Journal of Science Education and Technology *Vol. 28 2019 651-676*
https://doi.org/10.1007/s10956-019-09799-3

A Scoping Review of Empirical Research on Recent Computational Thinking Assessments

Maria Cutumisu, Cathy Adams, Chang Lu

Faculty of Education, University of Alberta, Canada

{cutumisu, caadams, clu4}@ualberta.ca

Abstract

Computational Thinking (CT) is regarded as an essential 21st-century competency and it is already embedded in K-12 curricula across the globe. However, research on assessing CT has lagged, with few assessments being implemented and validated. Moreover, there is a lack of systematic grouping of CT assessments. This scoping review examines 39 empirical studies published within the last five years, coded by the specific competencies outlined in existing CT frameworks, to identify and classify the key features of existing CT assessments. Results show that most studies target K-12 settings, focus on interventions that promote CT concepts and practices, adopt a quasi-experimental design, use selected-response items as the dominant testing form, and mainly assess algorithmic thinking, abstraction, problem decomposition, logical thinking, and data. Finally, few CT assessments have been validated in educational settings. Implications include identifying gaps in the CT assessment literature, deepening our understanding of the nature of CT, focusing on the validation of CT assessments, and guiding researchers and practitioners in choosing developmentally-appropriate CT assessments. Cognitive and educational implications for future research inquiry include the development of new assessment tools that comprehensively assess CT and its relation to learning.

Keywords: Computational Thinking; Computational Literacy; Coding; K-12 Education; Learning and Assessment; Programming and Programming Languages

Citation:

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*, 651–676. https://doi.org/10.1007/s10956-019-09799-3

Introduction

Computational Thinking (CT) has been steadily growing in popularity in the field of education since its resurgence in 2006. CT is regarded as a set of problem-formulation and problem-solving processes inspired by computing science practices that should be a part of all K-12 students' analytic toolkit in the 21st century, with applications well beyond computing. Ideas regarding the need for computation in students' lives, regardless of their discipline of studies, were introduced by Alan Perlis (Grover & Pea, 2013; Guzdial, 2008), as a "theory of computation", and by Seymour Papert (1980), as part of his constructionist framework. In 2006, in the wake of the 21st-century digital revolution, Jeannette Wing drew attention to CT, later redefining it as "an approach to solving problems, designing systems and understanding human behaviour that draws on concepts fundamental to computing" (Wing, 2008, p. 3717). Then, Wing rearticulated it 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" (Cuny et al., 2010). The first operational definition of CT as an approach to problem-solving included formulating problems so they can be solved using a computer; organizing and analyzing data logically; representing data through abstractions (e.g., models and simulations); automating solutions through algorithmic thinking; identifying, analyzing, and implementing efficient and effecting solutions; and generalizing and transferring problem-solving processes to solve many different problems (Barr et al., 2011, p. 21; ISTE, 2011, p. 1). CT dispositions or attitudes were also considered essential to problem solving, including confidence in the face of complexity; persistence; tolerance for ambiguity; and communication and collaboration (Barr et al., 2011, p. 21). Grover and Pea (2018) defined CT as the "thought processes involved in formulating a problem and expressing its solution(s) in such a

way that a computer – human or machine – can effectively carry out" (p. 21). In sum, although there is some blurring of boundaries between formal definitions of CT and CT frameworks, CT is generally defined as a problem-solving process concerned with a set of cognitive and meta-cognitive activities using computational strategies and creatively expressed as algorithms. Closely related to the work of computing scientists, CT involves problem formulation, decomposition, and solving that can be carried out with or without a machine (Wing, 2014). In a digital society, being versed in computational thinking is a *sine qua non* condition for success in navigating a complex and ubiquitous computing environment.

Grover et al. (2015) have emphasized the importance of assessments in CT-related research and discussed the challenges of successfully integrating CT in the K-12 curriculum when assessment issues are not effectively addressed. In general, assessments aim to assist students in developing their abilities and in enhancing their learning (Bransford et al., 1999). Despite the growing interests and increasing trials of introducing CT in K-12 education, there is a paucity of widely-accepted assessments that measure CT skills and extended abilities (Zhong et al., 2016, 2017). The variety of definitions and CT constructs proposed in previous literature further exacerbates the difficulty in measuring students' CT abilities with a valid tool. Researchers have emphasized that validated assessments are pillars of effective learning, as they measure students' progress in meeting the learning outcomes prescribed by the programs of study. As well, currently, a single assessment instrument cannot measure students' CT abilities across all grade levels. In sum, developing accurate and valid assessments of CT is still a challenge (Shute, Chen, & Asbell-Clark, 2017).

In the past decade, researchers around the world have been striving to develop and validate assessment tools for students in different grade levels. However, although there are

several reviews of CT instruction in the literature (Hsu et al., 2018; Jacob et al., 2018), the existing literature and studies on CT assessments have yet to be systematically reviewed and categorized. Thus, to provide a more organized view of current CT assessments, we conducted a scoping review on CT assessments implemented within the last five years to summarize all the recent empirical studies on CT assessment tools and extract their distinctive features and patterns (Adams, Cutumisu, & Lu, 2019). Specifically, to fill the gap in the CT literature, the present scoping review focused on answering the following research questions:

1. *What are the characteristics of the empirical studies on CT assessments (e.g., participants' countries and grade levels) and their assessment tools?*
2. *Are there any interventions that foster CT in these empirical studies?*
3. *What measurement instruments and CT competencies does each assessment adopt?*
4. *What is the format of each assessment tool?*

The rest of the article is organized as follows. First, the theoretical framework underpinning this research is presented, supported by a literature review of CT, which emphasizes the need to conduct research on the effects of CT in the curriculum. Then, the method employed to conduct the literature review is outlined, followed by the results that address four main research questions. The article concludes with a discussion, followed by limitations and recommendations for future research.

Theoretical Framework

Brennan and Resnick (2012) describe CT as a three-dimensional competency consisting of concepts, practices, and perspectives. CT *concepts* include sequences, loops, parallelism, events, conditionals, operators, and data. The four main CT *practices* include thinking in an incremental and iterative manner, testing and debugging, reusing and remixing, and abstracting

and modularizing. The CT concepts and practices under the computing paradigm constitute key cognitive abilities that focus on information processing (Denning & Freeman, 2009). The cognitive psychology literature conceptualizes cognitive abilities, which include problem-solving skills, in terms of logical reasoning and abstract thinking (Goldstein et al., 2015). CT *perspectives* are concerned with the evolving views of self and of the relationships with others and with their digital surround, manifested through expressing, connecting, and questioning. The main modalities of assessing these dimensions include project portfolio analyses, artefact-based interviews and observations, as well as design scenarios, all revolving around the *Scratch* visual programming tool (Brennan & Resnick, 2012).

Following Brennan and Resnick's (2012) framework, Zhong et al. (2016) linked CT competencies with each of the CT dimensions, emphasizing the CT perspectives dimension. This framework underlined their design of the *Three-Dimensional Integrated Assessment (TDIA)*, as shown in Table 3. Moreover, Atmatzidou and Demetriadis (2016) describe five CT skills (abstraction, generalization, algorithm, modularity, and decomposition) at the foundation of their "CT skills model" (p. 664).

Grover and Pea (2018) also proposed a *competency framework* recently, divided into CT concepts and CT practices. CT *concepts* include logic and logical thinking, algorithms and algorithmic thinking, patterns and pattern recognition, abstraction and generalization, evaluation, and automation (Grover & Pea, 2018, p. 23). CT *practices* or *approaches* include problem decomposition, creating computational artefacts, testing and debugging, iterative refinement (incremental development), as well as collaboration and creativity (Grover & Pea, 2018). Their approach was informed by the CS Principles of computational thinking put forward by the College Board and the National Science Foundation (NSF), developed around the seven big

ideas of computing: creativity, abstraction, data as a facilitator for creating new knowledge, algorithms, programming as a generative process, digital system interconnectivity fostering complex problem solving, and innovation through the application of computing to other fields (Grover & Pea, 2013). In this framework, Brennan and Resnick's CT perspectives have been included under CT practices (collaboration and creativity), while some of the Brennan and Resnick's CT practices, such as abstracting, were included under CT concepts in the Grover & Pea framework.

Overall, CT is largely seen as a form of problem-solving thought process that can be decomposed into several cognitive and non-cognitive dimensions under an integrated framework of CT abilities. More empirical studies on CT assessments favor the competency framework, though many tools only adopt a part of the framework, targeted at specific audiences (Grover & Pea, 2018; Moreno-León, Robles, & Román-González, 2015, 2016; Román-González, Pérez-González, Moreno-León, & Robles, 2018a; Román-González, Pérez-González, Moreno-León, & Robles, 2018b). The popularity of the competency framework could be attributed to its readiness of applying its underlying test constructs. The discussions regarding the CT constructs and working definitions are ongoing and are yet to be reconciled. However, the current research employs the framework proposed by Brennan and Resnick (2012) to ensure that all three dimensions are equally scrutinized: concepts, practices, and perspectives.

Literature Review

CT has been incorporated in various subjects of the curriculum, including computing science (Basnet et al., 2018; Basu et al., 2017; Psycharis & Kallia, 2017; Rojas-López & García-Peñalvo, 2018), robotics (Atmatzidou & Demetriadis, 2016; Chen et al., 2017), mathematics (Fessakis et al., 2013; Kahn et al., 2011), physics (Garneli & Chorianopoulos, 2018), and even

literature and arts (de Paula et al., 2018). The context in which students engage with CT is important, as it can influence the development of their cognitive abilities, and of their knowledge and skill acquisition in complex environments (Knee et al., 2006).

Several emerging reviews have summarized the state-of-art of CT teaching and learning. Lye and Koh (2014) examined 27 intervention studies on the CT introduction in K-12 education through programming and found that computational concepts were included in most of the studies selected. They then proposed that a constructionist, problem-solving environment should be designed to support computational practices and perspectives. Hsu et al. (2018) also conducted a meta review of articles on CT application, participants, teaching tools, and learning strategies from 2006 to 2017, reporting on current trends of how to learn and teach CT in K-12 settings. Their taxonomy of 19 different CT steps illustrates the wide discrepancy in the meaning and scope of CT across the current CT literature. More importantly, they offered five suggestions for further research on CT. Two of these suggestions are concerned with CT assessments. One suggestion supports effective and appropriate assessment of targeted groups of learning outcomes, which could inform future teaching strategies. Another suggestion calls for ascertaining students' learning achievement levels, to offer proper feedback and scaffolding to students.

An important aspect of choosing a CT assessment is the quality of that assessment. Specifically, CT assessments, like any other assessments, need to meet certain criteria and psychometric standards, such as validity and reliability, to ensure proper reporting of test-takers' achievements (McMillan, Hellsten, & Klinger, 2011). Also, the type of methods used to assess CT is important because some methods are more appropriate for measuring different levels of the cognitive domain in Bloom's Revised Taxonomy (Anderson et al., 2001). Researchers have

signaled a gap in the research literature regarding the validity and reliability of CT assessments (Grover & Pea, 2013), which has been a factor that has precluded such assessments from being widely used, especially across disciplines. As there is no generally accepted definition of what constitutes CT, validity and reliability cannot be effectively measured and assessments cannot be rigorously compared (Shute et al., 2017).

So far, most of the CT efforts have addressed issues related to the definition of CT and the promotion and development of CT tools for teaching and learning. However, less attention has been dedicated to assessments of CT. The next section discusses the method applied to review the main issues and characteristics related to CT assessments, the focus of this scoping review. Finally, the article organizes and presents the retrieved assessments according to the specific CT concepts, practices, and perspectives proposed by Brennan and Resnick's (2012).

Method

A scoping review is an exploratory method that systematically maps the literature on a topic, identifying key concepts, theories and sources of evidence, and addressing broader research topics where many different study designs might be applicable. It is not meant to provide an exhaustive account of research on a topic. Instead, it intends to explore "the extent, range, and nature of research activity in a topic area" (Pham, et al., 2014, p. 371). Following Arksey and O'Malley (2005) approach to conduct a scoping literature review, a five-stage searching and selecting method was employed that included: 1) identifying the research questions; 2) identifying relevant studies; 3) selecting studies; 4) charting the data; and 5) collating, summarizing, and reporting the results.

Identifying the Research Questions

Research on computational thinking is relatively new, with a particular paucity of publications on CT assessments. To thoroughly and systematically search for the most relevant studies, the databases of Figure 2 covering most of the journals of interest on computer science and education were chosen. The key purpose of this study is to identify and analyze assessment tools in recent empirical studies for various audiences and categorize their constructs and features. Therefore, this scoping review is guided by the following research questions:

5. *What are the characteristics of the empirical studies on CT assessments?* What are the participants' countries and grade levels, and what are the corresponding assessment tools employed? What measures does each tool adopt to assess CT competency?
6. *Are there any interventions to foster CT in these empirical studies?* What forms of intervention are adopted?
7. *What measurement instruments does each assessment adopt?* Does the instrument assess all three dimensions of CT competency? What skills does the instrument assess on each dimension of CT? What psychometric evidence (e.g., reliability, validity) has been provided for these assessments?
8. *What forms does each assessment tool take*: paper-based/computer-based, cognitive/non-cognitive, and automatic/non-automatic?

