australasian journal of
TECHNOLOGY EDUCATION

The Australasian Journal of Technology Education is a peer refereed journal and provides a forum for scholarly discussion on topics relating to technology education. Submissions are welcomed relating to the primary, secondary and higher education sectors, initial teacher education and continuous professional development, and general research about technology education. Contributions to the ongoing research debate are encouraged from any country. The expectation is that the Journal will publish articles at the leading edge of development of the subject area.

The Journal seeks to publish

- reports of research,
- articles based on action research by practitioners,
- literature reviews, and
- book reviews.

Publisher: The Technology, Environmental, Mathematics and Science (TEMS) Education Research Centre, which is part of the Division of Education, The University of Waikato, publishes the journal.

Contact details: The Editor, AJTE, wendy.fox-turnbull@waikato.ac.nz

Cover Design: Roger Joyce

This journal provides immediate open access to its content on the principle that making research freely available to the public supports a greater global exchange of knowledge.

ISSN: 2382-2007

australasian journal of
TECHNOLOGY EDUCATION

Computational thinking: Visible in the classroom but invisible in the curriculum

Helena Isaksson Persson
KTH Royal Institute of Technology
Sweeden

Abstract

In our technology-intensive world, computing and programmed technological solutions have gained in importance, and their influence on curriculum, teaching, and learning has been substantial worldwide. Sweden, along with many countries, has integrated programming into the compulsory school curriculum as an integrated part of the teaching of Mathematics and Technology. In addition to a focus on programming, the new curricula also place significant emphasis on digital skills and on enhancing awareness of how the digitalisation of society affects us. Programming is described as a digital competence and computational thinking (CT) as important knowledge through which to facilitate learning and understanding of programming. Thus, it seems that CT, as seen in the Swedish context, should relate to both programming and digital competence. In this article, the aim is to examine the presence of CT in Swedish research literature and as a part of the discourse around the development of the Swedish curriculum. A content analysis of the curriculum and a thematic analysis of research publications show that CT is not well integrated into Sweden's educational system. However, CT-related activities are found in several subjects and research about CT, and programming is thriving. Requirements for the design of complex systems where understandings of humans and technology are equally important put new demands on education. Meeting these demands in education can be a challenge, but one subject in the Swedish curriculum seems to be suitable for the task, the technology subject. We conclude that the subject technology should be revised to include a greater focus on creativity regarding CT and the construction of computational technological artefacts.

Keywords

Computational thinking; curriculum; K-12 education; programming

Introduction

Computational thinking (CT) is a widely debated and elusive concept (Denning & Tedre, 2019; Pears et al., 2021). It is also doubtful whether today's focus on programming and coding gives students the right knowledge to live in an even more digitalised world in which artificial intelligence and machine learning systems are likely to dominate (Pears et al., 2021). This article examines the concept's position in technology education, taking Sweden as an example. Our primary data sources are the Swedish curriculum (Skolverket, 2018a, 2018b) for compulsory school and research on CT related to Swedish education (Appendix 1).

It is difficult to discuss CT and education without considering Seymour Papert. Piaget's knowledge-based learning (Papert, 1980/2020) influenced Papert to develop two dimensions of Piaget's theory in relation to how working with computers can support the development of, [1] intellectual structures in the child and [2] "the design of learning environments that are resonant with them" (1980/2020, p.181). The environment that Papert described was designed to support children's learning in mathematics and physics by "doing science" (1980/2020, p. 107).

The artificial world was important to Papert and things (computational artefacts) occupied the learning environment he created. It was in the interaction and communication with these "things" that learning took place (1980/2020). Papert described computers and robots as learning aids and the child communicated with these devices using an artificial turtle and the language LOGO (1980/2020). When the child tried an idea, the turtle showed the result as a drawing on the computer screen or with the help of a robot drawing the results on paper. It was important to Papert that the child was in control over the interaction with the technical artefacts and he argued, " . . . my central focus is not on the machine but on the mind . . . " (1980/2020, p. 8).

Papert described the potential for learning through child-machine interaction already in the 1980s, but robots and computers became everyday educational tools much later. Teachers lacked knowledge in computer science and CT. Instead, teaching focused on skills with digital tools, such as word processing software and desktop publishing (Denning & Tedre, 2019). Some decades after Papert, Jeanette Wing attracted considerable attention with her article "Computational Thinking" (Wing, 2006). She described a vision of a generalised application of computer science principles, arguing that a way of thinking using concepts and approaches from computer science will be a central competency area for future citizens. She argued that computational processes, with or without computers, will be central in understandings of the modern world. Both Papert and Wing emphasised the human aspect of the interaction between people and computers. Papert emphasised the exploration of ideas and Wing argued for CT as a fundamental skill (Papert, 1980/2020; Wing, 2006). Although Papert and Wing are influential, findings in this article demonstrate that there is a productive field of research active in research related to CT and teaching.

Much attention in society, however, has been on the skills of instructing the machine, on programming (Pears et al., 2021; Wanngård et al., 2015). Sweden, along with many countries, responded to political and societal pressure to address the need for these types of skills by integrating programming into the compulsory school curriculum (Wanngård et al., 2015). A revised curriculum was published in 2018 (Skolverket, 2018a, 2018b) with the aim of strengthening students' digital competence (Utbildningsdepartementet, 2017). The same year the Swedish National Agency for Education also published a research overview, disseminating knowledge on how digitalisation affects education (Skolverket, 2018c). The report described programming as a digital competence and CT as important for gaining an understanding of programming (Skolverket, 2018c). Concerning these directives, CT appears to be important and related to both programming and digital competence. The revision of the curriculum and the research overview serves as examples of how the authorities formulate desired goals for education and how it affects teachers' conditions for teaching (Larsson & Westberg, 2019).

After the curriculum revision, during the implementation of programming in schools, teachers faced several challenges, such as lack of knowledge and resources. In this process, the curriculum and supporting materials did not provide adequate support (Vinnervik, 2021). Consequently, the intentions of the governing documents do not always match the teachers' real conditions. Considering this discrepancy, we are interested in investigating how CT integrates into the Swedish curriculum and in school practice. This broad research interest will be explored through a qualitative content analysis of the curriculum (research question (a)), and a thematic analysis of previous research (research questions (b) and (c)). Research question (b) aims to define CT, while research question (c) aims to provide perspectives on how CT is presented in research literature.

a. What conditions provide the curriculum for CT to appear as a prioritised subject matter?
b. Which frameworks are used to define CT in the publications?
c. Which perspectives on CT are found in the publications?

