IJCSES

Volume 6, No: 4
Nov 2024
ISSN 2513-8359

International Journal of Computer Science Education in Schools

Special Issue on the Assessment of Computational Thinking

Guest Editors

Dr d'Reen Struthers

Dr May Irene Furenes Klippen

www.ijcses.org

International Journal of Computer Science Education in Schools

Nov 2024, Vol 6, No 4

DOI: 10.21585/ijcses.v6i2

Table of Contents

	Page
Editorial by Dr d'Reen Struthers and Dr May Irene Furenes Klippen	3-5
Valerie Critten, Hannah Hagon, Melike Aslan Unlu A Curriculum Framework and Assessment Approach for Computational Thinking in the Early Years	6-30
Sanna Forsström, Melissa Bond Computational Thinking Assessment in Primary and Secondary Education: A Meta-Synthesis of Tools, Methods And Pedagogical Approaches	31-73
Yasemin Allsop, Filiz Kalelioglu, Melike Aslan Unlu Evaluating the Primary Trainee Teachers' Knowledge of Computational Thinking Concepts Using a Card Sorting Activity	74-108

Editorial by Dr d'Reen Struthers and Dr May Irene Furenes Klippen

This special issue collects research on the assessment of computational thinking (CT) skills, with a focus on education from early childhood to secondary school. As CT becomes increasingly vital in modern education, understanding how to effectively teach and assess these skills is critical.

With little agreement on a definition of computational thinking, it becomes harder to develop standardised assessment tools, let alone invite agreement on what kind of activities can best contribute to CT from early years to Secondary school. However, all three papers included in this special edition draw on and make use of the following: Algorithmic thinking, abstraction, decomposition, patterns and generalisations, and evaluation, offering a way forward to explore this landscape of teaching and assessing computational thinking across education.

The three studies reviewed here explore different approaches to integrating and evaluating CT in classrooms, highlighting the importance of practical methods and best practices for assessment.

The first study, *A Curriculum Framework and Assessment Approach for Computational Thinking in the Early Years*, offers a theoretical argument for the possibility computational thinking can be taught and assessed in Early Years settings. The paper asserts that CT is not an additional extra but can be facilitated in activities already undertaken in these settings, as it aligns with the core national curriculum areas already being covered. With the inclusion of a sample lesson and ways to both teach and assess CT, the paper suggests that the next generation of early years practitioners and teachers have no reason to shy away from including CT in their teaching. The rationale for assessing progress in CT is made as the foundation for how CT offers the baseline knowledge for computer sciences and the world of STEM subjects.

The second study, *Computational Thinking Assessment in Primary and Secondary Education: A Meta-Synthesis of Tools, Methods, and Pedagogical Approaches*, provides a comprehensive analysis of how CT skills are assessed across various educational levels. The study synthesises evidence from thirteen reviews and identifies six key CT skills: core CT components, programming concepts, cognitive processes, problem-solving strategies, collaborative and communicative skills, and dispositions and attitudes. It highlights the diversity of assessment methods—ranging from direct observation to innovative approaches—but emphasises the gap between academic research and practical classroom tools. The study calls for more user-friendly, integrated assessment methods that align with teachers' needs.

The third study, *Evaluating the Primary Trainee Teachers' Knowledge of Computational Thinking Concepts using a card sorting Activity,* moves the reader to consider the kinds of activities which can effectively support student teachers' understanding of CT in initial teacher education. This paper highlights that there is little supporting material about ways to teach and assess the development of CT concepts in initial teacher education. As a contribution to the field, the "Match it" card sorting activity was evaluated as a tool to assess prospective teachers' knowledge and understanding of CT concepts within a fuller series of lessons. Significantly, the sub-dimensions of computational thinking skills were used, and the findings suggest that the practical application of algorithmic tasks might be easier to understand rather than the more abstract CT concepts. The tool itself, however, prompted students to reflect on their understanding and supported their developing understanding and application of ideas. The mention of a possible rubric with a built-in scoring mechanism for a possible online version could offer immediate feedback. This theme is also explored in the Fifth study.

Discussion and Conclusion

This special issue foregrounds the way effective assessment of computational thinking (CT) skills in education, spanning from early childhood to secondary school, might be possible. As CT becomes increasingly vital in the modern educational landscape, it is crucial to identify practical methods for teaching and evaluating these skills.

A significant challenge identified is the gap between academic research on CT and the practical tools available for educators. While various assessment methods exist, many remain complex and difficult to implement in classroom settings. To address this, there is a pressing need for user-friendly, integrated assessment strategies that align with teachers' pedagogical goals.

Ultimately, bridging this gap will enable educators to cultivate CT skills more effectively with their students. By focusing on developing practical assessment tools, educators can foster a deeper understanding of CT, ensuring that students are well-prepared to navigate a future increasingly influenced by technology and innovation.

Editors

	Dr d'Reen Struthers, Lecturer at the IoE-UCL, London, has been involved in Primary and Secondary Teacher Education for over 3 decades in both England and New Zealand. With a keen interest in learning and teaching, innovative pedagogies, philosophy and educational psychology, her own research has explored a variety of topics including the use of technology in the music education, alternative assessment practices, Pedagogical partnerships and Teacher Resilience. She is currently supervising two PhD students who are exploring the impact of digital technology on the student teacher relationships in HE during COVID and computational thinking in the Early Years. She also contributes to the leadership of the Educational Doctorate at the IoE.
	May Irene Furenes Klippen is an Associate Professor at the Knowledge Centre for Education at the University of Stavanger, Norway. Her research focuses on co-creation and feedback processes, with a particular emphasis on fostering collaborative learning and improving pedagogical practices in educational settings. She also specializes in systematic review methodologies, working across all levels of education, from Early Childhood Education and Care to higher education.

International Journal of Computer Science Education in Schools, November 2024, Vol. 6, No. 4
ISSN 2513-8359

A Curriculum Framework and Assessment Approach for Computational Thinking in the Early Years

Valerie Critten[1]

Hannah Hagon[2]

Melike Aslan Unlu[3]

[1]Open University, UK

[2]Unplugged Tots, UK

[3]University College London, UK

DOI: 10.21585/ijcses.v6i4.230

Abstract

In light of current developments, there is an increasing effort to integrate computing-oriented activities into the education of children as young as two years old. Although the computing strand is not officially addressed in the Early Years Foundation Stage Statutory Framework (DfES, 2024), a small number of early years teachers in England implement computing-oriented activities to ensure that young children progress from early years to Key Stage 1. A particular gap in the field is that previous research on computational thinking concepts never or rarely establishes curriculum links in a way that teachers can utilise in their practices. This theoretical article, therefore, proposes a curriculum-based framework for both teaching and assessing computational thinking (CT) in early years education, as assessment is not possible without pedagogic guidelines. Offering a sample lesson plan with links to the Early Learning Goals, this framework aims to encourage teachers, including those without specific computing training, to integrate CT concepts more explicitly into their teaching and enables them to monitor and assess their pupils' progress in relevant skills.

Keywords: computational thinking, early years, identifying CT, curriculum framework, assessing CT skills.

1. Introduction

Many researchers of computational thinking (CT) propose that basics of computational science or digital technology could be taught in preschool establishments, and have published studies in which very young children successfully learn to program and code utilising computer games or robots (Bers et al., 2019; Fessakis, 2013; Papadakis, 2020) or other coding devices such as Code-a-pillar (Wang, 2021). It is suggested that preschool children would benefit from learning CT skills alongside reading and writing (Lee, 2022; OECD, 2023; Wing, 2006), however it has been noted that often teachers do not have the time or training to teach these skills (Wang, 2021; Wang et al, 2023) and are reluctant to add to an overloaded curriculum (Dong, 2018; Ireland, 2015). While CT studies may take place within preschool establishments, they rarely advise on how the subject can be integrated into lessons (Yücelyiğit, 2023), that the specialist equipment may be expensive for early years' establishments (Dong, 2018), and that teachers will usually only teach to national curriculum guidelines (Barr & Stephenson, 2015), but there are many suggestions that the National Curriculum needs to change, and younger children need to start to learn some form of CT (Dong, 2018: Lee et al, 2022).

This innovative article examines the possibility and practicality of teaching and assessing CT skills to children in the early years or preschool. While many research articles support the concept of changing the curriculum there are no, or very few concrete suggestions that teachers may follow. We examined whether the Early Years curricular framework could be adapted by early years teachers in order that children could be assessed on the development of elementary CT skills before entering into the more formal education required by the Key Stage 1 curriculum guidelines and provide suggestions for CT activities that can be incorporated into the existing curriculum.

Literature review

Computational thinking as a concept for teaching has come into prominence in the last few years. While it is often used interchangeably with computer programming in the literature (Shute et al., 2017), some theorists suggest this is a misconception and that CT should be thought of as an ongoing thinking process rather than having a code-like outcome (Lee et al., 2022; Wing, 2006). Researchers have discussed what CT could consist of, and relevant assessments developed for, primary-aged children by mapping the teaching and assessment to a specific curriculum (Snow et al., 2019; Waterman et al., 2020), however, there is a paucity of information on CT for children in the early years in common with many other countries such as Sweden (Otterborn et al., 2019) and Brazil (Gomes et al., 2018).

Consideration must be given to why it is important to teach CT to very young children particularly as the subject is not addressed in many countries' national curriculum (Çimşir et al., 2024; Dong, 2018). Results from research studies suggest that it is possible to start teaching aspects of CT from the ages of 3 years (Bers et al., 2019) using robotic toys such as the KIBO, while Critten et al (2021) started teaching unplugged coding using guided play to children aged from 2 years. The results in both these and other studies showed that the activities and guided play '… promoted communication, collaboration and creativity…' (Bers et al., 2019, page 1). Many studies utilised robot-type toys in their studies while a number of researchers suggested using play activities to promote CT were related to lower costs and could provide a better foundation for plugged computer lessons in later education (Saxena et al., 2020). Whichever approaches are used to teach CT in the early years, Bers et al., (2019) stressed that more research needs to focus on how learners engage with CT and learn from their lessons, and how teachers can introduce these new educational areas into their curriculum.

Thus, many researchers into CT regard the introduction of aspects of CT in the early years to be desirable as it could enhance the development of the children's creativity and communication abilities (Bers, 2019), however many teachers regard the use of computers and digital media to be a threat to children's social abilities and the development of their play

(Dong, 2018). Further, a lack of training opportunities and practical classroom guidance have resulted in many teachers unsure and resistant to teaching CT in their lessons (Wang et al., 2021; Wang et al., 2023), while Bers et al., (2019) consider that appropriate pedagogical approaches must be developed. In order to ensure this many teachers need to have the time and opportunity to undergo training in CT and how it can be incorporated into lessons (Çimşir et al., 2024). Teachers in China were interviewed to find out their views on the teaching of CT before they took part in the training of pedagogical approaches to CT in their schools. The teachers felt that designing lessons involving ICT took a lot of preparation time and effort, but after their training had taken place, they felt that 'awareness-raising' was the most important aspect of the training, and many of the participants appreciated the way that CT could be incorporated into other subjects such as drama lessons (Dong, 2018). These arguments raise the question of why aren't opportunities given to promote CT in the early years and why aren't teachers given the benefit of training in a subject which is considered by researchers to be so important.

Theoretical framework for teaching CT in the early years

Government guidelines for Key Stage 1 children, aged from 5-7 years in England, propose that children should have knowledge of computer programming: '(to) *understand: what algorithms are, how they are implemented as programs on digital devices, and that programs execute by following precise and unambiguous instructions*.' (DfE, 2013, p.2). There is a case to be made that foundations could be laid in the early years period which would enable children to gain concepts of CT before they reach Key Stage 1. Wing (2006) suggests *'To reading, writing, and arithmetic, we should add computational thinking to every child's analytical ability'* (pp 33) as they will need CT mental tools for their development in computer science. This case is made stronger by the fact that many young children aged between 0-5 years are already using digital apps on phones and tablets (Bers et al, 2014; OECD, 2023; Ofcom, 2023). It has been suggested that preschool children benefit from learning to program as it may encourage the development of computational thinking skills such as communication and collaboration; logical thinking and reasoning; and the organisation and evaluation of ideas (Fessakis et al., 2013; Gomes et al., 2018). This theoretical article examines how CT skills could be incorporated into the existing Early Years Curriculum Framework (DfES, 2024) and consists of four main areas for consideration:

- Concepts of teaching CT in the early years, in particular the CT skills that can be developed for preschool children;
- Incorporating computational thinking in the early years' curriculum and how these skills can be taught utilising the existing framework;
- Developing children's thinking abilities by providing appropriate lessons and resources;
- Examples of assessment which can be included in the existing assessment programmes.

2. Concepts of teaching CT in the early years

In order to examine whether it is appropriate to teach CT skills in the early years, it is necessary to consider what aspects of CT can be taught to children who may not yet have developed reading or writing skills and may not be able to use a computer. Figure 1 shows a breakdown of the main CT themes identified in the literature (see Bers et al, 2014; Fessakis et al, 2013; Lee et al, 2022; Wing, 2006).

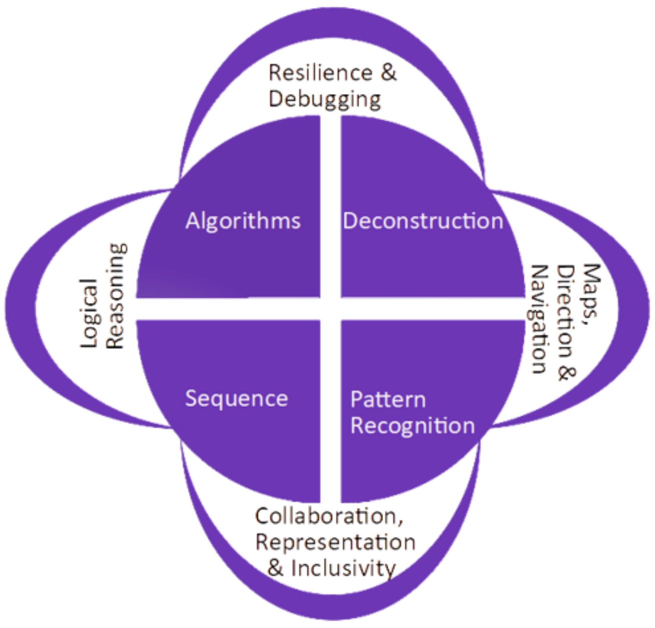


Figure 1 A representation of the key components of computational thinking (Hagon, 2024)

Looking at a representation of CT skills (Hagon, 2024) in Figure 1 and Table 1, certain aspects of CT can be incorporated in play activities/ and or lessons for children in the early years such as:

- Listening, communication and collaboration – how to work together, teaching children that with team work things can be easier (Bers et al., 2019), and links with communication (Papert & Harel, 1991). Pairs and group working, listening and communicating to promote inclusivity, being able to celebrate everyone for their unique skills. Many young children may be more familiar with apps and digital devices, and co-constructing and collaborating with teachers may take place in lessons and activities (Dong, 2018)

- Logical thinking, analysing and problem-solving (Caeli & Yadav, 2020) – the idea of questioning a process first or establishing logical reasoning to communicate with an adult to explain why a sequence was debugged or explain the rationale behind a particular activity or action (Bers, 2018). This can be done prior to an activity or after depending on whether the child has had previous experience, aided by teachers scaffolding the learning, assessing progress, and then giving opportunities to repeat the activity or a similar activity to measure progress

- Algorithms, sequencing, debugging, resilience – following rules will help when the child has to both give and receive instructions and directions; investigating where the problems are with an algorithm, sequence or pattern and backtracking to fix them; learning the art of behaving appropriately when something unexpected happens, trying again, thinking of an alternative, asking for help; fostering a growth mindset and developing confidence in their abilities (Smiley & Dweek, 1994). Pattern recognition, sequencing and debugging as these enable children to learn about creating loops in code

- Maps/Navigation and Direction – when learning about software programs such as ScratchJR and BeeBots in later academic years, the knowledge of left and right, backwards and forwards is critical so utilising map skills with landmarks and navigation supports the scaffolding for this learning ((Kalogiannakis & Papadakis, 2020; Papadakis, 2021). This affords the teachers lots of opportunities to look at the local environment in the classroom, playground and local play areas which is already in many curricula.