The answers to these questions will help identify the gaps in the CT assessment literature and aid teachers and researchers in choosing developmentally-appropriate, valid CT assessments. Findings will also help crystalize examples and best practices of learning environments and pedagogy surrounding CT.

Identifying Relevant Studies: Keyword Search

The keywords employed were "computational thinking" AND ("assessment" OR "measurement"). Results were restricted to peer-reviewed journal articles published in the last five years, from 2014 to 2018. The search process is described in Figure 1 and the results can be found in Table 1. The initial search returned 178 results.

Selecting Studies: Inclusion and Exclusion Criteria

Figure 1 illustrates the three-round article-selection process employed in this study. In the first round of article selection, 42 duplicates and 3 abstracts were deleted. In the second round of screening, articles which were irrelevant were excluded (n=36). In the third round of in-depth article reading and selection, studies that did not meet the sampling criteria were rejected. The following inclusion criteria were used to select articles:

1. Full peer-reviewed paper or article.
2. Empirical study conducted in an educational setting.
3. Empirical study focused on CT assessments with integration of CT abilities.
4. Empirical study (i.e., either quantitative or qualitative) and not a literature review, framework, proposal, or an introduction of an online learning platform or CT curriculum.
5. The CT assessment featured in the study targets students rather than pre-service/in-service teachers.

After inclusion and exclusion of the articles (Table 2), 39 articles were retained after three rounds of filtering (Table 3). To verify the requirements of article selection, a final round of proofreading was conducted to ensure that the retained articles were highly related to the research purposes outlined in this scoping review.

Data Analyses

The analysis of the articles selected with regards to their inclusion of CT concepts, practices, and perspectives was informed by Brennan and Resnick's (2012) framework. Two of the authors of the present article coded each assessment tool by identifying the specific CT competencies that it claimed to measure and the third author solved the discrepancies through follow-up discussions with the other authors. Some of the competencies were explicitly mentioned in the articles reviewed, while others were inferred from the items on the test. Although there are some competencies that are reported in different categories in the literature, depending on researchers' CT definition and framework employed, the discrepancies were resolved by referring to Brennan and Resnick's (2012) framework that was adopted in the current work, with some sub-categories of CT competencies adapted based on Grover and Pea (2018) and Zhong et al. (2016). Specifically, the CT Concepts resemble Grover and Pea's (2018), the CT Perspectives are more related to those of Zhong et al. (2016), and new sub-categories were also added to each dimension. The CT competencies assessed using each assessment tool presented in the articles reviewed are included in Table 7 and Table 8.

Results

1. Characteristics of the Empirical Studies and of CT Assessments

Classification of the Empirical Studies

Distribution of Articles on CT Assessments by Database. After the study selection phase, we identified 39 CT assessment-related studies extracted from SCOPUS, followed by SpringerLink, ScienceDirect, ERIC, IEEE Explore, and the ACM Digital Library, as shown in Figure 2.

Distribution of Articles on CT Assessments Over Time. The yearly distribution of the 39 articles illustrated in Figure 3 reveals an upward trend in the publication of empirical studies on CT assessment in recent years, consistent with the growing popularity and importance of CT.

Students' Countries and Grade Levels Featured in CT Assessment Studies. One aim of this scoping review was to identify the grade levels and contexts associated with each CT assessment tool. The assessment tool identified in each article as well as the country where the tool was tested and the grade level of the participants were illustrated in Table 4. Most participants were sampled from Europe (19) and North America (14), followed by Asia (4), and South America (2). The most representative European countries were Spain (6), Greece (4), and Turkey (4), as shown in Figure 5. The distribution of the samples is illustrated in Figure 12.

CT Assessment Tools

Tools such as *Dr. Scratch* (Garneli & Chorianopoulos, 2018; Moreno-León et al., 2015; Moreno-León et al., 2016), *Computational Thinking Scale* (*CTS*; Doleck, Bazelais, Lemay, Saxena, & Basnet, 2017; Durak & Saritepeci, 2018; Korkmaz, Cakir, & Ozden, 2017; Pellas & Vosinakis, 2018), and the *Computational Thinking test* (*CTt*; Román-González, Pérez-González, & Jiménez-Fernández, 2017; Román-González et al., 2018a; Román-González et al., 2018b) were used in more than one empirical study. Other assessments, both named and unnamed in the studies selected, were also designed in response to specific curricula, courses, or experiments. Several studies used more than one form of assessment. For instance, Pellas and Vosinakis (2018) employed a mix of qualitative and quantitative tools including an updated 15-item version of the *CTS* test, a questionnaire (Lahtinen, Ala-Mutka, & Järvinen, 2005), code-tracings in *Scratch* and *OpenSim*, think-aloud protocols, and semi-structured interviews with their

qualitative data analyzed in nVivo. In sum, there were 31 unique assessments found in the selected articles, as shown in Table 5.

Figure 4 shows the results for the different categories of grade levels. Assessments in K-12 settings account for the largest proportion (84.6%) of all the tools used in the studies reviewed. In total, there are 29 tools for K-12 students, 5 tools for undergraduate students, and 2 for Kindergarten children, with two assessment tools being repeated across categories, as they were designed for multiple grade levels. For example, *Dr. Scratch* can be used for K-12 students. On the other hand, some tools are designed for a specific targeted grade. *ScratchJr Solve It* Assessment (Strawhacker, Lee, & Bers, 2018) has only been used for kindergarten kids; and *Computational Thinking using Simulation and Modeling* (*CTSiM*; Basu et al., 2017) has only been used for sixth graders.

Classification of CT Assessments

From the studies selected, several types of assessments were identified: n=7 graphical programming environments such as block-based programming or web-based simulations authoring tools (n=3 using *Scratch*, n=3 using *Dr. Scratch*, and n=1 using *Alice*); n=11 block-based selected-response/constructed-response items; n=3 robotics/game-based tasks (n=1 using *AgentSheets* and *AgentCubes* as well as n=2 using the *Knowledge test* designed for the Robotics Curriculum; Chen et al., 2017); n=4 self-report scales (*Computational Thinking Scales*; Korkmaz et al., 2017; Doleck et al., 2017; Durak & Saritepeci, 2018); and n=10 combinations of several types of assessment methods (Grover et al., 2015; Leonard et al., 2016). Generally, selected-response formats, specifically n=14 multiple-choice items, were the most widely used. In addition, n=9 assessments also combined multiple measures, such as programming artefacts, surveys, and tangible tasks, to assess different facets of CT, as shown in Figure 6.

CT Skill Assessments: Block-based Assessments. Block-based programming languages such as *Scratch*, *Alice*, *App Inventor*, *Snap!*, as well as *AgentSheets* and *AgentCubes* are widely used in schools to scaffold and develop CT.

Scratch-based assessments. *Scratch*, designed at the MIT Media Lab, is especially popular, and has been used as the basis of several assessment tools. *Dr. Scratch* is an open-source web application assessment that determines students' programming competence (Moreno-León, Robles, & Román-González, 2015, 2016). It automatically analyzes *Scratch* programming artefacts, generating analytics (Moreno-León et al., 2015). *Dr. Scratch* targets seven dimensions of CT competency: abstraction and problem decomposition; parallelism; logical thinking; synchronization; flow control; user interactivity; and data representation. It is most frequently used as a formative rather than a summative assessment with *Scratch* projects (Moreno-León et al., 2016). Grover, Pea, and Cooper (2015) designed *'systems of assessments'* that involve both cognitive and non-cognitive (i.e., intrapersonal and interpersonal) aspects, formative and summative tests, open-ended programming assignments based on *Scratch*, and a transfer test that assessed deeper learning, all embedded in a massive open online course (MOOC) they developed on the edX platform. Specifically, formative multiple-choice questions tend to test students' understanding of CS concepts and constructs based on coding in *Scratch* to help familiarize the students with programming languages. A transfer test also required students to complete a project with *Scratch* (Grover et al., 2015). The *'System of assessments'* was embedded in a larger framework, *Foundations for Advancing Computational Thinking (FACT)*, with the purpose of preparing students to solve complex scenarios involving problem solving and algorithmic thinking skills (Grover, Pea, & Cooper, 2015).

Alice-based assessments. *Alice*, developed at Carnegie Mellon University, is another widely used block-based programming platform for CT assessment. *Alice* enables students to exercise their CT skills by creating 3D virtual worlds. The *Alice*-based *Fairy Assessment* developed at the University of California aims to measure students' understanding and use of abstraction, conditional logic, algorithmic thinking, and other CT concepts that middle-school students utilize to solve problems (Denner, Werner, Campe, & Ortiz, 2014; Werner, Denner, & Campe, 2014). Human raters score the students' solutions. However, its psychometric properties have not yet been reported (Denner et al., 2014). Zhong et al. (2016) also designed a *Three-Dimensional Integrated Assessment (TDIA)* framework. The constructs of *TDIA* are aligned with Brennan and Resnick's (2012) framework, aiming to assess CT concepts, practices, and perspectives with six tasks in semi-finished programming projects in *Alice* environment. Task 1 and 3 are closed tasks, task 2 and 4 are semi-open tasks, and task 5 and 6 are open creative tasks. Tasks 1-4 aim to measure students' sequence, testing, debugging, loops, parallelism, modularizing, and testing skills. Task 5 and 6 measure students' planning and designing skills, being creative and expressing their ideas with ease, abstracting and modelling, as well as testing and debugging. In these two open tasks, iteration and optimization, modularizing and reusing are also evaluated. Results show that students who completed closed and open tasks obtained significantly higher scores than students completing semi-open tasks, which might be due to semi-open tasks measuring more dimensions of CT competency.

The *Computational Thinking test* (*CTt*; Román-González et al., 2017, 2018a, 2018b) is one of the most popular block-based assessments, as it is not tied to a specific subject or programming language. *CTt* employs a multiple-choice question format. It is one of the few validated assessments and its design is informed by the practical guide from the international

standards for psychological and educational testing with application in middle school (AERA; APA; NCME, 2014). It is also informed by the practical guide to validating computing science knowledge assessments (Buffum et al., 2015) following the steps: identify the purpose of the test; define the construct of interest; prepare the test specifications and test format including determining number of responses each questions and total test length; generate the test items; conduct formal review on validity; pilot the test; and iteratively refine test items and re-test. It is designed to assess students between 12 and 14 years of age (7th and 8th grade) but it can also be used in lower (5th and 6th grade) and upper grades (9th and 10th grade; Román-González et al., 2017). It contains 28 items that take approximately 45 minutes to complete. Each item addresses one or more of the following seven computational concepts: basic directions and sequences (4 items); loops-repeat times (4 items); loops-repeat until (4 items); if-simple conditional (4 items); if/else-complex conditional (4 items); while conditional (4 items); and simple functions (4 items). Like *TDIA*, the CT concepts measured by CTt are consistent with Brennan and Resnick's (2012) framework and the Computer Science Teachers Association (CSTA) Computer Science Standards for 7th and 8th grade (Seehorn et al., 2011).

CT Skill Assessments: Robotics/Game-Based Assessments. Robotics and Game Design curricula are other constructionist pathways to introducing and assessing CT skills (Kafai & Resnick, 1996). For instance, *AgentSheets* and *AgentCubes* are web-based simulation game authoring tools (Leonard et al., 2016). They are often used as intervention tools to teach students how to program in object-based language and, ultimately, to promote CT. Leonard et al. (2016) collected and analyzed data such as field notes, screenshots, and photos of students' artefacts from *AgentSheets* and *AgentCubes* as evidence of CT skill development in a two-year longitudinal study. Other research groups have developed robotics curricula as means of teaching

students CT concepts and practices. Accordingly, different assessment tools have been implemented for these curricula or courses. For example, Chen et al. (2017) designed a new humanoid robotics curriculum based on a CT framework adapted from the CSTA (Seehorn et al., 2011) standards to assess fifth-grade students' CT skills on five CT components. The 23 items were contextualized in two types of CT applications: coding in robotics and reasoning of everyday events. The paper-based instrument, consisting of multiple-choice and open-ended questions, was administered as a pre-test and post-test measure in an elementary school where a new humanoid robotics curriculum was adopted by a fifth-grade class. Results showed that the instrument was validated and it had good psychometric properties and the potential to reveal students' CT learning challenges and growth. Bers et al. (2014) implemented the *TangibleK* Robotics Program in a Kindergarten class. In their study, students were required to complete a robotics or programming project and human raters were assigned to score each project. However, this assessment measures only a few CT skills, such as debugging, sequencing, correspondence between actions and instructions, and control flow.

