Inspiring or confusing – a study of Finnish 1–6 teachers' relation to teaching programming

Ray Pörn[1], Kirsti Hemmi[2] and Paula Kallio-Kujala[2]

[1] Faculty of Technology and Seafaring, Novia University of Applied Sciences, Vaasa, Finland
[2] Faculty of Education and Welfare Studies, Åbo Akademi University, Vaasa, Finland

There is limited research on teaching and learning of programming in primary school and even less about aspects concerning teaching programming from teachers' viewpoint. In this study, we explore how Finnish 1-6 primary school teachers (N=91), teaching at schools with Swedish as the language of instruction, relate to programming and teaching of programming, one year after the introduction of the new national curriculum that included programming. The teachers' relation to programming is studied by analyzing their view on programming, perceived preparedness to teach programming and their attitudes towards teaching programming. The main results of the present study are that the responding teachers approach programming in school with mixed emotions, but the majority claim to have sufficient preparedness to teach programming, and many of them have a positive attitude towards the subject. The findings indicate that the most important factor for high perceived preparedness and positive attitude is sufficient domain knowledge. The teachers' views on programming are very diverse, ranging from focusing only on the connection to elementary step-by-step thinking to more sophisticated reasoning connecting to central aspects of computational thinking and other educational outcomes. The findings suggest that there is a need for educational efforts to make the connection between mathematical content and programming more visible for primary school teachers.

Keywords: elementary education, mathematics education, programming, teacher professional development, 21st century abilities

ARTICLE DETAILS

LUMAT General Issue
Vol 9 No 1 (2021), 366–396

Received 18 May 2020
Accepted 7 May 2021
Published 1 June 2021

Pages: 31
References: 52

Correspondence:
ray.poern@novia.fi

https://doi.org/10.31129/LUMAT.9.1.1355

1 Introduction

Digital technology affects our daily lives, and programming and coding are at the very heart of this technology. Teaching computer science and programming, or in a broader sense computational thinking (Papert, 1996; Wing, 2006), has been a much-discussed subject in education during the last decade. There has been an increasing emphasis on integrating computer science into the school curriculum from lower grades in several countries (e.g. Schulte, Hornung, Sentence, Dagiene, Jevsikova, Thota, et al., 2012; Brown, Sentence, Crick & Humphreys, 2014; Duncan & Bell, 2015). This trend is to some extent driven by economic and technological demands for a future workforce (Chen, Shen, Barth-Cohen, Jiang, Huang, & Eltoukhy, 2017), but it has also been stressed that computational thinking and code literacy are important skills for full participation in modern society (Dufva & Dufva, 2016). The notion of

code literacy relates to the understanding of code and to the intentions and context of the code. “In the same way that not all literate individuals become authors, not all code-literate individuals become developers. Still, literate people have the necessary skills and the apprehension of reading and writing.” (Dufva & Dufva, 2016, p. 2). In addition, authors point out that computer science education in early grades influences students’ persistence in the domain and therefore, also their future career choices (Margolis, Estrella, Goode, Holme, & Nao, 2010; Yardi & Bruckman, 2007).

A practical way to enhance children’s code literacy is to incorporate programming at an early stage in the educational system. Several countries, for example, Finland, Sweden, Estonia, the United Kingdom and the United States have included programming in the national core curriculum. This is accomplished in different ways (Hubwieser, Armoni, Giannakos, & Mittermeir, 2014; Hubwieser, Armoni, & Giannakos, 2015). Some countries have introduced computer science as a subject of its own, Computing in England (Department for Education, 2013), while others have decided to integrate programming into other subjects, by, for instance, making programming an interdisciplinary element throughout the curriculum. This is the case in Finland, where programming is included in the generic competencies to be developed in all subjects and explicitly integrated in mathematics and handicraft (FNBE, 2016).

The path from the inclusion of programming in the national core curriculum to enacting lessons targeting it in a relevant manner is complex (Mannila, Dagiene, Demo, Grgurina, Mirolo, Rolandsson, & Settle, 2014). As Mannila et al. point out there are several issues to be discussed and defined to succeed in the implementation process. For example, it is not clear what exactly should be taught at different grade levels, neither what materials should be used. Primary school teachers as generalists need widespread professional development concerning technical skills and understanding of suitable pedagogies to successfully implement new curriculum ideas (Benton, Hoyles, Kalas, & Noss, 2017).

While some studies describe the use and impact of specific programming tools and classroom activities (Falloon, 2016; Sáez-López, Román-González, & Vázquez-Cano, 2016), only a few explore programming from the perspective of primary school teachers. Yet, teachers’ knowledge, beliefs and motivation are important to consider if we are to succeed in laying a solid ground in all students’ computational thinking and awakening their interest towards programming and technology (e.g. Ertmer, 2005; Ball, Thames, & Phelps, 2008; Sentance, Sinclair, Simmons, & Csizmadia,

2018; Hubwieser et al., 2015). The overall aim of this study is to investigate Finnish primary school teachers' relation to programming and to teaching of programming. The following research questions (RQ) guide the study:

RQ1: What are the studied primary school teachers' views on programming?
RQ2: What is the studied primary school teachers´ perceived preparedness to teach programming?
RQ3: What are the studied primary school teachers´ attitudes towards teaching programming?

2 Literature review

2.1. Programming and computational thinking in school

Programming for K-12 students was first introduced in the 1960s when Logo programming was presented as a potential framework for teaching mathematics (Feurzeig & Papert, 2011; Lye & Koh 2014 p. 52). During programming activities, students are engaged in computational thinking that involves general central concepts from computer science. The origins of computational thinking in mathematics education can be traced back more than thirty years to the work of Papert who developed computer software to facilitate children to engage and explore computer programming as a natural problem-solving tool in their mathematics studies (Papert, 1980, 1996). The term re-entered the pedagogical research community in 2006 when Wing pronounced that computational thinking represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use. She suggested the definition "computational thinking involves solving problems, designing systems and understanding human behavior, by drawing on the concepts fundamental to computer science" (Wing, 2006 p. 33). After that, several organizations and authors have presented different definitions of computational thinking. For example, the International Society for Technology in Education (ISTE, 2020) views computational thinking and its application as a cross-curricular skill. The core components of computational thinking according to ISTE: decomposition; gathering and analyzing data; abstraction; algorithm design; and how computing impacts people and society. These definitions and views are quite general and may indeed involve activities not necessarily directly connected to programming and coding.

Brennan and Resnick (2012) proposed a programming-based view on computational thinking, focusing especially on visual programming using Scratch for K-12 students. They introduced a framework with three dimensions of computational thinking: computational concepts, computational practices and computational perspectives. The first dimension includes concepts that programmers commonly use as they develop programs, such as variable, iteration and function. Computational practices reflect different problem-solving practices that occur in the programming process, such as testing, debugging and reusing. The third dimension, perspectives, involves the programmer's connection and relationship to other members of the programming community and to the surrounding technological world.

The definitions and interpretations of computational thinking are diverse, but the essence in computational thinking comprises at least thinking in a way that can be represented and processed by machines to facilitate a solution. A model is needed for representation and a set of structured computational steps (algorithm) is required for its solution.

In a recent study, Popat and Starkey (2019) reviewed research identifying educational outcomes, other than computer science and computational thinking, of programming in school. Their results concluded that when students are learning to code, a range of other educational outcomes could be learnt or practiced through the process of learning coding. These included mathematical problem-solving, critical thinking, social skills, self-management and more general academic skills. Students learning to code are coding to learn, according to Popat and Starkey (2019).