The content analysis results indicate that the curriculum provides weak support for CT. On the other hand, the thematic analysis presents several perspectives of CT in the research literature. Despite the weak guidance from policy documents, CT content is present in school practice.

Method

To investigate how CT is integrated into the Swedish curriculum and in school practice, we analyse two primary data sources, the Swedish school curriculum for the compulsory school revised 2018 and research literature published in the period 2015 to 2021. The analysis objects are hence documents. According to Rapley (2007), documents are not just containers with objective information but tell about the specific context in which they are used. "We never just somehow neutrally or abstractly engage with documents, they are always engaged with in [sic] a specific local context; as such, they are always read or used in a specific way, to do specific work." (Rapley, 2007, p. 97). In line with Rapley's (2007) argument, we consider our data sources to reflect conditions for teaching and learning. The curriculum is analysed with content analysis and the research literature with thematic analysis.

Content analysis

Communicative material is the basis for content analysis. Examples of communicative material are transcripts of interviews, answers to open-ended survey questions and electronic written communication (White & Marsh, 2006). Quantitative content analysis and qualitative content analysis originate from different research traditions. Quantitative content analysis stems from a positivist tradition and uses statistical methods to test and present hypotheses. Qualitative content analysis stems from a humanistic research tradition and answers research questions by examining and identifying patterns in texts (White & Marsh, 2006).

Summative content analysis is a qualitative content analysis despite its procedure of counting keywords to explore the use of certain words in a text. The word count reveals patterns in the text and opens up for interpretation (Hsieh & Shannon, 2005). Prior (2003) states that words, as part of a text, are references to the social context described by the text. In this sense, reference counting in content analysis is a powerful tool for revealing patterns and providing information about the context in which the text is used. In this study is summative content analysis used to answer the research question:

a. What conditions provide the curriculum for CT to appear as a prioritised subject matter?

However, the number of references in a content analysis will not provide a complete understanding of the context (Prior, 2003). The use of a text needs to be studied to fully understand the relationship between text and context. Therefore, the content analysis in this study does not provide answers about how teachers interpret and use the curriculum. Rather, the analysis shows what conditions exist for teaching based on the support the curriculum provides.

Two documents in PDF format were analysed. Firstly, the curriculum for the compulsory school, preschool class, and school-age educare (Skolverket, 2018b). This curriculum was in Swedish and revised in 2018. Second, the same curriculum in English (Skolverket, 2018a). The tools used in the analysis process were Acrobat Reader, Excel, and NVivo.

The compulsory school in Sweden is preschool class (one year) and grades 1 to 9. The curriculum consists of three parts: fundamental values and tasks of the school, overall goals and guidelines, and syllabi. In our analysis, the Swedish words digital (digital), programmering (programming), and datalogi/sk/t (computational) were counted as references. Combinations or variations of these words in the curriculum were included in the analysis. Each word combination was counted as one reference. For example, datalogiskt tänkande (computational thinking) is a combination of datalogi/sk/t (computational) and tänkande (thinking) and counted as one reference, see Figure 2.

Rapid literature review

A rapid literature review was used to find research related to teaching and learning of CT in Swedish schools. The publications, the results of the rapid literature review, were later analysed with thematic analysis.

The process for our review follows the characteristics of a rapid literature review in the sense that the selection process was systematic, had a narrow focus, and was completed within a limited time (approximately 12 months, not full-time) (Denscombe, 2016; Folkhälsomyndigheten, 2017; Ganann et al., 2010). Rapid literature reviews can be extensive and form the basis for authorities' decisions (Folkhälsomyndigheten, 2017) or be the ground for analysis in small-scale qualitative research studies (Denscombe, 2016). The rapid literature review in this study was used in a small-scale qualitative research project. Compared to a systematic review, a rapid review may miss some perspectives since it is not as comprehensive as a systematic review and may not cover everything. Therefore, we do not claim to cover all research, but only that which emerged through the search procedure shown in Figure 1.

In order to find relevant research, we used the databases SCOPUS, ERIC, and Web of Science. Different combinations of keywords were tested. The best results were generated when the following terms were combined: *computational thinking*; *school*; *education*; *Sweden*; and *programming*. The term *programming* was added to broaden the search, while the term *school* was added to exclude studies related to higher education. The keyword combinations and examples of search strings are shown in Table 1.

Table 1. ***Keywords and Example Search Strings for Database Searches***

	Keyword combinations
	education-computational thinking-Sweden
	school-computational thinking-Sweden
	education-programming-Sweden
	education-school-programming-Sweden
Database	**Examples of search strings**
ERIC	Years 2002–2021(All years). References ERIC-**education** AND **computational thinking** AND **Sweden**

SCOPUS	Years 2015–2021 (All years). References (**school** AND **computational** PRE/0 **thinking** AND **Sweden**)
Web of Science	ALL FIELDS: (**education programming Sweden**) Timespan: All years.

The search in four databases resulted in four parallel search and selection procedures (Figure 1). The following selection procedure is used:

- STEP 1. Identification, screening titles
- STEP 2. Identification, reading abstracts
- STEP 3. Removing duplicates
- STEP 4. Read full texts
- STEP 5. Inclusion, selections of texts

Besides the keywords, the research should concern teaching and learning from preschool to upper secondary school. The earliest study meeting these criteria was published in 2015 and the latest in 2021, the same year the search for publications ended.