CT Perspective Assessments: Self-Report Scales. Ten of the CT assessments in the articles reviewed were based on surveys (Durak & Saritepeci, 2018; Korkmaz et al., 2017; Pellas & Vosinakis, 2018), while two were based on individual interviews and observations (Weintrop & Wilensky, 2018; Zhong et al., 2017). The most frequently-adopted survey tool is a self-report questionnaire that measures students' perceptions of their own CT skills, the *Computational Thinking Scales* (*CTS*; Durak & Saritepeci, 2018; Korkmaz, Cakir, & Ozden, 2017; Yağcı, 2018). It uses a five-point Likert-type scale and consists of 29 items across five factors: creativity, algorithmic thinking, critical thinking, problem solving, and cooperativity. *CTS* items have been selected from other scales including: *"How Creative Are You?"* (Whetton &

Cameron, 2002), the *"Problem Solving Scale"* developed by Heppner and Peterson (1982), the *"Cooperative Learning Attitude Scale"*, and *"The Scale of California Critical Thinking Tendency"* (Korkmaz, 2012). Korkmaz and colleagues (2017) suggested that the validated *CTS* questionnaire can be used by participants of different education levels and age groups. Durak and Saritepeci (2018) applied Structural Equation Modelling (SEM) to determine the relationship between CT skills and thinking styles, academic achievement, and attitudes toward mathematics for the *CTS* assessment.

Psychometric Properties. We investigated the psychometric evidence of the CT assessments in the studies included in this review. Results revealed that only a few studies reported psychometric evidence (e.g., reliability and validity), as shown in Table 10. Most of the studies reviewed implemented their own assessment tools or adapted existing tools to serve their purposes, but did not report reliability or validity indicators (Fronza, Ioini, & Corral, 2017; Mouza, Marzocchi, Pan, & Pollock, 2016; von Wangenheim et al., 2018; Weintrop & Wilensky, 2018; Witherspoon, Higashi, Schunn, Baehr, & Shoop, 2017). Some studies claimed that the tool they adopted had been validated by previous studies (Durak & Saritepeci, 2018; Román-González et al., 2017; Román-González et al., 2018a; Román-González et al., 2018b). Of the articles selected, 21 included reliability or validity indicators.

Some studies validated their assessment tools by comparing the scoring results with other validated tools. Moreno-León, Robles, and Román-González (2016) correlated the results obtained from *Dr. Scratch* with two classic, well-acknowledged metrics: McCabe's *Cyclomatic Complexity* and *Halstead's metrics* to confirm its external validity. A positive moderate correlation was found in the study that validated the performance of *Dr. Scratch*. Romero, Lepage, and Lille (2017) analyzed the difference between scoring given by the automatic grader

Dr. Scratch and human raters of *#5c21* (the five key CT competencies in twenty-first century education detailed in the Appendix), and concluded that *#5c21* performed better, especially on the algorithmic complexity dimension.

Other studies examined the psychometric properties of their tests and provided validity evidence. In the *FACT Systems of Assessments*, Grover, Pea, and Cooper (2015) reported that the inter-rater reliability (Cohen's Kappa) of the final transfer test was 82.1%. Werner, Denner, and Campe (2015) also included multiple coders and peer debriefing to minimize bias and increase reliability by introducing a wider range of perspectives. Sáez-López, Román-González, and Vázquez-Cano (2016) conducted a comprehensive validation including both content and construct validity of their *Visual Blocks Creative Computing Test (VBCCT)* and of a questionnaire on students' learning processes and attitudes separately, through expert judgment and exploratory factor analysis. Jun et al. (2014) reported reliability (Cronbach's alpha) as their validity indicator, while Tsarava et al. (2018) reported both reliability (Cronbach's alpha) and convergent validity.

2. CT Interventions

Results presented in Table 6 show that most studies adopted a quasi-experimental design where an intervention was included.

In total, 24 interventions were conducted in the studies selected. In terms of educational settings, the main types of interventions include formal activities (e.g., courses of Computer Science, Mathematics, or Science) and informal activities (e.g., after-school workshops, and robotics or tangible game-design programs), as shown in Figure 7.

Most interventions (19) lasted between 2 to 6 months, one lasted less than one day, and, in two longitudinal studies (Leonard et al., 2016; Sáez-López, Román-González, & Vázquez-Cano, 2016), the interventions lasted more than one year, as shown in Figure 8.

Findings show that, overwhelmingly, the studies selected focused on programming and Science, Technology, Engineering, and Mathematics (STEM) fields. Thus, we found that a programming language was typically incorporated in the interventions since CT is assumed to be closely related to CS skills. For novice learners, a block-based visual programming language such as *Scratch* was used the most, as shown in Figure 9. However, even more conventional assessments employing selected-response items were used in novel ways, such as the *CTt*.

3. Measurement Instruments and the CT Dimensions Assessed: CT Concepts, Practices, and Perspectives

Different assessments are aligned with various theoretical frameworks of CT. In this scoping review, we draw on Brennan and Resnick's (2012) framework to summarize the CT concepts, practices, and perspectives targeted by each assessment tool. All the CT competencies (concepts, practices, and perspectives) assessed across all the studies, together with the assessment tools, are included in Table 7. The information extracted from the articles, this time by the assessment instrument, is organized in Table 8.

Results show that all assessments included in this study tested CT concepts. Figure 10 displays the most assessed CT concepts among the reviewed studies. Algorithms, algorithmic skills, and algorithmic thinking were assessed in 21 studies in different forms, followed by logic and logical thinking in 7 studies, and by data in 5 studies. Of the CT practices, problem decomposition, abstraction (found in 9 articles), testing, and debugging are the most assessed, as

shown in Figure 10. However, only seven studies assessed CT perspectives, with creativity and cooperativity being the main CT perspectives examined across all studies.

4. Forms of Assessment Tools

This research question aimed to identify the key features of each assessment tool. Specifically, it explored the CT assessments reviewed in terms of their form of administration: whether each assessment was computer-based or paper-based, automatic or non-automatic, whether it assessed cognitive skills like abstraction or non-cognitive skills like cooperation, and whether it employed a specific programming language. Results are presented in Figure 11 and Table 9.

Most of the assessments reviewed were computer-based but some were combinations of paper-based and computer-based tests. They were not tied to a specific programming language (i.e., they do not require students to know a specific programming language prior to taking the test). In addition, 23 tools assess only cognitive CT skills, while only 8 tools assess both cognitive and non-cognitive CT skills. Among the articles, only two kinds of automatic tools are identified (*Dr. Scratch* and *CodeMaster*). Most studies adopt non-automatic assessments.

Discussion, Conclusions, and Implications

Assessment is an essential part of learning and pedagogy, used to evaluate students, instructors, programs of instruction, and instrumental in providing needed feedback for improvement. If CT is to be adopted widely in the K-12 curriculum, CT assessments must align with the CT concepts, practices, and perspectives that are taught. This study systematically reviews and categorizes existing CT assessments published within the last five years. It focuses mainly on Brennan and Resnick's (2012) CT framework (i.e., CT concepts, practices, and

perspectives) but is also draws on the frameworks of Grover and Pea (2018) and Zhong et al. (2016).

Challenges. Several challenges prevent CT assessments from being widely developed and adopted. First, the many operational definitions of CT available in the literature made it difficult to establish and reconcile various approaches to measure the core features of CT (Grover et al., 2015). Second, a lack of a comprehensive widely-accepted CT framework as well as consensus regarding the membership of CT competencies into clear categories of concepts, practices, and perspectives hinders a systematic organization of CT assessments. Third, CT is a relatively new field that did not have the benefit of many iterations in both developing and evaluating CT programs of instruction. Although many countries around the world have adopted or are in the process of adopting a CT curriculum, many others are lagging behind (Hsu et al., 2018). Fourth, there is a paucity of teacher education programs that can prepare instructors to teach and assess CT effectively and, thus, there is a worrisome scarcity of CT teachers (Wilson et al., 2010). The ramifications of this challenge are deep, as there are not enough trained professionals to fill computing jobs. According to the Bureau of Labor Statistics (2018), employment of computing occupations will grow faster than the average for all occupations, by 13% from 2016 to 2026, projected to add about 557,100 new jobs. Fifth, CT is largely embedded in STEM field subjects, in part because CT has been viewed as central to STEM, leading to an easier translation of CT concepts, practices, and perspectives into these fields in comparison to social studies. Also, this may also be in part because of more efforts directed at preparing STEM teachers to engage with CT concepts and their implementation in practice (Denning, 2017).

Conclusion. The results of this scoping review show a breadth of methods employed to assess CT in both formal and informal environments but which are largely not yet validated to warrant their integration in the K-12 curriculum and their adoption by educational researchers.

First, the results indicate that although different types of CT assessment tools have been developed for grade levels ranging from Kindergarten to university students, most of them (84.6%) are applicable to K-12 settings. These results are consistent with the growing emphasis on CT curricula for primary-school and middle-school students. Second, in response to the many existing CT conceptual frameworks, various CT skills and extended abilities are measured by different assessments. However, algorithms, algorithmic skills and algorithmic thinking, abstraction, problem decomposition, logic and logical thinking, data, and communication are the abilities evaluated by most of the tools. This is to be expected, as these represent the keystones of CT. This finding also provides some insights into future designs of CT assessments. Third, most assessments are non-automatic, computer-based, and measure cognitive rather than interpersonal or intrapersonal CT skills. Although CT is closely related to computer programming, most assessments are generic (i.e., they do not use a specific programming language). This points to CT's fundamental focus on problem solving rather than on programming abilities and highlights the "computational literacy" emphasis on the cognitive and social aspects of computational thinking rather than on programming processes (DiSessa, 2001). Some assessments employ graphical programming environments that guarantee that a solution will not yield syntax errors (e.g., in environments such as *Scratch*, blocks that fit only specific other blocks will ensure that no programming syntax errors occur). Fourth, most assessments focus on interventions that promote CT concepts and CT practices, while perspectives are rarely evaluated. Fifth, most CT assessments are aligned with STEM-related subjects rather than non-STEM subjects. This is

reminiscent of traditionally using "computing as a medium" of expression for subjects such as mathematics and science (DiSessa, 2001; Kay & Goldberg, 1977). Sixth, results show that most studies adopt a quasi-experimental design. Seventh, most studies use selected-response items (e.g., multiple-choice questions) as their most dominant testing form, followed by programming projects, tangible tasks, and self-report scales. Finally, one of the concerns about existing CT assessments and an important finding of this review is that, although tools have been developed to assess CT skills, few of them have undergone validation or extensive, large-scale application.

Limitations. Even though the present scoping review involves a comprehensive examination of state-of-the-art CT assessments, it has several limitations. First, we focused on research regarding assessment tools published in the last five years, from 2014 to 2018. We made this decision because the field of CT has not fully developed since its resurgence in 2006 and research on assessment tools has not emerged until recent years. Second, although there are many articles proposing new frameworks, prototypes, or platforms for assessing CT, only empirical studies are included in this review. We chose only empirically-tested assessment tools to help us guide the pedagogical practices of CT and reveal hurdles that K-12 students face when engaging with CT early on, such as specific CT concepts, practices, and perspectives. Third, although our search expanded over several databases, we were limited by the scarcity of assessment tools empirically tested and by the information that was available about them.

Implications for Policy and Practice. This scoping review has identified a number of gaps in the literature that require the attention of several stakeholders, including researchers and practitioners. The findings of this scoping review reveal that, despite a large body of CT research, work on CT assessments is still in its infancy. Without diverse assessments for all CT concepts, practices, and perspectives, various age groups, and more coverage of the core school

subjects, the measurement of CT progress for teachers, students, and programs of instruction will not be reliable. Also, although the CT assessments examined were targeting various levels of the Bloom's Revised Taxonomy of the cognitive domain (Anderson et al., 2001), such as creating and evaluating (e.g., *Dr. Scratch*), analyzing and applying (e.g., *Bebras*), and understanding and remembering (e.g., *CTt*), more assessments targeting higher-order thinking skills need to be implemented. Hence, it will take a while until CT assessments could be reliably embedded in the curriculum. This issue is linked to the shortage of teachers ready to teach CT and assess their students accordingly. Thus, more effort needs to be invested in teacher-training programs focused on CT and in creating CT courses for pre-service teachers in the undergraduate curriculum (Rich & Yadav, 2019; Yadav et al., 2017; Yadav et al., 2019). Also, most assessments included in this review are more reflective of Europe and North America, so more attention and research need to be directed to the rest of the world, especially as CT has already been adopted into the curriculum of many countries around the world (e.g., Australia, New Zealand, Israel, etc.). Only a few attempts have been made to address CT perspectives. Future efforts need to focus on CT dispositions and attitudes that have the potential to aid the learning of CT skills and to reveal potential misconceptions in acquiring CT competencies. Finally, many more assessments need to address CT perspectives and examine their relation with learning outcomes, as well as potential gender differences.

Future research. Further efforts could focus on exploring the validity or the pedagogical usefulness of existing CT assessments. Moreover, there is still much research needed regarding the creation of a universal CT assessment that is suitable to be administered across ages and that is also age invariant. This could be accomplished by either defining rubrics that could objectively assess artefacts or by automatically assessing artefacts through decomposition into CT concepts,

practices, and perspectives. These findings provide insights that will inform the design of future CT assessments that would help researchers and practitioners to better measure CT skills across development, to relate them with cognitive ability tests, and to create appropriate CT interventions to improve students' computational thinking skills and attitudes.

