Digital Experiences in Mathematics Education (2022) 8:27–49
https://doi.org/10.1007/s40751-022-00102-5

Computational Thinking in the Primary Mathematics Classroom: a Systematic Review

Siri Krogh Nordby[1] · Annette Hessen Bjerke[1] · Louise Mifsud[1]

Accepted: 27 January 2022 / Published online: 11 February 2022

Abstract
Computational thinking (CT) has acquired the status of a necessary 21st-century skill and is currently being introduced in school curricula around the world, despite a lack of consensus about what it entails. The aims of this review are to provide an overview of the existing literature on CT activities in primary mathematics education, and to articulate how it is integrated into the teaching and learning of primary mathematics. This systematic review presents and analyses the findings of 10 empirical studies, revealing a recent increased focus on the inclusion of CT in primary mathematics classrooms, as most studies are published around 2020. Our findings indicate two categories of such activities, one focusing on skills (such as mainly sequencing, looping, conditionals, debugging, decomposition, and abstraction) and one on process-oriented activities (communication, creativity, exploration, and engagement). Furthermore, we found that, while there are studies reporting on mathematics being taught directly through CT activities (full integration), in most studies, the mathematics content was emphasised, with CT built in as a way for students to demonstrate their understanding of mathematics concepts (partial integration). This review identifies current gaps in the field and the need to investigate further such process-oriented activities, the use of these activities in accelerated mathematics, and the need for different methodological approaches in primary mathematics.

Keywords Computational thinking · Primary mathematics · Computational thinking activities · Systematic literature review · Partial integration · Full integration · Mathematics learning

✉ Siri Krogh Nordby

Annette Hessen Bjerke
anetsen@oslomet.no

Louise Mifsud
lomi@oslomet.no

1 Oslo Metropolitan University, Oslo, Norway

Computational thinking (CT) in education has recently received considerable attention in policy initiatives (Bocconi et al., 2018; Hsu et al., 2018). Despite this extensive attention—through which it has been highlighted that CT is a necessary 21st-century skill that is crucial for fostering children's critical and analytical thinking, and creativity and competence in problem solving (Voogt et al., 2015)—there seems to be little or no agreement about what it encompasses (Brennan & Resnick, 2012; Grover & Pea, 2013; Shute et al., 2017; Weintrop et al., 2016; Zhang & Nouri, 2019). Surprisingly, despite this lack of consensus, several countries have introduced CT into their curricula, with science and mathematics pinpointed as the natural subjects within which CT should be integrated (Weintrop et al., 2016).

Drawing on Piaget's theories of cognitive development, Papert (1980) argued that, when children learn to programme computers, the "process of learning is transformed" (p. 21). This happens as learning becomes more active, personal, and self-directed. His constructionism is grounded in the belief that learning grows out of the active construction of ideas that are formed and transformed when expressed through different media, actualised in particular contexts, developed through interactions, and worked out by individual minds (Ackermann, 2001; Papert, 1980). Papert initially linked programming to mathematics but, ultimately, also to facilitating thinking and learning across multiple disciplines, including science and literature (Papert, 1980, 1996). Papert's (1996) notion of CT did not include a definition, but was related to the construction of ideas that are "explicative" as well as accessible and powerful (p. 116).

Due to the difficulty of learning programming languages and the use of learning activities that did not reflect children's interests, Papert's idea of "CT for all" was, to a certain extent, ahead of its time (Resnick et al., 2009). Empirical studies of Logo programming indicated that teachers provided assistance more than instruction, and few children improved their thinking skills (Kurland et al., 1986; Pea & Kurland, 1983). Hence, the inclusion of Logo in school contexts disappeared within a decade, mainly because of a lack of subject-matter integration and a lack of qualified instructors (Noss and Hoyles, 1996). However, subsequently, Wing's (2006) definition cast CT back into the educational limelight, and it involved "solving problems, designing systems and understanding human behaviour" (p. 33).

In the wake of Wing's appeal for CT to become a ubiquitous skill among children, and the ensuing discussion of what it is, several researchers have attempted to address the ambiguity that has characterised its discussion in education (Brennan & Resnick, 2012; Grover & Pea, 2013; Lye & Koh, 2014; Resnick et al., 2009; Shute et al., 2017; Weintrop et al., 2016). Discussions about the definition of CT have labelled it as "[a] key tool for supporting […] cognitive tasks" (Grover & Pea, 2013, p. 40) and as "[a] foundation required to solve problems effectively and efficiently" (Shute et al., 2017, p. 151). The distinction between programming and CT is, at best, diffuse and is made even more blurred through discussions of non-computer problem-solving (or "unplugged") activities. Grover and Pea (2013) criticised unplugged activities for keeping learners from having "crucial computational experiences" (p. 40).

This paper does not attempt to define CT, but rather to present an overview of which of its activities have been addressed in primary mathematics education research and how it has been integrated into the learning of mathematics, according to various studies. The ambiguity of the term *CT* itself has resulted in diverse ways of labelling its activities. Brennan and Resnick (2012), for instance, used CT *concepts* (e.g. sequences, loops, and conditionals), *practices* (i.e. the practices designers develop as they engage with the concepts, such as debugging), and *perspectives* (i.e. the perspectives designers form about the world around them and about themselves), while Weintrop et al. (2016) focused on CT *practices* (e.g. data practices, modelling and simulation practices, computational problem-solving practices, and systems thinking practices), and Shute et al. (2017) used CT *facets* (e.g. decomposition, abstraction, algorithms, debugging, iteration, and generalisation). In this paper, we use the term CT *activities* to describe the tasks, practices and perspectives that are used in the primary mathematics classroom.

This systematic review aims to contribute to an increased understanding of CT in terms of operationalising CT activities in the context of primary school mathematics. When investigating how different mathematics experiences can benefit from its inclusion, Gadanidis et al.'s (2017) point of departure was that CT in education appears to be an isolated curriculum objective, rather than being integrated with existing subject areas (p. 78). They highlighted the need to understand better how it might improve mathematics education, and how this might be sustained (p. 94). In integrating CT into mathematics, several authors have pointed to the natural ways in which the disciplines complement one another (Barcelos et al., 2018; Shute et al., 2017; Sneider et al., 2014; Weintrop et al., 2016). For example, Sneider et al. (2014) created a Venn diagram of mathematics and CT, highlighting problem solving, modelling, data analysis and interpreting, and statistics and probability as common aspects. Shute et al. (2017) also described CT as being similar to mathematical thinking, involving beliefs, problem solving, and justification.