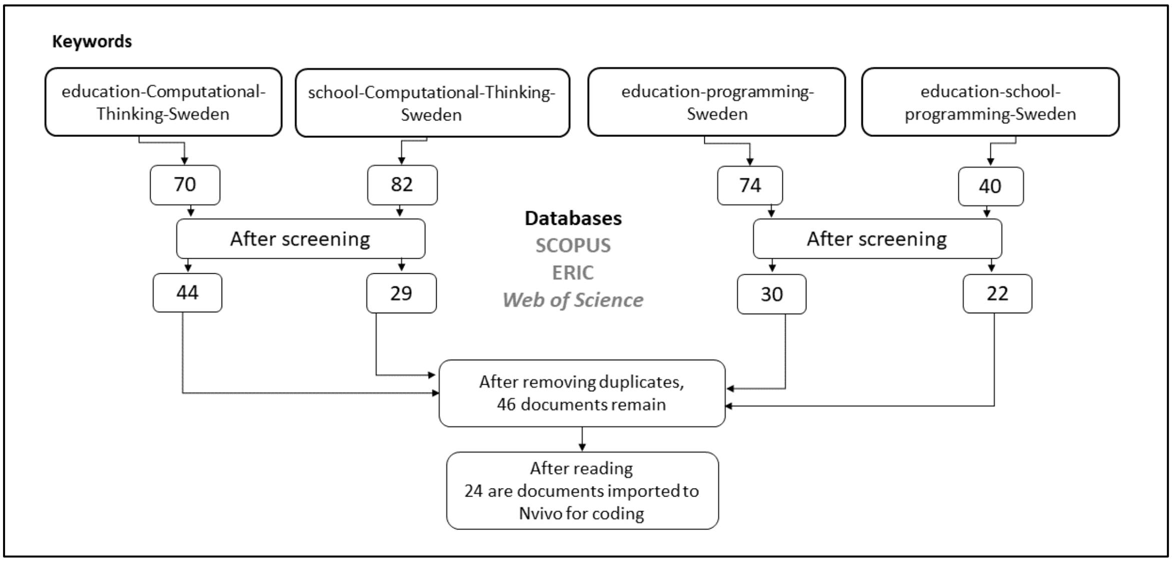

Figure 1. The seach and selection procedure for the rapid literature review.

After screening and selection, 24 research articles and papers were identified (Figure 1; Appendix 1). Three publications focus on preschool, one on upper secondary school and 20 refer to primary or grades K-12. Ten have been published prior to, and 14 after, the publication of the revised Swedish curriculum in 2018.

Thematic analysis

The thematic analysis is a qualitative method for identifying patterns within data. It is a flexible method independent of theory and therefore it is important for the researcher to have an active role and account for the strategies used in the analysis (Braun & Clarke, 2006). One way to start the analysis is the theoretical approach. Data for a particular analysis is selected from the corpus, that is, the entire research material. This data set is then searched for patterns and the patterns are coded in relation to the specific research question. The research question can also be developed through coding, an inductive approach (Braun & Clarke, 2006). Regardless of the chosen approach, similar codes are merged into themes. Ryan and Bernard (2003) respond to descriptions of thematic analysis where themes appear to emerge without explanation and emphasise the active role of the researcher in the search for knowledge and theoretical understanding. The researcher should thus be consistent when defining themes and report on the decisions made during the analysis. Braun and Clarke (2006) distinguish between semantic or latent levels of identification. Semantic identification is to identify a theme through descriptions from the data and thereby give it meaning on a superficial, explicit level. The interpretation is later built upon and summarises the descriptions. Latent identification of themes, on the other hand, goes beyond the semantic content of the data and examines underlying ideas. Reading the corpus and generating alternate codes and themes, the procedure continues and ultimately results in an interpretation that summarises the broad meaning of the data (Braun & Clarke, 2006; Ryan & Bernard 2003). In this study, we use a thematic analysis to answer the following research questions:

b. Which frameworks are used to define CT in the publications?
c. Which perspectives on CT are found in the publications?

Initially, the data set was analysed to find definitions of CT. When the data provided opportunities for further analysis, research questions (b) and (c) were developed. In this way, the analysis began theoretically, but an inductive approach was gradually developed. The analysis was close to the text and themes were identified at a semantic level. Table 2 shows examples of data extracts generated into codes and mapped to a theme. The first excerpt describing the importance of Papert and Wing was coded separately into two codes, *Papert* and *Wing (2006)*. The second excerpt refers to a framework by Brennan and Resnick. This extract was coded to *Brennan and Resnick (2012)*. All three codes were mapped to the theme *Frameworks* (Table 2).

The entire analysis followed the following steps in line with previous research (Braun & Clarke, 2006; Ryan & Bernard 2003).

Selection of 24 publications based on the rapid literature review (Appendix 1)

- The corpus was imported into the analysis software, NVivo. Excel was used as a supplementary tool.
- Descriptions in the text were coded at the semantic level.
- Codes were identified through an initial theoretical approach.
- An inductive approach generated research questions (b) and (c)
- Reading of corpus and individual publications was alternated.
- Codes were compiled into themes (Table 2)
- Codes and themes were generated until the themes provided clear meaning to the data (Braun & Clarke, 2006).

Table 2. ***Data Extract Coded for Three Codes and One Theme***

Data extract	Coded for	Theme
Contemporary discussions around programming in childhood education often draw from Papert's (1980) ideas of procedural thinking, and as a result of Wing's (2006) influential paper, **have manifested in the idea of CT**. (Otterborn et al., 2020, p. 254)	- Papert - Wing (2006)	Frameworks Frameworks
Brennan and Resnick (2012) **have developed a framework** for assessment of CT where it is suggested that CT stretch over computational concepts,	- Brennan and Resnick (2012)	Frameworks

computational practices, and computational perspectives as its three key dimensions. (Humble et al., 2019, 2.4 Computational thinking, para. 2)		

Results

The results consist of two parts. First, a summative content analysis of the Swedish curriculum (Skolverket, 2018a, 2018b). Second, a thematic analysis of the research publications provided with the rapid literature review.

A summative content analysis of the curriculum

A summative content analysis examined research question (a). This question concerns what conditions the curriculum provides for CT to emerge as a priority subject.

Table 3 shows the presence in the curriculum of the references *Computational Thinking*, *Programming*, and *Digital* (Skolverket, 2018a, 2018b). Table 3 indicates the word Yes, the presence of references and the word No, the absence of references. *Digital* is in all but three syllabi and there are no references at all to *Computational Thinking*.

Programming references are in the syllabi for three subjects: Civics, Mathematics, and Technology (Table 3). It is clear that references to *Digital* dominate the curriculum.

Table 3. Reference to Digital, Computational Thinking, and Programming in the Curriculum

Curriculum Syllabi (rows 1–21)	Reference count		
	Digital	Programming	Computational thinking
1: Art	Yes	No	No
2: Biology	Yes	No	No
3: English	No	No	No
4: Physics	Yes	No	No
5: Geography	Yes	No	No
6: Home and consumer studies	No	No	No
7: History	Yes	No	No
8: Physical education and health	Yes	No	No
9: Chemistry	Yes	No	No
10: Mathematics	Yes	Yes	No
11: Modern languages (optional)	Yes	No	No
12: Mother tongue tuition (optional)	Yes	No	No
13: Music	Yes	No	No
14: Religion	Yes	No	No
15: Civics	Yes	Yes	No
16: Sami (optional)	Yes	No	No

17: Crafts	Yes	No	No
18: Swedish	Yes	No	No
19: Swedish as a second language (optional)	Yes	No	No
20: Sign language for the hearing (optional)	No	No	No
21: Technology	Yes	Yes	No
Other parts of the curriculum			
Fundamental values and tasks of the school	Yes	No	No
Overall goals and guidelines	Yes	No	No
Preschool class	Yes	No	No
School-age educare	Yes	No	No

Figure 2 shows all variants and word combinations included in the references *Digital*, *Programming* and *Computational Thinking*. The difference in the number of references to *Digital* is markedly higher (178 references) than *Programming* (14 references). The most common references are *Digital tools*, followed by *Digital media* and *Digital aid/s*. Figure 2 confirms that there are no references to *Computational Thinking*.