Criticism against the term computational thinking in an educational context is often based on the lack of consensus of the exact meaning of the term and its multiple interpretations. In addition, the relevance and importance of computational thinking as a general skill in everyday life has been questioned (Grover & Pea, 2013, p. 40). Some authors question the claim that computational thinking is an important skill for all students to learn and others react to why computational thinking should be superior to other types of thinking processes (e.g. Denning, 2017).

2.2. Teachers' relationship to programming

There are many studies about the important role of a teacher when implementing new ideas in school curriculum (e.g. Guskey, 2002; Hijón-Neira, Santacruz-Valencia, Pérez-Marín, & Gómez-Gómez, 2017). It has also been pointed out that teachers' content knowledge and pedagogical knowledge are important and affect student

progress and achievement in mathematics (Ball, Thames & Phelps, 2008; Baumert, Kunter, Blum, Brunner, Voss, Jordan, Klusmann, U., et al., 2010). Thorough content knowledge is important also in teaching of programming as several studies witness students' difficulties in learning concepts related to programming (e.g. Cetin, 2013; Denner, Werner & Ortiz, 2012). Moreover, the way in which teachers relate to new ideas is crucial for successful curriculum reform. Still, while there is a growing body of studies focusing on students' learning of programming at different school levels (e.g. Sáez-López et al. 2016; Chen et al., 2017; Duncan & Bell, 2015), only a few studies focusing on primary school teachers' and teaching of programming can be found.

Misfeldt, Szabo and Helenius (2019) investigated mathematics teachers' conception of the relationship between mathematics and programming. The results suggest that the teachers, on average, feel that there is a relationship between the two subjects and that mathematics teachers are interested in working with programming but that they do not feel well prepared for taking on that task. Funke, Geldreich and Hubwieser (2016) interviewed six primary school teachers about their views of computer science, and the findings pointed out that the teachers had no clear image of what computer science in school is, but they highlighted the importance of implementing computer science in an early educational stage. Govender and Grayson (2008) studied pre-service teachers' experiences when learning object-oriented programming. These pre-service teachers mostly talked about programming as finding a niche to design or upgrade a program that is needed somewhere. When probing the connections between programming and problem solving, they agreed that there must first be a problem, and then a program is constructed to solve it (Govender & Grayson, 2008). Recently Nouri, Zhang, Mannila & Norén (2019) investigated which skills 19 teachers interested in programming themselves aimed to develop among pupils. Apart from Brennan and Resnick's (2012) dimensions, they found some general skills related to digital competency and 21st century skills.

Several researchers highlight the critical role of the teachers in making explicit and systematic links between programming and students' existing and developing mathematical knowledge (e.g. Benton et al., 2017; Hickmott, Prieto-Rodriguez & Holmes, 2018). Hickmott et al. (2018) state that there is a lack of empirical studies that include concrete ideas or practices for K–12 educators that explicitly link the learning of mathematics and computational thinking. Kilhamn and Bråting (2019) investigate the relationship between programming and algebra in school. They emphasize the awareness of possible pitfalls concerning syntax and semantics in these

two areas to avoid confusion among students when working with, for example, algorithms and variables.

Mannila et al. (2014) surveyed teachers' experiences about and perceptions of computational thinking in five European countries, Finland, Italy, Lithuania, Netherlands, and Sweden. An important contribution of the study was the revealing of different aspects of computational thinking that already has become a part of teachers' classroom practices and how this is done. The survey data suggest that some teachers are already involved in activities that have strong potential for introducing some aspects of computational thinking. One of the concluding questions for further research in that study was students' and teachers' attitudes and experiences of introducing computational thinking in the classroom.

There are some studies investigating teachers' attitudes towards programming (Cetin & Ozden, 2015; Cetin, 2016; Hijón-Neira et al., 2017). Cetin (2016) shows that the pre-service teachers who learned programming through Scratch mastered central concepts better than the pre-service teachers in the control group did. They also experienced learning programming more meaningful. Hijón-Neira et al. (2017) investigated primary school teachers' views on programming in schools in one region in Spain through a questionnaire, and they analyzed the responses of 46 teachers. The teachers agreed on the benefits that programming provides in several areas, for example the development of thinking skills, the organization of ideas, the ability of abstraction and problem solving, motivational aspects, and the opportunities offered by teaching through games.

3 Context and educational setting of the study

We commence by briefly describing the Finnish school context and the role of programming in the national core curriculum (FNBE, 2016).

The comprehensive school (grades 1-9, ages 7-16) in Finland is the same for all students as there is no tracking. Hence, the national core curriculum is the same for all students. A primary school teacher teaches almost all subjects in grades 1-6, including mathematics. Primary school teachers in Finland are highly educated since they have a master's degree in education. Finland has two official languages, Finnish (88.7 %) and Swedish (5.3 %). According to existing legislation, education is organized separately for both language groups in parallel monolingual schools that follow the same national core curricula. Approximately 5 % of students in compulsory education attend a school where Swedish is the language of instruction. The target group of this

study is primary school teachers that work in schools where the instructional language is Swedish.

The Finnish national core curriculum describes generic competencies as a way to meet the challenges of the future world related to the 21st century skills and integrative instruction across school subjects. Programming is included in the generic competence of information and communication technology. The general task of mathematics education is to develop students' logical, accurate and creative thinking. Programming is included in the content of mathematical thinking skills and applies to all students from grade 1 up to the end of grade 9 (Hemmi, Krzywacki, & Partanen, 2017). Learning programming in mathematics starts in grades 1-2 with constructing simple algorithmic instructions by using symbols in written or oral form and testing them. During grades 3-6, the emphasis is on formulating instructions in a graphical programming environment. Programming is also included in the subject of handicraft from grade 3. In handicraft, students should practice programming through activities in, for example, robotics and automation. In grades 7-9, students develop and deepen their algorithmic thinking and their skills in applying programming in the mathematical problem-solving process.

In Finland, the national core curriculum only offers a general frame, and the municipalities, schools and teachers are to concretize the curriculum intentions (Hemmi, Lepik, & Viholainen, 2013). Teachers can freely choose their curriculum resources, and the mathematics textbooks are commercially produced without any national control (Hemmi, Krzywacki, & Koljonen, 2017).

4 Materials and methods

In this section, we present the data material used, the methods of data collection and how the data is analyzed.

4.1 The questionnaire

The empirical data for this study was obtained using a web-based survey that was sent to Swedish primary schools (grade 1-6) in Finland. The survey contained 39 questions and was divided into four different sections. Some questions were obligatory, some were optional, and some questions were branched. The minimal number of answered questions for each respondent was 28, and the maximal number was 39, depending on the number of optional and branched answers. Section 1 asked for background

information on the respondents. Questions on programming were posed in section 2, and section 3 contained questions related to teaching programming. Finally, in Section 4 there were questions on participation in in-service training and the possibility to give general comments. In the present study, answers from 18 questions (obligatory or optional) were collected and analyzed. The connection between the research questions and these survey questions are presented in section 4.3.

4.2. Data collection and informants

The data collection was carried out from the 3rd of April 2017 to the 10th of May 2017, and some preliminary results were reported in an unpublished thesis in educational sciences (Kallio-Kujala, 2017). In all, 110 teachers answered the questionnaire, but 19 answers were removed from the data material due to incompleteness. The final group of respondents consisted of N=91 teachers, 70 female and 21 men. The regional distribution of the respondents between the three major Swedish-speaking regions of Finland where 37/29/19 and six respondents came from other parts of Finland.