This description is similar to that of Barcelos et al. (2018), who stressed that CT in mathematics involves higher order skills such as semiotic representations, identifying patterns and building models. Teachers tend to be unfamiliar with it and, for that reason, struggle to see the connections between mathematics curricula and CT (Shute et al., 2017). Hence, teachers' lack of knowledge and experience to guide students when things go wrong or encourage deeper exploration when things go right is a challenge (Resnick et al., 2009, p. 63). Despite the fundamental and historical connections between mathematics and CT (Aho, 2012; Gadanidis et al., 2017; Weintrop et al., 2016; Wing, 2006), these connections are not necessarily accessible to generalist primary mathematics teachers.

In previous years, there have been several reviews of CT in education. For example, Grover and Pea (2013) examined the current state of the discourse on CT in K–12 education, taking Wing's (2006) rally for CT in education as their point of departure. Their review highlighted the under-investigation of CT as a medium for teaching other subjects. Lye and Koh (2014) analysed 27 available intervention studies, presenting the current trends of empirical research on developing CT through programming in K–12. Of the 27 studies, nine were carried out among K–12 students, but no link between CT and mathematics was addressed, and none of the included studies was conducted in a primary school setting. They concluded that visual programming languages were the most commonly used programming languages in K–12 education. Zhang and Nouri's (2019) review investigated K–9 students' development of pertinent skills through Scratch, concluding that only 9% of the reviewed studies experimented with the combination of Scratch and mathematics.

Reviews investigating CT in combination with mathematics are sparse, and to our knowledge, no review to date has focused explicitly on it in primary mathematics. This is not to say that the application of CT in mathematics is non-existent. For example, Hickmott et al. (2018) analysed literature from the fields of both computer science (CS) and mathematics education, with the aim of identifying peer-reviewed studies published from 2006 to 2016 that related to CT in the K–12 educational context. They sought to determine whether, and in what ways, these studies linked CT to the learning of mathematics. The authors found that many of the evaluated studies involved the use of a programming language when teaching CT; they further concluded that studies explicitly linking the learning of mathematical concepts to CT were uncommon and that reports of students' learning outcomes in mathematics were rare.

Barcelos et al. (2018) presented another literature review of studies published from 2008 to 2017, with the aim of identifying studies investigating how the relationship between mathematics and CT has been demonstrated through didactic activities at all educational levels. They found that the didactical activities utilised in the included studies were related to a wide range of mathematical contents, with the activities using diverse computational tools. Hence, they asserted that there is great potential for computational concepts and software tools to support the teaching of mathematics.

The present systematic review extends both Hickmott et al.'s (2018) and Barcelos et al.'s (2018) reviews in two ways. First, it picks up where these two stopped, as our included papers span the period of 2015 to 2021 (even though the search period spanned from 2000 to 2021). Second, in focusing both on the teaching and on the learning of CT in primary mathematics education, it extends and elaborates on both of their work. Accordingly, and because of the recent development of the inclusion of CT not only in secondary mathematics education, but also in primary mathematics education, there is a need to understand whether and how primary mathematics can benefit from the inclusion of CT, which is the overall aim of this paper.

Consequently, the following research questions guide this review:

RQ1: What CT activities are highlighted in primary mathematics research?
RQ2: How are CT activities integrated in the learning of mathematics?

This paper is organised in the following manner. We first present our methodological approach, including our search strategies, as well as inclusion and exclusion criteria. We then present our findings and conclude by discussing the implications of CT for primary mathematics education.

Methodology

A systematic review process, inspired by Shute et al.'s (2017) and Fink's (2019) methodology, was applied to provide a comprehensive understanding of the integration of CT into mathematics education.

Search Strategy

As outlined above, CT is a 21st-century skill that has received increasing attention within school curriculum design and educational research around the world (Heintz et al., 2016). An initial search for research publications related to CT was conducted to gain an overview of how CT has been addressed in recent publications. This initial search enabled the identification of several terms considered important for capturing the publications that were of interest for this literature review. The authors discussed the initial search and decided to operationalise CT by including several variations of the term.

The following list of core search terms was used in the search string: *computational thinking*, *algorithmic thinking*, *programming*, and *computer science*. Additionally, as we wanted to investigate connections between CT and mathematics, the process was repeated, and mathematics was operationalised in the following list of core search terms: *mathematics* and *math/maths*. Moreover, due to the focus of our research questions, the search terms *elementary/primary school* and *elementary/primary education* were added to the search string, capturing the targeted age span of interest for this review.

The search string was created in cooperation with an educational science librarian and was adapted to suit the different database interfaces used, ensuring that the core search terms were identical in each search. The search encompassed the text from the papers' titles, subject descriptions, keywords, and abstracts. The search was conducted among four disciplinary, topic-specific bibliographic databases through EBSCOhost – the Educational Resources Information Center (ERIC), MathSciNet, Teacher Reference Center (TRC) and Education Source – and one interdisciplinary database – Scopus; the search was limited to peer-reviewed papers published between 2000 and 2021. The search was conducted in June 2020 and was replicated in December 2021 to include late-2020 and 2021 publications. We decided to use 2000 as our starting point both because a main argument for the inclusion of CT in mathematics education is that it is a necessary 21st-century skill and because of the renewed interest in CT after Wing's (2006) appeal.

Inclusion and Exclusion Criteria

The inclusion and exclusion criteria of this study are described in Table 1. We decided to use search terms in the English language only, and an abstract in English

Table 1 This study's inclusion and exclusion criteria

Type of criterion	Inclusion	Exclusion
Type of publication	Peer-reviewed journal papers Peer-reviewed conference papers	Reports Dissertations Books/book chapters[1]
Publication period	2000–2021	Prior to 2000
Type of study	Empirical studies with research question(s)	Literature reviews Theoretical studies
Research method	Qualitative studies Quantitative studies Mixed-methods studies	Missing methods section
Abstract language	English	Any language other than English
Paper language	English	Any language other than English
Educational level	Primary/elementary education (i.e. students aged 6–12)	Pre-school/kindergarten education Secondary education Higher education Special education
Setting	School context	Summer school
Object of study	Students (engaging in activities) Teachers (orchestrating activities) Lesson plans (showing activities)	Teachers' and students' perceptions/beliefs/knowledge[2]
Key term in the title or abstract	Computational thinking Programming Algorithmic thinking Computer science Mathematics Math/Maths Primary school/education Elementary school/education	STEM (science, technology, engineering and mathematics)

[1]Books/book chapters were excluded because it is not always easy to determine whether they are peer-reviewed

[2]Studies that focused on teachers' and students' perceptions/beliefs/knowledge were excluded because the aim of this review is to contribute to an understanding of CT activities and their integration in the learning of mathematics

was required to be considered for inclusion. Publications that provided an abstract in English, but were written in a language other than English, were excluded.

