Tampere University

Yen Can Nguyen Hai

INTEGRATING HACKER CULTURE INTO CODE LITERACY EDUCATION

Faculty of Information Technology and Communication Sciences
Master's Degree Programme in Digital Literacy Education
Master's thesis in Social Science
April 2021

Tampere University
Faculty of Information Technology and Communication Sciences
YEN CAN NGUYEN HAI: Integrating hacker culture into code literacy education
Master's thesis in Social science, 78 pages, 3 pages of appendices
April 2021

In this algorithm-driven era, code literacy is being deliberately promoted as a complement to digital literacy. The concept of *hack* has recently been paid attention to as a phenomenon in code literacy education. Educational initiatives such as *hackathon* and *hackerspace* have been under the spotlight. However, there is still a lack of academic research delving into those initiatives, or more broadly, into code literacy education, from the hacker culture perspective. Thus, the scope of this study is to understand how the characteristics of hacker culture are integrated into code literacy education.

The current study is designed under the interpretivist paradigm and qualitative research approach. Six thematic interviews were conducted with six Finnish educational experts and practitioners to explore how they perceived hacker culture as an educational concept and how they had brought the characteristics of hacker culture into their educational practices. The interview data then was analyzed using thematic analysis method.

The findings show that the integration of hacker culture into code literacy education is beneficial for both students and educators. This integration encourages students' collaboration, creativity, tinkering skills and helps to bring more playfulness to the class. Besides, student diversity, which manifests in students' different levels, interest and genders, is also embraced in a coding class. In addition, the findings show that Finnish students have a fairly easy access to tools and devices required for their technological exploration. The present study also reveals the perception of educators towards hacker culture as an educational concept. From their viewpoint, promoting hacker culture in code literacy education is to advocate self-directed learning, exploratory attitude, open sharing and collaboration.

This research also gains some insights into the implementation of code literacy education in Finland. Code literacy is being actively advocated by different education sectors. However, the vagueness of instructions in the Finnish National Core Curriculum regarding code literacy makes the teaching practice depend strongly on teachers who are directly responsible for organizing coding activities in the class.

Finally, the study brings out some implications for educational policy and future research work. The educational policy focusing on nurturing talented students, who are good at and interested in coding, should be set out, as those students will form the backbone of hacker culture in the future. In the academic area, more studies should be conducted with a larger sample size to capture a more holistic picture of this research topic.

Keywords: code literacy education, hacker culture, digital literacy education

CONTENTS

1 INTRODUCTION

On August 28th, 2019, Apple Inc., one of the four Big Tech companies, was forced to formally send an apology letter to customers, because it was discovered that Apple had secretly collected conversations between its users and Siri—a virtual assistant integrated within Apple device operating systems. On April 11th, 2019, in the United State congressional hearing, Facebook CEO Mark Zuckerberg had to answer a series of questions regarding whether or not Facebook eavesdrops on users either directly or through a third party, as well as gathers and uses personal images and data that users are unaware of or unable to control. It was reported that after users have verbal interaction, their Facebook page immediately displayed advertisements with similar content that they have talked about before. The situation seems to escalate when in March 2021, once again, the CEO of Facebook, together with the CEOs of Google and Twitter, attended a US congressional hearing on the problem of misinformation and disinformation on online platforms, which led to many political riots. Despite the efforts of the authorities, it cannot be denied that general digital users can hardly take control of their digital life. Living in one's own "filter bubble" (the word by Pariser, 2012) has gradually become such a normal that they are mostly unaware of its presence. Ironically, those kinds of algorithms are creeping through and affecting every nook and cranny of human's digital life while digital users have barely understanding of what they are and how they work.

Another shocking technology news broke in mid-March 2016: Sophia the robot was unveiled. Sophia could mimic 62 human's facial expressions and answer interview questions fluently. This humanoid robot also shocked people with her staggering statements such as she would destroy humans or she wanted to start a family (CNBC, 2016). Public attention was at its peak when the robot was granted citizenship. However, the controversy surrounding this human-like robot erupted soon after, especially when the source code of it was publicized on GitHub—the world's largest programming community. They called Sophia "a scam", "a marketing stunt" or, more mockingly, "a puppet" (Urbi & Sigalos, 2018; Tardif, 2020; Vincent, 2018). This was because what had been programmed in Sophia was not of anything more intelligent than many other advanced technologies at that time (see, Parviainen & Coeckelbergh, 2020, for more detail). These technologies have been familiar to digital users with Amazon's Alexa or Apple's Siri application. Even though, the technological knowledge still

seems too bewildering for general digital consumers to digest. As a result, the public is easily inflamed, or even scared, by information concerning technological advances.

The above examples not only evidence that algorithm-driven technology has been penetrated more and more deeply into our daily life, but they also evoke many thoughts on how little digital users understand and control what is going on behind their own digital devices and platforms. It can be seen that now even the digital veterans, who are considered media- or digital-literate, cannot avoid being impacted by algorithm-based applications they are using every day. However, there is a remarkable gap between using technology and understanding it. Digital consumers are familiar yet strange with technology at the same time.

To close that knowledge gap, besides digital literacy and media literacy, there have been many attempts to bring code literacy to young generations. Code literacy refers to "a holistic competency for technical communicators" (Duin & Tham, 2019). This concept not only points to the programming skills but also implies "the understanding of the code and its intentions and context" (Dufva & Dufva, 2016). Therefore, from an education perspective, code literacy education can equip youngsters with comprehensive knowledge, skills, and ethics regarding programming and technology. As such, it can be said that code literacy is a complement to digital literacy.

Although digital literacy is becoming more and more intertwined with code literacy, this relationship is still insufficiently researched. Some studies treat code literacy as a separate subject (see, e.g., Dufva & Dufva, 2016), some others place code literacy in the framework of digital literacies but have not yet deeply examined their inverse relationship (see, e.g., Pegrum et al., 2018). For that reason, the first objective of the current study is to highlight this conceptual integration.

Given the importance of promoting code literacy among young people, computer programming has been featured in many national curriculums and non-formal education institutions around the world. In a report of European Schoolnet (2015), 16 European member countries had officially integrated programming into the national or regional curriculum. This number has grown since then (some countries (e.g., Finland and Belgium) were not counted at the time the report was published as they were just in the planning phase). The phenomenon is the same in many Asian countries. Coding is mandatory in Japan's primary schools from April 2020. In Korea, software courses are compulsory for primary school students from 2017 and for high schools from 2018. After four years of encouraging schools to blend coding into computer courses, China has endorsed to include computer programming in their national curriculum for primary and middle school students. In Malaysia, Year Four students are introduced to AI, robotics, and programming from 2020. In the United States, each state can make their own decision on how to bring computer science into their regional curriculum based on the K–12 Computer Science Framework. In another part of the world, South Africa shows their determination in

programming education by starting training teachers to teach coding and piloting computer programming as a school subject in 1000 schools from the 2020 academic year, despite the public debates on the success of this attempt considering low literacy levels of students at the moment. In addition, code literacy is also advocated in the non-formal education setting. The flexibility of this education sector could help educators and learners adapt easily to the ever-changing nature of technology. Various coding non-formal educational institutions and activities, both online and offline, have been established around the world. As of April 2021, there are 2412 hackerspaces all over the world, according to the statistic of Hackerspace Wiki[1]. Online coding courses, such as codecademy.com and freecodecamp.org, are becoming prevalent and attracting both beginning and advanced learners of all ages. Not to mention numerous extracurricular activities for school students are being organized to advocate coding in a relaxing and playful way.

In the field of technology education, for the past fifteen years, *hacking* has emerged as a phenomenal concept. On the one hand, in public discourse, *hacking* refers to criminal acts in the digital environment. Hackers are digital criminals who break into computer systems for reputation and (or) personal gain (Alleyne, 2018). On the other hand, it is widely used in the terms such as *hackathon* and *hackerspace*, pointing to creativity, problem-solving skills, and the think-out-of-box attitude. Hackathon, the term derived from *hack* and *marathon*, is considered an innovative pedagogy (Angarita & Nolte, 2020; Suominen et al., 2018). The hackerspace model has been replicated worldwide as a favorable learning environment to nurture young digital makers (Guthrie, 2014). However, there is still a lack of academic research delving into those initiatives, or more broadly, into code literacy education, from the hacker culture perspective. Besides, notwithstanding its conceptual complexity, hacker culture seems not to attract as much attention as maker culture does, even though it is the stem for maker culture to flourish.

Therefore, the main objective of this study is to understand how the characteristics of hacker culture are integrated into code literacy education. Considering the growth of code literacy education in both formal and non-formal settings, this study is contextualized in both contexts.

This research chooses Finland to be a case study for some justifiable reasons. First and foremost, this thesis project is done on completion of a master's study program in Finland, thus it will be the most feasible when targeting Finland as a research subject. Furthermore, according to The European Commission, Finland is ranked the first out of 28 EU Member States in the Digital Economy and Society Index (DESI) 2020. In Finland, introducing coding and embedding it as a mandatory component of the school curriculum is a long-term action for qualified ICT-related labor. The Finnish

[1] https://wiki.hackerspaces.org/List_of_Hacker_Spaces

government aims to attract 1% of the population (approximately 55,000 people) to learn more about the basics of artificial intelligence as machine learning and neural networks. To reach those ambitious goals, ICT is deemed one of the pillars of the Finnish National Core Curriculum 2014 (FNCC14)[2] which has taken effect since 2016, and programming is also integrated into math subject. The implementation of programming education in FNCC14 is supported by various non-formal education organizations across the country, e.g, hackerspaces and makerspaces, youth work centers, coding schools, coding hobbyist communities, and so on. Finland can be considered a reference model of the cooperation between different education sectors in promoting code literacy.

The structure of this thesis consists of seven chapters. The *Introduction* part indicates the research problem and justifications for the study. It is followed by the second chapter—*Code literacy, digital literacy, and hacker culture*, in which previous studies related to the thesis's topic are highlighted. Then, the *Methodology* section will explicitly describe the research design and research implementation. The results gained from the data collection and data analysis process will be presented in detail in the *Findings* chapter, and they are also summarized and discussed further in the *Discussion* section. The trustworthiness and ethical consideration of this study is examined in the *Evaluation* chapter. And finally, the *Conclusion* section closes the study with implications for educational practice and policy as well as future research work.

[2] In this study, Finnish National Core Curriculum is abbreviated as FNCC. The new curriculum which was published in 2014 and takes effect since 2016 is written as FNCC14.

2 CODE LITERACY, DIGITAL LITERACY, AND HACKER CULTURE

2.1 Code literacy and digital literacy

2.1.1 Digital literacy in the era of algorithm-dominated technology

The concept of digital literacy started coming under the spotlight when the term literacy was extended beyond text reading and writing and became "a metaphor for any kind of skill or competence" (Fransman, 2005). Since being popularized by Gilster (1997), this concept has attracted many research attempts, and as a result, many definitions of digital literacy have been provided. In general, digital literacy is the awareness, attitude, and abilities of individuals towards digital tools and environment (Martin & Grudziecki, 2006). Scholars offered many keywords to specify the digital abilities, such as "understand and use information" (Glister, 1997), "access, manage, navigate, analyze, create, reflect, evaluate, communicate, thinking, relate, take action, be associated with digital media" (Jones & Hafner, 2012; Hobb, 2017; UNESCO, 2018, Baron, 2019). The ultimate purpose of digital literacy is to prepare digital citizens to "enable constructive social action" (Martin & Grudziecki, 2006), secure employment, and participate in the modern digital information economy (UNESCO, 2018; Baron, 2019).

Those definitions of digital literacy present either conceptual interpretation or standardized sets of operations (Lankshear & Knobel, 2015). Either way, they focus on information and information retrieving tasks using digital tools. A digitally literate person is defined as a user who effectively utilizes the digital mediums to perceive and produce information—by information, it means the content transmitted by varieties of meaning-making modes, namely visual, audio, and text modes (see, e.g., The New London Group, 1984). It can be seen that the discussions about digital literacy have revolved around the idea of tool users rather than tool creators. It implies that the digital devices and platforms are ready-programmed, and the users do not have complete control over the tools that they are using.

For the past years, algorithms—the weapons serving monopoly power of many media corporations—are changing the way we use digital devices and perceive information from digital platforms. The recommendation algorithm used in most leading digital platforms (i.e., YouTube, Facebook, Google, Amazon) is a prime example of the correlation between the content users obtain

and the technological aspect of digital mediums delivering that content. The content users absorb is suggested by the recommendation algorithm which works based on users' preferences and past activities. Significantly, these algorithms not only reflect users' preferences but also actually shape them (Adomavicius et al., 2019). They "alter the way we encounter ideas and information" (Pariser, 2011). With algorithms, users are stimulated to reach specific ranges of content which leads to the state of "filter bubbles"—the intellectual isolation created by personalized information (Pariser, 2011). In such cases, even the digital veterans could hardly avoid being affected by algorithms.

Young people make up a large part of digital users, but many of them are not aware of the impact of algorithm-based media on their digital life (Kotilainen et al., 2020). Kotilainen et al. pointed out that the concepts such as "technology optimizing content for services" and "recommendations by applications and services" are even not on youth's radar. Most of the young informants participating in interviews conducted by Okkonen and Kotilainen (2019) revealed that they embraced the content offered by recommendation algorithms and found those content "interesting and attractive". They trust artificial intelligence algorithms and think they cause no harm to their privacy.

In the situation that youth still do not precisely evaluate the effects of algorithm-based media on their digital media practices, scholars are calling for the integration of basic concepts of algorithms and artificial intelligence into digital literacy education. Ptaszek (2020) used the term "media education 3.0" to point to a new situation in which media education has been redefined by big data, algorithms, and artificial intelligence. Kotilainen et al. (2020) addressed that users' understanding of such basic technology concepts is an essential part of media and information literacy nowadays. This requirement brings code literacy closer to digital literacy education than ever.

2.1.2 Code literacy and digital literacy: a supportive relation

Definition of code literacy

Code and *coding* are commonly considered specialized terms used in the sphere of computer science. Code is referred to as "the machine-readable language programmed to instruct computer software" (Williamson, 2015). Coding is an action of computer programmers when they compose lines of code. Coding has the same notion as programming, and they are typically used interchangeably.

Vee (2017) stated that the concept of literacy has been used "as a cipher for the kind of knowledge a society values". Therefore, when code is considered from the literacy point of view, first and foremost, it implies the urge to acquire coding knowledge and skills. Code, which lies at the heart of all technological devices, has the power of connecting and establishing a new form of society, shaping

people's lives and challenging existing mindsets as well as opening up new discussions (Dufva & Dufva, 2016). In that sense, code is a new force to change the world.

Also, code has become a new language for humans. Indeed, some researchers compared coding skills to text reading and writing skills. Vee (2013) quoted Alan Perlis (1961) that "all undergraduates should be taught programming, just as they are taught writing in first-year composition courses" and compares the development and necessity of programming languages today with the fact that reading and writing became "central to employment and citizenship in the nineteenth and twentieth centuries". It is the only language that we can use to communicate with the programmable machines. The more digital users proficient in this language, the more they can control their digital life.

As code is penetrating into every aspect of human life, it is not only discussed merely from the technological viewpoint but also from the societal and ethical perspectives. Building upon the theory of metaphors as an approach to understand abstract concepts in systems thinking and organizational studies, Dufva and Dufva (2016) propose four paradigms and nine metaphors of code, traversing from mechanical, cultural, political to creative aspect of code. Accordingly, code is considered not just as "linear sequences of commands" but as an intelligently man-made brain that influences our political systems and cultures. Code is even seen as an artistic tool for producing digital creative works. It implies that code brings social effects through the digital products that it makes up, and also reflects the social values, intentions, and even subconscious of the programmer. Considering that significance of code, the scholars have suggested a more holistic understanding of code literacy instead of perceiving it as a set of merely technical skills.

Betts (2011), as cited in Duin and Tham (2018), defined code literacy as "the ability to identify, understand, interpret, create, communicate, and use rules that shape and reshape information to participate fully in the creation of new information". Duin and Tham also presented three facets of code literacy: code as language, code as tool, and code as structure. Code as language refers to programming languages, such as Python, C++, Java. Code as tools points to the programming applications, analytic software, programming libraries, and sharing platforms such as GitHub. Code as structure deals with the processes in which information is mapped and proceeded, such as social media mining or machine learning. Dufva and Dufva (2016) gave a more comprehensive definition: "Code literacy includes both understanding the more ambiguous and multiplexed issues that exist around code, and the basic principles and logic of coding".

Code literacy has a close relationship to digital literacy. On the one hand, code literacy helps enhance youngsters' digital literacy. Khromov and Kameneva (2016) classified code literacy as one of the components of digital literacies. On the other hand, digital literacy facilitates beginners when they start to learn coding. For example, Scratch programming environment was created to help beginners

learn to code through interaction with graphical elements, interactive stories, games, and animations. In this way, learners only need to be proficient in using computer and retrieving digital information to take their first steps to coding; and even children at a very young age can do so (Kazakoff, 2015).

This study desires to distribute to the discussions on the relationship between code literacy and digital literacy. In the following sections, that relationship will be demonstrated based on the argument that code literacy complements digital literacy, directly and indirectly, through computational thinking and digital creation.

Code literacy improves computational thinking and computational knowledge

Learning to code is a direct way for students to grasp the notion of computational thinking and computational knowledge.

The term *computational thinking* was coined with the invention of the BASIC programming language, which promoted the "code for everyone" idea. Then, this concept was deepened by Seymour Papert with his famous works during the 1950s and the 1970s, including the book "Mindstorm" and the creation of the LOGO programming language (Kong & Abelson, 2019). Over a long period of development of the core concepts and definitions of computational thinking, it is now being widely introduced in the K-12 curriculum as an essential skill of digital citizens in the 21st century (see e.g., Denning, 2009; Hemmendinger, 2010; Wing, 2006).

Angeli et al. (2016) synthesized different views on computational thinking and came up with five essential components of this concept, namely abstraction, generalization, decomposition, algorithm, and debugging. *Abstraction* focuses on the skill to recognize the main elements of an object or a problem to keep the most important information and ignore the rest. *Decomposition* is about breaking down a problem into smaller parts so that a complex issue can be tackled easily and thoroughly. After the core of a problem is identified through abstraction and decomposition, a generic solution can be applied to other similar situations, which is called *generalization*. *Algorithm* in computational thinking refers to the ability to put a series of actions into a step-by-step sequence so that the problem can be solved logically and smoothly. The final element, *debugging*, addresses the skills to tinker and optimize the problem-solving process by detecting and fixing errors.

