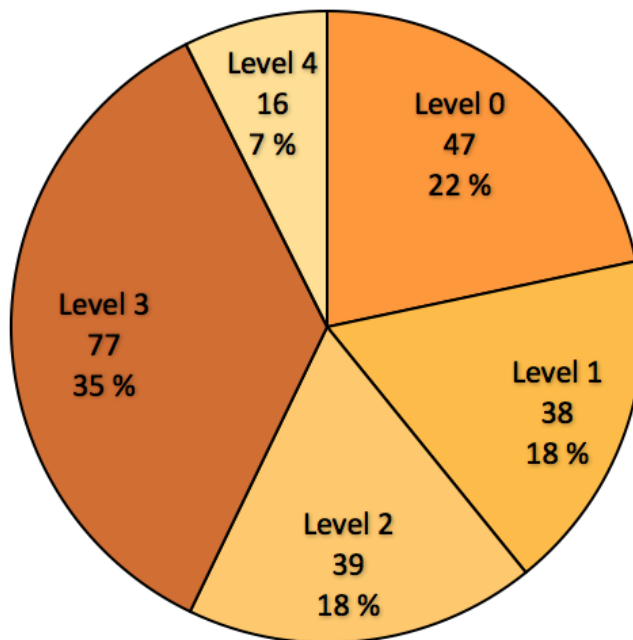


## 4.1 Observations

The observations revealed that generally, the participants seemed to collaborate quite well. Very frequently one team member would start taking over the mouse and begin the programming process, while others contributed with suggestions to the game design, or ideas on how to create different implementations by reading the paper tutorial. In general, we also experienced that teams compromised on what they were creating. Some participants would therefore choose the game design, while others designed the graphical aspects of the game elements. In many teams, the members would rotate between having the control of the computer mouse, so that every member could try programming for themselves. The researchers also experienced repeatedly that in triad teams, it was a higher chance that one member got separated from the rest of the team by talking to teachers, browsing on the Internet, or wandering around the room to see what other teams were doing. This happened regardless of all team members demonstrating an interest towards the activities. In contrast, when the levels of skills were different in dyads, we experienced that the less interested or inexperienced members got a boost when given a chance to experiment with the games.

Furthermore, the researchers detected communication and collaboration between teams as well. If the games did not work as expected, there was a chance that team members asked other fellow participants for help. On the other side, if participants were successful in implementing something, they usually demonstrated and showed this off to other teams. These participants would also try to assist teams that struggled with implementing the same functionality.

As mentioned before, investigating the performance in computational thinking can provide additional information about the sample. While investigating the different help levels in the field notes, we found the results as shown in Figure 4.1. In this pie chart, darker colors represent the higher frequencies, while lighter colors represent lower frequencies. During the workshops, the researchers observed that the teams asked for assistance 217 times in total. By comparing the numbers for help levels, we found that level 3 had a higher frequency than the others (77 times). This implies that participants had an idea of what things were when given the name of a concept, but they did not know how to implement it.



**Figure 4.1:** The ratio of the indicated help the teens received in the workshops

A second thing we found was a high frequency of level 0 assistance (47 times). We noticed in the field notes that teams who demonstrated more experience in programming and game creation had a higher chance of asking the student assistants to try out their games. In contrast, those who demonstrated less experience wanted confirmation on what they were creating. The latter would many times postpone the programming, and mainly focus on the graphics during the game creation phase. When they finally needed to start programming, they would ask for help instead of trying first. While there is nothing wrong with asking for assistance, these teams would receive help, continue making small graphical changes to their game, and then ask for more implementation help. In other words, these team did not try implementing code on their own.

The following tables 4.1 and 4.2 describe the different observed purposes of help and their corresponding frequencies in decreasing order. The table contains 213 purposes, which means that we are missing four. The most common purpose of help was validation of game (47 times), while the second and third common were regarding graphics (31 times) and moving a sprite horizontally or vertically (22 times). More "complex" functionalities such as timers, point systems, and high scores were less common help purposes.

<b>Purpose of help</b>	<b>Frequency</b>	<b>Includes</b>
Validation of game	47	The team telling the teacher to try the game, wanting confirmation of game elements, or explaining the game to the teacher
Graphics	31	Creating a new background or sprite, setting a background or change the costume of a sprite, shrinking or enlarging a sprite, as well as creating the effect of a never-ending background
Movement problems	22	Moving a sprite horizontally or vertically in a way the team desires
Jumping	20	Making the sprite glide naturally or making the sprite jump to left or right
Loops	16	Explanations of loops, if else, if touching, and help to create different loops for different purposes
Hide/show	14	Helping the teams to hide or show a sprite when something happens, debugging when existing scripts do not work, and explaining how they can hide or show something
Coordinates	10	Explanation of the coordinate system, how coordinates are used in the games, and adding the "correct" x and y values
High score	9	Creating a high score system, debugging existing high score code, and explaining how a team can create a high score system
Point system	9	Implementation of a point system, how to keep track of existing points, and assisting when points should be decreased or increased in a game
Game over	9	Restarting the game, making a Game over-screen appear when a player lost the game, or making the game over screen show at the correct time

**Table 4.1:** Overview of the purposes of help and frequencies in the workshops

<b>Purpose of help</b>	<b>Frequency</b>	<b>Includes</b>
Respawning and randomize	8	Help to respawn a sprite when the game is over, when something is touched, when it is out of the screen, and giving sprites new positions in a random manner
Variables	8	Explaining the concept of variables and how they can be used
Finding the right block	3	Help to navigate the Scratch GUI
Copy existing code	3	How to copy code from an existing sprite to a new one
Timer	3	How to create a timer that either counts up or down during a game
Game planning	1	Help on how to start planning the game the team is creating, and possibilities of Scratch

**Table 4.2:** Overview of the purposes of help and frequencies in the workshops (continued)

## 4.2 Interviews

Note that most of the interview quotes were translated from Norwegian into English, and names have been changed to preserve participant confidentiality. The interviews revealed various things: Many participants enjoyed the given activities, as they were in charge of both planning and realizing the games. Of the planning part, they expressed appreciation towards the fact that they could be creative and were allowed to create almost anything they wanted. These elements were different from what they usually had in school. When it came down to the implementation of games, they found it engaging to see how implementations were built up by different blocks:

*"It was fun to put together different program parts and see that they worked together"*

*- Magnus*

*"It was really fun to see how things worked when we changed something, even the slightest bit, and it changed the whole thing"*

*- Su*

*"I thought it was fun, especially because we were allowed to do whatever we wanted"*

*- Bendik*

What participants expected was something that could influence their attitudes towards the activities. Some teens expressed that they did not look forward to participating, as they did not really like programming. Others thought programming was hard, and that they did not know enough to create something. Compared to those who found programming harder than anticipated, almost twice as many implied that it was either easier than expected or better than they thought. More experienced participants proclaimed that the reasoning for this could be the easiness of Scratch, and the flexible opportunities that comes with the environment:

*"At times, you have people who aren't really interested in programming. They can create easy things in Scratch"*

*- Peder*

*"You can create a game in a short amount of time"*

*- Christoffer*

Teens gave various statements regarding what they learned from the activity. Several participants found it enjoyable to transfer existing knowledge to new situations, and others said that they had started to understand that things would not happen if they did not explicitly give the instructions. After some time with programming, they got a better idea of what the different blocks did, and that they could reuse code in other scenarios than they initially thought of:

*"When you have learned something, you could use it (again)"*

- My

*"I think that it is more fun to learn from your mistakes"*

- Cathrine

Others expressed that it was challenging to create smaller scripts and connect them together into one game. It was especially hard to tweak the existing scripts since this could end up breaking the whole game. The teens established assistance from assistants and paper tutorials as two remedies to reduce frustration when overcoming obstacles. Support and help from the assistants were mainly found valuable, as this could guide the participants closer to their goals:

*"It has been very helpful to get a lot of help, to receive help from people who look over your shoulder and point you in the right direction"*

- Jens

*"It wasn't frustrating because we received help almost immediately"*

- Frederik

*"We managed to solve it by following the tutorial"*

*- Tuan*

Something else that could ease frustration was to receive input and knowledge from other teams. If the paper tutorials were insufficient, and student assistants were unavailable, teams would turn to fellow participants for assistance. A participant gave an example where her team could not find what they were looking for, so they looked at how other teams to see how they did it. In addition to this, participants would draw inspiration from games that belonged to other teams, go back to their own computers, and try to implement new game concepts into their own games.

More experienced participants agreed that Scratch was more of a limitation than an aid for programming. These teens implied that the environment was unimpressive since they knew how to write textual-based code. In addition to this, the easiness of Scratch made it difficult for them to create what they wanted as they were used to write code themselves:

*"It was harder because you can't specify whatever you really want to do"*

*- Kristian*

Another highlight from the interviews was the collaborative aspects of the workshops, which in most cases seemed to be uplifting and beneficial to participants. They expressed several times that the end results were better than expected, and that it was fun to build something together:

*"Fun to create something. A product!"*

*- Didi*



In some cases, the teens described that the end results would not be as good if they executed the game creation on their own. These participants also stated that they were involved in the game creation, but that their partners often did more:

*"If I was alone I wouldn't manage to do the same"*

- Anna

*"I felt like we did as much, but I feel like he might have done a little more"*

- My

Regardless of the amount of work each member put into their games, teens still viewed their involvement as vital. One teen said that he participated in the activities by thinking about what the team could do and create. Another girl stated that she contributed to the areas she could and that she did a pretty good job at it. Although they were aware that they did less, this was not what they focused the most on. Compared to the amount of work each member put into the games, participants pointed out that it was more important that they had some input in the games, that their team was able to create what they had planned, and that team members were open for their ideas:

*"It was like our group, our project, like, listen to everyone equally"*

- Lars

*"It was fun when things worked and we thought about things and they actually worked"*

- Per

Another participant expressed that being more than two in a team would however limit the freedom to create anything they wanted to create:

*"It was good that we were two in the team, if we were more you wouldn't be able to do whatever you want"*

- *Huong*

We also experienced traces of uneven roles in the collaborations. Teens who felt that they were dominant during the game creation usually reflected over the division of roles in the teams. They realized that they took a leadership role in the team, and regretted being too bossy:

*"I felt maybe that I got too active and others did not get to say as much"*

- *Harald*

*"It was maybe that I took too much control"*

- *Peder*

In few situations, we experienced bad collaboration because the team members could not agree on ideas and the division of tasks. These participants ultimately became frustrated and uninterested in the associated tasks:

*"I had to sit on the side, they did not let me try ... It was hard to finish my task because they just wanted the control of the computer back"*

- *Daniella*

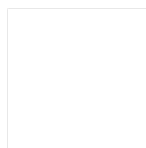
*"It was every man for himself"*

- *Hannah*

Aforementioned in Chapter 3, the results from the coding process were a large number of codes, categories, and themes. The following tables 4.4 and 4.3 describe the categories we found during that process. Further breakdowns of the categories can be found in Appendix E and Appendix F.

<b>Category</b>	<b>Description</b>	<b>Codes</b>
<b>Difficulties</b>	This includes encountered difficulties in Scratch, being unable to create the functionality the participants wanted, and expressing mistakes and frustrations related to the game creation phase	46
<b>Emotions</b>	Participants express some kind of emotion such as confidence, achievement, and fun	30
<b>Scratch</b>	Participants' opinion on Scratch in general, and challenges when moving from paper to Scratch	27
<b>Collaboration</b>	Participants speaking up about their programming experience in small teams, distribution of roles, and how the collaboration was during the idea creation	23
<b>Easy</b>	Participants reflecting on the easiness of tasks and Scratch	16
<b>Environmental issues</b>	The participants say something about the theme environmental issues	7
<b>Time issues</b>	The participants reflect on the time aspect of the workshop, and how it could have affected their end results	7

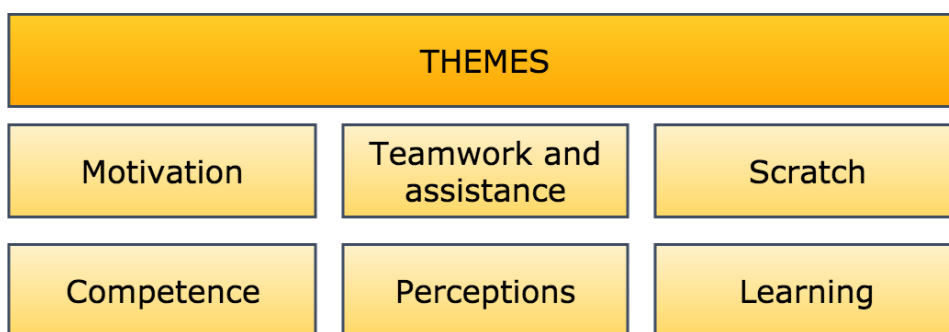
**Table 4.3:** Overview of categories and number of codes after coding Tappetina interviews



<b>Category</b>	<b>Description</b>	<b>Codes</b>
<b>Collaboration</b>	Participants describe their opinions on the collaborative learning, their own contributions, and roles	136
<b>Difficulties</b>	Participants state what they found hard in Scratch, with game planning and while programming	128
<b>Out of school experience</b>	Participants indicate that they enjoyed problem-solving outside of school. This includes the freedom to create anything, looking how things were built up, and being creative	59
<b>Easy</b>	Participants express what they found easy, why Scratch is hard for textual-based programmers, and the easiness of using Scratch to program	57
<b>Fun</b>	Participants experience fun while using Scratch, being challenged, achieving to create a game, and being creative	46
<b>Achievement</b>	This category unifies what the participants felt about the final result and the evolution of the game creation	23
<b>Scratch potential</b>	Participants looking at the possibilities with Scratch and their previous experiences	21
<b>Help</b>	This category unites types of help provided, when they needed help, and how some assistance made the participants feel	21
<b>Change of perceptions</b>	Participants expected programming to be harder, easier, or more boring	21
<b>Scratch discoveries</b>	Participants speaking about their discoveries of blocks and creating scripts in Scratch	20
<b>Learning</b>	Participants saying what they learned in general, about instructions, and programming	18
<b>Intention</b>	Participants look at their intentions for programming and Scratch in the future	4

**Table 4.4:** Overview of categories and number of codes after coding Kodeløypa interviews

The final results of the coding process were in total six themes, as illustrated in Figure 4.2: **Motivation, Teamwork and assistance, Scratch, Competence, Perceptions**, and finally **Learning**. The underlying categories of each theme can found in Appendix G. In the light of concepts and research presented in earlier chapters, the connections between the results and the four distinct parts of engagement were examined. The connections will be further discussed in the next chapter.