Our search was divided into three selection stages. In selection stage 1, the initial search resulted in 1158 hits on Scopus and 1411 hits using EBSCOhost (i.e. ERIC, MathSciNet, TRC, and Education Source), giving a total of 2,569 studies. After the results were exported into a Microsoft Excel file, duplicates were removed manually, and 1909 studies were assessed for eligibility at the title and abstract level. The 1909 studies were screened for any that clearly did not fit the inclusion criteria. A total of 1716 studies were excluded based on title and abstract screening, utilising the criteria given in Table 1; this left us with 193 studies. Then, the coding process was done independently by all three researchers, with each using their own spreadsheet, thereby ensuring a blind review.

Papers could be labelled as "maybe" and discussed among the researchers; any ensuing disagreements were resolved through discussion. To verify reliability with respect to the selection of studies, all three authors were involved in this process. Another 118 studies were excluded after reading the papers' introductions and conclusions, leaving 75 studies for full-text examination. We created a table summarising the full set of these studies. Following Shute et al. (2017, p. 144), papers were deleted from the list for the following reasons: (a) involving a tangential focus or no focus specifically on CT and (b) being empirical papers that measured something other than CT and mathematics as the outcome. The final selection consisted of nine studies. This concluded the first selection stage.

To broaden the search, in selection stage 2, a central paper—Weintrop et al. (2016)—was selected due to the effort its authors put into defining CT for mathematics and science classrooms. A Google Scholar search revealed a list of 714 studies that had cited this paper. After duplicates from selection stage 1 were removed, 682 studies were screened by title and abstract, and by reviewing their introductions and conclusions, following the same process as that used in stage 1. The second selection stage resulted in one additional study for inclusion in our literature review.

To ensure that all suitable studies were included, in the final stage (stage 3), all reference lists in the 10 included studies were screened, searching for unidentified publications. No such publication was found. Together, stages 1 to 3 resulted in the inclusion of 10 papers. A schematic overview of the screening process is given in Fig. 1.

Data Extraction and Analysis

The analysis of the ten included studies was performed in four steps by all three authors. In the first step, a coding scheme, inspired by Popat and Starkey (2019), was applied. An Excel worksheet was designed to extract key information, including the year of publication, author, journal/conference, country, participants, programming language and artefacts, theoretical framework, aim, research questions, methods, mathematical domains, duration, results, and implications.

In the second step, we identified patterns across the 10 included studies. To do so, a re-reading of the studies was necessary in order to identify a set of vocabulary that would best capture the nature of what was done in the different studies.

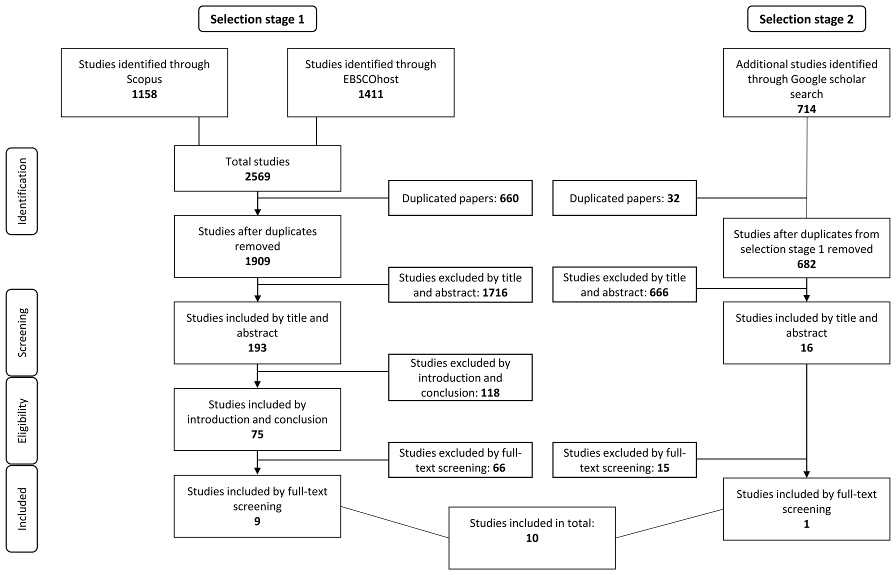
Selection stage 1
Selection stage 2
Identification
Screening
Eligibility
Included
Studies identified through Scopus 1158
Studies identified through EBSCOhost 1411
Additional studies identified through Google scholar search 714
Total studies 2569
Duplicated papers: 660
Duplicated papers: 32
Studies after duplicates removed 1909
Studies after duplicates from selection stage 1 removed 682
Studies excluded by title and abstract: 1716
Studies excluded by title and abstract: 666
Studies included by title and abstract 193
Studies included by title and abstract 16
Studies excluded by introduction and conclusion: 118
Studies included by introduction and conclusion 75
Studies excluded by full-text screening: 66
Studies excluded by full-text screening: 15
Studies included by full-text screening 9
Studies included by full-text screening 1
Studies included in total: 10

Fig. 1 The review process employed in this study

In our re-reading, we analysed not only the text, but also the figures, illustrations, and tables presented in the papers. We used established categories (i.e., number, algebra, geometry, and data and chance), taken from the large-scale international assessment Trends in International Mathematics and Science Studies (TIMSS), to categorise the mathematical content domains found in the included publications. In the same manner, we created a list of the programming languages and artefacts used, and we decided to use the label *unplugged* when no programming language/artefact was used.

The results of this second step are given in Table 2, which summarises information from the included studies and gives an overview of the school level analysed in each study, how the study was organised (i.e. whether researchers took over the teaching or worked closely with teachers), the duration of the study, the methodology, and the types of programming languages and artefacts used for coding. In addition, we have included an overview of the mathematical content domains reported in the studies and the study types of the analysed articles (i.e., case study, intervention, quasi-experimental design, or document analysis).

In the third step, we identified how the included studies integrated CT into mathematics teaching using the three categories outlined by Israel and Lash (2020): (a) no integration; (b) partial integration; and (c) full integration. In the fourth and final step, we identified how the included studies reported on the learning of mathematics. To do this, we analysed how the studies presented their findings in order to understand how the studies outlined the learning of mathematics when CT was integrated.