Whilst the following are not primarily considered to be CT themes, they support multiple positive skills and attitudes that complement and enhance the EYFS PSHE content:

- Inclusivity – the act of being inclusive, celebrating everyone for their unique skills, advocate for oneself and each other based on observing skills and abilities
- Representation – fostering the confidence in children to stand up for them/their selves and their peers.

Adding these themes to the curriculum present a huge challenge to teachers when the curriculum is already overloaded (Dong, 2018; Wang et al, 2021). This next section looks at how many of these themes can be incorporated within existing school projects.

3. Incorporating computational thinking in the early years' curriculum

This section describes how CT can be incorporated into the curriculum given that many teachers are not trained in teaching any form of digital technology. While researchers advise the incorporation of digital technologies into the early years curriculum, the guidance is very much at the level of policy directives rather than direct guidance for teachers' lessons, (e.g. OECD, 2023). Many experienced teachers are now being asked to teach CT skills without having been trained in computer science or computing education (Sentance & Csizmadia, 2015). Research suggests that teachers in the early years commonly lack proficiency and confidence in programming-related knowledge and skills, and they seek clearer curriculum guidelines to accompany their teaching efforts (Otterborn et al., 2019; Otterborn et al., 2020). We have married up existing targets in the framework with concepts of CT from Figure 1 that are linked by researchers (e.g. Lee et al, 2022). Table 1 displays the three main areas in the framework: communication and language; physical development, and PSHE.

Table 1: Linking CT skills with the Early Learning Goals

Early Learning Goals (ELGs)	**Specific Targets**	**Links with CT skills (taken from Figure 1)**
Communication and Language	Interactions with others	Listening, communicating and collaborating
	Commenting	
	Answering questions	

Listening, attention and understanding	Language and vocabulary development	
	Self-commenting	Logical thinking, analysing and problem-solving
	Asking questions	

Physical development	Singing and dancing	Algorithms, sequencing, debugging, resilience
	'Robot' dancing	
Gross motor skills	Directionality/ orienteering	Learning directional language (forwards, backwards, left, right and pause)
Fine motor skills	Sequences of movement	
	Making patterns Guided play activities	Identifying, sequencing, debugging

Physical, Social and Health Education (PSHE)	Friendships	Pairs and group working, listening and communicating
	Collaboration	
	Interpersonal skills	
Self-regulation Managing self Building relationships	Modelling	Collaborating, listening, logical thinking, problem solving

Previous research has identified teachers' reluctance to integrate computing and CT into their teaching practice when there are no clear curriculum or government guidelines (Dong, 2018; Israel et al., 2015). Linking the above skills to CT themes will assist practitioners in introducing elements of CT into the setting and these will provide an introduction or baseline for the National Curriculum requirements for CT in Key Stage 1

(DfES, 2024). A further advantage is that practitioners can use the current Early Years assessment framework which addresses the areas of learning as well as covering CT skills, further utilising cross curricular learning opportunities.

It is recognised that '*children's development, learning and wellbeing are best served through play*' (Smedley & Hoskins, 2020, p.1202), thus, by integrating CT themes into structured, play based activities, children can learn the concepts in an accessible way and teachers can teach these concepts without having to learn over-engineered techniques or overwhelming skills that are simply irrelevant to young children (Lee et al., 2022). CT skills can dovetail into a typical school's curriculum planning and by thinking laterally, many lesson targets can be cross curricular with a slight change of focus and a little more direction by the teacher to allow for the sessions to have a CT focus. For example, lessons incorporating patterns are looked at in the maths curriculum by exploring and identifying similarities and differences in shapes, colours and sizes in the surrounding world of young children (Gifford, 2005). Additionally, basic concepts of algorithms can be addressed in communication and language with a self-regulation early learning goal stating that children should follow a set of instructions (DfES, 2024).

4. Developing children's thinking skills

There has been discussion by researchers as to how CT could be incorporated into teaching and learning within early years establishments (Lee et al., 2022), and if it is appropriate to try to develop these skills at such a young age. Earlier studies have suggested that very young children are not able to think abstractly and so are limited in learning programming and coding by their developmental levels (Clements and Guilo, 1984). These constraints are also suggested by Piaget's theory as he reports that preschool children have difficulty in using logic and reasoning; being able to mentally manipulate information, and taking other's perspectives (Piaget & Inhelder, 1973). However, a shift in perspective has emerged due to additional research, which demonstrated that preoperational children were less egocentric and animistic than Piaget had initially suggested (Newcombe & Huttenlocher, 1992). His theory, as a result, received criticism for potentially underestimating children's cognitive development (Siegler, 2016). Starting from a very young age, children are now recognised as individuals with elevated cognitive capabilities and an inherent tendency for exploration (Gopnik, 2010). It can be difficult for a teacher untrained in computer science to change their pedagogical focus, but researchers suggest that with the right teaching approaches, equipment and support from adults, preschool children are able to develop CT skills and coding skills (Bers et al., 2014; Bers et al., 2019; Gomes et al., 2018).

Many research studies start with encouraging the children to learn algorithmic thinking with the use of games and songs such as ‘Simon says’ and various nursery rhymes the Hokey Cokey (Hokey Pokey) and the use of particular books which have a clear sequence or pattern of events (Relkin et al., 2020). Further suggestions for teaching CT concepts such as algorithms, decomposition, sequencing, and patterns (Lee et al, 2022); communication and collaboration (Critten et al., 2021); problem-solving and debugging (Relkin et al., 2020) can be taught using unplugged activities that are ‘… *concrete, hands-on, and play-based’* (Lee et al., 2022, pp 4). Activities such as basic cooking (icing biscuits or pizza-making), bathing a doll, building a Lego castle, and getting dressed in the correct order can all be utilised in guided play to encourage the children to sequence, evaluate, problem solve and debug (see Critten et al, 2021). These activities provide an introduction before asking children to create their own sequences of movements, to encourage directionality or distance and can be/ and are already taught in early years, but not explicitly for CT development.

A number of CT skills are already being taught in preschools with older children aged 5-6 years, especially in the field of spatial thinking (Palmer, 2017) with the use of programmable manipulatives such as toy robots (Bers et al, 2019) and other digital toys (Wang et al., 2021). Palmer (2017) suggests that concepts such as orientation skills; counting especially regarding one-to-one correspondence; an understanding of symbols (the arrows on a robot); and communication and collaboration can all be taught with the use of a robot and guided interactions from a teacher. Robots such as the Bee-Bot have been utilised in a number of studies involving preschool children, but these can be problematic for very young children (Bakala et al., 2021). One of the difficulties these authors identified is the use of physical buttons to program the robot which seems to be very accessible for young children, but it involves directionality and the use of left and right commands which young children may not have mastered. Critten et al. (2021) suggest that teaching Bee-Bots might be better suited to those over four years of age and generally the majority of the research articles involving robots such as Bee-Bots are for children of 4-5 years and over. Papert and Harel suggests that the use of robotics in a creative and '*constructionist*' process can encourage children to build their own games and projects and evaluate and amend, especially in a communicative and collaborative environment (Papert & Harel, 1991, 2002).

5. Approaches to assessing Computational Thinking in the Early Years

An important aspect of teaching CT skills in the early years is assessment, both of the lessons themselves and of the children's understanding and progress, and these are always related (Zygotsky et al., 1980).

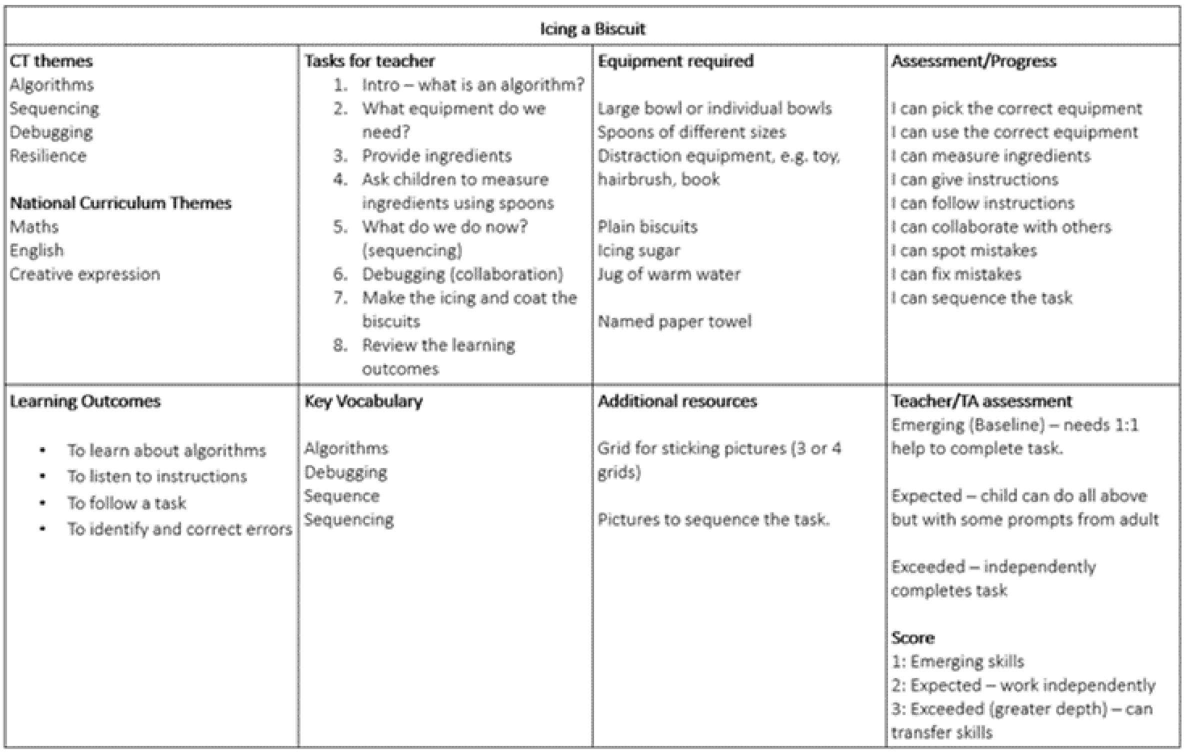
Assessment in the preschool years is mainly carried out by observations in the classroom, within small groups and individually in order to establish the children's knowledge and understanding. Not only are the staff interested in seeing progress in specific subjects, but they are also interested in children's social development particularly in their interactions with their peers, adults around them, and their play with toys (DfE, 2022). In preschool establishments, staff often show children's progress with a collection of photos, examples of work and descriptions of behaviours and these are presented together in a Learning Book or journal. These can be viewed at school by parents or, more commonly, digital formats can be readily shared throughout the year, for example, using apps such as Marvellous Me or Tapestry.

Early years education adopts a holistic approach by working on and with the whole child (Brassard & Boehm, 2007; DfES, 2024). Assessment of such holistic learning, therefore, requires a multi-faceted evaluation (Allsop, 2019), and assessing young children's CT progress is likely to serve multiple functions. Specifically, teachers will understand the needs and strengths of children across developmental areas and accordingly plan instruction and other forms of early intervention; adjust learning activities, monitor children's progress, and set reachable goals for each individual in the classroom; evaluate the effectiveness of applied activities and intervention programs; evaluate their own teaching for the purposes of accountability (Brassard & Boehm, 2007).

Because there are no statutory guidelines for CT assessment, researchers have mainly developed their own assessment frameworks (Shute et al., 2017) and these assessments are often based on a breakdown of the skills needed to perform a task such as on Scratch or Alice software (see Allsop, 2019). However, this type of assessment is designed for older pupils who are using computer screens. So how can teachers assess very young pupils in preschool establishments? Assessment criteria in the EYFS Baseline Assessment does hint at CT themes. In Language for communication and listening, ‘Talks activities through, reflecting on and modifying actions’ is logical reasoning. Linking sounds and letters, ‘Joins in with rhyming and rhythmic activities’ links to pattern recognition. In Shapes, space and measures ‘*Describes shapes in simple models, pictures and patterns,*’ can look at pattern recognition again. In Knowledge and understanding of the world ‘*Shows curiosity and interest by exploring surroundings’* can be great examples of utilising maps, navigation and landmarks. Creative development encompasses many of the unplugged activities available as the children have creative freedom throughout the structured play activity and this can allow opportunities for child/adult conversations (ELG Listening, Attention and understanding, page 11, DfES, 2024).

Given that there are no official guidelines, teachers could use an evidence-centred design [ECD] (Mislevy & Haertel, 2007; Snow et al., 2019) based on children’s play activities in the classroom, for example bathing or dressing a doll; doing simple cookery; or following a route in the playground. One of the aspects of ECD is assessment-designing in which teachers provide the content, and focus on the student to observe what skills they are developing. To do this the teacher has to establish how skills are measured using an evidence model, and as these have not been previously measured, teachers will need to devise their own criteria. Additionally, teachers can use a task model using situations to elicit behaviours, for example dressing up, or setting a table and monitor and assess children's skills from their play (Clarke-Midura et al., 2021).

Utilising many of the areas of learning from Table 1, a scheme of work/ lesson plan was designed for an activity (icing biscuits) for 2–4-year-old children at a coding club (Hagon et al, 2020, see Figure 2). The scheme of work contains CT concepts, learning outcomes and suggestions for vocabulary; while the lesson plan section contains lists of equipment, and assessments of progress using an 'I can…' approach (Lilly et al, 2014). In many schools, there are facilitated activities where children use self-assessment and peer assessment so that the onus is not always on the practitioners to determine progress. However, as mentioned earlier, there must be an element of scaffolding to the learning to ensure that the children are secure in the most basic knowledge of the practitioners' expectations. The goal is to set the children up for success with '*hard fun'* (Papert, 1988) but to ensure this, teachers need to create activities in which children feel secure in their abilities (Smiley & Dweek, 1994), and create self-assessments in which individual children can identify task-achievement (see Figure 2, Assessment/ Progress).



Icing a Biscuit			
CT themes Algorithms Sequencing Debugging Resilience **National Curriculum Themes** Maths English Creative expression	**Tasks for teacher** 1. Intro – what is an algorithm? 2. What equipment do we need? 3. Provide ingredients 4. Ask children to measure ingredients using spoons 5. What do we do now? (sequencing) 6. Debugging (collaboration) 7. Make the icing and coat the biscuits 8. Review the learning outcomes	**Equipment required** Large bowl or individual bowls Spoons of different sizes Distraction equipment, e.g. toy, hairbrush, book Plain biscuits Icing sugar Jug of warm water Named paper towel	**Assessment/Progress** I can pick the correct equipment I can use the correct equipment I can measure ingredients I can give instructions I can follow instructions I can collaborate with others I can spot mistakes I can fix mistakes I can sequence the task
Learning Outcomes • To learn about algorithms • To listen to instructions • To follow a task • To identify and correct errors	**Key Vocabulary** Algorithms Debugging Sequence Sequencing	**Additional resources** Grid for sticking pictures (3 or 4 grids) Pictures to sequence the task.	**Teacher/TA assessment** Emerging (Baseline) – needs 1:1 help to complete task. Expected – child can do all above but with some prompts from adult Exceeded – independently completes task **Score** 1: Emerging skills 2: Expected – work independently 3: Exceeded (greater depth) – can transfer skills

Figure 2 Scheme of work/ lesson plan for a CT skills activity in preschool

This lesson plan (Figure 2) contains much of the information needed to both teach and assess a common play activity but also includes a CT skills focus. The plan includes CT themes; the tasks for the teacher or parent (assessment design); the equipment needed; the learning outcomes; the key vocabulary (which may be used depending on the age of the children); and additional resources needed. The lesson plan also gives a breakdown on the skills needed to complete the task (an evidence model) which can be assessed by the teacher or can be used by the children themselves to help them self-assess. If evidence of the children's abilities is required for the children's reports, then a paper activity can be completed after the task in which children are asked to sequence the order in which they completed the activity. Figure 3 shows a two-year-old icing a biscuit followed by another child in the group sticking four pictures showing the order in which the girls made icing and iced a biscuit (Hagon et al., 2020). The pictures showed a box of icing sugar, a jug of water, the icing mixture, and the icing on the biscuit.