Future work could also focus on comparing new CT assessments against new theoretical developments, especially as new CT definitions, frameworks, and models continue to be proposed in the related literature. As the present study shows, assessments can reveal commonalities among frameworks, highlight potential weaknesses, and suggest changes to existing frameworks. This approach can ensure a more inclusive representation of CT theorizing in future assessments of computational thinking.

Acknowledgements

We would like to thank the Callysto project and the Government of Canada CanCode program: Innovation, Science and Economic Development Canada, as well as our partner organizations, Cybera and the Pacific Institute for the Mathematical Sciences (PIMS).

Compliance with Ethical Standards

This study is not based on empirical data. Instead, it is a review of several published empirical studies. Therefore, no ethics application nor informed consent were necessary to conduct this study.

Conflict of Interest. The authors declare that they have no conflict of interest.

Figures

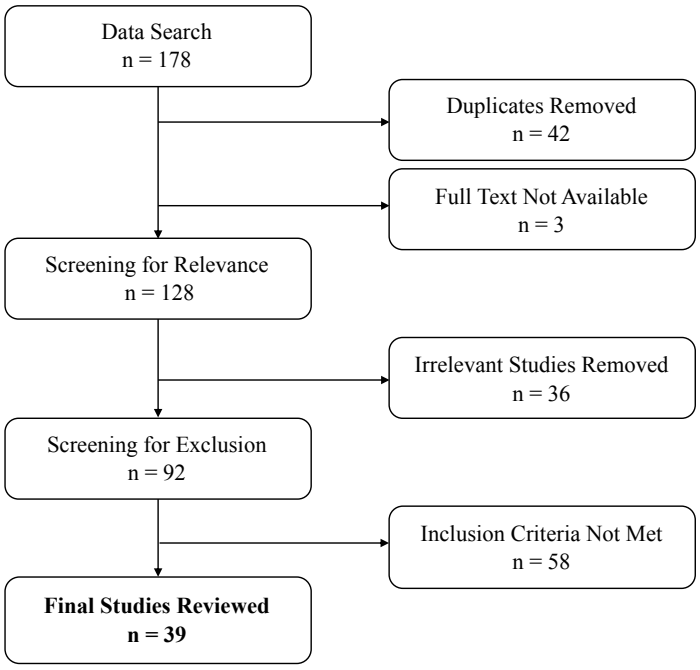

Figure 1. Flow diagram for the article selection process

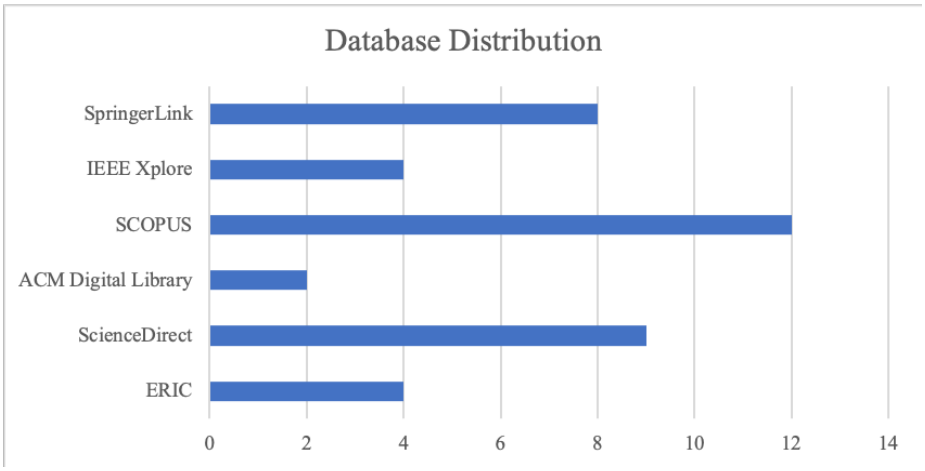

Figure 2. Database Distribution: most research was published in SpringerLink, followed by ScienceDirect and SCOPUS, ERIC, and, finally, ACM Digital Library and IEEE Explore.

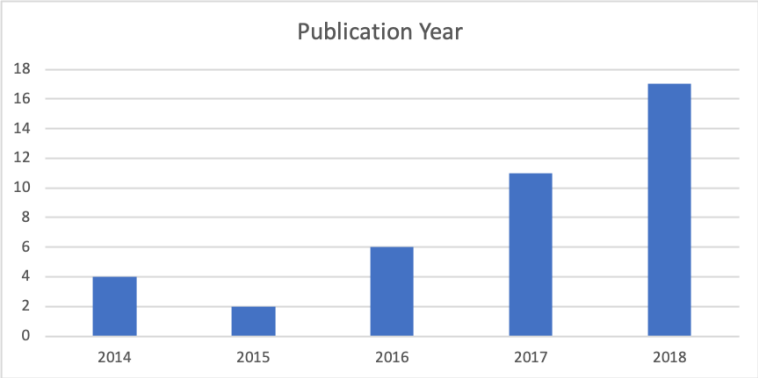

Figure 3. Distribution of the selected articles by publication year.

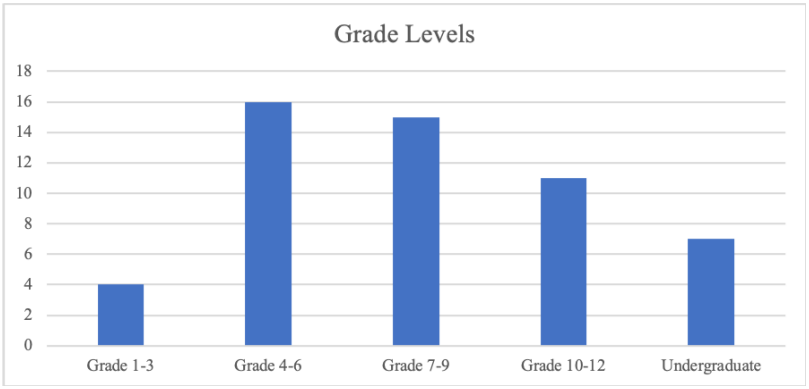

Figure 4. Participants' grade levels ranging from kindergarten to university.

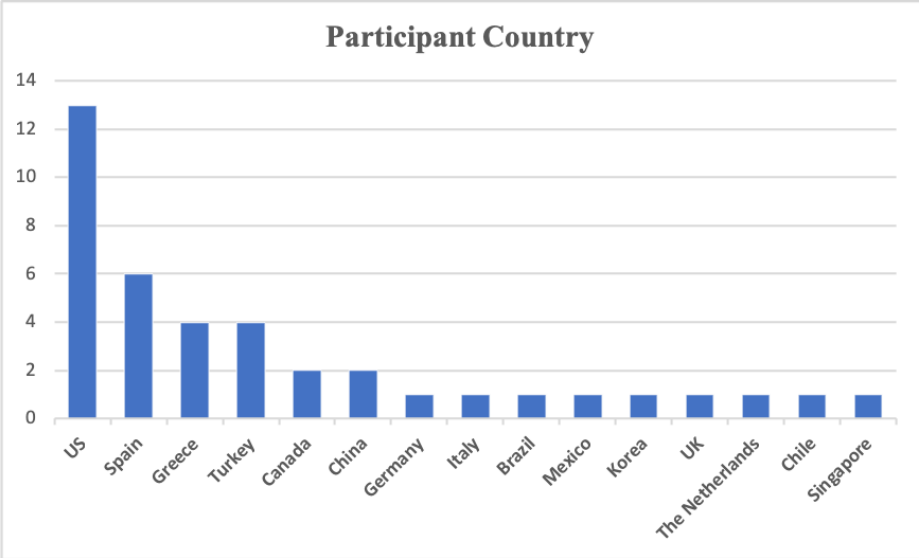

Figure 5. The countries of the participants in the empirical studies reviewed.

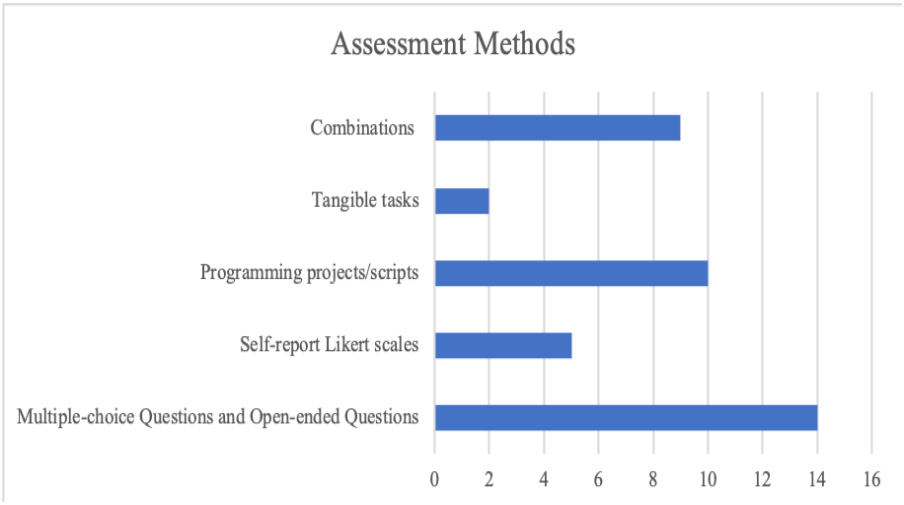

Figure 6. CT assessment methods: multiple-choice questions (MCQ), constructed-response questions (CRQ), self-report Likert scales, programming, tasks, and combinations of these items.

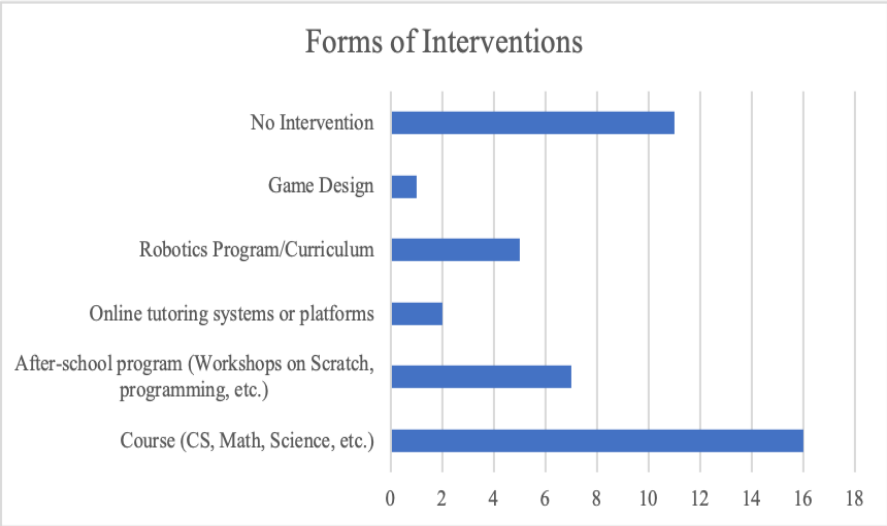

Figure 7. Forms of intervention used in the empirical studies selected.

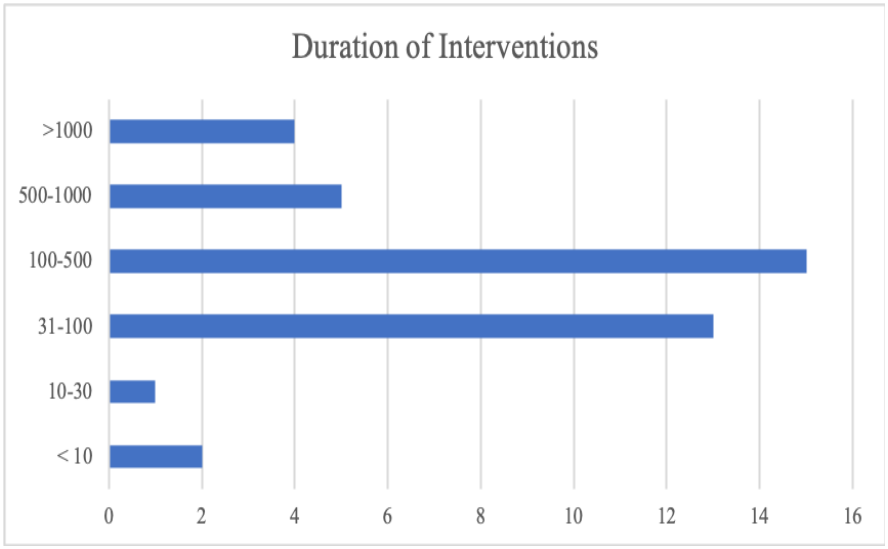

Figure 8. The duration of the interventions included in the empirical studies examined.