Table 2 Overview of the ten included studies

Study	Participants	Study organisation	Duration of CT support	Data	Programming language/artefacts	Type of study			
						Case study	Intervention	Quasi-experimental design	Document analysis
Benton et al. (2018)	Fifth- and sixth-grade students ($n = 49$); fifth-grade teachers ($n = 2$); sixth-grade teachers ($n = 2$)	Teacher-participant in professional development project	One lesson for fifth-grade and one lesson for sixth-grade students	Classroom observations (audio and field notes)	Scratch	**X**	**X**		
BartoliniBussi and Baccaglini-Frank (2015)	First-grade students ($n = 18$)	Researchers as teachers	One lesson a week for four months	Post-study test, classroom observations (video), protocols, photos, graphical productions	Bee-Bot		**X**		
Calder (2019)	Sixth-grade students ($n = 26$)	Classroom study	Two weeks (number of lessons not given)	Blogs, interviews (recorded, but type of recordings not given), classroom observations (recorded, but type of recordings not given), photos	Scratch	**X**			
Fanchamps et al. (2019)	Fifth- and sixth-grade students ($n = 62$)	Researcher as supportive teacher	Nine lessons	Pre-/post-study tests	Lego Mindstorms		**X**	**X**	

Table 2 (continued)

						Case study	Intervention	Quasi-experimental design	Document analysis
Fofang et al. (2020)	Fourth-grade students ($n = 21$)	Researcher working closely with teacher	Fifteen lessons	Classroom observation (video), interviews (notes)	Sphero robot		X		
Gadanidis et al. (2018)	Third- to sixth-grade students ($n = 415$) and teachers ($n = 19$)	Researchers collaborating in lesson planning and involved in co-teaching	Three months (number of lessons not given)	Classroom observations (field notes), photos, written reflections from teachers and students	Scratch, Google's Blockly, unplugged	X	X		
Israel and Lash (2020)	First- to fifth-grade lesson plans ($n = 47$)	Teacher participant in professional development project	Tenlessons	Document analysis	Scratch, Code.org, unplugged				X
Miller (2019)	Second-grade students ($n = 135$)	Researcher as teacher	One lesson a week for 6 weeks	Pre-/post-study tests, classroom observations (video)	Scratch		X	X	
Rodríguez-Martínez et al. (2020)	Sixth-grade students ($n = 47$)	Researcher as teacher	One lesson a week for 8 weeks	Pre-/post-study tests	Scratch, unplugged		X	X	
Sáez-López et al. (2019)	Sixth-grade students ($n = 93$)	Classroom study	No information about duration provided	Pre-/post-study tests, participant observations	mBot	X	X	X	

Findings

The 10 publications included in this systematic literature review were published from 2015 to 2020 (one in 2015, two in 2018, four in 2019 and three in 2020), revealing a recent increase of focus on the inclusion of CT in mathematics classrooms. Geographically, we found that CT in the context of primary mathematics education has been investigated across several countries: Australia (1); New Zealand (1); the USA (2); Canada (1); Italy (1); the UK (1); the Netherlands (1); and Spain (2). We notice that no Asian or African countries are represented in our overview. This may be due to how we limited our search to include English-language texts only. Hence, while this does not necessarily show the full picture of CT in mathematics, it helps give an impression of how prevalent it is worldwide to include CT in primary mathematics education (see Table 3).

With regard to mathematics content domains, in line with Hickmott et al. (2018), we found that all but one of the domains (i.e. data and chance) were prominent in (at least) one of the papers, with geometry and numbers being the most frequently addressed. This is no surprise, considering that these are the main topics of primary school mathematics. Moreover, one additional reason why CT activities tend to be anchored in geometry and numbers may be the widespread use of Scratch (see Table 2). Six of the studies included in this review used Scratch, compared with just one that used Bee-Bot, one Lego Mindstorms, one a Sphero robot, one mBot, one Google's Blockly, and one Code.org. In Scratch, coordinate systems and angles (i.e. geometry) and operators that are used to perform numerical manipulations (i.e. numbers) are embedded in the block commands (Brennan & Resnick, 2012).

This Findings-section is divided into two parts. First, we focus on the CT activities that are highlighted in primary mathematics research, providing an answer to

Table 3 The ten-included studies' locations (country and content domains)

Study	Country	Mathematics content domain(s)
Benton et al. (2018)	United Kingdom	Numbers
BartoliniBussi and Baccaglini-Frank (2015)	Italy	Geometry
Calder (2019)	New Zealand	Geometry
Fanchamps et al. (2019)	The Netherlands	Numbers
Fofang et al. (2020)	United States	Numbers
Gadanidis et al. (2018)	Canada	Algebra
Israel and Lash (2020)	United States	Geometry Numbers Algebra
Miller (2019)	Australia	Algebra
Rodríguez-Martínez et al. (2020)	Spain	Numbers
Sáez-López et al. (2019)	Spain	Geometry Numbers

our first research question. Next, we focus on the integration of CT in the learning of mathematics, addressing our second research question.

CT Activities in Primary Mathematics Research

Although many studies of CT in mathematics have been conducted over the past few decades, the approaches and terms used have varied considerably. In this subsection, we outline the contours of this field in relation to primary mathematics. The confusion starts with the main term itself, as we discovered in our literature search, which had to be extended to include the terms *computer science* (CS), *programming*, and *algorithmic thinking* (AT). The difference between these terms is still not clear cut. Three studies included in this review directly referred to CT (Fofang et al., 2020; Gadanidis et al., 2018; Rodríguez-Martínez et al., 2020), while four used the term *programming* (Benton et al., 2018; Bartolini Bussi & Baccaglini-Frank, 2015; Miller, 2019; Sáez-López et al., 2019). Furthermore, the terms *programming* and AT are both used in Fanchamps et al. (2019), and the terms *coding* and CT are used in Calder (2019). Interestingly, these studies all built on the same theoretical foundation of constructionism (Papert, 1980), wherein the learners were actively engaged in creating some type of external artefact. Despite these differences in the terminology used (i.e. CT, CS, AT, or *programming*), our findings indicate that, in the analysed studies, CT activities were often approached through problem-solving.

We identified two broad categories of CT activities related to problem-solving: activities focusing on skills and process-oriented activities. Within the former, one of the main activities is that of creating instructions for a sprite or robot to follow, termed as *coding*, *developing algorithms*, or *programming*. For example, Miller's (2019) description of coding activities in a year 2 class (i.e. having the students code instructions to make a sprite in Scratch rotate 90°) was similar to Fofang et al.'s (2020) description of students creating algorithms for a robot, and to Benton's (2018) description of students programming (i.e. creating instructions) in Scratch. Within this activity, we see several structural CT activities, such as sequencing, looping, conditionals, decomposition, debugging, and abstraction. Within the latter category (i.e. process-oriented activities), we see activities that relate to the broader dimensions of CT, focusing on communication, exploration, creativity, and engagement. Our findings are structured according to these two main categories (i.e. activities focusing on skills and process-oriented activities).

Activities Focusing on Skills

Six of the studies reported using the block-based virtual interface Scratch for CT activities, while three of the studies reported using a block-based interface to give a particular robot instructions. One study, that of Bartolini Bussi and Baccaglini-Frank (2015), used a robot for which instructions were given directly via arrows placed on top of the robot. Last, three studies reported utilising unplugged CT activities for the different tasks assigned (see Table 2 for more details).