Computational thinking can be considered as a bridge that connects the thinking patterns and techniques from the field of computer science to our daily basis problems (Furbur, 2012). In the context of algorithm-based media presenting in all aspects of our lives, computational thinking helps young people enhance digital literacy in such a way that swift from content orientation to critical evaluation of platforms and devices (Kotilainen et al., 2020). Valtonen et al. (2019) suggest some specific

computational content that should be integrated into media literacy, namely tracking, recommenders, dynamic content creation, deep learning by computers, reinforcement learning, attention engineering, and content-filtering curation. These computational contents enhance students' critical thinking when participating in digital culture.

Code literacy supports digital creation and media expression

Code literacy has many potentials in supporting media creation and expression. There is recently a new trend in code literacy teaching in which creative coding is at the center. Creative coding refers to a type of programming in which expression is focused over function (Dufva, 2018). To put it simply, creative coding is coding, but it serves the purposes of making creative products rather than only focuses on the functional aspect. In the contemporary digital context, even music and poems can be composed by computer codes. For example, Sonic Pi is a coding program that allows programmers to perform live music composed with computer codes. Coding is "a new material for expression" (Maeda, 2004). Ideally, in the very near future, many students can learn how to use code to express themselves through computational products just as they are using pictures and words in social media nowadays. Creative coding not only brings a new perspective to our understanding of code but also suggests new approaches to both programming education and media education. It is also a demonstration for a broader understanding of code literacy which views code more than a functional and mechanical tool.

Another approach that helps students enhance code literacy is through digital fabrication and robotics (Peppler & Kafai, 2009). When designing a robot, students not only learn to program the sensors or movements of robots, but they also can investigate how the emotional and social aspects of robots are programmed (Kotilainen at al., 2020). This practice links tightly to the social aspect of computer code. In digital fabrication activities, students learn coding through making creative products. For instance, with a LEGO boost creativity toolkit, students can program a creative product such as a guitar or a sound sculpture. Digital fabrication activities typically take place in the making laboratories, such as makerspaces, hackerspace, fabulous (or fabrication) laboratories (Fab Lab).

2.2 Code literacy education in Finland

This study categorizes the Finnish approaches to code literacy education into two segments: *formal education* (school subjects, universities' programming courses), and *non-formal education* (outside school educational organizations, including coding schools, youth work centers, makerspaces). Due to

the limitation of time and resources of a master's thesis project, this study does not inquire into *informal education* which covers spontaneous and incidental learning activities.

2.2.1 Code literacy education in Finnish formal education sector

Computer programming has been officially brought into the new Finnish National Core Curriculum (FNCC14) since 2016. It should be strongly addressed that this is not an abrupt change in the FNCC. In fact, Finland has a fairly long history of integrating computer programming into formal basic education, which can be considered an important precondition for the development of code literacy education in this country today.

According to Tuomi et al. (2018), information technology and communication (ICT) was introduced in FNCC firstly in high schools in 1982 and then in secondary schools from 1987-1988. At that time, the concept of computer literacy aligned with Finland's policy and vision to become an information society. To ensure the effectiveness of computer teaching, additional programming courses were provided for teachers, mostly for mathematics and physics ones. However, due to the limitation of school facilities at that time, computer activities took place mainly in after-school clubs. The scholars also noted that the activities of computer hobbyists and unofficial computer clubs burgeoning during the 1980s played an important role in the advancement of code literacy education and the technology industry of Finland (Saarikoski, 2011). They set up the foundation for the computer culture, as well as the wave of ICT companies (e.g., Nokia, Tieto) in the 1990s. From 1994 onwards, ICT stopped being an individual school subject. Instead, it is considered a set of skills that traversed through multiple subjects.

Besides, technology education in Finland has an enduring connection with crafts. Crafts subject has been playing an indispensable role in FNCC since the establishment of the Finnish school system in 1866 (Pöllänen, 2009). Especially, boys are equipped with skills to work with electronic devices. They even educated themselves by using electronics guidebooks to build the hardware from scratch or tinker with it.

Situating in that context, it can be concluded that the reform of the FNCC14 as regard to computer programming is a continuity of the previous technology education practices in Finland. In the FNCC14, ICT is still not a separate subject, but it is rather deemed as a pillar of the whole curriculum, as well as a set of transversal competencies which students can obtain through multiple school subjects. As a result, code literacy is nested in different subjects, mainly in math and handcrafts.

In math subject, although the term *code literacy* is not used, its essence presents in various ways. For example, for students in grades 1-2, acquiring code literacy means "familiarising themselves with

the basics of programming by formulating and testing step-by-step instructions". In this way, students understand the basic principles of how the computer works—the step-by-step instructions given by algorithms. For students in grades 3-6, code literacy means "understand how decisions made by people affect the way technology works" and students start to "plan and execute programs in graphic programming environments." When students are in grades 7-9, the objective for them is to deepen their algorithmic thinking so that it can be applied in different problem-solving tasks. Students are hoped to "use their own or ready-made computer programs as a part of learning mathematics".

Code literacy is also blended in crafts subject from grade 3. Craft classes are considered as "Finnish own makerspaces" (Jaatinen & Lindfors, 2019). It is stated clearly in the FNCC14 that in grades 3-6, students "practice with functions produced with the help of programming, such as robotics and automation". In grades 7-9, programming is applied in the designing and producing craft products, e.g., in using embedded systems such as Arduino.

At the higher education level, study courses in computer science are run by a variety of universities and universities of applied sciences. Those courses are offered either as an obligation for computer science students or as elective ones for non-tech students. The content of these courses, therefore, spreads from basic to more advanced programming such as machine learning.

2.2.2 Code literacy education in Finnish non-formal education sector

Non-formal education, which typically takes place outside schools or training institutions, has become a primary educational force in the digital era (Romi & Schmida, 2009). It has more degrees of freedom and flexibility compared to the formal education sector (Tolppanen, Vartiainen, Ikävalko & Aksela, 2015), thus it can easily keep up with the radical change of technology. Non-formal education has also been considered a solution for the decline of the motivation of adolescents in learning sciences, because it brings more elements of fun and eases the tension from formal learning (Tisza et al., 2020; Tolppanen, 2015, Patrick & Mantzicopoulos, 2015). By increasing students' interest in sciences, non-formal education also has an impact on adolescents' consideration towards STEM-related career choices (Hsu et al., 2019).

The establishment of many coding schools (e.g., Code School Finland, Robotti Art and Craft school, Kodarit, to name a few) is a blooming phenomenon in non-formal code literacy education in Finland. These schools are organized as extracurricular learning centers and positioned in the Basic education in the arts sector in the Finnish educational system. Study courses implemented in these schools are based on and actively support the FNCC14, while teachers still have a certain degree of freedom to bring their own syllabus and pedagogy into practice.

Other non-formal programming learning activities, such as coding clubs and camps, are promoted by various educational organizations, e.g., LUMA Centre Finland, Koodikerho, Juniversity. The activities of this kind of learning center are disparate, depending on their resources, aims, and missions. For example, LUMA's objectives are to engage students of all levels in mathematics, science, and technology through the latest methods and activities of science and technology education. Koodikerho is a network of coding clubs across the country which are run weekly afternoons to help children learn the basics of coding in seven weeks. The teaching and learning facilities of Koodikerho clubs are supported by local schools. Voluntary teachers, consisting of IT teachers and parents working in the IT field, after signing up to the Koodikerho network, can freely use games and interactive exercises on the website of Koodikerho in their teaching plan. Differently, Juniversity offers after-school coding and electronics clubs and camps, as well as hosts the visits of school students to the Tampere University's Hervanta campus to explore programming hands-on activities. Those are just three exemplars among many other similar organizations existing now in Finland.

Youth work centers also play an important role in promoting code literacy education. Coding skills, along with other critical media skills, have been advocated in youth clubs since the late 1990s. (Tuominen, 2017). According to Lundqvist and Kiviniemi (2017), in the contemporary context, youth work should act as "an enabler", which means to facilitate and to provide youngsters with opportunities to experiment and enhance coding skills by themselves, e.g., Scratch or electronic construction kits such as LittleBits, Makey Makey or mBots should be introduced in daily youth centers.

Also, the enduring tradition of Finnish crafts education has a strong influence over the current maker culture in the country. The network of hackerspaces in Finland (hacklab.fi) consists of 15 hacklab members from all over the nation. Many city libraries have an integration of makerspaces. Some do-it-yourself (DIY) festivals have been organized in Finland, such as Wärk:fest (in 2012, 2013), Espoo Mini Maker Faire (in 2015 and 2017 with the cooperation of Wärk Association, Aalto Fablab and Espoo's Iso Omena Library).

2.3 Hacker culture and its implication for education

2.3.1 The notion of *hack* and *hacker*

Different generations have different definitions for the terms *hack* and *hacker* (Thomas, 2002). Nowadays, this concept is even vaguer as it has entered many areas beyond computers and technology. Any small improvement that brings convenience to life can be called a hack and anyone with a "problem-solving spirit" can be a hacker (Raymond, 2001). As such, the term hack is simply used as a

synonym of creativity or tinker. However, when tracing back the history of this concept since when it was attached to technology, we can draw more implications for education rather than just the idea of doing differently.

Throughout the development history of the hacker concept, it mostly points to two subjects: computer villains and computer geeks. Hacker and its connotation of criminality had not appeared in the mainstream media until the 1980s, with the premiere of the movie WarGames (1983) and the two books The Cuckoo's Egg (by Clifford Stoll, 1989) and Cyberpunk (by Katie Hafner and John Markoff, 1991). These media products made the image of hackers linked tightly with rebellious youngsters who break into computer systems for reputation and (or) personal gain (see e.g., Thomas (2002), Rayner (2018), Alleyne (2018)). This image of hackers becomes even more iconic in the media with the reputation of the international hacktivist group Anonymous, who involved in many cyber-attacks against governments.

However, hacker as *offender* is not the original meaning of this term. When the notion of hacking was first popularized in The Model Railroad Club—a students' electronics and programming club at the Massachusetts Institute of Technology in the 1950s, hackers were the ones who formed hacker culture as a subculture by sharing a philosophy of "sharing, openness, decentralization, and getting your hands on machines at any cost to improve the machines and to improve the world" (Levy, 2010), whereas computer criminals who circumvented the security systems were called crackers (Raymond, 2001). Hackers are purely interested in being intellectually challenged. To them, hacking means "making the technology accessible, open, free and 'beautiful'" (Thomas, 2002).

The Oxford dictionary offers a definition for *subculture* that "the behavior and beliefs of a particular group of people in society that are different from those of most people". This notion can be encountered in terms such as hip-hop subculture, cosplay subculture, or youth subculture. Steven Levy, in his authoritative book on the topic of hacking (1984; 2010), listed out six ethics ("Hacker Ethics") which represent "the behavior and beliefs" of hackers. These ethics have still been fundamental for current discussions about hacker culture and the application of this notion to non-technology areas. According to Levy (1984), the prerequisite for the operations of hackers is the "total and unlimited" access to computers or any other kind of tool. Only with hands-on practice can they draw critical knowledge about the machine and the world. Hackers have a pure interest in decomposing the system, sometimes messing it up, to get insights into how the computer works. They always lean toward tinkering, debugging, optimizing, and improving. Their longing for flawless products leads to the second ethic which is "all information should be free" so that anyone could contribute to and benefit from the best version of coded products. Instead of each individual building a computer program from scratch, the best version of the program should be public so that everyone can freely approach the code

and improve upon it. It gathers the intelligence and creativity of the whole community and emphasizes collaboration in creating products. It is related to the third ethic which is "mistrust authority—promote decentralization". This ethic addresses the equality of the hackers' system where no one gives order and there is no boundary between hackers and resources they need for creating and improving. However, decentralization does not mean that individual characteristics are not respected. As Levy stated in the explanation of the fourth ethic ("hackers should be judged by their hacking, not criteria such as degrees, age, race, sex or position"), the ignorance of personal traits is not necessarily rooted in the morality issues of hackers but it is because hackers put their attention more on the improvement and efficiency of the programs—they rather choose to prove themselves through the "art and beauty" that they create on a computer. Art and beauty mentioned in the fifth ethic, for hackers, mean creativity and optimization in solving problems. For them, the aesthetic in programming is when one can program the machine to do complicated tasks with very few instructions. It is considered a gamified challenge that each hacker delightfully takes on. These beautiful lines of code should be clear and easy for others to follow and tinker. Ultimately, as stated in the sixth ethic of hackers, computers can change their life for the better. Computers not only serve them as an absorbing hobby or a philosophy of life but it also is a means to impact and improve society.

2.3.2 Implications of hacker culture for education in the digital era

Hacker ethics draw plenty of profound implications for education in the digital era. Several scholars borrow the notion of hacking and hacker ethics to propose innovative pedagogies or educational philosophies in the digital age.

Smith et al. (2018) used the philosophical essence of hacking as a powerful theoretical metaphor to elaborate a new theory of *hacking education*. The researchers collect many educational practices as examples for how we can hack the schooling system, curriculum, pedagogies and highlight the intersection of technology and culture to reconceptualize our thinking towards education.

Wizel (2017; 2018) proposed the term "teachers as hackers" to indicate the autonomy of teachers when they act innovatively in the class without strict administrative control. The researcher synthesized the skills and habits of "teachers who hack" including idealistic and passionate, collaborative, adaptive, reflective, not afraid of taking the risk and having the ability to maximize the potential of resources and embracing uncertainty.

To bring the concepts of hacking closer to technology education, Menezes and Pretto (2019) develop an idea of "hacker pedagogy" which refers to the pedagogical practices carried out in hackerspaces. Specifically, they viewed hacker pedagogy as a pedagogy of engagement and listed out

four types of engagement in hackerspaces: (1) technical engagement (the free access to technological tools), (2) affective engagement (the relationship between people in hackerspaces, and between them and the practices they experienced in the place in which 'fun', 'curiosity' and 'sociability' are emphasized), (3) ideal engagement (the philosophical essence of 'hacking' is considered a way of living), (4) activist engagement (the voluntary of members when participating in hackerspaces and their engagement with social issues).

In a more holistic and systematic point of view, Aguado and Canovas (2019), stemming from their analysis of scientific and practical works containing two keywords *hacker* and *education* from 2013 to 2018, synthesized and proposed 15 principles of hacker education. Accordingly, students should be provided with opportunities to do things out of their passion, with having freedom as a fundamental (principle 1, 2). When joining a learning community, learners should be aware of each other, respect, and encourage diversity (principle 3, 4). Full access to learning resources, especially media products (such as computers, books, and the Internet) is a requirement (principle 5). All decisions and information of the learning community should be open to make the whole process more democratic and participatory (principle 6). Hacker education is the education that promotes action and active participation of learners; knowledge is shared in practice, in action and creative participation (principle 7). In parallel with stimulating creativity and critical mindset (principle 13), curiosity and making mistakes are encouraged as they are essential parts of learning (principle 8, 9, 10). Since each distribution and information within a learning community is open, hacker education promotes copying, reusing, and remixing so that anyone can build their knowledge and learning products from what already exists (principle 12). Hacker education, to some extent, has an activist and political nature (principle 11, 15). It is nonpartisan, but the way it operates based on the consensus of people brings some sense of activism. Ultimately, hacker education should extrapolate the physical learning spaces to penetrate into the way of living of students and teachers. The philosophical essence of hacking should be contemplated as a way of living (principle 14).

2.4 The characteristics of hacker culture in code literacy education

2.4.1 Hacking and making

Making is not a new concept coined from 21st-century education. It actually is the extension of the learning-by-doing concept (Halverson & Sheridan, 2014) which was grounded by Jean Piaget in his classic studies on cognitive development of children, and then elaborated by Seymour Papert who is considered as "the father of the maker movement" (Stager, 2017). However, with the advancement of

digital technologies in recent decades, this concept has been back in the spotlight. As a result, it has witnessed the flourish of makerspaces, hackerspaces and fablabs as promising pedagogical environments which are built based on the idea of maker culture. As of April 2021, there are 2412 hackerspaces all over the world according to the statistics by HackerspaceWiki.org. It can be said that the development of technology is the foremost condition for the making practice to come back as a trend in education (Richterich & Wenz, 2017; Blikstein, 2017). It is also the interdependence between the maker culture and digital technology that makes this concept distinguish from traditional maker culture which involves materials and crafts, and links more closely to hacker culture.

Hacker and maker culture share some characteristics, for example, learning-by-doing, collaboration, creativity, and the pure joy of exploration. In the "Maker movement manifesto", Hatch (2014) proposed nine points that make up the maker culture: make, share, give, learn, tool up, play, participate, support and change. These characteristics of making culture align with those of hacker culture that was discussed by, for example, Levy (1984; 2010) and Thomas (2002). At some points, when scrutinizing hacker culture, we can adopt the nature of making culture. Even so, it should be addressed that the most common point between these two concepts is that they promote hands-on experiences and free (or easy) access to technological tools. Many people join the communities such as makerspaces and hackerspaces for their free access to technological tools and devices, i.e., 3-D printers, laser cutters, or the newest technological toys (Halverson & Sheridan, 2014; Menezes and Pretto, 2019). Similarly, for hackers, the ability to freely access tools and devices is the precondition for their exploration (Levy, 1984).

There is still not a consensus on the remarkable difference between hackerspaces and makerspaces (González-González & Arias, 2018). Nevertheless, some researchers attempted to delve into the relationship between the two terms hacking and making. The term *making* is sometimes used as a replacement for *hacking* to prevent any confusion and concern related to the moral aspect of the term hack (Cavalcanti, 2013). Richterich and Wenz (2017) refer to an online discussion on the topic of hacking versus making to present four points of difference between these two concepts. Firstly, all hackers are makers, but the reverse is not surely true. Secondly, the term making creates the feeling of "family-friendly" that should be used to attract the involvement of children, parents, and other members who are familiar with the discourse of hacking as an illegal act. Thirdly, making is a commercial term; in other words, making is a commercialized hacking activity. It leads to the fourth difference that hacking is treated as a subculture and making as a commercialized practice.