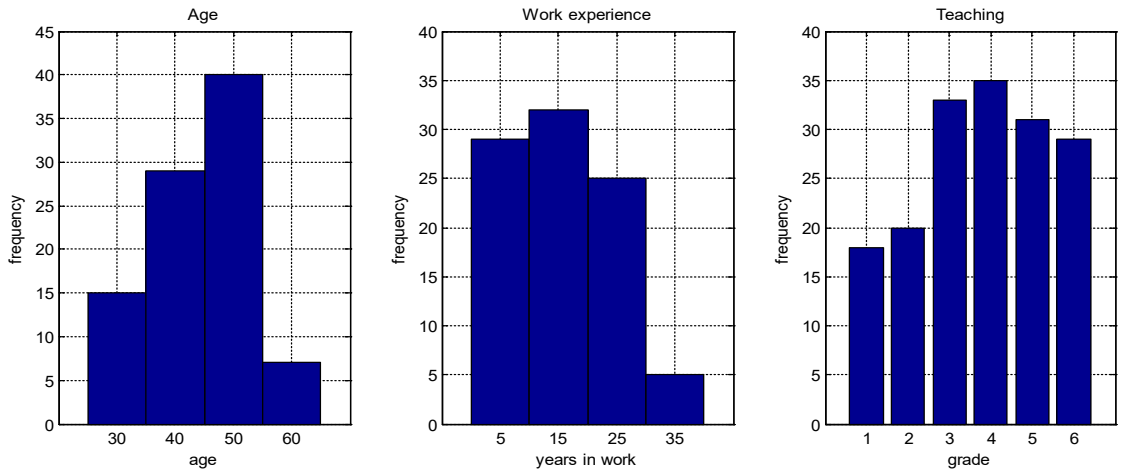

Figure 1. Background overview on the respondents' age, teaching experience and which grade they teach during the on-going year.

Most teachers had an age from 35 to 55 years. The distribution of teachers' work experience was evenly distributed between 0-30 years with a few teachers exceeding 30 years. The extended number of responses in the rightmost chart in Figure 1 is due to that several teachers teach multiple grades. Out of the 91 responding teachers, 84 was certified primary school teachers with a master's degree in education. Four respondents had a bachelor's degree, and three lacked a university degree in education. Programming is explicitly mentioned in the curriculum in mathematics

(grade 1-6) and handicraft (grade 3-6) and 86 of the respondents had taught mathematics and 35 had handicraft during the ongoing school year.

4.3. Connection between the survey and research questions

Below, we explain how the research questions are connected to different survey questions.

RQ1: What are the studied primary school teachers' views on programming in school?

This research question was answered by analyzing the teachers' answers of the open question: "What is programming? Please, focus on programming in primary school, but you can also discuss programming in general." From the context of the questionnaire, it is evident that this question is directly related to how the current change of the national curriculum (inclusion of programming) affected the mathematics content. The analysis of teachers' answers, following Bryman (2001), uses an iterative data-driven approach, in our case, going through several cycles of analysis. First, the teachers' written responses (in Swedish) were read and summarily analyzed. During this step, we identified certain similarities and generalities among the answers, which lead to the identification of six different categories. Thus, the identified six categories were a result of the analysis. Then the teachers' responses were read again, interpreted, and assigned to categories, in an iterative approach, by the authors.

RQ2: What is the studied primary school teachers´ perceived preparedness to teach programming?

The term perceived preparedness is associated with teachers' preparation before the curriculum reform. This preparation includes familiarization with the new curriculum, preempting relevant teaching material, collaborative preparation with colleagues, support from school management and self-perceived preparation level. Seven questions in the survey correspond to this topic. The teachers responded to the questions on a 6-point scale ranging from "no … at all" to "very much … ", with alternatives 1-3 on the negative side and 4-6 on the positive side. The translated questions can be found in Appendix A.

The teachers also responded (yes/no) to whether or not they had participated in in-service training. A two-sided unpaired *t*-test with α-level 0.05 was conducted to compare if there were a significant difference in the perceived preparedness for those that had participated in in-service training compared to those that had not participated. Both samples are approximately normally distributed, and the standard deviations are approximately equal. To complement the quantitative items, the teachers had the possibility to respond to two open questions that relate to their perceived preparedness to teach programming:

- What type of support and help have you obtained from your colleagues or school on how to teach programming?
- What type of material do you have and how have you obtained it?

The teacher responses to the last question were categorized into six different categories using an open data-driven approach similar to the one used in RQ1.

RQ3: What are the studied primary school teachers´ attitudes towards teaching programming?

Teachers responded to statements related to their attitude to programming and to teach programming in primary school. Five statements correspond to this topic. The teachers responded to the statements on a 6-point Likert scale (e.g. Oppenheim, 2000) with alternatives 1-3 on the negative side and 4-6 on the positive side. The translated statements can be found in Appendix A. These five statements are similar with the computer programming attitude scale developed by Cetin and Ozden (2015). Their scale included affection, cognition and behavior as three dimensions of attitude. The cognitive dimension consists of beliefs about the attitude object, the affective dimension includes feelings towards the object, and the behavioral dimension refers to action tendencies towards the object. The five scale items in this study relate to affection and cognition. A two-sided unpaired *t*-test with α-level 0.05 was conducted to compare if there were a significant difference in attitude for those that had participated in in-service training compared to those that had not participated. Both samples are approximately normally distributed, and the standard deviations are approximately equal. To complement the quantitative scale items, the teachers also answered one multiple-choice question and two open questions related to attitude.

- Which of the following words describes your emotions to teach programming? (See Table 3)
- What do you think has influenced your attitude to teach programming in primary school?
- State your arguments why students should learn programming in primary school.

The teacher responses to the second question were categorized into five different categories using an open data-driven approach similar to the one used in RQ1.

5 Results

Next, we report the findings and results of our analysis of the teachers' relation to teaching programming. We follow the order of the RQs.

5.1. Analysis and result of RQ1: views on programming

The teachers' views on programming and teaching programming were categorized as 1) sequential, 2) logical, 3) algorithmic, 4) problem-solving, 5) technological and 6) progressional (see Table 1). Due to the openness of the question, one answer could be assigned to several categories. Below in this section, we explain and describe the categories in more detail and exemplify them with teachers' responses translated to English. The number of words in the different teacher answers (in Swedish) varied from one word to 108 words and the mean number of words in the answers were 25. The distribution of the teachers' views on programming in relation to the six analytical categories can be seen in Table 1. One answer can be assignment to several categories. The number of answers assigned to different number of categories are: 0 (6), 1 (48), 2 (24), 3 (10), 4 (2), 5 (1). That is, six answers were uncategorized and 24 answers belonged to two different categories. No answer was assigned to all six categories. The total number of assigned answers are 139.

Below, we describe and exemplify the categories identified for teachers' views on programming in school. A single excerpt often below to several categories. Underlining has been used to indicate belonging to a certain category. Some of the results related to RQ1 has recently been published in a proceeding paper (Pörn, Hemmi, & Kallio-Kujala, 2021).

Table 1. Distribution of teachers' views on programming with respect to the six categories

Category (view on programming)	*n* (% of 91 teachers)
1. Sequential	59 (65)
2. Logical	29 (32)
3. Algorithmic	10 (11)
4. Problem-solving	17 (19)
5. Technological	9 (10)
6. Progressional	15 (16)
Total	**139**

Sequential view

The sequential view connects programming with the explicit action of giving (or writing or following) step-by-step instructions to a computer, robot or fellow student. This category is the most common among the answers as 65 % of the teachers' response could be connected to this. Teachers connect these kinds of actions to activities associated with spatial thinking and step-by-step procedures. This is exemplified in the following answers:

> In primary school education, it is important to let students test to program a computer, give instructions to another person or to a robot and try to make it complete the desired task. (Teacher 10)
> A simple way is to say; Go two steps to the right, one backwards and then five steps forward. Then you have come to the finish. (Teacher 33)
> Programming is to give detailed step-by-step instructions that do not offer space for misinterpretations or ambiguity. (Teacher 72).

Several teachers pointed out that the instructions need not to be given to a computer or robot, but equally well to a fellow student.