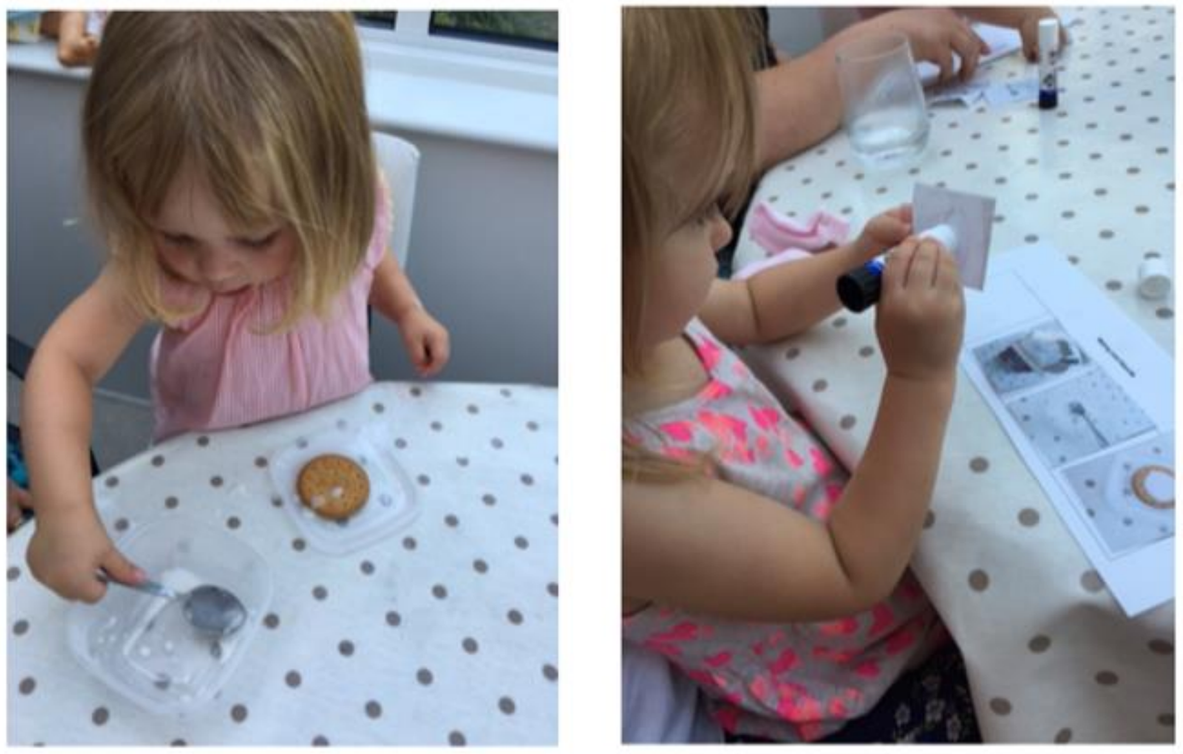


Figure 3: Icing a biscuit activity in a group of two-year-old children: the four pictures had to be stuck in the correct order to assess whether the children remembered or could work out the sequence of the activity (Hagon et al., 2020)

In Hagon's article (2020), the children were encouraged to communicate collaboratively, making suggestions for the equipment used, the sequence of the activity, and to debug if there were any errors (e.g. the icing was too runny). The activity also covers a number of the early learning goals: Maths as the children measured out spoonfuls of the icing sugar and the water; Physical Development, as the children used fine motor skills to complete the task; Communication and Language skills, as the children had to follow instructions, interact with each other, and ask and answer questions; and PSHE, as the children had to self-regulate their behaviours and build relationships within the group.

This activity as well as others, e.g. getting dressed or bathing the doll, was used in research on preschool coding clubs by Critten et al (2021) in which the children were assessed using a three-point scale similar to the one displayed in Figure 2 Teacher/ TA Assessment, and is often used in the early years foundation stage (EYFS) assessment reports in England at present (DfES, 2020). These have now been updated (see DfES, 2024) but the wording is very similar, and the scoring is still relevant.

Some educationists may disagree with the idea that CT themes could be built into the play activities in early years establishments, particularly that they should be assessed as other subjects on the curriculum. However, the teaching and assessment of CT skills can be organised within schools by the subject coordinators, the Ed. Techs, particularly within reception classes with children aged 4-5 years. A gradual build-up of CT skills could aid the development of computer science in later years but may also help to develop other associated skills in STEM subjects.

6. Discussion

This theoretical article was designed to help and encourage teachers and other practitioners to understand the importance of developing CT concepts in an increasingly digital world in which most children have already become consumers. While programming and coding in elementary forms are already part of the curriculum in schools around the world, researchers have suggested that teaching the basics of CT could encourage children to plan and think more explicitly rather than just take part in play activities.

One of the aims of this article was to show a way in which CT activities can be incorporated into the early learning guides in the English National Curriculum, so that preschool teachers were not expected to add to already crowded curriculum (Wang et al., 2023). Examples of CT concepts shown in Figure 1 are addressed by researchers such as Lee et al.., (2023) who suggested appropriate activities such as ‘*puzzle, block play… separate steps of an activity such as handwashing…*’ but did not elaborate in how these tasks could be utilised in lessons or how they could be assessed. Similarly, Saxena et al., 2019, suggested that pattern recognition could encourage children’s sense of order (identifying and sequencing) but did not give guidelines to teachers in how this could be done. The question here is not just providing these activities but how they can provide a base in which CT is introduced, and this calls on the skills of the teaching staff to talk to the children and question them about their ideas in order to focus on CT concepts. In other words, these concepts need to be taught explicitly within those play situations, and in such a way that children can learn individually or within group situations (Vygotsky et al., 1980) promoting communication and collaboration which is one of the key targets of the early learning guides.

One of the problems identified with incorporating CT into the early years guides is that teachers often do not have the training or the motivation or indeed the awareness of how these concepts can improve or encourage children's planning and thinking skills (Arfe, 2019). Introducing these concepts are said to be fundamental in teaching computer sciences in later education (Ciftici et al, 2019), and a number of teachers after undergoing training understood how CT could be introduced in other areas of the curriculum (Cimsir et al 2024; Dong, 2019). Ideas for subject cross-overs can be seen in Table 1, where subjects like pattern recognition and sequencing already are part of the ELG, but outdoor activities such as in a playground, nursery garden or forest school can be utilised to encourage the children to learn direction, route-making and landmarks which will aid them when working on software such as Scratch (Critten et al., 2021). These unplugged activities do not require expensive hardware, or even robotic toys, as CT concepts can be encouraged with basic classroom equipment or paper and pencil tasks (Messer et al., 2018), however, they do require planning so teachers need to take time putting together their ideas into project schemes and lesson plans such as the one in Figure 2 (see Dong, 2019).

The other aim of the article was for the assessment of CT and if children are able to understand the basics of CT concepts. Using concrete tasks and appropriate play activities will help children to develop an understanding of the concepts alongside the ELG targets in Table 1 and teachers can communicate with and question individual children and in groups to both teach and assess children's progress (Vygotsky et al., 1980). Utilising Allsop's principles of using the breakdown of the tasks as a way of assessing children's understanding of the individual steps (Allsop, 2019) the lesson plan in Figure 2 gives examples of assessment of progress using the ‘I can…’ method (Lilly et al., 2014). In this way teachers can mediate by supporting students in their learning and by encouraging them to identify and find solutions when problems arise (Wang et al, 2023).

7. Conclusion

At present, there is a mismatch between researchers who suggest that CT should be part of the Early Years Framework; many teachers who know very little about CT skills; and UK government guidelines which ignore the early development of CT skills for this age group. CT is referenced in other areas of the national curriculum in later years of a child’s academic journey but in the Early Years Framework, there are many CT themes that can be/ are already looked at but not necessarily expanded upon to allow a more in-depth introduction for children.

There is a huge opportunity to embrace CT concepts in early years education and align them with the core national curriculum areas to avoid duplication of effort and overwhelming workloads. If the activities are taught in a similar vein to the basic lesson outlined in here, the assessment opportunities present themselves very easily for teaching staff. Children and staff then have clear expectations of the baseline of knowledge in CT and ultimately computer sciences, and a pathway to encourage children to progress in their studies. With these skills being taught to children at the ages of 2-5 years, we would hope that these transferable skills become intrinsic elements of their personalities which will help them as they journey through their academic life and beyond to enter a world rich in STEM experiences and potentially careers.

References

Allsop, Y. (2019). Assessing computational thinking process using a multiple evaluation approach. International Journal of Child Computer Interaction, 19, 30–55. https://doi.org/10.1016/j.ijcci. 2018.10.004

Bakala, E., Gerosa, A., Hourcade, J. P., & Tejera, G. (2021). Preschool children, robots, and computational thinking: A systematic review. *International Journal of Child-Computer Interaction*, *29*, 100337. https://doi.org/10.1016/j.ijcci.2021.100337

Barr, V & Stephenson, C. (2011). Bringing computational thinking to K-12: what is involved and what is the role of the computer science education community? *ACM Inroads, 2*, 1, 48-54. https://doi.org/10.1145/1929905

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

Bers, M. U., González-González, C., & Armas–Torres, M. B. (2019). Coding as a playground: Promoting positive learning experiences in childhood classrooms. *Computers & Education*, *138*, 130-145. https://doi.org/10.1016/j.compedu.2019.04.013

Brassard, M., & Boehm, A. E. (2007). Assessment of emotional development and behavior problems. *Brassard MR, Boehm AE. Preschool assessment: principles and practices. New York: Guilford Publications, Inc*, 508-576.

Caeli, E. N., & Yadav, A. (2020). Unplugged approaches to computational thinking: A historical perspective. TechTrends, 64(1), 29-36. https://doi.org/10.1007/s11528-019-00410-5

Çiftci, S., & Bildiren, A. (2020). The effect of coding courses on the cognitive abilities and problem-solving skills of preschool children. *Computer science education, 30*(1), 3-21. https://doi.org/10.1080/08993408.2019.1696169

Clarke-Midura, J., Lee, V. R., Shumway, J. F., Silvis, D., Kozlowski, J. S., & Peterson, R. (2023). Designing formative assessments of early childhood computational thinking. *Early Childhood Research Quarterly*, *65*, 68-80. https://doi.org/10.1080/08993408.2021.1877988

Clements, D. H., & Gullo, D. F. (1984). Effects of computer programming on young children's cognition. Journal of Educational Psychology, 76(6), 1051. https://doi.org/10.1037/0022-0663.76.6. 1051

Critten, V., Hagon, H., & Messer, D. (2021). Can pre-school children learn programming and coding through guided play activities? A case study in computational thinking. *Early Childhood Education Journal*, 1-13. https://doi.org/10.1007/s10643-021-01236-8

Department for Education. (2013). National curriculum in England: computing programmes of study. https://www.gov.uk/government/ publications/national-curriculum-in-england-computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study

DfES, (2024). Early years foundation stage statutory framework Early years foundation stage (EYFS) statutory framework - GOV.UK (www.gov.uk)

Dong, C. (2018). 'Young children nowadays are very smart in ICT' – preschool teachers' perceptions of ICT use. *International Journal of Early Years Education*. https://doi.org/10.1080/09669760.2018.1506318

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 and Education, 63, 87–97. https://doi.org/10.1016/j.compedu.2012.11.016

Gifford, S. (2005). *Teaching Mathematics 3-5: Developing Learning In The Foundation Stage: Developing learning in the Foundation Stage*. McGraw-Hill Education (UK).

Gomes, T. C. S., Falcão, T. P., & Tedesco, P. C. D. A. R. (2018). Exploring an approach based on digital games for teaching programming concepts to young children. International Journal of Child-Computer Interaction, 16, 77–84. https://doi.org/10.1016/j. ijcci.2017.12.005

Gopnik, A. (2010). How babies think. *Scientific American*, *303*(1), 76-81. https://www.jstor.org/stable/10.2307/26002102

Hagon, H. (2024). Unplugged tots: Screen-free pre-computer coding skills for 2½ to 8 year olds. https://www.unpluggedtots.com

Hagon, H., Critten, V. & Messer, D. (2020). An introduction to unplugged coding for preschool children. *Hello World.* 13. pp 57-59. https://helloworld.raspberrypi.org/issues/13

Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. Computers & Education, 82, 263-279. https://doi.org/10.1016/j.compedu.2014.11.022

Jacobson, L. (2016, April 5). Never Too Young To Code. School Library Journal [2016, October 3] http://www.slj.com/2016/04/technology/never-too-young-to-code/

Lee, J., Joswick, C., & Pole, K. (2023). Classroom Play and Activities to Support Computational Thinking Development in Early Childhood. *Early Childhood Education Journal*. https://doi.org/10.1007/s10643-022-01319-0

Lilly, J., Peacock, A., Shoveller, S., & Struthers, D. R. (2014). Beyond levels: Alternative assessment approaches developed by teaching schools. National College for teaching and leadership. Beyond levels: alternative assessment approaches developed by teaching schools (ucl.ac.uk)

Messer, D., Thomas, L., Holliman, A., & Kucirkova, N. (2018). Evaluating the effectiveness of an educational programming intervention on children's mathematics skills, spatial awareness and working memory. *Education and Information Technologies, 23,* 2879-2888. https://doi.org/10.1007/s10639-018-9747-x

Mislevy, R. J., & Haertel, G. D. (2006). Implications of evidence-centered design for educational testing. *Educational measurement: issues and practice*, *25*(4), 6-20. https://doi.org/10.1111/j.1745-3992.2006.00075.x

Newcombe, N., & Huttenlocher, J. (1992). Children's early ability to solve perspective-taking problems. *Developmental psychology*, *28*(4), 635. https://psycnet.apa.org/doi/10.1037/0012-1649.28.4.635

OECD (2023). Empowering young children in the digital age, starting strong. Paris: OECD Publishing. https://doi.org/10.1787/50967622-en

Ofcom. (2023). *Children's Media Use and Attitudes Report 2023*. https://www.ofcom.org.uk/__data/assets/pdf_file/0027/255852/childrens-media-use-and-attitudes-report-2023.pdf

Otterborn, A., Schönborn, K., & Hultén, M. (2019). Surveying preschool teachers' use of digital tablets: general and technology education related findings. *International journal of technology and design education*, *29*(4), 717-737. https://doi.org/10.1007/s10798-018-9469-9

Otterborn, A., Schonborn, K., & Hulten, M. (2020). Investigating preschool educators' implementation of computer programming in their teaching practice. Early Childhood Education Journal, 48, 253–262. https://doi.org/10.1007/s10643-019-00976-y

Palmér, H. (2017). Programming in preschool-with a focus on learning mathematics. *International Research in Early Childhood Education*, 8(1), 75–87 ISSN 1838-0689 education.monash.edu/research/publications/journals/irece

Papert, S. (1988). Does easy do it? Children, games, and learning. Game Developer Magazine, 5(6), 88–89.