2.4.2 Hacking and playfulness

Play, playfulness, and playful learning are concepts that have caught much attention of educators and educational researchers. Playfulness in education can bring many benefits to students, e.g., encouraging creativity, imagination, risk-taking, and enjoyment (see e.g., (Lieberman, 1977; Rice, 2009; Whitton, 2018).

While playful learning refers to an educational approach in which learning takes place through play, the state of playfulness is gained not necessarily through play. Play is an activity, whereas playfulness points to an attitude when someone engages deeply in context and objects and at the same time respects the purposes and goals of those contexts and objects (Sicart, 2014). Hence, playfulness should not be merely viewed as the joy and fun obtained through play.

Hacking has a long history in terms of relation to playfulness. Initially, when being flourished as a tradition of students at Massachusetts Institute of Technology (MIT), hacking meant pranks. For hackers, playfulness can be manifested by their enjoyment of immersing themselves in the machines. When it comes to code literacy education, we can borrow the framework of "Playfulness design elements" elaborated by Meij, Broerse and Kupper (2017) to examine the presentation of playfulness in learning to code.

Accordingly, playfulness is associated with narration, imagination, action-reflection, co-creation, experimental space, focus, and stimulating guidance. To attain playfulness, the learning instructions should be given clearly yet be open to imagination and immediate reflection. The students should be provided with opportunities to collaborate while not being afraid of being judged. Playfulness, therefore, is more of self-directed learning and creativity than playing itself.

2.4.3 Hacking and collaborative learning

As a subculture, hacker culture is established by the members of its community. Bringing hacker culture to code literacy education is to accentuate the collaboration in learning activities.

According to Laal and Laal (2012), collaborative learning is an umbrella term that traverses from the joint efforts of learners in solving a problem to more structured teamwork (cooperative learning). Collaborative learning involves groups of students, either a small group or a class, working together to perform a learning activity, e.g., solving a problem (MacGregor, 1990; Dillenbourg, 1999). It can also be the togetherness of students and teachers, but the emphasis is placed on the active work of students instead of teachers' explanation or presentation (Smith & MacGregor, 1992). Some researchers accentuate the social nature of collaborative learning by highlighting the idea that it is through students talking with each other that learning occurs (Golub et al., 1988; Gerlach, 1994).

In this research, we delve into the broad understanding of collaborative learning presented by Panitz (1999). Accordingly, collaborative learning is related to "a personal philosophy" rather than just "a classroom technique". In all situations when people (or students, in this research) come to work together, any members' abilities and contributions are respected. Students hold an equal extent of authority and responsibility, which against the idea of competition in which there are some individuals being considered better and dominating other group members. Also according to Panitz, this philosophy of collaborative learning can be applied in many different situations of learning, e.g. in classrooms, at committee meetings, within community groups, or in families.

To guide the analysis of collaborative learning, Panitz (1999) proposes five principles of it: (1) the results of working together should be a better understanding than when each member works independently, (2) this better understanding is contributed by spoken and written interactions, (3) the relationships between social interactions and that better understanding can be aware of, (4) that better understanding may contain the idiosyncratic and unpredictable elements, (5) members voluntarily participate and freely enter into collaborative working.

2.4.4 Proposed framework and research question

To guide the analysis, an initial framework that specifies the characteristics of hacker culture in a coding class is proposed as in Table 1 below. The elements included in the table are based on the synthesis of literature in previous sections.

TABLE 1. Proposed framework for analysing the characteristics of hacker culture in code literacy education

Learning environment	• students' access to learning tools and devices • student diversity in the class (genders, levels, motivation and interest)
Learning activities	• hands-on activities and differentiation • playfulness • students' collaboration • impetus for tinker and creativity • teachers' perception of hacker culture
Learning outcomes	• coding products • students' perception of computer programming

The research question of this study is: **How are the characteristics of hacker culture integrated into code literacy education?**

To answer the main question above, two sub-questions are put to guide the research:

1. How does code literacy education support digital literacy education through enhancing students' computational thinking?
2. How is hacker culture perceived as a concept in education?

3 METHODOLOGY

3.1 Research design

The research question of this study is "How are the characteristics of hacker culture integrated into code literacy education?". To answer the question of *how*, the current study is carried out under the interpretivist paradigm. The qualitative approach is recruited to collect non-numerical data. Educational experts and practitioners' in-depth insights and opinions regarding the topic are gained through thematic interview and thematic analysis method.

Research paradigm is "a set of fundamental assumptions and beliefs" which defines how a social phenomenon is perceived by the researcher and forms a thinking framework that guides their research behaviour (Jonker & Pennink, 2010; Wahyuni, 2012). In other words, a research paradigm is a lens through which the researcher scrutinizes reality. This study is conducted within the interpretivist paradigm, instead of positivist, transformative and pragmatist ones. The fundamental belief is that each educational expert and practitioner experiences teaching in their own way, which then forms their subjective interpretation on the characteristics of hacker culture in code literacy education. By synthesizing different perspectives on the topic, the current study will capture a holistic and colorful picture on how hacker culture is blended into different education sectors.

Qualitative research is the most beneficial approach for the interpretivist research paradigm. By seeking for non-numerical data, qualitative research helps to understand "how people interpret their experiences, how they construct their worlds, and what meaning they attribute to their experiences" (Merriam, 2009).

Thematic, semi-structured interview is a qualitative research method, which is "sufficiently structured" to focus on the research questions yet flexible and versatile enough for interviewees to bring new meanings to the research topic (Galletta & Cross, 2013, p.24). Beside a set of questions which was prepared to address the prominent themes of the research topic, the follow-up questions were asked during the interview, so that the participants could clarify their ideas or provide new insights.

3.2 Research context and participant recruitment

This study was conducted in Finland within one year (from April 2020 to April 2021). All the interviews took place within one month (from the 12th of January to the 12th of February 2021). This is the period of time when the COVID-19 pandemic and the social distancing strongly hit education around the world. Therefore, almost all the phases of this study, including the interviews, were done online. Besides, as of 2020, the new Finnish National Core Curriculum (FNCC14) has been in effect for roughly four year. This duration of time seems to be long enough for the educational experts and practitioners to reflect on the pros and cons of integrating computer programming into the FNCC14.

The purposeful sampling method was employed for participant recruitment. Purposeful sampling method helps with "selection of information-rich cases for the most effective use of limited resources" (Patton, 2002; as cited in Palinkas et al., 2015). It focuses on recruiting the participants who are "especially knowledgeable about or experienced with a phenomenon of interest" (Creswell & Plano, 2011).

There are some specific criteria for participant recruitment. Firstly, the potential participant should have several years teaching or researching in the field of ICT education, especially in programming, computational thinking and/or maker education. It could be fruitful if they have a deep understanding of how ICT is integrated in the FNCC14 and how it takes place in practice. Secondly, they should work closely with youth (9-18 years old youngsters) to have practical knowledge about how youngsters react while learning computer science and programming. Thirdly, the number of participants should be balanced in terms of genders so that the gender bias is eliminated. Finally, as this study aims to understand Finnish code literacy education from both formal and non-formal education sectors, the participants should have different working experience in various education contexts (e.g., schools, youth work centers, universities, clubs).

After identifying the criteria for participant recruitment, the researcher of this study contacted several experts so that their networks can be utilized to find potential interviewees. P1 and P2 were introduced to this research by the coordinator of a non-formal learning organization. The contact of P3 and P4 were found while the researcher was searching on Google for information about their workplace, and they were approached individually via email after that. P5 and P6 were suggested by the supervisor of this study. P5 had also been the guest lecturer in a seminar in which the researcher of this research took part in.

An invitation was sent to each potential participant via email, along with a short explanation about the aim of recruitment and how the interview would be conducted (including estimated length, video conference platforms to be chosen and main themes of the interview questions). The potential

participants decided on the suitable time for their interview, as well as their preferable online conference platform (Zoom or Microsoft Teams). At least one week before the interview, they received (via email) a list of interview questions, a consent form and a link to join the online interview (except for the participant P6—the consent and questions were sent only one day beforehand, because the interview was scheduled only 2 days in advance). This work was also a reminder to the participants about their appointments. The interviewees were required to sign and send the consent back to the interviewer before the interview. Only after the recruited person signed the consent and sent it back to the researcher did the recruitment process truly complete.

By sending and receiving the questions beforehand, the interviewees were better prepared so that they would be more confidently talking and could bring the most out of their thoughts to the conversation. Because both interviewer and interviewees are not native English speakers, there was a possible problem that two parties have different understanding of one term. For example, one informant was not sure about the way she understood the concept of "tinkering" used in this study, which urged her to send an email to the interviewer to clarify the definition of this term. Hence, the preparation phase with questions sent in advance helped the researcher and participants reach a consensus in some terms and concepts used in the interview. However, a probable problem of knowing the questions beforehand is that the participants could prepare or even make up their answers. Nonetheless, as the interviews are semi-structured, the informants could not prepare for all questions they would encounter, especially for the questions requiring them to describe or explain in detail what they are talking about. Therefore, the trustworthiness of the interviews was still guaranteed.

Within one months, six interviews were conducted with six Finnish teachers and educational experts. The list of participants is as follows:

TABLE 2. List of participants

Participant	Expertise/ teaching experience	Duration of experience	Gender
P1	Electronics and coding club instructor	About 3 years	Male
P2	Coding club instructor and organizer	About 4 years	Female
P3	Educational expert (Researcher, ICT teacher,	About 6 years	Male

	head of ICT department in a secondary school)		
P4	Educational expert (Researcher, makerspace instructor, coding school coordinator)	About 5 years	Female
P5	Youth work expert	About 20 years	Male
P6	Educational expert (Researcher, university lecturer)	About 30 years	Female

Among six participants, three people were based in Tampere, two in Helsinki, and one in Oulu. As presented in the table above, four participants were working in the non-formal education sector and three of them were serving in the formal education sector (the participant P4 is working in both sectors). In terms of gender, the number of informants is completely balanced (three females, three males).

3.3 Data collection

Interview questions were generated based on the knowledge gained from literature. It is hoped to not only explore as much relevant information as possible but also be open enough for the interviewees to bring new insights to the conversations.

To test the understandability and the openness of the questions, three pilot interviews were conducted with three people. Two among them are Vietnamese ICT teachers who had had five and ten years of experience in teaching programming and robotics, as well as managing their school's makerspace in Vietnam. Although those teachers have a strong background in ICT education and maker culture, they did not have any experience with Finnish ICT and education context. Therefore, one programmer, who had been working in the field of ICT in Finland for 5 years and had experience in an EdTech (education and technology) start-up in Finland, was recruited to complement the pilot phase. The pilot interviews flowed smoothly and all the pilot participants found the questions understandable. One of them suggested that one more question on how the teachers measure students' performance should be added, since it could help the interviewees think more deeply and carefully about what kind of student should be defined as "talented students" as posed in the pilot questions. The final set of questions is included in the Appendix 1 of this thesis.

Due to the social distancing caused by the COVID-19 pandemic, the interviews were conducted online using Microsoft Teams and Zoom video conferencing platforms. Online interview, or e-interview, is the interview implemented through computer-mediated communication (Salmons, 2015). This way, both the respondents and the researcher remain in their comfortable and personal space (e.g., their home) while participating in the interviews, and it prevents them from the feeling of being physically imposed by each other which can be a problem with the on-site interview (Hanna, 2012). The interviews in this study were carried out in a synchronous way (real-time), in which the interviewer and the interviewee are online at the same time to directly ask and answer; it is different from asynchronous interviews (non-real-time, e.g., via emails) (Janghorban, Roudsari & Taghipour, 2014). Throughout the synchronous interview, the interviewer can observe and understand the feeling of interviewees through their voices or their facial expression, so that the way of posing questions and giving answers can be slightly modified or clarified for clearer understanding. The recording function of Zoom and Microsoft Teams was utilized so that both the video and audio parts of the interviews were conveniently recorded.

This research followed the guideline of Galletta (2013) to divide the interview into three main segments. The opening segment aimed to establish the rapport between the interviewer and interviewees, and to open the conversation in a pleasant atmosphere so that the respondents could comfortably tell their *stories*. The participants also were asked about their background information (expertise and duration of working). The middle segment is the main stage of the interview, in which in-depth questions about participants' experience with code literacy education were inquired. The follow-up questions were asked if there was something unclear in the answer or if the interviewer desired to explicit more ideas from the respondents. Sometimes the researcher repeated the answer of the interviewees in order to not misunderstand their opinions. The concluding segment, according to Galletta (2013), should provide participants with an opportunity to reflect their prior narrative in relation to the theory-driven questions. At this stage, some questions related to participants' familiarity with the concept of hacker and maker education were asked. Finally, the interviewer expressed gratitude to participants for their contribution to this research. Each interview lasted roughly 30 to 45 minutes.

After each interview, the questions were revised so that the next interview would proceed more smoothly and efficiently. All the remarkable notices were also jotted down and reflected right after the interview when the interviewer's memory is still fresh. The analysis was done soon after each interview.

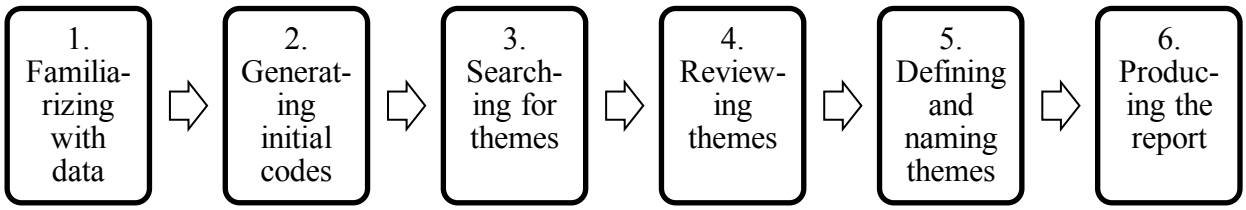
3.4 Data analysis

The data analysis process is not completely separate from the data collection phase—these two processes mutually reinforce. After each interview, the collected data was initially analyzed; thereby, the researcher can draw on experience and pay more attention to the knowledge that can be explored more deeply with the next participants.

Before being analysed, each recorded audio was listened to at least once so that the researcher can get immersed in it. A text transcription was generated automatically with the help of sonix.ai — an automatic transcription online software. The transcription then was revised and edited for accuracy. During this revising phase, the researcher wrote down some comments about each interview (for example, what the participants are most excited when talking about, what they were not sure about, what problems they addressed). In total, there were 61 pages of transcription for analysis. After getting all necessary transcriptions, all recordings and transcriptions were permanently deleted from sonix.ai platform.

The analysis process followed the guideline set by Braun and Clark (2006) with six steps as follows:

FIGURE 1. Six steps of data analysis (Braun & Clark, 2006)



Step 1: Familiarizing with data

As the interviewer was also responsible for analyzing data, the prior knowledge and familiarity with the data set was already available. Even though, all the transcriptions were read through multiple times so that the repetitive information and the initial patterns were noticed and jotted down. Furthermore, as the analysis is theoretical-driven, all the answers related to the main themes were marked so that they could be coded more carefully and critically in the next phases.

Step 2: Generating initial codes

The initial coding was done with the assistance of atlas.ti—a data analysis software supported by Tampere University. The version for desktop was provided by the IT Helpdesk service of the university,

and it was installed in the personal computer of the researcher. Six transcriptions were uploaded to the platform for coding. Each of them was read line by line carefully; when one important word (or phrase, or sentence, or paragraph) was spotted, it was highlighted, and code(s) was added to it. An important word (or phrase, or sentence, or paragraph) was defined as being relevant to the topic, or being strongly emphasized by informants, or triggering new ideas.

One word (or phrase, or sentence, or paragraph) can go with multiple codes. For example, the following is what P6 said about how computational thinking and programming is presented in FNCC14:

> "So you can see from there [FNCC14] that logical thinking and algorithmic thinking have been mentioned, but it would require more detailed instructions that what we mean by [pause] Every teacher is free to interpret what he or she thinks that it is [pause]. So it needs some more specific bullet points that 'this has to be covered, this has to be covered'. Now it's too vague. It's not clear enough. The instructions are not clear enough. So in a sense, I think the problem is that it needs to be more specific. It needs testing, some standards, and it needs to be more detailed." (P6)

It can be seen that different codes come out from this paragraph. The whole paragraph was coded as *the vagueness of instructions in the FNCC14* and the sentence "Every teacher is free to interpret what he or she thinks that it is" was additionally coded as *teachers' autonomy in teaching programming*. One more insight gained from this answer was that even though the term *computational thinking* is not directly mentioned in the FNCC14, the terms "logical thinking" and "algorithmic thinking" were pointed out as the representatives of that concept, so they were coded as *computational thinking in the FNCC14*. (It should be noted here that this example is used only for illustrating the research method; it is not a representative for the main theme of this study).

After the phase of initial analysis, 81 codes were generated. All of them then were revised, compared, merged and unmerged. Finally, 64 final codes were produced.

Step 3: Searching for themes

64 final codes were initially sorted into 15 themes. Those codes were grouped based on what specific topic they could support. For example, the codes *boys and girls: difference in number* and *how to attract more girls to coding activities* could fall under the theme of *the gender gap*.

Step 4: Reviewing themes

In this step, themes emerged from step 3 were reviewed and re-grouped. Some themes were combined into one. For example, in the first place, *computational thinking* and *digital fabrication* were two separate themes. However, both of them play a role in setting up the foundation for teaching

practice, and some participants talked about computational thinking in a relation with digital fabrication, so those two themes could be grouped into one.

Consequently, the number of prominent themes was reduced to 7. The list of themes can be found in Table 3. Theme 2 and theme 7 are the inductive results of the analysis, whereas theme 1, 3, 4, 5, 6 are based on the theoretical framework developed from literature review.

Step 5: Defining and naming themes

Each final theme was named so as to cover main findings the theme presents. They also served as a guide for the Findings chapter to be written up.

