Peter Vinnervik holds a PhD degree in Pedagogical work with specialisation in Technology. He is a certified school teacher in Mathematics, Science and Technology and has more than 20 years of experience as a teacher educator at Umeå University, with a special focus on digital transformation in schools.

Berit Bungum is Professor of science and technology education at The Norwegian University of Science and Technology, Trondheim, and works at the Resource Center for Mathematics, Science and Technology ("Skolelaboratoriet"). She has a broad background in research; she currently works with ways of incorporating programming with subject learning and creativity and is manager for the project KreTek.

PETER VINNERVIK
Centre for Educational Development, Umeå University, Sweden
peter.vinnervik@umu.se
https://orcid.org/0000-0002-4094-4503

BERIT BUNGUM
Department of Physics, The Norwegian University of Science and Technology, Trondheim, Norway
berit.bungum@ntnu.no
https://orcid.org/0000-0003-1739-7887

Computational thinking as part of compulsory education: How is it represented in Swedish and Norwegian curricula?

Abstract
In recent years, many countries have revised their school policy documents to incorporate digital competence, computational thinking and programming. This study examines and compares how and in what contexts Nordic curricula, Swedish and Norwegian in particular, embody aspects of computational thinking. Results show that only parts of the practices defined in the computational thinking framework used for analysis could be explicitly identified in the curriculum documents. The most salient computational thinking practice represented in both the Swedish and Norwegian curricula is programming, and programming is primarily recognized as a method and tool for learning other subject content and not as a knowledge domain in its own right. Both curricula leave leeway for teachers to implement a broader approach to computational thinking. However, this would require much time and considerable teacher competence, and the requirements for such an effort seems to be under-communicated in the curricula, leaving schools and teachers with major challenges.

INTRODUCTION

In recent years, school policy documents in many countries have been revised to incorporate emerging knowledge and skills considered important in a technology dependent society (Voogt & Roblin, 2012). Such knowledge and skills encompass, for example, information literacy, problem solving abilities and creativity, and are generally referred to as part of the broad notion of digital competence (European Education and Culture Executive Agency, Eurydice, 2019). Part of this increased international discourse on digital competence has come to include elements of computer science, for example programming (Crick, 2017; Passey, 2017).

Further, the broader concept of computational thinking (Barr, Harrison, & Conery, 2011; Barr & Stephenson, 2011; Bocconi et al., 2016; Grover & Pea, 2013; Voogt, Fisser, Good, Mishra, & Yadav, 2015) has caught the attention among school policy makers internationally. The notable interest for computing has also reached the Nordic compulsory education discourse. Among the Nordic countries, Finland (in 2016), Sweden (in 2018) and Norway (in 2020) have thus far introduced revised compulsory school curricula that emphasize the role and importance of digital competence for life and work in the future.

This can be seen as an example of travelling policy ideas (Ozga & Jones, 2006). Such ideas are always interpreted and contextualised in the varying national contexts, and it is therefore of interest to investigate how the current focus on programming and computational thinking is manifested in curricula in Nordic countries. This paper reports an explorative study of curriculum documents in Sweden and Norway, with the purpose to identify to what extent and how computational thinking practices are manifested. By this, we illuminate the potential for integrating computational thinking practices in compulsory school in these two countries. The study draws on a framework for computational thinking designed by Weintrop et al. (2016), and the research question is: *In what ways are computational thinking practices represented in curricula for subjects in Swedish and Norwegian compulsory education?*

BACKGROUND

Computational thinking (hereafter referred to as CT) derives from an academic debate which began in the mid 20th Century, about whether the thought processes and procedural aspects of computing could be beneficial for other knowledge domains, and hence become a sort of general-purpose thinking tool (Tedre & Denning, 2016). The exact expression, ‘computational thinking’, is generally thought to have been introduced in Seymour Papert’s seminal book on computer literacy for children and how programming could be a tool for thinking and learning (Papert, 1980). In 2006, the concept received renewed attention when Wing (2006) argued that CT “represents a universally applicable attitude and skill set” (p. 33) for solving problems, closely related to computer science but beneficial for and transferable to other knowledge domains. Wing proposed several characteristics that define CT and since then, a wide range of proposals to define CT have been designed (Shute et al., 2017). More elaborate frameworks have come from scholars (e.g. Brennan & Resnick, 2012; Weintrop et al., 2016) as well as from various stakeholders (Barefoot Computing, n.d.; BBC, n.d.). Despite the clear connection to computer science, and Papert’s efforts to introduce children to computer programming, CT is not to be equated with programming. Nevertheless, programming is generally seen as a tool that can facilitate the understanding and development of CT skills (Voogt et al., 2015). The discussion about what CT means, whether it is necessary to even talk about CT, and how the skills best can be developed and assessed is ongoing (Guzdial, 2020; Guzdial, Kay, Norris, & Soloway, 2019; Saqr, Ng, Oyelere, & Tedre, 2021).

Nordic approaches

In this section, we describe Nordic approaches to introducing CT in compulsory education at an overall level, before we present an analysis of curriculum documents in Sweden and Norway. A recent report from the European Schoolnet (Bocconi, Chioccariello & Earp, 2018) illuminates that CT as a concept is not explicitly deployed in Nordic school curricula. Instead, the predominant idea, as seen in Finland, Sweden, and Norway, has been to integrate programming in the curricula for existing subjects (Bocconi, Chioccariello & Earp, 2018). In all three countries, programming is part of the Mathematics curriculum. Programming is further included in the Swedish Technology subject (Teknik), and in Crafts in Finland. In Norway, programming is included in the Arts and Crafts, Music, and Science curricula, in addition to Mathematics.

In Denmark, digital competence permeates many contexts and subjects, but there is little about programming in the curriculum for compulsory school. The subject curriculum for Mathematics refers to the use of computer programs, but not how they are constructed (Børne- og Undervisningsministeriet, 2019a). In Science (Natural sciences/Technology), there is a strong focus on modelling competence, but computer programs are not mentioned (Børne- og Undervisningsministeriet, 2019b). However, a new subject called Technology comprehension has been piloted in many Danish schools between 2018 and 2021 (Børne- og undervisningsministeriet, 2021) and the subject includes CT as an explicit part of the content.