Nine of the studies reported activities focusing on creating algorithms, involving activities such as sequencing, looping, and conditionals (Benton et al., 2018; Calder, 2019; Gadanidis et al., 2018; Israel & Lash, 2020; Miller, 2019; Rodríguez-Martínez et al., 2020; Fanchamps et al., 2019; Fofang et al., 2020; Sáez-López et al., 2019). For example, in Benton et al.'s (2018) study, the students used sequencing, looping, and conditionals to create a stopwatch in Scratch, focusing on the mathematical content of place value (with the mathematics content domain being numbers; see Table 3). When creating algorithms for robots using block-based commands, students in Sáez-López et al.'s (2019) study used the block-based coding editor mBlock to create sequencing, looping, and conditionals, with the aim of learning about coordinates, integers, and negative numbers (also in the mathematics content domain of numbers).

Another approach is seen in Rodríguez-Martínez et al.'s (2020) study, where students worked on sequencing, looping, and conditionals to learn the concepts of the greatest common divisor (GCD) and the least common multiple (LCM; in the numbers category also). The researcher-as-teacher gave the students an unplugged task in order to introduce sequencing, looping, and conditionals, before transferring the same activities to Scratch. Drawing on their findings, Rodríguez-Martínez et al. claimed that introducing CT activities in an unplugged environment worked as a bridge to help students understand the CT activities "and to practice these more easily than in the real programming environment" (p. 322).

CT activities using the robot Bee-Bot can, to a certain extent, be classified as "hybrid-unplugged" CT activities, as the Bee-Bot only allows for creating instructions in a sequence through pressing different buttons on the robot. However, these activities are not fully unplugged, as they involve giving directions to a robot on the robot itself. Bartolini Bussi and Baccaglini-Frank's (2015) study of first-grade students involved creating sequences to cause a robot to move in the shape of a rectangle, first on paper and by having the students pretend to be the Bee-Bot, giving each other commands, and then by transferring the sequences to the actual Bee-Bot.

While not all of the studies directly referred to sequencing, looping, and conditionals, a re-reading of the studies, focusing on figures and illustrations, indicated that, although they were not always mentioned explicitly, there was often a focus on these CT activities. One example of this is visible in Calder's (2019) study, in which 26 students designed and built a mathematics game suitable for facilitating number understanding among their year 1 "buddies". The use of conditionals is visible in the block commands shown in Fig. 2 of the study (p. 52), but is not mentioned in the main text. In a similar manner, students in Gadanidis et al.'s (2018) study worked on creating rotations (categorised as algebra) in Google's Blockly. While the authors did not specifically refer to the type of CT activities performed when creating these rotations, the figure on page 44 of the study indicates that, in order to create the rotations, the students created sequences and loops.

Three of the studies also reported decomposition as being a key CT activity (Benton et al., 2018; Israel & Lash, 2020; Miller, 2019). Miller (2019), for example, investigated how primary school students develop mathematical patterns and structures through, among other things, decomposition. Her study underscored the importance of decomposition as a CT activity, concluding that, when writing computer codes, students began to "engage with early algebraic thinking including forming generalisations" (p.925).

Five of the studies highlighted debugging as a key skill involving identifying errors embedded in codes and developing a strategy for fixing them (Benton et al., 2018; Calder, 2019; Gadanidis et al., 2018; Israel & Lash, 2020; Miller, 2019). In Miller's (2019) study, students solved problems, attempted to fix errors, and amended codes through trial and error. However, Israel and Lash's (2020) analysis of lesson plans revealed that there was a discrepancy between teachers' intention to integrate debugging into their lesson plans and their actual instructional practices. As a method, debugging is a skill in which mathematics teachers are well versed and which demands that students and teachers know not only how to approach and solve a problem but also how to find mistakes—a skill that is essential, but nonetheless challenging.

Three studies highlighted the importance of using abstraction (Calder, 2019; Gadanidis et al., 2018; Miller, 2019). Gadanidis et al. (2018) and Miller (2019) suggested that abstraction is important not only in CT, but also in mathematics, as it gives learners the opportunity to engage in a more complex mathematical understanding. Gadanidis et al. (2018) exemplified this by illustrating how students wrote code to abstract the concept of the rotation symmetries of a square (p. 38), and how this "simple" code could be transferred to other regular polygons. Through recognising patterns in the codes that they had developed, students in Miller's (2019) study were required to simplify and shorten their own codes through looping, as a stepping-stone towards abstraction. In much the same manner, students in Calder's (2019) study simplified and refined a game that they had developed, helping them to understand the processes that could bring about the desired outcome. From there, they could adjust the generalised code to a more specific and refined code.

Process-Oriented Activities

Our findings indicate that CT activities in the primary mathematics classroom do not focus only on coding skills, but also on broader, process-oriented activities. We observed these activities such as communication, creativity, exploration, and engagement with mathematical ideas in five of the analysed studies (Calder, 2019; Benton et al., 2018; Gadanidis et al., 2018; Bartolini Bussi & Baccaglini-Frank, 2015; Fofang et al., 2020; Miller, 2019). Calder (2019), for example, described how students reflected on and modified their approaches to the codes they had created when given feedback from the programme, fellow students, and the teacher. He found that the communication and responses the students received developed their engagement with and reflection on their work, which prompted them to use creative and innovative approaches to address issues that they encountered when coding (p. 55).

Another example can be found in Benton et al.'s (2018) study, which focused on exploration. They found that, when students got the opportunity to explore in the programme, some of them took the initiative to explore the mathematical ideas embedded in the task. In the same manner, Bartolini Bussi and Baccaglini-Frank (2015) worked with 18 students, who discussed and planned possible paths for a Bee-Bot to take in order to create a rectangle. They argued that this phase of planning gave the students the opportunity to explore space and constitute a large experimental base from which to "study" plane figures, which contributed to placing the students in the centre of their own learning process.

Another example of student exploration can be found in Fofang et al.'s (2020) study, in which 21 students programmed robots to complete an obstacle course on a carpet. For the robot to complete the course, the students needed to plan and calculate the length of each course segment from a reference sheet provided by the teacher, before trying it out on the carpet. The planning phase helped the students explore and interact with the codes in the programme and demonstrated, "an understanding of covariation and proportionality mediated by the task of programming the Sphero [robot]" (p. 4). In addition, the study highlighted the value of having students work in groups, as the collaboration and communication process made the students go back and forth between considering the reference sheet used in the planning and the actual obstacle course on the carpet.