TABLE 3. List of themes

Theme 1: Code literacy, computational thinking and digital fabrication
Theme 2: Code literacy education in FNCC14
Theme 3: Educators' perception of hacker culture
Theme 4: Students' access to tools and devices
Theme 5: Students' diversity in a programming class
***Subthemes**:* (1) Students' different levels and motivation, (2) Specialized students, (3) Difference between boys and girls' interest and ability in programming activities, (4) Students' perception of computers and programming
Theme 6: Hands-on activities supporting students in learning to code
***Subthemes**:* (1) Collaborative and indigital learning activities, (2) Playfulness, (3) Tinkering and creativity
Theme 7: Teachers as facilitators

Step 6: Producing the report

Among all the quotes, the ones that most vividly illustrate each theme and code were remarked. Then, an outline for the report was generated so that it could deliver a coherent content. The report is presented in chapter 4 (Findings) of this study.

4 FINDINGS

4.1 Code literacy, computational thinking and digital fabrication

The informants agreed that computational thinking is one of the critical skills for youngsters to live in the 21st century. Three participants shared the idea that computational thinking is extended beyond the area of computer programming and becomes a life skill which helps students handle complicated problems in their life more logically and optimally. An informant stated clearly that "nowadays there is a blurring boundary between programming skills and more general 21st century skills", and the concept of computational thinking "is not just related to programming concept or skills, but more like attitude or metacognitive aspect" (P4).

Another participant addressed that the concept of computational thinking should not be overly simplified: "It is not about students learning to think like a computer; that's just one part of it" (P3). It rather refers to a creative way of thinking while working with a computer. That interviewee raised a discussion:

> "The other part I think about computational thinking is being creative about it. Like, when you know a computer can perform this type of task, then the interesting question is that 'ah, what can we do with it?'. You have a device that performs a set of functions and does it repeatedly without errors. Then, how can you make it do more interesting stuff?... And I think it is the heart of the hacker culture".

According to the participants, youngster's code literacy can be enhanced through computational thinking development, and vice versa. "I think that learning or studying programming brings you computational thinking or helps you develop computational thinking. And having computational thinking will help studying programming or developing programming skills. It's vice versa, I think", said P4. This resonates with the opinion of P6. In addition to the idea that "computational thinking is the very essence of programming", P6 made a comparison between the ability of mathematics and computer programming in developing students' computational thinking skills. From her viewpoint, mathematics heavily supports computational thinking, and coding can benefit it even more. With computational thinking, students can learn to solve problems more efficiently and optimally; and "this kind of thinking comes from programming more than mathematics".

Computational thinking skills can also be enhanced through digital fabrication. For example, when programming the controllers inside a fabrication product, students simultaneously learn about components of computational thinking, such as modularization or iteration: “In digital fabrication, we use a lot of technologies, and inside of that we also use microcontrollers and we need to program it. So, by programming the microcontroller, they [students] are in a process that will enhance computational thinking”, said P4. To illustrate that idea more explicitly, P4 gave an example that when students design a box with a sensor inside, they have to decide which electronic parts are needed, which steps should be followed and in what order students should take those steps. During that process, students can learn the concept of modularization. Another advantage of digital fabrication is that it could deeply engage students by giving them the joy of playing with physical devices. When being fully immersed in that activity, students can learn computational thinking naturally, as P4 put it:

> “I want to highlight that digital fabrication and maker culture, it would engage students in doing it because they like the physical material and play with physical material doing their own project. And maybe they don't realize that they are using it [computational thinking] or developing it; but in the process, they are very engaged. And in that process, they are developing computational thinking naturally.”

That participant also stated that because computational thinking is an abstract concept, it is more easier for students to grasp its notion when the physical devices are utilized:

> “Computational thinking, it's really like a broad and abstract concept. But like when you work on the physical device, like physical materials, it will help you to understand that abstract concept in practice.”

4.2 Code literacy education integrated in FNCC14

The integration of computer programming into math and craft subject in the Finnish basic education brings out many discussions.

According to the informants, Finns have a tradition of handcrafts and do-it-yourself (DIY) culture, which is a fundamental for the development of hacker culture and varieties of hands-on programming activities. The interest of many teachers in fabrication activities help them absorb the idea of maker culture and digital fabrication more easily, as one participant shared that:

> “I think it's really great that they [teachers] are already interested in fabrication and making. And it's good that the DIY culture has been in Finland for a long time. People like to do something with hands and they are really talented. And in schools like… wood work, technology work, technology subject and technical subject, and also handcrafts… These kinds of things can be easily connected with the maker culture.” (P4)

This comment resonates with thoughts of P5 on code literacy education in youth work, with robotics and hands-on activities, as a continuation of the Finnish traditional approach to handcrafts learning:

> "In youth work, traditionally, there have been lots of different clubs for handcrafted works or drawing or things like that. So I think that ICT or technological education is just the sort of the continuity to those traditional ways of doing youth work and how to provide non-formal learning for youngsters." (P5)

In formal education sector, the integration of coding into math subject in FNCC14 is reasonable because mathematics and computer science strongly support each other, as a respondent put it: "Still, I have a vision that it benefits both subjects, they benefits in cooperation, computer science benefits mathematics and mathematics benefits computer science" (P6).

However, this integration is still in "a transition phase" (P6), which means that there exists some weakness in the implementation. The main problem that most of the participants pointed out is the vagueness of the instruction for teachers on how to bring computer programming into their classes. An interviewee shared that:

> "A couple of years ago, we introduced the new national core curriculum in Finland. And that included programming as… well, not a subject on its own, but inside mathematics. Our pupils were supposed to be doing some kind of programming exercises, but it's pretty vague how to implement that." (P3)

This vagueness is a double-edged sword. On the one hand, teachers have their autonomy when they can "freely interpret what they understand" (P6) of programming education mentioned in the FNCC14. On the other hand, it affects the consistency and equity when different teachers bring different amounts of computer programming into their math and craft classes. This problem will be elaborated more in the section 4.7 of this chapter. Besides, despite being a mandatory element in basic education, computer programming has not appeared in any important test or exam. As a result of all these downsides, coding has not received enough attention as it is supposed to, as an informant said that "it can be easily neglected in math classes" (P6). To sum up the opinions regarding the limitation of code literacy education in Finland at the moment, P6 pointed out that "I think the problem is that it needs to be more specific. It needs testing, some standards, and it needs to be more detailed."

Also, in the FNCC14, to reinforce students' code literacy, computational thinking is fused in math subject. The phrase "computational thinking" is not directly mentioned; instead, it is referred to with the words "logical thinking" or "algorithmic thinking". However, as other instructions in the FNCC14 regarding computer programming, these notions are not explained clearly enough: "They have been mentioned but it would require more detailed instructions", said P6.

When it comes to the comparison between formal and non-formal education sectors in terms of implementing teaching initiatives such as hacker and maker education, almost all participants agreed that non-formal sector is more flexible. One interviewee stated:

> "So trying to implement something new and something like that, a lot of teachers think about extra work. Even though it is in the curriculum, it is very, very difficult. Even for someone like me who really knows something about it [coding and hacker culture], and thinks it is important, it's still very hard to implement in regular classrooms in school. So it's definitely way easier to do it through, like, hobby stuff". (P2)

4.3 Educators' perception of hacker culture

When being asked about their own definition of the term *hacker*, almost all participants found it difficult to offer a clear explanation. For them, this term may refer to both *criminal* and *explorer*. Although all participants reflected upon both the positive and negative sides of the concept of hacking, they only referred to the positive one when discussing more deeply into the educational aspect of this notion. Hackers are self-directed learners who are eager to explore and optimize the tools and devices at hand. P6 gave the most definitive answer, which also covered the ideas of other participants:

> "My definition of a hacker is that it's a type of tinkering people who is not afraid of any challenges and actually is trying to fight this kind of trial-and-error method of doing some… I think that… Not maybe using it always as the manner as it's meant, but inventing other ways of using the thing. So this kind of thinking out of the box… For example, you give an API and you think that this programming API should be used this way and this way. And suddenly you notice that these hackers, they have found multiple other ways which were not intended. So you are like "ah, you can use it that way also". I think that means to be self-directed". (P6)

Also, according to that informant, the weakness of many hackers is their tendency to focus on fixing the immediate problem more than conceptualizing it:

> "But, I think that sometimes some hackers, they remain at that level. So they never try to make it further. So some hackers, they don't develop further. They are always eager to solve it and see everything as a puzzle that they have to conquer immediately. But they don't try to, like, think that 'what is the pattern, what I can learn from this'. Sometimes they miss this reflection phase that they can learn from and they can make it more abstract. They should have made a rule. So I think that they sometimes miss that." (P6)

With regard to hacker culture, two interviewees stated clearly that from their viewpoint, hacker culture is not just about coding, it rather presents in every facet of information technology. According to P2, hacker culture points to the ethics and mentality when someone works with information

technology. That participant expressed that if she needed to introduce this concept to students, she would rather steer hacker culture away from the act of breaking into the database or any computer systems. P5 extended the definition of hacking to other areas of life, rather than sticking it to the field of technology: "Hacking is not just about coding or anything like that. I think picking a lock is also a sort of a hack".

Four out of six participants, albeit using different words, shared the same point that hacking is "the art of taking things apart and seeing how they work" (P1). Throughout this self-learning process, hackers are not afraid of taking on intellectual challenges and constant trial-and-error repetition. Interestingly, they also stated that, by profoundly understanding the mechanics of tools and devices, hackers can do creative work with those things. As asserted by P4, this way of thinking and working lies "at the heart of the hacker culture". Other informants shared the same ideas:

> "I think hacking is that you take something, you put it into pieces, you learn what makes it work and then you think 'Well, how can I make this work differently?' or 'How can I make this work the way I want it?'" (P5).

Another critical element of the hacker culture is the openness and collaboration among its members. As P3 put it:

> "When it comes to the hacker culture, one of the key things is that everything is shared openly. There are usually passionate discussions about how things work, how you can build stuff, or modify it."

When it comes to the comparison between hacker culture and maker culture, the participants offered their own explanation for the fact that maker culture is promoted more widely than hacker culture in current times. They agreed that these two subcultures "have similar stories" (P5), however, maker culture is the fresher and more current term, which makes it can approach more people than hacker culture does. Maker culture can reach a wider audience also because this term sounds more positive. As stated by P5, even when white-hat hackers are mentioned in public discussion nowadays, the term hacker still arouses the negative vibes:

> "I think maybe maker culture seems to be more positive than hacker culture. I think hacker culture is something that most people are not really familiar with. When we talk about hackers or hacking, usually people think about breaking into the information system and something criminal".

In addition, hacker culture seems to "attract a certain type of people, like nerds and geeks" (P6), which forms the perception among people that hacker culture is not something for them. All in all, both hacker and maker culture need to be paid attention in current times, as stated by P5.

One informant honestly shared that she hesitated to promote the concept of hacker culture because of the gender issue. From her viewpoint, hacker culture is "maybe a bit exclusive for girls" (P6). That interviewee commented that hacker communities are now dominated by males. Such communities could bring the sense of belonging which is beneficial for boy students: "I know that for some boys, that is very nice to be part of the team and part of the community. And it brings some learning affordances that are difficult to reach in other ways". However, the downside is that girls tend to be excluded in the community which is dominated by boys.

4.4 Student's access to tools and devices

Block-based and graphic-based programming language, such as Scratch, is mentioned by most of the participants. According to them, this type of programming language is the most favourable for students, especially young pupils, when they are at the very beginning stage of learning to code. More sophisticated text programming languages, such as Python, are mentioned when the interviewees talked about students who are gifted or interested in coding. Other programmable robot kits and devices were also mentioned by participants during their conversations, namely Arduino, ATtiny85, Micro:bit, Lego Mindstorm, Sphero Bolt, Cozmo, RaspberryPi. Most respondents shared that their students have eagerness to get their hands on the devices, as an informant indicated: "I would say they are not afraid of new equipment. They are really 'brave' when it comes to trying new robots and things like that" (P2).

Despite the abundant availability of programming tools and devices, the participants tend to choose the most flexible and inexpensive ones for their classes. The flexibility of the tools and devices means that teachers can use it in different teaching situations. In other words, it helps teachers with differentiated teaching, as P2 put it:

> "One of the most important things, always in the class, is differentiating the tasks that kids do. So if a students is in an advanced level, we would give them additional tasks which they can do better, for example they may do different projects with Micro:bit 1. I really love Micro:bit, because you can switch between Python and block [block-based programming language] all the time; so sometimes we like to teach them Python if they are familiar with that sort of thing." (P2)

P2 also addressed another reason for her preference for Micro:bit is that its resource is available online, so that "kids can do a bunch of stuff online and test it out online without the physical device". Thus, the flexibility of tools and devices could help students and teachers surmount the difficulties of lacking physical resources and make the most out of their materials.

Similarly, the opportunity of students in accessing programming tools and devices is increased when the equipment is inexpensive. By utilizing inexpensive tools and devices, students can possess their own learning kits and freely explore with such equipment, as P1 put it:

> "I tried to use really inexpensive things such as Arduino which are like 2-3-euro pieces. And in some projects, we use ATtiny85 with 1 euro per chip. Trying to keep it accessible… So, when we do electronics projects, the students can have the end results for themselves. We don't have to disassemble things to use those parts for the next groups. Students can own the things that they made. The project ends up in a way that if the kids are interested in it, they can reuse the own parts that they have put together to do something else." (P1)

Similarly, another informant asserted the benefit of inexpensive tools and devices for students' access at home: "Microbit is also pretty cheap, so a lot of kids end up buying it for home too" (P2).

Students sometimes have a certain degree of privilege to decide which tools and devices should be equipped in the educational institution; in that case, educators are the ones who consider and comply with students' demand. One participant shared that: "If there is an order for it, if the youngsters want that sort of thing, then that's also always something that the youth workers have to react to. So they say we want to code, then youth workers have to think, OK, how can we make this happen?" (P5).

Typically, students can borrow or freely use the tools they need in school or any other non-formal learning centers. One informant shared that:

> "And if they don't let youth take it [tools and devices] to home, at least it is usually free to use at the youth centers, so the youth can come to the youth centers to use the programs and devices there. So usually it's at least like that. Sometimes they even give them to youth, like 'OK, you could take this for a week' or 'you can take this for two weeks and please bring it back then', some youth workers even do that." (P5)

However, there are still some limitations in students' using or borrowing equipments. For example, students can use the tools mainly in the class in school-hours. The borrowing system mainly serves teachers, as one interviewee shared:

> "We have a borrowing system. But I don't think any of the students actually borrow those things because it's mainly for teachers. So it's not acceptable for the students to take these stuff home, so they mainly use… especially like in the robot class, they just use them in our club and that's it" (P2).

This comment was echoed by the idea of another informant that there are some restrictions to students' using tools and devices, and the teachers hold decisions on this problem:

> "If the schools have those devices and students can use it, it would help students to do some projects. But I think that it mostly depends on the teachers. If teachers want

> to do that or not, like students cannot really… like it's not common that the students can decide 'OK, I want to do this and this at school'. If they have some kinds of club, or if they have some free spaces that the students can come and do something, it's possible. But in school context, I think it's mainly the teachers' decision if the student can use the devices or not. Of course, easy access is important, but access plus teachers' permission, or encouragement, or such things, are also important. And outside the school context, it's like… Of course, if they can access Fablab, it makes it easier for those who are interested in fabrication or in technology in general" (P4).

From a broader perspective, students' access to tools and devices depends heavily on the resources of each educational institution and even the economic status of each Finnish municipality. For example, for some educational organizations which are autonomous in operation to some extent, such as youth work centers, the quantity and quality of tools hinge on the human and financial resources that the organization possesses. The financial support coming from local administration is based on the budget application of the youth work centers, as well as "what sort of organization and where it's located" (P5). Particularly, for code literacy education (and ICT education in general), the youth workers decide to what extent ICT is integrated in their educational project and call for a fund for that project—"it is more like funding the project" (P5). More importantly, the knowledge of youth workers themselves is the critical factor, which determines what kind of tools would be used in a youth center. One participant concluded:

> "One big point is the resources: Does a youth work organization have enough resources? Do they have enough money or do they have the know-how and the what-to: what to buy and to use in ICT education?" (P5).

From the municipality level, one interviewee asserted:

> "I think it depends much on the economic situation of the city. So, for some cities, they have everything you can imagine, for example, in Kauniainen, they have a 'dream school', it's called Unelmakoulu. They have everything—every imaginable thing. And I think that also the normal schools that are working with the university, they have good facilities. But in some poorer cities: No. So, I think that is the matter of economic consideration, it reflects a lot on what is offered" (P6).

4.5 Student diversity in a coding class

This theme examine how student diversity is embraced in a coding class. The term *student diversity* mentioned in this category refers to the difference between students in ages, interest, ability, and whether there is difference between boys and girl students in their ability and interest in regard to coding activities.

Students' levels and interest vary in a coding class

When being asked about whether there is a wide age range of students in their classes, most of the informants said "no". In the school class and university course, students are typically at the same age. Similarly, even in the non-formal setting, i.e. coding clubs and fablabs, students are arranged based on their age group. Both P1 and P2, two club instructors, shared the same experience that the students coming to their club are grouped by age:

> "Usually the age ranges are like… for lower elementary age group, from grade 1 to grade 3, so 6 to 9 years old; then upper elementary students who are like from grade 4 to grade 6, so 10 to 12 years old." (P2)

That said, students' age is not the factor affecting the difference in students' interest and ability in a class. However, it can be seen from the answers above, the level of students are somehow assessed based on their age group, i.e., students in the lower grade are automatically classified in the lower level. This results in skills and interest gaps among the students in the class.

During the interviews, the students who either are at a higher level or have more interest in computer programming than their peers are called as *gifted students, talented students* or *specialized students*. All respondents agreed that there are always students who are into coding more than their classmates. Those students could attend the same type of coding class before, even multiple times, so they have previous experience with coding. P1 spotted talented students based on "how often they ask for help, what kind of things they ask for help, and what they have created". P2 shared that, for some students, "it's very easy for them to figure things out on their own", whereas some others "need a lot of help" and "it's sometimes super difficult to get them comfortable of just trying things on their own, and they just basically copy the code". This skill gap is truly remarkable.