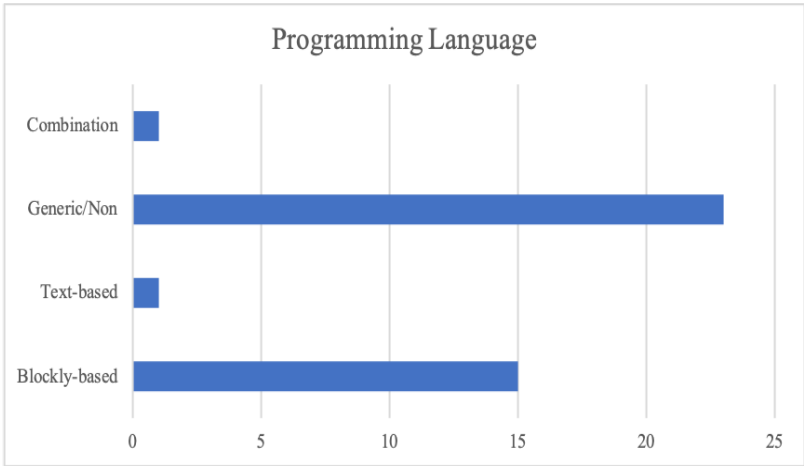

Figure 9. Types of programming languages used in the intervention.

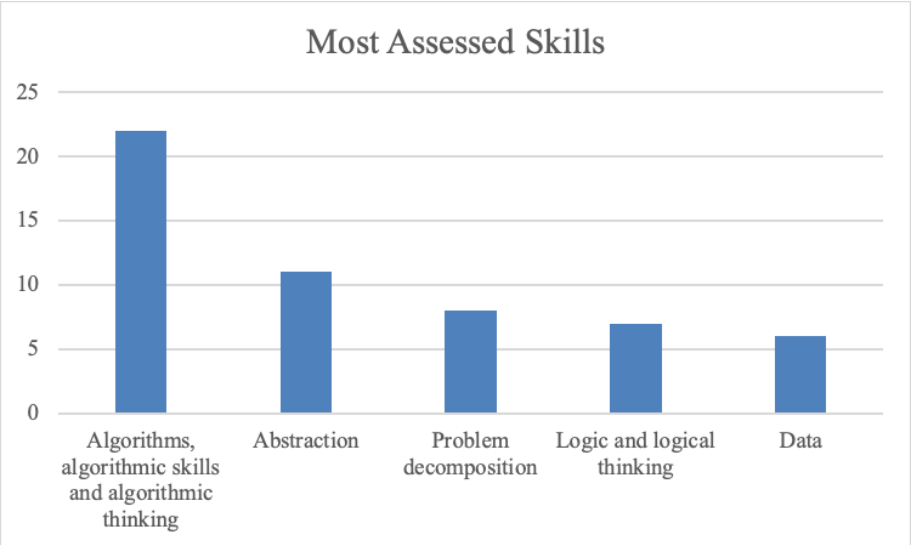

Figure 10. The CT skills that were most assessed in the empirical studies examined.

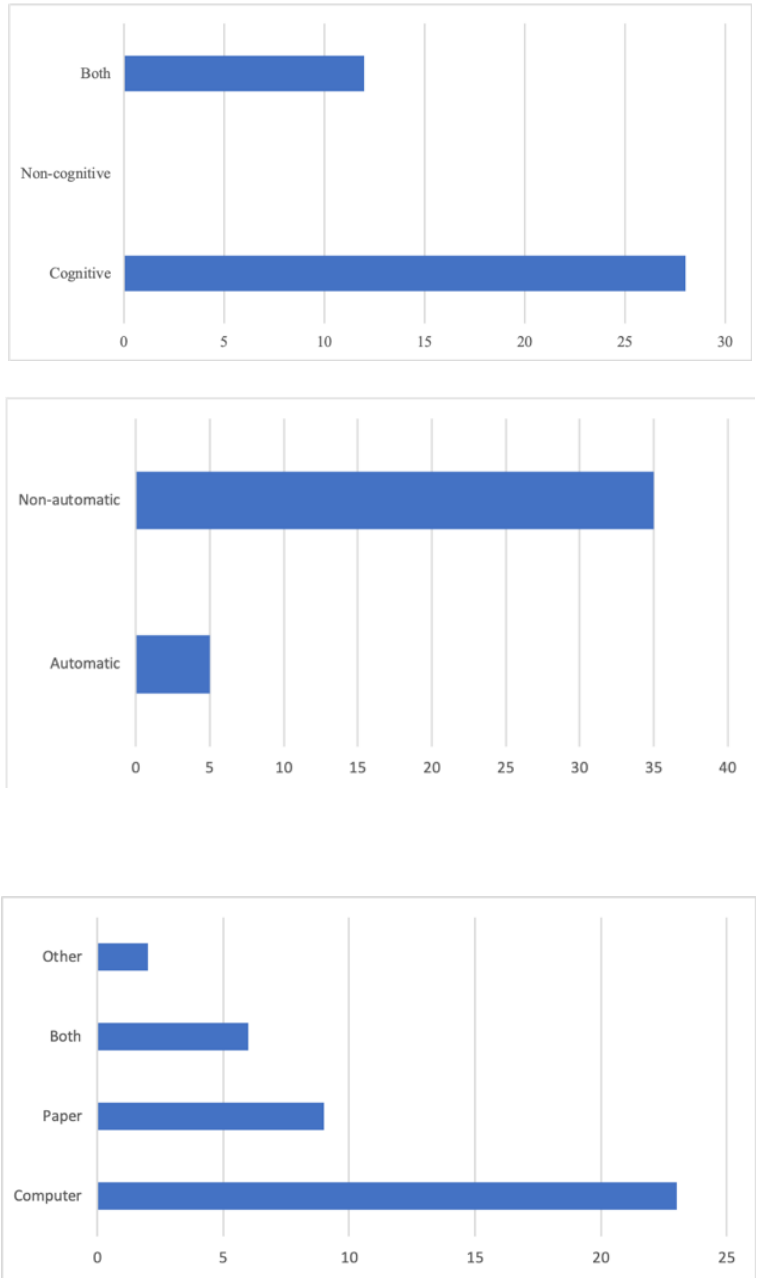

Figure 11. The delivery form of the CT assessment tools.

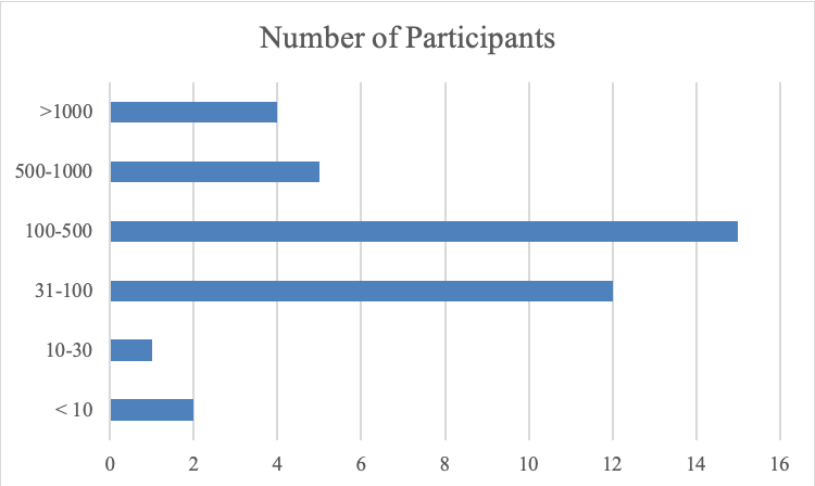

Figure 12. The number of participants included in all the articles selected.

Tables

Table 1. The initial search results.

Database	N	Other restrictions
SpringerLink	24	Search under the discipline: <Computers and Education>
IEEE Xplore	8	
SCOPUS	27	
ACM Digital Library	13	
ScienceDirect	71	“Review articles and Research articles”
ERIC	10	Scholarly "Peer reviewed" Journals
Web of Science	25	
Total	178	

Table 2. Article screening and criteria for inclusion and exclusion.

Screening for exclusion	**Total**
Not a journal article	3
Duplicates among different databases	42
Not related to education	19
Not related to assessment	17
Involved education or assessment without a real integration of computational thinking	29
Involved computational thinking but the assessments used did not assess CT skills	1
An introduction or evaluation of an e-learning tool or online tutoring system	2
A literature review or theoretical/conceptual framework but not an empirical study	18
Assessing teachers' not students' CT skills	8
Total after exclusion	39

Table 3. The articles selected for this review, their venues, and their assessment tools.

	Year	Authors	Journal Venue	Assessment Tool
[1]	2015	Moreno-León, Robles, & Román-González, 2015	Revista de Educación a Distancia	Dr. Scratch
[2]	2016	Moreno-León, Robles, & Román-González, 2016	IEEE Global Engineering Education Conference: EDUCON	Dr. Scratch
[3]	2018	Garneli & Chorianopoulos, 2018	Interactive Learning Environments	Dr. Scratch
[4]	2017	Romero, Lepage, & Lille, 2017	International Journal of Educational Technology in Higher Education	#5c21(the five key competencies for the 21st-century education)
[5]	2017	Basu, Biswas, & Kinnebrew, 2017	User Modeling and User-Adapted Interaction	Computational Thinking using Simulation and Modeling (CTSiM)
[6]	2015	Grover, Pea, & Cooper, 2015	Computer Science Education	FACT 'Systems of Assessments'
[7]	2016	Sáez-López, Román-González, & Vázquez-Cano, 2016	Computers & Education	Visual Blocks Creative Computing Test (VBCCT)
[8]	2018	Strawhacker, Lee, & Bers, 2018	International Journal of Technology and Design Education	ScratchJr Solve It Assessment
[9]	2016	Zhong, Wang, Chen, & Li, 2016	Journal of Educational Computing Research	TDIA (Three-Dimensional Integrated Assessment)
[10]	2014	Werner, Denner, & Campe, 2014	ACM Transactions on Computing Education	Game Computational Sophistication (GCS)
[11]	2014	Denner, Werner, Camp, & Ortiz, 2014	Journal of Research on Technology in Education	The Fairy Assessment
[12]	2017	Korkmaz, Cakir, & Ozden, 2017	Computers in Human Behavior	CTS (Computational Thinking Scales)
[13]	2017	Doleck, Bazelais, Lemay, Saxena, & Basnet, 2017	Journal of Computers in Education	CTS (Computational Thinking Scales) Survey
[14]	2018	Pellas & Vosinakis, 2018	Education and Information Technologies	Scratch, OpenSim, Lahtinen survey, modified CTS (Computational Thinking Scales), think-alouds, and semi-structured interviews
[15]	2018	Durak & Saritepeci, 2018	Computers & Education	CTS (Computational Thinking Scales)
[16]	2017	Román-González, Pérez-González, & Jiménez-Fernández, 2017	Computers in Human Behavior	CTt (Computational Thinking Test) Survey

[17]	2018	Román-González, Pérez-González, Moreno-León, & Robles, 2018a	Computers in Human Behavior	CTt (Computational Thinking Test)
[18]	2018	Román-González, Pérez-González, Moreno-León, & Robles, 2018b	International Journal of Child-Computer Interaction	CTt (Computational Thinking Test)
[19]	2014	Bers, Flannery, Kazakoff, & Sullivan, 2014	Computers & Education	Robotics program
[20]	2016	Leonard et al., 2016	Journal of Science Education and Technology	AgentSheets and AgentCubes
[21]	2016	Atmatzidou & Demetriadis, 2016	Robotics and Autonomous Systems	Robotics-related tools
[22]	2017	Witherspoon, Higashi, Schunn, Baehr, & Shoop, 2017	ACM Transactions on Computing Education	MCQ format assessment (Robotics)
[23]	2017	Chen et al., 2017	Computers & Education	An instrument with 23 items (15 MCQ and 8 CRQ)
[24]	2016	Mouza, Marzocchi, Pan, & Pollock, 2016	Journal of Research on Technology in Education	Multiple-choice content assessment (Scratch)
[25]	2014	Jun, Han, Kim, & Lee, 2014	Educational Assessment, Evaluation and Accountability	ICT Literacy Test Tool (Survey)
[26]	2018	von Wangenheim et al., 2018	Informatics in Education	CodeMaster
[27]	2017	Psycharis & Kallia, 2017	Instructional Science	Cornell Critical Thinking test (CCTT)
[28]	2017	Fronza, Ioini, & Corral, 2017	ACM Transactions on Computing Education	Project Analysis and Artefact-based Interviews
[29]	2017	Flanigan, Peteranetz, Shell, & Soh, 2017	Contemporary Educational Psychology	Computational thinking knowledge test
[30]	2018	Tsarava, Moeller, & Ninaus, 2018	International Journal of Serious Games	Board Game: Crabs & Turtles
[31]	2018	Rojas-López & García-Peñalvo, 2018	IEEE-RITA	Bebras & Talent Search Computing Olympiad
[32]	2018	Weintrop & Wilensky, 2018	International Journal of Child-Computer Interaction	Pencil.cc
[33]	2018	de Paula, Burn, Noss, & Valente, 2018	International Journal of Child-Computer Interaction	MissionMaker
[34]	2018	Yağcı, 2018	Education and Information Technologies	CTS (Computational Thinking Scales) Survey
[35]	2018	Looi et al., 2018	Computer Science Education	Sorting Algorithms artifacts & Qualitative Comparative Analysis (QCA)

[36]	2018	Kong, Chiu, & Lai, 2018	Computers & Education	Programming empowerment survey
[37]	2018	Munoz et al., 2018	IEEE Access	CTt
[38]	2018	Rijke et al., 2018	Informatics in education	Abstraction and decomposition tasks
[39]	2018	Bati et al., 2018	Cogent Education	Computational thinking test (CTT)

Table 4. Information on student sample size, grade level, and country.