Logical view

This category was the second most common as 32 % of the teacher responses point out that programming is connected to logical thinking or the identification of patterns. Most responses in this category state that programming promotes the development of logical thinking, as shown in the following extracts:

> Programming is, for example, to split a problem into smaller parts, to see relations, to learn to think logically, to create something new. (Teacher 64)
> I think programming is very much about logical thinking and recognizing patterns. (Teacher 45).

Many teachers connect programming to a combination of handling instructions and applying logical thinking.

Algorithmic view

The algorithmic view is connected to the central concepts of computer science and development of programs such as algorithm, abstraction, modularization, planning and testing. Eleven percent of teachers' responses are categorized as algorithmic. A typical example is the following excerpt:

> It is about coding, solving complex problems by splitting them into smaller pieces, identifying patterns, creating abstractions and writing algorithms. (Teacher 16)

Teachers that connect to these concepts may have more in depth knowledge of programming and to the process of applying programming to solve problems.

Problem-solving view

In this view programming is connected to the usage as a mathematical problem-solving tool. This aspect of programming is highlighted in 19 % of the answers.

> Programming is a really good activity that trains the ability to solve problems. (Teacher 43)
> Programming is about logical thinking, ability to solve problems, systematics, and creativity . . . Programming is mathematics. (Teacher 73)

Despite the close and important connection between mathematical problem solving and programming, no teacher answer is giving any explicit example of such a problem-solving activity.

Technological view

A few teacher descriptions (10 %) consider programming from a more general perspective that involves the connection to modern technology and digitalization of society. Some responses address directly the importance of understanding the relation between human and modern technology:

> To realize that everything a machine can do is due to a human that has programmed it. (Teacher 10)

> Several things in our close environment work with aid of programming, e.g. machines, computer games and telephones. Industry uses robots that have been programmed. (Teacher 75)

This category captures more general aspects of programming, pointing out the human-machine relationship and connection to modern technology.

Progressional view

The aspects of curriculum and progression concerning programming in primary school and comments on the importance of knowledge for the future work-life are present in 16 % of the answers. Several teachers saw programming as a positive element in mathematics lessons and important for all students to learn, for example to prepare for future work life.

> We have to prepare them for the working life after school when they must be prepared to think creatively. (Teacher 58)

On the other hand, there were teachers who were not convinced about the importance of learning programming for all students and those that lack clear information on the progress throughout the grades 1-6.

> I think programming is fun, but I do not see it as a useful subject. That type of thinking can be acquired in many other ways. (Teacher 22)
> Interesting, but I would like to have a clearer plan about what to do each school year. (Teacher 69)

Variation in teachers' descriptions

Due to the openness of the question, the range and the depth in teachers' responses varied a lot. Some of the teachers touched several categories while others only responded with short sentences categorized into one category. The following extract is an example of the former and was coded into categories 1, 2, 3 and 6.

> Programming is a working process were you construct an algorithm, a hypothesis or a plan of how something should be executed or work. This plan is then tested and updated in order to work correctly. On a basic level, it can be as easy as working with numbered instructions. For older students it proceeds to the creation of block-based events using apps and computer programs and then finally in the highest grades by coding using a text-based language. (Teacher 62)

This teacher captures several important concepts and practices in computational thinking, such as instructions, events, algorithm, planning and testing as well as the progression of the topic. The next example reflects a logical, algorithmic and problem-solving view on programming.

> Programming is all about logical thinking and problem solving. It is about coding, solving complex problems by splitting them into smaller pieces, identifying patterns, creating abstractions and writing algorithms. You can practice programming using different programs, games and languages. Programming is a new way of thinking. (Teacher 16)

The focus in this answer is on problem-solving, the thinking aspect and the creation of algorithms and abstractions. The last example is coded into categories 2, 4 and 5.

> Programming is a way to teach students logical thinking, understanding of relations and problem solving. They develop both cognitively and linguistically. In time, they will understand that all new technology they use is based on programming. (Teacher 40)

This teacher specifically lifts logical thinking and problem solving as important learning outcomes and the technological view is also present. The answer also highlights the communicative (social) aspect of programming as being important. This social aspect of programming was mentioned in two answers.

Connections to mathematical content

There were few answers that explicitly connected programming to other mathematical content. The reason for this could indeed be the openness of the survey question. Still, no teacher answer spontaneously relates programming activities to measurement, arithmetic expressions, equation solving, nor probability. Some answers made connections to the broad area of problem-solving but provided no explicit example of what type of problem-solving was actually involved. The few examples with mathematical content can be connected to elementary spatial thinking (how to move along a pre-defined path) and simple geometrical shapes (how to form a square).

5.2. Analysis and result of RQ2: perceived preparedness to teach programming

The seven items connected to preparedness were summed together to get a measure of the teachers' overall perceived preparedness to teach programming. The internal consistency reliability (Cronbach's alpha) was 0.88 for this composite variable. The correlation (Pearson's r) between the seven items ranged from 0.26 to 0.81. Figure 2 shows the distribution of the variable "Perceived preparedness" on the scale 1-6 and the mean values of the seven individual items. Fifty-six (62 %) of the teachers scored higher than 3.5 (positive side responses) on the perceived preparedness scale, and 35 (38 %) scored lower than 3.5 (negative side responses). The perceived preparedness of the responding teachers is on the positive side, but several teachers also express a clear lack of knowledge about programming and an explicit need for more education on the subject. The following teacher comments address this need:

> First, I need to know what programming actually means! (Teacher 10)
> I wish for more clarity in what to exactly teach and on which grade. Education is needed. (Teacher 37)
> Much more education and support is needed, since this topic is completely new to me. (Teacher 29)

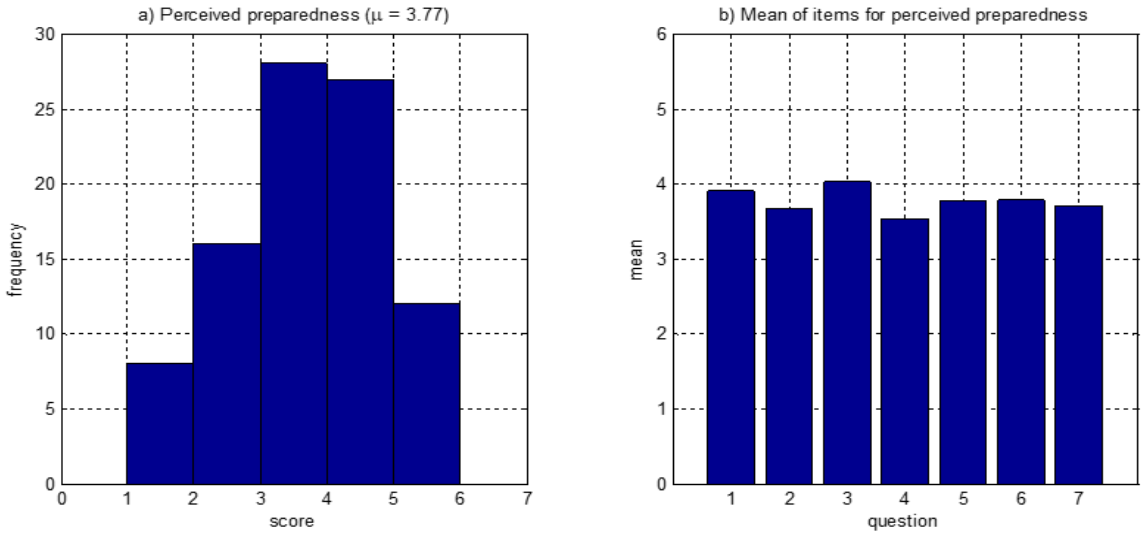

Figure 2. a) Distribution of teachers' perceived preparedness to teach programming, b) mean of individual items for perceived preparedness.