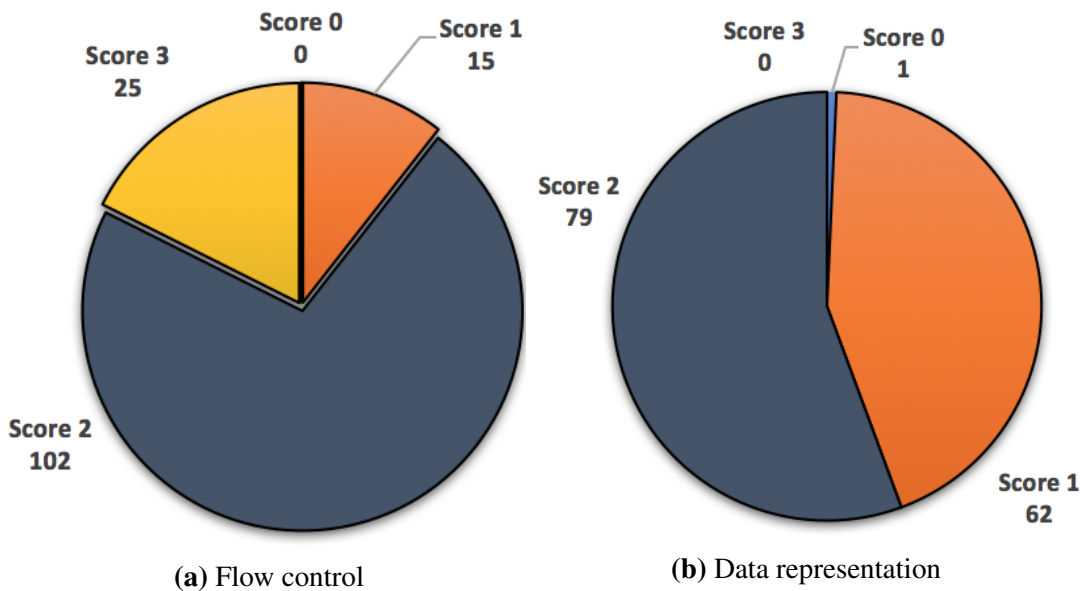
**Figure 4.2:** Final themes after coding interviews from both workshops

### 4.3 Artifacts

Tracking Dr.Scratch scores in a spreadsheet made it easy and clear to find the frequencies and percentages for each assessed score, especially since we collected 142 game versions. The ratios of scores for the computational thinking concepts used by Dr.Scratch are presented in the following pie charts, and for the sake of consistency, the scores are distinguished with the same color in each pie chart. Overall, the results imply that participants generally performed a basic and developing level of computational thinking skills. However, more diverse results in the performances were detected for synchronization, parallelism, and logic.

Starting from **flow control**, none of the artifacts scored level 0, and 15 of them gave a score of 1, meaning that all the teams managed to reach a basic level of flow control. 102 game versions reached a score of 2, implying that repeat and forever blocks were used successfully, and the flow control among the participants was in general developing. 25 of the game versions scored 3 and showed signs of proficiency.

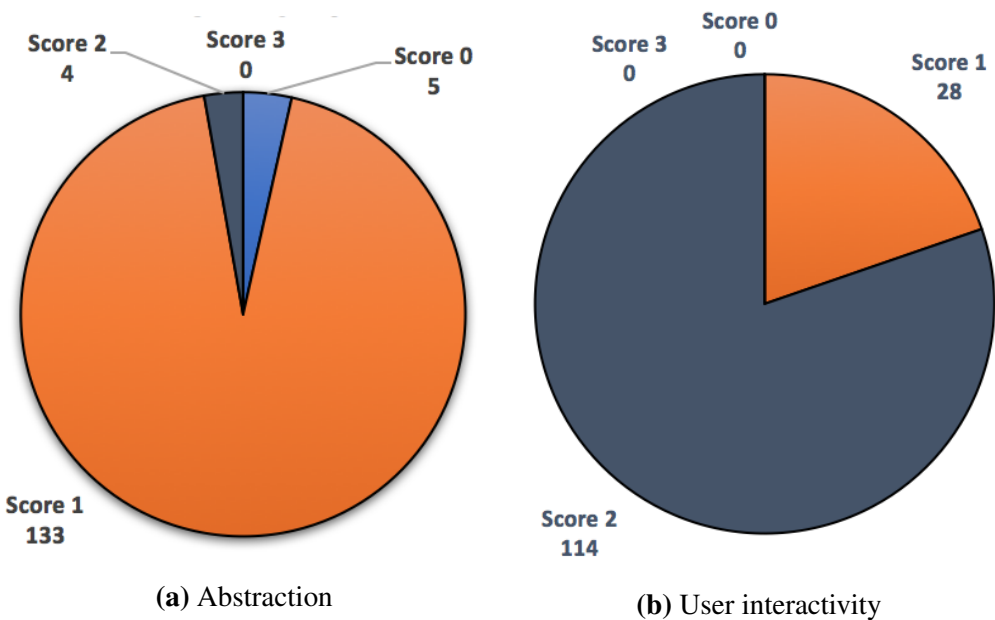
For **data representation**, only one game version produced a rank of 0, meaning that the version did not even contain modifiers of sprites properties. However, 62 artifacts reached the fundamental level, and contained modifiers, while the majority of the game versions (79) had a developing level (score 2) with operations on variables. None of the Scratch projects reached the proficiency level (score 3), which included carrying out operations on lists.



**Figure 4.3:** The ratio of Dr.Scratch scores for artifacts

Continuing with **abstraction and decomposition**, five game versions resulted in score 0, meaning that they did not contain at least one script and a sprite. Almost all the game versions (133) held a basic level of abstraction with more than one script and sprite, while only four versions reached a developing level with own definition of blocks. None of the versions resulted in score 3 for abstraction.

**User interactivity** scored high as 114 game versions resulted in score 2. In other words, the majority of the participating teams managed to create functionality reacting on key and mouse presses, green starting flags, clicks on sprites, as well as the block ask and wait. None of the versions reached the level of proficiency.



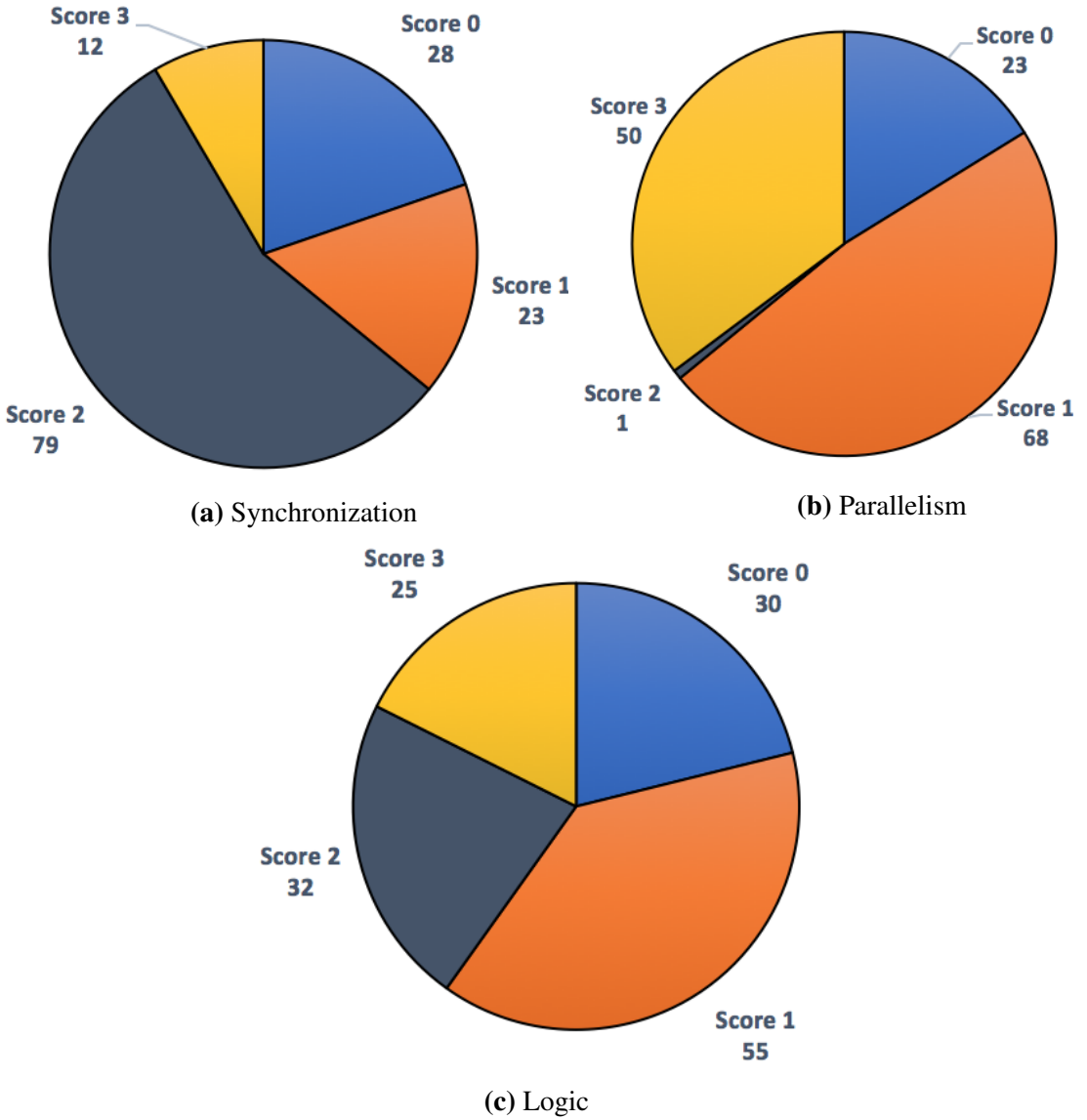
**Figure 4.4:** The ratio of Dr.Scratch scores for artifacts (continued)

When it comes to **synchronization**, 28 artifacts scored 0, and 23 scored 1. The teams were required to implement a wait block in their games in order to reach a fundamental level. A more significant part of the game versions resulted in a developing level, meaning that participants implemented functionality for broadcasting, stopping sprites, stopping the program, or stop everything. 12 game versions scored 3, and introduced blocks for broadcast and wait, wait until, and when backdrop change to.

Scores for **parallelism** varied, but 23 artifacts produced score 0, signifying that maximum one sprite reacted on the green start flag. 68 game versions reached the basic level of score 1, which involves two sprites reacting on the green flag. Only one game version ended up with score 2, while 50 game versions reached a proficiency level. The latter includes for instance creating clones and executing two scripts when a message is received.

Lastly, **logic** had diverse results. 30 game versions did not reach the basic level, while 55 artifacts implemented if conditionals. Furthermore, 32 game versions resulted in score 2 by using if else code blocks, and the remaining 25 had a proficiency level by carrying out logical operations successfully.

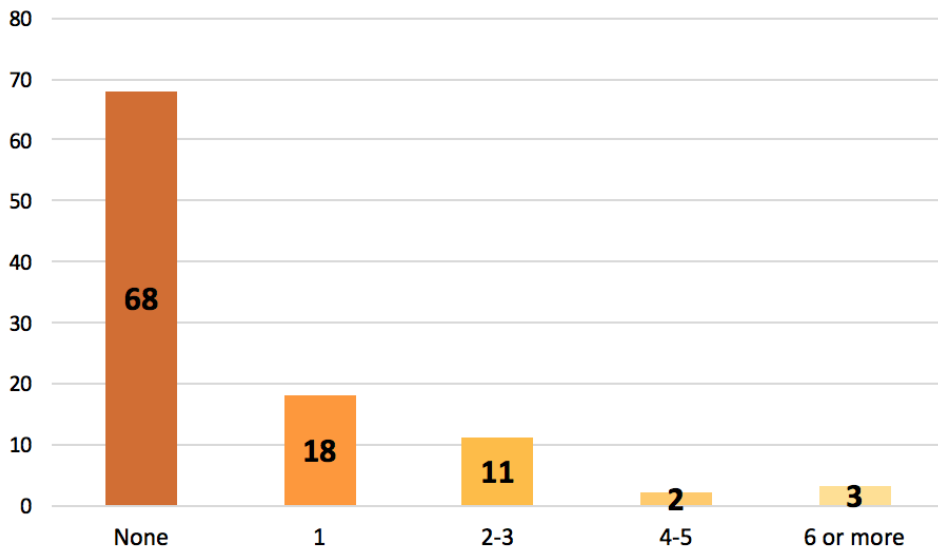




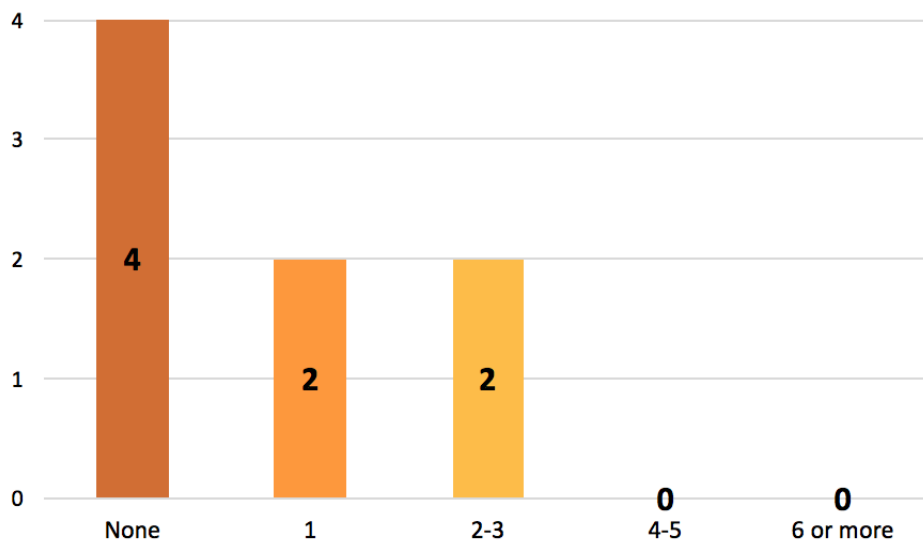
**Figure 4.5:** The ratio of Dr.Scratch scores for artifacts (continued)

## 4.4 Questionnaires

Out of all the participants in our workshops, at least half of them reported that they had never participated in a workshop with visual programming before. For Kodeløypa the percentage was as high as 63%, while it was 50% for Tappetina, as illustrated in Figure 4.6 and Figure 4.7. In contrast, almost 68% from Kodeløypa answered that they had experience with programming to some extent prior to the workshop, while 6 out of 7 Tappetina participants reported the same.



**Figure 4.6:** The number of previous visual programming activities before the teens participated in the Kodeløypa workshop



**Figure 4.7:** The number of previous visual programming activities before the teens participated in the Tappetina workshop

As mentioned earlier, the Kodeløypa questionnaire aimed to investigate the collaborative learning experienced by each team member. A clear majority of the participants (81%) agreed that they felt a part of the learning community in the team to various extents. 92% reported that they were actively sharing their ideas with their team, and 85% were overall satisfied with the collaborative learning in the workshops. When we asked the participants if they developed any knowledge or new skills from their team, 75% reported that they did, while only 18% said they did not.

Almost every participant from the Tappetina workshop strongly disagreed that programming activities were boring, and every participant agreed that their decision to attend this kind of activity was a wise one. Their general intentions to attend programming activities were high, which can imply that the teens in general had motives for participating. The teens also found programming useful and exciting, meaning that the participants had an interest in programming. In contrast, almost every participant disagreed that

they found programming flexible. They also agreed that the process of programming was unclear, and could be misunderstood at times. Nonetheless, many teens reported that programming activities would improve their performance and effectiveness in programming, and that learning as much as possible when programming was one of their goals. This signifies that the teens were motivated to learn and improve their existing skills.

The questionnaire results can be found in the appendixes H and I.

# Chapter 5

## Discussion

Chapter 4 presented the results from the data collection, such as code categories, themes, and Dr.Scratch scores for the collected artifacts. This chapter describes how the results relate to existing research and this study's research inquiry: Which factors can influence teens' engagement in computational thinking activities? Section 5.1 describes the computational thinking performance the teens were able to reach in this workshop. This brief assessment will provide a better picture of this study's sample. The following section 5.2 assesses the academic engaging factors of the activities and their significance. Section 5.3 describes how the factors affect behavioral, emotional, cognitive, and agentic engagement, while Section 5.4 discusses the generalizability, implications, and limitations of these findings. Finally, Section 5.5 concludes and presents the closing words of this study.

## 5.1 Teens' position in computational thinking

The findings in this study indicate that the majority of teens had little to no experience with visual programming activities before their participation. Through questionnaires, it was self-reported that approximately 65% had never taken part in a similar activity, less than 20% had participated in one, and less than 1% had participated in four or more similar workshops. By observing the teens during the workshops, we detected various things, for instance the types of help each team received, and for which purposes. The second most frequent purpose was about graphics, which included for example how to create new sprites and backgrounds, and resizing of sprites. While aesthetics can be significant for the success of a game, Maloney et al. (2008) found that novices mainly use Scratch as a media manipulation and composition tool. The focus on graphics can therefore be a sign of little programming experience among the participants.