Similar examples are found in Gadanidis et al.'s (2018) study, where students explored possible geometric transformations through what the authors described as the use of "agency", "access", "surprise", and "audience" (pp. 36–37). In using these terms, they underscored the importance not only of CT activities such as sequencing and looping but also of empowerment, through what can be viewed as a playful and exploratory environment. They also highlighted several important aspects of process-oriented activities—namely, students' opportunities to act as "active agents in their learning and the freedom to make choices, investigate and discover" (p. 36), as well as to have coding experiences with room for exploration while observing,"immediate feedback and potentially surprising results leading to conceptual insights" (p. 39).

They concluded that, when teachers highlight these kinds of activities, students engage with mathematics at a higher level. While we find that the most prevalent CT activities are those focusing on programming skills, including in the primary mathematics classroom, it is also important to investigate how tasks that prepare students for the inclusion of such activities in mathematics are integrated. In the next sub-section, we present the results of our analysis that relate to our second research question and, hence, to understanding more about how, if at all, CT activities are integrated into the learning of mathematics.

Integration of CT in the Learning of Mathematics

It is commonly accepted that CT and mathematics have a fundamental connection (Aho, 2012; Papert, 1980; Weintrop et al., 2016; Wing, 2006). Israel and Lash (2020) described the relationship between them as reciprocal. They investigated the extent to which they were interconnected within 47 lesson plans across grades 1 to 5, and their analysis revealed three types of lessons: those with no integration of CT (46.8%); those with a partial integration of CT (29.7%); and those with a full integration of CT (23.4%). The lessons with no integration typically served the purpose of pre-teaching CT skills in preparation for more integrated lessons, while in the lessons with partial integration, the mathematics content was emphasised, with CT built in as a way for students to demonstrate their understanding of mathematics concepts. In the last type of lesson plan, that featuring full integration, mathematics was taught directly through CT activities.

In drawing attention to the different ways in which the included papers reported on the integration of CT in the learning of mathematics, the categories developed by Israel and Lash (2020) are helpful, and we utilise them here to systematise the findings of the included studies. Due to the nature of this review, the category of no integration will, for obvious reasons, not be explored in further detail.

Partial Integration

According to our analysis, five of the included studies fell within the category of partial integration (see column 7 in Table 2). Calder (2019), for example, investigated how year 6 students ($n = 26$) using Scratch managed to design and build mathematics games that were suitable for their year 1 "buddies" (p. 50). In this way, Scratch was used to demonstrate students' understanding of different mathematical concepts in geometry. However, even if the participants drew on problem-solving activities in geometry when developing their games, Calder was uncertain about the extent to which new mathematics learning occurred during this process, which underlines that, in this study, the CT activities were only partially integrated with the learning of mathematics.

This was also the case in Fanchamps et al.'s (2019) study. By implementing an experimental research design, they found that the thirty-three fifth- and sixth-grade students in their experimental group tended to apply more algorithms, construct more correct algorithms, and solve more difficult algorithms when solving mathematical grid diagrams using programming in a Lego robotics context, compared with the twenty-nine students in their control group. However, the authors only partly confirmed that applying Lego robotics led to more advanced AT in students. As such, the programming context was used to reinforce AT in students, which puts these CT activities in the category of partial integration.

While the two studies above were unable to confirm that the integration of CT resulted in learning mathematics more effectively, Sáez-López et al.'s (2019) study revealed more positive outcomes. They reported on the advantages of including visual programming languages and robotics in the learning of mathematics. The 93 sixth-grade students involved in their study were set the task of creating codes to control the movements of an mBot, as the authors asserted that this task reinforces students' understanding of co-ordinates and whole numbers. They found statistically significant improvements in the students' understanding of mathematical concepts and in their acquisition of computational concepts. In this way, the students created scenarios in the programming context in which they could demonstrate their mathematical understanding, indicating that this study aligned with Israel and Lash's (2020) description of partial integration.

The same was true of Benton et al. (2018), who investigated a two-year intervention that sought to understand better how students can engage with and express important mathematical ideas through computer programming. The study reported on 99 fifth- and sixth-grade students exploring place values in two-digit numbers. The authors explored how to exploit programming functionalities in Scratch in ways that allowed students "to play with the ideas directly rather than simply learning about them" (p. 69), revealing partial integration. Their findings showed that the majority of pupils were able to engage in challenging programming concepts and to use them to explore in a mathematical context; the findings also showed how difficult it is to move the students beyond procedural knowledge. Nevertheless, the authors concluded that, with carefully designed and sequenced learning tasks and appropriate teacher support, such use of Scratch can allow students to engage with difficult mathematical ideas in new, meaningful, and generalisable ways (p. 68).

The fifth study in this category is that of Rodríguez-Martínez et al. (2020). They investigated whether the use of Scratch during mathematical instruction can have a significant effect on the mastery of sixth-grade students in the resolution of problems involving the use of the LCM and GCD. Their quasi-experimental design enabled them to observe how Scratch improved the proficiency of the students in the experimental group in solving word problems related to LCM and GCD; no significant improvement occurred in the control group. This showed how the CT programming language Scratch worked as a way for the 47 involved students to demonstrate and improve their understanding of mathematical concepts, indicating partial integration.

Mix of Partial and Full Integration

Fofang et al. (2020) worked closely with a mathematics teacher when developing two activities designed to integrate CT into the teacher's fourth-grade mathematics classroom. In the first activity, the 21 fourth-grade students used their previous knowledge of prime numbers to programme a Sphero robot to navigate a chart. It was thought that the use of prime numbers as the context in which the task given would help deepen the students' mathematical engagement—and it did. The authors observed how one task related to prime number paths prompted the students to refer to textbook definitions to search for strategies that could be used to determine whether a given number was prime or not.

Because mathematics and CT were both present, but played parallel roles in this task, we viewed it as a partial integration in the learning of mathematics. In the second activity (which is also described in the sub-section about process-oriented activities), the students were given a drawing of an obstacle course that they had to navigate on a carpet by programming a Sphero robot. Here, mathematics and CT were given proportional attention and described as mutually supportive, revealing its full integration into the learning of mathematics. The authors reported that the 21 fourth-grade students engaged in mathematics learning, while a number of CT activities were also on display.

Full Integration

Three studies showed how the full integration of CT is possible into the learning of mathematics. Bartolini Bussi and Baccaglini-Frank (2015) carried out a 4-month experiment in a first-grade classroom with 18 students. In their study, in which the researchers acted as the teacher, the overarching aim was, "[to] sow the seeds for a mathematical definition of rectangles that includes squares" (p. 391) by programming a robot to move along paths and to look at these paths as boundaries of figures with sets of geometric characteristics. By doing this, the students were met with definitions of rectangles through the use of a Bee-Bot. This situated the task in the category of full integration, where mathematics is taught directly through CT activities. However, a post-study test revealed that the squares were not seen as special rectangles by the students; hence, the study was unable to draw any clear conclusions about whether the CT activities contributed to the introduction of geometric shapes to students, or whether there was any value added by the task assigned. Nevertheless, the nature of the task was consistent with full integration.

