NJCIE Nordic Journal of Comparative and International Education

ISSN: 2535-4051 Vol 8, No 4 (2024) https://doi.org/10.7577/njcie.5744

Article

Sign of the Times: The Framing of Computational Thinking in Danish, Finnish, and Norwegian Curricula

Katarina Pajchel
Oslo Metropolitan University
Email: kapaj@oslomet.no

Louise Mifsud
Oslo Metropolitan University
Email: louise.mifsud@oslomet.no

Thomas Frågåt
Inland Norway University of Applied Sciences
Email: thomas.fragat@inn.no

Mads M. Rehder
University College Copenhagen
Email: mamr@kp.dk

Kalle Juuti
University of Helsinki
Email: kalle.juuti@helsinki.fi

Yurdagül Bogar
Hakkari University
Email: yurdagul.bogar@helsinki.fi/yurdagul-bogar@hotmail.com

Jari Lavonen
University of Helsinki/University of Johannesburg
Email: jari.lavonen@helsinki.fi

Vibeke Schrøder
University College Copenhagen
Email: vs4@kp.dk

Siv G. Aalbergsjø
Oslo Metropolitan University
Email: siguaa@oslomet.no

André Rognes
Oslo Metropolitan University
Email: andro@oslomet.no

Abstract

This paper discusses the significance of school curricula in reflecting societal priorities and needs, focusing on the incorporation of computational thinking (CT) in Nordic national curricula. Our point of departure is that the preparedness of future generations for a digitally driven society can be determined by analysing how CT is either explicitly or implicitly framed in school curricula. Accordingly, this study examined the school curricula of Denmark, Finland, and Norway in terms of their similarities and differences in how they framed CT, as these countries have different approaches to the inclusion of CT. A framework for analysis that was grounded in influential works on CT in education was developed, focusing on problem-solving, algorithmic and transversal practices. National-level curricula were examined using a content analysis. Despite the differences in the approaches used in these countries, our findings indicate similarities across all three curricula, with an emphasis on how CT was framed.

Keywords: Curricula, computational thinking, transversal practice, algorithm practice, problem-solving practice

Introduction

School curricula are specifications of what is to be formally taught or learned and, consequently, are a sign of what is viewed as relevant knowledge for the current society and future adult citizens (Ross, 2003). As such, it is important to analyse current curricula as a reflection of a country's societal priorities and needs (Kácovský et al., 2022, p. 384). A national-level curriculum is an official document that describes an intentional instructional agenda at school, with close links to the needs of society (Autio, 2013; Cuban, 1992). According to Cuban (1992), a national-level curriculum organises the body of knowledge and skills that teachers should teach, and students should learn. In this respect, curricula can be viewed as

knowledge and skills that are perceived as valuable for society.

Computational thinking (CT) has been included in school curricula in several countries, including the Nordic countries (Bocconi et al., 2022). Arguments for the inclusion of CT have focused on the needs of a highly digitalised society where understanding technology is presented as a requirement for a changing labour market (Iversen et al., 2018; McGarr & Engen, 2024) to enable empowered decision-making (Iversen et al., 2018) and as a competence that future generations must develop in light of a digitalised society (Voogt & Roblin, 2012; Zhang & Nouri, 2019). In a digitalised society, where computing is part of daily lives, CT is highlighted as an important competence, giving a 'basic understanding of what algorithms are and how automation plays a central part in our everyday lives' (Hansen et al., 2024, p. 234). Furthermore, another argument for the inclusion of CT in education is that subjects such as science and mathematics must meet the expectations of the current society (Weintrop et al., 2016) and enhance science through data collection and analysis and modelling (Barr & Stephenson, 2011). These arguments are in line with Papert's (1980) original vision of computers as 'objects to think with' with a potential to become tools for learning and in giving children a 'sense of empowerment and achievement' (p.21).

The aim of this study was to elucidate and compare how CT is framed in Danish, Finnish, and Norwegian national-level curriculum documents for basic education. The Swedish curriculum has already been analysed in depth (Vinnervik, 2023), and the Swedish and Norwegian curricula have been compared previously (Vinnervik & Bungum, 2022). However, it is interesting to compare Denmark, Finland, and Norway, as their approaches to CT are varied: being integrated into subjects (Norway); suggested as part of a newly constructed subject (Denmark); both cross-curricular and subject-specific (Finland) (Andersen et al., 2023; Bocconi et al., 2022). In addition, these countries are also at different stages of introducing and integrating CT into their curricula, with Finland having introduced CT in their 2014 curriculum (Finnish National Board of Education [FNBE], 2014), Norway having introduced CT in 2020 (Norwegian Directorate for Education and Training [NDET], 2020a-d), and Denmark having piloted different approaches (Ministry of Children and Education [MoCE], 2023) planning implementation in the forthcoming curriculum (MoCE, 2024). Finally, all the authors were familiar with the countries and their national-level curricula. Understanding the conceptualisations of CT that are emphasised, either explicitly or implicitly, in the national-level curricula in these countries is an indication of the desired level of preparedness for future generations to actively engage in a digitally driven society. To examine these countries national-level curricula, we raised the following research questions:

> (1) How is CT framed in Danish, Finnish, and Norwegian national-level curricula?
>
> (2) What are the similarities and differences in how CT is framed in Danish, Finnish, and Norwegian national-level curricula?

Computational Thinking

While it has been proposed that CT is an increasingly important skill for future citizens (Voogt & Roblin 2012; Zhang & Nouri, 2019), no unified understanding of CT exists (Grover & Pea, 2013; Haseski et al., 2018). For example, in their review of CT definitions, Haseski et al. (2018) identified no less than 59 definitions of CT in 99 articles on CT, reflecting the broadness of CT as a concept. Definitions ranged from being able to think 'like a computer scientist' (Wing, 2006, p. 34) to involving problem solving, where problems are broken down to be solved using a computer (Shute et al., 2017), and to being a social practice (Kafai, 2016). As such, CT is described as a transversal competence. Care et al. (2018) described transversal competences as skills, values, and attitudes required for learners' holistic development and for learners to become capable of adapting to change. Transversal or 'twenty-first century' competences focus on critical and creative thinking, enquiry, and collaboration competences, and an understanding of core ideas or concepts (Binkley et al., 2012; Voogt & Roblin, 2012).

To create a basis for our understanding of CT, we took the most cited articles on CT and education as our point of departure, cutting off where a wider gap to the less cited articles was identified. This resulted in six articles: Brennan and Resnick (2012), Grover and Pea (2013), Lye and Koh (2014), Bers et al. (2014), Weintrop et al. (2016), and Shute et al. (2017). The articles were screened for how they defined or presented CT. Two articles were excluded: that by Lye and Koh (2014), as they applied the framework presented by Brennan and Resnick (2012), and that by Bers et al. (2014), which focused on preschool and, as such, were outside the scope of this paper, as we are concerned with curricula intended for compulsory schooling. The remaining four articles provided frameworks that have been applied in several empirical studies of CT in education, thereby representing a broad scope of definitions that advocate suitability for analysing the framing of CT in the curricula of the countries included in this study. The included articles are presented in chronological order of publication.