The other help purposes and artifact analysis can also support the idea of participants having little programming experience. Generally, the teens experienced problems with implementing movements, loops, and jumping. By comparing these frequencies with the frequencies of more complex functionalities such as high score and point systems, the former was noticeably higher. This implies that the participants needed assistance for the more basic implementations in Scratch. The artifact analysis revealed that synchronization, parallelism, and logic had higher percentages of score 0 compared to the other computational thinking concepts assessed with Dr.Scratch. While the scores of synchronization and parallelism are dependent on the design of games, they require knowledge regarding the order of instructions, Boolean logic, and connecting the right code blocks together. In contrast, logical thinking is considered essential for any programming project and requires knowledge about for instance Boolean logic and conditionals. Approximately 21% of the game versions in this study did not

contain any successful if conditionals, while 39% of the versions reached a basic level for logical thinking, and implemented functional if conditionals. Similarly, Denner et al. (2012) found that 82% of 108 games made by novice programmers contained conditionals or events, but 18% of these did not have any functional interaction. In only 35% of the 108 games, all the implementations with conditionals were successful. The levels of computational thinking in this study match the performances of novice programmers in other studies. In a like manner, Werner et al. (2014) found that conditional logic and subsequent sequential execution of a program are challenging concepts for novice programmers. On the other side, Maloney et al. (2008) stated that variables and Boolean logic are less easily discovered computational thinking concepts for youth, and that the uses increase over time when they gain more experience. With some few exceptions, the results indicate that the participants of this study were newcomers with little to no experience with computational thinking.

## **5.2 Factors of academic engaging activities**

The findings of this study imply that teens in a computational thinking activity are motivated by what they can create with visual programming environments, what they can produce in teams, and how they can modify the tasks to own personal expectations. They react positively to the collaborative and creative aspects of the activities, as well as the help they receive to reach their goals.

### 5.2.1 Teamwork and assistance

Children become confident when they have the chance to learn and receive ideas from others (Giannakos and Jaccheri, 2013). This way, activities can also become more enjoyable and creative for the teens. This study indicates that the collaborative aspects of the activities are positive for teens' academic engagement (Juvonen et al., 2012; Riga and Chronopoulou, 2014). Teamwork is found uplifting in cases where team members feel less capable than their partners, in that sense that different team members can contribute in areas that they feel comfortable with. Less skilled teens find it motivating to learn from more skilled teens, while more experienced teens enjoy teaching less experienced ones (Wentzel and Watkins, 2002). Especially creating things with others is valuable to the teens, and creating a functional and good end product has a higher importance than individual efforts. However, the findings also imply that the *feeling* of equality between team members is important. This equality includes that team members should consider their partners' opinions equally, and that every team member should try to contribute to the game as much as possible.

The research reveals some possible differences between dyads and triads as well. Having three team members can make one member prone to being less involved in the games, as maintaining equality is harder, and teams have to compromise more on their ideas. Teams with a large number of members can ultimately force the members to consider more opinions, and therefore lead to fewer individual contributions to the end product (Downer et al., 2007).

On the other hand, bad collaboration can absolutely lead to a decrease in engagement. The research implies that instances of inequality between the team members can result in negative feelings towards the associated activities (Wentzel and Watkins, 2002; Furrer and Skinner, 2003). Furthermore, disagreements and inabilities of compromising within the teams can lead to



an inferior end product, and cause frustrations for the team members.

This study's findings suggest that assistance and feedback are another positive factors for the academic engagement. As discussed, less skilled teens enjoy learning from more skilled individuals, which includes learning from more authorized persons such as teachers and student assistants (Reeve, 2013). Assistance is found to ease frustrations in challenging situations, and as a push in the right direction (Furrer and Skinner, 2003; Appleton et al., 2008; Pianta et al., 2012). In times where students assistants are busy, researchers should provide further assistance in the form of paper tutorials and related example projects to avoid decreases of engagement. In computational thinking activities, validation from both peers and student assistants are also invaluable (Skinner and Belmont, 1993). Teens feel validated by asking teachers to try out games, receiving confirmations of game elements, and explaining their games. The high number for the validation of games in this study implies that teens appreciate and consider responses to their accomplishments made by others (Basawapatna et al., 2010).

### **5.2.2 Competence**

This research suggests that teens' levels of competence can influence their academic engagement (Bandura and Cervone, 1986; Bircan and Sungur, 2016). It implies that novice learners in similar computational thinking activities will enjoy learning new things, setting goals, and lastly realizing their goals. They will however be discouraged during the workshop by not knowing what they can create, challenging implementations, and failing to meet their goals (Lykke et al., 2015). When teens are in charge of planning and implementing their own games, they decide the complexity of their games, as well as how challenging their learning process should be. Despite having little to no experience with programming, teens express that activities are more gratifying and entertaining with some practice, and

that it is fun to learn from mistakes. This finding supports Bandura and Cervone (1986) who found that students who set goals for their academic activities, have a higher willingness towards achieving their goals. It should nonetheless be a balance between the teens' effort and the outcomes of it, as an apparent imbalance can cause frustrations, and decreases in academic engagement among teens (Wentzel and Watkins, 2002).

By looking at the games, it is evident that teens are inspired by games they have played before, for instance Flappy Bird, and use game elements that they are familiar with such as Game over-screens, obstacles, and continuous moving backgrounds. Game creation has been found to increase motivation and engagement for both inexperienced and experienced teens (Repenning et al., 2010; Çakır et al., 2017). It is also considered a beneficial way to learn programming and computational thinking, and can teach students to transfer existing knowledge to other situations (Adams and Webster, 2012). According to Basawapatna et al. (2010), games are further able to enable more peer-to-peer interaction and peer learning in middle school, as teens naturally will walk and investigate what others have created, go back to their computers, and implement what they have learned. Since games count as entertainment, viral peer learning will therefore take place with little or no input (Basawapatna et al., 2010; Buffum et al., 2016; An, 2016).

### **5.2.3 Motivation**

This study indicates that teens' emotions and their senses of achievement and fun influence their motivation, and thus their academic engagement (Papert, 1980; Bandura and Cervone, 1986; Caraway et al., 2003; Turner et al., 2014). While some enjoy the processes of putting programming blocks together and the results of it, others find it exciting to be challenged and play the end results. They all agree that it is engaging that they think about something, and their ideas seem to work (Reeve and Tseng, 2011).

Their confidence levels increase when they manage to do something, while it decreases if they fail at doing so.

In terms of motivation, looking at teens' intentions is interesting. Regardless of prior knowledge, teens find building on existing knowledge or acquirement of programming experiences motivating. This motivation makes teens want to continue to program during the breaks or after the workshop programs.

### **5.2.4 Perceptions**

Something interesting in the findings is that teens enjoy the out of school context and participating in activities that are different from what they usually do in school. Room for personalization, self-exploration, and the possibility of adjusting learning processes to own values are found especially valuable (Kelleher et al., 2007; Giannakos et al., 2013; Giannakos and Jachcheri, 2013). By providing autonomy to teens, they feel stronger about creating the games, and the final end results (Turner et al., 2014). The findings imply that teens appreciate that they are allowed to think on their own, which result in feelings of ownership for the games.

Some teens find programming harder than expected, while a higher number find it easier and better. Although programming can be hard sometimes, the teens realize that they in general end up with better results than expected. They achieve more than they think they can and gain insight into the amount of work that lies behind a game. Some teens express that they got new perspectives of programming (Grover et al., 2014b), and many feel that it is interesting to program and play games that fellow participants create. However, there are naturally a few cases where participants are found to be disengaged in the activities. These reactions are typically results of their attitudes since they find programming to be boring. Although Furrer

and Skinner (2003) suggested that emotional disengaged and underachieving students are sensitive and responsive to opportunities that fulfill their motivational needs, the research implies that teens who do not feel emotionally connected to the associated tasks do not have positive increases of engagement. Although students are autonomous and competent learners, they still need to feel a sense of relatedness to be engaged (Furrer and Skinner, 2003; Park et al., 2012).

However, teens acknowledge the creative aspect of the workshops as a decisive factor for their engagement (Giannakos and Jaccheri, 2013; Giannakos et al., 2013; Riga and Chronopoulou, 2014). Creativity is in general found to make activities more interesting for both inexperienced and experienced teens. Where inexperienced teens lack skills, they can still contribute to the idea or graphics in the game. Consequently, this can mean an increased participation during the game creation, and thus an improvement of engagement (Shell et al., 2014).

### **5.2.5 Scratch**

The findings support that Scratch make the learning experiences with programming more playful and understandable for the teens (Resnick et al., 2009; Mannila et al., 2014; Riga and Chronopoulou, 2014). The visual programming environment extends what teens can create since it provides many elements teens do not think about while planning, such as high scores and timers. Teens express that it is easier and faster to sketch on paper than to create something in Scratch, since paper sketches are less detailed. Sketching is however found to be a good starting point for planning games.

The design of Scratch is very user-friendly, and the environment enables teens to connect manipulations to code blocks (Weintrop and Wilensky, 2015). It can also meet the needs of users who only want to manipulate

media and graphics (Maloney et al., 2008). This study indicates that teens enjoy that they can create programming projects in a short amount of time, and that Scratch supports what they want to build in a simple and efficient way. The opportunities that come with Scratch also make it attractive, since teens can create a lot of different programming projects with it, for instance science simulations and projects related to their majors (Appleton et al., 2008; Basawapatna et al., 2010). The low floor makes it possible for even inexperienced teens to create something, and considering that Scratch makes it more effortless to succeed with programming, it contributes to motivating teens in computational thinking activities. The research suggests that teens who have little to no practice with programming will in particular be engaged.

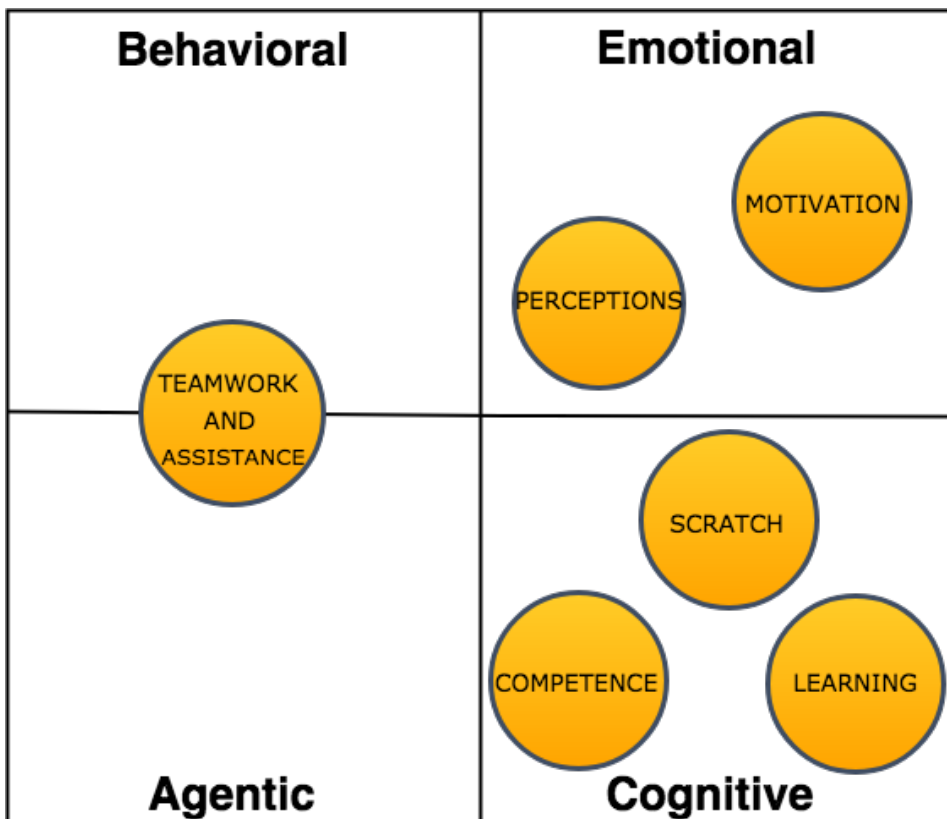
### **5.2.6 Learning**

The learning outcomes of the activities are something that can engage teens in computational thinking activities. The teens in this study express that scripts do not always work as expected, and that they gain more understanding of the blocks and how they can be used when allowed to test and experiment. Others explain that they can relate new insights to their existing knowledge and other situations. The research suggests that majority of participants gain some new knowledge by participating in computational thinking activities. While only a few participants remember what kind of blocks they use in their games, their idea of the blocks is usually correct.

Since we did not test teens' comprehension and knowledge before and after the workshops, a correct conclusion of actual learning outcomes concerning the performance in computational thinking is not possible. The assessment of performance is nonetheless outside of this study's scope and targeted in other studies (Denner et al., 2012; Franklin et al., 2013; Werner et al., 2014; Fields et al., 2016).

### 5.3 Addressing academic engagement types

Figure 5.1 illustrates the placements of coding themes according to the different features of academic engagement. The placements were based on theory introduced in Chapter 2, along with the focus points from Table 3.5. The next subsections will further describe the connections between each theme and engagement type.



**Figure 5.1:** Placements of themes according to the presented academic engagement features

### **5.3.1 Behavioral engagement**

Teens' behavioral engagement is a key condition that supports academic engagement, and is suggested to have a direct influence on raising achievement: The more students actively participate in the activities, the more they will learn the contents (Gregory et al., 2014). Half of the participants agree that they actively exchanged ideas with their partners, and 65% strongly agree that learning in teams was facilitated. This signifies that most participants are involved in computational thinking activities. This study supports that effective and efficient collaborations lead to teens being more involved in computational thinking activities, in particular teams with a proportionate division of tasks between the partners (Wentzel and Watkins, 2002; Riga and Chronopoulou, 2014; Buffum et al., 2016). On the other side, this research also supports that less satisfying relationships between team members have negative impacts on children's self-reports of behavioral engagement (Furrer and Skinner, 2003; An, 2016). High behavioral engagement is associated with visibility in teams rather than individual work, so larger groups are proven to affect student motivation and behavioral engagement in learning negatively (Downer et al., 2007). Members of smaller teams are usually more actively working, compared to larger teams which are associated with more passive engagement such as sitting and listening.

This study's findings suggest creativity and personalization in collaborative activities as measures to increase desire and curiosity among inexperienced teens for learning programming (Burke and Kafai, 2012; Giannakos and Jaccheri, 2013; Giannakos et al., 2013). Creativity, personalization, and collaborations can consequently result in deepening the participation among less interested teens. Teens who already have an interest in programming will actively try to approach it, while disinterested teens can improve their enthusiasm by contributing to the creative aspects of activities. The results point in a direction where the collective learning and

accomplishments of the teams are more meaningful than what an individual can achieve individually. Somehow similarly, teens' self-reports disclose the same findings. Over half of the participants agree to some extent that learning in their teams was useful, while 86% express satisfaction towards the collaborative learning in the workshops. The findings also suggest that to inexperienced teens, the feeling of achievement and ownership of what they are doing can influence their behavioral engagement positively (Buf-fum et al., 2016).

How authorized persons such as teachers interact with the teens affects teens' behavioral engagement (Hubwieser et al., 2015). Positive teacher-student interactions where teachers challenge and interest students result in rich learning opportunities (Gregory et al., 2014; Lykke et al., 2015). Teacher instructions that require analysis and interference of the student can also result in greater behavioral engagement, compared to tasks that require basic skills and rote learning from the student (Downer et al., 2007). In addition to this, feedback from teachers and other peers has the ability to raise teens' self-efficacy by teaching them to improve their developing knowledge, skill, and effort, and further avoid discouragement caused by poor performances or lack of abilities (Pianta et al., 2012; Gregory et al., 2014). High-quality feedback includes extended feedback loops, extra information about the topic under discussion, and exchanges of thoughts and knowledge.