When being in the same class or learning group with lower-level peers, the specialized students may easily feel bored or lose their patience. As one participant put it: "I remember one time we have a class doing games with Scratch. There is one kid basically doing the everything thing multiple times. He attended the same or similar class previously at least 2-3 times. So, he ended up being quite bored" (P1). Another interviewee shared: "There's always some of them, well, who are like, 'OK, I'd like to know the hard stuff right now, so please go into the point'. There's always those types of youth or kids in a bunch" (P5).

All participants shared their own opinions with regard to nurturing talented students. Those opinions will be presented in section 4.6 of this chapter.

Student gender differences

Three out of six participants indicated clearly that there is no difference between boys and girls' ability in learning programming. Three other interviewees shared that insight, even though they did not mention it directly, as a participant put it: "I have to say I can't be sure about that. I don't have any facts of this matter. I would like to think that there is close to no difference. I like to think that at least in Finland, there's an equal opportunity for boys and girls to pursue these sorts of hobby or career or whatever" (P5). One informant illustrated this idea by sharing his own observation that even when the girls and the boys are grouped themselves by gender, he did not see any difference in their performance eventually:

> "Overall, when we did these coding projects as a mandatory exercise, I wouldn't say there is any difference between the performance or interest between girls and boys. We did some of these exercises in small groups and they got to form the groups themselves, so there were boy groups and girl groups, that kind of naturally happens. But looking at the groups' work level there, I wouldn't say there's any difference. Like, some groups were not interested and some were really excited, but the gender didn't play too much of a role in my perspective anyway" (P3).

However, when it comes to the number of boys and girls voluntarily participating in the coding classes, there is a remarkable difference. If it is either a mandatory course (e.g., school subject) or the outside school courses that the parents sign up for their children, there is no difference between the proportion of boys and girls joining in. But as for the elective courses, the situation totally changes. An informant reflected her own experience:

> "There is a really interesting phenomenon to me, because on the lower class, the age range from 1st graders to 3rd graders, all of the time the parents sign the kids to those classes. So in those classes there actually isn't much difference between how many boys and how many girls there are. It's pretty equal to their entrance. But when we move to the 4^{th} to 6^{th} classes, there's a big difference. There are usually 12 kids there, and I would say about 9 or 10 of them are boys usually, and only a couple of girls there." (P2)

Another interviewee also asserted firmly based on his own observation that: "I think with the younger children, the division is not that big. It is more like 50/50 with younger children, which I think is very positive"; but for the older students, "it is more weighted on the boys side. I don't have a number at hand, but if I have to guess a number, I think it would be something like 80-20 percentages of boys and girls" (P5). Another interviewee totally agreed with that idea: "If they had to pick the subject they want to study, like art, or music, or computer things, and stuff like that, boys tend to pick computer science lessons more than girls" (P3).

Each participant gave their own explanation for this difference. Four participants thought the reason lies in the inherent interest of boys towards technology, as one interviewee put it:

> "Boys are commonly quite keen on technique and constructing Legos and such things, of course some girls are too, but I think that it's more typical to males that they are somehow inherently interested in technology." (P6)

Also according to that participant, girls can be attracted to technology through handcrafts more than coding itself:

> "I would say that art and handcrafts, for example, programmable sewing machines, could raise the interest within girls maybe more. So, I think that the arts are… at least when I was young, girls seemed to be into arts more than boys. So, for example, animation or programming SVT, this kind of scalable vector graphic things using the laser cutter to make a rotary or something like that, I guess they would be a very good means to get girls more interested in doing these things." (P6)

That opinion aligned with the idea of another participant that not all topics related to technology could engage girl students:

> "Digital fabrication is so broad. So the workshop, for example, for the high schools, it's an IoT workshop and it might be more technology-oriented… then the design-oriented… I'm not sure, because I haven't really asked them, but it's happened like sometimes girls are interested in, sometimes boys are more interested in those activities. So maybe it depends on the topic. And how you present it." (P4)

In addition, two participants were afraid that the way adults talk about computer and technology, as well as the adults' idea of the gender differences, could affect the perception of youngsters. One informant stated:

> "It's the way that your family is related to these things, or how your parents see these things, or what are the signals that when you are in school, you get that 'these are the boy things, these are the girl things and this is the line between them'. So I think it's more about the ways people think and the ways people are making the children think than the actual problem between boys and girls." (P5)

One participant thought it probably was their marketing approach that caused the problem:

> "I don't know if it has something to do with the way our market these things… like maybe the words we use to market apply more to boys, or something… I don't know what it is, but for some reasons more boys signing up than girls." (P2)

Interestingly, one participant thought the problem lies in the difference in self-efficacy between boys and girls. According to that participant, boys are likely to be more risk-taking and more self-confident in what they are doing, and this attitude leads to their different approaches to coding:

> "Sometimes girls don't believe in their capability. And they seem to be afraid of making mistake, and if they are not absolutely sure about how to do something, they usually ask about it before they do anything themselves. Then, the boys just… they are very confident and certain that they can do whatever they wish, and they start trying to do a project and throw together something. And only after they have created a complete 'disaster' then they come asking for help and find why it doesn't work." (P1)

Students' perception of computer programming

Two interviewees indicated that, from many students' point of view, coding pertains to computer games. An informant shared:

> "Some kids really like to talk about how they play computer games, and when they start doing their own project in Scratch, I hear very often that they say things like 'I'm going to make my own version of Battle Field' or something like that, like they try to duplicate the games they play on computer. So, for those students, it is an important part of life and experience." (P1)

In the school setting, many students find computer programming not easy to acquire. "In regular classes, I would say a lot of them just think that it is hard and they are not that enthusiastic about that sort of thing", P2 shared. Conversely, for the students who have much experience with coding, computer programming in school tends to be perceived as their leisure activity instead of a compulsory learning task. Those students are likely playing with computer rather than completing an exercise. An interviewee expressed:

> "Some might be into it [coding] because it is fun and because they have previously good experiences with this kind of stuff. So, it is like playing around, like they're not thinking about it too much with regards to a school subject, so to speak, but like activities without learning goals. Of course, there are learning goals, but maybe students are not thinking about it too much" (P4).

In addition, students are well aware of the application and the relevance of computer programming in their life. Compared with more abstract subjects such as mathematics, students seem to have fewer wonders about the relevance of coding. "When I teach mathematics, sometimes they ask me how it is applied in real life and what do we do with this. But I don't get these questions in the coding classes", said P4.

However, the participant teachers almost do not have adequate understanding of students' perception towards coding. Three out of six participants openly confessed that they do not know much about students' opinions on computer programming. They gain insights into this topic mainly though their own observation and assumption.

4.6 Hands-on learning activities

4.6.1 Collaborative and individual learning activities

The participants addressed both the significance of collaboration and the necessity of individual work. Two participants stated that many students prefer working on their own to working in groups. Another participant asserted that students' preference for either individual work or group work depends on the aim of each activity. The important point is that teachers should envision which learning objective is suitable for either collaborative or individual learning. Individual tasks seem to be the most favorable for equipping students with basic knowledge and skills, whereas collaboration is more fruitful when students need to exchange ideas or deal with complicated learning tasks. As P4 put it:

> "I usually want them to do the basic exercises on their own, so everyone gets the basic grasp of things. But then when we do something that requires a kind of innovation or planning and such, I think it's important that students have the skills to work in a group, and they can talk about the concepts, and throw the ideas around."

The above opinion somehow harmonized with the statement of another participant that the learning objectives play an important role in defining which learning approach is more suitable:

> "In the coding classes where the coding without robots, I would say that they do work mainly by themselves, but sometimes, you know, they get friends there and they want to do the projects together. And that's, of course, encouraged. And that's OK if they want to do that. But then on the robotics clubs, I think it's probably about 50-50, like doing by themselves and doing things as a group" (P2).

Interestingly, two informants teaching in coding clubs shared that student collaboration sometimes happens naturally. P1 shared that:

> "That [collaboration] usually happens organically. In the class, there are kids yelling at each other like 'Hey, come check out my game' and 'come test my game', and then they go and look at each other's work, test it, and play with each other. So I do not really have to do much extra to encourage their collaboration, because that's something that I find organically happens."

This resonates with experience of P2. That interviewee stated that it is when solving complicated hands-on learning tasks, e.g., coding a game, that students' collaboration occurs the most naturally. They help each other in tackling the difficult situation without the requirement or instruction of the teacher. In those circumstances, the responsibility of the teacher is just to encourage students' teamwork.

In other situations, teachers need to intentionally organize learning activities in order for students to have more opportunities to collaborate. The participants listed out some activities that they had organized for students' collaboration. Three out of six interviewees mentioned competition, for example: "We might have sorts of competition that we divide the class into small groups and they have a huge amount of parts to build their own robots to achieve the task, and they have to work in groups" (P1), or "We give them some sort of task, like make or build a robot that can do this and this and this. And then we will test them all together, either through like a friendly competition or like just playing against each other or playing together or things like that" (P2). Another participant shared an interesting teaching idea that he once implemented: "We ended up building the devices and putting them inside a glass cabinet in the school corridor so everyone could see. And there was kind of a part of this was to make a presentation that would be on a computer screen. There is a paper like telling people how it was done and how they came up with the product" (P3). The other interviewee mentioned Kooditorio, a coding counselling session, in which students come doing the coding tasks independently with teachers available for help, or they could do the tasks in group and help each other out, even though the undergraduates likely prefer working by themselves:

> "At the university level, we have Kooditorio. That's our hands-on laboratory. So they [students] come and just sit there and start programming, and we go around. There is a kind of… like… reservation, so you come and sign your name on the blackboard, and we go like one by one, and we sit there and try to solve the problem until it's solved. We sit there and look at this thing and do it together till it works. And also they can, like, discuss with each other. But it's not very common. It's it's a pity that they don't too much help each other. If you have some friends that are good at coding, you can get help. But if you happen not to have so then you are basically quite alone." (P6)

Along with physical spaces for group work, they also have a Microsoft Teams channel where its members can be available at any time for supporting each other. That participant also suggested that, at high school level, it would be beneficial if this teaching model is replicated.

One participant shared that she was trying to form the student group in a way that each group had students at different levels and strengths. At this point, it also reflected the student diversity in the class.

Two participants shared that there is still a lack of coding community for students at a large scale. According to them, students' programming communities in Finland are now mainly held spontaneously though they are quite active. One participant discussed:

> "It [community] is something that can be facilitated by youth work. But I think not… at least not in the larger scale, maybe something very local, maybe like a youth house may have some small Discord servers, or something related to programming, or something like a very small community held up with youngsters that the youth workers are familiar with in somewhere else—not from the Internet or anything like that but from the actual youth work, from the youth house or from the area that they are working in. So, sadly not, nothing in the larger scale. Nothing that I know in Finland right now" (P5).

This comment was in line with opinion of another participant:

> "We have LUMATE. This organization is giving some extra material and kind of fun, extracurricular activities. LUMATE is good. And then we have some hackers… but that's not national. Yeah. Not formal. But this just… how to say… some firms and some active persons are organizing something" (P6).

4.6.2 Playfulness

All participants agreed that playfulness is crucially important and that they pay much attention in bringing this factor into their coding class. Interestingly, when being asked about the manifestation of playfulness in the coding class, all the participants' answers implied that playfulness is more than just playing games.

Playfulness could present in each and every activity in the class. Sometimes it seems to be a motto of the class instead of activities. One participant shared that:

> "In our class, the whole point is all the kids are voluntary, at least we hope that they are voluntary, and not because their parents force them to. And this is why everything should be, first of all, fun." (P1)

P2 confessed that sometimes she finds herself spend too much time on planning a teaching session to ensure it would be a playful learning session for students:

> "Absolutely in every single class that I teach, it has to be enjoyable, whether it is us playing like games or making games or working with new equipment, it is always playful."

Playfulness associates with students' intrinsic motivation and self-directed learning. According to P1 and P2, as students' participation in their coding club is voluntary, playfulness is the most crucial

element in their activity. Three out of six informants shared the idea that playfulness manifests in students' self-directed learning progress. From their viewpoint, playfulness is when students have the feeling of joy and proud of finding out their own ways to solve a learning task. The answers of two participants regarding that idea are excerpted as follows:

> "If someone has an idea like a coding project and it is different from what we are trying to achieve or what we have previously trying to do, it's still alright. It might be difficult for the teachers to instruct, but… well… it's more important that the kids can do something they find meaningful and fun. I mean, if they follow my instruction, well, of course, they can end up with something, yes it works. But if they go a bit on their own way, they will have something that they are much more proud of." (P1)

> "I think a few years ago, the fablab offered kind of free activities, like total freedom, like they have certain things that they need to do, they need to use the microcontroller, they need to use some kinds of device to make a sound or light or a movable part, but they could come up with something that they want, and something that they're interested in, not like studying or schools, but more like hobby, playing with the devices and so on. So I think that this is an example of the activities that focus on playfulness." (P4)

In addition, playful learning activities can make coding more appealing and easier for students to approach. P3 shared that:

> "At the beginning of the project, nobody knew how to program the devices, what does it need to pump water or measure soil humidity, or how would it work. And the groups just started building prototypes and experiment, like how does this pump work and "see what happens" type of thing. And it was really playful. And I think that's something really important because otherwise, if it's dull, you don't want to do it… and the amount of time it takes to figure out like the parameters for that stuff… and how difficult the problem is… everything is too much if it is dull."

The above idea is consistent with the following perception of another participant, which emphasized that playfulness is non-direct and easier way to bring coding closer to students:

> "You can have non-direct ways to do things. I think Scratch is a great example how to learn coding with animation or very simple games. And the way Scratch uses the blocks that you use to make your own code, it's quite brilliant. And it makes one learn the logic behind the coding, not necessarily the code itself, but the logic how things work. And I think it's really great. And I also know that some people have used like drawings… make the youth draw about the code, or the things that they want to code, or what the code should do. So, there are lots of these sorts of thing that are used in the area of playfulness if you will. (P5)

Two participants suggested that regardless of what kind of activities, as long as they engage students and bring out motivation, they can be considered playful. For instance, one suggested that

sometimes students can learn computer science while doing projects of other school subjects: “I think that it could be integrated in many subjects, for example, biology or history, to make it like project-based learning” (P6), and when doing so, students could use computer programming to report their project results. Teachers can also leverage multimedia to make coding more engaging to students. Another interviewee mentioned “playful competition between groups of students” (P3) as a way to generate more fun and motivation in the class.

Finally, when it comes to the games and websites that facilitate learning to code in a playful way, the names mentioned by participants are Ten Monkeys, DragonBox Algebra, Makey Makey, code.org, and VILLE learning management system.

4.6.3 Tinkering and creativity

Tinkering is another pivotal element of hacker culture, as P1 put it:

> “One point in hacker culture is taking things apart and see how things work, and tinkering is a really big part of finding out how stuff actually work. That is the way they [students] actually learn stuff.”

Likewise, P2 asserted that “To me, tinkering is one of the most important things or one of the most useful things that kids can learn from coding”.

Tinkering is also associated with creativity and problem-solving skills. Through the trial-and-error process, students can simultaneously enhance their creativity. P4 explained the relationship between tinkering and creativity:

> “When you do fabrication activities, it’s not like straightforward. It's not like a straight line, but it's more like you do something then realize, ‘oh, maybe this kind of idea can be changed, we can do this, and we can do that, and we can add more sensors, and we can make this kind of sensor better. It’s like explorative. So, students can be more creative. And they are not following the instructions, but more doing by themselves and then trying something and testing.”

Two respondents delivered concrete examples to illustrate how tinkering can be taught in the coding class. Interestingly, both those exemplars point to fixing pre-made code as a way to learn tinkering. An interviewee told that it also was the way he learned coding by himself before entering university:

> “We had pre-made code. That's pretty like the Arduino code. It's not intuitive. It's kind of difficult because if you start from scratch, when you are missing a semicolon then the program will give out code errors and stuff like that. But when you approach it with a pre-made code that is pretty short and pretty simple, and then you have these

> kind of values that they need to find and figure out how can I make our program make sense, like how can I fix it. Like this is my anecdotal experience. That's how I learned to code in the first place" (P3).

Another participant was also planning to exploit pre-made code to teach students tinkering:

> "But we also have tried to put it in our club plans as well, like, for example, the latest club that I just planned was that I make codes for our game, that are like ready games, like you can play them, but they're like very unfinished in a way that there is a lot of improvement that needs to be made. And the reason I wanted to make a club like that is so that the kids would get comfortable, like changing other people's codes and making those changes, and then hopefully then become more comfortable with making changes to their own things… like, you know, changing the mindset a little bit." (P2)

Prepare students with the mindset in regard to tinkering is not less important than actually teach them about tinkering. Youngsters seem lack the patience. Many of them "have the mindset that it needs to be perfect right away, no changes needed to be made, and they get frustrated really easily if they cannot figure it out right away", P2 pointed the problem out. This impatience is a hindrance for their tinkering process, as well as a challenged task for teachers to deal with, as P2 put it:

> "And we always kind of try to make sure that the kids are learning patience. Because tinkering takes a lot of patience and kind of 'errors and learning from errors' mindset. We always try to encourage that. But that is actually really hard to do with some kids. Some kids are super sensitive to that sort of thing, and they just always just want to get the tasks done and move on and do not want to improve things. And that is a very common mindset I find in children, especially like the elementary school age children. So we try to with those sort of kids, we try to push it like a little bit."

Another teacher talked about another mindset that affects students' attitude towards tinkering: "I think for many people it seems that some are afraid of touching anything that they are not really familiar about" (P1). According to that informant, when students are afraid of making mistakes, they would hesitate to try different approaches, especially new ones, if a solution does not work. By changing this mindset of students, that teacher wished to encourage tinkering more in the class.

