Title	Tools and approaches for integrating computational thinking and mathematics: A scoping review of current empirical studies
Author(s)	Shiau-Wei Chan, Chee-Kit Looi, Weng Kin Ho and Mi Song Kim

Copyright © 2022 SAGE Publications. All rights reserved.

This is the accepted author's manuscript of the following article:

Chan, S.-W., Looi, C.-K., Ho, W. K., & Kim, M. S. (2022). Tools and approaches for integrating computational thinking and mathematics: A scoping review of current empirical studies. *Journal of Educational Computing Research*. Advance online publication. https://doi.org/10.1177/07356331221098793

Tools and Approaches for Integrating Computational Thinking and Mathematics: A Scoping Review of Current Empirical Studies

Abstract

The importance of computational thinking (CT) as a 21st-century skill for future generations has been a key consideration in the reforms of many national and regional educational systems. Much attention has been paid to integrating CT into the traditional subject classrooms. This paper describes a scoping review of learning tools for integrating CT and mathematics in current empirical studies published from 2015 to 2021. The review showed that most of the studies implemented CT-intensive Math-connected integration (CTIMCI). While the majority of the studies employed integrated assessments, only about half of the assessments have been validated by providing psychometric evidence of their validity and reliability. Five major types of CT tools had been identified, i.e., digital tangibles, apps and games, programming languages, formative or summative assessments, and other technological tools. In many instances, the tools also provide functions of assessment of CT skills. The most assessed CT competencies were including algorithms and algorithmic thinking, abstraction, testing and debugging, loops, and sequences. Geometry and Measurement was the most assessed mathematics topic. Our scoping review is beneficial in the investigation of the literature on CT and mathematics education, as well as guides those who are interested in developing curriculum, programs, or assessments that involve the integration of CT and mathematics.

Keywords: computational thinking; mathematics; tools; approaches; scoping review; empirical studies; assessment

For Peer Review

1. Introduction

The idea of computational thinking (CT) originated from Papert (1980) in his book "Mindstorms: Children, computers and powerful ideas." As "learning to communicate with a computer may change the way other learning takes place" (Papert, 1996, p. 6), CT has been advocated by many researchers to "forge ideas" (p.13). For example, Wing (2006) contended that "to reading, writing, and arithmetic, we should add computational thinking to every child's analytical ability" (p.33). CT is viewed as an essential ability that ought to be possessed by everyone, not just beneficial for computer scientists. Further, Wing (2010) described CT as "the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent." (p. 1). The information processing agent can be a human being, a computer, or a machine (Wing, 2006).

Other researchers and scholars came up with their definitions relative to their specific research fields, but to date, there is no consensus on an operational definition of CT yet (Hsu et al., 2018). Zhong et al. (2016) argued that the definitions of CT can be categorized into three different perspectives, namely CT is a problem-solving process, CT is a crucial form of expression, and CT is the three-dimensional CT framework by Brennan and Resnick (2012). Hoyles and Noss (2015) viewed CT as involving decomposition (solving a problem includes solving a set of smaller problems), abstraction (seeing a problem at different detail levels), pattern recognition (seeing a new problem as associated with previous problems), and algorithmic thinking (seeing tasks as to smaller linked discrete steps). These four elements are regarded as a four-stage problem-solving process in accordance with the study of Tabesh (2017). Moreover, Buitrago Flórez et al. (2017) proclaimed CT as a means of reasoning that accumulates some high-level practices and skills in the core of computing, but was pertinent to various areas far beyond computer science.

For Peer Review

There is a growing interest in integrating CT into the mathematics curriculum to provide students with a more realistic insight into mathematics and to prepare them for pursuing future careers as well as to assist them to become science, technology, engineering, and mathematics (STEM)-savvy citizens (Weintrop et al., 2016). English (2018) claimed that CT and mathematics are natural partners. Weintrop et al. (2016) and Ho et al. (2017) also declared that CT and mathematics have a mutual relationship, which is about utilizing CT to enrich mathematics and employing mathematics context to enrich CT. Hardin and Horton (2017) stated that immersing CT in the mathematics syllabus not only provides computational skills to the students but also enables the students to better comprehend mathematical concepts and skills. CT practices also can serve as an effective technique to encourage students to be involved in powerful ideas of mathematics and stimulate their mathematical habits of mind (Pei et al., 2018). Durak and Saritepeci (2018) and Durak et al. (2019) suggested that students' achievement in mathematics positively influences their CT skill levels. Hence, the instructors ought to design the appropriate curricula and learning tools related to CT practices and infuse them into the mathematics curriculum.

Nevertheless, it could be a challenge to merge the content areas of CT and mathematics while at the same time conserving the integrity of the two domains. It is hard to find a good match between CT concepts and certain mathematical content (Israel & Lash, 2019). The integrated approach to instruction may affect the design of appropriate assessments. The assessments used can be problematic as the interactions between these two disciplines may disprove the outcomes of the assessments and result in measurement interference. This is due to the outcomes in one discipline that sometimes might be easier to be measured when compared to

For Peer Review

another discipline (Bortz et al., 2020). Hence, this study intends to uncover how CT has been incorporated into the mathematics classroom from recent empirical studies.

Several reviews of CT in mathematics instruction have been published in previous years such as studies from Barcelos et al. (2018) and Hickmott et al. (2018). In these earlier reviews, there is no detailed information about the types and forms of CT tools, and the assessed CT competencies and mathematics topics in the integrated instruction. This current scoping review seeks to fill in the missing literature. It is important to explore these research gaps as earlier studies such as Berkaliev et al. (2014) emphasized the power of computational tools in solving problems in mathematics that cannot be done manually. The usage of computational tools could improve the students' performance in mathematics. diSessa (2018) also stressed that the application of computational tools affected the students' experience in learning mathematics. Thus, this paper aims to examine the integration of CT into mathematics instruction, provide a more organized view of the use of CT tools and approaches in mathematics education, and determine the CT competencies and mathematics topics being assessed in the integrated instruction. Three research questions stated below drive this study:

(1) How has CT been integrated into mathematics instruction in recent empirical studies?

(2) What are the types and forms of CT tools being applied in the integrated instruction?

(3) What are the CT competencies and mathematics topics being assessed in the recent empirical studies?

2. Literature Review

Several scholars and researchers discussed the inclusion of CT in tandem with mathematics, but their ideas on the integration of CT into the mathematics classroom varied from

For Peer Review

each other. For instance, Barr and Stephenson (2011) enumerated nine main CT concepts and capabilities that could be incorporated into mathematics instruction, namely data collection, data analysis, data representation, abstraction, problem decomposition, algorithms and procedures, simulation, parallelization, and automation. According to Sneider et al.'s (2014) Venn diagram model, the common factors of CT and mathematics included analyzing and interpreting data, modeling, problem-solving, and statistics and probability. Meanwhile, the non-overlapping factors of CT were robotics, programming, networking, and simulation whereas the non-overlapping factors of mathematical thinking were algebra, calculus, arithmetic, and set theory. Barcelos and Silveira (2014) identified three transferrable high-order skills between CT and mathematics: (a) alternating between different semiotic representations, (b) establishing relationships and identifying patterns, and (c) building descriptive and representative models. Weintrop et al. (2016) developed a model that integrates mathematics and CT, resulting in four major categories of data practices, modeling and simulation practices, computational problem-solving practices, and systems thinking practices.

To incorporate CT and mathematics, Waterman et al. (2018) proposed three levels of CT integration: exist, enhance, and extend. CT skills, practices, and concepts that already exist in the lessons can be explained with instances of how they can be linked to technology. For example, the existing activity of using physical models to comprehend certain phenomena is connected to CT concepts. Besides, extra lessons or tasks are added to enhance the disciplinary concept and come up with a clear link to computing concepts. For instance, the students are required to collect data on their own, construct a visual representation manually, and analyze the data. New lessons or series of lessons extend the disciplinary concepts as a foundation for computer science exploration, usually involving programming tasks. For example, the students utilize a computer

https://mc.manuscriptcentral.com/joecr

simulation and alter underlying code or variables to explore how dynamic systems change over time.

Israel and Lash (2019) also put forth three levels of integration for CT and mathematics, namely no integration, partial integration, and full integration. No integration refers to the lessons of CT or mathematics being taught separately in the classroom. In the lesson, the teachers were either focused on CT or mathematics, for example, students were introduced to the Scratch blocks “Ask a question and wait” and “Answer” in the Sensing tab in the unplugged CT lesson. Meanwhile, partial integration means CT is employed to strengthen mathematics, or mathematics is employed to strengthen CT in the lessons. By adapting Kiray’s (2012) model, if CT outcomes are more dominant than the mathematics outcomes, then the instruction is regarded as CT-intensive Math-connected Integration (CTIMCI); if mathematics outcomes are more dominant compared to CT, then the instruction is viewed as Math-intensive CT-connected Integration (MICTCI). For instance, students were allowed to demonstrate their comprehension of fractional parts through a CT activity in which students design their own fractional part stories with animations in Scratch. The mathematics content was greatly focused in the lesson and CT activity was built into it. Full integration is the content of both CT and mathematics are taught together or the affordances of CT or mathematics are employed to teach the other domain. For example, students would learn about different polygons by animating them in Scratch. In the lesson, the students would walk the shapes of the polygons and then code those shapes. In this case, the teachers taught mathematics concepts directly through CT activities rather than teaching mathematics lessons, and then had students demonstrate understanding through CT (Israel & Lash, 2019).

For Peer Review

There were several recent reviews conducted on the integration of CT and mathematics. A recent review study implemented by Nordby et al. (2022) analyzed ten empirical studies to present an overview of the existing literature from 2000 to 2021 on CT activities in primary mathematics education and to explain how they are integrated into primary mathematics teaching and learning. They found two types of CT activities in the primary mathematics research, i.e., activities focusing on skills (for instance abstraction, decomposition, looping, debugging, conditionals, and sequencing) and process-oriented activities (creativity, communication, engagement, and engagement). In the reviewed studies, there were five studies put under the category of partial integration, one study put under the category of a mix of partial and full integration, and three studies put under the category of full integration.

Kallia et al (2021) conducted a systematic literature review to determine what characterizes CT in mathematics education and which CT characteristics can be addressed within the subject. They conducted a Delphi study that gathered the opinions of 25 math and computer science experts on the possibilities for addressing CT in mathematics education. The Delphi study's findings, which support the literature review's conclusions, indicate three key parts of CT that need to be addressed in mathematics education, namely, problem solving, cognitive processes, and transposition.

Another review study was performed by Hickmott et al. (2018) by recognizing peer-reviewed articles on CT in K-12 educational settings that were published between 2006 and 2016 and figuring out how these research-related CT to mathematics learning. The results indicated that the majority of the studies: (1) originate from computer science academics rather than education experts, (2) involve mathematics but primarily focus on teaching programming skills, (3) present small-scale research designs on self-reported attitudes or beliefs, and (4) seldom deal

For Peer Review

with concepts in mathematical domain areas such as statistics, probability, functions, and measurement. Barcelos et al. (2018) conducted a review of the articles published between 2008 and 2017, to identify studies that looked at how the relationship between mathematics and CT has been proven through didactic activities at all levels of education. They discovered that the didactical activities used in the studies were related to a wide range of mathematical concepts and that the activities were carried out utilizing a variety of computer tools.

3. Method

Scoping review in this study was intended "to summarize and disseminate research findings" and "to identify research gaps in the existing literature" (Arksey & O'Malley, 2005, p. 6-7). This scoping review was conducted using the five framework stages of Arksey and O'Malley (2005), i.e. (a) identifying the research questions; (b) identifying relevant studies; (c) study selection; (d) charting the data; and (e) collating, summarizing and reporting the results. Six databases were utilized as the search engines to identify the relevant studies, namely *Scopus*, *Web of Science*, *Science Direct*, *SpringerLink*, *Taylor & Francis Online*, and *ERIC (Education Resources Information Center)*. The keywords performed for the document search in any part of the article including title, abstract, keywords, and full text, were: "computational thinking" AND ("mathematics" OR "math"). As Table 1 shows, the total search results were 2233 documents including Scopus (491), Web of Science (363), Science Direct (341), SpringerLink (584), Taylor & Francis Online (297), and ERIC (157).

-Insert Table 1 here-

Subsequently, three-round of screening was conducted for study selection to filter out the inappropriate articles as demonstrated in Figure 1. For the first round of screening, the duplicates

https://mc.manuscriptcentral.com/joecr

For Peer Review

of the articles (n = 336) and the articles without full text (n = 23) were excluded. Hence, a total of 1874 articles were filtered for the screening for relevance in the second round. The irrelevant articles (n = 743) were removed in the second round of screening. Then, a total of 1131 articles were filtered for exclusion in the third round. In the third round of screening, 1058 articles that did not meet the requirements of the inclusion criteria were eliminated. The inclusion and exclusion criteria for the scoping review comprise (a) Must be published between the year 2015 and 2021 (the last seven years); (b) Must be peer-reviewed journal articles, not conference proceedings, book chapters, and book series; (c) Must be empirical research, either qualitative or quantitative, and not just presentations of framework or literature review; (d) Must involve the integration of CT and mathematics; (e) Must be empirical studies that employed some form of CT tools; (f) Must be published in the English language. Based on all these inclusion and exclusion criteria, eventually, 73 articles are recognized to be applicable and chosen for the next stage of analysis.

Next, 64 unique CT tools were determined from the 73 articles. Each unique CT tool was coded by the CT competencies based on three CT dimensions for Brennan and Resnick's (2012) framework, i.e., computational concepts, computational practices, and computational perspectives. Some of the sub-categories of CT competencies were adapted according to the frameworks of Zhong et al. (2016), Grover and Pea (2018), and the 3D Hybrid CT Framework (Adams et al., 2019). During the review process, two researchers compared and classified all the 73 articles as well as tabulated them into them according to the predefined criteria, namely year, authors, level of integration, separated/integrated assessments, validated/not validated, types of CT tools, forms of CT tools, CT competencies, and mathematics topics. Then, they reviewed the articles based on the aforementioned predefined criteria and worked collaboratively on the

For Peer Review

coding process. Lastly, the intercoder agreement was sought to confirm the reliability of the coding scheme (MacPhail et al., 2016) and it was 90%. The parts that had disagreements, were resolved through further review and discussion until reaching a mutual agreement.