The Icelandic compulsory school curriculum includes an interdisciplinary subject, Information and Communication Technology, that covers a broad range of competences under the headline of information and media literacy. Learning to use digital tools and computer programs is central. Programming, however, is given a limited space in the Icelandic curriculum. It is mentioned once, in the Information and Communication Technology syllabus which declares that grade 10 students are supposed to be able to "use software for programming and communication in a creative manner" (Ministry of Education, Science and Culture, 2014, p. 237). Programming does not form part of the stated learning outcomes in Mathematics, Science, or other subjects.

THEORETICAL FRAMEWORK

To analyse the Swedish and Norwegian compulsory school curricula, an analytical framework had to be identified. The predominant part of the CT practices in both Sweden and Norway is manifested in the Mathematics, Science and Technology subject curricula. The CT framework developed by Weintrop et al. (2016) was therefore found to be a suitable tool for analysis and presentation of results. In the framework, CT is represented as four categories of practices (see Table 1), where each is composed of a subset of interrelated practices that relate to familiar conceptualisations of practices in science and technology. The *Data Practices* category represents skills and knowledge of the role and use of data for scientific and mathematical endeavours. The category *Modelling and Simulation Practices* represents skills and knowledge required to design, create, and use models and simulations to represent and explore scientific or mathematical phenomena with the help from computational devices. (However, in other subjects, 'model' may have different meanings). Both of these categories have similarities with recent development in how scientific practices are described as part of the science curriculum (Crawford, 2014). The *Computational Problem Solving Practices* category represents problem solving practices "especially effective for working with computational tools and derived from the field of computer science" (Weintrop et al., 2016, p. 138). The practices within this category encompass, for example, algorithm development, computer programming, and to systematically identify and correct encountered problems when computational tools are used. The category *System Thinking Practices* represents skills and knowledge necessary to understand systems (e.g., technological, biological, economical) and how its parts interact and function as a whole and change over time.

Table 1. Computational Thinking Framework (Weintrop et al., 2016)

Data Practices	**Modelling and Simulation Practices**	**Computational Problem Solving Practices**	**System Thinking Practices**
Collecting Data	Using Computational Models to Understand a Concept	Preparing Problems for Computational Solutions	Investigating a Complex System as a Whole
Creating Data	Using Computational Models to Find and Test Solutions	Programming	Understanding the Relationships within a System
Manipulating Data	Assessing Computational Models	Choosing Effective Computational Tools	Thinking in Levels
Analysing Data	Designing Computational Models	Assessing Different Approaches/Solutions to a Problem	Communicating Information about a System
Visualising Data	Constructing Computational Models	Developing Modular Computational Solutions	Defining Systems and Managing Complexity
		Creating Computational Abstractions	
		Troubleshooting and Debugging	

METHODS

In this study, we analysed the curriculum documents that govern teachers' work, and limited the analysis to the documents teachers themselves are most likely to use in their work. We have not analysed any strategic education policy documents, for example about school digitalisation strategies and so forth, since we consider such documents as not primarily intended to provide guidance for teachers' work. In concrete terms, the governing formal written curricula for compulsory schooling (age 6-16) in Norway (LK20) and Sweden (Lgr11, implemented in 2018) were selected as main targets of analysis. Both curricula follow the northern continental curriculum tradition represented by a subject-based approach (Mølstad & Karseth, 2016) and are structured around a core curriculum with values and principles for compulsory schooling, and curricula for subjects. A subject curriculum captures the overall aim of the subject and further presents the core content and specific learning outcomes, divided into age spans. Here, differences in structure, level of detail, and ideology between Norway and Sweden emerge. Common for both countries is that teachers are entrusted to make their own informed decisions about subject content in detail, teaching materials and methods (Mølstad & Karseth, 2016).

In addition, three official complimentary documents, so called commentary materials, to the Swedish formal written curriculum were included. These texts provide teachers with additional, and optional, information regarding the governing guidelines in the formal written curriculum regarding a specific subject or theme (e.g., the integration of digital tools and media in education). The data set also contains a framework for algorithmic thinking developed by The Norwegian Directorate for Education and Training. In total, six documents were selected for analysis (see table 2).

Table 2. Documents selected for analysis

Document	Publisher	Comment	Status
Norwegian compulsory school curricula (LK20), age 6-16 (Years 1-10)	Norwegian Directorate for Education and Training	Analysis of the core curriculum and subject curricula for Mathematics, Arts and crafts, and Natural Science	Governing
Algoritmisk tenkning	Norwegian Directorate for Education and Training	Norwegian framework for computational thinking	Guidance (Optional reading)
Swedish Curriculum for Compulsory School (Lgr11), age 7-16 (Years 1-9)	Swedish National Agency for Education (NAE)	Analysis of the subject curricula for Mathematics, Technology, and Natural Science subjects	Governing
Commentary materials to Lgr11	Swedish National Agency for Education (NAE)	Analysis of three commentary materials for the subjects Mathematics and Technology, and about digital tools and media in compulsory education.	Guidance (Optional reading)

The analysis was conducted through close reading and qualitative content analysis (Krippendorff, 2004). When the initial selection of documents was made (see table 2), the framework by Weintrop et al. (2016) was used as an analytical tool to identify the potential for CT practices. In the first step of analysis, we examined the documents to identify all instances related to the practice of programming, since programming is the main CT practice that has received most attention in the development and implementation of the curricula. This identification process was made via close reading and searching for significant keywords ('programming' and related stems, and potential related terms, e.g., 'coding'). In the succeeding step, we searched for instances of CT more broadly. This was done by searching for segments of text that explicitly or implicitly represent any of the CT practices defined in the framework. In the third step of analysis, each identified segment was carefully examined to establish which specific CT practice it represented. Table 3 provides an example of how a text unit was categorised in terms of main and subordinated CT practices.