	Assessment Tool	N	Country	Grade Level
[1]	Dr. Scratch	109	Spain	5-8 (10-14-year olds)
[2]	Dr. Scratch	95	Spain	Not indicated
[3]	Dr. Scratch	44	Greece	9
[4]	#5c21	120	Canada	Undergraduate
[5]	CTSiM	98	US	6
[6]	FACT 'Systems of Assessments'	54	US	5-8
[7]	Visual Blocks Creative Computing Test (VBCCT)	107	Spain	5-6
[8]	ScratchJr Solve It Assessment	222	US	Kindergarten
[9]	TDIA (Three-Dimensional Integrated Assessment)	144	China	6
[10]	Game Computational Sophistication (GCS)	325	US	5-8
[11]	The Fairy Assessment	320	US	5-8
[12]	CTS (Computational Thinking Scales)	1309	Turkey	Undergraduate
[13]	CTS (Computational Thinking Scales)	104	Canada	Pre-university
[14]	CTS (Computational Thinking Scales)	785	Turkey	High school
[15]	Scratch, OpenSim, and CTS	50	Greece	High school
[16]	CTS (Computational Thinking Scales)	156	Turkey	5-12
[17]	CTt (Computational Thinking test)	1251	Spain	5-10
[18]	CTt (Computational Thinking test)	1251	Spain	5-10
[19]	CTt (Computational Thinking test)	314	Spain	8-9
[20]	Robotics program	53	US	Kindergarten
[21]	AgentSheets and AgentCubes	124	US	5-8
[22]	Robotics related tools	164	Greece	9 and 12
[23]	MCQ format assessment (Robotics)	564	US	6-8
[24]	An instrument with 23 items (15 MCQ and 8 CRQ)	767	US	5
[25]	Multiple-choice content assessment (Scratch)	41	US	4-6
[26]	ICT Literacy Test Tool	40072	Korea	3-6
[27]	CodeMaster	16	Brazil	12
[28]	Cornell Critical Thinking test (CCTT)	66	Greece	3rd-year high school
[29]	Project Analysis and Artefact-based Interviews	42	Italy	6
[30]	Computational thinking knowledge test	443	US	Undergraduate
[31]	Board Game: Crabs & Turtles	36	Germany	8-9-year-olds
[32]	Bebras & Talent Search Computing Olympiad	360	Mexico	1st-year undergraduate
[33]	Pencil.cc	90	US	High school
[34]	MissionMaker	2	UK	9
[35]	Sorting Algorithms artifacts & QCA (Qualitative Comparative Analysis)	35	Singapore	9
[36]	Programming empowerment survey	287	Hong Kong (China)	4-6
[37]	CTt	7	Chile	7-12
[38]	Abstraction and decomposition tasks	200	The Netherlands	1-6
[39]	Computational thinking test (CTT)	104	Turkey	8

Table 5. The unique assessment tools retrieved from the articles selected for this review.

	Reference	First Year	Assessment Tool
1	[1] [2] [3]	2015	Dr. Scratch
2	[4]	2017	#5c21 (the five key competencies for the 21st-century education)
3	[5]	2017	Computational Thinking using Simulation and Modeling (CTSiM)
4	[6]	2015	FACT 'Systems of Assessments'
5	[7]	2016	Visual Blocks Creative Computing Test (VBCCT)
6	[8]	2017	ScratchJr Solve It Assessment
7	[9]	2016	TDIA (Three-Dimensional Integrated Assessment)
8	[10]	2014	Game Computational Sophistication (GCS)
9	[11]	2014	The Fairy Assessment
10	[12] [13] [15] [34]	2017	CTS (Computational Thinking Scales)
11	[14]	2018	Scratch, OpenSim and CTS (Computational Thinking Scales)
12	[16] [17] [18] [37]	2017	CTt (Computational Thinking Test) Survey
13	[19]	2014	Robotics program
14	[20]	2016	AgentSheets and AgentCubes
15	[21]	2016	Robotics related tools
16	[22]	2017	MCQ format assessment (Robotics)
17	[23]	2017	An instrument with 23 items (15 MCQ and 8 CRQ)
18	[24]	2016	Multiple-choice content assessment (Scratch)
19	[25]	2014	ICT Literacy Test Tool (Survey)
20	[26]	2018	CodeMaster
21	[27]	2017	Cornell Critical Thinking test (CCTT)
22	[28]	2017	Project Analysis and Artefact-based Interviews
23	[29]	2017	Computational thinking knowledge test
24	[30]	2018	Board Game: Crabs & Turtles
25	[31]	2018	Bebras & Talent Search Computing Olympiad
26	[32]	2018	Pencil.cc

27	[33]	2018	MissionMaker
28	[35]	2018	Sorting Algorithms artifacts & QCA (Qualitative Comparative Analysis)
29	[36]	2018	Programming empowerment survey
30	[38]	2018	Abstraction and decomposition tasks
31	[39]	2018	Computational thinking test (CTT)

Table 6. Interventions and assessment methods.

	Intervention	Educational Setting	Subject	Duration	Study Design	Tasks
[1]	Workshop on Dr. Scratch	Informal	NA	1 hour	Quasi-experimental (pre-post-test)	Scratch programming projects
[2]	NA	NA	NA	NA	Correlational (validation study)	Scratch programming projects
[3]	Science project on electric circuit through simulation constructions and video game design	Formal	IT/Science	5 weeks (2 hours per week)	Quasi-experimental (between-group pre-post-test)	Scratch programming projects
[4]	NA	NA	NA	NA	Quasi-experimental (between-group pre-post-test); Correlation with external assessments (Dr. Scratch)	Scratch programming projects
[5]	Computer class in school using Scratch programming	Formal	Science	3 weeks	Quasi-experimental	Several tasks and paper-based pre-post tests
[6]	FACT curriculum in a face-to-face classroom context, and a blended in-class learning involving Stanford's Open edX MOOC platform	Formal	Computer science	7 weeks for two iterations	Quasi-experimental (pre-post-test)	Formative and summative quizzes and tests, artefact assessment (directed and open-ended programming assignments in Scratch), and a transfer test. (Computational knowledge test, PFL test, Prior experience survey, CS perceptions survey)
[7]	Visual Programming Languages using Scratch in classroom practice	Formal	Science/Arts	2 years	Mixed methods	Multiple-choice questions for CT concepts and practices, Likert scales for attitudes
[8]	ScratchJr	Formal	Computer science	4-8 lessons for 2 months using ScratchJr	Mixed methods	Video-based multiple-choice and open-response questions
[9]	School-based curriculum learning storytelling by programming	Formal	Computer Science	18 weeks, 40 minutes per week	Experimental	Six tasks in semi-finished programming projects, rated by reflection reports, creative design reports, rubrics, and coding schemes

[10]	Voluntary technology class during or after school	Formal/ Informal	Technology class	10 hours during a semester	Case study	Programming projects (rating scales)
[11]	Solo or pair programming in a three-stage acquisition progression called Use–Modify–Create	Formal/Informal	Game programming	20 hours during a semester	Experimental	Alice programming knowledge questions, attitude surveys, performance assessment of project programming
[12]	NA	NA	NA	NA	Correlational (validation study)	CTS, self-report 5-point Likert scales
[13]	NA	NA	NA	NA	Correlational (Structural Equation Modeling)	CTS, self-report 5-point Likert scales
[14]	Workshops on computer games simulations	Formal/ Informal	Computer Science (computer games simulations)	4 weeks with 6 sessions (30 min each)	Quasi-experimental (pre-post-test)	Simulation game task and CTS
[15]	NA	NA	NA	NA	Correlational (Structural Equation Modeling)	CTS, self-report 5-point Likert scales
[16]	NA	NA	NA	NA	Correlational (validation study)	Multiple-choice questions: CTt
[17]	NA	NA	NA	NA	Correlational (validation study)	Multiple-choice questions: CTt
[18]	Code.org and Khan Academy	Informal	NA	12 weeks	Correlational (validation study)	Multiple-choice questions: CTt
[19]	The TangibleK Robotics Program	Formal	Robotics curriculum	20 hours of classwork	Experimental	Projects making robot/program
[20]	Robotics and Gaming, LEGO EV3, Robotics, MINDSTORMS, and Scalable Game Design software (AgentSheets and AgentCubes)	Formal/ Informal	NA	3 years longitudinal	Quasi-experimental (pre-post-test)	Self-efficacy survey and a STEM attitude and career survey. Programming on Scalable Game Design
[21]	Robotics related in-class training seminars	Formal	NA	2 hours per week for 11 weeks (sessions)	Quasi-experimental	Profile Questionnaire, Intermediate Questionnaires, Student Opinion Questionnaire, Think-Aloud Protocol, interview, observation
[22]	Virtual Robotics Programming Curriculum	Formal	Robotics curriculum	5 days a week for 5-7 weeks	Quasi-experimental (pre-post-test)	Multiple-choice questions

[23]	Robotics curriculum	Formal	Robotics curriculum	45-60 minutes every week for six months	Quasi-experimental (pre-mid-post-test), validation study	Multiple-choice and open-ended questions
[24]	After-school CS program	Informal	Computer science (Informal)	9 weeks	Quasi-experimental (pre-post-test)	Multiple-choice questions, Scratch programming projects, and attitude survey
[25]	NA	NA	NA	NA	Correlational	Multiple-choice questions
[26]	NA	NA	NA	NA	Multi-method research	Programming projects in App Inventor and Snap!
[27]	Computer Science course	Formal	Mathematics/Physics	A term (90 min in-class instruction)	Quasi-experimental (pre-post-test)	Multiple-choice questions and questionnaires
[28]	Agile Software Engineering Methods	Formal	CS/History/Art	A term (90 min in-class instruction)	Quasi-experimental	Programing projects and artefacts
[29]	CS1 Course	Formal	Computer science	60 hours (4 hours per week for 15 weeks)	Quasi-experimental (pre-post-test)	Multiple-choice questions
[30]	NA	NA	NA	One term	Quasi-experimental (pre-post-test)	Card games unplugged adventures
[31]	Classroom intervention	Formal	NA	NA	Quasi-experimental (pre-post-test)	Task questions, multiple-choice questions, transfer
[32]	Pencil.cc programming environment in an introductory programming course	Formal	Computer science (Introductory programming course)	5 weeks	Quasi-experimental	Interviews and data from Pencil.cc
[33]	After-school program: Playing Beowulf	Informal	NA	6 1-hour sessions for 2 months	Ethnographic research	Projects on MissionMaker
[34]	NA	NA	NA	NA	Correlational	Self-report 5-point Likert scales

[35]	Unplugged CT activities in school computing class	Formal	Computing	1 class	Mixed methods	Pseudo code artifacts, videos, and observations
[36]	Computational thinking curriculum	Formal	Computer science	1 school year	Survey (SEM)	self-report 5-point Likert scales
[37]	Game Building Workshop	Informal	Computational thinking	5 days with 3 hours each day	Quasi-experimental (pre-post-test)	Multiple-choice questions, observation, artifacts
[38]	Two unplugged lessons	Formal	Computing	A term	Quasi-experimental (pre-post-test)	Abstraction and decomposition tasks, and self-report 5-point Likert scales
[39]	Time Teaching Program (TTP) consists of modules and activities in a science course	Formal	Science	A term	Mixed methods	Problem case questions

Table 7. CT concepts, practices, and perspectives assessed.

	CT Competency	Assessment Tool
Concepts	logic and logical thinking	Dr. Scratch, #5c21, AgentSheets and AgentCubes, CodeMaster, MissionMaker, Pencil.cc
	data	Dr. Scratch, Robotics Curriculum Instrument, Multiple-choice content assessment (Scratch), CodeMaster, MissionMaker, Computational thinking test (CTT)
	synchronization	Dr. Scratch, CodeMaster
	algorithms/algorithmic skills/algorithmic thinking	#5c21, CTSiM, FACT, VBCCT, ScratchJr Solve It Assessment, GCS, The Fairy Assessment, CTS, CTt, Robotics program, Robotics related tools, Robotics MCQ, Robotics Curriculum Instrument, Multiple-choice content assessment (Scratch), CodeMaster, Project Analysis and Artefact-based Interviews, Computational thinking knowledge test, Board Game: Crabs & Turtles, Bebras & Talent Search Computing Olympiad, MissionMaker, Pencil.cc, Sorting Algorithms artifacts & Qualitative Comparative Analysis (QCA)
	pattern/pattern recognition	ScratchJr Solve It Assessment, TDIA
	evaluation	TDIA, Bebras & Talent Search Computing Olympiad, Sorting Algorithms artifacts & Qualitative Comparative Analysis (QCA)
	automation	TDIA
Practices	abstraction	Dr. Scratch, AgentSheets and AgentCubes, Robotics MCQ, Board Game: Crabs & Turtles, Sorting Algorithms artifacts & Qualitative Comparative Analysis (QCA), Abstraction and decomposition tasks
	problem decomposition	Dr. Scratch, #5c21, TDIA, Robotics related tools, Board Game: Crabs & Turtles, Bebras & Talent Search Computing Olympiad, Sorting Algorithms artifacts & Qualitative Comparative Analysis (QCA), Abstraction and decomposition tasks, Computational thinking test (CTt)
	problem solving	FACT, CTS, Cornell Critical Thinking test (CCTT)
	organization	#5c21
	planning	FACT
	testing and debugging	#5c21, ScratchJr Solve It Assessment, TDIA, Project Analysis and Artefact-based Interviews
	modularizing and modeling	#5c21, The Fairy Assessment, Robotics related tools, Project Analysis and Artefact-based Interviews, Computational thinking test (CTT)
	being incremental and iterative	#5c21, VBCCT, TDIA, Project Analysis and Artefact-based Interviews
	user interactivity	CodeMaster, Dr. Scratch
	functionality	Sorting Algorithms artifacts & Qualitative Comparative Analysis (QCA)

	system administration	Computational thinking test (CTT)
Perspectives	creation	FACT, CTS, TDIA, Programming empowerment survey
	self-expression	FACT, TDIA
	communication	FACT, TDIA
	collaboration	FACT, CTS, TDIA, Programming empowerment survey
	questioning	TDIA
	reflection	FACT
	transfer	FACT, Bebras
	generalization	Robotics related tools, Bebras & Talent Search Computing Olympiad, Sorting Algorithms artifacts & Qualitative Comparative Analysis (QCA)

Table 8. CT assessment tools and measured CT concepts, practices, and perspectives.