Seventy-one (78 %) of the teachers had participated in at least one in-service training. There was a significant difference in the means for perceived preparedness between group 1 (in-service training; n=71, m=3.972, sd=1.036) and group 2 (no in-service training; n=20, m=3.064, sd=1.130) with t(89)=3.392, p=0.001. The result

indicates that teachers who have participated in in-service training had higher perceived preparedness than those that have not.

In addition, the teachers responded to the open question: What type of support and help have you obtained on how to teach programming? Sixty-two (68 %) of 91 teachers claimed that they had obtained sufficient support from their school and colleagues in their preparation to teach programming in primary school. Common examples of supporting factors were the possibility to participate in many in-service training events, discussions with colleagues, explicit interest from and engagement by the principal and systematic visits at school by local ICT-tutors. On the negative side, most comments reflected on inadequate equipment and material at school and some pointed out the lack of a broader discussion regarding a more holistic perspective on the implementation of programming in primary school.

> More practical examples is needed on how to embed programming into a primary school context so that we don't have to do programming just for programming itself. (Teacher 75)

The teachers also responded to the open question: What type of material do you have and how have you obtained it? Fifty-three teachers answered this optional open question. The answers were categorized based on how they had obtained their material. In Table 2, one answer can belong to several categories.

Table 2. Categorization of how the teachers had obtained their teaching material and artifacts.

Self-made	Purchased	Borrowed	Web-based	From in-service training	From colleagues
9	15	6	24	14	10

Self-made material includes, for example, different types of programming cards and games for unplugged activities. A typical example of purchased material was educational robots (e.g. BeeBots, Sphero, Lego Mindstorms). It is also common for schools in the same municipality to have shared material pools with more expensive educational material that can be borrowed. It is most common to obtain programming material from the internet and many teachers use web-based tools like Scratch and code.org as well as applets like ScratchJr and Lightbot. Since many of the responding teachers had participated in in-service training courses, some material is directly obtained through those events. Ten respondents mention explicitly that a colleague provides the material.

5.3. Result of RQ3: attitudes towards teaching programming

The five items connected to attitude were summed to get a measure of the teachers' overall attitude to teach programming. Cronbach's alpha was 0.79. The correlation between different items ranged from 0.13 to 0.73. Figure 3 shows the distribution of the variable "Attitude" on the scale 1-6 and the mean values of the five individual items. Eighty (88 %) of the teachers scored higher than 3.5 on the attitude scale and 11 (12 %) scored lower than 3.5. The responding teachers' attitude to teach programming in primary school is clearly on the positive side. There was a significant difference in the means for attitude between group 1 (in-service training; n=71, m=4.64, sd=0.779) and group 2 (no in-service training; n=20, m=3.888, sd=0.872) with t(89)=2.381, p=0.02. The result indicates that teachers who have participated in in-service training scored slightly higher on attitude towards teaching programming than those that had not.

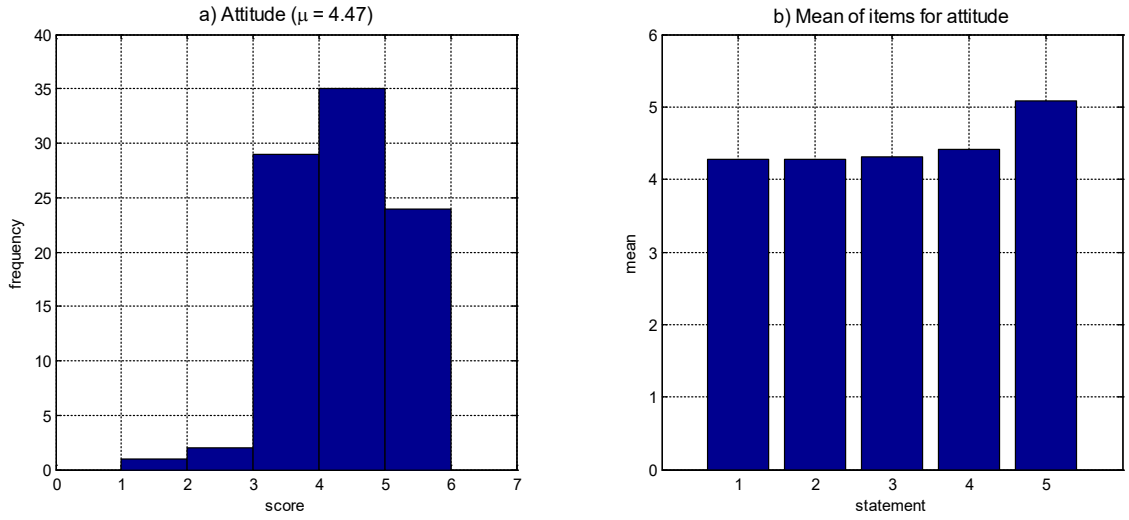

Figure 3. a) Distribution of teachers' attitudes to teach programming,
b) mean of individual items for attitude.

In addition, teachers responded to the multiple-choice question: Which of the following words describes your emotions to teach programming? One respondent could choose several words. Table 3 shows that the majority of the answers relate to positive emotions, like inspiration, motivation, joy, enthusiasm and involvement. Many of the teachers also felt insecurity, confusion and even annoyance. Fifty-eight (64 %) of the teachers reported that they felt emotions associated with negative words, 70 (77 %) had positive emotions and 40 (44 %) had mixed (both positive and negative) emotions towards teaching programming. Three respondents did not select a single

word. Nine of the respondents stated that they felt both inspired and confused when it comes to teaching programming in primary school.

Table 3. Words that primary school teachers chose to describe their emotions to teach programming.

Negative	***n***	**Positive**	***n***
Annoyance	12	Enthusiasm	33
Fright	0	Inspiration	41
Confusion	23	Involvement	27
Desperation	1	Joy	36
Dread	4	Motivation	41
Insecurity	47	Optimism	25
Indifference	7	Passion	3
Total	**94**	**Total**	**206**

Fifty-eight teachers responded to the question "What do you think has influenced your attitude towards teaching programming in primary school?" The open answers were categorized into the following six categories displayed in Table 4. One answer can belong to several categories.

Table 4. Categorization of factors that influenced teachers' attitude to teach programming.

Own interest	**Participation in in-service training**	**External factors**	**Digitalization and technical development**	**Interested students and colleagues**	**Other**
12	17	12	7	9	8

Participation in in-service training seems to be an important factor for attitude in teachers' experience, as well as own interest in the subject together with external factors. Some teachers also mention students' interest and eagerness to program as a factor that explicitly influenced their own attitude towards teaching programming. Some teacher excerpts that express factors with positive influence on attitude are:

> My interest in mathematics and logic and the knowledge that this is something that is here to stay. (Teacher 1)
> My curiosity and interest for technical development. (Teacher 9)
> The fact that I have seen how interested students are of programming. (Teacher 20)
> It was, without a doubt, the in-service training that gave me courage to try. (Teacher 32)

Several teacher answers highlighted to positive impact of participating in in-service training. The responses that were categorized into the five categories where all positive. Six negative and two positive teacher responses to this question were labelled

"Other". Some of these critical teacher comments reflected insecurity, annoyance, confusion and opposition to teach programming:

> I feel that I am the only one in this world that do not know what this is all about. Neither has it come to my attention why this should be taught in school nor why it is so important. (Teacher 10)
> The management in our school believes that if something is technical or digital it is a good thing. The management does not care about pedagogical content. (Teacher 21)
> I try to avoid everything about programming and give that responsibility to those that are interested. (Teacher 50)

Finally, teachers responded to the open question: State your arguments why students should learn programming in primary school. Some teacher comments were:

> Everybody has the right to learn the basics of programming. There is a future demand for programmers. (Teacher 42)
> Everyone will clearly not need programming, nevertheless it is beneficial to know something about programming. (Teacher 29)
> It is not programming itself that is important, it is the additional value it gives to the student's mathematical thinking and reasoning skills. (Teacher 74)
> There are much more important things to learn. Let the professionals do the programming. (Teacher 50)

The first comment considers programming as an important skill that is certainly useful for everybody, while the second comment tones down the usefulness but agrees on that it is valuable to have some basic knowledge. The third comment focus on the possibility of a supportive effect of programming activities to mathematical thinking and reasoning skills in general. The last comment is clearly negative to the teaching of programming in school due to other more important content. These four examples are representative for the 24 responses to this optional question.