#### **5.3.2 Emotional engagement**

Academic engagement is related to teens' experienced emotions when carrying out work associated with school. In this study, many participants expressed that they enjoy the out of school context. Mainly doing something that is somewhat unusual in academic settings is meaningful. Instead of looking at someone demonstrate, the teens have been able to create some-



thing in collaboration with others by experimenting with different ideas, improvise, test hypothetical situations, and solve problems, as suggested by Riga and Chronopoulou (2014). This implies that teens like to self-explore the opportunities and capabilities of both the environment and themselves.

This study suggests that teens appreciate setting personal goals and choose how challenging their learning processes should be. This supports what research has found about enabling teens to set their own goals will raise their eagerness to set even higher goals, make them put more effort into achieving these goals, and consequently make them persistent on meeting those goals (Bandura and Cervone, 1986; Caraway et al., 2003).

Although the overall achievements of a team seem to be more critical than individual efforts, team members who are less involved with programming recognize that they could have done more. This awareness can tell us something about how inexperienced teens react to challenges or how they value themselves. Many of these teens express that the games would never be as good if they created them alone. Their solution in these situations is failure avoidance and letting their perceived competent partners have the lead roles, and code all or the majority of the game (Lykke et al., 2015). Instead of focusing on what they manage to create, inexperienced teens tend to focus more on what they cannot do, and all the struggles they encounter. Research has found that students with a fear of failures tend to demonstrate less in school-related tasks (Caraway et al., 2003). This finding supports that perceptions of competence contribute to students' achievements (Boggiano et al., 1992). By recognizing that teens have different goals and different competence levels, researchers and teachers should allow teens to choose between a collection of tasks with varying degrees of complexity to avoid decrements of academic engagement. There should also be a balance between effort and outcomes, so the tasks are not unnecessarily challenging. In addition to this, affective engagement towards more inexperienced and hesitant teens can contribute to building up their confidence.

### **5.3.3 Cognitive engagement**

Cognitive engagement concerns teens' reflections, strategic use, own experiences, connections, and willingness for academic work. By looking at indications of competence, we get a clue of the levels the teens can reach during these kinds of workshops. The findings of this study imply that despite prior experiences, almost every teen acquire new knowledge from participating in such activities, and that they have fun in creating something related to what they care about (Appleton et al., 2008).

The study also indicates that Scratch has the ability to lower the floor of programming to something teens can manipulate and understand (Brennan and Resnick, 2012; Maloney et al., 2010). Several participants feel impressed by the environment and its intuitive structure. This research also suggests that Scratch and the creative aspects are found to increase teens' future intentions (Giannakos and Jaccheri, 2013). Many teens agree that more knowledge and competence with Scratch can result in increased involvement and better end results, since more insight will give the teens a better idea of the opportunities and limitations of Scratch. For inexperienced participants, the planning and starting to program are found challenging because they do not know where to start (Lykke et al., 2015). This unsureness has some effect on their engagement since it takes more time for them to begin, which can result in frustrations when they do not meet their goals. However, more knowledge and experience are found to motivate teens to want to learn more in the future.

On the other side, participants with experience in textual-based programming express that Scratch can be difficult because of its simplicity. These teens are unable to write and specify their own code in the environment, which makes them say that Scratch is easier to learn for a novice with no experience with textual-based programming. In addition to this, they express that Scratch has the ability to raise inexperienced teens' motivation

for learning to program. While the findings can imply that the ceilings of Scratch are not raised enough for teens with textual-based programming experiences, it can be an adequate gateway for inexperienced teens to programming.

As discussed, teams seem to value their final products more than individual efforts. For some of the teams, this study detects a willingness to finish the games, and an unwillingness to learn as much as possible. Some teams focus more on creating an exciting game for their friends to try, so they copy student assistants' code and ask for further help instead of working on their own. This implies that there is some competitiveness among teens which cause them to cut corners at the expense of their learning. The research also suggests that this was not the case for all teams, as the majority try on their own by looking for solutions in paper tutorials and asking other teams for help (Bandura and Cervone, 1986). In general, the availability of assistance is considered a positive measure to ease frustrations within the teams. To ensure that every teen actively learns something while receiving help, teachers should get more skilled at facilitating student involvement and promote exchanges of thoughts and knowledge between the two parties (Pianta et al., 2012).

### **5.3.4 Agentic engagement**

Agentic engagement is present when students influence their instruction and learning experience for instance by communicating their needs and thoughts. Teens' self-reports of their own active contributions support that many contributed to their learning experiences. The findings also reveal that a high number of teens learn new skills and knowledge from others. This implies that teens who share with others can influence their peers to do the same (Buffum et al., 2016; An, 2016).

In addition to making individual contributions, having personal values that align with the activities are vital for teens (Reeve and Tseng, 2011). This study implies that when teens make the activities personally relevant, challenging, or need satisfying, they become more agenticly engaged, and prone to sharing expressions and thoughts with their peers. Especially teams with members who have the same personal values seem to be agenticly engaged (Furrer and Skinner, 2003).

This research also suggests that teachers are responsible for creating motivationally enriching classroom conditions for teens to maintain an agentic engagement (Reeve, 2013). Teachers should also provide teens insights and autonomy to spark curiosity in challenging tasks rather than easy ones (Deci et al., 1981).

## **5.4 Implications and recommendations**

This study points out how several factors can affect teens' academic engagement in computational thinking activities, and how they target the presented types of academic engagement. Some factors are dependent on teens' internal factors such as competence, while others are dependent on external factors such as learning outcomes and collaborations with others.

### **5.4.1 Generalizability and applicability**

Many results from this study are comparable to existing research, and the results can contribute to a better awareness when it comes to raising teens' academic engagement in computational thinking activities. According to Fields et al. (2014), there is a need for investigating the quality of engagement among teens to deepen their participation in computing, and this study's findings can supply more insight in this field. Note that the discoveries cannot be applied precisely to other visual programming environments, samples, or computational thinking activities and produce the exact results. Nonetheless, this study can be considered a source of measures which contribute to more engaging computational thinking activities for inexperienced students, and thus decrease disinterest among them. Researchers and lecturers who plan on implementing similar activities in or outside of school can undoubtedly benefit from this information. Figure 5.2 outlines the generalizable factors which were found to affect the four engagement types in this study.

<p><b>Behavioral</b></p> <ul style="list-style-type: none"> <li>● Creativity</li> <li>● Equal division of tasks between team members</li> <li>● Room for personalization</li> <li>● Validation from others</li> </ul>	<p><b>Emotional</b></p> <ul style="list-style-type: none"> <li>● Building on existing knowledge</li> <li>● Effort vs. outcome</li> <li>● Meeting goals</li> <li>● Room for self-exploration</li> </ul>
<ul style="list-style-type: none"> <li>● Alignment of tasks and own values</li> <li>● Encouragement of expression and thought</li> <li>● Exchanges of knowledge</li> </ul> <p><b>Agentic</b></p>	<ul style="list-style-type: none"> <li>● Alignment of tasks and future goals</li> <li>● Competence vs. tasks</li> <li>● New knowledge</li> <li>● Availability of assistance</li> </ul> <p><b>Cognitive</b></p>

**Figure 5.2:** The bullet points represent factors that affect academic engagement

This study indicates that inner factors such as competence, motivation, intentions, knowledge, and personal values affect teens’ engagement for academic work. In many cases, teens who doubt their competence and knowledge limit their capabilities. On the other side, outer factors such as interactions with peers and teachers, openness for expression and thought, learning outcomes, and opportunities of creating creative artifacts, self-exploration, and personalization can engage teens even further. Allowing teens to perform and achieve despite their lack of experience can excite them in a way that they want to learn more. The findings of this study notably point out that access to other people has been positive and indispensable, either if it concerns creating something with or for others. Teens explain that they

are more capable when they collaborate with others, either if they ask other students or teachers for help, have critical discussions, or by interacting with other teams' regarding their code. By creating for others, teens feel a sense of reward when peers or teachers try or comment on their games, regardless of it being for educational, engagement, or entertaining purposes. They experience satisfaction and empowerment from creating something that others find fascinating and likable, which can result in cutting corners to create the best game. Collaborative learning is therefore proved to be meaningful to teens, as they enjoy sharing their knowledge and learning from and with others. These results ultimately support the rise of more sociological and cultural centered reasons for learning technology and programming, and therefore the growth of computational participation (Kafai and Burke, 2013).

The research suggests that the potentials of programming environments can expand what students think that they can achieve (Resnick and Silverman, 2005). This study used Scratch, but several visual programming environments are available for use to fit the objectives of this study. Scratch is an environment with the ability to increase interest for teens with low motivation for learning programming. Interactive manipulations in Scratch can enable teens to make connections between code blocks and results. The findings imply that visual programming is adequate for introducing students to computational thinking concepts such as building algorithms, problem-solving, simulation, and collaboration (Kelleher et al., 2007; Maloney et al., 2010). The results of this study point to a direction where visual programming offers novice users a simplified pathway to programming and computational thinking. How Scratch is used in activities can however result in different learning outcomes. Teachers can for instance use it to demonstrate individual Computer Science concepts such as loops or larger projects where teens need to use mixtures of several concepts.

The opportunities of the visual programming environment can however be adjusted to the composition of teens.

This study reveals that the engagement and appeal for learning computational thinking are not only dependent on competence, teachers, perceptions, and existing knowledge. It is additionally influenced by the extent teens are able to create with and for others in a creative, personalized, and self-exploring way, along with meeting own values, and short-term or long-term goals. The interest in computational thinking activities can further improve by adjusting the activities to something teens find valuable and stimulating. The findings also indicate that visual programming languages can introduce teens to computational thinking and change their negative attitudes and misperceptions towards the discipline. To conclude, the generalizable factors and overall findings of this study are transferable.

### **5.4.2 Limitations**

Readers should have in mind that there are limitations to this study. First and foremost, the generalizability of the results must be approached carefully since the study was conducted in a specific context (e.g., age, participants, experience). Secondly, the time with the participants was very limited, and instead of focusing on long-term learning and transfer effects, there was an emphasis on affective and immediate effects. Only indications of teens' performance were collected in this study, so a definite conclusion about the computational thinking competence of the sample cannot be made. However, the results from this study catch some informative insights about how the teens performed, which can support the speculations.

During the artifact collection, the researchers had in mind to save different versions of each game. Due to technical problems and human flaws, we ended up having inconsistent numbers of savings at each workshop.



The student assistants were also in charge of uploading the game versions after each workshop, but occasionally versions were lost during this process.

The response bias is something that can affect the results. In Kodeløypa, school classes applied to participate in the program, which can imply that the schools were more open to technology than others. Additionally, some of the participating classes were elective technology classes. Participants from these classes are commonly more interested in technology than the average teen, and can therefore express a more positive attitude towards the given activities. In the Tappetina workshop, the participants chose to attend the workshop voluntarily. Their motivation to learn and attitudes towards computing might also differ from other teens.

Another limitation is the self-report data that are vulnerable to exaggeration and memory issues. The results of this study relied on the authenticity and truthfulness of what the participants stated. In addition to this, results from this study were mainly correlational, and can therefore be speculative to an extent.

Lastly, this study did not take into account the participants' performance in academics, their developmental levels, their learning skills, their intentions, nor their own expectations for themselves. These are individual factors that can be important for how engaged a participant is.

### **5.4.3 Future research**

This study gives more insight about how inexperienced teens perform in terms of computational thinking skills, and how we can increase academic engagement in computational thinking activities with programming. A recommendation for future work is to address the limitations of this study and see if the factors apply to other computational thinking activities with or without programming. Furthermore, each engagement type and factor that affect the academic engagement can be examined further for increased awareness of how effective they are, and how much they affect academic engagement. Testing of the factors over a more extended period in similar activities can be done to collect knowledge about possible long-term effects. Another suggestion for further work is to investigate the differences in academic engagement between dyads and triads in computational thinking activities. In addition to this, addressing how efficient the factors are for a more diverse and larger sample can be done for further insight. Another suggestion is to consider different contexts, for instance activities in school, genders, and ages.

## **5.5 Closing words**

Computational thinking can prepare teens for the future by developing and enhancing unique thinking skills, while academic engagement can deepen motivation and make students more personally involved in academic work. Researchers have found that programming can promote the development of computational thinking, so the choice of this research has been the successful visual programming environment Scratch. This study has had in goal to detect factors that are academically engaging to teens in computational thinking activities. With observations, artifacts, and self-reports

through interviews and questionnaires, this research indicates that effective and active collaborations are essential and uplifting for the teens. Teens enjoy working together and ending up with a shared product. They seek for validation and assistance from their peers and teachers. How teachers interact with their students is also proven to have a significant impact on students' engagement and willingness to participate. Teachers should therefore provide tasks and activities to maintain, test, and improve student capabilities, in addition to encourage expressions and thoughts, so that students can experiment and test alternative ideas. This study also reveals that teens appreciate having the opportunity to personalize, explore, build on existing knowledge, gain new experience, set goals, and make connections between their personal values and the associated tasks. In addition to this, this study provides a mapping of the factors and how they affect the different constructs of academic engagement, namely the behavioral, emotional, cognitive, or agentic engagement. This research provides additional awareness about teens' quality of engagement in computational thinking activities, which can contribute to deepen participation among teens.

For better or worse, technology's emerging position in our society will affect humans somehow. Regardless of what we think or feel about it, we need to decide how we will interact with it. Will we use technology solely as a tool to simplify processes in our lives, or will we use it to extend our current capabilities? Will we go from consumers to tool builders? Also, will we let the technology dictate us, or will we dictate it? The choice is clear. We need to engage teens to understand that they can benefit more from technology than they already are, and educational systems should certainly commit to promoting this through the development of computational thinking skills. Technology is here to stay, so let us have some fun and embrace it!

# Bibliography

Ackermann, E., 2001. Piaget's constructivism, papert's constructionism: What's the difference? In: CONSTRUCTIVISM: USES AND PERSPECTIVES IN EDUCATION, VOLUMES 1 2). CONFERENCE PROCEEDINGS, GENEVA: RESEARCH CENTER IN EDUCATION/ CAHIER 8 / SEPTEMBER 01. PP. pp. 85–94.

Adams, J. C., Webster, A. R., 2012. What do students learn about programming from game, music video, and storytelling projects? In: Proceedings of the 43rd ACM Technical Symposium on Computer Science Education. SIGCSE '12. ACM, New York, NY, USA, pp. 643–648.

URL <http://doi.acm.org/10.1145/2157136.2157319>

Aho, A. V., Jul. 2012. Computation and computational thinking. *Comput. J.* 55 (7), 832–835.

URL <http://dx.doi.org/10.1093/comjnl/bxs074>

Allen, I. E., Seaman, C. A., 07 2007. Likert scales and data analyses. *Quality Progress* 40 (7), 64–65, copyright - Copyright American Society for Quality Jul 2007; Document feature - Tables; ; Illustrations; Last updated - 2014-05-21; CODEN - QUPRB3; SubjectsTermNotLitGenreText - United States–US.

---

URL <https://search.proquest.com/docview/214764202?accountid=12870>

An, Y.-J., 2016. A case study of educational computer game design by middle school students. *Educational Technology Research and Development* 64 (4), 555–571.

Appleton, J. J., Christenson, S. L., Furlong, M. J., 2008. Student engagement with school: Critical conceptual and methodological issues of the construct. *Psychology in the Schools* 45 (5), 369–386.

Armoni, M., Meerbaum-Salant, O., Ben-Ari, M., Feb. 2015. From scratch to “real” programming. *Trans. Comput. Educ.* 14 (4), 25:1–25:15.