Assessment Tool	CT Concepts	CT Practices	CT Perspectives
Dr. Scratch ([1], [2], [3])	parallelism, logical thinking, data representation, synchronization	abstraction, problem decomposition, user interactivity	NA
#5c21 ([4])	code literacy (algorithms, logic), technological systems literacy	identify the problem, organize and model the situation, create a computer program, evaluate and improve iteratively	NA
CTSiM ([5])	algorithms	NA	NA
FACT 'Systems of Assessments' ([6])	algorithmic thinking (serial execution, conditionals, loops)	problem solving, planning	creativity, expressing, communicating, collaborating, reflecting, transferring
VBCCT ([7])	sequences, loops, conditional statements, parallel execution, coordination, event handling, keyboard input	experimentation, iteration	NA
ScratchJr Solve It Assessment ([8])	sequencing, symbol recognition	debugging, goal-oriented sequencing	NA
TDIA ([9])	patterns and pattern recognition, evaluation, and automation	problem decomposition, creating computational artefacts, testing and debugging, iterative refinement	creating and expressing, communicating and collaborating, and understanding and questioning
GCS ([10])	programming constructs	patterns, game mechanics	NA
The Fairy Assessment ([11])	algorithmic thinking	making effective use of abstraction and modeling	NA
CTS ([12], [13], [14], [15], [34])	critical thinking, algorithmic thinking	problem solving	creativity, cooperativity
CTt ([16], [17], [18], [37])	basic directions and sequences, loops-repeat times, loops-repeat until, if/else-simple conditional, if/else-complex conditional, while conditional, functions	debugging	confidence
Robotics program ([19])	correspondence, sequencing, control flow	debugging	NA

AgentSheets and AgentCubes ([20])	formulating problems, logical thinking	abstraction	NA
Robotics related tools ([21])	algorithm	abstraction, modularity, decomposition, generalization	NA
MCQ format assessment (Robotics) ([22])	developing algorithms, evaluating algorithms	documentation and process, developing abstractions	NA
An instrument with 23 items ([23])	syntax, data, algorithms, representation, efficiency	NA	NA
Multiple-choice content assessment (Scratch) ([24])	handling an event, loop, conditionals, data, parallelism, script execution	NA	NA
ICT Literacy Test ([25])	fundamental concepts, computational literacy (CL)	contemporary skills	NA
CodeMaster ([26])	logic, parallelism, data representation, flow control, synchronization	abstraction, user interactivity	NA
Cornell Critical Thinking test (CCTT) ([27])	NA	reasoning, problem solving	NA
Project Analysis and Artefact-based Interviews ([28])	sequences, events, parallelism, conditionals	being incremental and iterative, reusing and remixing, testing and debugging, modularizing and abstracting	NA
Computational thinking knowledge test ([29])	selection, looping, arrays, functions, algorithms, search, and sorting	NA	NA
Board Game: Crabs & Turtles ([30])	sequences, operators, constant/variables, conditionals, events, loops, algorithms	decomposition, patterns, abstraction	NA
Bebras & Talent Search Computing Olympiad ([31])	algorithmic design, evaluation	decomposition, abstraction, generalization	skill transfer
Pencil.cc ([32])	variables, conditional logic, iterative logic, functions.	NA	NA
MissionMaker ([33])	algorithms, data, logic	NA	NA
Sorting Algorithms artifacts & Qualitative Comparative Analysis (QCA) ([35])	algorithmic design, evaluation	abstraction and functionality, decomposition, generalization	NA
Programming empowerment survey ([36])	NA	NA	meaningfulness, impact, creative self-efficacy, programming self-

			efficacy, interest in programming, attitudes towards collaboration
Abstraction and decomposition tasks ([38])	NA	abstraction, decomposition	NA
Computational thinking test (CTT) ([39])	data and information	modeling and simulation, computational problem-solving, system administration	NA

Table 9. Characteristics of Assessment Tools

	Assessment Tool	Format	Automatic/Non-automatic	Cognitive/ Non-cognitive	Programming Language
[1]	Dr. Scratch	Computer	A	Both	Scratch
[2]	Dr. Scratch	Computer	A	Both	Scratch
[3]	Dr. Scratch	Computer	A	Both	Scratch
[4]	#5c21(the five key competencies in twenty-first century education)	Computer	N	Cognitive	Scratch
[5]	Computational Thinking using Simulation and Modeling (CTSiM)	Both	N	Cognitive	Generic
[6]	FACT 'Systems of Assessments'	Computer	N	Both	Scratch/pseudo-code/Generic
[7]	Visual Blocks Creative Computing Test (VBCCT)	Both	N	Cognitive	Generic
[8]	ScratchJr Solve It Assessment	Both	N	Cognitive	ScratchJr
[9]	TDIA (Three-Dimensional Integrated Assessment)	Computer & Paper	N	Both	Alice
[10]	Game Computational Sophistication (GCS)	Computer	N	Cognitive	Alice 2.2 or The Storytelling Alice
[11]	The Fairy Assessment	Computer	N	Cognitive	Alice 2.2 or The Storytelling Alice
[12]	CTS (Computational Thinking Scales)	Paper	N	Both	None
[13]	CTS (Computational Thinking Scales)	Paper	N	Both	None
[14]	Scratch, OpenSim, and CTS (Computational Thinking Scales)	Computer	N	Both	None
[15]	CTS (Computational Thinking Scales)	Computer	N	Both	None
[16]	Computational Thinking test (CTt)	Computer	N	Cognitive	Generic
[17]	Computational Thinking test (CTt)	Computer	N	Cognitive	Generic
[18]	Computational Thinking test (CTt)	Computer	N	Cognitive	Generic
[19]	Robotics program	Computer	N	Cognitive	Generic
[20]	AgentSheets and AgentCubes	Both	N	Cognitive	AgentSheet and AgentCubes
[21]	Robotics related tools	Paper	N	Cognitive	Generic

[22]	MCQ format assessment (Robotics)	Computer	N	Cognitive	ROBOTC Graphical
[23]	An instrument with 23 items (15 MCQ and 8 CRQ)	Paper	N	Cognitive	Generic
[24]	Multiple-choice content assessment (Scratch)	Computer	N	Cognitive	Scratch
[25]	ICT Literacy Test Tool	Paper	N	Cognitive	Generic
[26]	CodeMaster	Computer	Y	Cognitive	App Inventor and Snap! (Block-based)
[27]	Cornell Critical Thinking test (CCTT)	Paper	N	Cognitive	None
[28]	Project Analysis and Artefact-based Interviews	Both	N	Cognitive	Scratch, Mind Map
[29]	Computational thinking knowledge test	Computer	N	Both	Generic
[30]	Board Game: Crabs & Turtles	None (card games)	N	Cognitive	Generic
[31]	Bebras & Talent Search Computing Olympiad	Computer	N	Cognitive	Generic
[32]	Pencil.cc	Computer	N	Cognitive	Block-based, text-based, and hybrid
[33]	MissionMaker	Computer	N	Cognitive	MissionMaker
[34]	CTS (Computational Thinking Scales)	Paper	N	Both	None
[35]	Sorting Algorithms artifacts & Qualitative Comparative Analysis (QCA)	Paper	N	Cognitive	None
[36]	Programming empowerment survey	Computer	N	Both	None
[37]	Computational Thinking test (CTt)	Computer	N	Cognitive	Generic
[38]	Abstraction and decomposition tasks	None (in-class tasks)	N	Cognitive	None
[39]	Computational Thinking test (CTt)	Paper	N	Cognitive	None

Table 10. Assessment methods and validity indicators

	Assessment Methods	**Validity Indicators**
[1]	Artefact assessment	N
[2]	Artefact assessment	Correlation with other tools
[3]	Artefact assessment	N
[4]	Artefact assessment	Correlation with other tools
[5]	Artefact assessment and knowledge test	N
[6]	Artefact assessment, knowledge test, and survey	Inter-rater reliability
[7]	Knowledge test and survey	Inter-rater reliability, content validity, and construct validity
[8]	Knowledge test	N
[9]	Artefact assessment	N
[10]	Artefact assessment	Inter-rater reliability
[11]	Knowledge test, survey and artefact assessment	N
[12]	Survey	Reliability and validity (EFA and CFA)
[13]	Survey	Reliability, internal consistency, convergent validity, and discriminant validity
[14]	Survey	Validated
[15]	Survey	Validated, Cronbach alpha
[16]	Knowledge test	Criterion validity, content validity, convergent validity, concurrent validity, reliability
[17]	Knowledge test	Predictive validity
[18]	Knowledge test	External validity
[19]	Artefact assessment	N
[20]	Artefact assessment and survey	Reliability (Cronbach's alpha)
[21]	Survey, interview and observation	N
[22]	Knowledge test	N
[23]	Knowledge test	Expert judgment
[24]	Knowledge test, survey and artefact assessment	N
[25]	Knowledge test	Reliability (Cronbach's alpha)
[26]	Artefact assessment	N
[27]	Knowledge test and survey	Validated
[28]	Artefact assessment	N

[29]	Knowledge test	Validated
[30]	Artefact assessment	Reliability (Cronbach's alpha), convergent validity
[31]	Knowledge test	Validated
[32]	Artefact assessment and interviews	N
[33]	Artefact assessment	N
[34]	Survey	Content validity, construct validity, factorial validity, reliability
[35]	Survey	N
[36]	Survey	N
[37]	Knowledge test, observation, and artefact assessment	N
[38]	In-class tasks, and survey	N
[39]	Knowledge test	N

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)*, Las Vegas, NV, March 18-22.

American Educational Research Association, American Psychological Association, & National Council on Measurement in Education. (2014). AERA, APA, & NCME. *Standards for Educational and Psychological Testing. Washington, DC: American Educational Research Association.*

Anderson, L.W. (Ed.), Krathwohl, D.R. (Ed.), Airasian, P.W., Cruikshank, K.A., Mayer, R.E., Pintrich, P.R., Raths, J., & Wittrock, M.C. (2001). *A taxonomy for learning, teaching, and assessing: A revision of Bloom's Taxonomy of Educational Objectives (Complete edition)*. New York: Longman.

Arksey, H., & O'Malley, L. (2005). Scoping studies: Towards a methodological framework. *International Journal of Social Research Methodology, 8*(1), 19-32. doi:10.1080/1364557032000119616.

Atmatzidou, S., & Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. *Robotics and Autonomous Systems, 75*, 661-670. doi://doi.org/10.1016/j.robot.2015.10.008.

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

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

Basnet, R. B., Doleck, T., Lemay, D. J., & Bazelais, P. (2018). Exploring computer science students' continuance intentions to use Kattis. *Education and Information Technologies, 23*(3), 1145-1158. doi:10.1007/s10639-017-9658-2.

Basu, S., Biswas, G., & Kinnebrew, J. S. (2017). Learner modeling for adaptive scaffolding in a computational thinking-based science learning environment. *User Modeling and User-Adapted Interaction, 27*(1), 5-53. doi:10.1007/s11257-017-9187-0.

Bati, K., Yetişir, M. I., Çalişkan, I., Güneş, G., & Gül Saçan, E. (2018). Teaching the concept of time: A steam-based program on computational thinking in science education. *Cogent Education*, *5*(1), 1-16.

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

Bransford, J. D., Brown, A., & Cocking, R. (1999). How people learn: Mind, brain, experience, and school. *Washington, DC: National Research Council*.

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

Vancouver, Canada, 1-25.

Buffum, P. S., Lobene, E. V., Frankosky, M. H., Boyer, K. E., Wiebe, E. N., & Lester, J. C. (2015). A practical guide to developing and validating computer science knowledge assessments with application to middle school. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (pp. 622-627). ACM. doi:10.1145/2676723.2677295.