Brennan and Resnick's (2012) definition of CT focused on three dimensions: *computational concepts*, *computational practices*, and *computational perspectives*. Computational *concepts* include sequences, loops, parallelism, events, conditionals, operators, and data. To a certain extent, these concepts can be related to the activity of creating an algorithm that they claim can be transferred to other programming or non-programming contexts. For computational *practices*, Brennan and Resnick pointed out that these focus on 'the process of thinking and learning, moving beyond what you are learning to how you are learning' (p. 6). Within this dimension, the authors highlighted both problem-solving activities such as testing and debugging, incremental and iterative practices, reusing, and remixing, and abstracting and modularising. In their computational *perspectives*, Brennan and Resnick focused on expressing, connecting, and questioning.

Grover and Pea (2013) suggested that CT is comprised of: abstractions and pattern generalisation (including models and simulations); systematic information processing; symbol systems and representations; algorithmic notions of control flow; structured problem decomposition (modularising); iterative, recursive, and parallel thinking; conditional logic; efficiency and performance constraints; and debugging and systematic error detection. Grover and Pea (2013, p. 40) viewed programming as a crucial 'tool in supporting cognitive tasks involved in CT' and suggested that programming is also 'a demonstration of computational competencies'.

Weintrop et al. (2016) presented a four-category taxonomy for CT in science and mathematics classrooms. Their approach to defining CT takes the form of a taxonomy of practices that focuses on the *application* of CT to mathematics and science and on data, modelling and simulation, computational problem solving, and systems thinking practices, which are composed of five to seven interrelated sub-categories. *Data practices* include data collection, creation, manipulation, visualisation, and analysis. *Modelling and simulation practices* focus on conceptual understanding, testing solutions, model assessment, and model construction. C*omputational problem-solving practices* include solution preparation, programming, tool selection, solution evaluation and development, abstraction, and debugging, while *systems thinking practices* focus on system investigation, understanding relationships, multi-layered thinking, and systems management.

Shute et al. (2017) defined CT as 'the conceptual foundation required to solve problems effectively and efficiently (i.e., algorithmically, with or without the assistance of computers) with solutions that are reusable in different contexts' (p. 143). Their model focused on approaching problems systematically. As such, problem solving is an underlying principle in their CT knowledge and skills model, which was built on six facets: decomposition, abstraction, algorithms, debugging, iteration, and generalisation. Decomposition and abstraction are related to understanding a problem, partly or in whole, in addition to understanding the relationship between the extraction of the principles of complex systems and functions. They also include elements such as identifying patterns and rules underlying the data and information structure, while data collection refers to collecting and analysing relevant information. Both debugging and iteration are, to a certain extent, linked to problem solving with respect to detecting and fixing errors, and repeating the process to refine solutions, respectively.

Although these models are widely used, they have also been subject to criticism. Brennan and Resnick's (2012) framework has been criticised for being limited to the tool Scratch and for being hierarchical (Shute et al., 2017). Weintrop et al.'s (2016) model was developed for computational practice in STEM subjects and, as such, is limited its scope (Shute et al., 2017). Shute et al.'s (2017) model linked generalisation to the notion of transfer to other domains. However, the notion of transfer is largely unverified (Lodi & Martini, 2019; Vinnervik, 2023).

Another issue that has been discussed is the ambiguous relationship between programming and CT. Papert (1980) used the term programming, Grover and Pea (2013), Lye and Koh (2014), and Shute et al. (2017) highlighted programming as a key inherent competence in promoting and supporting CT or even synonymously with CT (Lodi & Martini, 2021). This article does not aim to solve this issue but rather to utilise this broad range of definitions to examine and debate how CT is framed in curricula. Drawing on the four articles, we identified three types of CT *practices*: *problem-solving*, *algorithmic*, and *transversal practices* (see Table 1). In using the term practices, we highlight the active aspect of CT in national curricula.

Table 1. Overview of the terms used in presenting CT

	Problem-solving practices	**Algorithmic practices**	**Transversal practices**
Brennan and Resnick (2012)	Being incremental and iterative, testing and debugging, and abstracting and modularising, and reusing and remixing	Sequences, loops, parallelism, events, conditionals, operators, and data	Expressing, connecting, and questioning
Grover and Pea (2013)	Abstractions and pattern generalisations (including models and simulations); systematic information processing; structured problem decomposition (modularising); iterative and recursive thinking; debugging; systematic error detection	Symbol systems and representations, algorithmic notions of control flow, parallel thinking, conditional logic, efficiency, and performance constraints	
Weintrop et al. (2016)	Analysing data, using conceptual models to understand a concept and find and test solutions, assessing and designing computational models, constructing computational models, preparing problems for computational solutions, programming, choosing effective computational tools, assessing different approaches/solutions to a problem, developing modular computational solutions, troubleshooting, and debugging, and creating computational abstractions	Collecting, creating, and manipulating data, and programming	Visualising data, investigating complex systems as a whole, understanding the relationships within the system, thinking in levels, communication, information about a system, defining systems, and managing complexity
Shute et al.	Decomposition, abstraction,	Algorithms	Generalisation

(2017)	debugging, and iteration

CT practices

Problem-solving practices can range from fundamental error detection to decomposition and abstraction (Shute et al., 2017; Weintrop et al., 2016) and to designing solutions and working with information or data in a way that a computer or other person can help solve the problem (Brennan & Resnick, 2012; Grover & Pea, 2013; Shute et al., 2017; Weintrop et al., 2016). This approach to CT is not new and originates from Papert's (1980) constructionist approaches. Problem-solving practices embrace breaking down a problem into smaller ones (Grover & Pea, 2013; Shute et al., 2017) and iterative testing (Brennan & Resnick, 2012; Shute et al., 2017; Weintrop et al., 2016), such as abstraction, which involves understanding relevant data, recognising patterns within the data, and reorganising the data meaningfully for solving the problem (Brennan & Resnick, 2012; Grover & Pea, 2013; Shute et al., 2017; Weintrop et al., 2016). While all four frameworks include data practices, Weintrop et al. (2016), who focus on STEM, connect it also with modelling, which is a crucial problem-solving practice in these subjects. Programming is both an algorithmic and a problem-solving practice. Shute et al. (2017) argued that programming 'requires analysis of the problem' (p. 149) while Weintrop et al. (2016) places it under computational problem-solving and relates it to modelling practices. Referring to this broad understanding of programming, we choose to include programming as part of the problem-solving practises.

Algorithmic practices can range from reading or following a sequence or series of instructions to creating algorithms in a specific situation. Brennan and Resnick (2012) highlighted creating algorithms in programming and non-programming contexts through concepts such as sequences, loops, and conditionals (pp. 2–6). Algorithmic practices encompass the basic components needed to create, execute, or understand an algorithm in a digital or non-digital context. Algorithmic practices involve reading, following, or writing codes or symbol systems (Grover & Pea, 2013). *Coding* is a term often used in connection with CT referring to defining or expressing algorithms in some language. In our framework we choose not to use the term coding. However, we fully acknowledge this more technical aspect as part of the algorithmic practices. It is often regarded as a subset of programming but demands less intellectual effort than programming (Pears et al., 2021), being limited to skills related to encoding 'instructions in such a way that a computer can execute them' (Weintrop et al., 2016, p. 139). However, algorithms are not created in a contextless vacuum. Shute et al. (2017) emphasised that algorithms involve the design of 'logical and ordered instructions for rendering a solution to a problem' (p. 153), which requires that 'encoding' is strongly connected to problem-solving practices. To a certain extent, algorithmic practices represent an instrumental understanding of how algorithms work and how to create step-by-step instructions that can

be followed by another person or executed by a computer.