-Insert Figure 1 here-

4. Findings and Discussions

4.1 How has CT been integrated into mathematics instruction in recent empirical studies?

In this study, 73 peer-reviewed journal articles were chosen for review. Appendix A shows how has CT been integrated into mathematics instruction in these articles, including the level of integration, separated/integrated assessments, and being validated/not validated. Figure 2 indicates the levels of integration for CT and mathematics. Most of the studies (n = 41, 56.2%) were categorized under one of the partial integrations, i.e., CT-intensive Math-connected Integration (CTIMCI). For instance, Panskyi et al. (2019) determined the impact of informal computer science education on a visual creative programming course using Dr. Scratch for students aged 9-14. Across all the sessions, their Dr. Scratch projects were mainly assessed using seven CT and algorithmic thinking dimensions including logical thinking, parallelism, abstraction, problem decomposition, user interactivity, synchronization, data representation, and algorithmic notions of flow control. Mathematics was only partially integrated as one of the 12 sessions of teaching where the students learned the basics of mathematics and geometry.

We found 31 studies (42.5%) of fully integrated CT and mathematics in the classrooms. For example, Nam et al. (2019) determined the impact of a card-coded robotics syllabus and integrated the activities of sequencing and mathematical problem solving into the kindergarten curriculum. The treatment group attended the eight-week intervention that emphasized CT components of algorithmic thinking, patterns, efficient thinking, and procedural thinking. The

For Peer Review

students were given a pretest and posttest of sequencing and mathematical problem solving before and after treatment. Therefore, the study was categorized as full integration. There was only one study (1.4%) of Math-intensive CT-connected Integration (MICTCI) conducted by Messer et al. (2018). The authors examined the influence of an educational programming intervention on mathematics skills, spatial awareness, and working memory among children aged 5 to 6 years old. A total of 41 students from a UK primary school were randomly assigned to one of three groups: programming with iPad technology, programming with paper and pencils, or a comparison condition containing pencil and paper mathematics addition and subtraction tasks. In this case, only one group was given the intervention of programming using iPad technology to reinforce mathematics, indicating partial integration (MICTCI).

-Insert Figure 2 here-

The CT and mathematics assessments in the reviewed studies were utilized in three ways as demonstrated in Figure 3: Separated, Integrated, and Separated & Integrated. The majority of the studies (n = 37, 50.7%) employed an integrated assessment method to evaluate both concepts of CT and mathematics. For instance, one of the studies performed by Miller (2019) with primary students used pretest and posttest with a focus on patterning and coding along with video-recorded instructions. There were 10 items of patterning in the pretest and posttest which included forming patterns, making predictions about patterns, determining repeating patterns, and generalizing patterns. Meanwhile, there were also 10 coding items where all patterning concepts were mapped onto coding contexts.

Furthermore, 34 studies (46.6%) utilized separated assessments to assess students' concepts of CT and mathematics. A study by Rodriguez-Martinez et al. (2019) employed Computational Thinking Test (CTt) to measure four computational concepts such as handling of

For Peer Review

events, loops or iterations, conditional sentences, and sequences in a Grade 6 classroom. The Mathematical Knowledge Test (MKT) was used as another instrument to assess the chosen standard of mathematics learning. Both tests were administered as pretest and posttest. Additionally, two studies (2.7%) involved the use of separated and integrated assessments. For example, Tran (2018) applied a pretest and posttest on CT with 10 items to identify five concepts of computer science, namely, algorithm, debugging, sequence, conditionals, and looping. Another instrument that was used was 44 items of pretest and posttest surveys with five Likert scales to evaluate enjoyment in STEM-related tasks; goals, beliefs, and aspirations in STEM; and attitudes from peers and parents about STEM. These two instruments were separated assessments. The integrated assessment that was utilized was semi-structured interviews with open-ended questions to assess the constructs related to concepts of computer science and STEM.

-Insert Figure 3 here-

Figure 4 exhibits the portion of validated or not validated assessments in the integrated instruction. It is crucial to identify whether the assessment was validated or not as the validated assessments provided the psychometric evidence including validity and reliability to make sure the appropriate reporting of accomplishments among the test-takers (McMillan et al., 2011) and to confirm that the assessments being used were measuring what it should be measuring (Lai, 2013). There were 36 studies (49.3%) that validated their assessments. For instance, in the study of Magana et al. (2016), the reliabilities for control appraisals and value appraisals of the pretest and posttest self-belief measures were 0.86, 0.90, 0.80, and 0.91 respectively. The construct validity was obtained via factor analysis. Learning measures, another assessment was validated through face validity which was carried out by three experts. On the other hand, the assessments employed in the 37 studies (50.7%) were not validated. These studies applied their assessments

For Peer Review

or modified existing assessments for their research without reporting validity or reliability indicators. Some studies asserted that the assessment they used had been validated elsewhere, but they did not validate it themselves when applied in different contexts, such as in other countries. In fact, every assessment needs to be reevaluated for its validity and reliability prior to its use (Lai, 2013).

-Insert Figure 4 here-

4.2 What are the types and forms of CT tools being applied in the integrated instruction?

4.2.1 Types of CT tools

Table 2 displays the types of CT tools used in the 73 reviewed studies. From the 73 articles, 64 unique CT tools were identified. There were five major types of CT tools, namely digital tangibles, apps and games, programming languages, formative or summative assessments, and other technological tools. Three former types were adapted from the study of Namukasa et al. (2017). Two new categories were added based on the reviewed studies, which were formative or summative assessments, and other technological tools.

-Insert Table 2 here-

As indicated in Figure 5, the majority of the types of CT tools in the selected literature were programming languages (n = 21, 32.8%). Digital tangibles and formative or summative assessments were the second major type of CT tools with a frequency of 16 (25.0%). Meanwhile, the third major type of CT tools was apps and Games (n = 7, 10.9%) and the fourth major type of CT tools was other technological tools (n = 4, 6.25%).

-Insert Figure 5 here-

For Peer Review

4.2.1.1 Digital tangibles

Digital tangibles were defined as "physical artefacts designed to trigger various digital events, potentially provide innovative ways for children to play and learn" (Price & Pontual Falcao, 2011, p. 500). Code was produced to control programmable robots or circuits (Gadanidis et al., 2018). Educational robotics is generally deemed as a relevant medium for CT skill development (Durak et al., 2019). These digital tangibles allow young students to make abstract programming ideas and concepts more visible and accessible as well as more flexible for social learning. Not only that, children are dynamically involved in problem-solving and learn powerful concepts from robotics (Bers et al., 2019). 16 out of 64 unique CT tools were classified as digital tangibles including LEGO®EV3 robotics/Lego Mindstorms EV3/NXT robots, Lego® Education WeDo 2.0 robotics kit, robots/programmable robots and circuits, Arduino, mBot robot, SRA-programming Lego NXT Mindstorms robots, TurtleBot, KIBO robotics kit, Sphero, micro:bit, Zowi, Dash & Dot, Anprino, Codey Rocky, Bee-Bot robot, and Robotics Dream ER kits.

4.2.1.2 Apps and Games

On the other hand, the category of apps and games includes games and web-based simulations. Altanis et al. (2018) claimed that the use of games can enhance CT and mathematical understanding. The application of apps also can foster CT skills as asserted by Ehsan et al. (2017). In this study, seven CT tools grouped under apps and games were Scalable Game, Kinect games, Bee-bot iPad app, SmartMeasure App, Hour of Code, Lightbot, and Digital Educational Material (DEM) "Evolution".

4.2.1.3 Programming Languages

https://mc.manuscriptcentral.com/joecr
For Peer Review

Previous research showed that the integration of programming had improved performance in mathematics (Saez-Lopez et al., 2019). In this study, the 21 CT tools categorized as programming languages included Scratch, Scratch Jr., Dr. Scratch, MATLAB, Mathematica, Thermo-Calc, COMSOL, Statistical programming language R, Python, Google's Blockly/Code.org coding problems, Sketchpad, Logo, BlockyTalky, LaPlaya, ViMAP, Visual Studio IDE, Bitbloq, mBlock, Ardublockly, Processing, and Bootstrap Algebra. In Figure 6, there were seven major kinds of programming languages in the reviewed research, i.e., block-based programming, text-based programming, spatial programming, functional programming, agent-based programming, visual basic.net programming, and object-oriented programming.

Most of the reviewed studies used block-based programming including Scratch, Scratch Jr., Dr. Scratch, Google's Blockly or Code.org, LaPlaya, BlockyTalky, Bitbloq, mBlock, and Ardublockly. Scratch was the most prevalent CT tool. The usage of Scratch enabled the students to engage with tough mathematical concepts in new, meaningful, and generalizable ways (Benton et al., 2018). By using Scratch, students were able to create computer programs without worrying about syntax or spelling in block-based coding environments (Gadanidis et al., 2018). MATLAB, Mathematica, Thermo-Calc, COMSOL, Python, and statistical programming R were text-based programming. Two functional languages employed in this study were Logo and Bootstrap Algebra. There was only one study that utilized spatial programming (sketchpad), agent-based programming (ViMAP), Visual Basic.net (vb.net) programming (Visual Studio IDE), and object-oriented programming (Processing).

-Insert Figure 6 here-

4.2.1.4 Formative or Summative Assessments

For Peer Review

Formative assessment is used to enhance teaching and learning, identify the problems faced by the students ongoing, before and during instruction, and assess what is working and what needs to be improved. Meanwhile, a summative assessment is utilized to assess learning outcomes after the instruction, and evaluate students' understanding of the material and their readiness for the next level of activity (Dixson & Worrel, 2016). There are 16 CT tools: computational cognition test, Visual Blocks Creative Computing Test (VBCCT), CT Test (CTt), CT survey, pretest and posttest on computational thinking, CT Skills Scale (CTSS), Robotics Activities Attitudes Scale (RAAS), Bebras tasks, Solve-Its assessments, VELA activities, CT Lesson Screener, CT Lesson Enhancer, CT concept test, CT practice test, CT test, and Math + C worksheets.

4.2.1.5 Other Technological Tools

Other technological tools are the Easy Java Simulator (Ejs) tool, Lattice Land, the Science Behind Pixar (Pixar) exhibition, and spreadsheet. For instance, Lattice Land is deemed as a mathematical microworld that enables the students to investigate the geometrical idea by manipulating the polygon drawn with a discrete point on a plane. This microworld serves as a powerful tool that facilitates the students' CT skills and mathematical habits of mind (Pei et al., 2018).

4.2.2 Forms of CT Tools

The forms of CT tools used in the integrated instruction are reviewed in Appendix A including assessment methods, format, cognitive/non-cognitive, and automatic/non-automatic. There were several assessment methods used in this study: in-class tasks (assignments, open-

For Peer Review

ended tasks, modules, classroom activities, etc.), knowledge tests (pretest and posttest, performance, etc.), artefact assessment (projects), survey (questionnaire), interview, and observation. Three formats of CT tools were computer (e.g., iPad, robots, Python), paper (e.g., test, survey), or both (computer and paper). The example of cognitive skills were CT skills, mathematics skills, spatial awareness, and working memory, while non-cognitive skills were self-efficacy, attitudes, and perceptions. The assessment tools that provide scoring or feedback in an automatic way to the learners were considered as automatic tools. Non-automatic tools referred to the assessment tools which unable to give scoring or feedback automatically. In Appendix A and Figure 7, it was found that most of the studies (*n* = 34, 46.6%) involved the use of both computer and paper, followed by paper (*n* = 20, 27.4%), and computer (*n* = 19, 26.0%). Furthermore, nearly half of the studies assessed cognitive skills (*n* = 38, 52.0%). The studies that assessed both cognitive and non-cognitive skills were 26 (35.6%), while the least studies assessed non-cognitive skills with the number of nine (12.3%). The CT automatic tools were Dr. Scratch (2 studies), Kinect games (1 study), and Digital Educational Material (DEM) "Evolution" (1 study). Other CT tools from the 64 studies (93.8%) were viewed as non-automatic tools, such as Scratch, Python, Arduino, and Computational Thinking Test (CTt).

-Insert Figure 7 here-

4.3 What are the CT competencies and mathematics topics being assessed in the recent empirical studies?

In this study, 64 unique CT tools have been mapped under the sub-categories of CT competencies which were organized based on the three CT dimensions of Brennan and Resnick (2012)'s framework, i.e., computational concepts, computational practices, and computational

For Peer Review

perspectives. The framework's three primary dimensions are computational concepts (the concepts designers engage with as they program, such as iteration, parallelism, etc.), computational practices (the practices designers develop as they engage with the concepts, such as debugging projects or remixing others' work), and computational perspectives (the perspectives designers form about the world around them and about themselves)" (p. 1). As shown in Table 3, some of the computational concepts and practices from Brennan and Resnick's (2012) framework have been utilized including sequences, loops, events, parallelism, conditionals, operators, data, testing and debugging, and reusing and remixing.

Furthermore, computational concepts and practices adapted from both frameworks of Grover and Pea (2018) and Adams et al. (2019) were logic and logical thinking, algorithms and algorithmic thinking, pattern recognition, abstraction, generalization, automation, and modeling. Problem decomposition from both frameworks of Grover and Pea (2018) and Adams et al. (2019) were subsumed under computational concepts. Creativity in the Grover and Pea's (2018) framework and problem solving in the Adams et al.'s (2019) framework were employed. User interactivity and creation from the framework of Adams et al. (2019) were reorganized into computational concepts and computational practices respectively. Collaborating/collaboration in the frameworks of Zhong et al. (2016), Pinkard et al. (2019), and Adams et al. (2019) was used. Expressing, connecting and questioning in Brennan and Resnick's (2012) framework were revised to expressing and questioning about technology world, and connecting to the technical world. Some new sub-categories of CT competencies have been added based on recent empirical studies such as functions, subroutines, variables, procedural thinking, efficient thinking, agency, access, experimentation, problem formulation, confidence, appraisals, sense of identity, and so forth.

For Peer Review

-Insert Table 3 here-

From Table 3, it was noticed that most of the CT tools were used to assess the algorithms and algorithmic thinking (*n* = 32). The second and most assessed CT competency was abstraction (*n* = 26), and the third most assessed CT competency was testing and debugging (*n* = 23). Loops (*n* = 22) was the fourth most assessed CT competency, while the sequences (*n* = 21) was the fifth most assessed CT competency. This was followed by decomposition (*n* = 20), patterns/pattern recognition (*n* = 17), parallelism (*n* = 15), logic and logical thinking (*n* = 14), variables/Naming (*n* = 13), and iteration/repetition (*n* = 13). The most assessed CT competencies are demonstrated in Figure 8.

-Insert Figure 8 here-

From the reviewed studies, we found that there were some mathematics topics being assessed in the integrated instruction. The five mathematics topics were modified from the study of Hickmott et al. (2018), i.e., Numbers and Operations, Algebra, Calculus, Geometry and Measurement, and Statistics and Probability. However, not all the reviewed studies revealed their mathematics topics. The results for the mathematics topics and sub-topics being assessed were displayed in Table 4. The most assessed mathematics topic was Geometry and Measurement with a frequency of 23 studies. This followed by Number and Operations (*n* = 15), Algebra (*n* = 11), Statistics and Probability (*n* = 5), and Calculus (*n* = 1).