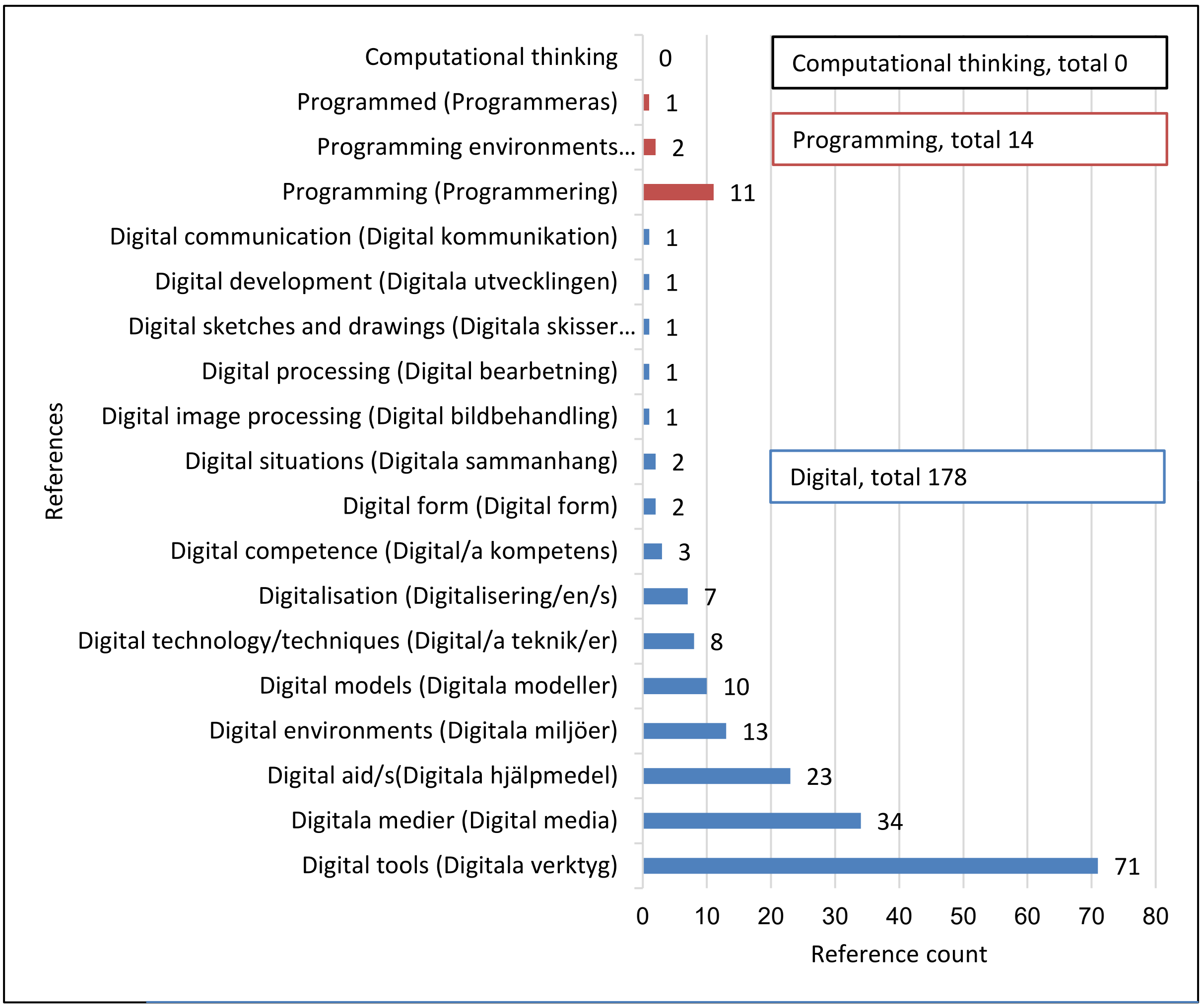

Figure 2. References counted in the content analysis

Found in almost all syllabi was *Digital tools,* the most common reference. *Digital media,* the second common reference, was in about a third of syllabi and in other parts of the curriculum. In comparison, the third most common reference, *Digital aid/s,* was only in the optional courses, *Modern languages* and *Mother tongue tuition*. Based on these differences *Digital tools* and *Digital media* was in the analysis suggested to have a greater impact on the curriculum as a whole than *Digital aid/s*.

A thematic analysis of research literature

The basis for the thematic analysis is the literature selected in the rapid literature review (Appendix 1) and further on are the results presented. The first results are in line with research question (b). This question concerns the frameworks used in the publications to define CT. The following results are in line with research question (c). This question concerns perspectives on CT found in the publications.

Framework used in the research literature

Fifteen frameworks were identified in the publications and mapped to the Theme *Frameworks* (Appendix 2). Figure 3 shows all codes mapped to this theme, most common were references to *Wing (2006)* followed by *Papert*.

Definitions of CT derived from Papert's work were learning by doing, exploring with code; testing and debugging; to construct artefacts from own ideas; and robot programming for children. Definitions of CT as a universal ability useful for all were derived from Wing's work. CT was regarded as a specific thinking or logic derived from computer science regarding problem solving in interaction with information agents e.g., computers.

Furthermore, definitions of CT other than Papert's and Wing's are listed below.

- Aho (2012) (referenced in Bråting & Kilhamn, 2020) and Kafai and Burke (2014) (referenced in Bowden, 2019; Kjällander et al., 2021; Palmer, 2017) described problem solving as a process where the problem is broken down into smaller components with solutions constructed as algorithms.
- Barr et al. (2011) stressed that certain attitudes are important when practising CT (referenced in Vinnervik, 2020).
- Bers et al. (2014) emphasised practice and argued for that certain mental tools and attitudes are in use when exploring and creating (referenced in Otterborn et al., 2020; Palmer, 2017; Zhang et al., 2020).
- Lye and Koh (2014) underlined the relationship between children and the digital world and the need for digital literacy when solving problems with programming (referenced in Bowden, 2019; Kjällander et al., 2021; Palmer, 2017).
- Perkovic (2015) (referenced in Mozelius & Öberg, 2017) addressed the understanding of processes and, more specifically, computational processes.
- Tedre and Denning (2016) (referenced in Vrieler, 2017) emphasised the development of CT through professional practice.