Transversal practices are the competences often described as twenty-first-century or generic competences (Binkley et al., 2012; Care et al., 2018; Voogt & Roblin, 2012). Within the context of CT, all four articles reviewed in this paper highlighted aspects that can be described as transversal practices. Brennan and Resnick (2012) emphasised the collaborative aspect of creating with others and communication in terms of creating not only for oneself but also for others. Weintrop et al. (2016) focused on the ability to visualise data or communicate information about a system. These transversal practices support problem-solving practices as they also involve moving the 'part' and understanding how systems function 'as a whole' (Weintrop et al., 2016, p. 141) or in other domains (Shute et al., 2017) and solving 'real-world problems' (Brennan & Resnick, 2012; Weintrop et al., 2016). Transversal practices also include generating ideas related to creativity, communication, collaboration, visualisation and relating these to the world around.

Methodology

To examine and compare how CT is framed in Danish, Finnish, and Norwegian national-level curricula, we analysed the Danish pilot-curriculum from the Ministry of Education [MoE] (2018b), the Finnish curriculum of 2014 (FNBE, 2014), and the Norwegian curriculum of 2020 (NDET, 2020a, 2020b, 2020c, 2020d[1], Table 2).

Table 2. Overview of the analysed curriculum documents

Country	Document	Publisher
Denmark	Curriculum for pilot program Technology Comprehension (Læseplan) year 1–9	MoE (2018a)
Denmark	Curriculum for pilot program Technology Comprehension (Måloversigt) year 1–9	MoE (2018b)
Finland	National Core Curriculum for Basic Education 2014	FNBE (2014)
Norway	Curriculum for Art and crafts (KHV0102)	NDET (2020a)
Norway	Curriculum for Mathematics year 1–10 (MAT0105)	NDET (2020b)
Norway	Curriculum for Music (MUS0102)	NDET (2020c)
Norway	Curriculum for Natural science (NAT0104)	NDET (2020d)

In our curriculum analysis, we used the official translations of the Norwegian curricula. In the Finnish case, we used an unofficial translation provided by the National Educational Office in Finland, while in the Danish

[1] Elective subjects in the Norwegian curriculum were excluded.

case, the curricula were translated by the authors. It is important to highlight the challenges involved in comparing national-level curricula. One challenge was the variety of CT terminology used and determining whether the term was explicitly mentioned in the curriculum. For example, in Norway, 'algoritmisk tenkning' (algorithmic thinking) is the official translation of computational thinking (NDET, 2020b) and it is only used in one place in the mathematics curriculum. The notion 'programmering' (programming) is used in all other contexts. In Finland, 'algoritminen ajattelu' (algorithmic thinking) and 'ohjelmointi' (programming) are used. In Denmark, 'computationel tankegang' (computational thinking) is used to denote a competence area along with 'programmering' (programming). In addition, the curricula in the three countries are organised differently.

The Danish pilot curriculum for Technology Comprehension (MoE, 2018a; 2018b) defines it as a separate subject divided into four competence areas. We focused on two areas: CT, including all its subcategories, and Technological Capability, including its subcategory Programming. These competence areas were selected as CT practices were presented explicitly in terms of formulations and implicitly in terms of progression and opportunities for CT.

The Finnish National Core Curriculum for Basic Education (FNBE, 2014) is written as guidelines for school providers (typically a city) and has two sections. The general part of the curriculum describes the value base like conception of learning, working methods and descriptions for transversal competences, which are to be integrated in all subjects. In this paper, the transversal competences T1 'Thinking and learning to learn' and T5 'ICT competence', were selected for the analysis, as this is where CT is explicitly or implicitly present. The second part of the curriculum is subject specific, describing the objectives of the subject, and includes teaching methods and guidelines. This study analysed the 'objectives for instruction' and 'core content' of mathematics, biology, chemistry, and physics, as well as music, visual arts, and crafts, where T1 and T5 are indicated together with the subject-specific objectives.

In the Norwegian curriculum, CT is included in four subjects: mathematics, natural science, music, and arts and crafts (NDET 2020a, 2020b, 2020c, 2020d). Norway's 2020 curriculum comprises the core curriculum, which outlines the values and principles for education, and the disciplinary sections. The latter are divided into several sub-sections: relevance and central values, core elements, five basic skills, interdisciplinary topics, competence aims, and assessment for the various school years. In the analysis, we focused on the core elements, basic digital skills, and competence aims for the four subjects where CT and programming were included.

In interpreting the curriculum, we draw on content analysis (Hsieh & Shannon, 2005; Kleinheksel et al., 2020) to identify, compare, and determine how CT was framed in curricula. Content analysis involves the

examination of textual data to better understand a phenomenon by establishing structure through systematic processes of interpretation (Kleinheksel et al., 2020). Content analysis can be viewed as sorting text into groups of 'related categories to identify similarities and differences, patterns, and associations, both on the surface and implied within' (Kleinheksel et al., 2020, p. 7113). In this study, categories were developed based on the framework described in the previous section: 1) problem-solving, 2) algorithmic, and 3) transversal practices. Units of analysis were at meaningful phrase levels, which described learning outcomes. These were then interpreted and categorised (see Tables 3-8 in Appendix).

To analyse the curricula, a five-stage process was undertaken:

1. The curricula were skimmed for phrases that explicitly mentioned CT.
2. The documents were searched for CT-related phrases.
3. The documents were searched for terms that are implicitly related to CT (see examples below).
4. Categorising and analysing in depth the selected curriculum content based on the framework developed in Table 1 (presented in Table 3 of the Appendix).
5. Illustrative curriculum content for presentation was selected.

In categorising the selected curriculum content as problem-solving, algorithmic, or transversal practices, we focused on CT-phrases as the primary indicators. For example, *create sequences* were categorised as algorithmic practices, as the focus was on the making of an algorithm; *explore* or *design* indicated using programming or CT to solve a problem, while *visualise* indicated an element of communication and was considered as a transversal practice. Furthermore, we searched the documents for explicit CT terms such as *computational thinking*, *programming*, *loops*, *conditionals*, and *data*, and terms that were implicitly related to CT, such as *step-by-step* and *exploration* (drawing on the framework described). Depending on the vocabulary of each curriculum, the presence of the words was not enough, they needed also a connection to thinking skills and information and communication technology competence to be interpreted as opportunities to learn computational thinking. Once the guidelines or outcomes were selected, they were categorised according to the framework presented in Table 1 (see Tables 3-8) in the Appendix for illustrative examples of the analysis). A minimum of three researchers categorised these independently and then compared and discussed their categorisation.

CT in National-level Curricula

In the following, we describe the characteristic descriptions and framings of CT in each of the Danish, Finnish, and Norwegian curricula. To understand and compare how CT is framed in national-level curricula, explicitly or implicitly, we first provided an overall description of the curricula and how CT was included before describing findings related to problem-solving, algorithmic, and transversal practices for each of the three countries.