Papert, S. & Harel, I. (1991). Situating constructionism. http://www. papert.org/articles/SituatingConstructionism.html

Piaget, J., & Inhelder, B. (1973). Memory and intelligence. Routledge and Kegan Paul

Relkin, E., De Ruiter, L., & Bers, M. U. (2020). TechCheck: Development and validation of an unplugged assessment of computational thinking in early childhood education. *Journal of Science Education and Technology, 29*(4), 482-498. https://doi.org/10.1007/s10956-020-09831-x

Sentance, S., & Csizmadia, A. (2015, July). Teachers' perspectives on successful strategies for teaching Computing in school. In *IFIP TC3 Working Conference 2015: A New Culture of Learning: Computing and Next Generations*. http://www.ifip2015.mii.vu.lt/proceedings#.WFpvmlzgmqQ

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

Siegler, R. S. (2016). Continuity and change in the field of cognitive development and in the perspectives of one cognitive developmentalist. *Child Development Perspectives*, *10*(2), 128-133. https://doi.org/ 10.1111/cdep.12173

Smedley, S., & Hoskins, K. (2020). Finding a place for Froebel's theories: early years practitioners' understanding and enactment of learning through play. *Early Child Development and Care*, *190*(8), 1202-1214. https://doi.org/10.1080/03004430.2018.1525706"https://doi.org/10.1080/03004430.2018.1525706

Smiley, P. A., & Dweck, C. S. (1994). Individual differences in achievement goals among young children. *Child development, 65*(6), 1723-1743. https://www.jstor.org/stable/1131290?seq=1&cid=pdf-reference#references_tab_contents

Snow, E., Rutstein, D., Basu, S., Bienkowski, M., & Everson, H. T. (2019). Leveraging evidence-centered design to develop assessments of computational thinking practices. *International Journal of Testing*, *19*(2), 103-127. https://doi.org/10.1080/15305058.2018.1543311

Su, J., & Yang, W. (2023). A systematic review of integrating computational thinking in early childhood education. *Computers and Education Open*, *4*, 100122. https://doi.org/10.1016/j.caeo.2023.100122"https://doi.org/10.1016/j.caeo.2023.100122

Vygotsky, L. S., Cole, M., John-Steiner, V., & Souberman, E. (1980). Mind in society: The development of higher psychological processes. Cambridge: Harvard University Press

Waterman, K. P., Goldsmith, L., & Pasquale, M. (2020). Integrating computational thinking into elementary science curriculum: An examination of activities that support students' computational thinking in the service of disciplinary learning. *Journal of Science Education and Technology*, *29*(1), 53-64. https://doi.org/10.1007/s10956-019-09801-y

Wang, X. C., Choi, Y., Benson, K., Eggleston, C., & Weber, D. (2023). Teacher's role in fostering preschoolers' computational thinking: An exploratory case study. *Early Education and Development*, *32*(1), 26-48. https://doi.org/10.1080/10409289.2020.1759012

Wang, X. C., Flood, V. J., & Cady, A. (2021). Computational thinking through body and ego syntonicity: Young children's embodied sense-making using a programming toy. In *Proceedings of the 15th International Conference of the Learning Sciences-ICLS 2021.*. International Society of the Learning Sciences.

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

Yücelyiğit, S. (2023). Coding practices in early childhood classrooms: A phenomenological study with ECE teachers in Turkey. *Issues in Educational Research*, 33(3).

Computational Thinking Assessment in Primary and Secondary Education: A Meta-Synthesis of Tools, Methods And Pedagogical Approaches

Sanna Forsström[1]

Melissa Bond[2]

[1]*University of Stavanger, Norway*

[2]*University College London, UK*

DOI: 10.21585/ijcses.v6i4.208

Abstract

Despite the widespread adoption of computational thinking (CT) across educational levels, challenges persist in its assessment due to diverse definitions, frameworks, and practical applications in classroom settings. This meta-synthesis investigates the assessment of computational thinking (CT) in primary and secondary education, synthesising evidence from 12 reviews across five international databases, focusing on tools, methods, and pedagogical practices employed in assessing CT, with the aim to outline practical approaches for evaluating CT components. The review delves into the primary focuses of these syntheses, the CT skills and components assessed, and the methods and tools utilised, identifying gaps in current practices. The findings highlight a prevalent focus on programming skills, with less emphasis on cognitive processes and collaborative aspects of CT. The synthesis also points to the need for developing assessment tools and methods that encompass the broader spectrum of CT skills, suggesting avenues for future research and practical application in educational settings.

Keywords: Assessment, Computational Thinking, Review of reviews, Meta-synthesis

1. Introduction

The shift towards digital education is reshaping the educational field, introducing new teaching methodologies and broadening the horizons of learning. As the educational sector navigates through advancements in artificial intelligence, data management, cloud computing, and green technologies, teachers encounter various obstacles. These include managing the classroom, assessing student learning outcomes, dealing with ethical issues, and the effective use of digital tools (González-Pérez & Ramírez-Montoya, 2022). Central to overcoming these challenges is enhancing students' computational thinking (CT) skills, which are essential for promoting teamwork, critical analysis, and ethical decision-making (González-Pérez & Ramírez-Montoya, 2022; Ye et al., 2022).

The widespread adoption of computational thinking in primary and secondary education highlights its significance (Balanskat & Engelhardt, 2015). Nonetheless, there exists a diversity in how CT is understood among teachers, policymakers, and the media. This ranges from focusing on the core principles of computer science to its application across various disciplines (Lodi & Martini, 2021). The evolution of CT spans from Papert’s (1980) vision of empowering individuals and enriching their understanding of complex topics through computational strategies to Wing’s (2006) perspective. Wing viewed CT as not merely a set of technical skills but a comprehensive methodology for problem-solving, system design, and understanding human behaviour through computational principles, such as the ways individuals think, act, and interact. This underscores the necessity of integrating CT into education as a foundational skill alongside reading, writing, and arithmetic.

Following Wing's (2006) foundational work, numerous researchers have aimed to clearly define computational thinking (CT) and outline its key components (Lodi, 2020). Despite the creation of various frameworks to categorise CT into multidimensional constructs, the task of assessing CT in primary and secondary education remains challenging. The diverse definitions and classifications, particularly those focusing on cognitive processes, such as logical reasoning or creativity, present difficulties for teachers in both understanding and evaluating CT.

Previous systematic reviews have examined the assessment of CT from different angles. Much of this research has focused on the theoretical aspects of CT assessment as part of research methodology (e.g., Araujo et al., 2016; Liu et al., 2021; Pan et al., 2023), with less attention given to practical implementation in the classroom. When studies do explore CT assessment within educational contexts, they often emphasise programming tasks using specific tools (e.g., Varghese & Renumol, 2023; Tan et al., 2023; da Cruz Alves et al., 2019) or concentrate on particular education levels (e.g., Fagerlund, 2021).

This meta-synthesis seeks to clarify these complexities by synthesising evidence syntheses that approaches the assessment of computational thinking (CT) in primary and secondary education with a variety of tools, methods, and definitions. This includes a focus on both research methodologies and methods used directly in schools. By interpreting these findings, our goal is to aggregate and thematically synthesise insights to outline practical assessment tools, methods, and pedagogical practices.

The central research question of this meta-synthesis is:

> *How can we understand the assessment of computational thinking in primary and secondary education through the synthesis of evidence syntheses?*

This inquiry also involves examining the connections between different methods and tools with specific CT components. To support this exploration, the following sub-questions guided the review:

1. What is the primary focus of these evidence syntheses?
2. Which CT skills and components are most frequently assessed?
3. What are the less commonly assessed CT components?
4. What methods and tools are used for assessing various CT skills/components?
5. How can different methods and tools be connected to specific CT components based on our interpretation of the results?
6. Which pedagogical practices for the formative assessment of CT are identified?
7. What limitations regarding assessment methods, tools, and pedagogical practices are reported?

2. Methodology

To address the research questions, a 'review of reviews' methodology was adopted, focusing on the selection, data extraction, and synthesis of evidence syntheses (Booth et al., 2022). This approach, recognised as a tertiary review (Kitchenham et al., 2009), aggregates findings from systematic reviews and follows documentation standards in accordance with PRISMA protocols (Page et al., 2021), ensuring rigour and transparency, with all search information available to be downloaded from the OSF[1].

2.1 Search Strategy and Selection of Studies

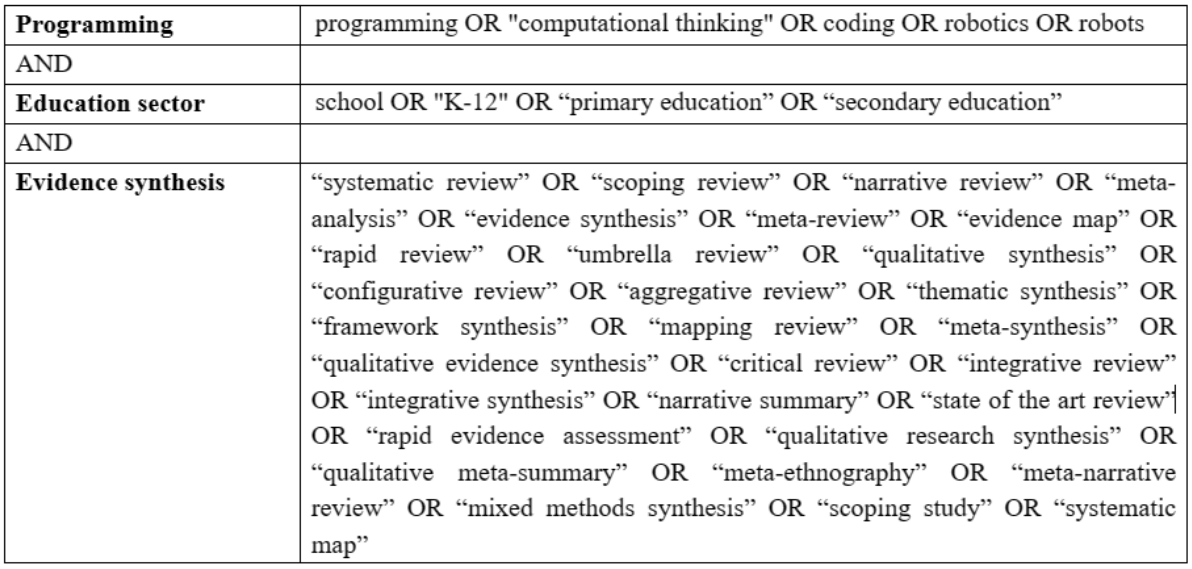
The evidence syntheses included in this study were identified through a wider scoping meta-review of programming, robotics and computational thinking studies, the methodological framework of which was inspired by prior tertiary analyses (Bond et al., 2024; Buntins et al., 2023).

Development of the Search String

The formulation of the search string (see Fig. 1) was adapted from earlier tertiary reviews (Bond et al., 2024; Buntins et al., 2023) and focused on programming, computational thinking, and robotics within the K-12 educational framework, alongside various evidence synthesis methodologies (Sutton et al., 2019). Unlike some reviews that specialise on a single secondary research type, such as meta-analyses (Higgins et al., 2012), this study aimed to encompass the

[1] https://osf.io/m8v3r/?view_only=b3c9360dfc2641a8b11c8d2d9924db50

entire spectrum of evidence synthesis techniques. This inclusive approach was chosen to fully map the domain without constraining the review to specific secondary research methodologies.



Programming	programming OR "computational thinking" OR coding OR robotics OR robots
AND	
Education sector	school OR "K-12" OR “primary education” OR “secondary education”
AND	
Evidence synthesis	“systematic review” OR “scoping review” OR “narrative review” OR “meta-analysis” OR “evidence synthesis” OR “meta-review” OR “evidence map” OR “rapid review” OR “umbrella review” OR “qualitative synthesis” OR “configurative review” OR “aggregative review” OR “thematic synthesis” OR “framework synthesis” OR “mapping review” OR “meta-synthesis” OR “qualitative evidence synthesis” OR “critical review” OR “integrative review” OR “integrative synthesis” OR “narrative summary” OR “state of the art review” OR “rapid evidence assessment” OR “qualitative research synthesis” OR “qualitative meta-summary” OR “meta-ethnography” OR “meta-narrative review” OR “mixed methods synthesis” OR “scoping study” OR “systematic map”

Figure 1. Meta-synthesis search string

Inclusion/Exclusion Criteria and Screening Process

The initial search was executed in April 2023, with follow-up searches up to January 17, 2024, ensuring comprehensive literature coverage. The platforms and databases searched were the Web of Science, Scopus, EBSCOHost (including ERIC), and Progress, as these have been found suitable and comprehensive for conducting evidence synthesis in the wider social sciences (Gusenbauer & Haddaway, 2020). The choice was made not to search the ACM Digital Library as well, as a recent tertiary review of AI in Education (Bond et al., 2024) included only one extra study from that database that was not found through other methods. Instead, the OpenAlex platform (Priem et al., 2022) was also searched via evidence synthesis software EPPI

Reviewer (Thomas et al., 2023), which indexes approximately 209 million publications. Forward and backward snowballing was also undertaken using OpenAlex.

The search yielded 4,369 records, which were imported into EPPI Reviewer as text or RIS files (see Fig. 2). An initial screening removed 485 duplicates. Two reviewers screened the same 200 titles and abstracts against predefined inclusion/exclusion criteria (see Table 1) to ensure inter-rater consistency. After reaching full agreement, the reviewers assessed the titles and abstracts of the remaining 3,684 records.

The inclusion criteria focused on K-12 programming or computational thinking evidence syntheses, published in English after 2010, identifying 195 articles to screen on full text. To confirm inter-rater reliability, ten additional articles were reviewed by both, achieving unanimous agreement. Ultimately, 120 evidence syntheses were selected for detailed analysis and synthesis in EPPI Reviewer. From these, 12 studies that focused solely on the assessment of computational thinking were identified for data extraction and synthesis.

Table 1. Inclusion/exclusion criteria

Inclusion Criteria	**Exclusion Criteria**
Published 2010 - 2023	Published before January 2010
Formal K-12 settings	Higher education, informal learning
A form of secondary research	Primary research or a review without a method section
English language	Published in a language other than English
Journal article or conference paper	

Applications of computational thinking assessment	Bibliometric reviews, editorials, book chapters, dissertations, posters, abstracts, workshop papers

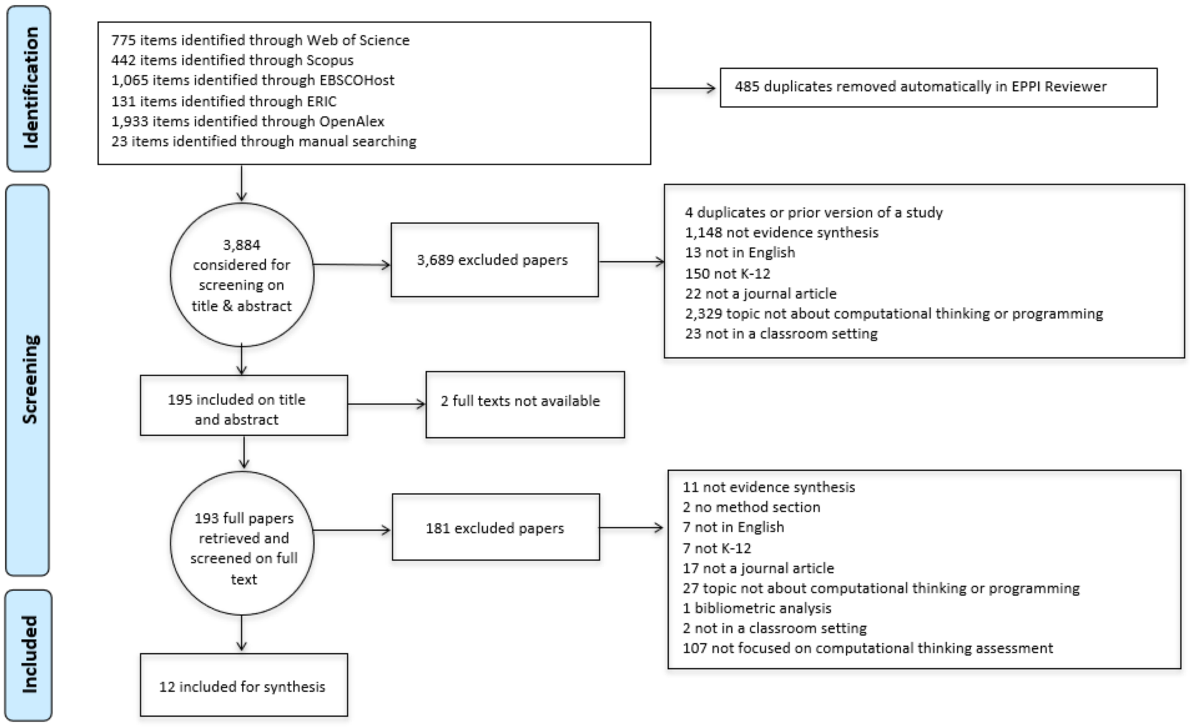

Identification
Screening
Included
775 items identified through Web of Science
442 items identified through Scopus
1,065 items identified through EBSCOHost
131 items identified through ERIC
1,933 items identified through OpenAlex
23 items identified through manual searching
485 duplicates removed automatically in EPPI Reviewer
3,884 considered for screening on title & abstract
3,689 excluded papers
4 duplicates or prior version of a study
1,148 not evidence synthesis
13 not in English
150 not K-12
22 not a journal article
2,329 topic not about computational thinking or programming
23 not in a classroom setting
195 included on title and abstract
2 full texts not available
193 full papers retrieved and screened on full text
181 excluded papers
11 not evidence synthesis
2 no method section
7 not in English
7 not K-12
17 not a journal article
27 topic not about computational thinking or programming
1 bibliometric analysis
2 not in a classroom setting
107 not focused on computational thinking assessment
12 included for synthesis

Figure 2. PRISMA diagram

2.2 Data Extraction

A data extraction coding tool was slightly modified from that of Bond et al. (2024)[2] and was developed in EPPI Reviewer. Codes included publication information (e.g., publication name and year published), authorship characteristics (e.g., country affiliation), review type (as self-identified by the authors), specific review focus if present (geographical or subject area), methodological characteristics (e.g., number of studies included), and benefits and challenges, which were inductively coded. Following the foundational work of Wing (2006), diverse

[2] See https://osf.io/m8v3r/?view_only=b3c9360dfc2641a8b11c8d2d9924db50 for the full coding tool.

definitions and categorisations of computational thinking (CT) have emerged (Lodi, 2020; Lodi & Martini, 2021). An initial review of the included evidence syntheses revealed a variety of definitions and categorisations. Consequently, we opted for inductive coding of the CT components used across these studies, developing our categories for a more coherent analysis. To thoroughly explore our research questions, we inductively coded all CT components/skills mentioned in the evidence syntheses. Additionally, we identified and coded the methods and tools utilised in CT assessment, along with any discussions on pedagogical practices and noted limitations. These coded elements formed the basis of a table that facilitated our synthesis process. Connecting methods and tools to assess CT components/skills involved analysing data from studies that explicitly detail these relationships. This included examining various tables and diagrams from sources such as Fagerlund et al. (2021), Babazadeh & Negrini (2022), da Cruz Alves et al. (2019), and Varghese and Renumol (2023), which illustrate the links between CT components and the assessment methods used. These elements were meticulously read, coded, and interpreted for relevance.