Miller (2019) investigated how a small intervention group ($n = 40$) of year 2 students undertaking coding lessons for six weeks could benefit from engaging with CT activities in their learning of mathematics. Acting as the teacher herself, the researcher had the students talk about what they saw and discovered in the patterns they were assigned to code using Scratch. This approach prompted the students to articulate the patterns they saw (and even to move to more generalisable features). It provided them with opportunities to think in more abstract ways about mathematics and, hence, to move to more abstract notations in mathematics. The success of this study demonstrates the benefits of a full integration of CT activities in the learning of more advanced mathematics.

Gadanidis et al. (2018) differed from the other studies in this category—and in our review as a whole—in how they engaged in designing mathematical experiences over time to present ideas related to group theory to students, as this study employed a combination of unplugged and CT programming languages and artefacts (Scratch and Google's Blockly). They grounded their reasoning for focusing on group theory as a concept for young children on its fundamental link to symmetry. (For those unfamiliar with the concept, group theory, as an algebraic concept, is most often introduced after the initial calculus courses at universities). They also wondered whether it is possible to design mathematical experiences for young mathematicians that bridge the transition from understanding symmetry (as an everyday and apparently obvious concept) to understanding the more abstract and complex ideas of symmetry and group theory. Their intention was not to teach group theory to students in grades 3 to 6, but rather to design mathematical experiences to help them develop concepts related to group theory, and to do this without losing the focus on mathematics.

Four principles (i.e. agency, access, surprise, and audience) guided their collaborative work with nineteen teachers and their students in grades 3 to 6 ($n = 415$) in a year-long, classroom-based research project. Their analysis showed how this approach made teachers, "less fearful to go beyond the curriculum [and how the teachers were surprised that,] "some of [the students] could go beyond what [the teachers] were showing them" (p. 47). In the same way, the parents were impressed that their children could engage in mathematics, "using a computer program and [get] excited about maths" (p. 47). The pupils' own reflections were connected to what they learned (e.g. "all the different types of symmetries"[p. 47]), what they felt (e.g. it "felt good because we learned new things" [p. 47]), and what surprised them (e.g. "how many reflections and rotations you can do with a square piece of paper" [p. 47]). In their project, CT was a tool for participating in advanced mathematics, and as such, this study was categorised as an example of the full integration of CT in the learning of mathematics.

Discussion and Conclusions

In systematically reviewing the use of CT in primary mathematics, this review contributes to the body of knowledge exploring why mathematics appears to be a natural area for the inclusion of computational thinking. Furthermore, this review adds insight to the understanding of whether and how primary mathematics can benefit from its inclusion of CT.

Categorising CT activities as either skills-focused or process-oriented highlights what teachers and researchers-as-teachers currently focus on. The majority of the studies (nine) tended to focus on creating algorithms, using activities such as sequencing, looping, conditionals, debugging, decomposition, and abstraction. In earlier attempts to address the ambiguity of how CT in education should look, these skills were found important, particularly in the areas of programming and coding, and they were often used to introduce it in education (Lye & Koh, 2014). Moreover, sequences, loops, and conditionals are vital skills regardless of the programming language being used (Zhang & Nouri, 2019).

However, our analysis of the included studies adds to this picture and shows that the process-oriented studies had an intrinsic value to add, as students' explorations with code and reflections on these explorations contributed to their engagement with, and appropriation of, mathematics on their own terms, as the studies by Miller (2019) and Gadanidis et al. (2018) showed. Approaching mathematics using both categories of CT activities allows for a combination of the Papertian constructionist approach, in which children learn mathematics through discovery and engagement in computer activities (Papert, 1980), and Wing's (2006) emphasis on "thinking like a computer scientist" (p. 33).

Despite Grover and Pea's (2013) criticism of unplugged activities as keeping students away from computational experiences, the role of unplugged activities must be given more consideration. One role of unplugged CT activities (e.g. sequencing) that our review highlights that of connecting everyday activities to CT, as a prequel to its introduction into mathematics; this is illustrated in the studies by Gadanidis et al. (2018), Sáez-López et al. (2019), and Fofang et al. (2020). Weintrop et al. (2016, p. 139) similarly stressed that computer programming is important for students to learn when working with CT, because they develop the ability to encode instructions in such a way that a computer can execute them, which is a powerful skill that can also be applied when solving mathematical problems.

Hickmott et al. (2018) concluded that full integration of CT in mathematics is complicated. There are hindrances that make fully integrated approaches far from straightforward; as Shute et al. (2017) reminded us, teachers tend to be unfamiliar with CT and, for that reason, struggle to see the connections between it and the learning of mathematics. This might provide an explanation to why most studies report a focus on skill-oriented activities at the expense of process-oriented activities. In this review, three studies set forth examples of the full integration of CT in mathematics. In each of these studies, the researchers worked closely with the teachers: in the studies by Miller (2019) and Bartolini Bussi and Baccaglini-Frank (2015), the researchers acted as teachers, while in that of Gadanidis et al. (2018), the researchers collaborated in the lesson planning and were involved in co-teaching. This researcher-supported implementation was, perhaps, necessary, keeping in mind Resnick et al.'s (2009) finding that teachers' lack of knowledge and experience in guiding students and encouraging deeper exploration in the area of CT is a challenge. This, in turn, points to a need for upskilling initiatives for teachers (Benton et al., 2018; Gadanidis et al., 2018; Miller, 2019; Rodríguez-Martínez et al., 2020; Sáez-López et al., 2019).

Our review identified an additional challenge in the examples of the full integration of CT in the learning of primary mathematics: all of the research projects used, to some extent, more advanced mathematical principles than are normally taught at their respective grade levels. In Bartolini Bussi and Baccaglini-Frank (2015), they "sow[ed] the seeds for a mathematical definition of rectangles that includes squares" (p. 391), something that is certainly beyond what is expected of most first-grade students. In Miller (2019), the students were provided with opportunities to think in more abstract ways about mathematics than they normally would and to engage with more abstract mathematical notations. Lastly, in Gadanidis et al. (2018), we see the most extreme example of accelerated mathematics, as students in grades 3 to 6 were introduced to the abstract and complex ideas of symmetry and group theory. Taken together, this indicates that full integration may still seem beyond reach for most teachers. However, the examples are promising and worth expanding in future initiatives.