5.4. Correlation between the results concerning different RQs

Finally, we analyzed possible connection and correlation between the different research questions. The composite variables perceived preparedness and attitude are dependent. The correlation (Pearson's r) between perceived preparedness and attitude is 0.58.

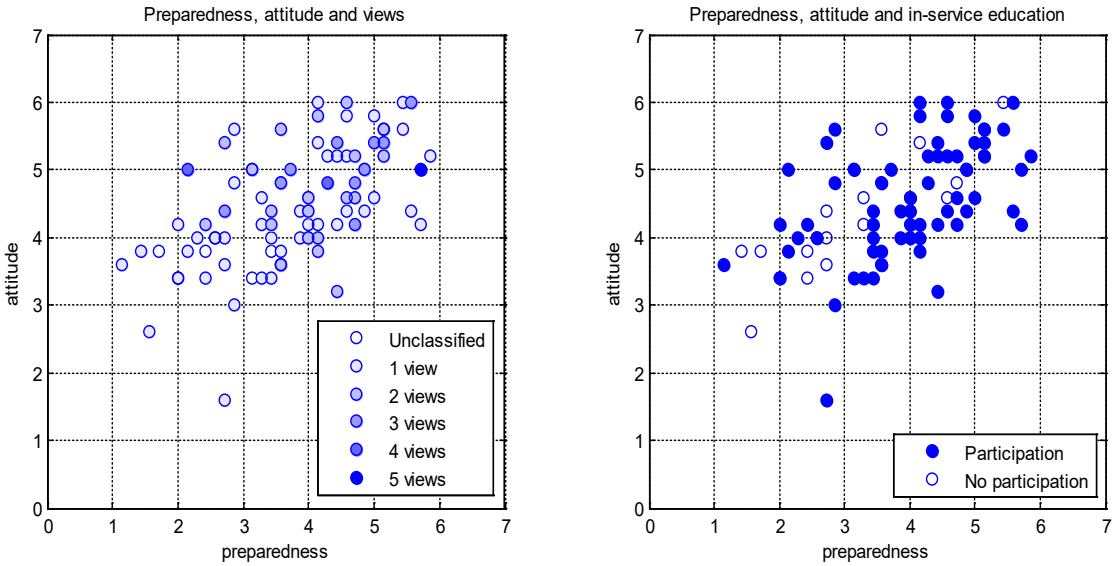

Figure 4. a) Perceived preparedness versus attitude with variety of view as shades of blue. b) Perceived preparedness versus attitude and participation in in-service education.

The variety of the teachers' views on programming increases slightly with preparedness and attitude. Many teachers with high preparedness and positive attitude express a broader and deeper view of programming. The teachers that participated in in-service education had higher perceived preparedness and attitude than those that had not participated. Note that 13 points overlap in Figure 4a and Figure 4b (78 distinct points, 91 data points).

5.5. Summary of results

The teachers' views on programming are very diverse. Most of the teachers in the study had a sequential view on programming that mainly connected programming to writing, giving and following of instructions. Programming was also considered to contribute to the development of logical thinking, serve as a valuable tool in problem-solving and be a useful skill in future work life (technological and progressional view).

Factors that contributed to a high perceived preparedness level were attendance in in-service training courses, supporting discussions with colleagues and existence of

relevant teaching material at their school. On the other hand, several teachers had an unclear view on what programming in primary school actually is and some teachers expressed a clear lack of knowledge regarding programming and highlighted an explicit need for more education and support on the subject. The results also showed that teachers that had participated in in-service training courses had higher perceived preparedness than those that had not.

Several teachers saw programming as a positive element in the new national curriculum and important for students to learn. Many of the teachers (44 %) in the study approach programming with mixed emotions. For example, they feel inspired and confused or enthusiastic and insecure at the same time. Inspired and enthusiastic, because programming was considered a modern, relevant and useful topic and many teachers stressed that students have the right to learn programming in school to prepare for future work life. Some teachers felt that they were doing important and valuable work, and programming was also considered a source for inspiration and creativity in the mathematics classroom. In addition, participation in in-service courses gave the confidence to connect to the subject, and the interest and engagement by fellow colleagues and pupils were considered important factors that influenced their own attitude to the subject in a positive way. Some teachers also felt confused and insecure since programming is a new topic for almost all primary school teachers, and many of them found it challenging to position this new topic within the mathematics curriculum.

6 Discussion

The present study contributes with some knowledge regarding teachers' views, perceived preparedness and attitudes of introducing programming in the primary school classroom.

The study reveals that, in all the Swedish speaking regions in Finland, there are teachers that are interested and deeply involved in the development of teaching programming in primary school (late Spring 2017). Although many of the teachers have mixed feelings towards teaching programming, a majority of the respondents consider themselves to have a sufficient level of perceived preparedness and a positive attitude. It is not possible to measure views, attitudes nor beliefs in an absolute sense (Reid, 2006). The reported perceived preparedness to teach programming does not necessarily correspond to actual preparedness. It might be that a teacher with a high perceived preparedness to teach programming has a limited and somewhat narrow

view on programming. For example, it can be the case that a teacher has in-depth knowledge of the Scratch program and experiences a high level of preparedness, but if another tool or environment is encountered the knowledge cannot be transferred to the new situation. Heintz and Mannila (2018) also noted and reflected on this when they summarized experiences from a large-scale computational thinking course in Sweden. Teaching programming has often been technology-driven and enthusiastic teachers and other actors have considered what they can do with a particular tool. Therefore, there might be a danger that a holistic picture of the learning path of children is not so clear for primary school teachers (Hemmi, Krzywacki, & Partanen, 2017).

The six identified categories of teacher views have connections to different frameworks for computational thinking developed in the literature. Some of the categories are clearly visible in the model of possible educational outcomes of programming in school (Popat & Starkey, 2019 p. 370) in the form of higher-order thinking skills (logical, algorithmic and problem-solving view) and curriculum and pedagogical design (technological and progressional view). The teachers' answers and the six identified categories also have a connection to the assessment framework by Brennan and Resnick. Many Finnish primary school teachers' use Scratch as a programming tool, and many have attended in-service training courses addressing Scratch. When they are to describe what they consider as programming, it might be that they, to some extent, view programming through the lens of Scratch.

Some of the teachers in this study had a broader and deeper view on programming in school, reflecting their knowledge, enthusiasm and engagement. A majority of the teachers had a positive attitude towards teaching programming, and they felt well prepared for this task. The findings suggest that participation in in-service training courses and education could have a positive impact on preparedness as well as on attitude and it may enrich the teachers' views on programming. Several teachers mentioned this as an important aspect also in their open responses.

On the other hand, some of the participating teachers expressed their lack of resources, content knowledge and a lack of a clear view of programming in school similar to the results of the study by Hijón-Neira et al. (2017) where the authors conclude, "However, many schools face serious teaching difficulties derived from the lack of adequate resources or properly trained teachers". Some teachers with lower perceived preparedness also had a more narrow, or negative, view on programming

in school. Several of them also questioned the purpose and potential benefit of the inclusion of programming in the national core curriculum.