-Insert Table 4 here-

5. Discussion and Conclusion

Our scoping review reports on the integration of CT into the mathematics classroom, the types and forms of CT tools in the integrated instructions, as well as the CT competencies being

For Peer Review

assessed in the integrated instruction. Regarding research question one, the results indicated that the majority of the reviewed studies employed CT-intensive Math-connected integration (CTIMCI) instead of full integration and Math-intensive CT-connected Integration (MICTCI). It means that the instructors or researchers tended to focus on the CT outcomes rather than the mathematics outcomes in the integrated classrooms. This result differs from those reported in the earlier studies including the studies of Israel and Lash (2019) and Nordby et al. (2022). Israel and Lash (2019) claimed that most of the lessons did not incorporate any integration between CT and mathematics, then followed by partial integration and full integration. Meanwhile, in the review of Nordby et al. (2022), they found that the majority of the studies which employed partial integration focused on math content with CT built-in as a technique to exhibit their understanding of mathematical concepts.

Furthermore, integrated assessments were utilized in most of the studies. In other words, the instructors or researchers were likely to integrate the assessments of CT and mathematics instead of separating both assessments. During the integrated instruction, the integrated assessments with specified identification of domains ought to be employed to yield a more complete picture of students' strengths and weaknesses in both domains (Bortz et al., 2020). At the same time, the students will gain benefit from the interplay of integrated domains towards achieving a deeper understanding of conceptual knowledge. Meanwhile, nearly half of the assessments in the reviewed studies were not validated which corroborates the statement of Tang et al. (2020) where many assessments lacked evidence of validity and reliability. It is crucial to develop a valid and reliable assessments to ensure the quality of the research findings (Chan et al., 2021) and allow the instructors and researchers to confidently use the assessments during the class and enhance the wide distribution of the assessments (Tang et al., 2020).

For Peer Review

For research question two, there were five main types of CT tools being utilized in the reviewed studies including digital tangibles, apps and games, programming languages, formative or summative assessments, and other technological tools. Additionally, most of the studies used CT tools that were non-automatic, assessed cognitive skills, as well as involved both computer and paper in the integrated instruction. They were consistent with the findings from the study of Cutumisu et al. (2019), except for the format of CT tools. This study mostly used both computer and paper in the integrated instruction, but Cutumisu et al. (2019) mostly involved computer only in the instruction. Several forms of assessment methods were in-class tasks, knowledge tests, artefact assessment, surveys, interviews, and observation. These findings can be instrumental for the instructors and researchers on how to design the assessments of integrated lessons in the contexts of CT and mathematics (e.g., partial integration or full integration) as well as effectively conveying the concepts of CT and mathematics in the instruction.

With regard to research question three, it was also found that algorithms and algorithmic thinking were the most assessed CT competencies, followed by abstraction, testing and debugging, loops, and sequences. Two of the CT competencies (algorithmic thinking and abstraction) were also put in the leading positions from the review of Kallia et al. (2021). Zhang and Nouri (2019) declared that loops and sequences were essential skills no matter what programming language was chosen. However, testing and debugging was not frequently highlighted in the empirical papers reviewed by Kallia et al. (2021). The findings of this study allow the teachers, scholars, and researchers to have a better understanding of the CT tools and CT competencies that can be gained in the mathematics classroom. Geometry and Measurement was the most assessed mathematics topics. This is followed by Numbers and Operations, and Algebra. This result differs from the findings of the review study from Hickmott et al. (2018)

For Peer Review

where most of the studies involved Numbers and Operations and followed by Algebra and Geometry. This scoping review has contributed to the areas of integration of CT in mathematics education. The findings of this scoping review have provided more information about CT and mathematics education on the existing literature and research themes. It can serve as a benchmark for many parties, including government, policymakers, school administrators, teachers, researchers, and scholars.

Although this scoping review involved a comprehensive review of current CT tools in the mathematics classroom, there were some limitations. First of all, this review only focused on the articles published in the last seven years, i.e., between the years 2015 and 2021. We made this decision as the area of integrated instruction for CT and mathematics has not been fully developed until recent years. Secondly, only empirical studies were used for the analysis as they had "verifiable" evidence of the reported practices of integrated instruction of CT and mathematics. Thirdly, although our search scope extends to multiple databases, we excluded the studies that did not state the CT tools and CT competencies measured. Therefore, it is suggested that future studies on CT in mathematics education should elucidate the CT tools and CT competencies being assessed, preferably with detailed descriptions including definitions.

For Peer Review

After reviewing the articles, there are some recommendations for future studies. Despite the studies done on CT and mathematics education, validated CT tools are still lacking in the mathematics classroom at different stages. Most of the CT tools did not undergo a comprehensive psychometric validation process to warrant their integration in the syllabus including the validity and reliability of the instruments. Such a situation will cause problems related to the infusion CT in the educational systems as argued by Roman-Gonzalez et al. (2019). Thus, future studies should provide more evidence of validity and reliability as it is crucial to construct universal CT tools which are appropriate to be administered across different contexts such as gender, ages and grade levels. Moreover, most of the studies tended to focus more on CT outcomes rather than mathematics outcomes. It is recommended to have a balance or full integration for both CT and mathematics in future studies. Bortz et al (2020) also proposed that if the assessments of CT and mathematics are integrated, the scoring systems should be differentiated to uncover learning in each domain including detailing the ways in which the integration alters epistemology.

For Peer Review

References

Adams, C., Cutumisu, M., & Lu, C. (2019). Measuring K-12 computational thinking concepts, practices and perspectives: An examination of current CT assessments. In *Proceedings of the Society for Information Technology & Teacher Education (SITE)* (pp. 18–22). Las Vegas, NV: March.

Altanis, I., Retalis, S., & Petropoulou, O. (2018). Systematic design and rapid development of motion-based touchless games for enhancing students' thinking skills. *Education Sciences*, 8(1).

Arfe, B., Vardanega, T., Montuori, C., & Lavanga, M. (2019). Coding in primary grades boosts children's executive functions. *Frontiers in Psychology*, 10, 2713.

Arksey, H., & O'Malley, L. (2005). Scoping studies: Towards a methodological framework. *International Journal of Social Research Methodology: Theory and Practice*, 8(1), 19-32.

Barcelos, T. S., & Silveira, I. F. (2014). Computational thinking and mathematics: Possible relationships revealed by an analysis of national curriculum guidelines. In *Handbook of* *research on education and technology in a changing society* (pp. 922-934). Hersey, PA: IGI-Global Publishers.

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

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

Benakli, N., Kostadinov, B., Satyanarayana, A., & Singh, S. (2017). Introducing computational thinking through hands-on projects using R with applications to calculus, probability and data analysis. *International Journal of Mathematical Education in Science and Technology*, 48(3), 393-427.

Benton, L., Hoyles, C., Kalas, I., & Noss, R. (2017). Bridging primary programming and mathematics: Some findings of design research in England. *Digital Experiences in Mathematics Education*, 3(2), 115-138.

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.

Berkaliev, Z., Devi, S., Fasshauer, G. E., Hickernell, F. J., Kartal, O., Li, X., McCray. P.,Whitney, S., & Zawojewski, J. S. (2014). Initiating a programmatic assessment report. *Primus*, 24(5), 403-420.

For Peer Review

Bers, M. U., González-González, C., & Armas–Torres, M. B. (2019). Coding as a playground: Promoting positive learning experiences in childhood classrooms. *Computers & Education*, 138, 130-145.

Bortz, W. W., Gautam, A., Tatar, D., & Lipscomb, K. (2020). Missing in Measurement: Why Identifying Learning in Integrated Domains Is So Hard. *Journal of Science Education and Technology*, 29(1), 121-136.

Brennan, K., & Resnick, M. (2012). *New frameworks for studying and assessing the development of computational thinking.* Paper presented at the American Educational Research Association, British Columbia, Canada.

Buitrago Flórez, F., Casallas, R., Hernández, M., Reyes, A., Restrepo, S., & Danies, G. (2017). Changing a generation's way of thinking: Teaching computational thinking through programming. *Review of Educational Research*, 87(4), pp. 834-860.

Buteau, C., Gueudet, G., Muller, E., Mgombelo, J., & Sacristan, A. I. (2019). University students turning computer programming into an instrument for 'authentic' mathematical work. *International Journal of Mathematical Education in Science and Technology*, 1-22.

Cendros Araujo, R., Floyd, L. & Gadanidis, G. (2019). Teacher Candidates' Key Understandings about Computational Thinking in Mathematics and Science Education. *Journal of Computers in Mathematics and Science Teaching*, 38(3), 205-229.

Chan, S.-W., Looi, C.-K., Ho, W. K., Huang, W.. Seow, P., & Wu, L. (2021). Learning number patterns through computational thinking activities: A Rasch model analysis. *Heliyon*, 7(9), e07922.

For Peer Review

Chiazzese, Arrigo, Chifari, Lonati, & Tosto. (2019). Educational Robotics in Primary School: Measuring the Development of Computational Thinking Skills with the Bebras Tasks. *Informatics*, 6(4).

Ching, Y.-H., Yang, D., Wang, S., Baek, Y., Swanson, S., & Chittoori, B. (2019). Elementary School Student Development of STEM Attitudes and Perceived Learning in a STEM Integrated Robotics Curriculum. *TechTrends*, 63(5), 590-601.

Chongo, S., Osman, K., & Nayan, N. A. (2020). Level of Computational Thinking Skills among Secondary Science Student: Variation across Gender and Mathematics Achievement. *Science Education International*, 31(2), 159-163.

Cui, Z., & Ng, O.-L. (2020). The Interplay Between Mathematical and Computational Thinking in Primary School Students' Mathematical Problem-Solving Within a Programming Environment. *Journal of Educational Computing Research*, 59(5), 988-1012.

Cutumisu, M., Adams, C., & Lu, C. (2019). A Scoping Review of Empirical Research on Recent Computational Thinking Assessments. *Journal of Science Education and Technology*, 28(6), 651-676.

diSessa, A. A. (2018). Computational literacy and "The Big Picture" concerning computers in mathematics education. *Mathematical Thinking and Learning*, 20(1), 3-31.

Dixson, D. D., & Worrell, F. C. (2016). Formative and summative assessment in the classroom. *Theory Into Practice*, 55(2), 153-159.

Durak, H. Y., & Saritepeci, M. (2018). Analysis of the relation between computational thinking skills and various variables with the structural equation model. *Computers & Education*, 116, 191-202.

For Peer Review

Durak, H. Y., Yilmaz, F. G. K., & Bartin, R. Y. (2019). Computational thinking, programming self-efficacy, problem solving and experiences in the programming process conducted with robotic activities. *Contemporary Educational Technology*, *10*(2), 173–197. https://doi.org/10.30935/cet.554493

Ehsan, H., Beebe, C., & Cardella, M. E. (2017). *Promoting computational thinking in children using Apps.* American Society for Engineering Education.

English, L. (2018). On MTL's second milestone: Exploring computational thinking and mathematics learning. *Mathematical Thinking and Learning*, 20(1), 1-2.

Falloon, G. (2016). An analysis of young students' thinking when completing basic coding tasks using Scratch Jnr. On the iPad. *Journal of Computer Assisted Learning*, 32(6), 576-593.

Fanchamps, N. L. J. A., 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*.

Farris, A. V., Dickes, A. C., & Sengupta, P. (2019). Learning to interpret measurement and motion in fourth grade computational modeling. *Science & Education*, 28(8), 927-956.

Francis, K., & Davis, B. (2018). Coding robots as a source of instantiations for arithmetic. *Digital Experiences in Mathematics Education*, 4(2-3), 71-86.

Gadanidis, G. (2017). Five affordances of computational thinking to support elementary mathematics education. *Journal of Computers in Mathematics and Science Teaching*, 36(2), 143-151.

Gadanidis, G., Cendros, R., Floyd, L., & Namukasa, I. (2017). Computational thinking in mathematics teacher education. *Contemporary Issues in Technology and Teacher Education (CITE Journal)*, 17(4), 458-477.

For Peer Review

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

Gleasman, C., & Kim, C. Pre-service teacher's use of block-based programming and computational thinking to teach elementary mathematics. *Digital Experience in Mathematics Education*, 6, 52–90.

Grover, S., & Pea, R. (2018). Computational thinking: A competency whose time has come. In S. Sentance, E. Barendsen, & C. Schulte (Eds.), *Computer science education: Perspectives on teaching and learning* (pp. 19-38). Bloomsbury.

Grover, S., Jackiw, N., & Lundh, P. (2019). Concepts before coding: non-programming interactives to advance learning of introductory programming concepts in middle school. *Computer Science Education*, 29(2-3), 106-135.

Grover, S., Pea, R., & Cooper, S. (2015). Designing for deeper learning in a blended computer science course for middle school students. *Computer Science Education*, 25(2), 199-237.

Günbatar, M. S., & Bakırcı, H. (2019). STEM teaching intention and computational thinking skills of pre-service teachers. *Education and Information Technologies*, 24(2), 1615-1629.

Hardin, J. S., & Horton, N. J. (2017). Ensuring that mathematics is relevant in a world of data science. *Notices of the American Mathematical Society*, 64(9), 986-990.

Harlow, D. B., Dwyer, H. A., Hansen, A. K., Iveland, A. O., & Franklin, D. M. (2018). Ecological design-based research for computer science education: Affordances and effectivities for elementary school students. *Cognition and Instruction*, 36(3), 224-246.

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*, 1-22.

For Peer Review

Ho, W. K., Toh, P. C., Tay, E. G., Teo, K. M., Shutler, P. M. E., & Yap, R. A. S. (2017). The role of computation in teaching and learning mathematics at the tertiary level. In D. Kim, J. Shin, J. Park, & H. Kang (Eds.), *2017 ICJSME Proceedings: Future directions and issues in mathematics education*, 2, 441-448.

Hoyles, C., & Noss, R. (2015). *Revisiting programming to enhance mathematics learning*. Math + coding symposium Western University. London, Ontario, Canada: Western University.

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. (2019). From classroom lessons to exploratory learning progressions: mathematics + computational thinking. *Interactive Learning Environments*, 1-21.

Jaipal-Jamani, K., & Angeli, C. (2017). Effect of robotics on elementary preservice teachers' self-efficacy, science learning, and computational thinking. *Journal of Science Education and Technology*, 26(2), 175-192.

Jurado, E., Fonseca, D., Coderch, J., & Canaleta, X. (2020). Social STEAM learning at an early age with robotic platforms: A case study in four schools in Spain. *Sensors*, 20(13), 1-21.