URL <http://doi.acm.org/10.1145/2677087>

Attride-Stirling, J., 2001. Thematic networks: an analytic tool for qualitative research. *Qualitative research* 1 (3), 385–405.

Bandura, A., Cervone, D., 1986. Differential engagement of self-reactive influences in cognitive motivation. *Organizational behavior and human decision processes* 38 (1), 92–113.

Barr, D., Harrison, J., Conery, L., 2011. Computational thinking: A digital age skill for everyone. *Learning & Leading with Technology* 38 (6), 20–23.

Barr, V., Stephenson, C., 2011. Bringing computational thinking to k-12: What is involved and what is the role of the computer science education community? *acm Inroads* 2 (1), 1–7.

Basawapatna, A. R., Koh, K. H., Repenning, A., 2010. Using scalable game design to teach computer science from middle school to graduate school. In: *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education*. ACM, pp. 224–228.

- 
- Beauchamp, G., 2016. *Computing and ICT in the Primary School: From pedagogy to practice 2*. Routledge.
- Ben-Ari, M., Mar. 1998. Constructivism in computer science education. *SIGCSE Bull.* 30 (1), 257–261.  
URL <http://doi.acm.org/10.1145/274790.274308>
- Bers, M. U., Ponte, I., Juelich, C., Viera, A., Schenker, J., 2002. Teachers as designers: Integrating robotics in early childhood education. *Information Technology in Childhood Education Annual 2002* (1), 123–145.  
URL <https://www.learntechlib.org/p/8850>
- Bircan, H., Sungur, S., 2016. The role of motivation and cognitive engagement in science achievement. *Science Education International* 27 (4), 509–529.
- Boggiano, A. K., Shields, A., Barrett, M., Kellam, T., Thompson, E., Simons, J., Katz, P., Sep 1992. Helplessness deficits in students: The role of motivational orientation. *Motivation and Emotion* 16 (3), 271–296.  
URL <https://doi.org/10.1007/BF00991655>
- Boyatzis, R. E., 1998. *Transforming qualitative information: Thematic analysis and code development*. sage.
- Brennan, K., Resnick, M., 2012. New frameworks for studying and assessing the development of computational thinking. In: *Using artifact-based interviews to study the development of computational thinking in interactive media design*.
- Brown, N. C. C., Kölling, M., Crick, T., Peyton Jones, S., Humphreys, S., Sentance, S., 2013. Bringing computer science back into schools: Lessons from the uk. In: *Proceeding of the 44th ACM Technical Symposium on Computer Science Education. SIGCSE '13*. ACM, New York, NY, USA, pp. 269–274.  
URL <http://doi.acm.org/10.1145/2445196.2445277>
-

- 
- Buffum, P. S., Frankosky, M., Boyer, K. E., Wiebe, E. N., Mott, B. W., Lester, J. C., 2016. Collaboration and gender equity in game-based learning for middle school computer science. *Computing in Science & Engineering* 18 (2), 18–28.
- Burke, Q., Kafai, Y. B., 2012. The writers' workshop for youth programmers: digital storytelling with scratch in middle school classrooms. In: *Proceedings of the 43rd ACM technical symposium on Computer Science Education*. ACM, pp. 433–438.
- Çakır, N. A., Gass, A., Foster, A., Lee, F. J., 2017. Development of a game-design workshop to promote young girls' interest towards computing through identity exploration. *Computers & Education* 108, 115–130.
- Caraway, K., Tucker, C. M., Reinke, W. M., Hall, C., 2003. Self-efficacy, goal orientation, and fear of failure as predictors of school engagement in high school students. *Psychology in the Schools* 40 (4), 417–427.
- Corbin, J., Strauss, A., Strauss, A. L., 2014. *Basics of qualitative research*. Sage.
- Council, N. R., 2010. *Report of a Workshop on the Scope and Nature of Computational Thinking*. The National Academies Press, Washington, DC.
- Creswell, J. W., 2014. *A concise introduction to mixed methods research*. Sage Publications.
- Creswell, J. W., Hanson, W. E., Clark Plano, V. L., Morales, A., 2007. Qualitative research designs: Selection and implementation. *The counseling psychologist* 35 (2), 236–264.
- Creswell, J. W., Miller, D. L., 2000. Determining validity in qualitative inquiry. *Theory into practice* 39 (3), 124–130.
- Deci, E. L., Nezlek, J., Sheinman, L., 1981. Characteristics of the rewarder

- 
- and intrinsic motivation of the rewarder. *Journal of personality and social psychology* 40 (1), 1.
- Deci, E. L., Vallerand, R. J., Pelletier, L. G., Ryan, R. M., 1991. Motivation and education: The self-determination perspective. *Educational Psychologist* 26 (3-4), 325–346.  
URL <http://dx.doi.org/10.1080/00461520.1991.9653137>
- Denner, J., Werner, L., Ortiz, E., 2012. Computer games created by middle school girls: Can they be used to measure understanding of computer science concepts? *Computers Education* 58 (1), 240 – 249.  
URL <http://www.sciencedirect.com/science/article/pii/S0360131511001849>
- DfE, 2013. National curriculum in england: computing programmes of study. Accessed: 2018-04-30.  
URL <https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study>
- Downer, J. T., Rimm-Kaufman, S., Pianta, R. C., 09 2007. How do classroom conditions and children’s risk for school problems contribute to children’s behavioral engagement in learning? *School Psychology Review* 36 (3), 413–432, copyright - Copyright National Association of School Psychologists Sep 2007; Document feature - Tables; ; Last updated - 2014-05-18.  
URL <https://search.proquest.com/docview/219646022?accountid=12870>
- Ertmer, P. A., Newby, T. J., 2013. Behaviorism, cognitivism, constructivism: Comparing critical features from an instructional design perspective. *Performance Improvement Quarterly* 26 (2), 43–71.



---

Fields, D. A., Giang, M., Kafai, Y., 2014. Programming in the wild: Trends in youth computational participation in the online scratch community. In: Proceedings of the 9th Workshop in Primary and Secondary Computing Education. WiPSCE '14. ACM, New York, NY, USA, pp. 2–11.

URL <http://doi.acm.org/10.1145/2670757.2670768>

Fields, D. A., Quirke, L., Amely, J., Maughan, J., 2016. Combining big data and thick data analyses for understanding youth learning trajectories in a summer coding camp. In: Proceedings of the 47th ACM Technical Symposium on Computing Science Education. SIGCSE '16. ACM, New York, NY, USA, pp. 150–155.

URL <http://doi.acm.org/10.1145/2839509.2844631>

Franklin, D., Conrad, P., Boe, B., Nilsen, K., Hill, C., Len, M., Dreschler, G., Aldana, G., Almeida-Tanaka, P., Kiefer, B., Laird, C., Lopez, F., Pham, C., Suarez, J., Waite, R., 2013. Assessment of computer science learning in a scratch-based outreach program. In: Proceeding of the 44th ACM Technical Symposium on Computer Science Education. SIGCSE '13. ACM, New York, NY, USA, pp. 371–376.

URL <http://doi.acm.org/10.1145/2445196.2445304>

Fredricks, J. A., Blumenfeld, P. C., Paris, A. H., 2004. School engagement: Potential of the concept, state of the evidence. *Review of Educational Research* 74 (1), 59–109.

URL <https://doi.org/10.3102/00346543074001059>

Fredricks, J. A., McColskey, W., 2012. The measurement of student engagement: A comparative analysis of various methods and student self-report instruments. In: *Handbook of research on student engagement*. Springer, pp. 763–782.

Furrer, C., Skinner, E., 2003. Sense of relatedness as a factor in children's

---

academic engagement and performance. *Journal of educational psychology* 95 (1), 148.

Garneli, V., Giannakos, M. N., Chorianopoulos, K., Jaccheri, L., 2015. Serious game development as a creative learning experience: Lessons learnt. In: *Proceedings of the Fourth International Workshop on Games and Software Engineering. GAS '15*. IEEE Press, Piscataway, NJ, USA, pp. 36–42.

URL <http://dl.acm.org/citation.cfm?id=2820144.2820155>

Giannakos, M. N., Jaccheri, L., 2013. Designing creative activities for children: The importance of collaboration and the threat of losing control. In: *Proceedings of the 12th International Conference on Interaction Design and Children. IDC '13*. ACM, New York, NY, USA, pp. 336–339.

URL <http://doi.acm.org/10.1145/2485760.2485827>

Giannakos, M. N., Jaccheri, L., Proto, R., 2013. Teaching computer science to young children through creativity: Lessons learned from the case of norway. In: *Proceedings of the 3rd Computer Science Education Research Conference on Computer Science Education Research. CSERC '13*. Open Universiteit, Heerlen, Open Univ., Heerlen, The Netherlands, The Netherlands, pp. 10:103–10:111.

URL <http://dl.acm.org/citation.cfm?id=2541917.2541927>

Golafshani, N., 2003. Understanding reliability and validity in qualitative research. *The qualitative report* 8 (4), 597–606.

Gregory, A., Allen, J. P., Mikami, A. Y., Hafen, C. A., Pianta, R. C., 2014. Effects of a professional development program on behavioral engagement of students in middle and high school. *Psychology in the Schools* 51 (2), 143–163.

---

Grover, S., Cooper, S., Pea, R., 2014a. Assessing computational learning in k-12. In: Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education. ITiCSE '14. ACM, New York, NY, USA, pp. 57–62.

URL <http://doi.acm.org/10.1145/2591708.2591713>

Grover, S., Pea, R., 2013. Computational thinking in k–12: A review of the state of the field. *Educational Researcher* 42 (1), 38–43.

Grover, S., Pea, R., Cooper, S., 2014b. Remedying misperceptions of computer science among middle school students. In: Proceedings of the 45th ACM technical symposium on Computer science education. ACM, pp. 343–348.

Han, A., Kim, J., Wohn, K., 2015. Entry: Visual programming to enhance children’s computational thinking. In: Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers. UbiComp/ISWC’15 Adjunct. ACM, New York, NY, USA, pp. 73–76.

URL <http://doi.acm.org/10.1145/2800835.2800871>

Hanson, W. E., Creswell, J. W., Clark, V. L. P., Petska, K. S., Creswell, J. D., 2005. Mixed methods research designs in counseling psychology. *Journal of counseling psychology* 52 (2), 224.

Hennessey, B. A., 2000. Self-determination theory and the social psychology of creativity. *Psychological Inquiry* 11 (4), 293–298.

URL <http://www.jstor.org/stable/1449624>

Hsieh, H.-F., Shannon, S. E., 2005. Three approaches to qualitative content analysis. *Qualitative health research* 15 (9), 1277–1288.

Hubwieser, P., Giannakos, M. N., Berges, M., Brinda, T., Diethelm, I., Magenheimer, J., Pal, Y., Jackova, J., Jasute, E., 2015. A global snapshot

---

of computer science education in k-12 schools. In: Proceedings of the 2015 ITiCSE on Working Group Reports. ACM, pp. 65–83.

Jaccheri, L., Jun 2017. The little doorman.

URL <https://letiziajaccheri.org/the-little-doorman/>

Jackson, L. A., Witt, E. A., Games, A. I., Fitzgerald, H. E., von Eye, A., Zhao, Y., 2012. Information technology use and creativity: Findings from the children and technology project. *Computers in human behavior* 28 (2), 370–376.

Johnson, B., Christensen, L., 2008. Educational research: Quantitative, qualitative, and mixed approaches. Sage.

Johnson, R. B., Onwuegbuzie, A. J., Turner, L. A., 2007. Toward a definition of mixed methods research. *Journal of mixed methods research* 1 (2), 112–133.

Juvonen, J., Espinoza, G., Knifsend, C., 2012. The Role of Peer Relationships in Student Academic and Extracurricular Engagement. Springer US, Boston, MA, pp. 387–401.

URL [https://doi.org/10.1007/978-1-4614-2018-7\\_18](https://doi.org/10.1007/978-1-4614-2018-7_18)

Kafai, Y. B., 2006. Playing and making games for learning: Instructionist and constructionist perspectives for game studies. *Games and culture* 1 (1), 36–40.

Kafai, Y. B., Burke, Q., 2013. The social turn in k-12 programming: Moving from computational thinking to computational participation. In: Proceeding of the 44th ACM Technical Symposium on Computer Science Education. SIGCSE '13. ACM, New York, NY, USA, pp. 603–608.

URL <http://doi.acm.org/10.1145/2445196.2445373>

- 
- Kafai, Y. B., Resnick, M., 1996. *Constructionism in practice: Designing, thinking, and learning in a digital world*. Routledge.
- Kalelioğlu, F., Gülbahar, Y., 2014. The effects of teaching programming via scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education* 13 (1), 33–50.
- Kelleher, C., Pausch, R., Kiesler, S., 2007. Storytelling alice motivates middle school girls to learn computer programming. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. CHI '07*. ACM, New York, NY, USA, pp. 1455–1464.  
URL <http://doi.acm.org/10.1145/1240624.1240844>
- Kim, K. H., 2011. The creativity crisis: The decrease in creative thinking scores on the torrance tests of creative thinking. *Creativity Research Journal* 23 (4), 285–295.  
URL <https://doi.org/10.1080/10400419.2011.627805>
- Koca, F., 2016. Motivation to learn and teacher–student relationship. *Journal of International Education and Leadership* Volume 6 (2), 1–20.
- Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J., Werner, L., Feb. 2011. Computational thinking for youth in practice. *ACM Inroads* 2 (1), 32–37.  
URL <http://doi.acm.org/10.1145/1929887.1929902>
- Lorås, M., 2017. Let the gamification begin!-a qualitative case study of student experiences in the gamified learning environment heimdall's quest. Master's thesis, NTNU.
- Lu, J. J., Fletcher, G. H., 2009. Thinking about computational thinking. *ACM SIGCSE Bulletin* 41 (1), 260–264.
- Lye, S., Koh, J., 12 2014. Review on teaching and learning of computational thinking through programming: What is next for k-12? 41, 51–61.

---

Lykke, M., Coto, M., Jantzen, C., Mora, S., Vandell, N., 2015. Motivating students through positive learning experiences: A comparison of three learning designs for computer programming courses. *Journal of Problem Based Learning in Higher Education* 3 (2), 80–108.

Maloney, J., Resnick, M., Rusk, N., Silverman, B., Eastmond, E., Nov. 2010. The scratch programming language and environment. *Trans. Comput. Educ.* 10 (4), 16:1–16:15.

URL <http://doi.acm.org/10.1145/1868358.1868363>

Maloney, J. H., Peppler, K., Kafai, Y., Resnick, M., Rusk, N., Mar. 2008. Programming by choice: Urban youth learning programming with scratch. *SIGCSE Bull.* 40 (1), 367–371.

URL <http://doi.acm.org/10.1145/1352322.1352260>

Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., Settle, A., 2014. Computational thinking in k-9 education. In: *Proceedings of the Working Group Reports of the 2014 on Innovation & Technology in Computer Science Education Conference. ITiCSE-WGR '14*. ACM, New York, NY, USA, pp. 1–29.

URL <http://doi.acm.org/10.1145/2713609.2713610>

Mays, N., Pope, C., 1995. Qualitative research: Observational methods in health care settings. *BMJ: British Medical Journal* 311 (6998), 182.

Meerbaum-Salant, O., Armoni, M., Ben-Ari, M. M., 2010. Learning computer science concepts with scratch. In: *Proceedings of the Sixth International Workshop on Computing Education Research. ICER '10*. ACM, New York, NY, USA, pp. 69–76.

URL <http://doi.acm.org/10.1145/1839594.1839607>

Millians, M., 2011. *Computational Skills*. Springer US, Boston, MA, pp. 396–396.

---

URL [https://doi.org/10.1007/978-0-387-79061-9\\_645](https://doi.org/10.1007/978-0-387-79061-9_645)

Mishra, P., Yadav, A., Group, D.-P. R., et al., 2013. Rethinking technology & creativity in the 21st century. *TechTrends* 57 (3), 10–14.