Table 3. Example of analysis

Text unit	Source	CT practices
Technical solutions that use electronics and how they can be programmed	Curriculum for Technology, Core content (Lgr 11) Age 13-16	**System Thinking**: Understanding the Relationships within a System **Computational Problem Solving:** Programming

RESULTS

This section presents the results of the analysis for each country separately. First, a general overview of how digital competence and CT in a broad sense are embedded in the curriculum is presented. Then follows a presentation of how programming is conveyed in the various subject curricula. Each section concludes with a presentation of how specific CT practices in terms of the framework from Weintrop et al. (2016) are represented in the curriculum. This structure of presentation reflects the process of analysis, where the curricula were investigated first with regards to programming and thereafter with regards to CT in a broader sense.

Representations of CT in the Swedish curriculum

General aspects of the Swedish curriculum related to CT

The current formal curriculum was updated in 2017 to incorporate and emphasise digital competence. The Swedish digital competence framework draws on the EU Digital Competence Framework (Carretero et al., 2017) and contains the following competence goals (NAE, 2017a):

1. Understanding the impact of digitalisation on society
2. The ability to use and understand digital tools and media
3. Having a critical and responsible approach (towards media and information sources)
4. The ability to solve problems and put ideas into action

There are several abilities and skills that the recent and ongoing push for digital competence in the curriculum is believed to help develop, such as creativity, curiosity, self-confidence, design thinking, problem solving, collaboration, and critical thinking (NAE, 2017a).

Programming was added to the curriculum in the 2017 revision and explicitly integrated in three subjects: Mathematics, Technology, and to a lesser extent in Civics. Programming is described as a tool for learning subject specific content, for example "to explore problems and mathematical concepts, make calculations and to present and interpret data" (NAE, 2018, p. 55). A broader aim of programming emerges in one of the optional commentary materials:

> … programming must be seen in a broader perspective that also includes creative making, control and regulation, simulation, and democratic dimensions. This broader perspective on programming is an important starting point in teaching and programming is thus included in all aspects of digital competence (NAE, 2017a, p. 10).

However, this role of programming is not reflected in the formal written curriculum. We therefore believe that a potential consequence is that many teachers miss out on this more comprehensive declaration of intent. Research (Vinnervik, 2022) suggests that to fully capture the organisational structure and intentions of programming, it is necessary to read not only the formal written curriculum (Lgr11) but also the non-governing commentary materials for mathematics, technology, and the cross-curricular commentary material about digital competence.

CT as a concept is not formally used by the school authorities in Sweden. The concept is however mentioned once in the thematic commentary material about digital tools and media in compulsory education in which CT is described as a process that encompasses "problem solving, logical thinking, pattern recognition and creating algorithms that can be used in programming" (NAE, 2017a, p. 9). Here, it is further argued that these aspects of CT are captured in "different parts of the curriculum and subject curricula". We interpret this as the school authorities taking the position that CT practices are represented in the formal curriculum but sees no need to be more precise.

Programming in the Swedish Mathematics curriculum

The main message about programming is conveyed in the subject curricula for Mathematics and Technology. Additional information, for example about cross-subject collaboration and learning progression is provided through commentary materials. In the Mathematics curriculum, programming is tied to digital tools and is presented as such, to be used to "explore problems and mathematical concepts, make calculations and to present and interpret data" (NAE, 2018, p. 55). In the commentary material for Mathematics, it is stated that children should learn the 'programming fundamentals' (NAE, 2017b). These fundamentals are captured in three condensed statements in the Mathematics curriculum, all linked to algebra:

- How unambiguous, step-by-step instructions can be constructed, described and followed as a basis for programming. The use of symbols in step-by-step instructions (Years 7-9)'
- How algorithms can be created and used in programming. Programming in visual programming environments (Years 4-6)
- How algorithms can be created and used in programming. Programming in different programming environments (Years 7-9)

These statements illuminate that algorithms are essential constructs in programming. The statements further seem to link learning progression to the inherent complexity of the programming environment.

A fourth core content statement is related to the working area *Problem solving:*

- How algorithms can be created, tested and improved when programming for mathematical problem solving (Years 7-9)

Here, we see that programming is tied to solving mathematical problems rather than specific programming problems. Although the statement does not provide any additional details of what constructs (e.g., sequence, repetition) a programming algorithm may be based on, it adds some understanding for procedures and working methods associated with programming.
In the commentary material for mathematics, additional details about the curriculum intentions emerge. Here, the iterative, back-and-forth, aspect of programming as a process is emphasized, cautiously implied in the core content bullet point for problem solving presented above. There is also some information about the role of programming environments, their inherent complexity and learning progression. Students are expected to first use block based environments, but with increased age, they should be "given the opportunity to deepen and broaden their programming skills and how programming can be used" (NAE, 2017b, p. 17). Therefore, they should also be given the opportunity to program in "different programming environments" (NAE, 2017b, p. 17). Implicitly related to the issue of programming environments concerns programming languages, but no specific information about languages is found neither in the Mathematics texts, nor in any of the other texts. Learning progression is generally expressed as growth in knowledge and/or skills, but there is no such information in terms of programming knowledge in any of the Swedish texts.

Programming in the Swedish Technology curriculum

In the subject curriculum for Technology, programming is mentioned in four core content statements (NAE, 2018, pp. 297–299):

- Controlling objects by means of programming (Years 1-3)
- Controlling pupils' own constructions or other objects by means of programming (Years 4-6)
- Technical solutions that use electronics and how they can be programmed (Years 7-9)
- Pupils' own constructions in which they apply control and regulations, including with the aid of programming (Years 7-9)

Here, programming is mainly represented as a tool to attain technological knowledge rather than being technological knowledge in itself, and primarily tied to technological systems, methods of control and regulation, construction work and electronics. The overarching aim of Swedish technology education, however, covers other aspects of technology that concern both programming and digital competence. For example, aspects of digital safety (dimension 3 in the Swedish digital competence framework), consequences of technological choices and the impact of technology on people, society, and the environment (dimension 1) and problem solving (dimension 4) are represented in the Technology curriculum. Furthermore, conceptual understanding is one of five core abilities of technology education (NAE, 2018). This core ability could also encompass the understanding of programming concepts, even though no such concepts are explicitly defined in the subject curriculum.