Kallia, M., van Borkulo, S.P., Drijvers, P., Barendsen, E., Tolboom, J., (2021). Characterising computational thinking in mathematics education: a literature-informed Delphi study. *Research in Mathematics Education*.

Kaufmann, O. T., & Stenseth, B. (2020). Programming in mathematics education. *International Journal of Mathematical Education in Science and Technology*, 52(7), 1-20.

For Peer Review

Kelly, A., Finch, L., Bolles, M., & Shapiro, R. B. (2018). BlockyTalky: New programmable tools to enable students' learning networks. *International Journal of Child-Computer Interaction*, 18, 8-18.

Kiray, S. A. (2012). A new model for the integration of science and mathematics: The balance model. *Energy Education Science and Technology Part B: Social and Educational Studies*, 4(3), 1181–1196.

Kong, S-C., Lai, M., & Sun, D. (2020). Teacher development in computational thinking: Design and learning outcomes of programming concepts, practices and pedagogy. *Computers & Education*, 151, 1-19.

Kopcha, T. J., McGregor, J., Shin, S., Qian, Y., Choi, J., Hill, R., Mativo, J., & Choi, I. (2017). Developing an integrative STEM curriculum for robotics education through educational design research. *Journal of Formative Design in Learning*, 1(1), 31-44.

Hava, K., & Koyunlu Unlu, Z. (2021). Investigation of the relationship between middle school students' computational thinking skills and their STEM career interest and attitudes toward inquiry. *Journal of Science Education and Technology*, 30, 484-495.

Lai, P. S. M. (2013). Validating instruments of measure: Is it really necessary? *Malays Fam Physician*, 8(1), 2-4.

Leonard, J., Buss, A., Gamboa, R., Mitchell, M., Fashola, O. S., Hubert, T., & Almughyirah, S. (2016). Using robotics and game design to enhance children's self-efficacy, STEM attitudes, and computational thinking skills. *Journal of Science Education and Technology*, 25(6), 860-876.

Leonard, J., Mitchell, M., Barnes-Johnson, J., Unertl, A., Outka-Hill, J., Robinson, R., & Hester-Croff, C. (2018). Preparing teachers to engage rural students in computational thinking

For Peer Review

through robotics, game design, and culturally responsive teaching. *Journal of Teacher Education*, 69(4), 386-407.

Lin, Y.-T., Wang, M.-T., & Wu, C.-C. (2019). Design and implementation of interdisciplinary STEM instruction: Teaching programming by computational physics. *The Asia-Pacific Education Researcher*, 28(1), 77-91.

Lockwood, E., & De Chenne, A. (2019). Enriching students' combinatorial reasoning through the use of loops and conditional statements in Python. *International Journal of Research in Undergraduate Mathematics Education*.

MacPhail, C., Khoza, N., Abler, L., & Ranganathan, M. (2016). Process guidelines for establishing Intercoder Reliability in qualitative studies. *Qualitative Research*, *16*, 198–212. doi:10.1177/1468794115577012

Magana, A. J., Falk, M. L., Vieira, C., & Reese, M. J. (2016). A case study of undergraduate engineering students' computational literacy and self-beliefs about computing in the context of authentic practices. *Computers in Human Behavior*, 61, 427-442.

Marcelino, M. J., Pessoa, T., Vieira, C., Salvador, T., & Mendes, A. J. (2018). Learning computational thinking and scratch at distance. *Computers in Human Behavior*, 80, 470-477.

Martín-Ramos, P., Lopes, M. J., Lima da Silva, M. M., Gomes, P. E. B., Pereira da Silva, P. S., Domingues, J. P. P., & Ramos Silva, M. (2018). First exposure to Arduino through peer-coaching: Impact on students' attitudes towards programming. *Computers in Human Behavior*, 76, 51-58.

For Peer Review

McMillan, J. H., Hellsten, L. M., & Klinger, D. A. (2011). *Classroom assessment: Principles and practice for effective standards-based instruction* (Canadian ed.). Toronto, ON: Pearson.

Mesiti, L. A., Parkes, A., Paneto, S. C., & Cahill, C. (2019). Building capacity for computational thinking in youth through informal education. *Journal of Museum Education*, 44(1), 108-121.

Messer, D., Thomas, L., Holliman, A., & Kucirkova, N. (2018). Evaluating the effectiveness of an educational programming intervention on children's mathematics *skills, spatial awareness and working memory. Education* and Information Technologies, 23(6), 2879-2888.

Miller, J. (2019). STEM education in the primary years to support mathematical thinking: using coding to identify mathematical structures and patterns. *Zdm*, 51(6), 915-927.

Mouza, C., Marzocchi, A., Pan, Y.-C., & Pollock, L. (2016). Development, implementation, and outcomes of an equitable computer science after-school program: Findings from middle-school students. *Journal of Research on Technology in Education*, 48(2), 84-104.

Munoz, L., Villarreal, V., Morales, I., Gonzalez, J., & Nielsen, M (2020). Developing an interactive environment through the teaching of mathematics with small robots. *Sensors*, 20(7), 1-23.

Nam, K. W., Kim, H. J., & Lee, S. (2019). Connecting plans to action: The effects of a card-coded robotics curriculum and activities on Korean kindergartners. *The Asia-Pacific Education Researcher*, 28(5), 387-397.

For Peer Review

Namukasa, I. K., Patel, M., & Miller, M. (2017). *Introductory CT and Math in the middle grades: Tools and Resources. Math + Code 'Zine*, 2 (2). Retrieved from http://researchideas.ca/mc/ct-and-math-in-middle-grades/

Ng, O., & Cui, Z. (2021). Examining primary students' mathematical problem-solving in a programming context: towards computationally enhanced mathematics education. *ZDM-Mathematics Education*, 53, 847-860.

Nordby, S. K. Bjerke, A. H., & Mifsud, L. (2022). Computational thinking in the primary mathematics classroom: A systematic review. *Digital Experiences in Mathematics Education*.

Olteanu, C. (2020). Programming, mathematical reasoning and sense-making. *International Journal of Mathematical Education in Science and Technology*, 1-19.

Ozcan, M. S., Cetinkaya, E., Goksun, T., & Kisbu-Sakarya, Y. (2021). Does learning to code influence cognitive skills of elementary school children? Findings from a randomized experiment. *British Journal of Educational Psychology*, 91(4), 1434–1455.

Panskyi, T., Rowinska, Z., & Biedron, S. (2019). Out-of-school assistance in the teaching of visual creative programming in the game-based environment – Case study: Poland. *Thinking Skills and Creativity*, 34.

Papadakis, S., & Kalogiannakis, M. (2019). Evaluating a course for teaching introductory programming with Scratch to pre-service kindergarten teachers. *International Journal of Technology Enhanced Learning*, 11(3).

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

For Peer Review

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

Pei, C., Weintrop, D., & Wilensky, U. (2018). Cultivating computational thinking practices and mathematical habits of mind in Lattice Land. *Mathematical Thinking and Learning*, 20(1), 75-89.

Piedade, J., Dorotea, N., Pedro, A., & Matos, J.F. (2020). On teaching programming fundamentals and computational thinking with educational robotics: A didactic experience with pre-service teachers. *Education Sciences*, *10*(9), 1-15.

Pinkard, N., Martin, C. K., & Erete, S. (2019). Equitable approaches: opportunities for computational thinking with emphasis on creative production and connections to community. *Interactive Learning Environments*, *0*(0), 1–15.

Polat, E., Hopcan, S., Kucuk, S., & Sisman, B. (2021). A comprehensive assessment of secondary school students' computational thinking skills. *British Journal of Educational Technology*, 52(5), 1965–1980.

Price, S., & Pontual Falcão, T. (2011). Where the attention is: Discovery learning in novel tangible environments. *Interacting with Computers*, 23(5), 499-512.

Psycharis, S. (2016). The impact of computational experiment and formative assessment in inquiry-based teaching and learning approach in STEM education. *Journal of Science Education and Technology*, 25(2), 316-326.

Rich, K.M., Yadav, A., & Larimore, R.A. (2020). Teacher implementation profiles for integrating computational thinking into elementary mathematics and science instruction. *Education and Information Technologies*, 25, 3161–3188.

For Peer Review

Rico Lugo, M. J., Olabe, X. B., & Nino, N. M. (2018). “Evolution”: Design and implementation of digital educational material to strengthen computational thinking skills. *The IEEE Journal of Latin-American Learning Technologies (IEEE-RITA)*, 13(1), 37-45.

Rodriguez-Martínez, J. A., Gonzalez-Calero, J. A., & Saez-Lopez, J. M. (2019). Computational thinking and mathematics using Scratch: an experiment with sixth-grade students. *Interactive Learning Environments*, 1-12.

Román-González M., Moreno-León J., & Robles G. (2019) Combining assessment tools for a comprehensive evaluation of computational thinking interventions. In S. C. Kong & H. Abelson (Eds) *Computational Thinking Education*. Springer, Singapore.

Roman-Gonzalez, M., Perez-Gonzalez, J.-C., Moreno-Leon, J., & Robles, G. (2018). Can computational talent be detected? Predictive validity of the Computational Thinking Test. *International Journal of Child-Computer Interaction*, 18, 47-58.

Saez-Lopez, J.-M., Roman-Gonzalez, M., & Vazquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using “Scratch” in five schools. *Computers & Education*, 97, 129-141.

Saez-Lopez, J.-M., Sevillano-Garcia, M.-L., & 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*.

Seo, Y.-H., & Kim, J.-H. (2016). Analyzing the effects of coding education through pair programming for the computational thinking and creativity of elementary school students. *Indian Journal of Science and Technology*, 9(46).

Sinclair, N., & Patterson, M. (2018). The dynamic geometrisation of computer programming. *Mathematical Thinking and Learning*, 20(1), 54-74.

For Peer Review

Sisman, B., Kucuk, S. & Yaman, Y. (2020). The effects of robotics training on children's spatial ability and attitude toward STEM. *International Journal of Social Robotics*, 13, 379–389.

Sırakaya, M., Alsancak Sırakaya, D., & Korkmaz, O. (2020). The impact of STEM attitude and thinking style on computational thinking determined via structural equation modeling. *Journal of Science Education and Technol*ogy, 29, 561–572.

Sneider, C., Stephenson, C., Schafer, B., & Flick, L. (2014). Teacher's toolkit: Exploring the science framework and NGSS: Computational thinking in the science classroom. *Science Scope*, 38(3), 10.

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.

Sung, W., Ahn, J., & Black, J. B. (2017). Introducing computational thinking to young learners: Practicing computational perspectives through embodiment in mathematics education. *Technology, Knowledge and Learning*, 22(3), 443-463.

Sung, W., & Black, J. B. (2020). Factors to consider when designing effective learning: Infusing computational thinking in mathematics to support thinking-doing. *Journal of Research on Technology in Education*, 53, 404–426.

Suters, L., & Suters, H. (2020). Coding for the Core: Computational thinking and middle grades mathematics. *Contemporary Issues in Technology and Teacher Education, 20*(3), 435-471.

Tabesh, Y. (2017). Computational thinking: A 21st century skill. *Olympiads in Informatics*, 11(SI), 65-70.

Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. *Computers & Education, 148*, 103798. https://doi.org/10.1016/j.compedu.2019.103798.

For Peer Review

Tran, Y. (2018). Computer programming effects in elementary: Perceptions and career aspirations in STEM. *Technology, Knowledge and Learning*, 23, 273–299.

Valentine, K. D. (2018). Tinkering with Logo in an elementary mathematics methods course. *Interdisciplinary Journal of Problem-based Learning*, 12(2).

Valovicova, L., Ondruska, J., Zelenicky, L., Chytry, V., & Medova, J. (2020). Enhancing computational thinking through interdisciplinary STEAM activities using tablets. *Mathematics*, 8(12), 1-15.

Vieira, C., Magana, A. J., Roy, A., & Falk, M. L. (2019). Student explanations in the context of computational science and engineering education. *Cognition and Instruction*, 37(2), 201-231.

Waterman, K., Goldsmith, L., Pasquale, M., Goldenberg, E.P., Malyn-Smith, J., DeMallie, A., & Lee, I.A. (2018). Integrating computational thinking into elementary mathematics and science curriculum materials and instruction. In Pixel (Ed.), *Conference Proceedings: the Future of Education 2018*. Florence, Italy: LibreriaUniversitariaEdizioni.

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

Wing, J.M. (2010). *Computational thinking: What and why?* Unpublished manuscript Computer Science Department, Carnegie Mellon University, Pittsburgh, PA.

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

For Peer Review

Zhong, B., Wang, Q., Chen, J., & Li, Y. (2016). An exploration of three-dimensional integrated assessment for computational thinking. *Journal of Educational Computing Research*, 53(4), 562-590. doi:10.1177/0735633115608444

Ziaeefard, S., Miller, M. H., Rastgaar, M., & Mahmoudian, N. (2017). Co-robotics hands-on activities: A gateway to engineering design and STEM learning. *Robotics and Autonomous Systems*, 97, 40-50.