There were only a few explicit connections to specific mathematical content among the views. The reason for this could indeed be the openness of the survey question. However, along the lines with the concerns mentioned by Benton et al. (2017), it might be that the primary school teachers do not fully apprehend the interplay between mathematical and programming content and learning. As several researchers point out, there is a need to make explicit the links between mathematics and programming for teachers (Benton et al., 2017; Hickmott et al., 2018; Kilhamn & Bråting, 2019).

This also relates to several teachers' concerns about lack of knowledge, information and relevant materials to be able to concretize the general goals of the national core curriculum. At the time of the study (Spring 2017), there was a lack of educational material for programming, especially material with a relevant and explicit connection to mathematics.

7 Limitations of the study

The teachers who completed the survey may not need to be representative for the whole population of primary school teachers in Finland teaching at schools with Swedish as the language of instruction. The survey was sent out to the principals in primary schools in Finland with Swedish as an instructional language. Participation in the survey was nonobligatory, and it is unclear if the principals enabled all teachers at their school to participate in the study or if the survey was directed only to a few active teachers at the school. Therefore, it is not possible to give any response rate for the survey.

It can also be the case that the sample of teachers in this study has a bias towards higher perceived preparedness and attitude than the "average" primary school teacher. It is likely that some of the respondents were "early adopters" that included programming in their mathematics classroom even before the implementation of the new national curriculum. It may also be the case that the principal directed the survey only to selected active teachers at his/her school.

Unpaired t-tests were conducted. The group sizes in the unpaired t-tests were different (n=71, n=20), but the sample variances were approximately equal. According to Rusticus and Lovato (2014), there is only a modest risk for errors when testing the difference in means between groups with unequal sizes and approximately equal variance.

It is also important to note that the measures of teachers' attitudes and preparedness were all based upon self-reported data.

8 Conclusions

In this paper, we studied 91 Finnish primary school teachers' relation to programming and to teaching programming by analyzing their views on the subject, perceived preparedness to teach the subject and their attitudes towards teaching the subject. Although our study concerns a specific context, the results are important for the international research field as it sheds light on a current issue, the teaching of programming in primary school mathematics from the teachers' perspectives. It is also valuable to have studied teachers' relation to programming directly after the curriculum implementation 2016. The results of our study are relevant to the international research field, as several countries are attempting to implement programming in primary school curriculum from lower grades. The case of Finland can reveal general aspects important to consider also in other countries and in further research targeting the inclusion of programming in primary school. The study also contributes to our knowledge about how primary school teachers relate to teaching programming.

Today primary school teachers have access to a variety of different tools and material when teaching programming. Although some of the responding teachers claimed to have a lack of or insufficient material, the crucial aspect is to use the material properly. To do so, sufficient domain knowledge of programming (and mathematics) is necessary. The findings in this study indicate that teachers' views on programming are very diverse, and this may lead to inequality in education. The findings suggest that participation in in-service training courses and education could have a positive impact on preparedness as well as on attitude, and it may enrich the teachers' views on programming. The findings also suggest that there is a potential need for educational efforts to make the connection between mathematical content and programming more visible for primary school teachers, for example, in the form of well-designed concrete exercises and pedagogical practices. Those working with teachers, teacher education and the production of study materials have an important role in this continuous endeavor.

Acknowledgements

The research project (Artisan) that this paper is based on is financed by Högskolestiftelsen i Österbotten.

References

Ball, D. L., Thames, M. H., & Phelps, G. (2008). Content Knowledge for Teaching What Makes It Special? *Journal of Teacher Education,* 59(5), 389–407.

Baumert, J., Kunter, M., Blum, W., Brunner, M., Voss, T., Jordan, A., Klusmann, U., et al. (2010). Teachers' mathematical knowledge, cognitive activation in the classroom, and student progress. *American Education Research Journal*, 47(1), 133–180.

Benton L., Hoyles C., Kalas I., & Noss R. (2017). Bridging Primary Programming and Mathematics: Some Findings of Design Research in England. *Digital Perspectives in Mathematics Education*, 3, 115–138. https://doi.org/10.1007/s40751-017-0028-x

Brennan, K. & Resnick, M., (2012). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American Educational Research Association*, Vancouver, Canada. http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf

Brown, N.C.C., Sentance, S., Crick, T., & Humphreys, S. (2014). Restart: The Resurgence of Computer Science in UK Schools. *ACM Transactions on Computing Education*, *14*(2), 9:1–9:22.

Bryman, A. (2001). Social Research Methods. Oxford: Oxford University Press.

Cetin, I. (2013). Visualization: A tool for enhancing students' concept images of basic object-oriented concepts. *Computer Science Education, 23*(1), 1–23.

Cetin, I., & Ozden M.Y. (2015). Development of Computer Programming Attitude Scale for University Students. *Computer Applications in Engineering Education*, 23(5), 667–672

Cetin I. (2016). Pre-service teachers' introduction to computing: Exploring utilization of Scratch. *Journal of Educational Computing Research*, 54(7), 997–1021.

Chen, G., Shen, J., Barth-Cohen, L., Jiang, S., Huang, X., & Eltoukhy, M. (2017). Assessing elementary students' computational thinking in everyday reasoning and robotics programming. *Computers & Education, 109*, 162–175.

Denner, J., Werner, L., & Ortiz, E. (2012). Computer games created by middle school girls: Can they be used to measure understanding of computer science concepts? *Computers & Education, 58*(1), 240–249.

Denning P.J. (2017). Remaining trouble spots with computational thinking. *Communications of the ACM*, 60(6), 33–39. https://doi.org/10.1145/2998438

Department for Education. (2013). National Curriculum in England: Computing programmes of study. https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study.

Dufva, T., & Dufva, M. (2016). Metaphors of code - Structuring and broadening the discussion on teaching children to code. *Thinking Skills and Creativity*, 22, 97–110.

Duncan, C., & Bell, T. (2015). A pilot computer science and programming course for primary school students. In *Proceedings of the Workshop in Primary and Secondary Computing Education*, ACM, 39–48.

Ertmer, P. A. (2005). Teacher pedagogical beliefs: The final frontier in our quest for technology integration. *Educational technology research and development, 53*(4), 25–39.

Falloon, G. (2016). An analysis of young students' thinking when completing basic coding tasks using Scratch Jnr. on the iPad. *Journal of Computer Assisted Learning*, *32*, 576–593.

Feurzeig, W., & Papert, S. A. (2011). Programming-languages as a conceptual framework for teaching mathematics. *Interactive Learning Environments*, *19*(5), 487–501.

Finnish National Board of Education (2016). National core curriculum for basic education 2014. Helsinki, Finland: Next Print Oy.

Funke, A., Geldreich, K., & Hubwieser P. (2016). Primary school teachers' opinions about early computer science education. In *Koli Calling '16: Proceedings of the 16th Koli Calling International Conference on Computing Education Research* (pp. 135-139). ACM. https://doi.org/10.1145/2999541.2999547

Govender, I., & Grayson, D. J. (2008). Pre-service and in-service teachers' experiences of learning to program in an object-oriented language. *Computers & Education*, 51(2), 874–885.

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

Guskey, T. R. (2002). Professional development and teacher change. *Teachers and teaching*, 8(3), 381-391.

Heintz, F., & Mannila, L. (2018). Computational Thinking for All – An Experience Report on Scaling up Teaching Computational Thinking to All Students in a Major City in Sweden. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (SIGCSE '18), 137-142, ACM.