The commentary material for technology (NAE, 2017c) reveals an intended organisational relationship between mathematics and technology with regards to programming. Here it is indicated that children should learn the ‘programming fundamentals’ in mathematics, but whether this knowledge is meant to scaffold and amplify the programming experience in technology is not clearly expressed.

The Swedish curriculum in terms of CT

Using the CT framework by Weintrop et al. (2016) as analytical tool, several categories and practices represented in the framework could be identified. At a category level, *Data Practices* is an already existing category of practices in the Swedish curriculum in several subjects. The CT perspective is potentially possible as there are guidelines in several subject curricula that informs teachers to collect, analyse and visualise data with digital tools, for example: “Tables and diagrams to describe the results of investigations, both with and without digital tools. Interpretation of data in tables and diagrams” (NAE, 2018, p. 58). The message is partly clarified in the thematic commentary material about digital tools and media in compulsory education. Here it is suggested that digital tools can be used to “make simple tables” as well as to “perform advanced and comprehensive calculations, handle large amounts of data or produce forecasts using mathematical models and programming” (NAE, 2017a, p. 22). This suggests that several of the CT practices subordinated to *Data Practices* could be covered. Worth noticing, however, is that the commentary materials are optional reading, which means that any clarifications provided in such texts may not be perceived by teachers.

The category *Modelling and Simulation Practices* is represented in the mathematics curriculum:

> Teaching should help pupils to develop their knowledge in order to formulate and solve problems, and also reflect over and evaluate selected strategies, methods, models and results (NAE, 2018, p. 55)

Even though this quote does not explicitly mention computational aspects, it reflects practices within the *Modelling and Simulation Practices* category, such as using (“selected”) and assessing (“reflect over and evaluate”) models and thereby, implicitly, opens for aspects of CT.

In the curriculum as a whole, the different categories of CT practices are to some extent intertwined, as shown by the following quote:

> … pupils should be given opportunities to develop knowledge in using digital tools and programming to explore problems and mathematical concepts, make calculations and to present and interpret data (NAE, 2018, p. 55)

The quote reflects several different CT practices. The practice *Using Computational Models to Understand a Concept* from *Modelling and Simulation Practices* is reflected through the connection between programming and understanding mathematical concepts. Digital tools (e.g., Geogebra) and programming are described as enabling tools to formulate and explore problems and concepts in mathematical settings, reflecting the practices *Preparing Problems for Computational Solutions* and *Programming* from the category *Computational Problem Solving Practices*. In addition, the two quotes above in combination could be interpreted to reflect *Choosing Effective Computational Tools* as well as *Assessing Different Approaches/Solutions to a Problem*. The second quote further adds a *Data Practices* perspective reflecting the practices *Analysing Data* and *Visualising Data*.

Modelling and Simulation Practices are reflected in the technology curriculum, for example: “How digital tools can provide support in technical development work, for example when producing drawings and simulations“ (NAE, 2018, p. 299) as well as in the natural science subjects, for example: “Systematic studies and how simulations can be used as support in modelling” (NAE, 2018, p. 181). In the natural science curricula, there is no explicit reference to programming or other computational aspects, besides the broad cross-curricular call for the use of digital tools in all school subjects.

The *Computational Problem Solving Practices* category is primarily represented through programming. There are, however, openings for other practices within this category, as previously shown for Mathematics. A competence aim in the Technology curriculum (year 9) states that: "Pupils can carry out simple work involving technology and design by studying and testing and retesting possible ideas for solutions and also designing developed physical or digital models" (NAE, 2018, p. 300). There is no explicit connection to programming or the use of computational tools, but the quote contains keywords that opens for such an interpretation. For example, "design by studying and testing and retesting possible ideas" points towards both *Assessing Different Approaches/Solutions to a Problem* and *Troubleshooting and Debugging,* while "digital models" could be interpreted to broadly point towards *Choosing Effective Computational Tools*. Having a systematic approach to testing and retesting is a core practice and competence aim for problem solving and content creation, and is represented in the curricula not only in the technology curriculum but also in Crafts and Art.

The category *System Thinking Practices* have similarities with how systems form part of Technology education in Sweden (Hallström & Klasander, 2020; Svensson, 2017) and consequently, the practices subordinated to this category can be identified in the Technology curriculum. The practices *Investigating a Complex System as a Whole, Understanding the Relationships within a System* and *Thinking in Levels* are reflected in the following core content statements (NAE, 2018, p. 299):

- Technical solutions that use electronics and how they can be programmed (Years 7-9)
- How components and subsystems work together in larger systems, such as the production and distribution of electricity (Years 7-9)

Representations of CT in the Norwegian curriculum

General aspects of the Norwegian curriculum related to CT

The new Norwegian curriculum, referred to as LK20, emphasises digital skills in every subject alongside reading, writing, numeracy, and oral skills. The digital skills are categorised as follows (NDET, 2012):

1. Search and process
2. Produce
3. Communicate
4. Digital judgement

These skills signal a view of the learner as primarily a user of digital tools and products, with less emphasis on the technological principles behind these tools, or how students themselves can develop digital tools and products, for example through programming. Programming could potentially have been part of the skill Produce, but this skill is described as "being able to use digital tools, media and resources to compose, reapply, convert and develop different digital elements into finished products, e.g., composite texts" (NDET, 2012, p. 12). This means that programming or other aspects of CT do not form part of how the curriculum describes digital skills on a general level, perhaps with the exception of *Visualising Data* as described in the framework by Weintrop et al. (2016). Nonetheless, programming is presented as part of digital skills specific for several subjects in the curriculum. Programming and other aspects of CT are also highly visible in the descriptions and competence aims of several subjects, as will be presented in the following.