Chen, G., Shen, J., Barth-Cohen, L., Jiang, S., Huang, X., & Eltoukhy, M. (2017). Assessing elementary students' computational thinking in everyday reasoning and robotics programming. *Computers & Education, 109*, 162-175. doi://doi.org/10.1016/j.compedu.2017.03.001.

Cuny, J., Snyder, L., & Wing, J. M. (2010). Demystifying computational thinking for non-computer scientists. *Unpublished Manuscript in Progress, Referenced in Http://Www.Cs.Cmu.Edu/~ CompThink/Resources/TheLinkWing.Pdf.*

de Paula, B. H., Burn, A., Noss, R., & Valente, J. A. (2018). Playing Beowulf: Bridging computational thinking, arts and literature through game-making. *International Journal of Child-Computer Interaction, 16*, 39-46. doi://doi.org/10.1016/j.ijcci.2017.11.003.

Denner, J., Werner, L., Campe, S., & Ortiz, E. (2014). Pair programming: Under what conditions is it advantageous for middle school students? *Journal of Research on Technology in Education, 46*(3), 277-296. doi:10.1080/15391523.2014.888272.

Denning, P. J. (2017). Computational thinking in science. *American Scientist, 105*(1), 13-17.

Denning, P. J., & Freeman, P. A. (2009). The profession of IT: Computing's paradigm. *Communications of the ACM, 52*(12), 28–30. doi:10.1145/1610252.1610265.

DiSessa, A. A. (2001). *Changing minds: Computers, learning, and literacy.* MIT Press.

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

Durak, H. Y., & Saritepeci, M. (2018). Analysis of the relation between computational thinking skills and various variables with the structural equation model. *Computers and Education, 116*, 191-202. doi:10.1016/j.compedu.2017.09.004.

Fessakis, G., Gouli, E., & Mavroudi, E. (2013). Problem solving by 5–6 years old kindergarten children in a computer programming environment: A case study. *Computers & Education, 63*, 87-97. doi://doi.org/10.1016/j.compedu.2012.11.016.

Fronza, I., Ioini, N. E., & Corral, L. (2017). Teaching computational thinking using agile software engineering methods: A framework for middle schools. *ACM Trans. Computers & Education, 17*(4), 19:1–19:28. doi:10.1145/3055258.

Garneli, V., & Chorianopoulos, K. (2018). Programming video games and simulations in science education: Exploring computational thinking through code analysis. *Interactive Learning Environments, 26*(3), 386-401. doi:10.1080/10494820.2017.1337036.

Goldstein, S., Princiotta, D., & Naglieri, J. A. (2015). Handbook of intelligence. *Evolutionary Theory, Historical Perspective,*

and Current Concepts.New York, NY: Springer.

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

Grover, S., & Pea, R. (2018). Computational Thinking: A competency whose time has come. *Computer Science Education: Perspectives on teaching and learning in school. London: Bloomsbury Academic*, 19-37. doi:10.1080/08993408.2015.1033142.

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.

Guzdial, M. (2008). Education paving the way for computational thinking. *Communications of the ACM, 51*(8), 25-27.

Heppner, P. P., & Petersen, C. H. (1982). The development and implications of a personal problem-solving inventory. *Journal of Counseling Psychology, 29*(1), 66-75. doi:10.1037/0022-0167.29.1.66.

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. doi://doi.org/10.1016/j.compedu.2018.07.004.

ISTE (2011). Computational Thinking in K–12 Education leadership toolkit. Computer Science Teacher Association: http://csta. acm. org/Curriculum/sub/CurrFiles/471.11 CTLeadershipt Toolkit-SP-vF. pdf adresinden alındı.

Jacob, S., Nguyen, H., Tofel-Grehl, C., Richardson, D., & Warschauer, M. (2018). Teaching computational thinking to English learners. *NYS TESOL journal, 5*(2).

Jun, S., Han, S., Kim, H., & Lee, W. (2014). Assessing the computational literacy of elementary students on a national level in korea. *Educational Assessment Evaluation and Accountability, 26*(4), 319-332. doi:10.1007/s11092-013-9185-7.

Kafai, Y. B., & Resnick, M. (Eds.). (1996). *Constructionism in practice: Designing, thinking, and learning in a digital world.* Hillsdale, NJ: Erlbaum.

Kahn, K., Sendova, E., Sacristán, A. I., & Noss, R. (2011). Young students exploring cardinality by constructing infinite processes. *Technology, Knowledge and Learning, 16*(1), 3-34. doi:10.1007/s10758-011-9175-0.

Kay, A., & Goldberg, A. (1977). Personal dynamic media. *Computer, 10*(3), 31-41.

Knee, J. A., Hirsh-Pasek, K., Golinkoff, R. M., & Singer, D. (2006). *Play = learning*. New York: Oxford University Press. doi:oso/9780195304381.001.0001.

Kong, S. C., Chiu, M. M., & Lai, M. (2018). A study of primary school students' interest, collaboration attitude, and programming empowerment in computational thinking education. *Computers & Education*, *127*, 178-189.

Korkmaz, Ö., Çakir, R., & Özden, M. Y. (2017). A validity and reliability study of the Computational Thinking Scales (CTS). *Computers in Human Behavior, 72*, 558-569. doi:10.1016/j.chb.2017.01.005.

Korkmaz, Ö. (2012). *A validity and reliability study of the Online Cooperative Learning Attitude Scale (OCLAS). Computers &*

Education, 59(4), 1162-1169. doi://doi.org/10.1016/j.compedu.2012.05.021.

Lahtinen, E., Ala-Mutka, K., & Järvinen, H. (2005). A study of the difficulties of novice programmers. In: *Proceedings of the 10th Annual SIGCSE Conference on innovation and Technology in Computer Science Education* (pp. 14–18).

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. doi:10.1007/s10956-016-9628-2.

Looi, C. K., How, M. L., Longkai, W., Seow, P., & Liu, L. (2018). Analysis of linkages between an unplugged activity and the development of computational thinking. *Computer Science Education*, *28*(3), 255-279.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12?. *Computers in Human Behavior, 41*, 51-61. doi://doi.org/10.1016/j.chb.2014.09.012.

Moreno-León, J., Robles, G., & Román-González, M. (2015). Dr. Scratch: Automatic analysis of scratch projects to assess and foster computational thinking. *Revista De Educacion a Distancia, 46*, 1-23.

Moreno-León, J., Robles, G., & Román-González, M. (2016). Code to learn: Where does it belong in the K-12 curriculum. *Journal of Information Technology Education: Research, 15*, 283-303.

Moreno-León, J., Robles, G., & Román-González, M. (2016). Comparing computational thinking development assessment scores with software complexity metrics. Paper presented at the *IEEE Global Engineering Education Conference (EDUCON)* (pp. 1040-1045). doi:10.1109/EDUCON.2016.7474681.

Mouza, C., Marzocchi, A., Pan, Y., & 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. doi:10.1080/15391523.2016.1146561.

Munoz, R., Villarroel, R., Barcelos, T. S., Riquelme, F., Quezada, Á., & Bustos-Valenzuela, P. (2018). Developing computational thinking skills in adolescents with autism spectrum disorder through digital game programming. *IEEE Access*, *6*, 63880-63889.

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

Pellas, N., & Vosinakis, S. (2018). The effect of simulation games on learning computer programming: A comparative study on high school students' learning performance by assessing computational problem-solving strategies. *Education and Information Technologies, 23*(6), 2423-2452. doi:10.1007/s10639-018-9724-4.

Pham, M. T., Rajić, A., Greig, J. D., Sargeant, J. M., Papadopoulos, A., & McEwen, S. A. (2014). A scoping review of scoping reviews: advancing the approach and enhancing the consistency. *Research synthesis methods, 5*(4), 371-385.

Psycharis, S., & Kallia, M. (2017). The effects of computer programming on high school students' reasoning skills and mathematical self-efficacy and problem solving. *Instructional Science, 45*(5), 583-602. doi:10.1007/s11251-017-9421-5.

Rich, K., & Yadav, A. (2019, March). Infusing Computational Thinking Instruction into Elementary Mathematics and Science: Patterns of Teacher Implementation. In *Society for Information Technology & Teacher Education International Conference* (pp. 76-80). Association for the Advancement of Computing in Education (AACE).

Rijke, W. J., Bollen, L., Eysink, T. H., & Tolboom, J. L. (2018). Computational Thinking in Primary School: An Examination of Abstraction and Decomposition in Different Age Groups. *Informatics in education*, *17*(1), 77.

Rojas-López, A., & García-Peñalvo, F. J. (2018). Learning scenarios for the subject methodology of programming from evaluating the computational thinking of new students. *Revista Iberoamericana De Tecnologias Del Aprendizaje, 13*(1), 30-36. doi:10.1109/RITA.2018.2809941.

Román-González, M., Pérez-González, J. C., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior, 72*, 678-691. doi://doi.org/10.1016/j.chb.2016.08.047.

Román-González, M., Pérez-González, J.-C., Moreno-León, J., & Robles, G. (2018a). Extending the nomological network of computational thinking with non-cognitive factors. *Computers in Human Behavior, 80*, 441-459. doi:10.1016/j.chb.2017.09.030.

Román-González, M., Pérez-González, J. C., Moreno-León, J., & Robles, G. (2018b). Can computational talent be detected? Predictive validity of the Computational Thinking Test. *International Journal of Child-Computer Interaction, 18*, 47-58. doi:10.1016/j.ijcci.2018.06.004.

Romero, M., Lepage, A., & Lille, B. (2017). Computational thinking development through creative programming in higher education. *International Journal of Educational Technology in Higher Education,14*(1), 1-15. doi:10.1186/s41239-017-0080-z.

Sáez-López, J., Román-González, M., & Vázquez-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. doi:10.1016/j.compedu.2016.03.003.

Seehorn, D., Carey, S., Fuschetto, B., Lee, I., Moix, D., O'Grady-Cunniff, D., . . . Verno, A. (2011). *CSTA K–12 computer science standards: Revised 2011.* New York, NY, USA: ACM.

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

Strawhacker, A., Lee, M., & Bers, M. U. (2018). Teaching tools, teachers' rules: Exploring the impact of teaching styles on young children's programming knowledge in ScratchJr. *International Journal of Technology and Design Education, 28*(2), 347-376. doi:10.1007/s10798-017-9400-9.

Tsarava, K., Moeller, K., & Ninaus, M. (2018). Training computational thinking through board games: The case of crabs &

turtles. *International Journal of Serious Games, 5*(2), 25-44.

von Wangenheim, C. G., Hauck, J. C. R., Demetrio, M. F., Pelle, R., da Cruz Alves, N., Barbosa, H., & Azevedo, L. F. (2018). CodeMaster - automatic assessment and grading of App Inventor and Snap! programs. *Informatics in Education, 17*(1), 117-150. doi:10.15388/infedu.2018.08.

Weintrop, D., & Wilensky, U. (2018). How block-based, text-based, and hybrid block/text modalities shape novice programming practices. *International Journal of Child-Computer Interaction, 17*, 83-92. doi://doi.org/10.1016/j.ijcci.2018.04.005.

Werner, L., Denner, J., & Campe, S. (2014). Children programming games: A strategy for measuring computational learning. *Trans. Comput. Educ., 14*(4), 24:1–24:22. doi:10.1145/2677091.

Whetton, D. A., & Cameron, K. S. (2002). Answers to exercises taken from developing management skills. *Northwestern University*.

Wilson, C., Sudol, L. A., Stephenson, C., & Stehlik, M. (2010). Running on empty: The failure to teach K-12 computer science in the digital age. *Association for Computing Machinery,* 26.

Wing, J. (2014). Computational thinking benefits society. *40th Anniversary Blog of Social Issues in Computing.* Retrieved from http://socialissues.cs.toronto.edu/2014/01/computational-thinking.

Wing, J. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 366*(1881), 3717-3725. doi:10.1098/rsta.2008.0118.

Witherspoon, E. B., Higashi, R. M., Schunn, C. D., Baehr, E. C., & Shoop, R. (2017). Developing computational thinking through a virtual robotics programming curriculum. *ACM Trans. Comput. Educ., 18*(1), 4:1–4:20. doi:10.1145/3104982.

Yadav, A., Gretter, S., Good, J., & McLean, T. (2017). *Computational thinking in teacher education.* In Emerging research, practice, and policy on computational thinking (pp. 205-220). Springer, Cham.

Yadav, A., Larimore, R., Rich, K., & Schwarz, C. (2019, March). Integrating computational thinking in elementary classrooms: Introducing a toolkit to support teachers. In *Society for Information Technology & Teacher Education* International Conference (pp. 93-96). Association for the Advancement of Computing in Education (AACE).

Yağcı, M. (2018). A valid and reliable tool for examining computational thinking skills. *Education and Information Technologies,* 1-23. doi:10.1007/s10639-018-9801-8.

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.

Zhong, B., Wang, Q., Chen, J., & Li, Y. (2017). Investigating the period of switching roles in pair programming in a primary school. *Educational Technology & Society, 20*(3), 220-233.