For Peer Review

Table 1. Summary of the Number of Articles from the Initial Search and Final Search

Databases	Number of papers from the initial search	First Round Screening	Second Round Screening	Third Round Screening	Number of papers from the final search
Scopus	491	25	67	366	33
Web of Science	363	133	47	173	10
Science Direct	341	16	157	159	9
Springer Link	584	69	284	219	12
Taylor & Francis Online	297	29	162	102	4
ERIC	157	87	26	39	5
Total	2233	359	743	1058	73

For Peer Review

Table 2. Types of CT Tools

Types of CT Tools	CT Tools	Studies
Digital Tangibles	LEGO®EV3 robotics/ Lego Mindstorms EV3/NXT robots	(Leonard et al., 2016; Francis & Davis, 2018; Piedade et al., 2020; Suters, & Suters, 2020)
	Lego® Education WeDo 2.0 robotics kit	(Jaipal-Jamani & Angeli, 2017; Chiazzese et al., 2019)
	Robots/Programmable robots and circuits	(Gadanidis et al., 2017; Kopcha et al., 2017; Ching et al., 2019)
	Arduino	(Martín-Ramos et al., 2018; Cui & Ng, 2020; Ng & Cui, 2021)
	mBot robot	(Saez-Lopez et al., 2019; Piedade et al., 2020)
	SRA-programming Lego NXT Mindstorms robots	(Fanchamps et al., 2019)
	TurtleBot	(Nam et al., 2019)
	KIBO robotics kit	(Bers et al., 2019; Jurado et al., 2020)
	Sphero	(Cendros Araujo et al., 2019)
	micro:bit	(Cendros Araujo et al., 2019)
	Zowi	(Piedade et al., 2020)
	Dash &Dot	(Piedade et al., 2020)
	Anprino	(Piedade et al., 2020)
	Codey Rocky	(Piedade et al., 2020)
	Bee-Bot robot	(Munoz et al., 2020)
	Robotics Dream ER kits	(Sisman et al., 2020)
Apps and Games	Scalable Game	(Leonard et al., 2016)
	Kinect games	(Altanis et al., 2018)
	Bee-bot iPad app	(Messer et al., 2018)
	SmartMeasure app	(Valovicova et al., 2020)
	Hour of Code	(Stigberg, & Stigberg, 2020)
	Lightbot	(Stigberg, & Stigberg, 2020)
	Digital Educational Material (DEM) "Evolution"	(Rico Lugo et al., 2018)
Programming Languages	Scratch	(Grover et al. 2015; Mouza et al., 2016; Benton et al., 2017; Gadanidis, 2017; Gadanidis et al., 2017; Gadanidis et al., 2018; Cendros Araujo et al., 2019; Grover et al., 2019; Israel & Lash, 2019; Rodriguez-Martinez et al., 2019; Marcelino et al., 2018; Miller, 2019; Gleasman, & Kim, 2020; Olteanu, 2020)
	Scratch Jr.	(Falloon, 2016; Sung et al., 2017; Stigberg, & Stigberg, 2020)
	Dr. Scratch	(Altanis et al., 2018; Papadakis & Kalogiannakis, 2019; Panskyi et al., 2019)
	MATLAB	(Magana et al., 2016; Vieira et al., 2019)
	Mathematica	(Magana et al., 2016)
	Thermo-Calc	(Magana et al., 2016)
	COMSOL	(Magana et al., 2016)
	Statistical programming language R	(Benakli et al., 2017)
	Python	(Gadanidis, 2017; Gadanidis et al., 2017; Cendros Araujo et al., 2019; Lin et al., 2019; Lockwood & De Chenne, 2019; Stigberg, & Stigberg, 2020)
	Google's Blockly/Code.org coding problems	(Gadanidis, 2017; Gadanidis et al., 2018; Roman-Gonzalez et al., 2018; Arfe et al., 2019; Israel & Lash, 2019; Piedade et al., 2020)
	Sketchpad	(Sinclair, & Patterson, 2018)

For Peer Review

	Logo	(Valentine, 2018)
	BlockyTalky	(Kelly et al., 2018)
	LaPlaya	(Harlow et al., 2018)
	ViMAP	(Farris, Dickes, & Sengupta, 2019)
	Visual Studio IDE	(Buteau et al., 2019)
	Bitbloq	(Piedade et al., 2020)
	mBlock	(Piedade et al., 2020)
	Ardublockly	(Piedade et al., 2020)
	Processing	(Kaufmann, & Stenseth, 2020)
	Bootstrap Algebra	(Suters, & Suters, 2020)
Formative or Summative Assessments	Computational cognition test	(Seo & Kim, 2016)
	Visual Blocks Creative Computing Test (VBCCT)	(Saez-Lopez et al., 2016)
	Computational Thinking Test (CTt)	(Benton et al., 2017; Roman-Gonzalez et al., 2018; Rodriguez-Martinez et al., 2019; Polak et al., 2021)
	Computational Thinking survey	(Leonard et al., 2018)
	Pretest and posttest on computational thinking	(Tran, 2018; Ozcan et al., 2021)
	Computational Thinking Skills Scale (CTSS)/Computational Thinking Levels Scale (CTLS)	(Durak & Saritepeci, 2018; Gunbatar & Bakırcı, 2019; Sırakaya et al., 2020; Polak et al., 2021; Hava & Koyunlu Unlu, 2021)
	Robotics Activities Attitudes Scale (RAAS)	(Ziaeefard et al., 2017)
	Bebras tasks	(Chiazzese et al., 2019)
	Solve-Its assessments	(Bers et al., 2019)
	VELA activities	(Grover et al., 2019)
	CT Lesson Screener	(Rich et al., 2020)
	CT Lesson Enhancer	(Rich et al., 2020)
	CT concept test	(Kong et al., 2020)
	CT practice test	(Kong et al., 2020)
	CT test	(Chongo et al., 2020)
	Math + C worksheets	(Chan et al., 2021)
Other Technological Tools	Easy Java Simulator (Ejs) tool	(Psycharis, 2016)
	Lattice Land	(Pei et al., 2018)
	The Science Behind Pixar (Pixar) exhibition	(Mesiti et al., 2019)
	Spreadsheet	(Chan, et al., 2021)

For Peer Review

Table 3. CT Competencies and Related CT Tools

	CT Competencies	CT Tools
Computational Concepts	Sequences	LEGO®EV3 robotics, Lego® Education WeDo 2.0 robotics kit, mBot robot, Robots/Programmable robots and circuits, Arduino, Scalable Game, Bee-bot iPad app, Scratch, Scratch Jr., Google's Blockly/Code.org coding problems, Sketchpad, Logo, LaPlaya, Visual Blocks Creative Computing Test (VBCCT), Computational Thinking Test (CTt), Pretest and posttest on computational thinking, Solve-Its assessments, Bitbloq, mBlock, Ardublockly, CT concept test
	Loops	LEGO®EV3 robotics, Lego® Education WeDo 2.0 robotics kit, KIBO robotics kit, Robots/Programmable robots and circuits, Arduino, Scalable Game, Scratch, Scratch Jr., Python, Google's Blockly/Code.org coding problems, Sketchpad, Logo, BlockyTalky, LaPlaya, Visual Studio IDE, Computational Thinking Test (CTt), Pretest and posttest on computational thinking, VELA activities, Robotics Dream ER kits, Bitbloq, mBlock, Ardublockly
	Events/Event Handling	Arduino, mBot robot, Scratch Jr., Sketchpad, LaPlaya, Visual Blocks Creative Computing Test (VBCCT), Computational Thinking Test (CTt), CT concept test
	Functions	Computational Thinking Test (CTt), Robotics Dream ER kits
	Iteration/Repetition	Arduino, mBot robot, Sphero, micro:bit, Scratch, Scratch Jr., Python, Visual Blocks Creative Computing Test (VBCCT), Computational Thinking Test (CTt), Solve-Its assessments, VELA activities, Processing, CT practice test
	Parallelism	mBot robot, Kinect games, Scratch, Scratch Jr., Dr. Scratch, Sketchpad, Visual Blocks Creative Computing Test (VBCCT), Zowi, Dash & Dot, Anprino, Codey Rocky, Bitbloq, mBlock, Ardublockly, CT concept test
	Conditionals	Arduino, mBot robot, KIBO robotics kit, Robots/Programmable robots and circuits, Scratch, Scratch Jr., Python, Google's Blockly/Code.org coding problems, Sketchpad, Visual Studio IDE, Visual Blocks Creative Computing Test (VBCCT), Computational Thinking Test (CTt), Pretest and posttest on computational thinking, Solve-Its assessments, VELA activities, Bitbloq, mBlock, Ardublockly, CT concept test
	Subroutines	Sketchpad
	Operators/Operations	Arduino, Scratch, Scratch Jr., Sketchpad, VELA activities, Bitbloq, mBlock, Ardublockly, CT concept test
	Data	Scratch, Scratch Jr., Sketchpad, VELA activities
	Variables/Naming	Arduino, Scratch, Scratch Jr., Python, Sketchpad, BlockyTalky, LaPlaya, VELA activities, Robotics Dream ER kits, Bitbloq, mBlock, Ardublockly, CT concept test
	Automation	Sphero, micro:bit, Scratch, Python, SmartMeasure app
	Initialisation	Scratch, LaPlaya
	Synchronisation/Synchronism	Kinect games, Dr. Scratch, Zowi, Dash & Dot, Anprino, Codey Rocky, Bitbloq, mBlock, Ardublockly
	User Interactivity/User Interface Design	Kinect games, Dr. Scratch, LaPlaya, Visual Blocks Creative Computing Test (VBCCT)
	Flow Control	Kinect games, Dr. Scratch, Zowi, Dash & Dot, Anprino, Codey Rocky
	Data Representation	Zowi, Dash & Dot, Anprino, Codey Rocky

For Peer Review

	Patterns/Pattern Recognition	TurtleBot, Sphero, micro:bit, Scratch, Scratch Jr., Python, Sketchpad, Lattice Land, The Science Behind Pixar (Pixar) exhibition, Zowi, Dash & Dot, Anprino, Codey Rocky, CT Lesson Screener, CT Lesson Enhancer, Math + C worksheets, Spreadsheet
	Abstraction	LEGO®EV3 robotics, Robots/Programmable robots and circuits, Arduino, Sphero, micro:bit, Scalable Game, Kinect games, Scratch, Scratch Jr., Dr. Scratch, Python, Google's Blockly/Code.org coding problems, Sketchpad, Bebras tasks, Lattice Land, The Science Behind Pixar (Pixar) exhibition, Zowi, Dash & Dot, Anprino, Codey Rocky, CT Lesson Screener, CT Lesson Enhancer, CT practice test, CT test, Math + C worksheets, Spreadsheet
	Generalization	LEGO®EV3 robotics, Scalable Game, Zowi, Dash & Dot, Anprino, Codey Rocky, CT test
	Decomposition	Sphero, micro:bit, Scratch, Scratch Jr., Dr. Scratch, MATLAB, Python, Sketchpad, LaPlaya, Lattice Land, The Science Behind Pixar (Pixar) exhibition, Zowi, Dash & Dot, Anprino, Codey Rocky, CT Lesson Screener, CT Lesson Enhancer, CT test, Math + C worksheets, Spreadsheet
	Evaluation	CT test
	Arithmetic	Bitbloq, mBlock, Ardublockly
	Relational	Bitbloq, mBlock, Ardublockly
	Sensors and Decisions/Decision Structures	Robots/Programmable robots and circuits, Scratch, LaPlaya, Visual Studio IDE, Robotics Dream ER kits
	Randomness and Expressions	Scratch
	Spatial location and cognition	Bee-bot robot
	Logic and Logical Thinking	LEGO®EV3 robotics, Sphero, micro:bit, Scalable Game, Kinect games, Scratch, Dr. Scratch, Python, Computational cognition test, Bebras tasks, Bee-bot robot, Bitbloq, mBlock, Ardublockly
	Algorithms and Algorithmic Thinking	LEGO®EV3 robotics, Lego® Education WeDo 2.0 robotics kit, Robots/Programmable robots and circuits, Arduino, SRA-programming Lego NXT Mindstorms robots, TurtleBot, Sphero, micro:bit, Scalable Game, Scratch, Dr. Scratch, Python, Sketchpad, Computational cognition test, Pretest and posttest on computational thinking, Computational Thinking Skills Scale (CTSS), Bebras tasks, Digital Educational Material (DEM) "Evolution", The Science Behind Pixar (Pixar) exhibition, Zowi, Dash & Dot, Anprino, Codey Rocky, Hour of Code, Lightbot, Bitbloq, mBlock, Ardublockly, CT practice test, CT test, Math + C worksheets, Spreadsheet
	Mathematical Thinking/Skills	Digital Educational Material (DEM) "Evolution"
	Critical Thinking	Computational cognition test, Computational Thinking Skills Scale (CTSS), Digital Educational Material (DEM) "Evolution"
	Abstract Thinking	Computational cognition test
	Sequential Thinking	Scratch Jr.
	Procedural thinking	TurtleBot, Scratch Jr., Bitbloq, mBlock, Ardublockly
	Recursive Thinking	Computational cognition test
	Efficient Thinking	TurtleBot
Computational Practices	Agency	Scratch, Python, Google's Blockly/ Code.org coding problems

https://mc.manuscriptcentral.com/joecr

	Access	Scratch, Python, Google's Blockly/ Code.org coding problems
	Surprise	Scratch, Google's Blockly/ Code.org coding problems
	Audience	Scratch, Python, Google's Blockly/ Code.org coding problems
	Creativity	Computational Thinking Skills Scale (CTSS)
	Data practices	LEGO®EV3 robotics, Scalable Game, Dr. Scratch, Statistical programming language R, Lattice Land, Bootstrap Algebra
	Experimentation	Visual Blocks Creative Computing Test (VBCCT)
	Incrementalism	Scratch Jr., CT practice test
	Problem-solving	Computational Thinking Skills Scale (CTSS), Lattice Land, Hour of Code, Lightbot, Processing
	Computational problem-solving	Bootstrap Algebra
	Problem Formulation	LEGO®EV3 robotics, Scalable Game
	Modularizing/Modularization Testing	Arduino, CT practice test
	Creating and Broadcasting	Scratch, The Science Behind Pixar (Pixar) exhibition
	Modeling and Simulation	Arduino, Scratch, MATLAB, Statistical programming language R, ViMAP, VELA activities, Easy Java Simulator (Ejs) tool, Lattice Land, Bootstrap Algebra
	Systems thinking practices	Bootstrap Algebra
	Reusing and Remixing	Arduino, Scratch Jr., CT practice test
	Testing and Debugging	Lego® Education WeDo 2.0 robotics kit, Arduino, Sphero, micro:bit, Bee-bot iPad app, Scratch, Scratch Jr., Python, Google's Blockly/ Code.org coding problems, Pretest and posttest on computational thinking, Solve-Its assessments, Lattice Land, The Science Behind Pixar (Pixar) exhibition, Zowi, Dash & Dot, Anprino, Codey Rocky, Bitbloq, mBlock, Ardublockly, CT Lesson Screener, CT Lesson Enhancer, CT practice test
	Copying and adjustments	Processing
Computational Perspectives	Self-efficacy	Computational Thinking survey
	Use of Culture	LEGO®EV3 robotics, Scalable Game
	Confidence	Arduino, Robotics Activities Attitudes Scale (RAAS)
	Appraisals/Appraising the outcome	Scratch Jr., MATLAB, Mathematica, Thermo-Calc, COMSOL
	Sense of Identity	The Science Behind Pixar (Pixar) exhibition
	Motor skills and perception	Bee-bot robot
	Communication skills	Hour of Code, Lightbot
	Motivation and Interest	Arduino, Computational Thinking survey, Robotics Activities Attitudes Scale (RAAS)
	Cooperation/Collaboration	Computational Thinking Skills Scale (CTSS), The Science Behind Pixar (Pixar) exhibition
	Expressing and Questioning about Technology World	Scratch Jr., The Science Behind Pixar (Pixar) exhibition
	Connecting to the Technical World	Arduino, Scratch Jr., Computational Thinking survey, Robotics Activities Attitudes Scale (RAAS), The Science Behind Pixar (Pixar) exhibition