Mldlan, D. K., P. R., et al., 1986. A study of the development of programming ability and thinking skills in high school students. *Journal of Educational Computing Research* 2 (4), 429–458.

Moreno-León, J., Robles, G., Román-González, M., 2015. Dr. scratch: Automatic analysis of scratch projects to assess and foster computational thinking. *RED. Revista de Educación a Distancia* (46), 1–23.

Moreno-León, J., Román-González, M., Hartevelde, C., Robles, G., 2017. On the automatic assessment of computational thinking skills: A comparison with human experts. In: *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. ACM, pp. 2788–2795.

Oates, B. J., 2006. *Researching Information Systems and Computing*. Sage Publications Ltd.

Papavlasopoulou, S., 2016. Engaging students with computer science through creativity: Toward better understanding and improved methods. In: *Advanced Learning Technologies (ICALT), 2016 IEEE 16th International Conference on*. IEEE, pp. 546–548.

Papavlasopoulou, S., Giannakos, M. N., Jaccheri, L., 2015. Designing creative programming experiences for 15 years old students. In: *Workshop of Making as a Pathway to Foster Joyful Engagement and Creativity in Learning (Make2Learn)*. p. 37.

Papavlasopoulou, S., Sharma, K., Giannakos, M., Jaccheri, L., 2017. Using eye-tracking to unveil differences between kids and teens in coding

- 
- activities. In: Proceedings of the 2017 Conference on Interaction Design and Children. IDC '17. ACM, New York, NY, USA, pp. 171–181.  
URL <http://doi.acm.org/10.1145/3078072.3079740>
- Papert, S., 1980. *Mindstorms: Children, Computers, and Powerful Ideas*. Basic Books, Inc., New York, NY, USA.
- Papert, S., Harel, I., 1991. Situating constructionism. *Constructionism* 36 (2), 1–11.
- Park, S., Holloway, S. D., Arendtsz, A., Bempechat, J., Li, J., 2012. What makes students engaged in learning? a time-use study of within- and between-individual predictors of emotional engagement in low-performing high schools. *Journal of youth and adolescence* 41 (3), 390–401.
- Pianta, R. C., Hamre, B. K., Allen, J. P., 2012. Teacher-student relationships and engagement: Conceptualizing, measuring, and improving the capacity of classroom interactions. In: *Handbook of research on student engagement*. Springer, pp. 365–386.
- Polkinghorne, D. E., 2005. Language and meaning: Data collection in qualitative research. *Journal of counseling psychology* 52 (2), 137.
- Rattray, J., Jones, M. C., 2007. Essential elements of questionnaire design and development. *Journal of clinical nursing* 16 (2), 234–243.
- Reeve, J., 2012. A self-determination theory perspective on student engagement. In: *Handbook of research on student engagement*. Springer, pp. 149–172.
- Reeve, J., 08 2013. How students create motivationally supportive learning environments for themselves: The concept of agentic engagement 105, 579.
- Reeve, J., Tseng, C.-M., 2011. Agency as a fourth aspect of students' en-
-



- 
- agement during learning activities. *Contemporary Educational Psychology* 36 (4), 257–267.
- Repenning, A., Webb, D., Ioannidou, A., 2010. Scalable game design and the development of a checklist for getting computational thinking into public schools. In: *Proceedings of the 41st ACM Technical Symposium on Computer Science Education. SIGCSE '10*. ACM, New York, NY, USA, pp. 265–269.  
URL <http://doi.acm.org/10.1145/1734263.1734357>
- Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., Kafai, Y., Nov. 2009. Scratch: Programming for all. *Commun. ACM* 52 (11), 60–67.  
URL <http://doi.acm.org/10.1145/1592761.1592779>
- Resnick, M., Silverman, B., 2005. Some reflections on designing construction kits for kids. In: *Proceedings of the 2005 Conference on Interaction Design and Children. IDC '05*. ACM, New York, NY, USA, pp. 117–122.  
URL <http://doi.acm.org/10.1145/1109540.1109556>
- Rich, P. J., Hodges, C. B., 2017. *Emerging Research, Practice, and Policy on Computational Thinking*, 1st Edition. Springer Publishing Company, Incorporated.
- Riga, V., Chronopoulou, E., 2014. Applying mackinnon's 4ps to foster creative thinking and creative behaviours in kindergarten children. *Education 3-13* 42 (3), 330–345.  
URL <https://doi.org/10.1080/03004279.2012.692700>
- Roberts, P., Priest, H., Traynor, M., Jul 2006. Reliability and validity in research. *Nursing Standard* (through 2013) 20 (44), 41–5, copyright - Copyright RCN Publishing Company Ltd. Jul 12-Jul 18, 2006; Last

---

updated - 2016-04-30; CODEN - NSTAEU.

URL <https://search.proquest.com/docview/219850149?accountid=12870>

Robson, C., McCartan, K., 2016. Real world research. John Wiley & Sons.

Runco, M. A., Acar, S., Cayirdag, N., 2017. A closer look at the creativity gap and why students are less creative at school than outside of school. *Thinking Skills and Creativity* 24 (Supplement C), 242 – 249.

URL <http://www.sciencedirect.com/science/article/pii/S1871187116301481>

Saldaña, J., 2015. The coding manual for qualitative researchers. Sage.

Sandelowski, M., 2000. Combining qualitative and quantitative sampling, data collection, and analysis techniques in mixed-method studies. *Research in nursing & health* 23 (3), 246–255.

Shell, D. F., Hazley, M. P., Soh, L.-K., Miller, L. D., Chiriacescu, V., Ingraham, E., 2014. Improving learning of computational thinking using computational creativity exercises in a college csi computer science course for engineers. In: *Frontiers in Education Conference (FIE)*, 2014 IEEE. IEEE, pp. 1–7.

Siemens, G., 2014. *Connectivism: A learning theory for the digital age*.

Skinner, E. A., Belmont, M. J., 1993. Motivation in the classroom: Reciprocal effects of teacher behavior and student engagement across the school year. *Journal of educational psychology* 85 (4), 571.

Soh, K., 2017. Fostering student creativity through teacher behaviors. *Thinking Skills and Creativity* 23 (Supplement C), 58 – 66.

URL <http://www.sciencedirect.com/science/article/pii/S1871187116301584>

---

Thanapornsanguth, S., Holbert, N., 06 2017. Bots for tots: Girls' perceived versus actual competency in technology and making.

Turner, J. C., Christensen, A., Kackar-Cam, H. Z., Trucano, M., Fulmer, S. M., 2014. Enhancing students' engagement: Report of a 3-year intervention with middle school teachers. *American Educational Research Journal* 51 (6), 1195–1226.

URL <https://doi.org/10.3102/0002831214532515>

Utdanningsdirektoratet, 2016. Forsøkslæreplan i valgfag programmering (prg1-01).

URL <https://www.udir.no/kl06/PRG1-01/>

Voogt, J., Fisser, P., Good, J., Mishra, P., Yadav, A., 2015. Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies* 20 (4), 715–728.

Vygotsky, L. S., 1990. Imagination and creativity in childhood. *Soviet Psychology* 28 (1), 84–96.

URL <http://www.tandfonline.com/doi/abs/10.2753/RP01061-0405280184>

Weintrop, D., Wilensky, U., 2015. Using commutative assessments to compare conceptual understanding in blocks-based and text-based programs. In: *Proceedings of the Eleventh Annual International Conference on International Computing Education Research. ICER '15*. ACM, New York, NY, USA, pp. 101–110.

URL <http://doi.acm.org/10.1145/2787622.2787721>

Wentzel, K. R., Watkins, D. E., 2002. Peer relationships and collaborative learning as contexts for academic enablers. *School Psychology Review* 31 (3), 366, copyright - Copyright National Association of School Psychologists 2002; Last updated - 2014-05-26.

- 
- URL <https://search.proquest.com/docview/219653970?accountid=12870>
- Werner, L., Denner, J., Campe, S., 2014. Learning, education and games. ETC Press, Pittsburgh, PA, USA, Ch. Using Computer Game Programming to Teach Computational Thinking Skills, pp. 37–53.
- URL <http://dl.acm.org/citation.cfm?id=2811147.2811150>
- Wing, J. M., 2006. Computational thinking 49 (3), 1–3.
- Wing, J. M., 2008. Computational thinking and thinking about computing. Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences 366 (1881), 3717–3725.
- URL <http://rsta.royalsocietypublishing.org/content/366/1881/3717>
- Yadav, A., Hong, H., Stephenson, C., Nov 2016. Computational thinking for all: Pedagogical approaches to embedding 21st century problem solving in k-12 classrooms. TechTrends 60 (6), 565–568.
- URL <https://doi.org/10.1007/s11528-016-0087-7>
- Yazzie-Mintz, E., 2007. Voices of students on engagement: A report on the 2006 high school survey of student engagement. Center for Evaluation and Education Policy, Indiana University.
- Yin, R. K., 1981. The case study as a serious research strategy. Knowledge 3 (1), 97–114.

# Appendix

Appendix A - Consent letter . . . . .	135
Appendix B - Interview questions for Kodeløypa . . . . .	139
Appendix C - Interview questions for Tappetina . . . . .	145
Appendix D - Dr.Scratch competence levels . . . . .	151
Appendix E - Kodeløypa categories for codes . . . . .	153
Appendix F - Tappetina categories for codes . . . . .	166
Appendix G - Themes and belonging code categories . . . . .	173
Appendix H - Results from Kodeløypa questionnaires . . . . .	176
Appendix I - Results from Tappetina questionnaires . . . . .	177

## Consent letter

This consent letter was written and sent out by the professors and researchers responsible for Kodeløypa at the Norwegian University of Science and Technology (NTNU). A variant of this consent letter was sent out to the participants at the Tappetina workshop.

### **Request for participation in research project - "Study in Kodeløypa programming workshop"**

#### **Background and Purpose**

Kodeløypa is a one-day workshop program for students in Trondheim, Norway which takes place at the Department of Computer and Information Science at NTNU. The main goal of this workshop is to motivate children and increase their interest in Computer Science. During this workshop, students are introduced to programming by playfully interacting with robots and creating a game using the Scratch programming tool. This research project

---

aims to evaluate this programming workshop using eye-tracking (e.g., in wearable goggles format) and data of video recordings to understand the learning process in a more profound way. Student's gaze, as in eye movements and pupil dilation, will be recorded while they collaborate within groups of three to complete the programming tasks. The participants in this project will be students from schools in Trondheim, Norway, whose teachers/school have applied to attend the Kodeløypa workshops. Kodeløypa is organized by the Department of Computer and Information Science at NTNU, and these schools have voluntarily signed up to be a part of the research project. Students attend the workshop as part of a school day, and the workshop lasts approximately for five hours. Skolelaboratoriet at NTNU is a resource center for teaching of Science, and they are responsible for sending an open call invitation to schools in Trondheim, Norway to participate at the Kodeløypa workshops. When the schools are selected, the researcher (a Ph.D. student at IDI-NTNU) will contact the schools to get the consent from both the child and the legal guardian. The responsible person for the project will be an Associate Professor at the Department of Computer and Information Science (IDI) at NTNU, Trondheim, Norway, a Ph.D. student at the same institution and a researcher at the Computer-Human Interaction in Learning and Instruction (CHILI) Lab, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, who will visit the department for the purpose of the study (see general information section).

### **What does participation in the project imply?**

For the purpose of the research project, data will be collected using photos, video and audio recordings, observations and paper-based questionnaires. Questions will concern participants' attitudes toward the programming experience at the workshop. Additionally, the project will collect physiological (eye-tracking) data. The duration of the intervention will be accord-

---

ing to the workshop's activities (two sessions, one approximately one hour and the second approximately three hours including breaks). The research project is organized in a way that the students can participate in the workshop without participating in the research project. Students' participation in the Kodeløypa workshop is their school's will and part of a school activity, but, participation in the research project is optional. More precisely, participation in the research project is voluntary: Only students who want to participate will take part, and they can withdraw at any time. If student or student's parents choose not to participate or later withdraw from the research project, their choice will not affect their participation in the Kodeløypa workshop and their assessment in school. In that case, none of the previously mentioned data will be collected, and the student will be able to attend all the activities of the workshop without any commitment to the research project. Legal guardians should give their consent on behalf of the child for participation in the research project and not the Kodeløypa workshop per se. They can request to see the questionnaire and ask for any additional information regarding the eye-tracking technique.

### **What will happen to the information about you?**

All personal data will be treated confidentially. Only the project group (see general information section below) will have access to the personal data. The list of names of the students will be stored on a computer in a network with internet access belonging to NTNU, only the project group will have access.

We state that the participants will not be recognizable in the publication. The project is scheduled for completion by 28th of August 2018, then all data be anonymized.



---

## **Voluntary participation**

It is voluntary to participate in the project, and you can at any time choose to withdraw your consent without stating any reason. If you decide to withdraw, all your personal data will be made anonymous.

## **General information-project group**

The leader of the project is Michail Giannakos, Associate Professor at Department of Computer and Information Science at NTNU, e-mail: michailg@ntnu.no, address: Sem Sælands vei 9, IT-bygget \* 103, phone number: +47 73593469. Visiting researcher: Kshitij Sharma, Doctoral assistant at Computer-Human Interaction in Learning and Instruction (CHILI) Lab, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, e-mail:

kshitij.sharma@epfl.ch

If you would like to participate or if you have any questions concerning the project, please contact Sofia Papavlasopoulou, a Ph.D. student at the Department of Computer and Information Science at NTNU, e-mail: spapav@ntnu.no, address: Sem Sælands vei 9, IT-bygget \* 102, mobile number: +47 45786588.

The study has been notified to the Data Protection Official for Research, NSD - Norwegian Centre for Research Data.

## **Consent for participation in the study**

I have received information about the project and am willing to give my consent for my child's participation.

---

Participant's name: \_\_\_\_\_

---

(Signed by parent/legal guardian, date)

# Appendix **B**

## Interview questions for Kodeløypa

### **B.1 English version**

#### **General instructions for the interviews**

- Search contact with the children for the interviews as soon as possible when you see that they are finishing up their projects. After the interviews, they should still have time to play games
- When you start recording at the beginning of the interview, you need to say: Boy or girl, age, team of two or three, and the indication code you are using for that child (for example: Zebra group of 2 and then the “code” you gave to that child, for example, “1” or “name”)
- We have only one recorder so, use your phones for recording the interview (put the microphone as close as possible to record well).
- We have to triangulate the data we collect from the children. Children who are interviewed will also fill out questionnaires. Write your indication of that child in the questionnaire and give it to him or her

- 
- For the first child you interview, you can hand out the questionnaire after the interview. For the second child, hand out their questionnaire before you start interviewing the first child
  - You do not have to stay while he or she fills out the questionnaire if you have an indication of who the child is. Give the questionnaire to him or her and let him or her fill it out while you continue with the interviews.

Guidelines for the interviews:

- The goal is that questions lead to detailed answers and not a simple “Yes” or “No”.
- Also, try to keep the interviews a bit short (I think around 10 min), so you can interview as many as possible each workshop (minimum two children each of the assistants so, minimum six each time).
- Depending on the answers you get, you decide if you have to ask the sub-questions, otherwise you move to the next one.
- Focus on the learning and creating games with Scratch (they have to reflect for their games, in that sense that their statements are general or about specific code elements they used in the Scratch programming environment)

## **Interview questions**

- What did you like/enjoy most during the game creation experience?
- How do you feel about the game creation experience?  
(fun/challenging/creative/boring)
- Which difficulties did you face during the game creation experience?