Alongside the curricula, The Norwegian Directorate for Education and Training has developed a framework and resources for CT. The document uses the term algorithmic thinking ("algoritmisk tenkning") with the explicit reference to this term as a translation of CT, despite that CT is used internationally in a much broader sense where algorithmic thinking is considered to be part of CT (Doleck et al., 2017; Labusch, Eickelmann, & Vennemann, 2019). The meaning of algorithmic thinking/CT is in the Norwegian Directorate described as:

> breaking down complex problems into smaller sub-problems that can easier be handled. It includes to organise and analyse information in a logical way and to create procedures (algorithms)

> to arrive at a desired solution. It also involves creating abstractions and models of the real world by omitting unnecessary detail and focusing on what is relevant for the problem at hand and its solution. A solution to a specific problem can often be generalized, in order to be applicable to other problems, and the solution to several sub-problems may be combined in order to solve more complex problems (NDET, 2019, authors' translation).

This description mirrors several practices in the category *Computational Problem Solving Practices* in the framework by Weintrop et al. (2016), but does not capture the other categories and hence not the totality of how this framework describes CT.

The curricula for subjects provide contextual descriptions of what digital competence means. For example, digital competence in terms of being a critical user is highly represented in Social Science, but this subject does not contain programming. In Music and Arts and Crafts, the curriculum states that students are to use programming in creative processes in the subjects. In Arts and Crafts, this is specified in a competence aim for Years 5-7 as "use programming to create interactivity and visual expressions" (NDET, 2019b, p. 7).

With these exceptions, we mainly find programming and other aspects of CT as part of the subjects Natural Science and Mathematics. In the following section, we outline in more detail what opportunities these subjects provide for integrating CT practices. We present how the curriculum for each subject places programming in core elements, digital skills and competence aims. The core elements are subject areas across years where a progression is made through specific competence aims for each year (for mathematics) or groups of years (for other subjects).

Programming in the Norwegian Mathematics curriculum

In the curriculum for Mathematics, CT forms part of the core element Exploration and problem solving. Here, problem solving is described this way:

> Problem solving in mathematics refers to developing a method for solving problems not encountered previously. Computational thinking is important in the process of developing strategies and procedures to solve problems and means breaking a problem down into sub-problems that can be systematically solved. This also includes assessing whether sub-problems would be best solved with or without digital tools. Problem solving also means analysing, rethinking and finding new ways of approaching known and unknown problems, solving them and assessing whether the solutions are valid (NDET, 2019c, p. 2).

This description is much in line with the curriculum's somewhat narrow description of CT (algorithmic thinking) as referred above, which suggests that CT is conceptualized with problem solving in mathematics as a starting point.

In the more specific competence aims in mathematics, essential concepts and skills in programming are explicitly represented from grade 5, in which the pupil is expected to be able to (NDET, 2019c, pp. 9–15):

- create and program algorithms with the use of variables, conditions, and loops (Year 5)
- use variables, loops, conditions, and functions in programming to explore geometric figures and patterns (Year 6)

From grade 7, programming is presented mostly as a tool to use while doing mathematics:

- use programming to explore data in tables and data sets (Year 7)
- simulate outcomes in random trials and calculate the probability that something will happen by using programming
- explore mathematical properties and relationships by means of programming (Year 10)

In Year 8, however, we also find the concept of algorithms explicitly mentioned in connection to programming: "explore how algorithms may be created, tested, and improved by means of programming" (NDET, 2019c, pp. 9–15). Other competence aims (Year 10) can be seen as inviting teachers to use programming, even if it is not stated explicitly, for example: "model situations related to real datasets, present the results and argue for the validity of the models". In sum, programming in mathematics appears as a distinct area of knowledge and skills, notably in Year 5 and 6, while in later school years, it is described mainly as a tool for working with other fields of mathematics.

Programming in the Norwegian Natural Science curriculum

In Natural science, the core element Technology contains explicit reference to programming:

> The pupils shall understand, develop and use technology, including programming and modelling, in their natural-science work. By using and creating technology, the pupils can combine experience and know-how with creative and innovative thinking. The pupils shall understand technological principles and procedures. They shall assess how technology can contribute to solutions, but also create new challenges. Knowledge of and competence in technology are therefore important in a sustainability perspective. Work with the core element technology shall be combined with work linked to the other core elements (NDET, 2019d, p. 3)

We here find the representation of programming as a tool, but also that students should develop technology, including programming. This clearly requires knowledge and skills in programming at a certain level.

In this subject, digital skills are described as:

> using digital tools to explore, register, calculate, visualise, program, model, document and publish data from experiments, fieldwork, and studies by others. Digital skills also refer to using search engines, mastering search strategies, critically assessing sources, and selecting relevant information on natural-science topics. The development of digital skills in natural science progresses from the ability to use simple digital tools to demonstrating increasing independence and judgement in the choice and use of digital sources (NDET, 2019d, p. 5)

The first sentence connects digital tools to real world data in ways that represent all the *Data Practices* in Weintrop et al.'s (2016) framework, and programming is listed as a tool students should be able to use. This is consistent with the competence aims, where we find programming mentioned in the following (NDET, 2019d, pp. 6–11):

- explore, make and program technological systems that consist of parts that work together (Years 5-7)
- use programming to explore natural-science phenomena (Years 8-10)
- explore, understand, and make technological systems that have a transmitter and receiver (Years 8-10)

Programming skills are here needed for a wide range of potentially comprehensive activities, where technological systems have received a prominent place. The curriculum does not state what is meant with a system (for example a boundary to environment and a flow of information, energy or matter as described by e.g., Hallström & Klasander, 2020). Still, it makes it possible to include *System Thinking Practices* from the model for CT (Weintrop et al., 2016), provided that the system is explicitly treated.

The Norwegian curriculum in terms of CT

In terms of the framework of CT (Weintrop et al., 2016), the competence aims in Mathematics that constitute programming knowledge and skills fall into *Computational Problem Solving Practices*. Programming may also form part of the core element Modelling and applications, which the curriculum for Mathematics describes this way:

> A model in mathematics is a description of reality using mathematical language. The pupils shall gain insight into how mathematical models are used to describe everyday lives, working life and society in general. Modelling in mathematics refers to creating such models. It also refers to critically assessing whether the models are valid, what limitations the models may have, assessing them in view of the original situations, and evaluating whether they can be used in other situations. Applications in mathematics refers to giving the pupils insight into how to use mathematics in different situations, in subject-related and other situations (NDET, 2019c, p. 3)

Although not explicitly mentioned in the core element, working with mathematical models with some sophistication often involves programming and it is natural that this is the case for students in later school years.