For Peer Review

Table 4. Mathematics topics assessed

Mathematics Topics	Sub-topics	Studies
Numbers and Operations	Multiplication and factors, place value, conversions of length, weight and time, different types of mathematical relationship including proportionality and ratio in the context of drawing, addition and subtraction mental math strategies, decimals and fractions, multiples and factors of whole numbers, number line estimation, arithmetic, negative numbers, percentages, mathematical equations, four operations (addition, subtraction, multiplication, and division), prime and composite number	(Benton et al., 2017; Gadanidis, 2017); Kopcha et al., 2017; Sung et al., 2017; Francis & Davis, 2018; Harlow et al., 2018; Messer et al., 2018; Rico Lugo et al., 2018; Israel & Lash, 2019; Lin et al., 2019; Nam et al., 2019; Saez-Lopez et al., 2019; Gleasman, & Kim, 2020; Ozcan et al., 2021; Ng & Cui, 2021)
Algebra	Number patterns/patterning, number sense, algebraic operations, Least common multiple (LCM) and the greatest common divisor (GCD)	(Gadanidis, 2017; Gadanidis et al., 2017; Kopcha et al., 2017; Sinclair, & Patterson, 2018; Israel & Lash, 2019; Miller, 2019; Nam et al., 2019; Rodriguez-Martinez et al., 2019; Gleasman, & Kim, 2020, Chan et al., 2021; Ng & Cui, 2021)
Calculus	-	(Benakli, 2017)
Geometry and Measurement	Symmetry, regular polygons, coordinates, rectangles, translations and reflections of regular polygons through the coordinates system, shapes, coordinate geometry, repeating patterns, area representations of fractions, area and perimeter relationships, symmetry and transformations, coordinate grid, coordinate plane, discrete geometry, segments, circles, transformations, volume, angles, distances, speed	(Falloon, 2016; Mouza et al., 2016; Seo & Kim, 2016; Benton et al., 2017; Gadanidis, 2017; Gadanidis, 2017; Kopcha et al., 2017; Altanis, 2018; Gadanidis et al., 2018; Harlow et al., 2018; Pei et al., 2018; Sinclair, & Patterson, 2018; Valentine, 2018; Cendros et al., 2019; Ching et al., 2019; Farris et al., 2019; Israel & Lash, 2019); Panskyi et al., 2019; Nam et al., 2019; Saez-Lopez et al., 2019; Gleasman, & Kim, 2020; Valovicova, et al., 2020; Ozcan et al., 2021)
Statistics and Probability	Data analysis, Binomial Theorem, permutations, combinations	Benakli et al., 2017; Gadanidis, 2017; Cendros Araujo et al., 2019; Lockwood & De Chenne, 2019; Nam et al., 2019)