Five frameworks were described in more detail (Figure 3; Appendix 2; Appendix 3).

- Barefoot (2019) described six concepts and five approaches (referenced in Mannila et al. 2020; Otterborn et al., 2020).
- Brennan and Resnick (2012) defined three dimensions: computational concepts, computational practices, and computational perspectives (referenced in Bråting & Kilhamn, 2020; Mozelius & Öberg, 2017; Zhang et al., 2020).
- CSTA and ISTE (2020) presented nine concepts suitable for K-9 education (referenced in Kjällander et al., 2021).

- Shute et al. (2017) described a model for CT with six aspects (referenced in Humble et al., 2019, 2020).
- Zhang and Nouri (2019) developed the framework of Brennan and Resnick (2012) with one concept, three practices, and one perspective (referenced in Zhang et al. 2020).

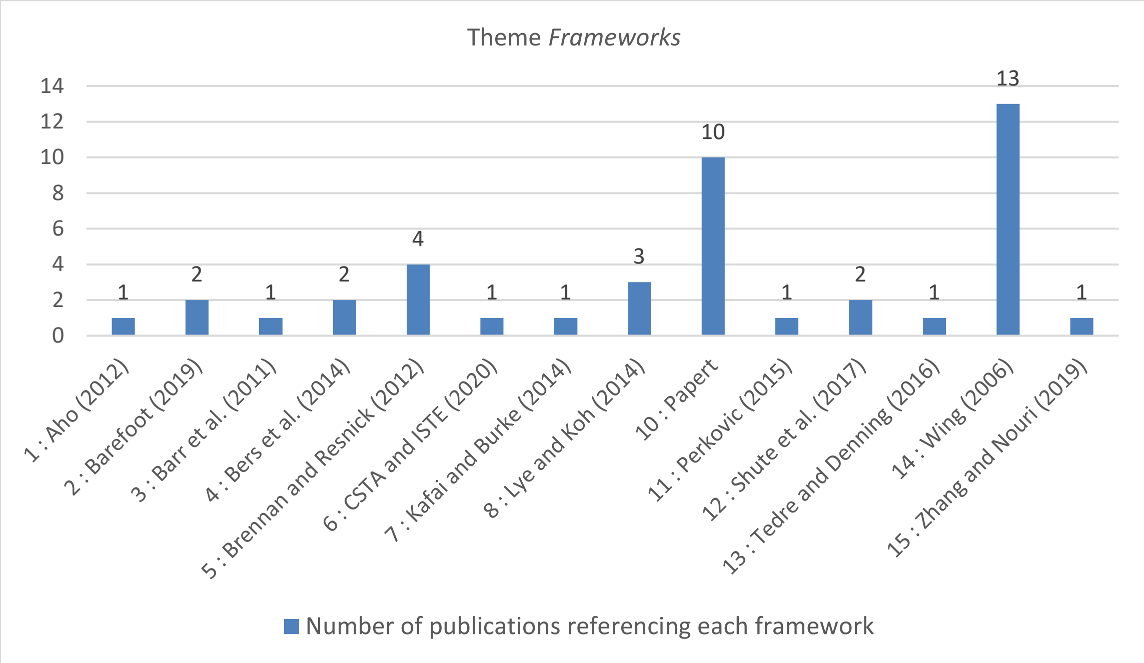

Figure 3. Number of publications referencing each framework.

There was no uniform definition of CT in the research reviewed, although some of the frameworks had overlapping terminology. An all over conclusion is that the legacy of Papert and Wing is clear. It is also clear that CT is a dynamic construct still evolving; for example, Zhang et al. (2019) building on the framework of Brennan and Resnick (2012) (referenced in Zhang et al., 2020).

Perspectives on CT in the research literature

Six themes were identified in the thematic analysis, Themes A-F (Table 4). These themes represent different perspectives of CT found in the publications. Theme (A) complements the findings on frameworks in line with research question (a). The number of definitions of CT in Appendix 3 therefore matches Theme A. Theme (B) describes CT as problem solving and is an additional finding in line with research question (a).

Theme (C) concerns CT and programming. It was hard to find distinctions between CT and programming in the analysis, as the two concepts were intertwined and hard to separate. Furthermore, it is not within the scope of this article to define programming. However, the analysis identified some distinctions. In some publications CT was described as the logic and strategies behind the problem-solving design. On the other hand, other publications described the development of CT abilities through programming activities.

Terms like language, code, instruction, and tools related to programming. For example, arguments about block versus text-based programming as suitable tools for different age groups.

Theme (D) concerns CT as a universal competence useful in the modern digital society. In the publications, general problem-solving ability, understanding of computational processes and products were examples of this competence. Theme (D) also complements findings in line with research question (a).

Themes (E) and (F) focus on educational issues. Theme (E) contains arguments concerning both the revised curriculum from 2018 and the implementation of CT and programming. Other topics discussed were assessment of CT skills, progression of CT skills, and CT for young children.

In Theme (F), *CT in school subjects*, is CT related to school subjects and the theme is further developed in Table 5.

Table 4. ***Themes Derived from the Thematic Analysis***

Perspectives on computational thinking (CT) derived from the reviewed publications.	
A. Definitions of CT	Definitions linked to frameworks (Appendix 3)
B. CT as problem solving	- general problem-solving skills based on computer science - problem solving as a process - problem is broken down into smaller components - solutions constructed as algorithms
C. CT and programming	-CT skills developed through programming -CT is logic, structure, and problem solving. Programming is language, codes, instructions, tools
D. CT as a universal competence	- in increasing digital society CT skills are beneficial for all
E. CT and educational perspectives	- CT in the curriculum -progression and assessment of CT skills -CT for young children
F. CT in school subjects	- focus on educational issues

Table 5 shows CT related to school subjects. In Mathematics CT relates to problem solving and to automation in Technology. Few results in the publications identified CT in relation to Civics apart from a description of a cross-disciplinary project with Mathematics. CT was also identified in subjects lacking programming learning objectives. In Physics, the students worked with simulations and in Art with visual interfaces. CT was also associated with Craft and Languages. Additional examples of classroom activities including CT were game design, animation, and visualisation. A common activity was block-based programming with Scratch software. But other resources besides Scratch was found in the analysis: programming languages e.g., Python; robots e.g., BlueBots and Lego EV3; embedded systems e.g., Arduino and Micro:bits (Chibas et al., 2018; Sjöberg et al., 2019; Svensson et al., 2020; Vinnervik, 2020; Tyrén et al., 2018).