CT in the Danish curriculum

CT was developed as a part of a new subject, Technology Comprehension, that was piloted between 2018 and 2021 in 46 Danish primary and lower secondary schools (MoE, 2018a). Technology Comprehension was tested both as a separate subject and integrated into existing subjects such as Danish, arts, physics and chemistry, social studies, mathematics, nature and technology, and arts and crafts (MoCE, 2023). The pilot programme was aimed at fostering insights, skills, and capacities necessary for children to engage critically and constructively with digital technologies through the development and testing of a new curriculum.

According to the political agreement from 2024 on the Danish public school, Technology Comprehension will be integrated both into selected existing subjects (grades 1 to 9) as well as offered as a new fifth two-year practical/musical elective subject in secondary education (MoCE, 2024). Since these new curricula are not yet developed, the analysis relies on the pilot curricula for technology comprehension from 2018.

The Technology Comprehension curriculum tested in the pilot programme was the most coherent and developed Danish attempt at creating a curriculum that included CT for mandatory curriculum in primary and lower secondary schools in Denmark. According to the Ministry of Education (MoE, 2018a; 2018b), the aims of Technology Comprehension are to strengthen students' prerequisites for understanding, creating, and acting meaningfully in a society where digital technologies and artefacts increasingly serve as catalysts for change. In the subject of Technology Comprehension, programming falls under the competence area 'Technological Capability' as the constructive and creative strand in the curriculum, while CT is defined as a separate competence area covering Data, Algorithms, Structuring and Modelling (MoE, 2018b).

Problem-solving practices

In general, in the Technology Comprehension curriculum, CT appeared to be related to problem-solving practices focusing on analysis. The general description of CT focused on the analysis, modelling, and structuring of data and data processes. The overarching description of the subject suggested an emphasis on problem-solving practices, as students are expected to acquire skills for analysing, designing, constructing, modifying, and evaluating digital artefacts for the understanding and resolution of complex problems.

The aims for CT after year 6 state that the student can follow and apply CT in dealing with concrete issues or problems, where problem-solving practices are explicitly mentioned. However, in year 9, the problem-solving practices were more implicitly embedded in the goals, also indicating a progression where the students are expected to *use* CT to reflect on and apply CT to issues from the surrounding world. This illustrates how these goals are connected to problem-solving practices as a primary strand throughout the CT curriculum, both explicitly and implicitly. Similar elements can be identified in the aims for programming

after year 9, where the programming language is to be used for systematic modification and construction of programs based on a problem specification. Programming also covers systematic testing and debugging of their own and others' programs both after years 6 and 9. Here, the link between problem-solving practices and algorithmic practices is explicated as key factors supporting each other.

Algorithmic practices

Algorithmic practices in the Danish curriculum were apparent in the competence area Technology Capability (subcategory Programming), where both block-based and text-based languages were emphasized. In the competence CT, sub-competence Structuring, the curriculum explicitly highlights using sequences, conditionals, selection and repetitions. Interestingly algorithmic practices are explicitly described after year 3 and 6, while in year 9 there is more emphasis on a problem-solving approach to algorithmic practices.

We see that there is a clear progression from block- to text-based programming languages. For example, after year 3, students are expected to follow and modify simple programs in at least one block-based language. After year 6, students are expected not only to follow and modify but also to construct block-based programs. Then, after year 9, the aims specify skills in text-based programming and constructing programmes based on specifications.

Transversal practices

Transversal practices were identified in the goals described for both Programming and CT, and therefore transcend different areas of the subject. In the aims for CT after year 3, CT is connected to describing familiar and delimited phenomena in everyday life, which can be considered as a communicative and creative way of using CT. This continues in the aims after year 9, where students can reflect on and apply CT to issues from the external world, which again points to the creative and communicative elements of CT and therefore links to transversal practices.

The organisation of CT in Technology Comprehension is such that it can be viewed as a competence that can be transferred between subjects and contexts. This underlines the overall purpose of the subject and the role of CT and programming in enabling students' digital empowerment as a societal practice. From this perspective, the overall aim of Technology Comprehension is to promote transversal practices. In general, CT is mainly related to problem-solving and transversal practices, where working with real-world problems is characterised as a critical and analytical competence area. However, there is also an explicit focus on algorithmic practices as a basis.

CT in the Finnish curriculum

In the Finnish National Core Curriculum 2014 for grades 1–9 (FNBE, 2014), CT was not explicitly mentioned. However, as described in the Methods section, transversal, algorithmic, and problem-solving practices were emphasised as both transversal competences and subject-specific objectives.

Aspects related to CT were emphasized in the descriptions of the transversal competences T1 and T5. The Finnish national-level core curriculum is organised according to the objectives for teaching, alongside related core content and transversal competences. For example, in mathematics, objective O20 for teaching is 'to guide the pupil to develop his or her algorithmic thinking and skills in applying mathematics and programming in problem-solving' (FNBE, p. 403). The curriculum indicates that this objective is to be related to the mathematics core content, 'thinking skills and methods', and four transversal competences, including T1 (Thinking and learning to learn) and T5 (ICT competence).

Problem-solving practices

The selected curriculum sections include terms such as data, design, and information in connection with problem-solving, indicating that problem-solving practices and transversal competences T1 and T5 are related. Problem-solving practices of CT are explicit in mathematic as shown above in the objective O20 where problem-solving is related to programming. Further, information management is stressed 'to guide the pupil to develop his or her information management and analysis skills and to instruct him or her in critical examination of information'. In biology, problem-solving is mentioned in general terms as for example 'acquiring, handling and analysing information' and not explicitly connected to CT. While in chemistry and physics, problem-solving is connected to students' own research i.e. inquiry learning. In crafts, it is framed in practical problems that students face.

Algorithmic practices

The framing of algorithmic practices was approached by investigating how the terms *algorithm*, *function*, *variable*, and *program* appeared in the selected curriculum sections. In the mathematics curriculum, *function* is often related to programming. *Algorithm* was only mentioned in mathematics in, for example, evaluation criteria: 'The pupil is able to apply the principles of algorithmic thinking and to programme simple programs'. *Programming* was mentioned mainly in mathematics and as a core content in crafts, which stated that 'Embedded systems are used in crafts, i.e. programming is applied in the designing and producing'. As *programming* was mentioned in the transversal competence description T5 'ICT competence', there should be opportunities for programming in all subjects. Objective O9 in visual arts is an example of this: 'to inspire the pupil to apply means of visual production from different times and cultures in his or her visual production' (FNBE 2014, p. 459). This objective is connected to transversal

competence T5 indicating that programming computer graphics could be a means to achieve this objective.

Transversal practices

Framing the transversal practices of CT was approached by selecting the terms *create*/*creativity*, *communicate*, and *collaboration*. Although not all these terms are found in the framework of Table 1, they were interpreted as being closest to the overall aim of transversal practices. Creating and being creative was stressed as a transversal competence in T1 and T5 and were therefore found in connection to several disciplinary objectives. In mathematics, creativity is connected to combining logical thinking with solving mathematical assignments, as in O5: 'to support the pupil in solving mathematical assignments that require logical and *creative thinking* and in developing skills needed in it' (FNBE, 2014, p. 403). This objective is related to transversal competences T1 stressing creative working approach and doing things together. Further the objective is related to transversal competence T5 emphasising the use of information and communication technology in creative work. Problem solving in mathematics it typically understood as open ended problems with several possible solutions thus connecting this kind of approach with creative thinking and ICT competences indicates computational thinking. Furthermore, both visual arts and music objectives connect creative expressions to T1 and T5, (see for example O7 in music, Appendix Table 8). In general, *creativity* was frequently mentioned in the selected curriculum selections. The term *communication* was framed rather loosely, using communication technology for several purposes. This may imply that the aim was for students to have opportunities to learn to communicate their thinking using computers (see for example O8 in biology, Appendix Table 8).