For Peer Review

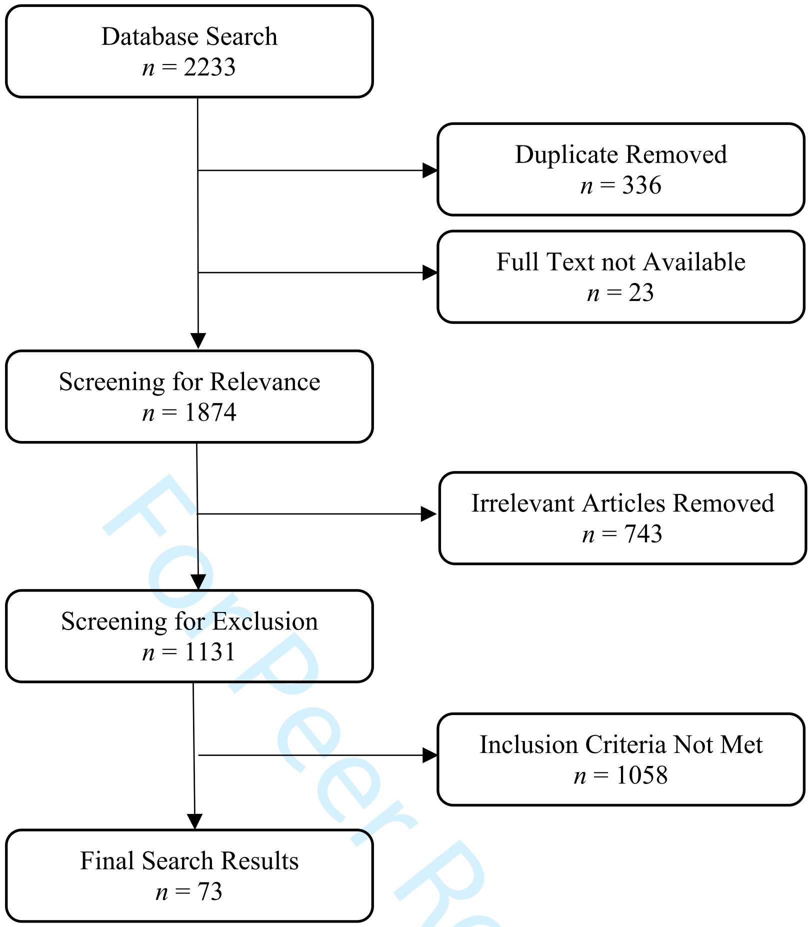

Figure 1. Flow diagram for study selection procedure

For Peer Review

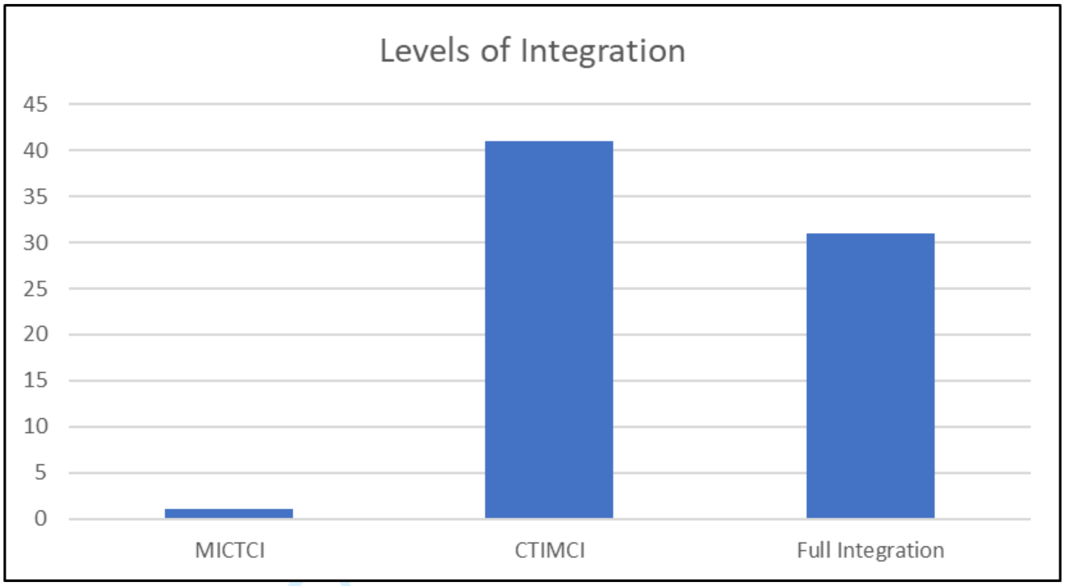

Figure 2. Levels of Integration

For Peer Review

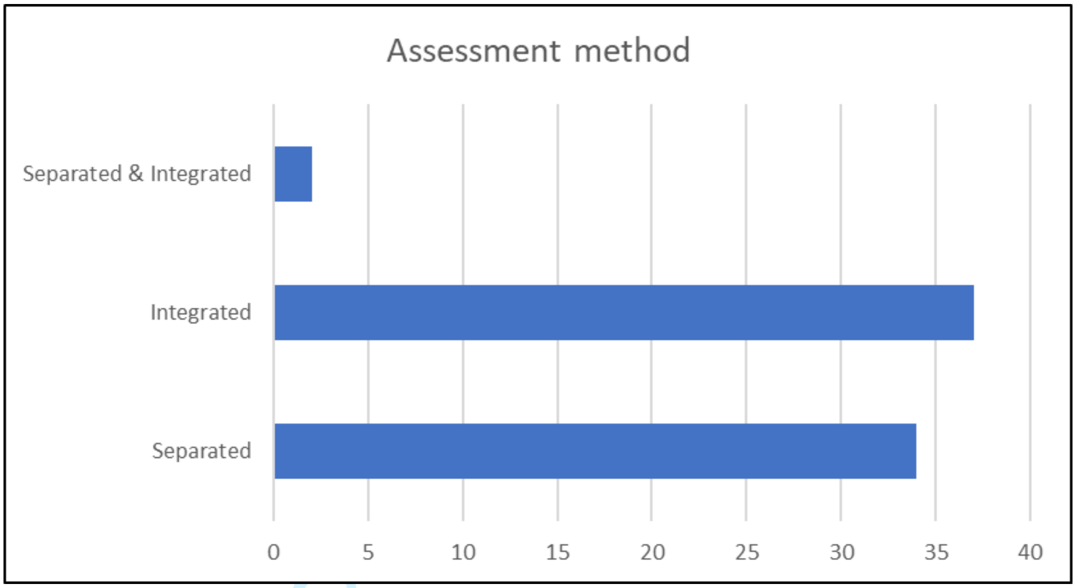

Figure 3. Assessment Methods

For Peer Review

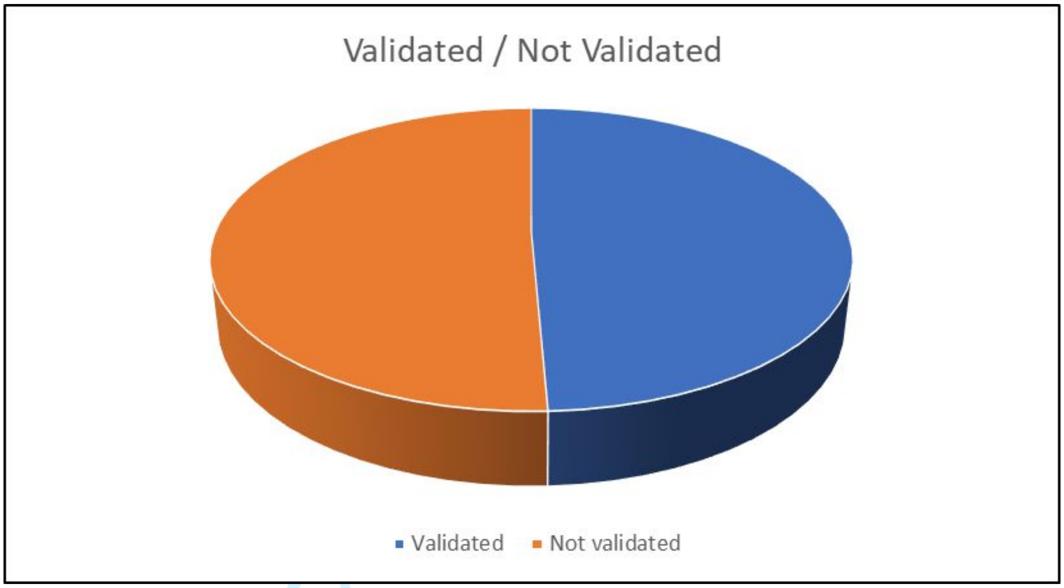

Figure 4. Percentage of Validated or Not Validated Assessments

For Peer Review

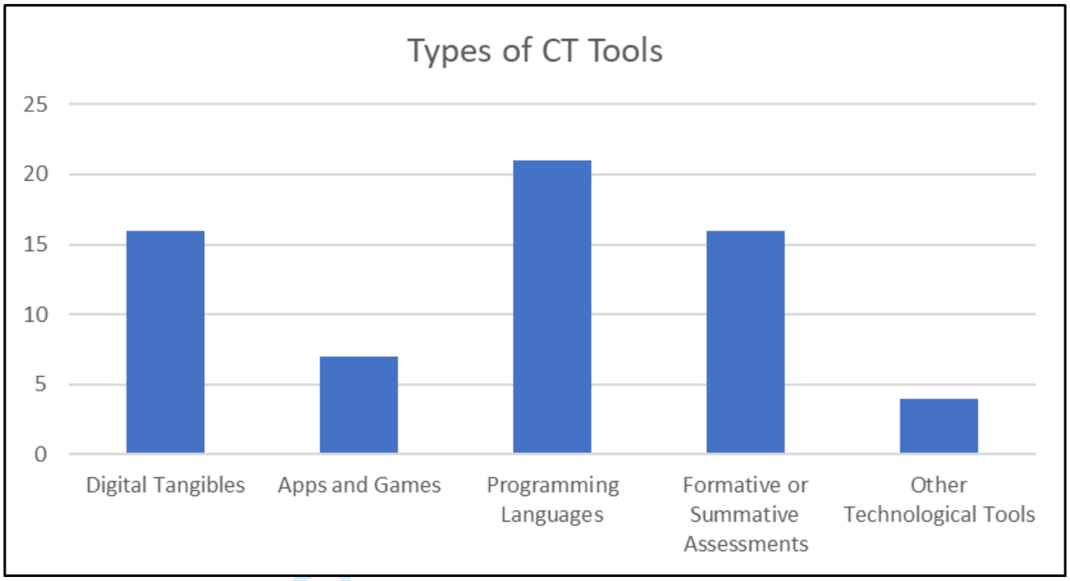

Figure 5. Types of CT Tools

For Peer Review

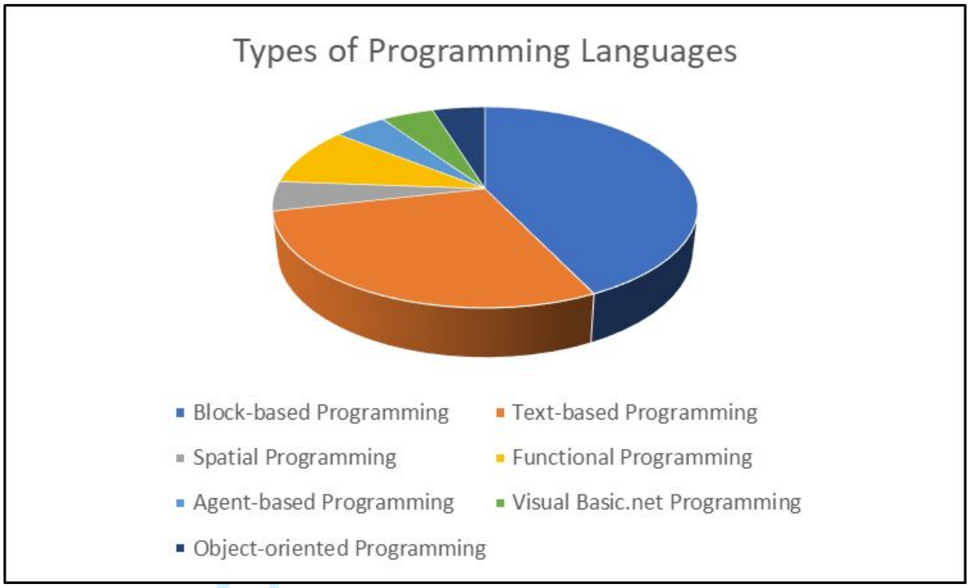

Figure 6. Types of Programming Languages

For Peer Review

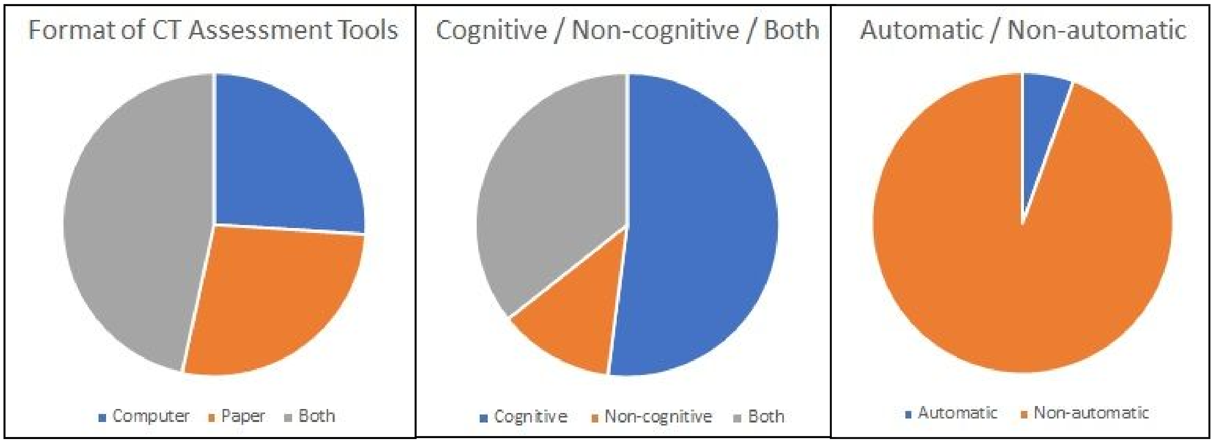

Figure 7. Forms of CT Tools

For Peer Review

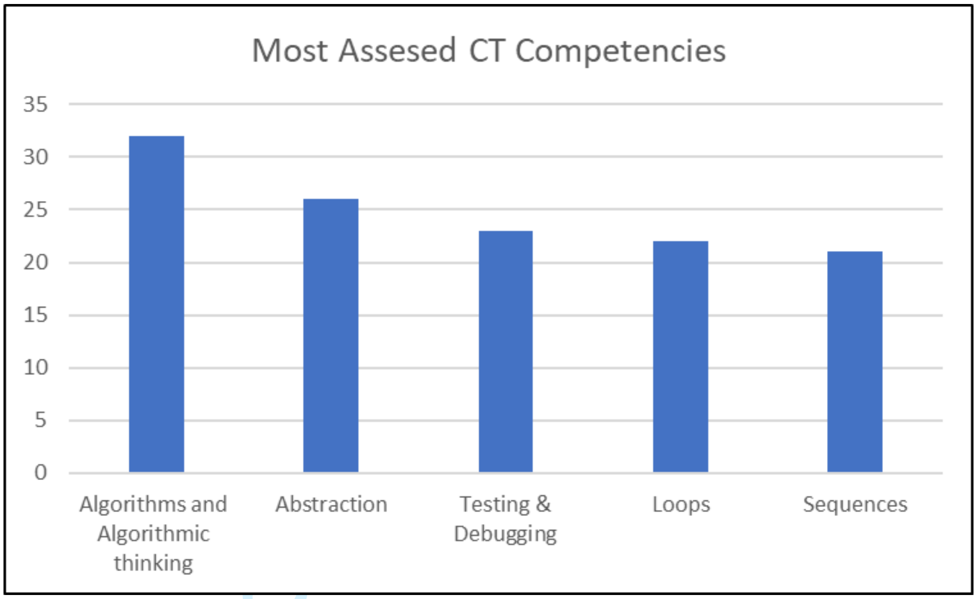

Figure 8. Most Assessed CT Competencies

For Peer Review

Appendix A. Tools and approaches of integrated instruction in the reviewed empirical studies

No	Authors	How has CT been integrated into mathematics instruction			Forms of CT tools					CT Competencies	Mathematics Topics
		Level of Integration	Separated/ Integrated Assessment	Validated / Not validated	CT Tools	Assessment Methods	Format of CT Tools	Cognitive/ non-cognitive	Automatic/non-automatic		
1	(Grover, Pea, & Cooper, 2015)	CTIMCI	Separated	Not validated	Scratch	In-class tasks, knowledge test, survey	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	Sequence of instructions – serial execution; repetition; selection, serial execution, simple nested loop + creative computing, forever loop, variables; user input, algorithms in different forms (analogous representations for deeper learning), loops, variables, creative computing, loops ending with Boolean condition, conditionals; event handlers, loops, variables, conditionals, Boolean logic	-
2	(Falloon, 2016)	CTIMCI	Integrated	Validated	Scratch Jr.	In-class tasks	Computer	Cognitive	Non-automatic	Sequencing, triggers and events, working with data, values and variables, conceptualizing (the task), conceptualizing (the toolset and resources), creating and testing code using incremental/non-incremental and/or iterative/non-iterative strategies, conceptualizing tasks and criteria, and identifying possible code errors, prior to testing. Debugging, prior to testing. conceptualizing code errors, after testing. Debugging, after testing. Appraising the outcome from running code (not action-oriented), Appraising the outcome from running code and modifying, if needed (action-oriented), sharing (connecting), questioning	Geometry
3	(Leonard, Buss, Gamboa, Mitchell, Fashola, Hubert, & Almughyirah, 2016)	CTIMCI	Integrated	Validated	LEGO®EV3 robotics/ Lego Mindstorms EV3/NXT robots & Scalable Game	In-class tasks, survey	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	Problem formulation, abstraction, logical thinking, algorithms, analyzing and implementing, generalizing and transfer, use of culture	-
4	(Magana, Falk, Vieira, & Reese, 2016)	CTIMCI	Integrated	Validated	MATLAB, Mathematica, Thermo-Calc, & COMSOL	Survey, knowledge test	Paper	Cognitive & Non-cognitive	Non-automatic	Control appraisals & Value appraisals	-
5	(Mouza, Marzocchi, Pan, & Pollock, 2016)	CTIMCI	Separated	Validated	Scratch	Observation, knowledge test, artefact assessment, survey	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	Cognitive: loops, conditionals, data (modifying a variable), parallelism within object, parallelism across objects, operators Non-cognitive: computing confidence, computer enjoyment, computer importance and perceived usefulness of computing, motivation to succeed in computing, computing identity and belongingness, gender equity and intention to persist	Geometry
6	(Psycharis, 2016)	CTIMCI	Integrated	Not validated	Easy Java Simulator (Ejs) tool	Knowledge test	Paper	Cognitive	Non-automatic	Modelling	-
7	(Saez-Lopez, Roman-Gonzalez, M., & Vazquez-Cano, 2016)	CTIMCI	Separated	Validated	Visual Blocks Creative Computing Test (VBCCT)	Knowledge test, survey	Paper	Cognitive & Non-cognitive	Non-automatic	Sequence iteration (looping), conditional statements, threads (parallel execution), event handling, user interface design, keyboard input, experimentation, and iteration	-
8	(Seo & Kim, 2016)	CTIMCI	Separated	Not validated	Computational cognition test	Knowledge test	Paper	Cognitive	Non-automatic	Abstract thinking, critical thinking, logical thinking, recursive thinking, and algorithmic thinking	Geometry
9	(Benakli, Kostadinov, Satyanarayana, & Singh, 2017)	Full integration	Integrated	Not validated	Statistical programming language R	survey	Paper	Non-cognitive	Non-automatic	Simulations, visualizations, data analysis	Calculus, probability, data analysis
10	(Benton, Hoyles, Kalas, & Noss, 2017)	Full integration	Separated	Not validated	Scratch & Computational Thinking Test (CTt)	In-class tasks, knowledge test	Both	Cognitive	Non-automatic	Sequencing, repetition, algorithm, debugging and abstraction, initialization, randomness and expressions, conditions, broadcasting	Symmetry, regular polygons, coordinates, multiplication and factors, place value, conversions of length, weight and time, different types of mathematical relationship including proportionality and ratio in the context of drawing rectangles, translations and

For Peer Review

											reflections of regular polygons through the coordinates system
11	(Gadanidis, 2017)	Full integration	Integrated	Not validated	Scratch, Python, & Google's Blockly	Artefact assessment	Computer & Paper	Cognitive	Non-automatic	Agency, access, abstraction, automation and audience	Shapes, addition and subtraction mental math strategies, coordinate geometry, repeating patterns, area representations of fractions, area and perimeter relationships, symmetry and transformations, probability
12	(Gadanidis, Cendros, Floyd, & Namukasa, 2017)	Full integration	Integrated	Validated	Scratch, programmable robots & circuits, Python	In-class tasks	Computer	Cognitive	Non-automatic	Algorithms, abstraction	Geometry, coordinate geometry, probability, patterning & algebra, measurement & number sense
13	(Jaipal-Jamani & Angeli, 2017)	CTIMCI	Integrated	Not validated	LEGO® WeDo robotics kits	Knowledge test, survey, in-class tasks	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	Algorithms, sequencing, debugging, and flow of control in the form of loops	-
14	(Kopcha, McGregor, Shin, Qian, Choi, Hill, Mativo, & Choi, 2017)	CTIMCI	Integrated	Validated	Robots	interview, survey	Paper	Non-cognitive	Non-automatic	Algorithms, abstraction	Decimals and fractions, multiples and factors of whole numbers, coordinate grid, and algebraic operations
15	(Sung, Ahn, & Black, 2017)	Full integration	Separated	Not validated	Scratch Jr.	Knowledge test, in-class tasks	Computer & Paper	Cognitive	Non-automatic	Task decomposition, sequential thinking, procedural thinking, and commanding skills to operate a surrogate, abstraction, pattern recognition	Number line estimation, addition and subtraction
16	(Ziaeefard, Miller, Rastgaar, & Mahmoudian, 2017)	CTIMCI	Integrated	Not validated	Robotics Activities Attitudes Scale (RAAS)	survey	Paper	Non-cognitive	Non-automatic	Real-life value, social motivation, interest/attitude, confidence	-
17	(Altanis, Retalis, & Petropoulou, 2018)	CTIMCI	Separated	Not validated	Kinect games, & Dr. Scratch	Artefact assessment, observation, survey	Computer & Paper	Cognitive & Non-cognitive	Automatic	Flow control, abstraction, user interactivity, synchronization, parallelism and logic	Geometry
18	(Durak & Saritepeci, 2018)	CTIMCI	Integrated	Validated	Computational Thinking Skills Scale (CTSS)	Survey	Computer	Non-cognitive	Non-automatic	Creativity, algorithmic thinking, cooperation, critical thinking, problem solving	-
19	(Francis & Davis, 2018)	Full integration	Integrated	Validated	LEGO®EV3 robotics/ Lego Mindstorms EV3/NXT robots	In-class tasks	Computer & Paper	Cognitive	Non-automatic	Loops, sequence	Number, arithmetic and multiplication
20	(Gadanidis, Clements, & Yiu, 2018)	Full integration	Integrated	Validated	Scratch & Google Blockly	Artefact assessment	Computer	Cognitive	Non-automatic	Agency, access, surprise, and audience	Symmetry
21	(Harlow, Dwyer, Hansen, Iveland, & Franklin, 2018)	CTIMCI	Separated	Not validated	LaPlaya	Interview, observation, in-class tasks	Computer & Paper	Cognitive	Non-automatic	Sequencing, Breaking Down Actions, Event-based Programming, User-centered Design, Initialization, Costume Changes, Scene Changes, Message Passing, Loops, Sensing and Decisions, Variables, Program a Full Game, Program Another Full Game	Coordinate planes, negative numbers and percentages, and decimal numbers
22	(Kelly, Finch, Bolles, & Shapiro, 2018)	CTIMCI	Integrated	Not validated	BlockyTalky	Artefact assessment	Computer	Cognitive	Non-automatic	Variables and loops	-
23	(Leonard, Mitchell, Barnes-Johnson, Unertl, Outka-Hill, Robinson, & Hester-Croff, 2018)	CTIMCI	Separated	Validated	Computational Thinking survey	Survey, artefact assessment	Paper	Cognitive & Non-cognitive	Non-automatic	Understanding CT, self-efficacy, intrinsic motivation, integration of CT in classroom practice, and career relevance of CT	-
24	(Marcelino, Pessoa, Vieira, Salvador, & Mendes, 2018)	CTIMCI	Integrated	Not validated	Scratch	Knowledge test, survey	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	Movement, pen, conditions and cycles, variables, sensors, and interactions	-
25	(Martín-Ramos, Lopes, Lima da Silva, Gomes, Pereira da Silva, Domingues, &	CTIMCI	Separated	Validated	Arduino	Artefact assessment, survey	Computer	Cognitive & Non-cognitive	Non-automatic	Confidence, interest, gender, usefulness and professional	-