4.7 Teachers as facilitators

Teacher autonomy is one of the treasures of being a teacher in Finland. Teachers hold complete control over activities in the class as long as they meet the requirements and standard set in the national core curriculum. Therefore, teachers' skills and knowledge have a huge impact on the performance of their students. As an informant shared:

> "I have to say that it's very much up to the teacher. So the level of the teaching in… because it is… so how much… it reflects so much that how this teacher, what is his or her attitude and how much he or she has educated himself" (P6)

Considering the vagueness in FNCC14 as to computer programming, teachers' competencies have influence on students' code literacy more than ever. The difference in students' coding skills is the result of the variation of teaching. An informant indicated:

> "And that's when we started to see more differences in the age group, depending on who is the previous teacher or from which school the students came. And yeah, this kind of escalated the situation, because previous to that there was also huge differences. But now I think there are even more, like, skill level variation among the pupils." (P3)

Besides, a new teaching content integrated into math subject which is "already very full" (P2), blending with a fuzzy guidance, results in teachers' overwhelming. It can be said that Finnish math teachers react differently to this remarkable change in the national curriculum. "Some teachers got really, really into it and did all kind of stuff with the Lego robots and things like that. But some teachers did not", said P3. For the veteran teachers, especially those who are responsible for inclusive education, these new changes place them into a challenging situation:

> "The teachers, they do not know much about computer science and they are so busy. So this new curriculum that came into effect has caused quite a lot of problems in schools, and I would say that also in inclusive education, you know, we have students in special needs. So it is a very challenging situation for math teachers. They are not really keen on learning new things such as computer science." (P6)

Nonetheless, P6 expressed her hope in the new generation of teachers who are enthusiastic and eager to bring their teaching initiatives to the coding class: "But it's a totally different attitude of the younger generation who are about to graduate and willing to apply what they have learned", P6 said.

Eventually, the confusion and objection of some teachers in regards to teaching coding in the math class is understandable. The interviewees pointed out many challenges that teachers commonly encounter when teaching coding. Firstly, there is a lack of teaching material. An interviewee shared that he even once had to utilize the exercises from a university's introductory course as an extra learning task for a gifted student in the class. Another respondent explained this challenge of lacking resources in more detail:

> "And then you also need to find the time to make the instruction sheets which normally… for example, in mathematics, you have the mathematics book that someone else did, and that is good. And with programming maybe… Well, there is some pre-made materials, some of them are good, and some of them fit the lesson

> that I have planned. But many times, there is a lack… there is not a whole book… Of course, there is that kind of material, but I haven't found the good material for it. And I don't know if there will be good material." (P3)

The pressure of teaching time as stated above blending with the diversity in students' levels is another challenge for teachers. Another interviewee shared about this issue:

> "I think the most difficult when planning the club is when I think about how to take into consideration all the different types of children, because I know that some children want to do harder things, and for a lot of the other ones, I don't really want to push them and I try to think what way of presenting a problem or helping the students out with a certain task. What is the best way considering all the different types of learners?" (P2)

Besides aiding general students, the educators were well aware of the importance of nurturing the gifted ones. Basically, Finland education emphasizes equity much more strongly than praising outstanding performance. It does not mean that Finnish education does not support gifted students. They rather support in a way that students has equal opportunities to develop at their own pace, as P6 put it:

> "We don't stress too much on good performances and this kind of… It's very good for them but they are not highlighted because we want to …how to say… we don't want to praise excellent students too much because we don't want those who are not that excellent feel that 'I'm nothing'. It's important that everyone develops optimally at their own level. So that's it. We don't want to say that 'you are nothing, look at this guy, he's everything'."

Therefore, teaching strategies for fostering excellent students mostly depends on each teacher. Three out of four informants emphasized the importance of encouraging the exploratory attitude and the self-directed learning of those students. For example, one interviewee stated that:

> "I think it's important to encourage those attitude that is…. Like, very… how to say… again, it's somehow related with the openness and interested-driven, like, for example, when a student starts doing something and teachers, or someone like adult, or some like colleagues, or friends can encourage that student's attitude to engage in something that they like. So I think this is one thing. Because I think kids are sometimes afraid when they like something too much and the other friends don't have interest in that. And they leave those… for example, if I am interested in programming but no one else is interested in, then I would leave my interest and play with friends." (P4)

Another interviewee concerned about the balance between providing specialized students with advanced knowledge and at the same time keeping their curiosity for the next lessons:

> "However, I think that's not necessarily a problem because those who are really high level are sophisticated. It's easy to kind of pique their interest a bit by having an open ended like "the last exercise" which is something difficult or something they need to figure out something that's above the learning goals for that lesson. So I try to and then this is… Furthermore, it's a bit difficult because of course if I want to continue on the next lesson, I don't want them to rush towards those topics, but I need to find some topics that are otherwise wouldn't be covered. Does this make sense to you? So we don't rush through the material, but they get the experience or learn something extra" (P5).

One participant mentioned the international competition, such as Majava (Bebras) competition, as a way to lift up gifted students and give them a chance to exchange knowledge with peers who share mutual interest. However, the participation of students in those competitions strongly depends on the network of teachers, "if a teacher is very good and has a kind of international network, and he or she knows about the competitions that he or she can send her students to" (P6).

5 DISCUSSION

5.1 Code literacy, digital literacy and computational thinking

The first sub-question of this study aims at clarifying the interplay of code literacy and digital literacy. The literature presented in the second chapter showed that students' digital literacy can be supported by code literacy. Computational thinking and digital fabrication can be deemed as a bridge connecting code literacy and digital literacy. It resonates with the findings of this research. Although the participants did not directly connect the notion of code literacy with digital literacy, they offered plenty of ideas and comments on computational thinking and digital fabrication.

All educational experts participating in this study agreed that computational thinking is the critical skills for youngsters in the digital era. Computational thinking is not just about thinking as a computer, but it is more of attitude and metacognition. That opinion is consistent with the literature discussed in the second chapter of this study (e.g., Furbur, 2012).

In addition, the findings of the present study raised a discussion on the concept of *creative computational thinking*. By understanding the basic principle of how computers work, students can be creative with it. This discussion aligns with the research of DeSchryver and Yadav (2015) with regard to the connection between computational thinking and creative thinking.

According to the informants, students can develop their computational thinking by learning to code, and vice versa. This finding accords with several previous studies. Depryck (2016) pointed out that there exists a complex link between coding and computational thinking, and this intertwined relationship is the real critical success factor for the reach of ICT, including coding skills.

Furthermore, both code literacy and computational thinking can be enhanced through digital fabrication, as many digital fabrication kits require students to use programming to produce the final product. The same finding can be found in Iwata et al. (2019). According to Iwata et al., complex problems formulated in digital fabrication activities can be solved by using computers for assistance. Through those activities, students learn to think logically and implement possible solutions optimally. The nature of fabrication activities which inquires the use of computers and complex problem-solving can encourage the development of computational thinking.

5.2 Finnish educational experts and teachers' perceptions of hacker culture

The fact that the notion of hacking can be understood as either a criminal or exploratory act made some participants find it hard to give a full definition for this concept. The complexity of this notion was also highlighted by Joran (2008) and Radziwill et al. (2015). However, the participants tried to mention the characteristics of hacker culture in different ways and form a connection between hacking and education. Their perceptions confirm the characteristics of hacker culture examined in existing literature.

According to the participants, hacker culture points to the attitude, beliefs and ethics when working with information technology. Hackers are depicted as self-directed learners who are eager to solve hands-on problems. This perception accords with the statement of Suiter (2013) that "hackers are autodidacts". Suiter also clearly indicated that doing or making is the self-learning approach that hackers adopt to obtain systematic knowledge structure.

Hackers formed hacker culture by sharing and collaborating towards technology. This knowledge was examined in authoritative literature such as Levy (1984; 2010) and Thomas (2002). Likewise, the participants in this study addressed open sharing as one of the key characteristics of hacker culture.

Regarding the reason why hacker culture seems not to be prevalent as maker culture even though they share some characteristics, the participants tend to comment that it is because maker culture evokes more positive impressions in the public, whereas hacker culture raises the idea of digital crime. Besides, hacker culture seems to attract a certain type of technology enthusiast while maker culture aims at alluring anyone fascinated with hands-on technical activities. These findings concur with Richterich and Wenz (2017) as discussed in the literature review.

In addition, the participants in this study pointed out two main downsides of hacker culture. The first one is that hacker culture appears to be dominated by males. This opinion can be found in many authoritative literature. For instance, Thomas (2002) indicated that hacker culture is typically seen as "boy culture", and "the typical hacker is a white, suburban, middle-class boy, most likely in high school. He is also very likely self-motivated, technologically proficient, and easily bored". Thomas also pointed out that there is a limited number of girls immersing themselves into this culture, but those ones take on the values set out by their male counterparts. This difference between boy and girl students will be discussed more in the later section of this chapter.

5.3 The characteristics of hacker culture presenting in code literacy education in Finland

All the findings in this section vividly illustrate the characteristics of hacker culture in education as discussed in the literature review (e.g., Aguado and Canovas, 2019) and as posed in the framework for analysis (see, Table 1).

Firstly, the access to tools and devices is the first and foremost condition for the formation of the hacker culture. In Finland, students have fairly easy access to the equipment needed for learning to code. Both in schools and in non-formal education institutions, students are equipped with various programming kits and languages. Different programming languages—from block-based to text-based—are introduced to students based on their interest and ability. Especially, visual learning environment, such as Scratch, was stated as the most beneficial for youngsters to get familiar with coding. It is in line with previous studies (e.g., Armoni, M., Meerbaum-Salant, O, & Ben-Ari, M., 2015, Kordaki, 2012). However, there exists some hindrance for students' access to technical equipment by themselves, such as teacher permission, the borrowing system of the educational institution, or at a distance, the economic capacity of the municipalities.

In a hacker community, the diversity of its members increases their versatile and extensive collaboration. In the coding class, the student diversity is presented in students' distinction in skill levels, interest, and perception of computer programming. That said, in the coding class, there should be different teaching approaches to different types of students based on their inclination. From teachers' point of view, students' ability in coding is not defined by their genders. Boy and girl students can achieve equivalent learning results in the coding class. However, most of the participants admitted the fact that there is a divergence in terms of boys and girls' interest in coding. The participants pointed out the remarkable gap in the number of boy and girl students voluntarily joining in coding activities. They tried to come up with an explanation for this phenomenon, such as adults' perceptions imposing on youngsters, the unconscious use of words in educational marketing, the inherent interest of boys towards mechanical objects, and boys' risk-taking tendency. One participant addressed that the capability of boys to embrace uncertainty helps them go further with trial-and-error activities when learning computer science. This finding is not only consistent with previous work on the male-dominated nature of hacker culture (e.g., Thomas, 2002) but also harmonized with research on the gender imbalance in the field of ICT education in general. For example, Francis and Skelton (2005) agreed that boys tend to be more adventurous than girls, while girls are likely more obedient, caring, and prefer making safe choices to taking risks. Wong and Kemp (2018) added a comment to this

discussion that gender stereotyped discourses can be imposed on students by parents, teachers and the media.

When it comes to the difference in students' coding skill levels, all participants shared that in the class, there are always talented students who even can learn to code by themselves and the lower-level students who find coding hard to digest. The ability of students in solving the coding tasks by themselves is one of the traits for teachers to spot the gifted students in the class. If those students are in the same class and without the differentiated teaching from the teacher, both kinds of students can easily get bored and lose their patience. The participants believe in the capacity of such students in exploring the technology by themselves. Therefore, they call for the encouragement of teachers and parents to those students, along with providing them with the favorable environment for self-learning.

In respect of students' perception of coding, for some students, coding pertains to computer games, for example, learning to code is to replicate their favorite computer games. For students who are interested in coding, they perceive coding as playing with the computer. Some participants revealed that almost all students do acknowledge the relevance of computer programming to their life and do not inquire about the applicability of it. However, most of the participants admitted that they do not have a full understanding of the students' perceptions of coding.

With regard to collaborative and individual learning, all participants in this study did not emphasize on either of those pedagogies solely. It is consistent with the characteristic of hacker culture that individual work is not less important than collaboration. The educators participating in this study considered individual learning tasks suitable for equipping each student with basic coding knowledge and skills while collaboration is more fruitful when students need to exchange ideas or deal with complicated learning tasks. Students' collaboration sometimes occurs naturally, especially in the coding club, when students have a desire to share their coding products with each other. This emphasis is in line with Panitz (1999) as discussed in the literature review. As Panitz clearly point out, collaborative learning is only meaningful when it helps students gain more insights into the learning topic than when they learn individually. Moreover, one of five principles for effective collaboration, also according to Panitz, is the voluntary participation of students. However, on a larger scale, the participants pointed out that there is still a lack of coding community for students' greater collaboration. Such communities now are mainly held spontaneously, for example, through Discord online platform.

All participants in this study agreed that playfulness plays an indispensable role in facilitating students in learning to code. Playfulness is the focal point that all literature regarding hacker culture discussed in this study paid attention to. It can be seen from most of the interviewees' answers that the concept of playfulness points to exploratory attitude rather than just playing games. This finding is supported by Sicart (2014) as discussed in the literature review. Playfulness embodies in students' self-

directed learning process. Playful learning activities effectively aid students at the beginner level as they make coding appear to be easier to obtain. All in all, playfulness can be manifested in each activity in the class, as long as they evoke from students the joy of exploratory and engagement.

Tinkering is one of the most useful habits of mind that students can obtain through coding. Tinkering has a close connection with problem-solving and creativity. According to the interviewees, as other characteristics of hacker culture, tinkering is also more of a matter of mentality and mindset. Similarly, Martinez and Stager (2013) indicated that "tinkering is a mindset—a playful way to approach and solve the problem through direct experience, experimentation, and discovery". Nevertheless, the participants also pointed out that youngsters seem to lack patience which is important for the tinkering process. Based on the teachers' experience, they suggested that having students to fix pre-made code can be an effective teaching solution to reinforce students' tinkering skills.

5.4 The implementation of code literacy education in Finland

The tradition of Finland's technology education laid a strong foundation for code literacy education in the current times, especially for advocating maker culture and hacker culture in teaching practice. Besides, the fact that many Finnish teachers have experience and interest in handcrafts helps the hacker and maker culture in Finland vigorously flourished. The persistent development of code literacy education in Finland was also highlighted in the research of Tuomi et al. (2018) and Saarikoski (2010) as discussed in literature review.

In Finland, code literacy education is being implemented in different education sectors. In the FNCC14, computer programming is blended into math and crafts subject. Algorithmic thinking is also integrated into math subject to develop students' computational thinking, and therefore as a pedestal for code literacy.

Besides, the findings of this study initially revealed some issues in its implementation. Some participants discussed the vagueness of the instruction regarding computer programming in the FNCC14. Furthermore, it would be more beneficial to have more teaching materials and references for teachers. With those attempts, not only the teaching practice of all teachers will be more consistent, but code literacy will also be paid more attention and firmly hold its important position in basic education as it is supposed to be.

The implementation of code literacy in Finland heavily depends on the perception of each teacher towards computer programming, as well as the economic capacity of each municipality. These findings confirm many previous studies on teachers' autonomy and the matter of trust in Finnish education. For example, Pollari et al (2018) pointed out that autonomy, along with high qualification and motivation,

are the strongest factors to explain the prominent success of Finnish education for the past two decades. Teachers play an important role in encouraging and engaging students in learning to code by organizing playful yet differentiated and goal-oriented learning activities. Given the vagueness of the instruction in the FNCC14, teachers' interpretation of computer programming education influences the way they integrate coding into their math and crafts class. In the non-formal education sector, teachers and instructors possess even more degree of autonomy in teaching practice. This finding concurs with the research of Tolppanen et al. (2015), Tisza et al. (2020), and Patrick and Mantzicopoulos (2015) on the flexibility of non-formal education and its potential in bringing more elements of fun to science learning. In the youth work sector, the finding that youth workers play an important role in connecting students and technology experts confirms the argument of Lundqvist and Kiviniemi (2017). Lundqvist and Kiviniemi stated that in the dramatically changing technology context, youth workers are "enablers" who arrange the favorable conditions for youngsters to pursue their interest with technology. In addition, another interesting finding that emerged from data analysis is that some teachers are hackers themselves. They optimize the use of cheap and multi-function tools and devices. This finding is supported by Wizel (2017) in which the concept of "teachers as hackers" is investigated. According to Wizel, teachers who hack are adaptive and have the ability to maximize the use of resources.

All in all, each education sector in Finland is responsible for promoting code literacy at different degrees: from enforcing coding as a compulsory subject at schools to supporting coding hobbyists at outside school clubs. It provides Finnish students with greater exposure to explore coding according to each student' preference and skill level.

6 EVALUATION

According to Cypress (2017), the rigor and validity of research should be ensured in all phases of the process rather than only be appraised when the work is already done. This study, from reviewing literature to reporting the collected data, was a constant process of researching and reflecting upon the credibility of itself. In this chapter, each phase of the present study is evaluated in terms of trustworthiness and ethical awareness.

6.1 Trustworthiness of the research

Problem statement, purpose and research question

The main topic of this study was chosen considering that the concept of code literacy has been evolving in the field of digital literacy education. The concept of *hacking* has also been paid attention for its implications in education. However, hacker culture is typically considered as an informal phenomenon, hence it is rarely researched through the lenses of formal education. As a result, there is a shortage of scientific literature with respect to the conceptual integration of formal education and hacker culture. However, this deficiency remarks the contribution of the present study.

Review of the literature

The literature review in this study was done in an integrative manner. Snyder (2019) stated that this type of review calls for the flexibility and creativity in collecting literature, because the aim of it is "not to cover all articles ever published on the topic but rather to combine perspectives and insights from different fields or research traditions". The conceptual framework of this study is predicated on the related works in varied research fields and subfields, e.g., educational history, pedagogy, social science, media and communication, information and technology. Peer-reviewed scientific articles and book chapters are especially preferred. The papers chosen for review were found on credible online databases, e.g, SpringerLink, ERIC, JSTOR, to name a few. The service of local libraries was also harnessed with the hope to cover the robustness of previous studies related to computer education in Finland. Therefore, although the literature review of the present study was not conducted in the

systematic manner, it still ensures to capture the main discussions around the topic. These efforts demonstrate that the literature review of this thesis consists of authorized books and scientific papers, and it was also conducted in a proper way.

Data collection

Validity in qualitative research, according to Leung (2015), points to the suitability of tools, processes and data. The justification for choosing qualitative approach, interpretivist paradigm and thematic interview method was presented clearly in chapter 3 of this thesis.