Given the structure of the Finnish curriculum, there is room for interpretation. If CT is understood in a narrow way, focusing solely on algorithmic practices, there are few guidelines for teachers that explicitly refer to it. However, the curriculum indicated that the transversal competences T1 and T5 related to CT were connected to numerous disciplinary teaching objectives, but in an open and somewhat vague manner. This may guide curriculum readers to recognise both explicit and implicit opportunities for including CT in teaching a wide range of topics and subjects.

CT in the Norwegian curriculum

The term *computational thinking* in the Norwegian curriculum was only used once in the mathematics curriculum (NDET, 2020b) as part of the core element of the basic skills and competence aims. The term used throughout the rest of the curricula was *programming*. Therefore, in the Norwegian case, we analysed how both CT and programming were described and framed. The mathematics competence aims included CT-related goals connected to algorithmic practices throughout all stages. From year 5, the term programming was used in mathematics, and the term appeared in the competence aims for years 5 to 7 in

science and arts and crafts, and years 8 to 10 in music.

Problem-solving practices

In mathematics, CT was introduced as part of the core element 'Exploration and problem-solving'. CT was presented as inherent to the 'process of developing strategies and approaches to solve problems' (NDET, 2020b). This section emphasises that CT is also used in problem-solving without digital tools, which also includes analysis, reformulation, evaluation, breaking down problems into sub-problems and elements of debugging. To a certain extent, we consider these as explicit problem-solving practices.

Typical problem-solving practices in mathematics are engaging in pattern recognition and analysis. At later stages, students should gain experience in developing and improving algorithms, which require skills in decomposition, debugging, abstraction, and other problem-solving practices. Programming is promoted as a means of exploring probability and simulations, and mathematical properties and relationships, thereby as a problem-solving practice.

Programming in the Norwegian science curriculum followed two tracks. One was related to the core element 'technology' and implied using programming in the exploration, design, and making of technical products and systems. The other track places programming in the context of the core element 'natural-science practices and approaches' as a means of exploring natural phenomena. Both tracks integrate programming as a means for exploration, modelling, and using and creating technology, and programming in science education reflects the 'nature of science' and 'nature of technology', allowing students to 'combine experience and know-how with creative and innovative thinking'. Thus, these two core elements encompass several problem-solving practices.

In music and arts and crafts, problem-solving practices were less prominent. However, in years 8 to 10, students should experiment with sound from different sources using programming in music. In arts and crafts students should explore how digital tools and new technology can be used as means of communication in creative processes and products in years 8 to 10, which is implicitly understood as a framing of CT.

Algorithmic practices

In mathematics, the competence aims started in years 1 to 3 by following and creating step-by-step rules. Gradually, these were expressed as algorithms using variables, conditions, and loops in year 4 and programming was first explicitly mentioned in mathematics year 5, where the students were to create and program algorithms with the use of variables, conditions, and loops. These goals explicitly covered algorithmic practices and may be related to abstraction and decomposition but were also framed within

creative activities such as play and games. In year 6, programming and algorithms were to be used to explore geometrical figures and data.

Programming is introduced into the science and arts and crafts competence aims for years 5 to 7, and music competence aims for years 8 to 10. To a certain extent, we observed a pre-supposition that students have been introduced to algorithmic concepts in mathematics. In science, the algorithmic practices were framed implicitly through programming, such as making "technological systems that have a transmitter and receiver". The competence aims in music and arts and crafts were not centred on algorithmic practices, but rather applying algorithmic and problem-solving practices in aesthetic and creative activities.

Transversal practices

We found transversal CT practices in all four curricula, however, the subjects all represented their own framing and set of practices. A broader context of mathematics competence aims indicates that these problem-solving practices are meant to support the development of students' transversal practices such as visualisation and interpretation of data and models, real-life applications, and critical evaluation and reasoning. In natural science, students practice data handling, visualisation, and modelling when they engage in observing and explaining natural phenomena. The competence aims for years 8 to 10 explicitly include the use of programming as a means for exploration in both science and mathematics. In the context of the core element 'technology' in science, skills in generalisation support the development of an understanding of products and systems. This contributes to creative and critical thinking expressed through modelling and problem-solving, which is at the heart of the core element 'technology' in science.

In music and arts and crafts, the descriptions of basic digital skills emphasised programming as a means of creativity (both subjects) and innovation (arts and crafts). In music, students start at the early stages to play and experiment with sounds and rhythms and create patterns, which also involve digital tools. The music curriculum did not explicitly mention programming until year 10, when the students were engaged in creating, rehearsing, and processing music using digital tools and programming. Prior to that, in arts and crafts, the curriculum specified that students use digital tools to create stories and work with photography, which does not necessarily involve programming. After year 7, digital tools and programming were explicitly mentioned as means to create visual expressions and to communicate. In summary, CT practices in the curricula reflect the methodologies and knowledge production in the subjects.

Discussion and Conclusions

The national curricula of Denmark, Finland, and Norway outlined various elements, aspects, and facets that were thematically and conceptually linked to the concept of CT. Although the curricula used different

strategies and terms for introducing CT, the findings from the analysis showed important similarities and differences, which may also represent common challenges.

The analysed curricula vary in the use of *computational thinking* as a term. CT is not explicitly used in the Finnish and Norwegian curriculum: both countries use the term 'algorithmic thinking'. In Finnish curriculum 'algorithmic thinking' has narrower interpretation and focused to algorithmic practices, while in Norway, it is intended as a translation of the English term computational thinking. The Danish curriculum however explicitly uses the term CT and treats it as a dedicated competence within the subject Technology Comprehension. Programming was mentioned in all curricula, and ascribed both a broad and narrow meaning. On the one hand, in line with Lodi and Martini (2021) and Shute et al. (2017), our findings indicate that programming was used interchangeably with CT. On the other hand, programming was promoted through explicit algorithmic practices. In Norway and Denmark, curricula focus on a progression from simple rules to more technical elements where algorithms are described as descriptions of procedures involving the use of programming syntax. Despite the use of different terms, the centrality of algorithmic practices and programming in all three curricula may lead to a tension or even shift of attention away from the purpose of the more technical aspects as means and tools for broad spectrum of problem-solving, exploration and complex thinking strategies that CT introduces in a variety of school subjects.

A striking similarity in all three curricula is the emphasis on the framing and applications of CT in the context of existing subjects. Curriculum learning outcomes were often described in general terms, with some exceptions such as those that specified the programming language used (block- versus text-based; e.g. MoE, 2018b) or program technological systems (NDET, 2020d). Our findings highlight the importance of understanding how curricula are framed to create awareness of the knowledge and skills that society view as valuable, as proposed by Cuban (1992). When including CT as a competence that future generations must develop in light of a digitalised society (Voogt & Roblin, 2012; Zhang & Nouri, 2019), our findings indicate both a focus on somewhat narrow algorithmic practices as well as an intertwined interplay between algorithmic practices, problem-solving practices, and transversal practices grounded in the subject areas and their 'real-world' applications. The interplay of these three computational practices reveals a high degree of complexity concerning CT in the analysed curricula.