For Peer Review

	Ramos Silva, 2018)										
26	(Messer, Thomas, Holliman, & Kucirkova, 2018)	MICTCI	Separated	Validated	Bee-bot iPad app	In-class tasks, knowledge test	Computer & Paper	Cognitive	Non-automatic	Sequence, debugging	Addition and subtraction
27	(Pei, Weintrop, & Wilensky, 2018)	Full integration	Integrated	Not validated	Lattice Land	In-class tasks, interview	Computer	Cognitive	Non-automatic	Problem decomposition, pattern recognition, abstraction, debugging, using computers to solve problems, modeling & simulation, data practices	Discrete geometry
28	(Rico Lugo, Olabe, & Nino, 2018)	CTIMCI	Integrated	Validated	Digital Educational Material (DEM) "Evolution	Interview, observation, knowledge test, survey, in-class tasks	Computer & Paper	Cognitive & Non-cognitive	Automatic	Algorithmic thinking, mathematical thinking & critical thinking	Basic mathematical operations
29	(Roman-Gonzalez, Perez-Gonzalez, Moreno-Leon, & Robles, 2018)	CTIMCI	Separated	Validated	Computational Thinking Test (CTt), and Code.org coding problems	Knowledge test, in-class tasks	Computer & Paper	Cognitive	Non-automatic	Basic directions and sequences; Loops—repeat times; Loops—repeat until; If—simple conditional; If/else—complex conditional; While conditional; Simple functions.	-
30	(Sinclair, & Patterson, 2018)	Full integration	Integrated	Not validated	Sketchpad	Artefact assessment	Computer	Cognitive	Non-automatic	Decomposition, algorithmic thinking, abstraction, & pattern recognition, loops, conditionals, subroutines, variables, sequences, parallelism, events, operators, data	Geometric objects (segments, circles), operations (transformations) & numerical and algebraic objects (measurements, calculations & functions)
31	(Tran, 2018)	CTIMCI	Separated & integrated	Validated	Pretest and posttest on computational thinking	Knowledge test, survey, interview	Paper	Cognitive & Non-cognitive	Non-automatic	Sequence, algorithm, loops, debugging, conditional	-
32	(Valentine, 2018)	Full integration	Integrated	Not validated	Logo	In-class tasks	Computer	Cognitive	Non-automatic	Looping, sequencing	Geometry
33	(Arfe, Vardanega, Montuori, & Lavanga, 2019)	CTIMCI	Separated	Not validated	Code.org coding problems	In-class tasks, knowledge test	Computer & Paper	Cognitive	Non-automatic	Drag and drop, sequence, debugging, shapes, loops	-
34	(Bers, Gonzalez-Gonzalez, & Armas–Torres, 2019)	CTIMCI	Separated	Validated	KIBO robotics kit, & Solve-Its assessments	survey, knowledge test, observation, interview	Paper	Cognitive & Non-cognitive	Non-automatic	Sequencing, repeats, conditionals, and debugging, complex sequences including repeat loops, conditional statements, and nesting statements	-
35	(Buteau, Gueudet, Muller, Mgombelo, & Sacristan, 2019)	Full integration	Integrated	Not validated	Visual Studio IDE	In-class tasks, interview	Computer & Paper	Cognitive	Non-automatic	Loops and conditional statements, decision control	-
36	(Cendros Araujo, Floyd, & Gadanidis, 2019)	Full integration	Integrated	Validated	Scratch, Sphero, Python, & micro:bit	In-class tasks	Computer	Cognitive	Non-automatic	Logical and algorithmic thinking, abstraction, pattern recognition, automation, decomposition, debugging, and iteration.	Geometry, coordinate grids probability and Binomial Theorem
37	(Chiazzese, Arrigo, Chifari, Lonati, & Tosto, 2019)	CTIMCI	Separated	Not validated	Lego® Education WeDo 2.0 robotics kit, and Bebras tasks	In-class tasks, knowledge test, survey	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	(1) logically analyzing and organizing data; (2) representing data through abstraction and formal encodings; (3) using algorithmic thinking as a way to automatize solution; and (4) implementing simple algorithmic procedures (coding)	-
38	(Ching, Yang, Wang, Baek, Swanson, & Chittoori, 2019)	CTIMCI	Separated	Validated	Robots	In-class tasks, survey, interview	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	Sequence, sensors, conditional statements, loops	Angles, distances, multiplications, measuring distances, estimating angles
39	(Fanchamps, Slangen, Hennissen, & Specht, 2019)	Full integration	Separated	Validated	SRA-programming Lego NXT Mindstorms robots	Survey, in-class tasks knowledge test	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	Algorithmic thinking	-
40	(Farris, Dickes, & Sengupta, 2019)	Full integration	Integrated	Not validated	ViMAP	Interview, observation, knowledge test, in-class tasks,	Computer & Paper	Cognitive	Non-automatic	Modelling	Perimeter, area, and angles of polygons, speed as a rate of the distance traveled in a unit of time
41	(Grover, Jackiw, & Lundh, 2019)	CTIMCI	Separated	Validated	VELA activities & Scratch	Artefact assessment, knowledge test, interview,	Computer & Paper	Cognitive	Non-automatic	Repeating pattern, looping, modeling, variables (basic idea of variation, naming, datatypes), arithmetic expressions, conditionals, Boolean outcomes (TRUE/FALSE), Boolean logic, Boolean expressions, arithmetic and Boolean expressions, data types, string data, modeling	-
42	(Gunbatar & Bakırcı, 2019)	CTIMCI	Separated	Validated	Computational Thinking	Survey	Paper	Non-cognitive	Non-automatic	Creativity, algorithmic thinking, cooperativity, critical thinking, problem solving	-

For Peer Review

					Skills Scale (CTSS)						
43	(Israel & Lash, 2019)	Full integration	Separated & integrated	Validated	Scratch, and Code.org coding problems	In-class tasks	Computer	Cognitive	Non-automatic	Sequencing, looping, and conditional logic	Addition, geometry, fractions, number sense through number stories, area and volume, algebraic thinking, operations (multiplication and division)
44	(Lin, Wang, & Wu, 2019)	CTIMCI	Integrated	Validated	Python	Artefact assessment, knowledge test, survey, interview	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	Variables, if-else conditional statement, for loop, while loop	Mathematical equations
45	(Lockwood & De Chenne, 2019)	Full integration	Integrated	Not validated	Python	In-class tasks, interview	Computer	Cognitive	Non-automatic	Loops and Conditional	Permutations, Combinations
46	(Mesiti, Parkes, Paneto, & Cahill, 2019)	CTIMCI	Separated	Not validated	The Science Behind Pixar (Pixar) exhibition	Artefact assessment, interview, survey	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	Three dimensions: concepts (i.e., algorithms, patterns, abstraction, and decomposition), practices (i.e., debugging, creating, collaborating), and perspectives (i.e., sense of identity, empowered to ask questions about technology, relationship to the technical world)	-
47	(Miller, 2019)	Full integration	Integrated	Not validated	Scratch	In-class tasks, knowledge test	Computer & Paper	Cognitive	Non-automatic	Explore the coding blocks and symbols used in scratch, measuring in pixels, building a code, running a code, editing a code if there are errors, using the repeat/loop function	Mathematical patterns and structures
48	(Nam, Kim, & Lee, 2019)	Full integration	Integrated	Validated	TurtleBot	In-class tasks, knowledge test	Computer & Paper	Cognitive	Non-automatic	Procedural thinking, Algorithmic thinking, Efficient thinking, Procedural and efficient thinking, Patterns, Efficient thinking, algorithmic thinking, and disassembling	Categorization, patterns, numbers, measurement, diagramming, statistics
49	(Panskyi, Rowinska, & Biedron, 2019)	CTIMCI	Integrated	Not validated	Dr. Scratch	Artefact assessment, survey	Computer & Paper	Cognitive & Non-cognitive	Automatic	Abstraction and problem decomposition, logical thinking, synchronization, parallelism, synchronization, algorithmic notions of flow control, user interactivity, and data representation.	Geometry
50	(Papadakis & Kalogiannakis, 2019)	CTIMCI	Separated	Not validated	Dr. Scratch	Knowledge test, artefact assessment, survey, interview	Computer & Paper	Cognitive & Non-cognitive	Automatic	Logical thinking, data-information representation, user interactivity, flow control, abstraction and problem decomposition, parallelism, and synchronisation	-
51	(Rodriguez-Martinez, Gonzalez-Calero, & Saez-Lopez, 2019).	Full integration	Separated	Not validated	Computational Thinking Test (CTt), & Scratch	Knowledge test, in-class tasks	Computer	Cognitive	Non-automatic	Sequences, iterations, conditionals, and events-handling	Least common multiple (LCM) and the greatest common divisor (GCD)
52	(Saez-Lopez, Sevillano-García, & Vazquez-Cano, 2019)	Full Integration	Separated	Validated	mBot robot	In-class tasks, knowledge test, observation	Computer & Paper	Cognitive	Non-automatic	Computational concepts and practices: sequence, iteration (looping), conditional statements, threads (parallel execution), event handling, robot programming Participation & interactions: active methods, motivation, critical thinking skills, problem solving, interest in the subject, participation, encouragement, fun	Whole numbers, coordinates, negative numbers
53	(Vieira, Magana, Roy, & Falk, 2019)	CTIMCI	Separated	Validated	MATLAB®	In-class tasks, survey, knowledge test	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	(a) Creating the function—when a function is being declared, (b) Setting up problem parameters—when a value related to the problem is computed, (c) Setting up supporting variables—when other variables that are used to solve the problem are computed, (d) Validating the result—when the result is being computed and sometimes printed for validation, (e) Iterating—when a loop structure starts, (f) Validation—when an if clause starts, and (g) End of the function—when the function is being closed by an end statement	-
54	(Chongo, Osman, & Nayan, 2020)	Full integration	Separated	Not validated	CT test	Knowledge test	Paper	Cognitive	Non-automatic	Abstraction, algorithmic thinking, decomposition, evaluation, and generalization	-
55	(Cui, & Ng, 2020)	Full integration	Integrated	Not validated	Arduino	In-class tasks	Computer	Cognitive	Non-automatic	Sequence, repetition & conditionals, computational abstractions, as well as troubleshooting and debugging.	-
56	(Gleasman, & Kim, 2020)	Full integration	Integrated	Not validated	Scratch	Survey, interview	Paper	Non-cognitive	Non-automatic	Sequences, loops, events, parallelism, conditionals, operators, data	Counting and cardinality, geometry, measurement & data, numbers & operations, operations &

For Peer Review

											algebraic thinking
57	(Jurado, Fonseca, Coderch, & Canaleta, 2020)	CTIMCI	Integrated	Not validated	KIBO robotics kit	In-class tasks, survey, interview	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	Parameters, loops, and conditions	-
58	(Kaufmann, & Stenseth, 2020)	Full integration	Integrated	Not validated	Processing	In-class tasks	Computer	Cognitive	Non-automatic	Copying and adjustments, planned iterations, problem-solving	-
59	(Kong, Lai, & Sun, 2020)	CTIMCI	Separated	Validated	CT concept test and CT practice test	Knowledge test	Paper	Cognitive	Non-automatic	Sequences, events, conditionals, parallelism, naming (variables), operators, reusing & remixing, incrementalism & iteration, abstraction & modularization testing & debugging, algorithmic thinking	
60	(Munoz, Villarreal, Morales, Gonzalez, & Nielsen, 2020)	Full integration	Integrated	Not validated	Bee-Bot robot	Knowledge test	Paper	Cognitive	Non-automatic	Spatial location and cognition, motor skills and perception, logic and strategy	-
61	(Olteanu, 2020)	Full integration	Integrated	Not validated	Scratch	In-class tasks, knowledge test, survey	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	Variables, conditional statements, loops, functions, and arrays	-
62	(Piedade, Dorotea, Pedro, & Matos, 2020)	CTIMCI	Separated	Validated	m-Bot, LEGO®EV3 robotics/ Lego Mindstorms EV3/NXT robots, Zowi, Dash & Dot, Anprino, Codey Rocky, mBlock, Bitbloq, Blockly, Ardublockly	In-class tasks, interview, survey	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	Abstraction, decomposition, generalization, pattern recognition, algorithms, flow control, data representation, parallelism, synchronization, testing and debugging, mean, sequences input/output, arithmetic, relational, and logical operators, variables and constants, conditional structures, loops, procedures	-
63	(Rich, Yadav, & Larimore, 2020)	Full integration	Separated	Validated	CT Lesson Screener and CT Lesson Enhancer	In-class tasks	Paper	Cognitive	Non-automatic	Abstraction, decomposition, debugging, and patterns	-
64	(Sisman, Kucuk, & Yaman, 2020)	CTIMCI	Separated	Validated	Robotis Dream ER kits	Survey	Paper	Non-cognitive	Non-automatic	Variables, loops, decision structures, functions	-
65	(Sırakaya, Alsancak Sırakaya, & Korkmaz, 2020)	CTIMCI	Separated	Validated	Computational Thinking Skills Scale (CTSS)	Survey	Paper	Non-cognitive	Non-automatic	Creativity, algorithmic thinking, cooperativity, critical thinking, problem solving	-
66	(Stigberg, & Stigberg, 2020)	Full integration	Integrated	Not validated	Hour of Code, Lightbot, Scratch Jr. & Python	Observation, interview	Computer	Cognitive	Non-automatic	Mathematical skills, problem-solving skills, communication skills, algorithms and structures	-
67	(Suters, & Suters, 2020)	Full integration	Integrated	Not validated	LEGO®EV3 robotics/ Lego Mindstorms EV3/NXT robots	In-class tasks, survey, knowledge test	Computer & Paper	Cognitive & Non-cognitive	Non-automatic	Data practices, modeling and simulation practices, computational problem-solving practices, and systems thinking practices	-
68	(Valovicova, Ondruska, Zelenicky, Chytry, & Medova, 2020)	CTIMCI	Separated	Not validated	SmartMeasure app	In-class tasks	Computer	Cognitive	Non-automatic	Automation	Measuring the distance
69	(Ozcan, Cetinkaya,Goksun, & Kisbu-Sakarya, 2021)	CTIMCI (Math is one part of the programme)	Separated	Validated	Pretest and posttest on computational thinking	Knowledge test	Paper	Cognitive	Non-automatic	Sequence, algorithm, loop, debug, and conditional	Four operations, fractions, angles, shapes, circumference and area of shapes, patterns, symmetry, measuring units, charts, and tables in a play and learn format
70	(Chan, Looi, Ho, Huang, Seow, Wu, 2021)	Full integration	Integrated	Validated	Math+C worksheets, Spreadsheet	In-class tasks	Computer & Paper	Cognitive	Non-automatic	Pattern recognition, decomposition, algorithm design, and abstraction	Number patterns
71	(Polat, Hopcan, Kucuk, & Sisman, 2021)	CTIMCI	Separated	Validated	Computational Thinking Test (CTt), Computational thinking levels	Knowledge test, survey	Computer	Cognitive & Non-cognitive	Non-automatic	Basic directions and sequences; Loops—repeat times; Loops—repeat until; If—simple conditional; If/else—complex conditional; While conditional; Simple functions, creativity, algorithmic thinking, collaboration, critical thinking, problem solving	-

For Peer Review

					scale (CTLS)						
72	(Ng, & Cui, 2021)	Full integration	Integrated	Not validated	Arduino	In-class tasks	Computer	Cognitive	Non-automatic	Sequences, loops, conditionals, events, operators, variables, modeling, algorithmic thinking, abstracting and modularizing, testing and debugging, remixing and reusing	Arithmetic sequence, a geometric sequence, and a geometric series, prime and composite number
73	(Hava & Koyunlu Unlu, 2021)	CTIMCI	Separated	Validated	Computational Thinking Scale	Survey	Paper	Non-cognitive	Non-automaic	Creativity, algorithmic thinking, cooperation, critical thinking, problem solving	-

For Peer Review