Table 5. ***Computational Thinking Related to Swedish School Subjects***

Subjects	Computational thinking (CT) related to school subjects. Summaries.
Mathematics	CT in Mathematics is discussed in relation to problem solving.
Technology	CT in Technology is linked to the control of objects, for example robots. Simulations using micro:bits are also described.
Art	CT in Art. Students redesign a game, including both coding and aesthetic features (Bowden, 2019; 2020).
Civics	CT in Civics. An interdisciplinary activity (Civics and Mathematics) contextualises mathematics when the students analyse voting results with programming to learn about the Swedish electoral system (Sjöberg et al., 2019).
Craft	CT in Craft. Ahmed et al. (2019) study students' perceptions of programming. Students associate programming with Craft. This association is explained by the fact that construction is a prominent element in programming and CT.
Language	CT in Language. There are several languages in the curriculum. Compulsory languages are Swedish/Swedish as a second language and English. Students associate programming with language, as there are similarities in structures and syntax. They also associate with language when using Swedish and English in programming (Ahmed et al., 2019). Heintz et al. (2015) see possibilities to introduce CT concepts during language lessons.
Physics	CT in Physics. Students analyse physics problems by programming simulations that visualise phenomena through animations. Svensson et al. 2020, claim that simulations and animations help students understand concepts not only in physics but also in other fields of science.

Discussion and conclusion

The curriculum is an expression of educational policy governance and formulates desired goals for education (Larsson & Westberg, 2019). Publications from the National Agency for Education claim that CT is important for understanding programming (Skolverket, 2018c), yet our analysis is that this is not reflected on a semantic level in the curriculum. While the content analysis only focuses on a semantic level and does not provide interpretations of underlying meanings, the results offer an understanding of the conditions for integrating CT into both the curriculum and practical implementation.

The results indicate a strong emphasis on digital competencies in the curriculum, with a particular focus on digital tools and media (Figure 2; Table 3). However, compared to the pervasiveness of digital competence in the curriculum, programming receives only moderate attention, and CT is entirely absent (Figure 2; Table 3). Therefore, we conclude that the conditions for CT to emerge as a priority subject are poor.

On the other hand, the research literature underlying the thematic analysis shows that CT is already integrated into classroom practice through programming activities (Table 5). Characteristically, the activities that include CT are programming using Scratch software. Although the reviewed literature offers various descriptions of CT, a unified definition is still difficult to achieve. Nevertheless, our analysis reveals four broad definitions: components, concepts, approaches, and practices (Appendix 3). There are also indications that CT continues to evolve. Researchers build on definitions of other researchers (Zhang et al., 2020), and strategies regarding assessment and progression being developed (Mannila et al., 2020; Zhang et al., 2020). Despite an increasing diversity of CT definitions, the legacy of Papert and Wing is clear (Figure 3). Their influences are also reflected in Themes (A), (B) and (D) in Table 4. These themes summarise the essence of CT as a universal competence useful for problem solving. Theme (C) highlights the interconnectedness between CT and programming activities.

We also see a contradiction in the results: Theme (D) presents CT as a universal competence and Theme (F) as domain dependent and linked to different school subjects (Table 4). This contradiction can actually help us to pinpoint CT. CT is a competence for everyone, not just for computer scientists as Wing claims (2006; 2017). It is also domain dependent, which is in line with Denning and Tedre (2019): “We see CT as mostly domain dependent – for example, how you think about computation in biology is different from physics” (p. 191). Despite the weak support from the curriculum and the diversity of CT definitions, the results of our thematic analysis show potential for using CT in specific subjects. Not only in subjects with programming objectives but in others as well. The results in Table 5 show examples of CT adapted to domain-specific use.

However, there can be problems with limiting CT to programming and problem solving. CT is changing and expanding to focus on the design of complex systems where a deep understanding of humans and technology are equally important (Denning & Tedre, 2019; KTH, 2022; Pears et al., 2021). Meeting these demands in education can be a challenge, but one subject in the Swedish curriculum seems to be suitable for the task, the Technology subject. The subject is already about the artificial world, the creation of technical artefacts and systems, ethical dilemmas in sustainable product development, and historical and future perspectives on technology (Skolverket 2018a; 2018b). The subject is characterised by engineering methods as the design process with a strong focus on human-centred design (Skolverket 2018a; 2018b). The design process is about finding the right problems, solving them by constructing artefacts, and thus fulfilling human needs (Hughes, 2009; Norman, 2013). It also seems to be the most appropriate school subject to take responsibility for the legacy of Papert and further develop his ideas on teaching strategies for the interplay between humans and technical artefacts, where the students’ own ideas are at the centre. Based on our previous arguments, we believe that the Technology subject offers significant potential as a research arena for exploring how to integrate CT into engineering approaches, including the design process and system design.

We conclude that Sweden, and other countries with a similar integrative approach to embedding CT in the curriculum, need to place greater emphasis on the relevance of CT and programming for school subjects like Technology, where programming is explicitly mandated to be taught, and where CT skills can be expected to be developed. We argue that the Swedish curriculum must be strengthened, providing a clearer role for the subject of Technology in developing CT and programming.

References

Braun, V., & Clarke, V. (2006) Using thematic analysis in psychology. *Qualitative Research in Psychology*, *3*(2), 77–101.

Denning, P.J. & Tedre, M. (2019). *Computational thinking*. The MIT Press.

Denscombe, M. (2016). *Forskningshandboken: För småskaliga forskningsprojekt inom samhällsvetenskaperna* (3rd ed.) Studentlitteratur.

Folkhälsomyndigheten (2017). *Handledning för litteraturöversikter: Förutsättningar och metodsteg för kunskapsframtagande baserat på forskningslitteratur vid Folkhälsomyndigheten* (2nd ed.). Folkhälsomyndigheten.

Ganann, R., Ciliska, D., & Thomas, H. (2010). Expediting systematic reviews: methods and implications of rapid reviews. *Implementation Science*, *5*(56). https://doi.org/10.1186/1748-5908-5-56

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

Hughes, J. (2009). Practical reasoning and engineering. In A. Meijers, D.M. Gabbay, P. Thagard, & J. Woods (Eds.), *Philosophy of technology and engineering sciences* (pp. 375–402). Amsterdam: North Holland.