2.3 Synthesis

The findings were synthesised narratively (Petticrew & Roberts, 2006), and a tabulation of included studies was developed (see Appendix 1). Additional tables are presented within the text or as appendices, created using Word and Excel, supplemented by narrative descriptions. Regarding RQ2, given the wide array of definitions and descriptors for CT skills, each component was listed individually, merging similar ones and grouping the most frequently mentioned components into themes. RQ 5 was answered by identifying, examining and interpreting results presented in the studies analysing connections between CT components and assessment tools and methods. This process was complex due to the variability in how studies defined components and categorised their findings. Our interpretation of these results, aligned with our categorisation, aimed to clarify the practical application and utility of these methods in the CT assessment landscape. To map the results visually and to provide an openly accessible database to practitioners and researchers, a web database was created using the EPPI Visualiser app[3]. This allows users to explore the data by creating frequency and cross-tabulations, view all raw coding, and directly export and save metadata.

2.4 Limitations

Whilst every attempt was made to conduct this meta-synthesis as transparently and rigorously as possible, methodological limitations must be acknowledged. A protocol was not pre-registered; however, all metadata and coding are openly accessible via the OSF[4] and the EPPI Visualiser database. Five international databases were searched, along with snowballing. However, it is possible that potential includes were missed, given the more Western focus of the research indexed in these platforms (Mertala et al., 2022). Likewise, this review was limited to the period 2010-2023 and English language-only publications due to project resources. However, in the future, research in languages other than English should be included to help

[3] https://eppi.ioe.ac.uk/eppi-vis/login/open?webdbid=597
[4] https://osf.io/m8v3r/?view_only=b3c9360dfc2641a8b11c8d2d9924db50

mitigate potential bias and improve generalisability (Bahji et al., 2022; Stern & Kleijnen, 2020). The choice to exclude evidence syntheses without a method section might also have led to the exclusion of more conceptual reviews, although this choice was made to include reviews that attested to being conducted to a particular standard of rigour (Bond et al., 2024).

3. Findings

3.1 General publication characteristics

The 12 studies included in this meta-review (see Appendix 1) were undertaken by the first authors from Europe (Babazadeh & Negrini, 2022; Fagerlund et al., 2021; Tikva & Tambouris, 2021), North America (Liu et al., 2021; Pan et al., 2023; Tan et al., 2023; Tang et al., 2020), South & Central America (Araujo et al., 2016; da Cruz Alves et al., 2019; Muñoz et al., 2023), and Asia (Haseski & Ilic, 2019; Varghese & Renumol, 2023), demonstrating a widespread interest in and appreciation of the importance of understanding how CT research is being undertaken in order to best inform practice. However, less than half (42%, n = 5) are available open access, which limits the extent to which the results of these studies can be applied in practice or used to inform policy. The majority of reviews (*n* = 10) included both primary and high school students, five of which also included higher education (e.g., Muñoz et al., 2023). Two evidence syntheses explored CT in primary schools only; Fagerlund et al. (2021) concentrated on students using Scratch, while Liu et al. (2021) explored research methodologies for assessing CTs.

3.2 RQ1: What is the primary focus of the evidence syntheses?

Although the topic of each included evidence synthesis is the assessment of computational thinking in schools, they all have a slightly different focus (see Appendix 1). Some syntheses evaluated CT skills through programming activities, with Da Cruz Alves et al. (2019) investigating block-based languages and Fagerlund et al. (2021) focusing on Scratch, whilst other studies focused on specific methods or tools for assessing CT. Haseski and Ilic (2019) investigated the efficacy of paper-and-pencil tests, Varghese and Renumol (2023) assessed the use of video games, and Pan et al. (2022) examined the utility of think-aloud interviews for understanding student thought processes. Tan et al. (2023) explored the application of machine learning in assessment processes, while Varghese and Renumol (2023) evaluated digital games for comprehensive CT assessment, highlighting the increasing importance of digital media in education. Babazadeh and Negrini (2022) uniquely focused on the European context, emphasising geographical specificity in their review.

These syntheses vary in their emphasis, ranging from the exploration of research methodologies in CT assessment to the provision of actionable tools for teachers. This range underscores the complexity of adapting intricate research methodologies for practical use in the classroom, revealing a disconnect between theoretical research and practical teaching needs. The detailed nature of some research methods presents challenges for straightforward application by teachers, indicating a need to bridge the divide between academic research and classroom practice to make CT assessment both meaningful and feasible.

3.3 RQ2: Which CT skills and components are most frequently assessed?

Based on our thematic synthesis we divided the assessed CT skills and components into six main categories: 1) Core CT Components, 2) Programming Concepts, 3) Cognitive Processes, 4) Problem-Solving Strategies, 5) Collaborative and Communicative Skills, and 6) Dispositions and Attitudes (see Table 2).

The evidence syntheses in this corpus focused on four core CT components; abstraction, algorithmic thinking, decomposition and pattern recognition (see Table 2). Abstraction was the most prevalent, which relates to identifying and extracting relevant information while ignoring irrelevant details, followed by algorithmic thinking (developing a step-by-step solution to a problem), decomposition (breaking down a complex problem into smaller, more manageable sub-problems), and pattern recognition (identifying similarities, differences and patterns within and across problems).

Table 2. Skills and components reported within evidence syntheses

Core CT components	***n***	**% of reviews**
Abstraction	7	58%
Algorithmic thinking	6	50%
Decomposition	6	50%
Pattern recognition	4	33%
Programming concepts		
Sequencing	7	58%
Conditionals	6	50%

Parallelism	6	50%
Loops/iteration	5	42%
Variables & Data Representation	5	42%
Modularity	4	33%
Events & Synchronization	3	25%
Cognitive processes		
Logic & Reasoning	5	42%
Creativity and Innovation	3	25%
Critical Thinking	1	8%
Problem-solving strategies		
Problem-solving	5	42%
Debugging	3	25%
Testing	3	25%
Efficiency	2	16%
Collaborative & communication skills		
Collaboration & cooperation	3	25%
Communication	1	8%
Dispositions & attitudes		
Interest & engagement	1	8%

Seven specific programming concepts were identified (see Table 2), with sequencing (arranging steps in a logical order) the most frequent, followed by conditionals (making decisions based on specific conditions), parallelism (executing tasks simultaneously to increase efficiency), loops/iteration (repeating a set of instructions until a specific condition is met) and variables and data representation (storing, retrieving and manipulating data). Modularity (dividing a program into smaller, reusable parts) and events and synchronization (coordinating and synchronizing actions and events) were less considered.

Three cognitive processes were identified (see Table 2), with logic and reasoning (applying logical thinking to solve problems) being the most mentioned. This was followed by creativity and innovation (e.g., Fagerlund et al., 2021), and critical thinking, which was only found in one review (Tikva & Tambouris, 2021). Problem-solving strategies were divided into general developing and applying strategies to solve problems (e.g., Varghese & Renumol, 2023), and specific task-oriented strategies: debugging, testing and efficiency (optimizing a solution for better performance). Three studies (Fagerlund et al., 2021; Haseski & Ilic, 2019; Tikva & Tambouris, 2021) focused on working with others to achieve a common goal, one study (Tikva & Tambouris, 2021) explored expressing ideas and solutions effectively, and one study (Tan et al., 2023) investigated studies that related to being motivated to learn and apply CT skills.

3.4 RQ3: What are the less commonly assessed CT components?

Several evidence syntheses highlight gaps in assessing various CT components, overemphasizing tangible programming skills at the expense of broader, complex CT skills and affective variables. Fagerlund et al. (2021) point to a predominant focus on Scratch projects, which primarily assess programming skills but overlook broader thinking skills. Liu et al. (2021) discussed the underassessment of cognitive processes, particularly visual behaviors and verbalizations in CT problem-solving, whilst Haseski and Ilic (2019) noted the lack of studies measuring CT through affective variables, such as self-efficacy and attitude.

Muñoz et al. (2023) identified debugging, simulation, and decomposition as seldom assessed CT components in educational settings, highlighting the challenge for teachers in crafting activities that accurately measure these competencies. Varghese and Renumol (2023) and Araujo et al. (2016) revealed that competencies such as conditional logic, iteration, modularity, modeling, and parallelization are minimally explored. Da Cruz Alves et al. (2019) implied an emphasis on quantifiable programming aspects rather than abstract CT concepts, a sentiment echoed by Tan et al. (2023), who points to the infrequent assessment of skills such as creativity and collaboration due to the difficulties in quantification. Similarly, Tang et al. (2020) observed a preference for assessing tangible programming skills over abstract CT components.

In summary, the literature indicates that higher-level thinking skills, affective variables, and complex CT competencies such as debugging are infrequently assessed. This is attributed to various factors, including the lack of standardized assessment models, the complexity of these skills, and the difficulty in quantifying them, especially for creative and collaborative skills.

3.5 RQ4: What methods and tools are used for assessing various CT skills/components?

Assessment methods

A variety of methods are employed in the assessment of computational thinking (CT) across the included evidence syntheses (see Appendix 1). Based on the thematic synthesis, assessment methods were categorized as direct, indirect, and innovative to capture the full spectrum of Computational Thinking (CT) strategies, aligning with their focus on either tangible outputs, cognitive processes, or the application of emerging technologies.

Four direct assessment methods were coded, with *standard tests* being administered to measure specific CT skills the most used, including multiple-choice or open-ended questions (*n* = 6, 50%; e.g., Babazadeh & Negrini, 2022), followed by *artifact analysis* in four reviews (33%), which involves evaluating the products of CT activities, such as code or digital artifacts, to determine the level of computational understanding (e.g., Pan et al., 2023). *Code analysis* was identified in two reviews (Muñoz et al., 2023; da Cruz Alves et al., 2019), which is when manual or automated code analysis is used to evaluate students' programming projects and can involve tools that specifically assess block-based programming languages like Scratch. Students undertaking *self-evaluation* of their own work or performance was found in one review (Fagerlund et al., 2021).

Four indirect assessment methods were identified. Nine reviews (75%) reported the use of *interviews and questionnaires* to gauge students' understanding and thought processes related to CT concepts and practices (e.g., Tang et al., 2020), with four reviews mentioning the use of *observations* (e.g., Tikva & Tambouris, 2021), where either researchers or educators observe students during CT activities to assess skill application in real-time. Three reviews identified think-aloud protocols, where students verbalize their thought processes while engaging in CT tasks, providing insights into their problem-solving strategies (e.g., Liu et al., 2021), and two reviews (Fagerlund et al., 2021; Pan et al., 2023) reported the use of *log data and error analysis*, where log data from digital tools or error patterns in code are analysed to understand students' learning processes and misconceptions.

Four innovative and emerging methods were mentioned in regard to CT assessment methods. Two reviews (Muñoz et al., 2023; Tikva & Tambouris, 2021) explored evidence-centered design, which is a systematic approach to developing assessments that align with the targeted competencies and skills. Additionally, two reviews (Tikva & Tambouris, 2021; Varghese & Renumol, 2023) considered the use of data mining and machine learning, where computational techniques are employed to analyze large datasets, such as responses or interaction patterns, to identify CT skills. Furthermore, one review mentioned eye-tracking (Liu et al., 2021), which offers deeper insights by tracking where and how students focus their attention during CT tasks. Another review (Tan et al., 2023) identified gamified assessments, which integrate assessment into the learning experience using game-based environments to capture data on students' CT skills.

Many studies also combined various methods to provide a more comprehensive assessment. For instance, think-aloud interviews may be used alongside programming assignments, which is a combination seen in Pan et al. (2023), or surveys might be combined with machine learning techniques to assess broader cognitive strategies and validate CT skill levels, as in Tikva & Tambouris (2021) and Varghese and Renumol (2023).

Assessment tools

In the exploration of tools used for assessing Computational Thinking (CT), the referenced evidence syntheses provide insights into a range of instruments tailored to capture the varied dimensions of CT skill development. These tools, classified into several thematic categories, serve distinct functions within the assessment process and are detailed as follows.

Programming and Development Environments: Tools such as Dr. Scratch, Scratch, Ninja Code Village, and App Inventor are frequently used for artifact analysis and development of CT skills (Fagerlund et al., 2021; Liu et al., 2021; Araujo et al., 2016; Tang et al., 2020). These environments allow for both the creation of digital artifacts and the assessment of the coding process itself.

Standardized Assessments and Testing Platforms: Various tests and tasks, including Bebras tasks, Visual Blocks Creative Computing Test, and multiple-choice or open-ended paper-and-pencil tests, are applied to evaluate specific CT competencies across a broad educational spectrum (Haseski & Ilic, 2019; Babazadeh & Negrini, 2022; Muñoz et al., 2023).

Questionnaires and Surveys: Self-efficacy scales, ability scales, and custom online questionnaires are implemented to gauge students' perceptions and self-assessed proficiency in CT, providing a subjective measure of cognitive and affective aspects of CT learning (Babazadeh & Negrini, 2022; Varghese & Renumol, 2023).

Interactive and Observational Tools: Think-aloud protocols, interviews, and observation strategies enable the assessment of students' thought processes and problem-solving strategies in real-time, offering qualitative insights into their CT approach (Fagerlund et al., 2021; Pan, 2023; Varghese & Renumol, 2023).

Robotic and Hands-on Tools: The use of robotic kits, microcontrollers, and web tools encourages the practical application of CT concepts, with students demonstrating their understanding through tangible, interactive projects (Muñoz et al., 2023; Tikva & Tambouris, 2021).

Data Analytics and Feedback Instruments: Log data, error analysis, and student response analysis through gamified assessments and evidence-centered design offer quantitative insights into students' learning processes, highlighting areas for growth and development in CT skills (Tan, 2023; Pan et al., 2023).

Emerging Technologies: Eye-tracking, game-based learning environments, and machine learning techniques represent the cutting edge of CT assessment, capturing nuanced data on how students interact with CT tasks and engage with problem-solving (Liu et al., 2021; Varghese & Renumol, 2023; Tikva & Tambouris, 2021).