- 
- What do you think was the most difficult part of the game creation experience?
    - Can you mention something specific from your script?
  - What do you think was the easiest part of the game creation experience?
    - Can you mention something specific from your script?
  - What frustrated you most? (at the game creation experience and Scratch)? \*
    - Can you mention something specific from your script?
  - What impressed you most? (at the game creation experience and Scratch)? \*
    - Can you mention something specific from your script?
  - Did you feel that you were actively part of the collaboration in the team during the process?
  - How do you feel about the collaboration in your team?
    - How much do you think you contributed to the team?
    - Do you feel that your opinions were taken into account from the team members?
  - Here we expect that the children answer if they were a valuable part of the team, if they were excluded, if they acted like a leader or not.

---

## B.2 Norwegian version

### Generelle instruksjoner for intervju

- Intervju personer så fort som mulig etter at gruppen er ferdig med prosjektet sitt, slik at vedkommende kan dra tilbake og fortsette med å spille spill etterpå
- På starten av intervjuet når du starter å ta opptaket, si: gutt eller jente, alder, lag på to eller tre, og indikasjonskoden du brukte for dette barnet (for eksempel: Zebra, gruppe av 2, og den koden du ga barnet, eksempelvis, "1" eller "navn")
- Vi har bare en opptaker, så bruk telefonene deres for å ta opp intervjuene (plasser mikrofonen så nærme som mulig)
- Vi må matche mest mulig data som vi innhenter fra en person, så skriv et nummer, hans/hennes kallenavn, eller noe annet på spørreskjemaet som personen du har intervjuet mottar
- For den første personen du intervjuer kan du gi spørreskjemaet på slutten av intervjuet, men for det andre barnet, si til han/henne at han/hun skal fylle ut sitt spørreskjema og gi det til deg på starten av intervjuet
- Du må nødvendigvis ikke bli mens han/hun fyller ut spørreskjemaet. Hvis du har en indikasjon om hvem vedkommende er, gi spørreskjemaet til han/hun og la han/hun fylle det ut mens du intervjuer neste person

Retningslinjer for intervjuene:

- Målet er at spørsmålene skal lede til detaljerte svar, ikke enkle "Ja" og "Nei".

- 
- Prøv å holde intervjuene korte (Rundt 10 minutter), slik at du kan ha så mange intervjuer som mulig per workshop (Minimum to stykker for hver assistent, så minimum seks per gang)
  - Avhengig av de svarene du får, kan du bestemme selv om du ønsker å stille sub-spørsmålene eller om du ønsker å gå videre til neste spørsmål
  - Fokuser på læring og laging av spill med Scratch (De må reflektere over spillene sine, på en måte at det kan være både generell refleksjon og om de mest viktigste kodelementene de brukte i Scratch)

## **Intervjuspørsmål**

- Hva likte du best ved det å lage spillet?
- Hva føler du om det å lage spill? (Gøy/utfordrende/kreativ/kjedelig)
- Hvilke typer utfordringer fikk dere mens dere lagde spillet?
- Hva synes du var det vanskeligste med å lage spillet?
  - Kan du nevne noe spesifikt fra skriptet/koden deres?
- Hva var det enkleste med spillagingen?
  - Kan du nevne noe spesifikt fra skriptet/koden deres?
- Hva frustrerte deg mest? (Gjelder både lagingen av spillet og bruken av Scratch) \*
  - Kan du nevne noe spesifikt fra skriptet/koden deres?
- Hva imponerte deg mest? (Gjelder både lagingen av spillet og bruken av Scratch) \*
  - Kan du nevne noe spesifikt fra skriptet/koden deres?

- 
- Følte du at du var en aktiv del av samarbeidet i gruppen mens dere lagde spill?
  - Hva føler du om samarbeidet i teamet?
    - I hvilken grad tror du at du bidro til gruppen?
    - Følte du at meningene dine ble hørt?
  - Her ønsker vi å vite om vedkommende var verdifull for teamet, om h\*n ble ekskludert, om h\*n oppførte seg som en leder o.l



## Interview questions for Tappetina

### C.1 English version

#### General instructions for the interviews

- Search contact with the children for the interviews as soon as possible when you see that they are finishing up their projects. After the interviews, they should still have time to play games
- When you start recording at the beginning of the interview, you need to say: Boy or girl, age, team of two or three, and the indication code you are using for that child (for example: Zebra group of 2 and then the “code” you gave to that child, for example, “1” or “name”)
- We have only one recorder so, use your phones for recording the interview (put the microphone as close as possible to record well).
- We have to triangulate the data we collect from the children. Children who are interviewed will also fill out questionnaires. Write your indication of that child in the questionnaire and give it to him or her

- 
- For the first child you interview, you can hand out the questionnaire after the interview. For the second child, hand out their questionnaire before you start interviewing the first child
  - You do not have to stay while he or she fills out the questionnaire if you have an indication of who the child is. Give the questionnaire to him or her and let him or her fill it out while you continue with the interviews.

Guidelines for the interviews:

- The goal is that questions lead to detailed answers and not a simple “Yes” or “No”.
- Also, try to keep the interviews a bit short (I think around 10 min), so you can interview as many as possible each workshop (minimum two children each of the assistants so, minimum six each time).
- Depending on the answers you get, you decide if you have to ask the sub-questions, otherwise you move to the next one.
- Focus on the learning and creating games with Scratch (they have to reflect for their games, in that sense that their statements are general or about specific code elements they used in the Scratch programming environment)

## **Interview questions**

- What did you like/enjoy most during the game creation experience?
- How do you feel about the game creation experience?  
(fun/challenging/creative/boring)
- Which difficulties did you face during the game creation experience?

- 
- What do you think was the most difficult part of the game creation experience?
    - Can you mention something specific from your script?
  - What do you think was the easiest part of the game creation experience?
    - Can you mention something specific from your script?
  - What frustrated you most? (at the game creation experience and Scratch)? \*
    - Can you mention something specific from your script?
  - What impressed you most? (at the game creation experience and Scratch)? \*
    - Can you mention something specific from your script?
  - What difficulties did you face in the sketching part?
  - What difficulties did you face in the coding part?
  - What did you have to change during the process that you could do in paper and not in Scratch?
  - What could you do in Scratch which you did not think about while sketching the storyboard (sketching part)?
  - Did you feel that you were actively part of the collaboration in the team during the process?
  - How do you feel about the collaboration in your team?
    - How much do you think you contributed in the team?
    - Do you feel that your opinions were taken into account from the team members?

- 
- Here we expect that the children answer if they were a valuable part of the team, if they were excluded, if they acted like a leader or not.
  - General question: What do you think about environmental issues?

## **C.2 Norwegian version**

### **Generelle instruksjoner for intervju**

- Ta med personen så fort som mulig etter at gruppen er ferdig med prosjektet sitt, slik at vedkommende kan dra tilbake og fortsette med å spille spill etterpå
- På starten av intervjuet før du starter å ta opptaket, si: gutt eller jente, alder, lag på to eller tre, og indikasjonskoden du brukte for dette barnet (for eksempel: Zebra, gruppe av 2, og den koden du ga barnet, eksempelvis, "1" eller "navn")
- Vi har bare en opptaker, så bruk telefonene deres for å ta opp intervjuene (plasser mikrofonen så nærme som mulig)
- Vi må matche så mye som mulig data som vi innhenter fra en person, så skriv et nummer, hans/hennes kallenavn, eller noe annet på spørreskjemaet som personen du har intervjuet mottar
- For den første personen du intervjuer kan du gi spørreskjemaet på slutten av intervjuet, men for det andre barnet, si til han/henne at han/hun skal fylle ut og gi det til deg på starten av intervjuet
- Du må ikke bli mens han/hun fyller ut spørreskjemaet. Hvis du har en indikasjon om hvem vedkommende er, gi det til han/hun og la han/hun fylle det ut mens du intervjuer neste person

---

Retningslinjer for intervjuene:

- Målet er at spørsmålene skal lede til detaljerte svar, ikke enkle "Ja" og "Nei".
- Prøv å holde intervjuene korte (Rundt 10 minutter), slik at du kan ha så mange intervjuer som mulig per workshop (Minimum to stykker for hver assistent, så minimum seks per gang)
- Avhengig av de svarene du får, kan du bestemme selv om du ønsker å stille sub-spørsmålene eller om du ønsker å gå videre til neste spørsmål
- Fokuser på læring og laging av spillet med Scratch (De må reflektere over spillene sine, på en måte at det kan være både generell refleksjon og de mest viktigste kodelementene de brukte i Scratch)

## Intervjuspørsmål

- Hva likte du best ved det å lage spillet?
- Hva føler du om det å lage spillet? (Gøy/utfordrende/kreativ/kjedelig)
- Hvilke typer utfordringer fikk dere mens dere lagde spillet?
- Hva synes du var det vanskeligste med å lage spillet?
  - Kan du nevne noe spesifikt fra skriptet/koden deres?
- Hva var det enkleste med spillagingen?
  - Kan du nevne noe spesifikt fra skriptet/koden deres?
- Hva frustrerte deg mest? (Gjelder både lagingen av spillet og bruken av Scratch) \*
  - Kan du nevne noe spesifikt fra skriptet/koden deres?

- 
- Hva imponerte deg mest? (Gjelder både lagingen av spillet og bruken av Scratch) \*
    - Kan du nevne noe spesifikt fra skriptet/koden deres?
  - Hvilke utfordringer fikk dere under skissedelen av storyboard?
  - Hvilke utfordringer fikk dere under kodedelen?
  - Måtte dere endre noe i prosessen siden det var noe dere kunne lage på papir, men ikke i Scratch?
  - Hva kunne dere gjøre i Scratch og som dere ikke tenkte på i skisseprosessen da dere lagde storyboardet?
  - Følte du at du var en aktiv del av samarbeidet i gruppen mens dere lagde spill?
  - Hva føler du om samarbeidet i teamet?
    - I hvilken grad tror du at du bidro til gruppen?
    - Følte du at meningene dine ble hørt?
  - Her ønsker vi å vite om vedkommende var verdifullt for teamet, om h\*n ble ekskludert, om h\*n oppførte seg som en leder o.l
  - Et generelt spørsmål: Hva synes du om miljøproblemer?

# Appendix D

## Dr.Scratch competence levels

<b>CT Concept</b>	<b>Null (0)</b>	<b>Basic (1)</b>	<b>Developing (2)</b>	<b>Proficiency (3)</b>
<b>Abstraction and problem decomposition</b>	-	More than one script and more than one sprite	Definition of blocks	Use of clones
<b>Parallelism</b>	-	Two scripts on green flag	Two scripts on key pressed, two scripts on sprite clicked on the same sprite	Two scripts on when I receive message, create clone, two scripts when %s is > %s, two scripts on when backdrop change to
<b>Logical thinking</b>	-	If	If else	Logic operations

**Table D.1:** Competence levels used by Dr.Scratch (Moreno-León et al., 2015)

---

<b>CT Concept</b>	<b>Null (0)</b>	<b>Basic (1)</b>	<b>Developing (2)</b>	<b>Proficiency (3)</b>
<b>Synchronization</b>	-	Wait	Broadcast, when I receive message, stop all, stop program, stop programs sprite	Wait until, when backdrop change to, broadcast and wait
<b>Flow control</b>	-	Sequence of blocks	Repeat, forever	Repeat until
<b>User Interactivity</b>	-	Green flag	Key pressed, sprite clicked, ask and wait, mouse blocks	When %s is >%s, video, audio
<b>Data representation</b>	-	Modifiers of sprites properties	Operations on variables	Operations on lists

**Table D.2:** Competence levels used by Dr.Scratch (Moreno-León et al., 2015)  
(continued)



# Appendix **E**

## Kodeløypa categories for codes

The following tables describe and summarize the codes and categories found for the Kodeløypa workshop. Codes are first level markings in the data set, and categories are groups of codes that are related to each other. References indicate the number of codes belonging to the category, while reference examples contain one or several quotes from the workshop participants. Some of the categories do not have belonging subcategories.

## E.1 Collaboration

Name and description	References	Reference example(s)
Indication of equal collaboration	29	<i>"Someone did something, and the other did something else", "Everyone were actively participating"</i>
My own contribution	29	<i>"I contributed where I could, and well, I think I contributed in a good way", "I wrote a lot, did a lot"</i>
Good collaboration	19	<i>"My team member and I got it to work"</i>
Understanding that the participant did not do much	15	<i>"If I knew more, I could've helped more", "If I was alone I wouldn't manage to do the same"</i>
Leader behavior	14	<i>"Got bored because one person was working on the computer", "The rest of the team wanted the one who knew about programming to be the leader"</i>
Collaborating with friends	12	<i>"We know each other and we are pretty safe around each other", "It was fun to create a game with a friend because it became more fun"</i>
Distribution of roles	9	<i>"I took one role as a programmer"</i>
Bad collaboration	6	<i>"Bad collaboration when we argued"</i>
Enjoyment of collaboration	3	<i>"Enjoyed that we worked together as a team", "I really like my team"</i>

**Table E.1:** Subcategories for collaboration, frequencies and examples

## E.2 Difficulties

<b>Name and description</b>	<b>References</b>	<b>Reference example(s)</b>
Hide and movements	22	<i>"Challenging to put together things to make it move to another position", "Make the figures hide when they were hitting the sides"</i>
Debugging and fixing code	18	<i>"Small things broke the game", "Put things together in the right order"</i>
Background, figures and graphics	13	<i>"Finding the right commands to get the character to do whatever we wanted", "We didn't manage to make some things as we wanted"</i>
Starting	10	<i>"It was hard because we thought that something was gonna do that, but it didn't", "The difficult part was to start"</i>
Understanding	8	<i>"Another challenge was to understand how to do thing"</i>
Code blocks	8	<i>"We did not find the right blocks for the game", "If you did something, it wasn't always what you expected"</i>
Life and high score	8	<i>"Counting points, removing stuff"</i>
Coding	7	<i>"Since I haven't coded before, the coding was challenging", "It would be more fun if you know how to do it"</i>

**Table E.2:** Subcategories for difficulties, frequencies and examples

---

<b>Name and description</b>	<b>References</b>	<b>Reference example(s)</b>
Game planning	7	<i>"Actually realizing the idea was really challenging", "The most challenging was actually to come up with a plan for the game"</i>
Jump	5	<i>"We didn't know how to make the character jump"</i>
Challenging in general	5	<i>"It was exciting and challenging"</i>
More time needed	4	<i>"(We) didn't have enough time"</i>
Coordinates	4	<i>"To write the coordinates, where to place the figures"</i>
Game creation	3	<i>"Just to understand the program better. To know what's what"</i>
Sound and music effects	3	<i>"Adding sounds was hard"</i>
No difficulties	3	<i>"I felt that there weren't a lot missing (In Scratch)", "It was really apparent what the bricks were doing"</i>

**Table E.3:** Subcategories for difficulties, frequencies and examples  
(continued)

---

### E.3 Out of school experience

<b>Name and description</b>	<b>References</b>	<b>Reference example(s)</b>
General about the workshop	25	<i>"It was boring to a certain amount", "Exciting to program games and robot", "It was interesting to try programming"</i>
Liked to be creative, self-explore	21	<i>"Creating a game exactly how we wanted to", "I liked seeing how things are built up", "To think for your own"</i>
Former experience	7	<i>"We have done a lot of programming with Scratch in school before", "I haven't really used Scratch before, so that is something new to me", "I had Scratch earlier in school"</i>
Liked the coding process	6	<i>"Enjoy to move the figures and fix the code", "I like finding the bugs in the game"</i>

**Table E.4:** Subcategories for out of school experience, frequencies and examples

## E.4 Easy

Name and description	References	Reference example(s)
Background figures	14	<i>"Creating figures and such was really easy", "To draw the main character. That was also the funniest"</i>
Scratch	12	<i>"It was pretty easy to use Scratch", "How easy it was to understand"</i>
Scratch movements	12	<i>"Easy to make the obstacles move", "The movement was pretty easy too"</i>
Complications when someone already knows coding	9	<i>"Scratch becomes hard since it is so easy", "Scratch is easier for novice users"</i>
Combine the blocks	8	<i>"The program made specific things to work really easy", "It was easy programming because you had to drag things", "Easy to put things you wanted to happen together"</i>
Repeat and forever	2	<i>"A lot of things were easy. Like repeat and forever"</i>