Hemmi, K., Krzywacki. H., & Koljonen, T. (2017). Investigating Finnish Teacher Guides as a Resource for Mathematics Teaching. *Scandinavian Journal of Educational Research*, 1 – 18. https://doi.org/10.1080/00313831.2017.1307278

Hemmi, K., Krzywacki, H., & Partanen, A-M. (2017). Mathematics curriculum. The case of Finland. In D. R: Thomson, M. A. Huntley, & C. Suurtamm (Eds.), *International Perspectives on Mathematics Curriculum* (pp. 71-102). USA: Information Age Publishing Inc.

Hemmi, K., Lepik, M., & Viholainen, A. (2013). Analyzing proof-related competences in Estonian, Finnish and Swedish mathematics curricula - towards a framework of developmental proof. *Journal of Curriculum Studies*, *45*, 354–378. https://doi.org/10.1080/00220272.2012.754055

Hickmott, D., Prieto-Rodriguez, E., & Holmes, K. (2018). A Scoping Review of Studies on Computational Thinking in K-12 Mathematics Classrooms. *Digital Experiences in Mathematics Education*, *4*, 48–69.

Hijón-Neira, R., Santacruz-Valencia, L., Pérez-Marín, D., & Gómez-Gómez, M. (2017). An analysis of the current situation of teaching programming in Primary Education. In *Computers in Education (SIIE), International Symposium on Computers in Education (IEEE), 9-11 Nov. 2017, Lisbon, Portugal,* (pp. 1-6). IEEE. https://doi.org/10.1109/SIIE.2017.8259650

Hubwieser, P., Armoni, M., Giannakos, M. N., & Mittermeir, R. T. (2014). Perspectives and visions of computer science education in primary and secondary (K-12) schools. *ACM Transactions on Computing Education 14*(2), 7. https://doi.org/10.1145/2602482

Hubwieser, P., Armoni, M., & Giannakos, M. N. (2015). How to implement rigorous computer science education in K-12 schools? Some answers and many questions. *ACM Transactions on Computing Education, 15*(2), 5. https://doi.org/10.1145/2729983

International Society for Technology in Education, ISTE (2020). Computational thinking competencies. https://www.iste.org/standards/computational-thinking

Kilhamn, C. & Bråting, K. (2019). Algebraic thinking in the shadow of programming. In U. T. Jankvist, M. van den Heuvel-Panhuizen, & M. Veldhuis (Eds.), *Proceedings of the Eleventh Congress of the European Society for Research in Mathematics Education* (pp. 566-573).

Utrecht, the Netherlands: Freudenthal Group & Freudenthal Institute, Utrecht University and ERME. http://www.mathematik.uni-dortmund.de/~prediger/ERME/CERME11_Proceedings_2019.pdf

Kallio-Kujala P. (2017). Klasslärares beredskap och förhållningssätt till att undervisa i programming. Unpublished thesis in pedagogical sciences. *Åbo Akademi University*.

Lye, S.Y., & Koh, J.H.L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 41, 51–61.

Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., & Settle, A. (2014) Computational Thinking in K-9 Education. In *ITiCSE '14 Proceedings of the 2014 conference on Innovation & technology in computer science education* (pp. 1-29). ACM.

Margolis, J., Estrella, R., Goode, J., Jellison-Holme, J., & Nao, K. (2008). Stuck in the Shallow End: Education, Race, & Computing. MIT Press: Cambridge, MA.

Misfeldt, M., Szabo A. & Helenius, O. (2019). Surveying teachers' conception of programming as a mathematics topic following the implementation of a new mathematics curriculum. In U. T. Jankvist, M. van den Heuvel-Panhuizen, & M. Veldhuis (Eds.), *Proceedings of the Eleventh Congress of the European Society for Research in Mathematics Education* (pp. 2713-2720). Utrecht, the Netherlands: Freudenthal Group & Freudenthal Institute, Utrecht University and ERME. http://www.mathematik.uni-dortmund.de/~prediger/ERME/CERME11_Proceedings_2019.pdf

National Research Council NRC (2012). A framework for K-12 science education: Practices, crosscutting concepts and core ideas. *The National Academies Press*.

Nouri, J., Zhang, L., Mannila, L., & Norén, E. (2019). Development of computational thinking, digital competence and 21st century skills when learning programming in K-9. *Education Inquiry*, 1–17.

Oppenheim, A. N. (2000). Questionnaire design, interviewing and attitude measurement. *Bloomsbury Publishing*.

Papert, S. (1980). Mindstorms: children, computers, and powerful ideas. *NY: Basic Books*.

Papert, S. (1996). An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning*, *1*(1), 95–123.

Popat, S. & Starkey, L. (2019). Learning to code or coding to learn? A systematic review. *Computers & Education*, 128, 365–376

Pörn, R., Hemmi, K., & Kallio-Kujala, P. (2021). "Programming is a new way of thinking" – teacher views on programming as a part of the new mathematics curriculum in Finland. I Y. Liljekvist, L. Björklund Boistrup, J. Häggström, L. Mattsson, O. Olande, H. Palmér (Red.), *Sustainable mathematics education in a digitalized world. Proceedings of MADIF12*. SMDF.

Reid, N. (2006). Thoughts on attitude measurement. *Research in Science & Technological Education*, 24, 3–27. https://doi.org/10.1080/02635140500485332

Rusticus, S. A. & Lovato, C. Y. (2014). Impact of Sample Size and Variability on the Power and Type I Error Rates of Equivalence Tests: A Simulation Study. *Practical Assessment, Research & Evaluation*, 19(11). https://doi.org/10.7275/4s9m-4e81

Sáez-López, J. M., Román-González, M., & Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using "Scratch" in five schools. *Computers & Education, 97*, 129–141.

Sentence, S., Sinclair, J., Simmons, C., & Csizmadia, A. (2018). Classroom-Based Research Projects for Computing Teachers: Facilitating Professional Learning. *ACM Transactions on Computing Education 18*(3) 14. https://doi.org/10.1145/3171129

Schulte, C., Hornung, M., Sentance, S., Dagiene, V., Jevsikova, T., Thota, N., ... Peters, A. (2012). Computer science at school/CS teacher education: Koli working-group report on CS at

school. In *Koli Calling '12: Proceedings of the 12th Koli Calling International Conference on Computing Education Research* (pp. 29-38). ACM. https://doi.org/10.1145/2401796.2401800

Wing J.M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33–35, 2006.

Yardi S., & Bruckman A. (2007). What Is Computing? Bridging the Gap Between Teenagers' Perceptions and Graduate Students' Experiences. In *ICER '07 Proceedings of the third international workshop on Computing education research* (pp. 39-50). ACM.

Appendix A

Questions related to perceived preparedness

1. To what extent are you familiar with the parts of curriculum where programming is mentioned? I am
   1: not familiar at all 6: very much familiar
2. To what extent are you familiar with what the students are supposed to learn about programming? I am
   1: not familiar at all 6: very much familiar
3. How well do you consider that your school has relevant and useful material for teaching programming? Our school has
   1: no material at all 6: very much material
4. How well do you consider that you have relevant and useful material for teaching programming? I have
   1: no material at all 6: very much material
5. How much support have you obtained from your school in your preparation to teach programming? I have obtained
   1: no support at all 6: very much support
6. How much support have you obtained from your colleagues in your preparation to teach programming? I have obtained
   1: no support at all 6: very much support
7. How well prepared (knowledge, skills, material) do you consider yourself to be to teach programming in primary school? I feel
   1: not prepared at all 6: very well prepared

Appendix B

Statements related to attitude (1: I strongly disagree 6: I totally agree)

1. Programming is an important skill.
2. Programming is interesting.
3. It is important to teach programming in primary school.
4. I relate positively to teach programming in primary school.
5. I feel very insecure with new technology.

The responses to statement 5 were reversed.