Digital skills in Mathematics are described as:

> the ability to use graphing tools, spreadsheets, CAS, dynamic geometry software and programming to explore and solve mathematical problems. It also means finding, analysing, processing, and presenting information using digital tools. The development of digital skills refers to choosing and using digital tools to an increasing degree that are well-reasoned as aids for exploring, solving, and presenting mathematical problems (NDET, 2019c, p. 5)

The formulation "choosing and using digital tools for exploring, solving and presenting mathematical problems" reflects the practice *Choosing Effective Computational Tools* in the CT framework (Weintrop et al., 2016) since programming forms one of the tools described. Interpreted this way, the description of digital skills in Mathematics may in fact reflect all the *Computational Problem Solving Practices,* and hence signals high ambitions with regards to this category of CT.

The ambition in the competence aims seem to be lower than in the description of digital skills in Mathematics with regards to *Computational Problem Solving Practices,* since the competence aims cover mainly the practice described as *Programming* in the framework. On the other hand, the competence aims also cover practices within *Data Practices* and to some degree *Modelling and Simulation Practices,* through the competence aims

- find and discuss measures of central tendency and measures of spread (variability) in real datasets (Year 9),

and

- model situations related to real datasets, present the results and argue for the validity of the models (Year 10)

Modelling is represented also in the curriculum for Natural Science, but not in any sense related to programming or other aspects of CT. For example, after Year 10, students are to be able to (NDET, 2019d, pp. 6–11) "use and make models to predict or describe natural-science processes and systems and explain the strengths and limitations of the models" (Years 8-10). As with Data Practices described above, this competence aim is formulated without reference to digital technology or programming but might be realised in ways that include these.

With regards to *Data Practic*es in Weintrop et al.'s framework, we may here also include the science curriculum's competence aims about data collection and data presentation. These are reflected across school years, without reference to the use of programming or computer software, for example (NDET, 2019d, pp. 6–11):

- use tables and figures to structure data, make explanations based on data and present findings (Years 3-4)
- analyse and use collected data to make explanations, discuss the explanations in the light of relevant theory and assess the quality of one's own and others' explorations (Years 8-10)

DISCUSSION AND CONCLUSION

The study reported in this paper has explored how practices related to computational thinking (CT) is represented in the Swedish and Norwegian compulsory school curricula. Both curricula have recently been reshaped in a surge of travelling reform (Ozga & Jones, 2006) focusing to incorporate broad and 'future-proof' 21st century skills, such as problem solving abilities, creativity and digital competence (Baker, 2015; Wahlström & Sundberg, 2015). Since CT is anchored in computation, and the framework used for this study (Weintrop et al., 2016) is based on Mathematics and Science, it is natural that results are dominated by these subjects. However, this did not limit the possibility of detecting CT practices in other subjects as well. In the analysis of curricula, it was found that programming is the most salient CT practice represented in both the Swedish and Norwegian curricula. In both cases, programming is broadly referred to as part of the digital competence and more specifically as a tool, not only for problem solving but for learning content within traditional subjects. This may suggest that programming, or CT more generally, is not fully acknowledged as a domain of knowledge in itself. A further similarity between the Swedish and Norwegian curricula is that programming is linked to mathematics and algorithms. In Sweden, Mathematics is positioned with the responsibility to provide for basic programming skills, but the essence of these skills in terms of practices and concepts is not made clear in the curriculum (Vinnervik, 2022). In the Norwegian curriculum, some attention is directed towards algorithms and certain basic programming concepts in Mathematics, while in other subjects, programming is represented mostly as a tool for learning subject matter.

The realisation of a curriculum in classrooms depends on teachers' interpretations and priorities. Results from the analysis show that the Swedish as well as the Norwegian curriculum provide a potential for incorporating more CT practices than programming in teaching in general education. In the Norwegian curriculum, descriptions of digital skills in the subjects signal higher ambitions for the incorporation of CT than what is reflected in the competence aims, and the latter tend to govern teachers' planning as well as the design of textbooks and other materials for schools. Considering only the competence aims of the subjects separately will lead to teaching that contains some programming in mathematics and its use in other subjects, notably Natural Science. This means that the curriculum, when realised in schools, may be fragmented and mainly reflect a narrow subset of *Computational Problem Solving Practices,* while *Data Practices* and *Modelling and Simulation Practices* are represented without the use of digital tools.

A broader interpretation, on the other hand, provides room for three of the main practices in Weintrop et al.'s (2016) framework: *Data Practices, Modelling and Simulation Practices* and *Computational Problem Solving Practices*. To realise the potential for a broad coverage of CT, teachers will therefore need to read the curriculum across subjects, and combine programming with data practices, modelling and system thinking. An example of how CT can be concretised in this broader sense, is illustrated in a teaching project (see Bungum & Mogstad, 2022) where Grade 9 students design, build and program their own weather station with electronic sensors that collect and transfer weather data by means of microcontrollers. The project is run as part of Science teaching, and data can be analysed as part of the teaching of statistics in Mathematics. This teaching project covers many of the practices in the framework by Weintrop et al. (2016), and meets a range of competence aims in Mathematics and Science, possibly also Arts and Crafts. However, the project is time consuming and requires an awareness for cross-curricular and cross-subject competences beside the subject specific competence aims.

Consequently, considerable resources must be put into teacher professional development if CT is to become a natural and constructive part of the curriculum realised in classrooms. Recent research has shown that teachers face several challenges during implementation of programming in existing subjects (Stigberg & Stigberg, 2020; Vinnervik, 2020). These challenges are, for example, related to resources (e.g., time, teaching materials), their professional understanding of programming as a domain of knowledge, and professional development opportunities. Further, teachers are uncertain how to use programming to amplify the understanding of other subject content (Kilhamn, Rolandsson, & Bråting, 2021). If such challenges are left unsolved, the result may be a reform that is not implemented as intended (Larke, 2019).