**Table E.5:** Subcategories for out of school experience, frequencies and examples

---

## E.5 Fun

<b>Name and description</b>	<b>References</b>	<b>Reference example(s)</b>
Fun in general	15	<i>"It was a really fun day"</i>
Fun in Scratch	14	<i>"It was fun to find out how to put things together", "Fun to see the progress in the project, more fun to decide what they (figures) should do"</i>
Hard but fun	6	<i>"I thought it was fun to be challenged"</i>
Achievement	5	<i>"It was fun when things worked and we thought about things and they actually worked"</i>
Learn from trials and errors	3	<i>"There are a lot of trials and errors when trying to create a game"</i>
Creativity	2	<i>"It was really fun to make simulations from Physics that you see in everyday life (in Scratch)"</i>
Fun to create a game	1	<i>"I liked most to play the game afterwards"</i>

**Table E.6:** Subcategories for fun, frequencies and examples

---

## E.6 Achievement

<b>Name and description</b>	<b>References</b>	<b>Reference example(s)</b>
Final result	12	<i>"We managed to make a game in the end, that it became something", "The result. I didn't expect it to be, for me to like it as much", "See when you were finished and manage to do it. And play it"</i>
Evolution	11	<i>"I don't really know where things are and stuff like that (in Scratch), but I learned it during my time here", "We managed to solve it following the tutorial", "We managed to solve the problems"</i>

**Table E.7:** Subcategories for achievement, frequencies and examples



---

## E.7 Scratch potential

<b>Name and description</b>	<b>References</b>	<b>Reference example(s)</b>
Possibilities with Scratch	12	<i>"You can create a game in a short amount of time", "Something so simple can do so much", "Some of the groups created many different games"</i>
Impressed by the code	6	<i>"The program impressed me", "(I was impressed) by making functions work together to create a game"</i>
Previous knowledge	3	<i>"It does not really work to create a game without knowing how it works and I understand how Scratch works compared to others"</i>

**Table E.8:** Subcategories for Scratch potential, frequencies and examples

---

## E.8 Help

<b>Name and description</b>	<b>References</b>	<b>Reference example(s)</b>
Only when needed	6	<i>"Got help to resolve problems"</i>
Frustration	5	<i>"The assistant gave us some help when we were frustrated", "It wasn't frustrating because we received help almost immediately"</i>
Random	3	<i>"The tutorial was well written, someone new to programming could understand", "The help we got made it much easier to see how to do things"</i>
Help in general	3	<i>"Easy to get help and finish"</i>
Getting help from other teams	2	<i>"When we didn't find anything, we looked at another group and saw how they did it"</i>

**Table E.9:** Subcategories for help, frequencies and examples

---

## E.9 Change of perceptions

<b>Name and description</b>	<b>References</b>	<b>Reference example(s)</b>
Expected harder	9	<i>"It looked complicated, even though it wasn't", "Scratch wasn't that hard", "The program was not complicated"</i>
Expected easier	7	<i>"I thought I did know how to do it, but I didn't", "Thought that it was easier to program, but it wasn't", "...About how easy it looked but also how complicated it was"</i>
Better in general	5	<i>"I liked that we managed to create it (game). Better than I thought", "I like programming more now because I know how it works"</i>

**Table E.10:** Subcategories for change of perceptions, frequencies and examples

---

## E.10 Scratch discoveries

<b>Name and description</b>	<b>References</b>	<b>Reference example(s)</b>
Able to say something from their script	12	<i>"We managed to make the game play in repeat", "We used "right arrow" as it is called to make it move", "Repeat. Repeat the code blocks several times"</i>
General about Scratch	8	<i>"It was really apparent what the bricks were doing", "Be careful when changing things in the code"</i>

**Table E.11:** Subcategories for Scratch discoveries, frequencies and examples

## E.11 Learning

<b>Name and description</b>	<b>References</b>	<b>Reference example(s)</b>
Learned "something"	8	<i>"The learning process was good", "I enjoyed learning new things", "I liked more that I learned new things"</i>
Learned about "programming"	7	<i>"When you have learned something, you could use it", "I learned to program different stuff"</i>
Learning "instructions"	3	<i>"I understand if, when, forever to a certain amount"</i>

**Table E.12:** Subcategories for learning, frequencies and examples

---

## E.12 Intention

Name and description	References	Reference example(s)
Intention	4	<i>"I won't program again", "I think I will try again", "Maybe I will do it again"</i>

**Table E.13:** The intention category, frequencies and examples

# Appendix **F**

## Tappetina categories for codes

The following tables describe and summarize the codes and categories found in this study for Tappetina. Codes are first level markings in the data set, and categories are groups of codes that are related to each other. References indicate the number of codes belonging to the category, while reference examples contain one or several quotes from the workshop participants. Some of the categories do not have belonging subcategories.

---

## F.1 Difficulties

<b>Name and description</b>	<b>References</b>	<b>Reference example(s)</b>
Difficulties	30	<i>"Things didn't work like we wanted (like the timer)", "Hard to debug", "Sometimes it was not very easy to find what we were looking for", "Jumping was hard", "Difficult to make things go up and down", "(It took) a lot of time to find a new block"</i>
Mistakes	9	<i>"We had to redo stuff", "Things got deleted", "We erased something"</i>
Difficult	7	<i>"(Not everything went so well", "(It was a) struggle", "Difficult to do it in the best way"</i>

**Table F.1:** Subcategories for difficulties, frequencies and examples

---

## F.2 Emotions

<b>Name and description</b>	<b>References</b>	<b>Reference example(s)</b>
Achievement	15	<i>"When I tried to do it, it touched a lot of other stuff, so I gave the task to the other girl", "We didn't manage to do everything we thought about", "We managed to do everything we drew", "We tried to make things happen"</i>
Confidence	9	<i>"I knew more than I thought I did", "How easy it was", "It wasn't complicated to code", "I thought it was much harder to make a game"</i>
Fun	6	<i>"It was fun to do stuff like that", "More fun because I knew more"</i>

**Table F.2:** Subcategories for emotion, frequencies and examples



---

## F.3 Scratch

<b>Name and description</b>	<b>References</b>	<b>Reference example(s)</b>
From paper to Scratch	14	<i>"We didn't have the portal in the original sketch", "We managed to make everything from paper to Scratch", "(We needed to) find an idea that wouldn't be so hard to create"</i>
Scratch in general	13	<i>"(We used) different bricks to make a big recipe", "Impressed me that we could copy code blocks and other things", "Scratch also has a lot of things you didn't know you needed", "Fun to program and stuff", "(It was) a little complicated"</i>

**Table F.3:** Subcategories for Scratch, frequencies and examples

---

## F.4 Collaboration

<b>Name and description</b>	<b>References</b>	<b>Reference example(s)</b>
Roles in collaboration	12	<i>"Everyone contributed as much as others", "I decided how things would look and behave", "It was hard to finish my task because they just wanted the control of the computer back", "They did not let me try"</i>
Good collaboration	4	<i>"No fights or discussions", "Collaboration was really good"</i>
Idea creation	5	<i>"We had an idea and we stuck with it", "I had the idea, and the rest wanted to join", "We collaborated in the poster more", "I contributed with the idea of the game"</i>
Bad collaboration	2	<i>"We disagreed with each other", "Not a good collaboration"</i>

**Table F.4:** Subcategories for collaboration, frequencies and examples

---

## F.5 Easy

Name and description	References	Reference example(s)
Easy	16	<i>"(The) easiest was to make things happen immediately after clicking the start button", "Putting the blocks together", "Easy to code ourselves", "Easy to find what we were looking for"</i>

**Table F.5:** The easy category, frequencies and examples

## F.6 Environmental issues

Name and description	References	Reference example(s)
Environmental issues	7	<i>"(We) could throw less garbage", "Animals are dying because of environmental issues"</i>

**Table F.6:** The environmental issues category, frequencies and examples

---

## F.7 Time issues

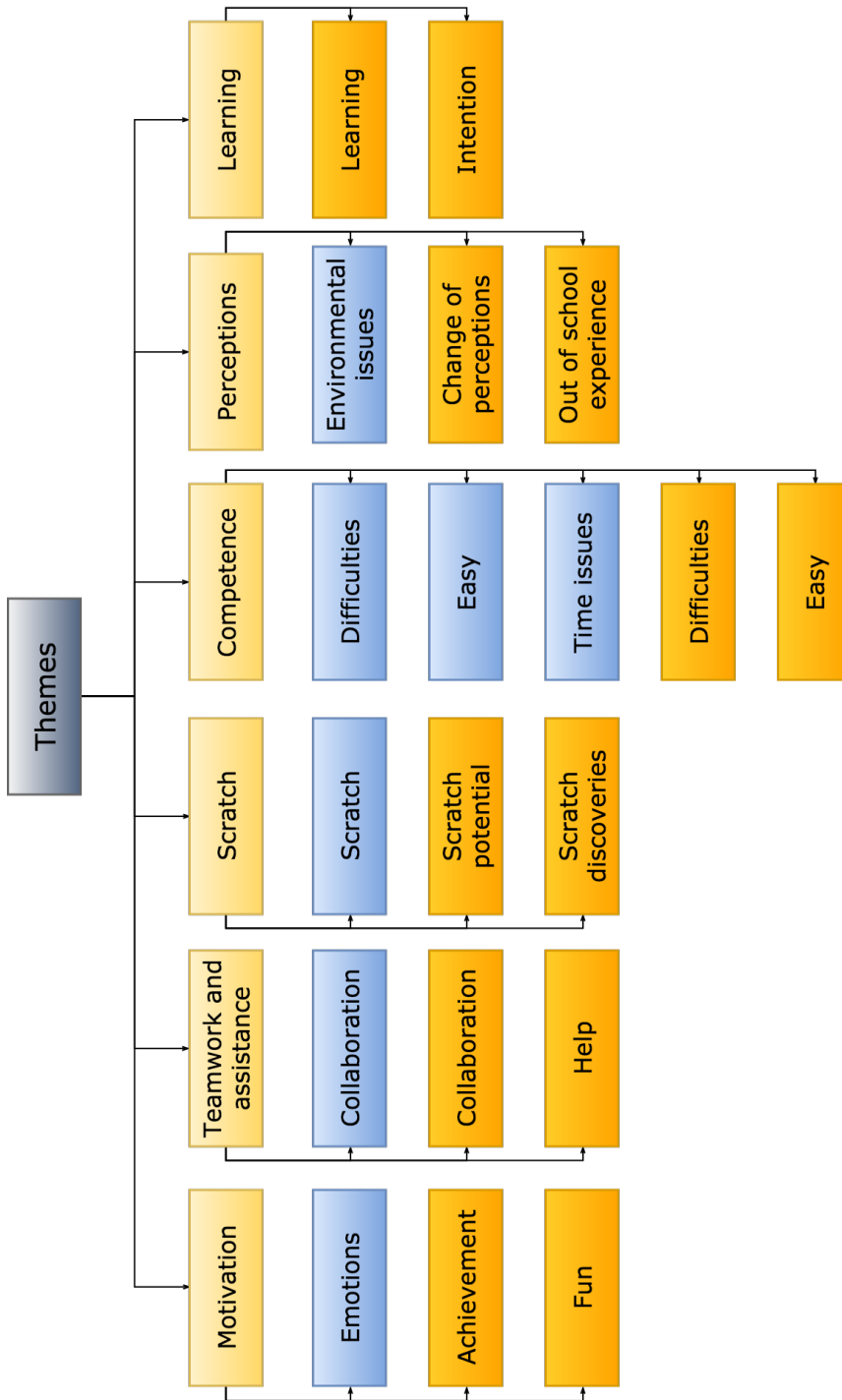
Name and description	References	Reference example(s)
Time issues	7	<i>"A lot of mess happened because we didn't have so much time left", "Not enough time to do everything"</i>

**Table F.7:** The time issues category, frequencies and examples

# Appendix G

## Themes and belonging code categories

The following illustration shows the final results of the coding procedure of interviews. It shows clusters of categories from the Tappetina and Kodeløypa workshops and their belonging theme. Blue boxes imply categories from Tappetina, while the orange boxes imply categories from Kodeløypa. The order of the categories is not significant. In total six themes were generated from the coding process.



**Figure G.1:** Clusters of coding categories made into themes

# Appendix H

## Results from Kodeløypa questionnaires

Statement	None	1	2-3	4-5	6 or more
Courses with Scratch, Alice, etc. prior workshop*	68	18	11	2	3

**Table H.1:** Frequencies for prior attended visual programming courses before the Kodeløypa workshop

---

<b>Statement</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>
Previous experience with programming prior workshop	36	20	11	23	6	2	5
Felt part of a learning community in my group	0	1	7	12	10	16	58
Actively exchanged my ideas with group members	0	1	2	5	15	23	58
Able to develop new skills and knowledge from other members in my group	3	3	12	7	26	22	29
Collaborative learning in my group was effective	3	3	6	9	18	19	44
Overall satisfied with my collaborative learning in this course	0	2	3	10	10	26	52

**Table H.2:** Frequencies for Likert scores given during the Kodeløypa workshop



# Appendix I

## Results from Tappetina questionnaires

Statement	None	1	2-3	4-5	6 or more
Courses with Scratch, Alice, etc. prior workshop*	4	2	2	0	0

**Table I.1:** Frequencies for prior attended visual programming courses before the Tappetina workshop

<b>Statement</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>
Previous experience with programming prior Tappetina	1	1	0	2	1	1	1
Satisfied with coding	1	0	0	3	1	1	1
Pleased with coding	0	0	0	1	2	2	1
My decision to attend coding activities is a wise one	0	0	0	0	3	1	3
I intend to attend coding activities in the future	0	0	0	1	3	0	3
My general intention to attend coding activities in the future is very high	0	1	0	2	1	0	3

**Table I.2:** Frequencies for Likert scores given during the Tappetina workshop

<b>Statement</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>
I will regularly attend similar coding activities in the future	0	0	1	2	3	0	1
I will think about attending similar coding activities	0	0	0	2	2	0	2
Coding is easy	0	1	2	3	1	1	0
I find coding flexible	0	0	3	2	1	0	1
The process of coding is clear and understandable	0	0	2	3	2	0	0
It is easy for me to attain skills with coding	0	0	1	2	1	2	1
Attending coding activities is enjoyable	0	0	0	3	3	0	1

**Table I.3:** Frequencies for Likert scores given during the Tappetina workshop (continued)

---

<b>Statement</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>
Attending coding activities is exciting	0	0	0	2	2	1	2
I am feeling good with coding	1	0	1	1	3	0	1
Attending coding activities is boring	4	0	0	2	1	0	0
I find coding useful	0	0	0	2	1	2	2
Coding activities improve my performance in coding	0	0	0	2	1	2	2
Coding activities enhance the effectiveness in coding	0	0	1	1	2	2	1
Coding activities increase my capabilities in coding	0	0	1	1	1	1	2
Being involved with coding I perform better than the acceptable level	0	0	0	2	0	2	0
Being involved with coding I perform better than can be expected from me	0	0	1	2	1	2	0
Being involved with coding I put in extra effort in my work	0	0	1	1	2	1	1
Being involved with coding I expend a great deal of effort carrying out my work	0	0	1	1	3	1	0
Being involved with coding I try to learn as more as possible	0	0	0	2	2	2	1
Being involved with coding the quality of my learning is top-notch	0	0	0	2	2	1	1

**Table I.4:** Frequencies for Likert scores given during the Tappetina workshop (continued)