Comparison of the Danish, Finnish, and Norwegian curricula indicates that CT was framed within applications linked to subjects and to real-world problems. With a focus on transversal practices in the curricula, such as creativity, communication, collaboration, and critical and scientific thinking, CT might enable future generations to respond to global challenges, in line with what CT literature suggests (Kafai, 2016; Shute et al., 2017; Voogt & Roblin, 2012). In emphasising problem-solving and transversal practices, the curricula appeared to promote students' future participation in a digitalised society. This indicates that

the curricula promote a vision of engagement in deeper learning through computational 'objects to think with' (Papert, 1980, p. 21), as well as what Kafai (2016) describes as computational participation - social and societal engagement. However, while our analysis revealed that the Danish and Finnish curricula had explicit guidelines and outcomes that highlighted transversal practices, the Norwegian curriculum implicitly provided opportunities for transversal practices across learning outcomes, especially if the curriculum is read across aims, as suggested by Vinnervik and Bungum (2022).

Our findings indicate that the curricula offered numerous opportunities for CT, both implicitly and explicitly. Comparing the three countries, we find that the Danish curriculum offers the most explicit framing of CT both in terms of how CT is introduced and offering concrete goals. In the Norwegian curriculum, we find explicit CT practices in the competence goals. Additionally, the inclusion of programming in the overarching core elements and basic digital skills implies a wider range of implicit possibilities for CT. In the Finnish curriculum, we find the broadest but also the most implicit framing of CT as it is integrated through the transversal competences to several instruction outcomes. Considering these findings, further research is needed to determine whether and how teachers interpret and use these opportunities, as this pre-supposes the competences of CT and its role in the disciplines and how it may support subject-specific learning and inter-disciplinary competences of the curriculum (Vinnervik & Bungum, 2022).

The study has limitations. One is related to the selection of only official curricula as the material for analysis. This means that analysis relies on a narrow selection of texts while we know that such curricula are influenced by a network of other policy documents (Andersen et al., 2023). The broader context of the analysis and interpretations are therefore informed by literature. Focusing on the curricula limits the scope to the intentional curriculum and gives no insights into the actual uptake of the ideas in the educational system. Further investigations into how teachers in the different countries interpret these curricula would therefore be of interest, but beyond the scope of this study.

This paper examined how CT was framed in the Danish, Finnish, and Norwegian curricula and highlights similarities and differences in framings of CT in these countries. Findings indicate that all curricula recognised the value of CT captured in the complex interplay of problem-solving, algorithmic and transversal practices as the motivation for and aim of CT integration.

Acknowledgement

This article is written as a part of a larger project, *Mathematics, Science and Computational Thinking (MASCOT)*, funded by The Research Council of Norway, grant number 320322.

References

Andersen, R., Frågåt, T., Boğar, Y., Jensen, J. J., & Mifsud, L. (2023). Representations of Computational Thinking in Policy Documents in an Educational Context: The Cases of Denmark, Finland, and Norway. In *Proceedings of the 17th International Conference of the Learning Sciences-ICLS* 2023, pp. 35-42. International Society of the Learning Sciences.

Autio, T. (2013). The internationalization of curriculum research. In W. F. Pinar (Ed.), *International handbook of curriculum research* (pp. 17–31). Routledge.

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. https://doi.org/10.1145/1929887.1929905

Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education*, *72*, 145–157. https://doi.org/10.1016/j.compedu.2013.10.020

Binkley, M., Erstad, O., Herman, J., Raizen, S., Ripley, M., Miller-Ricci, M., & Rumble, M. (2012). Defining twenty-first century skills. In P. Griffin, B. McGaw, & E., Care (Eds.), *Assessment and teaching of 21st century skills* (pp. 17–66). Springer.

Bocconi, S., Chioccariello, A., Kampylis, P., Dagienė, V., Wastiau, P., Engelhardt, K., Earp, J., Horvath, M., Jasutė, E., & Malagoli, C. (2022). *Reviewing computational thinking in compulsory education: State of play and practices from computing education*. Publications Office of the European Union. https://data.europa.eu/doi/10.2760/126955

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*.

Care, E., Kim, H., Vista, A., & Anderson, K. (2018). Education System Alignment for 21st Century Skills: Focus on Assessment. *Center for Universal Education at The Brookings Institution*.

Cuban, L. (1992). Curriculum stability and change. In P. W. Jackson (Ed.), *Handbook of research on curriculum* (pp. 216–247). Macmillan.

Finnish National Board of Education [FNBE] (2014). *The national core curriculum for basic education.*

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher*, *42*(1), 38–43. https://doi.org/10.3102/0013189x12463051

Hansen, S. B., Hachmann, R., & Dohn, N. B. (2024). Computational thinking beyond computer science. In S. H. Klausen & N. Mård (Eds.), *Developing a didactic framework across and beyond school* (pp. 232–242). Routledge.

Haseski, H. İ., İlic, U., & Tuğtekin, U. (2018). Defining a new 21st century skill-computational thinking: Concepts and trends. *International Education Studies, 11*(4), 29–42. https://doi.org/10.5539/ies.v11n4p29

Hsieh, H.-F., & Shannon, S. E. (2005). Three approaches to qualitative content analysis. *Qualitative Health Research*, *15*(9), 1277–1288. https://doi.org/10.1177/1049732305276687

Iversen, O. S., Smith, R. C., & Dindler, C. (2018). From computational thinking to computational empowerment: A 21st century PD agenda. *Proceedings of the 15th Participatory Design Conference (Volume 1).* https://doi.org/10.1145/3210586.3210592

Kácovský, P., Jedličková, T., Kuba, R., Snětinová, M., Surynková, P., Vrhel, M., & Urválková, E. S. (2022). Lower secondary intended curricula of science subjects and mathematics: A comparison of the Czech Republic, Estonia, Poland and Slovenia. *Journal of Curriculum Studies*, *54*(3), 384–405. https://doi.org/10.1080/00220272.2021.1978557

Kafai, Y. B. (2016). From computational thinking to computational participation in K–12 education. *Communications of the ACM*, *59*(8), 26–27. https://doi.org/10.1145/2955114

Kleinheksel, A., Rockich-Winston, N., Tawfik, H., & Wyatt, T. R. (2020). Demystifying content analysis. *American Journal of Pharmaceutical Education*, *84*(1). https://doi.org/10.5688/ajpe7113

Lodi, M., & Martini, S. (2021). Computational thinking, between Papert and Wing. *Science & Education*, *30*(4), 883–908. https://doi.org/10.1007/s11191-021-00202-5

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. https://doi.org/10.1016/j.chb.2014.09.012

McGarr, O., & Engen, B. K. (2024). Justifications for the study of computers on the curriculum: Neo-vocational ideology veiled in progressive educational discourse. *European Journal of Education*, 59(2). https://doi.org/10.1111/ejed.12630