KTH Royal Institute of Technology (2022, November 4). *School of Electrical Engineering and Computer Science*. https://www.kth.se/en/eecs

Larsson, E. & Westberg, J. (Eds.) (2019). *Utbildningshistoria: en introduktion* (3rd ed.). Studentlitteratur.

Mannila, L., Heintz, F., Kjällander, S., & Åkerfeldt, A. (2020). Programming in primary education: Towards a research-based assessment framework. *Workshop in Primary and Secondary Computing Education (WiPSCE '20),* October 28–30, 2020, Virtual Event, Germany. https://doi.org/10.1145/3421590.3421598

Norman, D.A. (2013). *The design of everyday things*. (Revised and expanded edition.). Basic Books.

https://www.qsrinternational.com/nvivo-qualitative-data-analysis-software/about/nvivo

Papert, S. (2020). *Mindstorms: Children, computers, and powerful ideas.* (Original work published 1980). Basic Books.

Pears, A., Tedre, M., Valtonen, T., & Vartiainen, H. (2021, October 13−16). *What makes computational thinking so troublesome?* 2021 IEEE Frontiers in Education Conference (FIE), Lincoln. Nebraska, USA. https://doi.org/10.1109/FIE49875.2021.9637416

Prior, L. (2003). *Using documents in social research*. Sage.

Rapley, T. (2007). *Doing conversation, discourse and document analysis*. Sage. https://doi.org/10.4135/9781849208901

Ryan, G., & Bernard, G. (2003). Techniques to identify themes. *Field* Methods, *15*(1), 85–109. https://doi.org/10.1177/1525822X02239569

Skolverket (2018a). *Curriculum for the compulsory school, preschool class and school-age educare 2011: revised 2018*. Skolverket.

Skolverket (2018b). *Läroplan för grundskolan, förskoleklassen och fritidshemmet 2011: reviderad 2018*. (5th ed.). Skolverket.

Skolverket (2018c). *Digitaliseringen i skolan - möjligheter och utmaningar* (Forskning för skolan). https://www.skolverket.se/publikationer?id=3971

Utbildningsdepartementet (2017). *Nationell digitaliseringsstrategi för skolväsendet*. https://www.regeringen.se/contentassets/72ff9b9845854d6c8689017999228e53/nationell-digitaliseringsstrategi-for-skolvasendet.pdf

Vinnervik, P. (2021). *När lärare formar ett nytt ämnesinnehåll: intentioner, förutsättningar och utmaningar med att införa programmering i skolan* [Doctoral thesis, Umeå University]. http://umu.diva-portal.org/smash/record.jsf?pid=diva2%3A1589572&dswid=8631

Wanngård, K., Rankka, M., & Lorentzon, M. (2015, February 10). Stockholms stad inför dataslöjd på skolschemat. *Dagens nyheter*. https://www.dn.se/debatt/stockholms-stad-infor-dataslojd-pa-skolschemat/

White, M., & Marsh, E. (2006). Content analysis: A flexible methodology. *Library Trends*, *55*. https://doi.org/10.1353/lib.2006.0053

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

Wing, J. M. (2017). Computational thinking's influence on research and education for all. *Italian Journal of Educational Technology*, *25*(2), 7–14. https://doi.org/10.17471/2499-4324/922

Zhang, L. C., Nouri, J., & Rolandsson, L. (2020). Progression of computational thinking skills In Swedish compulsory schools with block-based programming. *ACE 2020 - Proceedings of the 22nd Australasian Computing Education Conference*, Held in Conjunction with Australasian Computer Science Week, 66–75. https://doi.org/10.1145/3373165.3373173

Appendix 1

All publications used in the thematic analysis.

Ahmed, G., Nouri, J., Norén, E., & Zhang, L. (2019). *Students' perceptions of programming in primary school*. WiPSCE'19, October 23–25, 2019, Glasgow, Scotland UK, ACM International Conference Proceeding Series, https://doi.org/10.1145/3361721.3362112

Bowden, H. M. (2019). Problem-solving in collaborative game design practices: epistemic stance, affect, and engagement. *Learning, Media and Technology, 44*(2), 124–143. https://doi.org/10.1080/17439884.2018.1563106

Bowden, H. M., & Aarsand, P. (2020). Designing and assessing digital games in a classroom: an emerging culture of critique. *Learning Media and Technology*, *45*(4), 376–394. https://doi.org/10.1080/17439884.2020.1727500

Bråting, K., & Kilhamn, C. (2020). Exploring the intersection of algebraic and computational thinking. *Mathematical Thinking and Learning*, *23*(2), 170–185. https://doi.org/10.1080/10986065.2020.1779012

Chibas, A., Nouri, J., Noren, E., & Zhang, L. (2018). Didactical strategies and challenges hen teaching programming in pre-school. *Edulearn18: 10th International Conference on Education and New Learning Technologies*, pp. 3345–3350. IATED. http://dx.doi.org/10.21125/edulearn.2018

Heintz, F., & Mannila, L. (2018). Computational thinking for all: An experience report on scaling up teaching computational thinking to all students in a major city in sweden. *ACM Technical Symposium on Computer Science Education (SIGCSE)*, *9*(2), 65–71. ACM Inroads. https://doi.org/10.1145/3159450.3159586

Heintz, F., Mannila, L., Färnqvist, T. (2016). A review of models for introducing computational thinking, computer science and computing in K-12 education. *2016 IEEE Frontiers in Education Conference (FIE)*, Eire, PA, USA, 2016. (pp. 1–9). https://doi.org/10.1109/FIE.2016.7757410

Heintz, F., Mannila, L., Nygårds, K., Parnes, P., & Regnell, B. (2015). Computing at School in Sweden - Experiences from introducing computer science within existing subjects. In A. Brodnik, & J. Vahrenhold (Eds.), *Informatics in Schools. Curricula, Competences, and Competitions* /Lecture Notes in Computer Science and General Issues (Vol. 9378, pp. 118–130). Springer. https://doi.org/10.1007/978-3-319-25396-1_11

Humble, N., Mozelius, P., Sällvin, L. (2019). On the role of unplugged programming in K-12 education. *Proceedings of the European Conference on e-Learning, ECEL*, 2019-November, pp. 224–230. https://doi.org/10.34190/EEL.19.049