3.6 RQ5: How can different methods and tools be connected to specific CT components, based on our interpretation of the results?

Only Fagerlund et al. (2021), Babazadeh & Negrini (2022), Da Cruz Alves et al. (2019), and Varghese and Renumol (2023), provided information about connections between different CT components and tools and methods used in the assessment. The connections between CT components and skills were explicitly outlined only in Fagerlund et al. (2020). We have conducted the Figure 3 based on the results in Fagerlund et al. (2020), which demonstrates the relationship between the content of Scratch programming projects evaluated in Fagerlund et al. (2020) and the methodologies employed in their assessment. This figure indicates that artifact analysis and tests are predominantly utilized for evaluating code constructs. Here, Fagerlund et al. (2020) specifically refer to the logical structures in conducted Scratch programs, such as sequences of blocks. Coding patterns, which denote combinations of code constructs functioning as broader programming units, are primarily assessed through artifact analysis. Furthermore, the assessment of programming activities in the studies included by Fagerlund et al. (2020) is mainly conducted via observations, discourse analysis, and interviews.

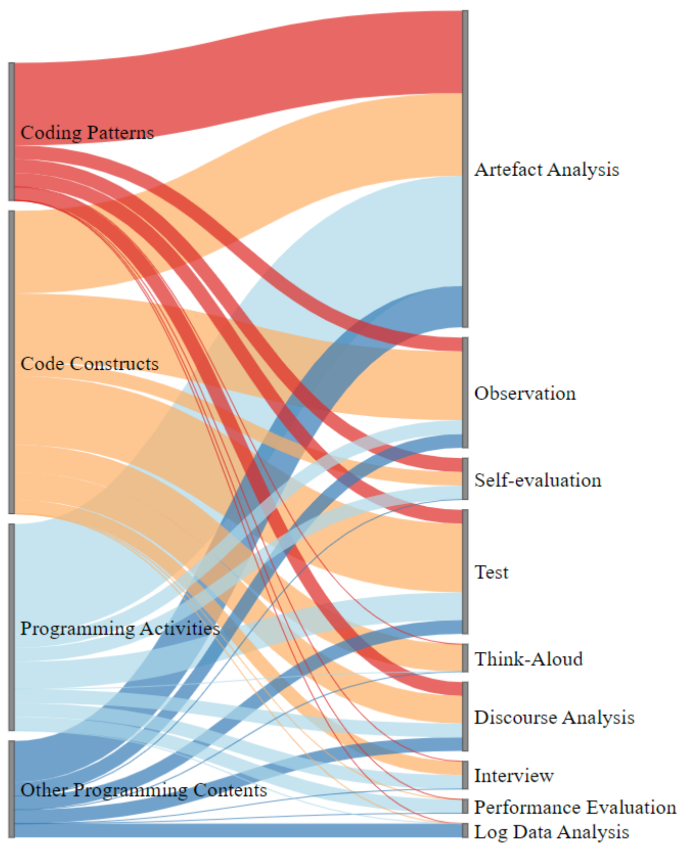


Figure 3. Our interpretation of the results in Fagerlund et al. (2020) about connections between assessed CT components and different methods.

Da Cruz Alves et al. (2019), Varghese and Renumol (2023), and Babazadeh and Negrini (2022) provide insights into the connections between assessed computational thinking (CT) components and the tools used in their assessment. Da Cruz Alves et al. (2019) focus on code analysis, thus the discussion is limited to tools appropriate for assessing codes. Varghese and Renumol (2023) specifically concentrate on video games as a tool for assessing CT, whereas Babazadeh and Negrini (2022) consider assessments solely in a European context but offer a broader range of assessment tools discussed.

The tools discussed in da Cruz Alves et al. (2019), including Hairball, Dr.Scratch, Ninja Code Village and Quizly, Fairy Assessment, were linked to CT components related to Core CT Components and Programming Concepts. These tools can quite straightforwardly be used to measure core CT components and programming concepts and they can be categorized under the tool category of Programming and Development Environments. This was expected, given that da Cruz Alves et al. (2019) concentrated solely on code analysis.

Varghese and Renumol (2023) discuss what CT skills were assessed by researchers with videogames. Based on our interpretations on the results in Varghese and Renumol (2023) most of the assessed skills handled Core CT Components (Abstraction, Algorithmic Thinking, Decomposition, Pattern Recognition), Programming Concepts (Conditionals, Iteration, Modularity, Parallelism, Synchronization) or Problem-Solving Strategies (Debugging, Problem-solving, Efficiency). There were no assessed components that matched directly with Cognitive Processes, Collaborative and Communicative Skills, or Dispositions and Attitudes. To assess these CT components researchers in the included studies in Varghese and Renumol (2023) used Interviews, Think aloud protocols, self-reported feedback survey as assessment methods.

With Figure 4, we illustrate our interpretation how different assessment tools are linked with various CT components, interpreted to align with our categories, in Babazadeh and Negrini (2022). It is apparent that Scratch, Alice, Dr. Scratch, and CT tests are frequently used to evaluate programming concepts. Core CT components are predominantly assessed through Java tasks, Bebras tasks, Robotics tasks, and Scratch tasks. The CT self-efficacy scale is also employed to evaluate core CT components. Cognitive processes are not that widely assessed, but the assessment is made with the CT ability scale, Scratch, and the CT self-efficacy scale. There are not many tools used to assess students' dispositions and attitudes or their collaboration and communication, based on our interpretation of the data in Babazadeh and Negrini (2022). Many of the used tools can be categorized as Programming and Development Environments or Standardized Assessments and Testing Platforms.

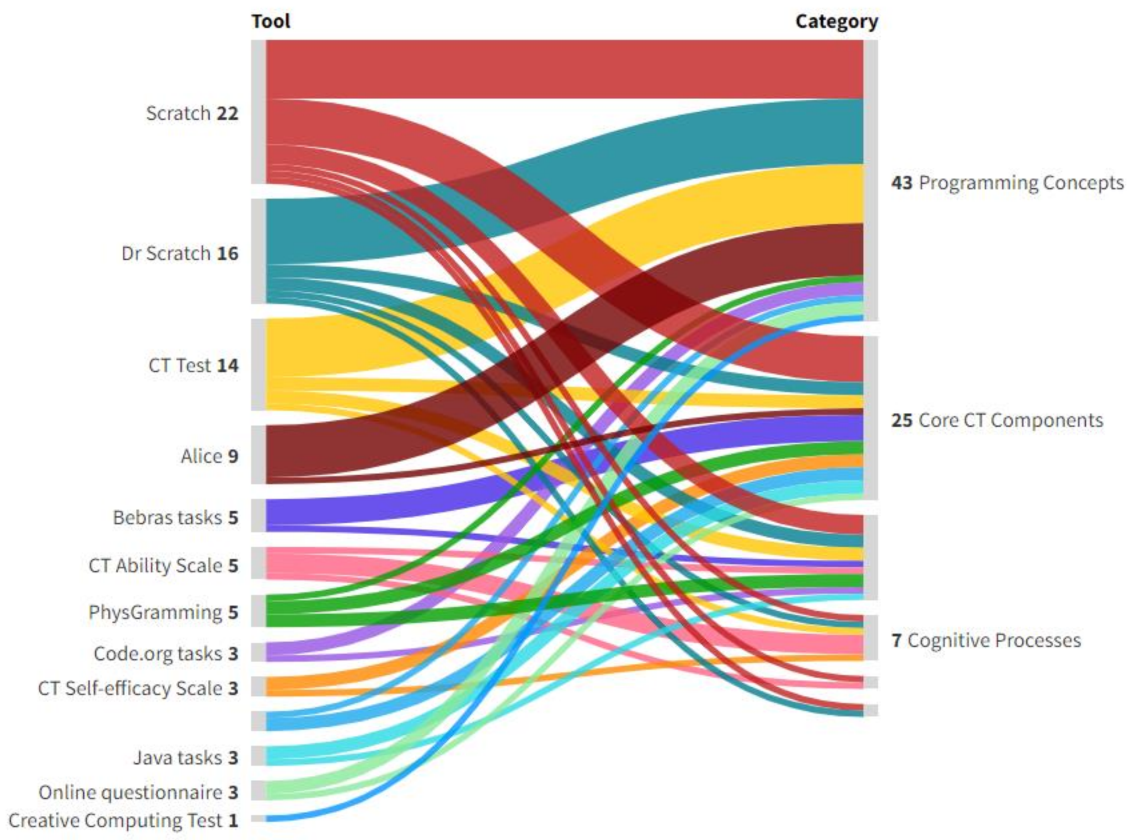


Figure 4. Our interpretation of the results in Babazadeh and Negrini (2022) connections between different assessment tools and assessed CT components

3.7 RQ6: Which pedagogical practices for the formative assessment of CT are identified?

The pedagogical practices in the formative assessment of Computational Thinking (CT) were not that widely discussed in the included systematic reviews. However, the discussions were still varied and reflective of the complexity inherent in CT itself.

Many of the studies mentioned the focus on *learning processes* instead of learning results in assessing CT. Pan et al. (2023) suggested that think-aloud protocols are effective in identifying students' learning processes, allowing teachers to diagnose and subsequently address gaps in CT comprehension and application. This method fosters an understanding of the individual student's thought process, providing insights into their problem-solving strategies. Similarly, Fagerlund et al. (2021) focuses on project evaluations that examine students' coding projects in Scratch, assessing not just the final product but also the developmental processes of CT skills.

Further the studies discussed the possibilities to give feedback for the students with the help of automated assessment tools. The use of immediate and *automated feedback* is exemplified by da Cruz Alves et al. (2019), who described how tools like Dr. Scratch can offer instant feedback on code quality and complexity. Tan et al.(2023) extends this concept with adaptive feedback informed by machine learning algorithms, delivering a personalized learning experience that evolves with the student's performance in CT tasks. Fagerlund et al. (2021) demonstrates the use of *scaffolding* through structured rubrics in Scratch projects, which guide students in developing their CT competencies. This strategic approach provides a clear pathway for students to follow, marking progressions in their skill development and understanding of CT principles.

Many of the assessment methods discussed in the studies are mainly used for research purposes. However, it seems that many of these methods can be used in school activities. One of these methods is a portfolio-driven approach that is discussed in Tang et al.(2020), which is a purposeful, systematic process of collecting and assessing different types of student products. The material can also include student observations and notes based on their work. Student observation was also mentioned in other studies (Fagerlund et al., 2020). However, many of these research methods were only mentioned briefly, with authors neglecting to include wider discussions about the pedagogical use of these methods in the classroom.

The studies also emphasized a *holistic approach* and *differentiated instruction* in CT assessment. The holistic approach, enhanced for instance by portfolios, assesses various CT components collectively through multiple methods (Fagerlund et al., 2020; Tang et al., 2020). While many studies suggested integrating these diverse methods to support differentiated instruction—tailoring assessments to meet students' unique learning needs (Liu et al., 2021)—they often stop short of discussing how to effectively combine these methods in practice, leaving a gap in the pedagogical application.

Finally, the included reviews mentioned approaches that can be referred to as *collaborative learning*. Tikva and Tambouris (2021) implied the use of collaborative learning strategies through the inclusion of game design and project-based learning, although the study does not detail specific formative assessment practices. Regarding collaborative learning, Fagerlund et al. (2020) also mentioned peer-to-peer assessment, referring to interactions between students during assessment processes. This was, however, only briefly mentioned and not discussed in other studies.

3.8 RQ7: What limitations regarding assessment methods, tools and pedagogical practices are reported?

The studies reported various challenges in assessment processes regarding pedagogy, tools used and methods, which are listed below.

Pedagogical Challenges:

- Complexity and Time Constraints: Tools designated for research are often impractical for classroom use due to their complexity and the extensive time required for implementation (Fagerlund et al., 2021).
- Insufficient Technological and Pedagogical Resources: Challenges such as lack of technological infrastructure, time for planning and material preparation, and limited instructional time (Tikva & Tambouris, 2021).
- Teachers' CT Content Knowledge: Uncertainties about appropriate CT content for different student age groups and the need for enhanced teacher knowledge and proficiency in teaching CT (Tikva & Tambouris, 2021).

Challenges with Tools:

- Narrow Assessment Tools: A focus on programming and coding, particularly with block-based languages like Scratch, while neglecting broader CT skills and other programming languages (Araujo et al., 2016; da Cruz Alves et al., 2019).
- Underexplored Data Analysis Techniques: The potential of data mining and machine learning is not fully harnessed in existing tools (Varghese & Renumol, 2023).

Challenges with Methods:

- Single-Method Limitations: The use of single-method approaches can fail to capture the full spectrum of CT, necessitating the development of multifaceted approaches (Muñoz et al., 2023).
- Variability and Lack of Standardization: Variability in how assessments are conducted, and a lack of standardized procedures challenge the methodological rigor and the consistency of assessments across different settings (Pan et al., 2023).
- Conceptual Consistency and Validation: A lack of consensus on the definition of CT leads to confusion in assessment constructs, and there are challenges in validating the reliability of assessment instruments (Haseski & Ilic, 2019; Tang et al., 2020).

4. Discussion

The landscape of assessing computational thinking (CT) in educational settings appears to be both rich and complex, revealing a 'jungle' that teachers must navigate. This review has undertaken an inductive approach to synthesize what is known in the field and identify the gaps that persist. Such an exploration is vital, as it not only highlights the diversity in assessment methods and tools but also underscores the challenges and opportunities facing teachers in implementing effective CT assessments.

The results of this review indicate a prevalent focus on programming and core CT components across various studies. These components are often assessed using direct methods such as artifact analysis and testing, employing tools from Programming and Development Environments or Standardized Assessments and Testing Platforms (e.g., Scratch, Dr. Scratch, Bebras tasks). However, a noticeable gap exists in assessing the broader spectrum of CT skills, particularly cognitive processes, dispositions, attitudes, and collaborative aspects. These are crucial elements of CT as defined by pioneers like Papert (1980) and Wing (2006), which transcend mere programming skills.

While much of the current research has concentrated on theoretical frameworks and methodologies for assessing CT (Araujo et al., 2016; Liu et al., 2021; Pan et al., 2023), there has been comparatively less emphasis on the practical implementation of these assessments in classroom settings. When CT assessment is explored within educational contexts, studies often highlight programming tasks using specific tools (Varghese & Renumol, 2023; Tan et al., 2023; da Cruz Alves et al., 2019) or focus on specific education levels (Fagerlund, 2021). Our findings underscore this trend and point to the need for a more holistic approach that not only includes a broader range of CT skills but also emphasizes their practical application in diverse classroom environments.

Interestingly, more indirect methods such as observation and interviews, typically utilized in research, offer insights into these less commonly assessed components. However, their application in classroom settings poses significant challenges due to their time-consuming nature and the specialized expertise required for conducting and interpreting such assessments. This discrepancy highlights a gap between research methodologies and practical pedagogical tools available for teachers.

4.1 Pedagogical Implementations and Limitations

This review suggests that while many tools and methods are designed for research purposes, their direct translation into pedagogical practice remains limited. The need for methodologies that are feasibly integrated into classroom activities is evident. Specifically, the use of peer assessment (Fagerlund et al., 2020) as a pedagogical method in formative assessment of CT requires further exploration. Such strategies could potentially address the gap in assessing soft skills like collaboration, creativity, and dispositions within CT education.

Furthermore, despite the recognition of CT as encompassing more than programming skills, the majority of assessment methods and tools remain focused on coding aspects. This imbalance points to a critical need for developing and validating tools that can effectively measure the full range of CT components. For teachers, this means exploring ways to operationalize research methods such as observations and interviews into their assessment practices, potentially through the development of observation sheets, interview guides, and other resources specifically designed for educational contexts.

4.2 Bridging Research and Practice

The transition from research to practice necessitates a deeper understanding and transformation of research methodologies into practical, pedagogical tools. There is a compelling need for resources that explicitly connect CT components with assessable elements, tailored for the teacher's use. Creating such resources, including detailed observation sheets and structured interview guides, could facilitate a more holistic assessment of CT skills in classroom settings. Moreover, integrating these tools with other assessment methods, such as self-assessment and peer-to-peer assessment, could enrich the formative assessment process, offering a more nuanced and comprehensive view of students' CT capabilities.