Ministry of Education (2018a). *Læseplan for forsøgsfaget teknologiforståelse* [Curriculum for experimental subject technology comprehension]. https://www.uvm.dk/-/media/filer/uvm/aktuelt/pdf18/181221-laeseplan-teknologiforstaaelse.pdf

Ministry of Education (2018b). *Teknologiforståelse—Måloversigt* [Technology comprehension—Objective overview]. https://emu.dk/sites/default/files/2019-02/GSK.%20F%C3%A6lles%20M%C3%A5l.%20Tilg%C3%A6ngelig.%20Teknologiforst%C3%A5else.pdf

Ministry of Children and Education (MoCE) (2023). *Forsøg med teknologiforståelse i folkeskolens obligatoriskeundervisning slutevaluering* [Experiments with technology comprehension in the primary school's compulsory education final evaluation]. https://www.uvm.dk/-/media/filer/uvm/aktuelt/pdf21/okt/211004-slutevaluering-teknologoforstaaelse.pdf

Ministry of Children and Education (MoCE) (2024). Aftale mellem regeringen (Socialdemokratiet, Venstre og Moderaterne) og Liberal Alliance, Det Konservative Folkeparti, Radikale Venstr og Dansk Folkeparti om folkeskolens kvalitetsprogram – frihed og fordybelse. (Agreement between the Government (the Social Democrats, the Liberal Party and the Moderate Party) and the Liberal Alliance, the Conservative People's Party, the Social Liberal Party and the Danish People's Party on the quality programme for primary and lower secondary schools – freedom and immersion). https://www.uvm.dk/-/media/filer/uvm/aktuelt/pdf24/mar/240320-aftale-om-folkeskolens-kvalitetsprogram-%E2%80%93-frihed-og-fordybelse.pdf

Norwegian Directorate for Education and Training (NDET). (2020a). *Curriculum for art and crafts (KHV01 02)*. https://www.udir.no/lk20/khv01-02?lang=eng

Norwegian Directorate for Education and Training (NDET) (2020b). *Curriculum for mathematics years 1–10 (MAT01 05).* https://www.udir.no/lk20/mat01-05?lang=eng

Norwegian Directorate for Education and Training (NDET) (2020c). *Curriculum for music (MUS01 02).* https://www.udir.no/lk20/mus01-02?lang=eng

Norwegian Directorate for Education and Training (NDET) (2020d). *Curriculum for natural science (NAT01 04)*. https://www.udir.no/lk20/nat01-04?lang=eng

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

Pears, A., Tedre, M., Valtonen, T and Vartiainen, H. (2021). *What makes computational thinking so troublesome?* 2021 IEEE Frontiers in Education Conference (FIE), Lincoln, NE, USA (pp. 1–7). IEEE. https://doi.org/10.1109/FIE49875.2021.9637416

Ross, A. (2003). *Curriculum: Construction and critique*. Routledge.

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

Vinnervik, P. (2023). An in-depth analysis of programming in the Swedish school curriculum—Rationale, knowledge content and teacher guidance. *Journal of Computers in Education*, *10*(2), 237–271. https://doi.org/10.1007/s40692-022-00230-2

Vinnervik, P., & Bungum, B. (2022). Computational thinking as part of compulsory education: How is it represented in Swedish and Norwegian curricula? *Nordic Studies in Science Education*, *18*(3), 384–400. https://doi.org/10.5617/nordina.9296

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. https://doi.org/10.1007/s10956-015-9581-5

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, *25*, 127–147. https://doi.org/10.1007/s10956-015-9581-5

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33–35. https://doi.org/10.1145/1118178.1118215

Zhang, L., & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K–9. *Computers & Education*, *141*. https://doi.org/10.1016/j.compedu.2019.103607

Appendix

The tables following below present illustrative examples of the selected parts of the Danish, Finnish and Norwegian curricula. The presented extracts describe how CT is integrated in the respective curricula categorised according to the three practices problem-solving, algorithmic and transversal practices. In addition, we highlight terms analysed according to the framework in Table 1 as related to CT, explicitly or implicitly.

Table 3. Comparison of problem-solving practices as framed in the general parts of the curricula.

Denmark	Finland	Norway
Computational thinking – as competency area: Year 3: The student can ***apply*** computational thinking to describe familiar and limited phenomena in everyday life Year 6: The student can ***follow*** and ***apply*** computational thinking in dealing with concrete issues or problems. Year 9: The student can ***reflect*** *on* and ***apply*** computational thinking to issues from the surrounding world. **Technological Capabilities – as competency area:** Year 3: The learner can, based on knowledge of the language and principles of digital technologies, ***act appropriately with digital technologies*** in specific situations Year 6: The student can, based on knowledge of about the language and principles of digital technologies***, act in an informed way with digital technologies*** in concrete situations Year 9: The learner can ***assess, select***, and skilfully ***apply*** digital technologies in authentic situations	**Transversal competencies** **T1 (Thinking and learning to learn):** In addition, thinking skills are developed by providing the pupils with diverse opportunities of both independent and collaborative problem solving, argumentation, reasoning, deduction, and understanding of interactions and connections between different issues, thus systemic thinking **T5 (ICT competence):** At the same time, the pupils practise source criticism and evaluate the way they and others, as well as different search engines and databases, work and produce information.	**Mathematics:** **Core element Problem-solving** Computational thinking is important in the process of developing strategies and approaches to solve problems and, means ***breaking a problem down into sub-problems*** that can be solved systematically. This also includes ***evaluating whether sub-problems can be solved best with or without digital tools***. Problem solving also means ***analysing and reformulating*** known and unknown problems, solving them and ***evaluating*** whether the solutions are valid. **Basic digital skills:** Digital skills in mathematics refers to the ability to use graphing tools, spreadsheets, CAS, dynamic geometry software and ***programming*** to explore and solve mathematical problems. **Science:** **Core element Technology** The pupils shall understand, develop and use technology, including ***programming*** and ***modelling***, in their natural-science work. [...] **Basic digital skills:** Digital skills in natural science refers to using digital tools to explore, register, calculate, visualise, ***program, model***, document and publish ***data*** from experiments, fieldwork and studies by others. [...]

Table 4. Comparison of algorithmic practices as framed in the general parts of the curricula.

Denmark	Finland	Norway
Technological Capabilities – as competency area: Year 3: The learner can, based on knowledge of the ***language and principles*** of digital technologies, act appropriately with digital technologies in specific situations Year 6: The student can, based on knowledge of ***about the language*** and principles of digital technologies, act in an informed way with digital technologies in concrete situations	**Transversal competencies** **T5 (ICT competence):** ***Programming*** is practised as a part of the studies of different subjects	**Mathematics:** **Basic digital skills:** Digital skills in mathematics refers to the ability to use graphing tools, spreadsheets, CAS, dynamic geometry software and ***programming*** to explore and solve mathematical problems. **Science:** **Core element Technology** The pupils shall understand, develop and use technology, including ***programming*** and modelling, in their natural-science work. [...] **Basic digital skills:** Digital skills in natural science refers to using digital tools to explore, register, calculate, visualise, ***program***, model, document and publish data from experiments, fieldwork and studies by others. [...] **Music** **Basic digital skills:** It [digital skills] also refers to using digital tools creatively to make recordings, process and manipulate audio and ***use programming*** in creative work **Arts and Crafts** **Basic digital skills:** They [digital skills] also involve ***using digital tools and programming*** in creative and innovative processes.