Humble, N., Mozelius, P. and Sällvin, L. (2020), Remaking and reinforcing mathematics and technology with programming – teacher perceptions of challenges, opportunities and tools in K-12 settings, *International Journal of Information and Learning Technology*, *37*(5), pp. 309–321. https://doi.org/10.1108/IJILT-02-2020-0021

Kjällander, S., Mannila, L., Åkerfeldt, A., & Heintz, F. (2021). Elementary students' first approach to computational thinking and programming. *Education Sciences*, *11*(2). https://doi.org/10.3390/educsci11020080

Mannila, L., Heintz, F., Kjällander, S., & Åkerfeldt, A. (2020). *Programming in primary education: Towards a research-based assessment framework.* Workshop in primary and secondary computing education (WiPSCE '20), October 28–30, 2020 [Virtual Event], Germany. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3421590.3421598

Mozelius, P., & Öberg, L. M. (2017). *Play-based learning for programming education in primary school: The Ostersund Model.* Proceedings of the 16th European Conference on E-Learning (Ecel 2017), 375–383.
Otterborn, A., Schönborn, K. J., & Hultén, M. (2020). Investigating preschool educators' implementation of computer programming in their teaching practice. *Early Childhood Education Journal, 48*(3), 253–262. https://doi.org/10.1007/s10643-019-00976-y
Palmer, H. (2017). Programming in preschool-with a focus on learning mathematics. *International Research in Early Childhood Education, 8*(1), 75–87. https://doi.org/10.1007/s10643-019-00976-y
Pears, A., Dagiene, V., & Jasute, E. (2017). Baltic and Nordic K-12 T teacher perspectives on computational thinking and computing. *Informatics in Schools: Focus on Learning Programming*, 141–152. https://doi.org/10.1007/978-3-319-71483-7_12
Sjöberg, C., Nouri, J., Sjöberg, R., Noren, E., & Zhang, L. (2018). Teaching and learning mathematics in primary school through Scratch. *Edulearn18: 10th International Conference on Education and New Learning Technologies*, 5625–5632.
Sjöberg, C., Risberg, T., Nouri, J., Noren, E., & Zhang, L. (2019). A lesson study on programming as an instrument to learn mathematics and social science in primary school. *13th International Technology, Education and Development Conference (Inted2019)*, 2230–2235. https://doi.org/10.21125/inted.2019.0622
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, K., Eriksson, U., & Pendrill, A. M. (2020). Programming and its affordances for physics education: A social semiotic and variation theory approach to learning physics. *Physical Review Physics Education Research*, 16(1). https://doi.org/10.1103/PhysRevPhysEducRes.16.010127
Tyrén, M., Carlborg, N., Heath, C., & Eriksson, E. (2018). Considerations and technical pitfalls for teaching computational thinking with BBC micro:Bit. Paper presented at the *ACM International Conference Proceeding Series,* Part F13770*2,* 81–86. https://doi.org/10.1145/3213818.3213829
Vinnervik, P. (2020). Implementing programming in school mathematics and technology: Teachers' intrinsic and extrinsic challenges. *International Journal of Technol*ogy, *32*, 213–242. https://doi.org/10.1007/s10798- 020-09602-0
Vrieler, T. (2017). Digital capital - A platform for developing computational thinking. *17th Koli Calling International Conference on Computing Education Research (Koli Calling 2017)*, 205–206. https://doi.org/10.1145/3141880.3143782
Zhang, L., Nouri, J., & Rolandsson, L. (2020). Progression of computational thinking skills in Swedish compulsory schools with block-based programming. *ACE 2020 - Proceedings of the 22nd Australasian Computing Education Conference*, 66-75. https://doi.org/10.1145/3373165.3373173

Appendix 2

Frameworks used in the reviewed publications.

1. Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal, 55*(7), 832–835. https://doi.org/10. 1093/comjnl/bxs074
2. Barefoot (2019). http://barefootcas.org.uk.
3. Barr, D., Harrison, J., & Conery, L. (2011). Computational thinking: A digital age skill for everyone. Learning and leading with technology. *ISTE Learning and Leading with Technology, 5191*(March/April), 21–22.

4. 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.

5. 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* (pp. 1–25), April 1317, 2012. Vancouver, Canada.

6. CSTA and ISTE (https://www.iste.org/explore/computational-thinking-all accessed on 30 December 2020).

7. Kafai, Yasmin B., and Quinn Burke. 2014. *Connected code. Why children need to learn programming*. The MIT Press.

8. 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 Humam Behaviour, 41*(4), 51–61. https://doi.org/10.1016/j.chb.2014.09.012

9. Mannila, L. (2017). *Att undervisa i programmering i skolan – varför, vad och hur?* Studentlitteratur AB.

10. Papert
- Papert, S. (1980). *Mindstorms: children, computers, and powerful ideas*. Basic Books.
- Papert, S. (1996). An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning*, 1(1), 95–123. https://doi.org/10.1007/BF00191473 I
- Papert, S. (1993). The children's machine. *Technology Review, 96*, 2828.

11. Perkovic, L. (2015). *Introduction to computing using Python*: *An application development focus*. Wiley.

12. Shute, V.J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, *22*, 142–158.

13. Tedre, M., & Denning, P.J. (2016). The long quest for computational thinking. *Koli Calling Conference on Computing Education Research*, 120–129. https://doi.org/10.1145/2999541.2999542

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

15. Zhang, L., & Nouri, J. (2019). A systematic review of learning by computational thinking through Scratch in K-9. *Computers & Education*, 103607.

Appendix 3

	Frameworks				= in framework
Terms used in the frameworks to define CT	Barefoot (2019)	Brennan & Resnick (2012)	CST & ISTE (2020)	Shute et al. (2017)	Zhang & Nouri (2019)
Components					
abstraction (also Concept)				■	
algorithms (also Concept)				■	
debugging (also Approach, Practice)				■	
decomposition (also Concept)				■	
iteration				■	
generalization				■	

Concepts					
abstraction (also Component)					
algorithms (also Component)					
automation					
conditionals					
data					
data analysis					
data collection					
data representation					
decomposition (also Component)					
evaluation					
events					
input/output					
logic thinking					
loops					
operators					
parallelism					
pattern recognition					
sequences					
simulation					
Approaches					
collaborating					
creating					
debugging (also Component, Practice)					
persevering					
tinkering					
Practices					
abstracting					
debugging (also as Approach, Component)					
iterative					
interpreting and communicating code					
modularizing					
multimodal design					
predictive thinking					
remixing					
reusing					
testing					