Our systematic review on the inclusion of CT in primary mathematics education identified some directions for future research initiatives. First, since we found limited use of process-oriented activities in primary mathematics classrooms, we suggest that more research is needed both to learn more about the benefits following such approaches and about how teachers can approach such process-oriented activities (perhaps as a resource to help the up-skilling initiatives requested by us and by, for instance, Benton et al. (2018), Gadanidis et al. (2018) and Miller (2019)). Second, an interesting finding is how the studies with full integration tended to address more advanced mathematics than normally taught at their respective grade levels. As such, future initiatives need to take this into consideration: How can CT activities be utilised to stimulate interest in mathematics, whether accelerated or otherwise? And how can mathematics be utilised to foster computational thinking? These are both interesting questions that future mathematics teaching would benefit from knowing more about.

Finally, one concern we detected during this review was whether the findings reported can be attributed to the type of study conducted, as studies using quasi-experimental design with associated analyses tend to give more striking results, while the other studies' conclusions were somewhat not clear (Bartolini Bussi & Baccaglini-Frank, 2015), not striking (Benton et al., 2018), vague (Fanchamps et al., 2019), or uncertain of the degree of mathematics learning that occurred (Calder, 2019). Even if different methodological approaches were not in the centre of attention in this review, we underscore the need for following up investigations into various intervention approaches, as well as the need for different methodological approaches.

Author Contribution Not applicable.

Funding Open access funding provided by OsloMet - Oslo Metropolitan University.

Availability of Data and Material Not applicable.

Code Availability Not applicable.

Declarations

Ethics Approval Not applicable.

Consent to Participate Not applicable.

Consent for Publication Not applicable.

Conflict of Interest/Competing Interests The authors declare no competing interests.

References

Ackermann, E. (2001). Piaget's constructivism, Papert's constructionism: What's the difference. *Future of Learning Group Publication, 5*(3), 438–449.

Aho, A. (2012). Computation and computational thinking. *The Computer Journal, 55*(7), 832–835.

Barcelos, T., Muñoz-Soto, R., Villarroel, R., Merino, E., & Silveira, I. (2018). Mathematics learning through computational thinking activities: A systematic literature review. *Journal of Universal Computer Science, 24*(7), 815–845.

Bartolini Bussi, M., &Baccaglini-Frank, A. (2015). Geometry in early years: Sowing seeds for a mathematical definition of squares and rectangles. *ZDM: The International Journal on Mathematics Education*, *47*(3), 391–405.

Benton, L., Saunders, P., Kalas, I., Hoyles, C., & Noss, R. (2018). Designing for learning mathematics through programming: A case study of pupils engaging with place value. *International Journal of Child-Computer Interaction, 16*, 68–76.

Bocconi, S., Chioccariello, A., & Earp, J. (2018). *The Nordic approach to introducing computational thinking and programming in compulsory education. Report prepared for the Nordic@BETT2018 Steering Group.* (https://doi.org/10.17471/54007)

Brennan, K., & Resnick, M. (2012). *New frameworks for studying and assessing the development of computational thinking*. *Paper presented at the 2012 Annual Meeting of the American Educational Research Association.* Vancouver, BC: AERA. (https://web.media.mit.edu/~kbrennan/files/Brennan_Resnick_AERA2012_CT.pdf)

Calder, N. (2019). Using Scratch to facilitate mathematical thinking. *Waikato Journal of Education, 23*(2), 43–58.

Fanchamps, N., Slangen, L., Hennissen, P., & Specht, M. (2019). The influence of SRA programming on algorithmic thinking and self-efficacy using Lego robotics in two types of instruction. *International Journal of Technology and Design Education, 31*(2), 203–222.

Fink, A. (2019). *Conducting research literature reviews: From the internet to paper*. Sage Publications.

Fofang, J., Weintrop, D., Walton, M., Elby, A., &Walkoe, J. (2020). Mutually supportive mathematics and computational thinking in a fourth-grade classroom. In M. Gresalfi& I. Horn (Eds), *The Interdisciplinarity of the Learning Science* (14th International Conference of the Learning Sciences) (vol. 3, pp. 1389–1396). Nashville, TN: International Society of the Learning Sciences.

Gadanidis, G., Clements, E., & Yiu, C. (2018). Group theory, computational thinking, and young mathematicians. *Mathematical Thinking and Learning, 20*(1), 32–53.

Gadanidis, G., Hughes, J., Minniti, L., & White, B. (2017). Computational thinking, grade 1 students and the binomial theorem. *Digital Experiences in Mathematics Education, 3*(2), 77–96.

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

Heintz, F., Mannila, L., &Farnqvist, T. (2016). *A review of models for introducing computational thinking, computer science and computing in K–12 education*. Paper presented at the 2016 IEEE Frontiers in Education Conference. Erie, PA: IEEE. (https://ieeexplore.ieee.org/document/7757410)

Hickmott, D., Prieto-Rodriguez, E., & Holmes, K. (2018). A scoping review of studies on computational thinking in K–12 mathematics classrooms. *Digital Experiences in Mathematics Education, 4*(1), 48–69.

Hsu, T.-C., Chang, S.-C., & Hung, Y.-T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers & Education, 126*, 296–310.

Israel, M., & Lash, T. (2020). From classroom lessons to exploratory learning progressions: Mathematics + computational thinking. *Interactive Learning Environments, 28*(3), 362–382.

Kurland, D., Pea, R., Clement, C., & Mawby, R. (1986). A study of the development of programming ability and thinking skills in high school students. *Journal of Educational Computing Research, 2*(4), 429–458.

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

Miller, J. (2019). STEM education in the primary years to support mathematical thinking: Using coding to identify mathematical structures and patterns. *ZDM: Mathematics Education*, *51*(6), 915–927.

Noss, R., & Hoyles, C. (1996). *Windows on mathematical meanings: Learning cultures and computers*. Kluwer Academic Publishers.

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

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

Pea, R., & Kurland, D. (1983). *On the cognitive prerequisites of learning computer programming* (Technical report #18). New York, NY: Center for Children and Technology, Bank Street College of Education. (https://web.stanford.edu/~roypea/RoyPDF%20folder/A17_Pea_Kurland_83.pdf)

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

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., & Kafai, Y. (2009). Scratch: Programming for all. *Communications of the ACM, 52*(11), 60–67.

Rodríguez-Martínez, J., González-Calero, J., & Sáez-López, J. (2020). Computational thinking and mathematics using Scratch: An experiment with sixth-grade students. *Interactive Learning Environments, 28*(3), 316–327.

Sáez-López, J., Sevillano-García, M., & Vazquez-Cano, E. (2019). The effect of programming on primary school students' mathematical and scientific understanding: Educational use of mBot. *Educational Technology Research and Development, 67*(6), 1405–1425.

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

Sneider, C., Stephenson, C., Schafer, B., & Flick, L. (2014). Computational thinking in high school science classrooms: Exploring the science 'framework' and 'NGSS.' *The Science Teacher, 81*(5), 53–59.

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

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*(1), 127–147.

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

Zhang, L., & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K–9. *Computers & Education*, *141*, (#103607).