Table 5. Comparison of transversal practices as framed in the general parts of the curricula.

Denmark	Finland	Norway
Computational thinking – as competency area: Year 3: The student can apply computational thinking to ***describe*** familiar and delimited phenomena in everyday life. The student can ***reflect*** on and apply computational thinking to issues from the surrounding world.	**Transversal competencies** **T1 (Thinking and learning to learn):** The arts deepen ethical and ***aesthetic thinking*** by stirring emotions and ***creating*** new inventive ideas In addition, thinking skills are developed by providing the pupils with diverse opportunities of both independent and collaborative problem solving, argumentation, reasoning, deduction, and understanding of interactions and connections between different issues, thus ***systemic thinking*** **T5 (ICT competence):** Practical skills and personal production: The pupils are encouraged to utilise information and communication technology independently in different learning assignments and guided in the selection of working approaches and devices appropriate for different tasks.	**Science:** **Basic digital skills:** Digital skills in natural science refers to using digital tools to explore, register, calculate, ***visualise***, program, model, ***document and publish data*** from experiments, fieldwork and studies by others. [...] **Music** **Basic digital skills:** It [digital skills] also refers to ***using digital tools creatively*** to make recordings, process and manipulate audio and ***use programming in creative work*** **Arts and Crafts** **Basic digital skills:** They [digital skills] also involve ***using digital tools and programming in creative and innovative processes.***

Table 6. Comparison of problem-solving practices as framed in the learning goals.

Denmark	Finland	Norway
Computational thinking **Data:** Year 9: The student can ***process, assess***, and visualise ***data*** in a reflected way, using digital technology. The student has knowledge of ***data*** quality criteria **Algorithms:** Year 9: The student can ***assess the applicability of different algorithms*** and can use different methods to ***test algorithms*** The student has knowledge of different parameters to ***assess the usability of algorithms*** **Structuring:** Year 9: The student can ***structure phenomena and concepts*** in a problem domain and in ***computational models*** The student has knowledge of the principles of ***abstraction and structuring*** of a problem area as well as fundamental techniques for ***structuring data and processes*** **Modelling:** Year 9: The student can construct digital ***models*** of reality and use them to make predictions and inferences and assess limitations the model The student knows how ***abstractions*** of reality can be used to describe and process it in digital ***models and how to test a model*** according to its intent **Technological Capabilities** **Programming:** Year 6: The student can describe, adjust, and ***construct programmes*** in block-based programming languages and ***systematically test and debug*** their own and others' programmes Year 9: The student can ***read and understand programmes*** written in a text-based programming language *a*nd use it to systematically modify and ***construct programmes based on a problem specification*** The student has knowledge of methods for ***analysing*** and predicting the behaviour of programs and techniques for ***systematic and incremental development of programmes***	**Mathematics:** To guide the pupil to develop his or her algorithmic thinking and skills in applying mathematics and ***programming in problem-solving*** To guide the pupil to develop his or her ***information management and analysis*** skills and to instruct him or her in critical examination of information **Biology:** Guiding the pupils to also use electronic learning environments in acquiring, handling, analysing, and presenting biological information is essential in achieving the objectives of the instruction of biology. **Physics:** When ***conducting research***, the relevant stages of the research process are emphasised, such as reflecting on a problem or a phenomenon, planning, setting up experiments, making observations and measuring, compiling, and processing results, as well as evaluating and presenting results	**Mathematics:** Year 4: ***Explore*** and describe structures and patterns in play and games Year 5: ***Create and programme*** algorithms with the use of variables, conditions and loops Year 7: ***Use programming to explore*** data in tables and datasets Year 10: ***Explore*** mathematical properties and relationships by means of programming **Science:** Year 7: ***Explore***, make and program technological systems that consist of parts that work together Use and ***assess models*** that represent phenomena that cannot be observed directly […] Year 10: ***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 ***Use programming to explore*** natural-science phenomena **Arts and Crafts:** Year 10: ***Explore how digital tools and new technology may provide possibilities*** for different types of communication and experiences in creative processes and products

Table 7. Comparison of algorithmic practices as framed in the learning goals.

Denmark	Finland	Norway
Computational Thinking **Algorithms:** Year 3: The student can identify and ***formulate simple algorithms*** in informal form related to everyday situations and predict the behaviour of simple algorithms Year 6: The student ***can recognise and adjust algorithms*** in different contexts and explain their function. The student has knowledge of the ***characteristics of algorithms*** and their structure and how they are used in different contexts **Structuring:** Year 3: The student can describe everyday procedures ***using sequences, conditional selection and repetitions*** **Technological Capabilities** **Programming:** Yaer3: The student can follow and adjust simple ***programmes in at least one block-based language***. The student has knowledge of basic ***constructs in block-based programming languages***. Year 6: The student can describe, adjust, and ***construct programmes in block-based programming languages*** and systematically test and debug their own and others' programmes Year 9: The student can read and understand ***programmes written in a text-based programming language*** and use it to systematically modify and construct programmes based on a problem specification	**Mathematics:** O20: to guide the pupil to develop his or her ***algorithmic thinking*** and skills in applying mathematics and programming in problem-solving	**Mathematics:** Year 3: Create and follow ***rules and step-by-step instructions*** in play and games related to the coordinate system Year 5: Create and ***programme algorithms with the use of variables, conditions, and loops*** **Science:** Year 7: Explore, make and ***program*** technological systems that consist of parts that work together **Music:** Year 10: Create and ***programme*** musical sequences by experimenting with sound from different sources

Table 8. Comparison of transversal practices as framed in the learning goals.

Denmark	Finland	Norway
Computational thinking **Data:** Year 6: The student can collect, store, and ***visualise data*** Year 9: The student can process, assess, and ***visualise data in a reflected way***, using digital technology **Algorithm:** Year 3: Knowledge: The student ***has knowledge of everyday situations that can be described with algorithms*** **Structuring:** Year 3: The student ***can describe*** everyday procedures using sequences, conditional selection and repetitions **Modelling:** Year 3: The student ***can describe*** the reality a model represents and adjust the model to new needs Year 9: The student knows ***how abstractions of reality can be used to describe*** and process it in digital models and how to test a model according to its intent	**Mathematics:** O5: to support the pupil in solving mathematical assignments that require logical and ***creative thinking*** and in developing skills needed in it **Biology:** O8: to guide the pupil to use biological research equipment and information and ***communication technology*** **Music:** O7: to guide the pupil to record music and use information and communication technology in ***creative expression*** both when making music and as a part of multidisciplinary projects	**Mathematics:** Year 4: Explore and ***describe*** structures and patterns in play and games **Science:** Year 4: Explore technological systems that are composed of different parts and ***describe*** how the parts function and work together ***Reflect*** on how technology can solve challenges, create opportunities and lead to new dilemmas Year 10: Use and make models to ***predict or describe*** natural-science processes and systems and ***explain*** the strengths and limitations of the models **Arts and Crafts:** Year 7: Use programming to ***create interactivity and visual expressions*** Year 10: Explore how digital tools and new technology may provide possibilities for ***different types of communication and experiences in creative processes and products***