The robustness of the current study would be enhanced if multiple forms of data were collected instead of single form, i.e., interview, as it was done. At the first place, this study was planned to be an ethnographic-based research; classroom observation was supposed to be the main research method and the interviews would provide additional data sets. However, the restriction of entering educational institutions due to the COVID-19 pandemic forced this study to change its orientation. Nevertheless, all attempts were made to guarantee that interview data still bring robustness and authenticity to the study. Authenticity refers to the diversity of research participants and accordingly the range of different views that the research depicts. Authenticity is shown in the selection of suitable participants and provision of a detailed description (Schou et al., 2011, as cited in Connelly, 2016). This study purposeful sampling, which means that the participants were intentionally chosen based on a set of criteria so that they are the most suitable candidates for the research. Pilot interviews were also carried out with experts in the field of technology and education, which could strengthen the validity of the interview protocol and guarantee the choice of most suitable actual participants. The major drawback of this data collection process perhaps is the small size of sample. With the aim of reaching experts from different educational sectors, this study succeeds in reaching informants from a broad spectrum, yet it still lacks the depth. Each expert in this study has expertise on one aspect, hence some ideas are not strongly harmonized. For example, there was only one participant from the youth work sector, and only one person who expertises on fablab operation. However, the fact that the interviewees come from different backgrounds and four of them are experts increases the credibility and robustness of the study. The detailed description of participants was delivered in chapter 3 of this thesis. None of the interviewees has a close relationship with the interviewer. Only one among them had met the interviewer once before, but it did not result in any prior assumption or relationship. The balance in participants' genders also prevents this research from gender bias.

Data analysis and findings

The collected data were analyzed with a consistent process. The data analysis procedure was reported genuinely in the third chapter of this thesis. The process of coding data and identifying themes was carried out not only once. That process was repeated continuously until the data were reasonably linked to the theoretical framework and the research questions of the study were answered. The subjective interpretation in data analysis may pose threat to the validity of the research. However, the subjectivity of the researcher is considered the nature of the research conducted under the qualitative approach and interpretivist paradigm.

Writing

To prevent participants' ideas from being misunderstood, all their words were put in the context when they were spoken. Besides, cohesion and coherence are the goals for the writing process. However, the thesis writer is also an English learner hence language mistakes probably cannot be avoided. This thesis strictly follows the APA writing style. The newest APA guideline (7th edition, 2020) was checked carefully to guarantee that the updated writing manner was used in this study. Plagiarism, the act of using someone else's words or ideas without giving any credit to the original sources (Pun, 2021), was inhibited intentionally. Turnitin, an online plagiarism detection service, was used to check the origin of this study before it is published.

6.2 Ethical consideration

The purposes of research ethics are to respect the autonomy of research subjects, avoid harm as well as safeguard the privacy and data (Finnish National Advisory Board on Research Ethics, 2009).

This study seriously took the anonymity of all participants into account. All participants were informed clearly about the aim of the research, the objectives of the interview, and their roles in this study (either as an expert or a practitioner). By signing the consents which were sent to them before the interview, the informants agreed that they understood all critical information related to this study and how their privacy was taken into consideration. They also were the ones who chose the video conference platform for the interview so that they could feel safe and comfortable.

After each interview using Microsoft Teams service, the recorded video was automatically stored in the Microsoft Stream platform. To protect the privacy of the interview, that video was immediately downloaded to the personal computer of the researcher and then was permanently deleted from all

Microsoft channels. For the interview using Zoom service, the recorded video and audio were stored in the researcher's personal computer without being automatically located in any online platform.

In the report, all informants are code-named as "P(number)". If their workplace and project are mentioned, they all were written under pseudonyms.

This study mentioned the gender issue at some points. Two genders discussed in this study are male and female (or they were referred as "boy" and "girl"). Other genders were taken out of this research purely due to an objective limitation on resources, i.e., time and data set, and the main scope of this study. In regard to research participants, in the report, the pronouns (he and she) were avoided as much as possible to reduce gender bias.

7 CONCLUSION

7.1 Summary

This study is placed in the context that code literacy education has become more and more prevalent all around the world. Code literacy education aims to provide youngsters with basic computational knowledge to live in the dramatically growing world of algorithm-driven media, thus it has a close connection with digital literacy. The first objective of this study is to conceptualize the relationship between code literacy and digital literacy. In the field of code literacy education, for the past fifteen years, the concept of *hack* has emerged as an educational phenomenon. Educational initiatives such as *hackathon* and *hackerspace* have been widely promoted and researched. However, despite flourishing based on the idea of hacker culture, such pedagogy and learning community have not been scrutinized from the hacker culture perspective. From a broader viewpoint, there has still been a lack of academic research delving more deeply into hacker culture and its characteristics in code literacy education. Not to mention the nuances of the term *hack* have made it to be misunderstood in public discourse. Considering all those problems, the main objective of this study is to understand how the characteristics of hacker culture are integrated into code literacy education.

The research question posed in this study is "*How are the characteristics of hacker culture integrated into code literacy education?"*. To seek answers for that question, the study was designed under the interpretivist paradigm and qualitative research approach. Thematic interview was employed as the research method. Six participants, who are educational experts and practitioners, took part in six interviews to bring their in-depth knowledge in the field of code literacy education into this research. The interviews were then analyzed using thematic analysis method.

Based on this study, it can be concluded that code literacy can strongly support digital literacy through computational thinking. When enhancing computational thinking, students simultaneously grasp the basic knowledge on the mechanics of computers and algorithms, which are closely related to coding skills. This finding is in line with previous research, such as Depryck (2006) and Iwata (2019).

Also, the findings reveal the perception of teachers and educational experts towards hacker culture as an educational concept. From their viewpoint, promoting hacker culture in education is to

advocate self-directed learning, exploratory attitude, open sharing and collaboration. These findings are supported by Suiter (2013), Levy (2010), Thomas (2012).

The findings also show that the characteristics of hacker culture are presented in many facets of code literacy education. Firstly, student diversity, which manifests in the difference in students' levels, interest, and genders, is embraced in the class. Although there is no gap between the capability between boy and girl students, there is a remarkable difference in their interest towards coding activities. Girl students tend to engage more in the coding activities having the integration of graphics, arts, and handiwork; and they also are likely to lack confidence when it comes to coding and technological activities. These findings are strongly supported by Francis and Skelton (2005), Wong and Kemp (2018). In regard to geek students who are either more interested in coding or at the higher coding skills level than their peers, they have been experiencing the shortage of learning community or formal learning activities to nurture their motivation. Secondly, in general, Finnish students are equipped with various programming tools and devices for their technological exploration, despite some hindrances such as teacher permission, borrowing system of the educational institution, or at a distance, the economic capacity of the municipalities. Thirdly, a variety of hands-on activities are organized in the class focusing on playfulness, tinkering, creativity, and the balance between individual work and collaboration. The finding that playfulness is more of attitude than playing itself is in line with the research of Sicart (2014). Likewise, tinkering is also the matter of mindset which addresses the patient yet playful problem-solving process, as presented by Martinez and Stager (2013). Regarding collaborative and individual learning, the focal point lies in the objective of the learning task. The finding of this study is consistent with Panitz (1999) that collaborative learning is most meaningful when it takes place voluntarily as well as it helps students acquire more knowledge and skills than when they learn individually.

Moreover, some new insights into the implementation of code literacy education in Finland has come out of the collected data. Code literacy is being advocated by different education sectors: formal schools, non-formal educational institutions, youth work centers. However, the vagueness in instructions for implementing code literacy in FNCC14 makes the teaching practice depend strongly on each teacher (or educator) who are directly responsible for organizing the coding learning session.

7.2 Implications

The findings of this study conveyed opinions of educators about the vagueness of instruction in the FNCC14 regarding to what extent coding should be integrated in math and craft class. It calls for more endeavours of the national education agencies to specify the standard and measurable outcomes of

computer programming in schools. Besides, the additional materials used in teaching coding, e.g., teacher's handbooks or sample worksheets, should be published widely soon to meet teachers' needs.

From the educational policy and administration perspective, more attempts should be made to pursue the equity in code literacy education, because as stated by some participants in this study, the status of code literacy education of a municipality somehow mirrors its economic and social development. In addition, the educational strategies focusing on students who have high-level competence need to be set out, especially when Finland is constantly holding its position as a reference society for both education and technology. Talented and enthusiastic programmers constitute the backbone of hacker culture, as Raymond (2012) put one of five elements of "The Hacker Attitude": "Attitude is no substitute for competence". For instance, the famous Finnish hacker and intellectual Linus Torvarlds has been gathering more than five thousand developers around the world to join his Linux open-source project[3]. The focus on students who have a strong interest in technology is definitely not contradictory to the pursuit of equity of Finnish education; it eventually helps to better reach equity.

All in all, similar to agile methodology in software development[4], the feedback loop and the cooperation is necessary to fulfil any strategy for improving code literacy education. This requires the involvement of all parties including policy makers, teachers, students and parents.

7.3 Future work

As mentioned in previous sections of this study, the topic of hacker culture in code literacy education is a niche at this moment. The common misunderstanding or partial understanding of hacker culture as criminality has inadvertently left out its implications for education. However, given the flourishment of coding in non-formal and informal educational settings as well as the association of *hacking* with problem solving and creativity, it can be anticipated that the concept of *hacking*, *hacker* and its so-called *ethics* or *spirit* would be more prevalent in the education area in the very near future. In the meantime, it requires more studies from different disciplines, i.e., education, social studies, and technology, to form a more solid conceptual foundation for the integration of hacker culture into education.

First and foremost, there should be more updated research work focusing on hackers themselves to investigate how they acquire coding skills and knowledge in this algorithm-driven era. Only by

[3] https://github.com/torvalds

[4] "Agile methodologies aim to deliver the right product, with incremental and frequent delivery of small chunks of functionality, through small cross-functional self-organizing teams, enabling frequent customer feedback and course correction as needed." (https://www.digite.com/agile/agile-methodology/)

having an in-depth understanding of hackers' mentality can researchers and practitioners have a more precise view towards the culture of hackers' community.

This study could be replicated either in Finland or other research sites with a larger sample size. The number of participants representing each education sector should be increased to gather divergent opinions and thus enhance the validity of the research. In addition, more research should be conducted with schoolteachers as well as from the perspective of students.

Additional data set and the mixed methods approach could be employed to capture a more holistic picture of this topic. The observation method could be employed to collect ethnographic data in hackerspaces, makerspaces or fablabs. Each of the characteristics presented in this study could also be an independent topic for future studies. In parallel with researching hacker culture as a whole, examining each facet of it could be a fruitful approach which will bring out deeper knowledge of the topic.

REFERENCES

Adomavicius, G., Bockstedt, J., Curley, S. P., Zhang, J., & Ransbotham, S. (2019). The hidden side effects of recommendation systems. MIT Sloan Management Review.

Aguado, A. G., Canovas, I. A. (2019). Educación hacker: una expresión emergente de la pedagogía crítica para la sociedad en red (Hacker education: an emerging expression of critical pedagogy for the online society). Teias Magazine v. 20 (Special Edition - 2019): Educação ativista na cibercultura: experiências plurais. http://doi.org/10.12957/teias.2019.43375

Alleyne, B. (2018). Computer Hacking as a Social Problem. In Trevino, A. J (Ed.), *Cambridge Handbook of Social Problems*. (pp. 127-142). Cambridge University Press.

Angarita, M., Nolte A. (2020). What Do We Know About Hackathon Outcomes and How to Support Them? – A Systematic Literature Review. In: Nolte A., Alvarez C., Hishiyama R., Chounta IA., Rodríguez-Triana M., Inoue T. (eds), *Collaboration Technologies and Social Computing*. CollabTech 2020. Lecture Notes in Computer Science, vol 12324. Springer, Cham. https://doi.org/10.1007/978-3-030-58157-2_4

Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016). A K-6 computational thinking curriculum framework: Implications for teacher knowledge. *Educational Technology & Society*, 19(3), 47–57.

Armoni, M., Meerbaum-Salant, O, & Ben-Ari, M. (2015). From Scratch to "Real" Programming. ACM Transactions on Computing Education, 14(4). https://doi.org/10.1145/2677087

Baron, R. (2019). Digital Literacy. In *The International Encyclopedia of Media Literacy* (pp. 1–6). John Wiley & Sons, Inc. https://doi.org/10.1002/9781118978238.ieml0053

Blikstein, P. (2018). Maker Movement in Education: History and Prospects. https://doi.org/ 10.1007/978-3-319-44687-5_33

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. Qualitative Research in Psychology, 3(2), 77–101. https://doi.org/10.1191/1478088706qp063oa

Cavalcanti, G. (2013). Is it a Hackerspace, Makerspace, TechShop, or FabLab? Makezine.Com

CNBC. (2016, March 16). Hot Robot At SXSW Says She Wants To Destroy Humans | The Pulse [Video]. YouTube. https://www.youtube.com/watch?v=W0_DPi0PmF0

Connelly, Lynne M. (2016). Trustworthiness in qualitative research. *MedSurg Nursing*, 25(6), p. 435.

Creswell, J.W. and Plano Clark, V.L. (2011) Designing and Conducting Mixed Methods Research. 2nd Edition, Sage Publications, Los Angeles.

Denning, P. (2009). Beyond Computational Thinking. *Communications of the ACM*, 52(6), 28-30. http://doi.org/10.1145/1516046.1516054

Depryck, K. (2016). From computational thinking to coding and back. In *Proceedings of the Fourth International Conference on Technological Ecosystems for Enhancing Multiculturality (TEEM'16)*. Association for Computing Machinery, New York, NY, USA, 27–29. DOI:https://doi.org/10.1145/3012430.3012492s

DeSchryver, M.D. & Yadav, A. (2015). Creative and Computational Thinking in the Context of New Literacies: Working with Teachers to Scaffold Complex Technology-Mediated Approaches to Teaching and Learning. *Journal of Technology and Teacher Education*, 23(3), 411-431.

Dillenbourg, P. (1999). What do you mean by collaborative learning?. In P. Dillenbourg. *Collaborative learning: Cognitive and Computational Approaches*, Oxford: Elsevier, pp.1-19

Dougherty, D., Conrad, A. (2016). *Free to Make: How the Maker Movement is Changing Our Schools, Our Jobs, and Our Minds*. North Atlantic Books.

Dufva, T., & Dufva, M. (2016). Metaphors of code—Structuring and broadening the discussion on teaching children to code. Thinking Skills and Creativity. https://doi.org/10.1016/j.tsc. 2016.09.004

Dufva, T. (2018). Creative Coding at the arts and crafts school Robotti (Käsityökoulu Robotti). CEUR Workshop Proceedings.

Duin, A. H., & Tham, J. (2018). Cultivating code literacy: A case study of course redesign through advisory board engagement. *Communication Design Quarterly*, 44-58.

European Schoolnet. (2015). Computing our future—Computer programming and coding: Priorities, school curricula and initiatives across Europe. http://www.eun.org/documents/411753/817341/Computing+our+future_final_2015.pdf

Fields, D. A., Giang, M., and Kafai, Y. B. (2013). Understanding collaborative practices in the Scratch online community: Patterns of participation among youth designers. In Rummel, N.,

Kapur, M., Nathan, M., & Puntambekar, S. (Eds.), *To See the World and a Grain of Sand: Learning Across Levels of Space, Time, and Scale*. CSCL 2013 Conference Proceedings, Volume 1

Finnish National Agency for Education. (2014). Finnish National Core Curriculum.

Finnish National Advisory Board on Research Ethics. (2009). Ethical principles of research in the humanities and social and behavioural sciences and proposals for ethical review.

Francis, B. & Skelton, C. (2005). *Reassessing gender and achievement: questioning contemporary key debates*. London: Routledge.

Fransman, J. (2005). Understanding literacy: a concept paper. Paper commissioned for the EFA Global Monitoring Report 2006, Literacy for Life.

Furbur, S. (2012). Shut down or restart? The way forward for computing in UK schools. Technical report, The Royal Society, London.

Galletta, A., & Cross, W. (2013). Mastering the Semi-Structured Interview and Beyond: From Research Design to Analysis and Publication. New York; London: NYU Press.

González-González, C. S., & Arias, L. G. A. (2018). Maker movement in education : maker mindset and makerspaces. Proceedings of the 2018 IV Jornadas de Interacción Humano-Computador (HCI); 23-27 April 2018. Popayán, Colombia: Interacción Humano- Computador., February, 1–4. http://lahoramaker.com/2016/02/01/la-hora-maker-010-fablabs-makespaces-

Gerlach, J. M. (1994). Is This Collaboration? In Bosworth, K. and Hamilton, S. J (eds.), *Collaborative Learning: Underlying Processes and Effective Techniques*. San Francisco, CA: Jossey-Bass, 5–14.

Gilster, P. (1997), Digital literacy. New York: Wiley Computer Publications.

Golub, J. & NCTE Committee. (1988). Focus on Collaborative Learning: Classroom Practices in Teaching English. Urbana, IL; USA, National Council of Teachers of English Publishing

Guthrie, C. (2014). Empowering the hacker in us: A comparison of fab lab and hackerspace ecosystems. In: *5th LAEMOS (Latin American and European Meeting on Organization Studies) Colloquium, Havana Cuba,* 2 (5).

Hatch, M. (2014). The Maker Movement Manifesto. The Maker Movement Manifesto.

Halonen, J., & Aksela, M. (2018). Non-formal science education: The relevance of science camps. LUMAT: International Journal on Math, Science and Technology Education, 6(2), 64–85. https://doi.org/10.31129/LUMAT.6.2.316

Halverson, E., Sheridan, K. (2014). The Maker Movement in Education. Harvard Educational Review 84 (4), 495–504. doi: https://doi.org/10.17763/haer.84.4.34j1g68140382063

Hobbs, R. (2017). Create to learn: Introduction to digital literacy. Retrieved from https://ebookcentral.proquest.com

Hsu, PS., Lee, E.M., Ginting, S., Smith, T., & Kraft, C. (2019). A Case Study Exploring Non-dominant Youths' Attitudes Toward Science Through Making and Scientific Argumentation. *International Journal of Science and Math Education*, 17, 185–207. https://doi.org/10.1007/s10763-019-09997-w

Jaatinen, J., Linfors, A. (2019). Makerspace for Innovation Learning: How Finnish Comprehensive Schools Create Space for Makers. *Design and Technology Education: An International Journal*, 24(2), 42-66. https://ojs.lboro.ac.uk/DATE/article/view/2623

Janghorban, R., Roudsari, R., & Taghipour, A. (2014). Skype interviewing: The new generation of online synchronous interview in qualitative research. *International Journal of Qualitative Studies on Health and Well-being*, 9(1). http://doi.org/ 10.3402/qhw.v9.24152

Jenkins, T. (2002). On the difficulty of learning to program. School of Computing; University of Leeds. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.596.9994&rep=rep1&type=pdf

John R. H., Hafner C. A. (2012). Understanding digital literacies, New York: Routledge.