The curricula examined in the present study can be seen as initial attempts to incorporate CT practices into a school curriculum. The study has shown similarities in how programming and CT are represented in Swedish and Norwegian curricula, even though the division into separate subjects is different. Working with cross-curricular competencies, such as CT, requires that schools and teachers agree on a unified effort and find deliberate and functional strategies for cross-subject collaboration (Fullan, 2007). The transfer between contexts and subjects is not a straightforward process. Therefore, the professional leeway left for teachers in both Sweden and Norway may work best within traditional subject boundaries, for which teachers are prepared.

Future curriculum revisions could represent a more elaborate and explicit approach to practices where computational aspects are applicable, as there are opportunities to strengthen the representation of CT practices in both the Swedish and Norwegian curricula. For example, a more deliberate combination of programming with *Data Practices* in terms of scientific processes rather than content, would represent a wider range of the practices that form part of CT as represented in the framework by Weintrop et al. (2016). Another area of development concerns the clarity of learning progressions between grades: What should the children know about programming (or CT) and when? Further, a practice that could serve the purpose of programming well is to clarify the relationship between programming, systematic testing and retesting, and the practice of *Troubleshooting and Debugging*. This would be in line with how debugging is emphasised as a very valuable aspect of learning programming in literature (McCauley et al., 2008). The curriculum concedes that improving algorithms is part of the programming process but do not explicitly stress the place and value of a systematic approach to debugging in benefit of learning the fundamentals of programming. Another area of improvement concerns the Swedish curriculum, which declares that students should first meet block-based programming environments, and eventually text-based programming environments. There is, however, no information about how such a transition from block-based to text-based programming environments may contribute to both deepen and broaden the children's understanding for programming principles and concepts. The implicit assumption seems to be that increased syntactic complexity leads to increased understanding (Vinnervik, 2022). Research (Weintrop & Wilensky, 2019) expresses concern and suggests that choosing appropriate programming environments and programming languages can be challenging and have implications for children's experience and understanding of programming.

To conclude, the most salient computational thinking practice represented in both the Swedish and Norwegian curricula is programming. It is primarily recognized as a method and vehicle for learning other subject content and not as a knowledge domain in its own right. Both curricula leave leeway for teachers to implement a broader approach to computational thinking. However, this would require much time and considerable teacher competence, and the requirements for such an effort seems to be under-communicated in the curricula, leaving schools and teachers with major challenges.

REFERENCES

Baker, D. P. (2015). A note on knowledge in the schooled society: Towards an end to the crisis in curriculum theory. *Journal of Curriculum Studies, 47*(6), 763–772. doi: 10.1080/00220272.2015.1088069

Barefoot Computing. (n.d.). Computational Thinking Concepts and Approaches. Retrieved 6 December 2021, from https://www.barefootcomputing.org/concept-approaches/computational-thinking-concepts-and-approaches

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

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? *Acm Inroads*, *2*(1), 48–54. doi: 10.1145/1929887.1929905

BBC. (n.d.). Introduction to Computational Thinking. Retrieved 6 December 2021, from https://www.bbc.co.uk/bitesize/guides/zp92mp3/revision/1

Bocconi, S., et al. (2016, June). Developing computational thinking: Approaches and orientations in K-12 education. In *EdMedia+ Innovate Learning* (pp. 13-18). Association for the Advancement of Computing in Education (AACE)

Bocconi, S., Chioccariello, A., & Earp, J. (2018). The Nordic approach to introducing Computational Thinking and programming in compulsory education. *Report prepared for the Nordic@ BETT2018 Steering Group*, *42*.

Bungum, B., & Mogstad, E. (2022) Building and programming a weather station: Teachers' views on values and challenges in a comprehensive STEM project, Research in Science & Technological Education. doi: 10.1080/02635143.2022.2103108

Børne- og undervisningsministeriet. (2021). Forsøg med teknologiforståelse i folkeskolens obligatoriske undervisning. Retrieved 6 December 2021, from https://emu.dk/grundskole/forskning-og-viden/paedagogisk-it/evaluering-af-forsoeg-med-teknologiforstaaelse?b=t5-t34-t2868

Børne- og Undervisningsministeriet. (2019a). Faghæfte for Matematik.

Børne- og Undervisningsministeriet. (2019b). Faghæfte för Natur/teknologi.

Brennan, K., & Resnick, M. (2012, April). 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* (Vol. 1, p. 25).

Carretero, S., Vuorikari, R., & Punie, Y. (2017). The digital competence framework for citizens. *Publications Office of the European Union*.

Crawford, B. A. (2014). From inquiry to scientific practices in the science classroom. In *Handbook of research on science education, volume II* (pp. 529-556). Routledge.

Crick, T. (2017). Computing education: An overview of research in the field. *London: Royal Society*.

Doleck, T., Bazelais, P., Lemay, D. J., Saxena, A., & Basnet, R. B. (2017). Algorithmic thinking, cooperativity, creativity, critical thinking, and problem solving: exploring the relationship between computational thinking skills and academic performance. *Journal of Computers in Education*, *4*(4), 355-369. doi: 10.1007/s40692-017-0090-9

European Education and Culture Executive Agency, Eurydice (2019). *Digital education at school in Europe*, Publications Office. doi: 10.2797/763

Fullan, M. (2007). The New Meaning of Educational Change (4th ed.). New York: Teachers College Press.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational researcher*, *42*(1), 38–43. doi: 10.3102/0013189X12463051

Guzdial, M. (2020, January 13). Computing education lessons learned from the 2010's: What I got wrong. *Computing Education Research Blog*. https://computinged.wordpress.com/2020/01/13/computing-education-lessons-learned-from-the-2010s-what-i-got-wrong/

Guzdial, M., Kay, A., Norris, C., & Soloway, E. (2019). Computational thinking should just be good thinking. *Communications of the ACM, 62*(11), 28–30. doi: 10.1145/3363181

Hallström, J., & Klasander, C. (2020). Making the invisible visible: Pedagogies related to teaching and learning about technological systems. In *Pedagogy for technology education in secondary schools* (pp. 65-82). Springer, Cham. doi: 10.1007/978-3-030-41548-8_4

Kilhamn, C., Rolandsson, L., & Bråting, K. (2021). Programmering i svensk skolmatematik: Programming in Swedish school mathematics. *LUMAT: International Journal on Math, Science and Technology Education*, *9*(1), 283-312. doi: 10.31129/lumat.9.2.1457

Krippendorff, K. (2018). *Content analysis: An introduction to its methodology*. Sage publications.