4.3 Implications for Practice, Future Studies, and Limitations

By synthesizing existing evidence and identifying practical assessment tools and methods, we have offered insights that can be useful in future research and development to create tools and practices for classroom use. Based on the results and methodology of this meta-synthesis, we can highlight some key implications for practice and future research.

Implications for Practice:

1. *Need for Practical Tools for Educators*: There is a need to develop user-friendly observation sheets and structured interview guides aligned with CT components identified in research, making them feasible for classroom use.
2. *Need for Integrated Assessment Methods*: It is necessary to use a combination of direct and indirect assessment methods, including self-assessment and peer-to-peer assessment, to capture a broader range of CT skills beyond programming tasks.
3. *Need for Professional Development for Teachers*: Professional development programs are needed to equip teachers with the skills and knowledge for effective CT assessment. These programs should focus on the practical application of research methodologies in the classroom, ensuring teachers are well-prepared to implement the tools and strategies identified in our review.

Need for Future Studies:

4. *Practical Classroom Applications*: There is a need to translate theoretical frameworks into practical classroom applications, developing and validating tools that can effectively measure the full range of CT components in real-world educational settings.
5. *Systematic Reviews in Different Subjects*: New systematic reviews are needed that focus on the detailed connections between assessment methods and their practical implementation. These reviews should be conducted in the context of different subjects to explore how CT can be integrated across various disciplines.

Limitations of the Study:

6. *Review of Reviews Methodology*: The wide scope of the review of reviews methodology may have overlooked specific details about the connections between methods and their operationalization. Future studies should focus on these detailed connections to provide more targeted insights.
7. *Evolving Nature of CT*: The field of CT is continuously evolving, and new methodologies and tools are constantly being developed. This tertiary review may not capture the latest primary studies in this field but is able to provide a broad overview of the current approaches.

5. Conclusion

The discussion in this tertiary review underscores a critical intersection between theoretical research and practical teaching needs in the assessment of computational thinking. While the field has advanced in identifying and employing a variety of assessment methods and tools, significant gaps remain in translating these into accessible and effective pedagogical practices. Addressing these gaps requires not only a re-evaluation of the tools and methods themselves but also a concerted effort to align them with pedagogical goals, ensuring that they are both meaningful and feasible for teachers to implement.

References

* Refers to a study included in the review corpus

* Araujo, A. L. S. O. de, Andrade, W. L., & Guerrero, D. D. S. (2016). A systematic mapping study on assessing computational thinking abilities. https://doi.org/10.1109/FIE.2016.7757419

* Babazadeh, M., & Negrini, L. (2022). How is computational thinking assessed in European K-12 education? A systematic review. *International Journal of Computer Science Education in Schools, 5*(4). https://doi.org/10.21585/ijcses.v5i4.138

Bahji, A., Acion, L., Laslett, A.-M., & Adinoff, B. (2023). Exclusion of the non-English-speaking world from the scientific literature: Recommendations for change for addiction journals and publishers. *Nordic Studies on Alcohol and Drugs*, *40*(1), 6–13. https://doi.org/10.1177/14550725221102227

Balanskat, A., & Engelhardt, K. (2015). *Computing our future: Computer programming and coding - priorities, school curricula and initiatives across Europe.* Brussel European Schoolnet.

Bond, M., Khosravi, H., De Laat, M., Bergdahl, N., Negrea, V., Oxley, E., Pham, P., Chong, S. W., & Siemens, G. (2024). A meta systematic review of artificial intelligence in higher education: A call for increased ethics collaboration and rigour. **International Journal of Educational Technology in Higher Education, 21**(4). https://doi.org/10.1186/s41239-023-00436-z

Booth, A., Sutton A., Clowes, M., Martyn-St James, M. (2022). *Systematic Approaches to a Successful Literature Review.* SAGE

Buntins, K., Bedenlier, S., Marín, V., Händel, M., & Bond, M. (2023). Methodological approaches to evidence synthesis in educational technology: A tertiary systematic mapping review. MedienPädagogik, 54, 167–191. https://doi.org/10. 21240/mpaed/54/2023.12.20.X

* da Cruz Alves, N., Gresse von Wangenheim, C., & Hauck, J. C. R. (2019). Approaches to Assess Computational Thinking Competences Based on Code Analysis in K-12 Education: A Systematic Mapping Study. *Informatics in Education, 18*(1), 17-39. https://doi.org/10.15388/infedu.2019.02

* Fagerlund, J., Häkkinen, P., Vesisenaho, M., & Viiri, J. (2021). Computational thinking in programming with Scratch in primary schools: A systematic review. *Computer Applications in Engineering Education, 29*(1), 12–28. https://doi.org/10.1002/cae.22255

González-Pérez, L. I. & Ramírez-Montoya, M, S. (2022). Components of Education 4.0 in 21st Century Skills Frameworks: Systematic Review. *Sustainability 14* (3). https://doi.org/10.3390/su14031493

* Haseski, H. İ., & Ilic, U. (2019). An Investigation of the Data Collection Instruments Developed to Measure Computational Thinking. *Informatics in Education, 18*(2), 297–319. https://doi.org/10.15388/infedu.2019.14

Higgins, S., Xiao, Z., & Katsipataki, M. (2012). The impact of digital technology on learning: A summary for the Education Endowment Foundation. Education Endowment Foundation. https://eric.ed.gov/?id=ED612174

Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J., & Linkman, S. (2009). Systematic literature reviews in software engineering—A systematic literature review. Information and Software Technology, 51(1), 7–15. https://doi.org/10.1016/j.infsof.2008.09.009

* Liu, R., Luo, F., & Israel, M. (2021). What Do We Know about Assessing Computational Thinking? A New Methodological Perspective from the Literature. In *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE 2021), June 26-July 1*, Virtual Event, Germany. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3430665.3456380

Lodi, M. (2020). Informatical Thinking. *Olympiads in Informatics, 14*, 113–132. https://doi.org/10.15388/ioi.2020.09

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

Mertala, P., Moens, E. & Teräs, M. (2022). Highly cited journal articles on educational technology: A descriptive and critical analysis. *Learning, Media and Technology*, 1-14. https://doi.org/10.1080/17439884.2022.2141253

* Muñoz, R. F. Z., Hurtado Alegría, J. A., & Robles, G. (2023). Assessment of Computational Thinking Skills: A Systematic Review of the Literature. *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje, 18*(4), 319-330. https://doi.org/10.1109/RITA.2023.3323762

Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hofmann, T. C., Mulrow, C. D., Shamseer, L., Tetzlaf, J. M., Akl, E. A., Brennan, S. E., Chou, R., Glanville, J., Grimshaw, J. M., Hróbjartsson, A., Lalu, M. M., Li, T., Loder, E. W., Mayo-Wilson, E., McDonald, S., & Moher, D. (2021). The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. BMJ (clinical Research Ed.), 372, n71. https://doi.org/10.1136/bmj.n71

* Pan, Z., Cui, Y., Leighton, J. P., & Cutumisu, M. (2023). Insights into computational thinking from think-aloud interviews: A systematic review. *Applied Cognitive Psychology, 37*(1), 71–95. https://doi.org/10.1002/acp.4029

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

Priem, J., Piwowar, H., & Orr, R. (2022). OpenAlex: A fully-open index of scholarly works, authors, venues, institutions, and concepts. ArXiv. https://arxiv.org/abs/2205.01833

Stern, C. & Kleijnen, J. (2020). Linguistic biases in systematic reviews: You only get out what you put in. *JBI Evidence Synthesis*, 18(9), 1818-1819. https://doi.org/10.11124/JBIES-20-00361

Sutton, A., Clowes, M., Preston, L., & Booth, A. (2019). Meeting the review family: Exploring review types and associated information retrieval requirements. Health Information and Libraries Journal, 36(3), 202–222. https://doi.org/10. 1111/hir.12276

* Tan, B., Jin, H.-Y., & Cutumisu, M. (2023). The applications of machine learning in computational thinking assessments: A scoping review. *Computer Science Education*. https://doi.org/10.1080/08993408.2023.2245687

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

Thomas, J., Graziosi, S., Brunton, J., Ghouze, Z., O'Driscoll, P., Bond, M., & Koryakina, A. (2023). EPPI Reviewer: Advanced software for systematic reviews, maps and evidence synthesis [Computer software]. EPPI Centre Software. UCL Social Research Institute. London. https://eppi.ioe.ac.uk/cms/Default.aspx?alias=eppi.ioe.ac.uk/cms/er4

* Tikva, C., & Tambouris, E. (2021). Mapping computational thinking through programming in K-12 education: A conceptual model based on a systematic literature review.

Computers & Education, 162, 104083. https://doi.org/10.1016/j.compedu.2020.104083

Varghese, V. V., & Renumol, V. G. (2023). Video games for assessing computational thinking: A systematic literature review. *Journal of Computers in Education*. https://doi.org/10.1007/s40692-023-00284-w

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

Ye, J., Lai, X., & Wong, G. K.-W. (2022). The transfer effects of computational thinking: A systematic review with meta-analysis and qualitative synthesis. *Journal of Computer Assisted Learning, 38*(6), 1620–1638. https://doi.org/10.1111/jcal.12723

Appendix 1: K-12 Computational thinking assessment reviews (n = 12)

Research Question 2: Which skills and components are most frequently assessed?

Author	Year	OA[1]	Type	# Studies	Years	Author Countries	Primary Focus	Core CT components				Programming Concepts							Cognitive Processes			Problem-solving				Collaborative & communicative skills		Dispositions & Attitudes
								Abstraction	Algorithmic thinking	Decomposition	Pattern recognition	Sequencing	Loops/Iteration	Conditionals	Variables & Data Representation	Modularity	Parallelism	Events & Synchronization	Logic & reasoning	Creativity & innovation	Critical thinking	Problem solving	Debugging	Testing	Efficiency	Collaboration & Cooperation	Communication	Interest & Engagement
Araujo et al.	2016	-	MR	27	2009 - 2016	Brazil	Identify, classify & review approaches & methodologies for assessing CT.	X	X	X		X	X	X	X	X	X		X			X	X					
Babazadeh & Negrini	2022	🔓	SR	26	2016 - 2020	Italy	Analyse how CT is assessed in European K-12 education, identifying tools used for assessment & CT dimensions evaluated.	X	X	X	X	X	X	X					X									
da Cruz Alves et al.	2019	🔓	MR	23	1997 – 2018	Brazil	Evaluate existing approaches for assessing CT through code analysis in K-12.					X		X	X	X				X				X				
Fagerlund et al.	2021	🔓	SR	30	2007 – 2019	Finland	Investigate the assessment of CT through Scratch in primary education.	X		X							X			X					X	X		
Haseki & Ilic	2019	🔓	SR	64	2010 - 2018	Turkey	Investigate the properties of paper-and-pencil data collection instruments to measure CT.	X	X	X	X	X					X					X				X		
Liu et al.	2021	-	IR	28	2010 - ?	USA	Analyse research methodologies for assessing CT in primary education.																	X				

[1] Cropped from the Open Access Logo image at https://creativecommons.org/about/program-areas/open-access/open-access-logo/

Note: SR = Systematic Review, MR = Mapping Review, MS = Meta-synthesis, IR = Integrative review, ScR = Scoping review, ML = Machine learning

Appendix 1: K-12 Computational thinking assessment reviews (n = 12)

Research Question 2: Which skills and components are most frequently assessed?

Author	Year	OA[2]	Type	# Studies	Years	Author Countries	Primary Focus	Core CT components				Programming Concepts							Cognitive Processes			Problem-solving				Collaborative & communicative skills		Dispositions & Attitudes
								Abstraction	Algorthmic thinking	Decomposition	Pattern recognition	Sequencing	Loops/Iteration	Conditionals	Variables & Data Representation	Modularity	Parallelism	Events & Synchronization	Logic & reasoning	Creativity & innovation	Critical thinking	Problem solving	Debugging	Testing	Efficiency	Collaboration & Cooperation	Communication	Interest & Engagement
Muñoz et al.	2023	-	SR	65	2012 – 2022	Colombia, Spain	Identify methods of assessing CT in K-12 & CT skills.	X				X		X	X		X	X	X									
Pan et al.	2023	-	SR	35	2011 – 2021	Canada	Examine the use of think-aloud interviews in CT & the methodology for understanding cognitive processes.															X						
Tan et al.	2023	-	ScR	20	2014 - 2021	Canada	Examine scope of ML approaches to assess CT (educational context, data, algorithms, aspects of CT assessed).																					X
Tang et al.	2020	-	SR	96	2011 – 2019	USA	Review how CT has been assessed (tools, educational context, subjects).	X	X	X	X	X	X	X	X		X		X									
Tikva & Tambouris	2021	-	SR	101	2006 – 2019	Greece	Develop a conceptual model based on K-12 CT programming studies.	X	X	X		X	X	X	X	X	X	X	X	X	X	X	X		X	X	X	
Varghese & Renumol	2023	🔓	SR	11	2010 – 2021	India	Examine the effectiveness of using video games for assessing CT skills.		X		X		X			X		X				X	X	X				

[2] Cropped from the Open Access Logo image at https://creativecommons.org/about/program-areas/open-access/open-access-logo/

Note: SR = Systematic Review, MR = Mapping Review, MS = Meta-synthesis, IR = Integrative review, ScR = Scoping review, ML = Machine learning

Appendix 1: K-12 Computational thinking assessment reviews (n = 12)

Research Question 4: What methods and tools are used for assessing various CT skills/components?

Author	Year	OA[3]	Type	# Studies	Years	Author Countries	Primary Focus	Direct assessment methods				Indirect assessment methods				Innovative & emerging methods			
								Self-/performance evaluation	Artifact analysis	Testing	Code analysis	Interviews & questionnaires	Observation	Think-aloud protocols	Log data & error analysis	Eye-tracking	Gamified assessments	Evidence-centred design	Data mining & machine learning
Araujo et al.	2016	-	MR	27	2009 - 2016	Brazil	Identify, classify & review approaches & methodologies for assessing CT.			X		X							
Babazadeh & Negrini	2022	🔓	SR	26	2016 - 2020	Italy	Analyse how CT is assessed in European K-12 education, identifying tools used for assessment & CT dimensions evaluated.			X		X	X						
da Cruz Alves et al.	2019	🔓	MR	23	1997 – 2018	Brazil	Evaluate existing approaches for assessing CT through code analysis in K-12.				X								
Fagerlund et al.	2021	🔓	SR	30	2007 – 2019	Finland	Investigate the assessment of CT through Scratch in primary education.	X	X			X	X	X	X				
Haseki & Ilic	2019	🔓	SR	64	2010 - 2018	Turkey	Investigate the properties of paper-and-pencil data collection instruments to measure CT.												
Liu et al.	2021	-	IR	28	2010 - ?	USA	Analyse research methodologies for assessing CT in primary education.		X	X		X		X		X			

[3] Cropped from the Open Access Logo image at https://creativecommons.org/about/program-areas/open-access/open-access-logo/

Note: SR = Systematic Review, MR = Mapping Review, MS = Meta-synthesis, IR = Integrative review, ScR = Scoping review, ML = Machine learning

Appendix 1: K-12 Computational thinking assessment reviews (n = 12)

Research Question 4: What methods and tools are used for assessing various CT skills/components?