Jonker, J., & Pennink, B. (2010). The Essence of Research Methodology: A Concise Guide for Master and PhD Students in Management Science. In Jonker, J., & Pennink, B. (Eds.), *The Essence of Research Methodology*. Springer-Verlag Berlin Heidelberg. http://doi.org/10.1007/978-3-540-71659-4

Jordan, T. (2008). Hacking: Digital Media and Technological Determinism. Polity Press.

Kazakoff, E. (2015). Technology-based Literacies for Young Children: Digital Literacy through Learning to Code. In: Heider K., Renck Jalongo M. (eds), *Young Children and Families in the Information Age. Educating the Young Child (Advances in Theory and Research, Implications for Practice)*, vol 10. Springer, Dordrecht. https://doi.org/10.1007/978-94-017-9184-7_3

Khromov S., Kameneva, K. (2016). Modern approach to digital literacy development in education. *Open Education*, (1), 60-65. https://doi.org/10.21686/1818-4243-2016-1-60-65

Kong, S-C., Abelson, H., Lai, M. (2019). Introduction to Computational Thinking Education. In S.-C. Kong and H. Abelson (eds.), *Computational Thinking Education*, https://doi.org/10.1007/978-981-13-6528-7_11

Kotilainen, S., Okkonen, J., Vuorio, J., Leisti, K. (2020). Youth Media Education in the Age of Algorithm- driven Social Media. *The Handbook on Media Education Research*. Wiley Blackwell.

Kordaki, M. (2012). Diverse categories of programming learning activities could be performed within Scratch. Procedia: Social and Behavioral Sciences, 46. 1162-1166. https://doi.org/10.1016/j.sbspro.2012.05.267

Laal, M., Laal, M. (2012). Collaborative learning: what is it? *Social and Behavioral Sciences*, 31(1), 491-495.

Lankshear, C., & Knobel, M. (2015). Digital Literacy and Digital Literacies: Policy, Pedagogy and Research Considerations for Education. Nordic Journal of Digital Literacy.

Lieberman, J. N. (1977). Playfulness: Its relationship to imagination and creativity. New York, NY: Academic Press.

Levy, S. (2010). Hackers: Heroes of the Computer Revolution. Beijing: OReilly.

Leung, L. (2015). Validity, Reliability, and Generalizability in Qualitative Research. *Journal of Family Medicine and Primary Care*, 4, 324-327.

Lundqvist, M., Kiviniemi, J. (2017). Digital tinkering and experimentation nooks in youth centres. In Kiviniemi, J., Tuominen, S. (Ed.), *Digital youth work—A Finnish perspective*, Verke. Access: https://www.verke.org/uploads/2021/01/a3f0ad24-digital-youth-work-a-finnish-perspective_verke.pdf

MacGregor, J. (1990). Collaborative learning: Shared inquiry as a process of reform. *New Directions for Teaching and Learning*, 42, 19-30. https://doi.org/https://doi.org/10.1002/tl.37219904204

Maeda, J. (2004). *Creative Code: Aesthetics + Computation*. Thames & Hudson.

Martin, A., & Grudziecki, J. (2006). DigEuLit: Concepts and Tools for Digital Literacy Development. Innovation in Teaching and Learning in Information and Computer Sciences. https://doi.org/10.11120/ital.2006.05040249

Martinez, S. L., & Stager, G. (2013). *Invent to learn: Making, tinkering, and engineering in the classroom*. Torrance: Constructing Modern Knowledge Press.

Meij. V., Broerse, M., & Kupper, F. (2017). Conceptualizing playfulness for reflection processes in responsible research and innovation contexts: a narrative literature review. Journal of responsible innovation, 4(1), 43-63. https://doi.org/10.1080/23299460.2017.1326258

Menezes, K., Pretto, N. (2019). Pirâmide da Pedagogia Hacker: de sonhos coletivos a engajamentos reais (Pyramid of Hacker Education: from collective dreams to real engagements). Teias A2. (2017/2018). http://doi.org/10.12957/teias

Merriam, S. B. (2009). Qualitative research: A guide to design and implementation. San Francisco, CA: Jossey-Bass

Okkonen, J., Kotilainen, S. (2019). Minors and artificial intelligence – implications to media literacy. *Advances in Intelligent Systems and Computing*, vol. 918. Cham: Springer.

Palinkas, L. A., Horwitz, S. M., Green, C. A., Wisdom, J. P., Duan, N., & Hoagwood, K. (2015). Purposeful Sampling for Qualitative Data Collection and Analysis in Mixed Method Implementation Research. *Administration and policy in mental health*, 42(5), 533–544. https://doi.org/10.1007/s10488-013-0528-y

Panitz, T. (1999). Collaborative versus Cooperative Learning: A Comparison of the Two Concepts Which Will Help Us Understand the Underlying Nature of Interactive Learning. *Cooperative learning and College teaching*, 8(2), 5-7.

Pariser, E. (2012). The filter bubble: how the new personalized web is changing our what we read and how we think. New York, NY: Penguin Books.

Parviainen, J., Coeckelbergh, M. (2020). The political choreography of the Sophia robot: beyond robot rights and citizenship to political performances for the social robotics market. AI & Society. https://doi.org/10.1007/s00146-020-01104-w

Patrick H., Mantzicopoulos P. (2015). Young Children's Motivation for Learning Science. In: Cabe Trundle K., Saçkes M. (eds), *Research in Early Childhood Science Education*. Springer, Dordrecht. https://doi.org/10.1007/978-94-017-9505-0_2

Pegrum, M., Dudeney, G., & Hockly, N. (2018). Digital literacy revisited. *European Journal of Applied Linguistics and TEFL*, 7(2), 3-24.

Peppler, K. A. ., & Kafai, Y. B. (2009). Creative coding: Programming for personal expression. 8th International Conference on Computer Supported Collaborative Learning (CSCL).

Pöllänen, S. (2009). Contextualising craft: Pedagogical models for craft education. *International Journal of Art & Design Education*, 28(3), 249–260.

Pollari, Pirjo; Salo, Olli-Pekka; and Koski, Kirsti. (2018). In Teachers We Trust – the Finnish Way to Teach and Learn. *Inquiry in education,* 10(1).

Ptaszek, G. (2020). Media Education 3.0? How Big Data, Algorithms, and AI Redefine Media Education. *The Handbook on Media Education Research*. Wiley Blackwell.

Pun, M. 2021. Plagiarism in Scientific Writing: Why It Is Important to Know and Avoid. *Journal of Political Science,* 21, 109-118.

Radziwill, N., Romano, J., Shorter, D., & Benton, M.C. (2015). The Ethics of Hacking: Should It Be Taught?. *Software Quality Professional*, 18(1), p. 11-15. ArXiv:abs/1512.02707.

Raymond, E. (2001). *How to become a hacker*. http://www.catb.org/~esr/faqs/hacker-howto.html

Rayner, T. (2018). Hacker Culture and the New Rules of Innovation. Boca Raton, FL: Routledge.

Rice, L. (2009). Playful Learning. Journal for Education in the Built Environment. https://doi.org/10.11120/jebe.2009.04020094

Richterich, A., & Wenz, K. (2017). Making and Hacking: Introduction. Digital Culture & Society, 3(1), 5-21. https://doi.org/10.14361/dcs-2017-0102

Romi, S., Schmida, M. (2009). Non-formal education: a major educational force in the postmodern era, *Cambridge Journal of Education*, 39(2), 257-273, http://doi.org/10.1080/03057640902904472

Rushkoff, D. (2010). *Program or Be Programmed: Ten Commandments for a Digital Age*. New York: OR Books. doi:10.2307/j.ctt207g7rj

Saarikoski, P. (2011). Computer courses in Finnish schools during 1980-1995. In *History of Nordic Computing*. Working Conference on History of Nordic Computing, Stockholm 18th–20th October 2010.

Sicart, M. (2019). Play Matters. In Play Matters. https://doi.org/10.7551/mitpress/10042.001.0001

Smith, B., MacGregor, J. (1992). What Is Collaborative Learning?. In Goodsell, A., Maher, M., Tinto, V., Smith, B. & MacGregor, M. (Eds.), *Collaborative Learning: A Sourcebook for Higher*

Education. The National Center on Postsecondary Teaching, Learning, and Assessment. Pennsylvania State University

Smith, B., Ng-A-Fook, N., Radford, L., & Pratt, S. S. (2018). Hacking education in a digital age: teacher education, curriculum, and literacies. Charlotte, NC: Information Age Publishing, Inc.

Stallman, R. (2002). On Hacking. http://stallman.org/articles/on-hacking.html (Retrieved May, 2020).

Stager, G. (2017). Papert: Father of the maker movement. *Hello World magazine*(1), 25.

Snyder, H. (2019). Literature review as a research methodology: An overview and guidelines. *Journal of Business Research*, 104, 333–339. https://doi.org/https://doi.org/10.1016/j.jbusre

s.2019.07.039

Suiter, T. (2013). Why "Hacking"? In Cohen D., & Scheinfeldt T. (Eds.), *Hacking the Academy: New Approaches to Scholarship and Teaching from Digital Humanities,* pp. 6-10. Ann Arbor: University of Michigan Press. doi:10.2307/j.ctv65swj3.4

Suominen, A. H., Jussila, J., Lundell, T., Mikkola, M., & Aramo-Immonen, H. (2018). *Educational Hackathon: Innovation Contest for Innovation Pedagogy*. In I. Bitran, S. Conn, K. R. E. Huizingh, O. Kokshagina, M. Torkkeli, & M. Tynnhammar (Eds.), *ISPIM 2018: Proceedings of the ISPIM Innovation Conference: Innovation, The Name of The Game (pp. 1-17)*. LUT Scientific and Expertise Publications, Reports, 78. Lappeenranta University of Technology; ISPIM.

Tardif, A. (2020, September 27). Is Sophia Robot using AI or is it a Marketing Stunt?. *Unite.AI*. https://www.unite.ai/is-hansons-robotics-sophia-robot-using-ai-or-is-it-a-marketing-stunt/

The New London Group. (1999). A pedagogy of Multiliteracies Designing Social Futures. Multiliteracies: Lit Learning, 66(1), 19–46. https://doi.org/10.4324/9780203979402-6

Tolppanen, S., Vartiainen, J., Ikävalko, V-L., Aksela, M. (2015). Relevance of non-fomral education in science education. *Relevant Chemistry Education – From Theory to Practice*, 335–354.

Thomas, D. (2002). *Hacker culture*. Minneapolis: University of Minnesota Press.

Tisza, G., Papavlasopoulou, S., Christidou, D., Iivari, N., Kinnula, M., & Voulgari, I. (2020). Patterns in informal and non-formal science learning activities for children–A Europe-wide survey study. International Journal of Child-Computer Interaction. https://doi.org/10.1016/j.ijcci.2020.100184

Tuomi, P., Multisilta, J., Saarikoski, P., & Suominen, J. (2018). Coding skills as a success factor for a society. *Educ Inf Technol* 23, 419–434. https://doi.org/10.1007/s10639-017-9611-4

Tuominen, S. (2017). Brief history of Finnish digital youth work. In Kiviniemi, J., Tuominen, S. (Ed.), *Digital youth work—A Finnish perspective,* Verke. Access: https://www.verke.org/uploads/2021/01/a3f0ad24-digital-youth-work-a-finnish-perspective_verke.pdf

UNESCO. (2018). A Global Framework of Reference on Digital Literacy Skills for Indicator 4.4.2. http://uis.unesco.org/sites/default/files/documents/ip51-global-framework-reference-digital-literacy-skills-2018-en.pdf

Urbi, J. Sigalos, M. (2018, June 5). The complicated truth about Sophia the robot — an almost human robot or a PR stunt. *CNBC*. https://www.cnbc.com/2018/06/05/hanson-robotics-sophia-the-robot-pr-stunt-artificial-intelligence.html

Valtonen, T., Tedre, M., Mäkitalo, K., and Vartiainen, H. (2019). Media literacy education in the age of machine learning. *The National Association for Media Literacy Education's Journal of Media Literacy Education,* 11 (2), 20 – 36.

Vee, A. (2013). Understanding Computer Programming as a Literacy. Literacy in Composition studies. 1(2). http://dx.doi.org/10.21623%2F1.1.2.4.

Vee, A. (2017). Coding Literacy: How Computer Programming Is Changing Writing. USA: The MIT Press.

Vincent, J. (2018, January 18). Facebook's head of AI really hates Sophia the robot (and with good reason). *The Verge.* https://www.theverge.com/2018/1/18/16904742/sophia-the-robot-ai-real-fake-yann-lecun-criticism

Wahyuni, D. (2012). The research design maze: understanding paradigms, cases, methods and methodologies. *Journal of applied management accounting research*, 10 (1), pp.69-80.

Whitton, N. (2018). Playful learning: Tools, techniques, and tactics. Research in Learning Technology. https://doi.org/10.25304/rlt.v26.2035

Williamson, B (ed.). (2015). Coding/learning: Software and digital data in education. A report from the ESRC Code Acts in Education project. Stirling: University of Stirling. Available at: https://codeactsineducation.wordpress.com/codinglearning-e-book/

Wing, J. (2006). Computational thinking. *Communications of the ACM* (49), 33–35.

Wizel, M. (2017). Teachers as Hackers: Implications for 21st Century Teacher Education (Dissertation). Lesley University, ProQuest Dissertations Publishing, 2017. 10288839.

Wizel, M. (2018). Preparing Educational Hackers. http://dx.doi.org/10.5772/intechopen.77036

Wong, B., Kemp, P. (2018). Technical boys and creative girls: the career aspirations of digitally skilled youths. *Cambridge Journal of Education*, 48:3, 301-316. http://doi.org/10.1080/0305764X.2017.1325443

APPENDICES

Appendix 1: Interview questions

1. First of all, please tell me a brief introduction about yourself (expertise, research focus/teaching subject, position at workplace, duration of teaching, any other experience related to your teaching field, etc.).
2. From your viewpoint, what is the relation between computational thinking and code literacy?
3. How is programming education integrated with math and other subjects (especially handicraft) in Finnish National Curriculum?
4. How do the hands-on activities support students in learning programming?
5. To what extent does students' easy access to tools and devices determine their success in learning to code? How easily can Finnish students access to tools and devices required for learning programming (e.g., assembly kits)? What are the learning equipments, coding software and programming languages that your class is using?
6. What is the proportion of boys and girls in your coding class? In general, is there any difference between boys' and girls' interest and ability in coding?
7. What do you think we should do to nurture the specialized students who are more interested in ICT-related activities (such as coding) or who are at a more advanced level in coding than their peers?
8. To what extent is students' collaboration important in code literacy education? How do your students collaborate with each other in the class?
9. What kind of activities that can be deemed as manifestations of playfulness in code literacy education?
10. How students' creativity and tinkering skills can be enhanced through digital fabrication/computational thinking/computer programming?
11. What kind of coding products that your students typically produce (e.g. video games, robots)? What are your students' perceptions of computer programming?
12. What are the main challenges of teachers/youth workers when organizing coding hands-on activities in the class? What are the main challenges of Finnish teachers, especially math teachers,

in combining two disciplines (math and CS) in their teaching? What are they doing well up to now?

13. As an expert/a teacher in the field of code literacy education, are the terms "hacker" and "hacker culture" familiar to you?
14. In technology education, “maker culture” has recently been under the spotlight, while the notion of “hack” (e.g., in *hackathon*, *hackerspace*) and hacker culture seem not to be paid much attention as such, even though it somehow is the stem for ‘maker culture’ to be flourished. What do you think the reason(s) for this phenomenon?

Appendix 2: Consent form for interviewees

INTERVIEW WITH TEACHERS AND EDUCATIONAL EXPERTS

Informed consent for interviewees

You are invited to participate in an interview which is carried out as part of the research titled “Integrating hacker culture into code literacy education”. This research is conducted as a master’s thesis within the Digital Literacy Education program at the Tampere University and it will be published online at the University library system.

Please take as much time as you need to read the following information and sign the accompanying consent.

Purpose of the study and the interview

We are asking you to take part in this interview because we are trying to acquire better understanding of how the characteristics of hacker culture are integrated into code literacy education in Finland.

You are identified as an educational expert in the field of code literacy education. Your participation will help to (1) examine how the characteristics of hacker culture are integrated into learning environment, learning activities and learning outcomes code literacy education and (2) analyze its relation with maker culture.

Procedures

You will be interviewed by the author of the master’s thesis who will ask you some questions about your experience regarding integrating maker and hacker culture into your teaching practice. The interview will take approximately 30-45 minutes and will be carried out online (e.g. via Zoom or

Microsoft Teams). The interview will be recorded (both audio and video). The recording will be transcribed fully and remove any names mentioned in the interview.

Potential risks and conflicts

There are no anticipated risks and conflicts of interest to your participation.

Confidentiality

The data obtained from this interview, such as the interview recording and transcripts, will only be used for the master's thesis described above. Everything you say during the interview is kept confidential.

Your name and other identifiable information will be kept anonymous, unless explicit permission is obtained from you to share your name to acknowledge your participation in this thesis project and/or to be cited in selected quotes from the interview.

Right to Refuse or Withdraw

You can choose whether to be part of this interview or not. If you volunteer to participate in this interview, you may withdraw at any time without consequences of any kind. During the interview, you may also refuse to answer any questions that make you feel uncomfortable and still remain in the study. You may be withdrawn from this research if circumstances arise which warrant doing so (e.g. conflict of interest).

Contact

Should you have any questions or concerns, please feel free to contact the author of the master's thesis (Yen Can) at yen.can@tuni.fi

I hope that you are provided with sufficient information. I would like to take this opportunity to thank you in advance for your assistance with this research, which I greatly appreciate.

Consent

Your signature below indicates that you have decided to volunteer as a research participant for this study, and that you have read and agreed with the information provided above.

Name: ……..…

Date: ……...

Signature