Labusch, A., Eickelmann, B., & Vennemann, M. (2019). Computational thinking processes and their congruence with problem-solving and information processing. In *Computational thinking education* (pp. 65-78). Springer, Singapore.

Larke, L. R. (2019). Agentic neglect: Teachers as gatekeepers of England's national computing curriculum. *British Journal of Educational Technology*, *50*(3), 1137-1150. doi: 10.1111/bjet.12744

McCauley, R., et al. (2008). Debugging: a review of the literature from an educational perspective. *Computer Science Education*, *18*(2), 67-92. doi: 10.1080/08993400802114581

Ministry of Education, Science and Culture. (2014). The Icelandic National Curriculum Guide for Compulsory Schools–With Subject Areas.

Mølstad, C. E., & Karseth, B. (2016). National curricula in Norway and Finland: The role of learning outcomes. *European Educational Research Journal*, *15*(3), 329–344. doi: 10.1177/1474904116639311

NAE. (2017a). Få syn på digitaliseringen på grundskolenivå – Ett kommentarmaterial till läroplanerna för förskoleklass, fritidshem och grundskoleutbildning. Stockholm: Skolverket.

NAE. (2017b). Kommentarmaterial till kursplanen i matematik. Stockholm: Skolverket.

NAE. (2017c). Kommentarmaterial till kursplanen i teknik. Stockholm: Skolverket.

NAE. (2018). Curriculum for the compulsory school, preschool class and school-age educare. Stockholm: National Agency for Education.

NDET. (2012). Framework for basic skills. Oslo: Norwegian Directorate for Education and Training. Retrieved 6 December 2021, from https://www.udir.no/in-english/Framework-for-Basic-Skills/

NDET. (2019a). Algoritmisk tenkning. Oslo: Norwegian Directorate for Education and Training. Retrieved 6 December 2021, from https://www.udir.no/kvalitet-og-kompetanse/profesjonsfaglig-digital-kompetanse/algoritmisk-tenkning/

NDET. (2019b). Curriculum for Arts and Crafts. Oslo: Norwegian Directorate for Education and Training. Retrieved 6 December 2021, from https://data.udir.no/kl06/v201906/laereplaner-lk20/KHV01-02.pdf?lang=eng

NDET. (2019c). Curriculum for Mathematics Year 1-10. Oslo: Norwegian Directorate for Education and Training. Retrieved 6 December 2021, from https://data.udir.no/kl06/v201906/laereplaner-lk20/MAT01-05.pdf?lang=eng

NDET. (2019d). Curriculum for Natural Science. Oslo: Norwegian Directorate for Education and Training. Retrieved 6 December 2021, from https://data.udir.no/kl06/v201906/laereplaner-lk20/NAT01-04.pdf?lang=eng

Ozga, J., & Jones, R. (2006). Travelling and embedded policy: The case of knowledge transfer. *Journal of Education Policy, 21*(1), 1–17. doi: 10.1080/02680930500391462

Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas. *New York: Basic Books. 10*. 1095592

Passey, D. (2017). Computer science (CS) in the compulsory education curriculum: Implications for future research. *Education and Information Technologies, 22*(2), 421–443. doi: 10.1007/s10639-016-9475-z

Saqr, M., Ng, K., Oyelere, S. sunday, & Tedre, M. (2021). People, Ideas, Milestones: A Scientometric Study of Computational Thinking. *ACM Transactions on Computing Education, 21*(3), 1–17. doi: 10.1145/3445984

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, 22, 142–158. doi: 10.1016/j.edurev.2017.09.003

Stigberg, H., & Stigberg, S. (2020). Teaching programming and mathematics in practice: A case study from a Swedish primary school. *Policy Futures in Education, 18*(4), 483–496. doi: 10.1177/1478210319894785

Svensson, M. (2017). Learning About Systems. In M. J. de Vries (Ed.), *Handbook of Technology Education* (pp. 1–16). Cham: Springer International Publishing. doi: 10.1007/978-3-319-38889-2_34-1

Tedre, M., & Denning, P. J. (2016, November). The long quest for computational thinking. In *Proceedings of the 16th Koli Calling international conference on computing education research* (pp. 120-129). doi: 10.1145/2999541.2999542

Vinnervik, P. (2020). Implementing programming in school mathematics and technology: teachers' intrinsic and extrinsic challenges. *International Journal of Technology and Design Education.* doi: 10.1007/s10798-020-09602-0

Vinnervik, P. (2022). An in-depth analysis of programming in the Swedish school curriculum - rationale, knowledge content and teacher guidance. *Journal of Computers in Education*, 1-35. doi: 10.1007/s40692-022-00230-2

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

Voogt, J., & Roblin, N. P. (2012). A comparative analysis of international frameworks for 21st century competences: Implications for national curriculum policies. *Journal of Curriculum Studies, 44*(3), 299–321. doi: 10.1080/00220272.2012.668938

Wahlström, N., & Sundberg, D. (2015). *En teoribaserad utvärdering av läroplanen Lgr 11*. Institutet för arbetsmarknads-och utbildningspolitisk utvärdering (IFAU).

Weintrop, D., & Wilensky, U. (2019). Transitioning from introductory block-based and text-based environments to professional programming languages in high school computer science classrooms. *Computers and Education, 142*(July), 103646. doi: 10.1016/j.compedu.2019.103646

Weintrop, D., et al. (2016). Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Technology, 25*(1), 127–147.

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