								Direct assessment methods				Indirect assessment methods				Innovative & emerging methods			
Author	Year	OA[4]	Type	# Studies	Years	Author Countries	Primary Focus	Self-/performance evaluation	Artifact analysis	Testing	Code analysis	Interviews & questionnaires	Observation	Think-aloud protocols	Log data & error analysis	Eye-tracking	Gamified assessments	Evidence-centred design	Data mining & machine learning
Muñoz et al.	2023	-	SR	65	2012 – 2022	Colombia, Spain	Identify methods of assessing CT in K-12 & CT skills.				X	X						X	
Pan et al.	2023	-	SR	35	2011 – 2021	Canada	Examine the use of think-aloud interviews in CT & the methodology for understanding cognitive processes.		X			X	X	X	X				
Tan et al.	2023	-	ScR	20	2014 - 2021	Canada	Examine scope of ML approaches to assess CT (educational context, data, algorithms, aspects of CT assessed).			X		X					X		
Tang et al.	2020	-	SR	96	2011 – 2019	USA	Review how CT has been assessed (tools, educational context, subjects).			X		X							
Tikva & Tambouris	2021	-	SR	101	2006 – 2019	Greece	Develop a conceptual model based on K-12 CT programming studies.		X			X	X					X	X
Varghese & Renumol	2023	🔓	SR	11	2010 – 2021	India	Examine the effectiveness of using video games for assessing CT skills.			X									X

[4] Cropped from the Open Access Logo image at https://creativecommons.org/about/program-areas/open-access/open-access-logo/

Note: SR = Systematic Review, MR = Mapping Review, MS = Meta-synthesis, IR = Integrative review, ScR = Scoping review, ML = Machine learning

International Journal of Computer Science Education in Schools, November 2024, Vol. 6, No. 4
ISSN 2513-8359

Evaluating the Primary Trainee Teachers' Knowledge of Computational Thinking Concepts Using a Card Sorting Activity

Yasemin Allsop[1]

Filiz Kalelioglu[2]

Melike Aslan Unlu [3]

[1]University College London, UK

[2]Baskent University, Turkiye

[3]University College London, UK

DOI: 10.21585/ijcses.v6i4.215

Abstract

This study investigated the effectiveness of the 'Match it' card sorting activity for evaluating the student teachers' knowledge and understanding of computational thinking (CT) concepts. One hundred forty-six primary student teachers were asked to sort 26 scenarios and words alongside nine images under five main computational concepts: algorithmic thinking, abstraction, decomposition, patterns & generalisation, and evaluation. The study found that the card sorting activity, as a method for assessment, was useful. However, the issues around the design and the content of the current card sorting activity were reported by students, which suggests that further revisions should be made to improve the effectiveness of the tool.

Keywords: Computational thinking, assessment, learning, card sorting activity, teacher training, primary teachers, student teachers.

1. **Introduction**

The recent inclusion of computer science concepts in the curricula of many countries, including England, has placed computational thinking at the centre of computing education

(Selby & Woollard, 2014). The primary national curriculum programme of study for computing in England describes the aims of computing education as “to equip pupils to use computational thinking and creativity to understand and change the world” (DfE, 2013, p. 188); however, the programme of study doesn’t provide any tools or guidance on how the CT concepts should be taught or assessed. Where examples of planning and teaching strategies have been widely developed and shared through online and offline resources, assessing students’ learning of CT skills is still a hazy area (Grover, Cooper, & Pea, 2014). Assessment and feedback are important elements of the learning process (Black & William, 1998; Hattie & Timperley, 2007), especially for identifying the student’s strengths and gaps in their learning to support learning (William & Thompson, 2008). As highlighted by Grover and Pea (2013), “without attention to assessment, CT can have little hope of making its way successfully into any K-12 curriculum”, and consequently, “measures that would enable educators to assess what the child has learned need to be validated” (p. 41).

Many reasons pose challenges for assessing the students’ learning of CT skills. The lack of an agreed definition for Computational Thinking and its characteristics (Gonzalez et al., 2017; Shute, Sun, & Asbell-Clarke, 2017) makes it difficult for educators to develop a standardised assessment tool. In many cases, researchers developed their own CT measures, such as questionnaires and surveys, to assess the knowledge of CT skills (Denner, Werner, Campe, & Ortiz, 2014; Yadav et al., 2014; Kim, Kim & Kim, 2013). Some studies focused on designing tests for assessing CT (Mühling, Ruf, & Hubwieser, 2015; Meerbaum-Salant, Armoni, & Ben-Ari, 2013), some developed formative tools focusing on feedback to support learners improving their CT skills (Moreno-León & Robles, 2015), while some suggested a multiple evaluation model for assessing children’s learning of CT skills from different facets (Grover & Pea, 2013; Grover, 2017). There were no studies found about the use of card sorting activity for evaluating both student and in-service teachers’ understanding of CT concepts. Therefore, this study would provide an example of using card sorting techniques in this context, which further studies can build upon.

2. Literature review

This section will provide information about what computational thinking is and different approaches for assessing and evaluating CT concepts.

2.1 What is Computational Thinking?

The term ‘Computational Thinking’ was coined by Papert (1980) in his book Mindstorms, where he discussed the benefits of teaching procedural thinking in the LOGO programming environment. Wing (2010) described CT as “the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent” (p.1). Many studies define CT, focusing on specific aspects. Selby and Woollard (2014) highlighted CT as a cognitive process, and Cuny, Snyder, and Wing (2010) described it as a problem-solving approach. Some highlighted the role of metacognition in the CT process (Brennan & Resnick, 2012; Kafai & Burke, 2015) and a few discussed CT by focusing on the automation of information when computers execute repetitive tasks efficiently (Aho, 2012; Lu & Fletcher, 2009). The current study defines CT as a set of concepts and skills that can be used for formulating solutions to problems that can be automated. The skills aspect includes specific programming concepts, which will be discussed in section 3.4.

2.2 Teachers’ and student teachers’ knowledge of CT

CT concepts and approaches have strong links to other skills and disciplines, including problem-solving and creativity (Yadav et al., 2014) therefore, teaching CT from early on will help students to familiarise themselves with the CT skills and apply these skills to solve more complex and abstract problems in different contexts.

There are a few studies focused on teaching teachers and student teachers about CT concepts and approaches. Yadav et al. (2011) conducted a study for implementing and evaluating their computational thinking module, which focused on teaching CT concepts to student teachers. They found that after completing the training module, the students' knowledge and understanding of the computational thinking process was improved. In terms of assessing teachers' knowledge of CT, Yadav et al. (2018) used vignette-based assessment prompts to analyse teachers' responses before and after completing a training course focusing on integrating CT into primary maths and science classrooms. They found that text-based vignette assessment allowed them to gain a better understanding than closed-ended assessment and make sense of teachers' conceptions of CT as these would allow teachers to reflect on their experience and understanding of CT in primary mathematics and science contexts.

Haines, Krach, Pustaka, and Richman (2019) conducted a research study where they focused on teaching CT concepts to STEM teachers as part of a professional development course. They found that teachers couldn't integrate the optimisation and generalisation concepts into their teaching plans. After examining the online discussion board, they concluded that the lack of teachers' experiences and their comfort level with different computational thinking tasks were the main reasons for this.

Rich, Mason and O'Leary (2021) conducted a study where 127 primary school teachers took part in a year-long professional development where they were gradually introduced to coding, CT concepts and practises using Technological Pedagogical Content Knowledge (TPACK) framework (Koehler & Mishra, 2009). They used the TBaCCT instrument to examine teachers' beliefs about computational efficacy, coding efficacy, and teaching efficacy before the first session and after the last one. They mentioned instructors using formative assessment to evaluate teachers' knowledge and perceptions. This shows the importance of using multiple tools for assessing learners' views and conceptualisation of CT, which will be discussed in detail in the next section.

2.3 Assessing Computational Thinking Skills

In recent years, several studies have been conducted to measure the CT skills that learners develop in schools (Brennan & Resnick, 2012; Grover, 2015; Werner et al., 2012). Brennan and Resnick (2012) proposed a framework with three dimensions for assessing CT skills in the Scratch environment: computational concepts, practises and perspectives. They listed sequences, loops, events, parallelism, conditionals, and data as the programming constructs that represent computational concepts. Grover (2015) also designed an assessment system which included both summative and formative tools to measure CT skills. She used quizzes, multiple-choice questions and a rubric to evaluate learners' knowledge of programming concepts.

Weintrop et al. (2016) developed a series of interactive online assessments to measure students' CT skills in mathematics and science classrooms. Their approach focused on measuring students' behaviour and thought processes rather than assessing factual knowledge. Lui and colleagues (2018) also highlighted the importance of focusing on the process of making. They used portfolio assessment to evaluate the students' learning. Although this type of assessment can be adapted to any learning context, it can be time-consuming and not necessarily offer a direct assessment of specific CT concepts.

As discussed above, there are many challenges to assessing CT skills using one method therefore, multiple means of assessment approach should be adopted to evaluate learners' knowledge of CT skills. With this conclusion in mind, the authors developed a simple card-sorting activity for evaluating the learner's knowledge and understanding of CT from the computational concepts' aspect. The next section will discuss the use of card-sorting activities as a technique for evaluating learning.

2.4 Card sorting activity as a technique for evaluating learning

According to Cooke (1994), card sorting is a technique for identifying the knowledge structures of participants. In the card sorting activity, participants are given a set of cards with a concept written on them. The participants sort out the cards into categories based on the semantic relations (Spencer, 2009). Fincher and Tenenberg (2005) pointed out that the way participants categorise things externally reflects their mental representation (internal) of these concepts. Several studies used card sorting activities as a tool for assessing, evaluating, and analysing participants' learning in different contexts.

Friedrichsen and Dana (2017) designed and used a card-sorting activity with prospective and practising teachers at the elementary, middle, and secondary levels to help them clarify what they believed about teaching and learning science. They concluded that card sorting activity was a "useful tool for helping student teachers begin to articulate their knowledge and beliefs about their own purposes and goals for teaching science" (2017, p.300).

Eli, Mohr-Schroeder and Lee (2011) investigated the ability of prospective middle-grade teachers to make mathematical connections while engaging in card-sorting activities. They designed twenty cards which had various mathematical terms, concepts, definitions, and problems on them. The participants, twenty-eight prospective middle-grade teachers, were asked to complete a repeated single-criterion open card sort and closed card sort. They found that card sorting activities could be used as both formative and summative assessment tools for mathematical connection-making that could be implemented into planning and teaching.

Hennissen, Beckers and Moerkerke (2017) used a card-sorting activity with 136 first-year student teachers to evaluate the effectiveness of the curriculum by analysing the cognitive schemas that they were able to develop. The participants were asked to rank 30 concepts printed on cards into between two and ten logical groups within fifteen minutes. The card-sorting technique was successful in analysing the development of cognitive schemas.

The review of the literature illustrated that the use of card-sorting activities for evaluating learners' knowledge of CT concepts is limited. Dorn and Guzidal (2010) used a sorting activity including 26 cards to investigate web developers' knowledge of introductory computing concepts. They used repeated single-criterion card sorting with both open and closed sorts. Although they reported that the card sorting activity was useful for evaluating web developers' understanding of computing concepts, it did not provide information about how they learn. Therefore, they decided to use qualitative data to explore this further.

3. Methodology

For this pilot study, a mixed research approach was adopted where both quantitative and qualitative methods were used to evaluate the effectiveness of the ‘Match it’ card sorting activity for evaluating CT skills from the aspect of computational perspectives. Adopting a mixed method approach enabled the authors to use data collection techniques that are available from both quantitative and qualitative approaches to address the research questions in a best-fit approach (Creswell, 2003) rather than being limited to either qualitative or quantitative approaches. The quantitative dimension of the research included single-group pre-project and post-project models as a quasi-experimental design. The independent variable of the research is programming activities for teaching computational thinking skills and programming to computer science teachers, whereas dependent variables include computational thinking skills. Semi-structured interview as the qualitative method was used to gain insight into participants' perspectives of the ‘Match it!’ card sorting activity as an evaluation tool and their ideas about how CT skills should be assessed.

3.1 Participants

One hundred forty-six student teachers with an age range of 20 to 50 years took part in this study. One hundred twenty-six of the participants were female, and 20 were male.

The participants were based in a university in London, studying a one-year Primary PGCE (Post Graduate Certificate in Education) course that awards them qualified teacher status when they complete the course. Nineteen students were interviewed at the end of the study.

While 62 students stated that they had previously studied Information and Communication Technology (ICT) as part of their secondary education, 84 students reported that they did not. Table 1 displays the subject areas that the students studied for their undergraduate degrees.

Table 1: The subjects participants studied for their undergraduate degree

Department	#
Psychology	18
History	8
Education Studies	5
English Literature	5
Politics	5
English	4
Law	4
Childhood Studies	4
Early Childhood Studies	3
Biology	3
Economics	3

American Studies	2
Classical Studies	2
Drama	2
Education	2
French	2
History and Politics	2
English and American Literature	2

3.2 Ethics

We created an information sheet and a consent form in line with BERA's (2018) ethical guidelines for participants. To ensure anonymity, no names were revealed during data collection, analysis and reporting processes. Instead, we used pseudonyms such as 'Student 1' and 'Student 2'. Ethical approval has been received from …(The institution name has been removed for the peer-review process) Research Ethics Committee.

3.3 Data Collection Techniques

The following data collection tools were used for investigating the effectiveness of card sorting activity as a tool for assessing students' learning of computational concepts.

'Match it!' card sorting activity

'Match it!', a card sorting activity, was developed by the researchers, which requires students to match the computational concepts with the related scenarios or images that represent each concept. Five computational concepts were included, and seven to eight scenarios for each dimension were presented to students in written text or image form. The list of concepts, scenarios and images can be seen in Appendices 1 and 2.

Semi-structured interviews

Nineteen students were interviewed at the end of the project individually. The interviews were recorded using a sound recorder and transcribed. Each interview was around 15-20 minutes long. Through the interviews, the students had an opportunity to reflect on their learning processes and perspectives. This also provided researchers with an opportunity to clarify any unanswered questions about how the students used the 'Match it!' card sorting activity to check their understanding of computational concepts and other areas that are relevant to the wider focus of this research.

3.4 Developing 'Match It!' card sorting activity

The 'Match it!' card sorting activity was developed to aid primary school teachers in evaluating their own CT skills from the computational concepts dimension. The sorting activity consists of scenarios and images that represent five specific computational concepts: algorithmic thinking, abstraction, decomposition, patterns & generalisation, and evaluation. Although more scenarios were included during the design process, this was reduced to eliminate the repetition and ensure that it would not require a very long time to complete, as this can disengage some students. Appendix 1 shows the list of scenarios, images, and relevant computational concepts. The students were asked to match the images and scenarios to the relevant CT concepts. Although the students were allowed to complete an online version of the sorting activity using computers, they were also given a physical copy of the cards to practise whilst working on the online version. Altogether, there were 35 items under five categories. The students received 1 point for each time they sorted the items under the correct categories. In total, they could have received 35 points. Table 2 shows the point system for the card sorting task.

Table 2: The Point system for the ‘Match it!” Card sorting activity

	Words	**Scenarios**	**Images**	**Total**
Algorithmic Thinking	2	4	2	8
Abstraction	2	3	2	7
Decomposition	2	4	2	8
Pattern & Generalisation	1	3	2	6
Evaluation	3	2	1	6
Total	10	16	9	35

The card sorting activity was selected as a method because it could be used as a self or peer assessment tool and provides learners with an opportunity to work on a practical task for authentic learning. William and Thompson (2008) suggested that irrespective of its purposes and methods, “classroom assessment must first be designed to support learning” (p.63). Sorting images and scenarios allows learners to continue learning through monitoring and evaluating their own or friends' understanding of computational concepts.

The tool that has been shared in Appendix 1 was finalised after it had been checked by a group of four students. This was useful for clarifying the vocabulary to ensure that they were understandable by the learners. For some items, examples were included to help students make sense of the scenarios. The sorting activity was designed to be completed individually, in partners or collaboratively in groups, however, for this study, the students were asked to complete it on their own as this would make it easier to evaluate the individual students’ progression in knowledge of computational concepts.

3. 5 Information about teaching sessions

The participants were taught four face-to-face sessions, each lasting two hours. The training programme, although not fully integrated, was designed with the Technological Pedagogical Content Knowledge (TPACK) framework (Koehler & Mishra, 2009) in mind. The students were given opportunities to develop their subject knowledge alongside teaching strategies and technical skills. Table 3 shows the session information in a link to the TPACK framework.

The first session focused on introducing the computing curriculum and then brief information about CT concepts through discussions and unplugged activities. In the second session, the participants were encouraged to have discussions about Computational thinking and its concepts in relevant studies. They were then taught about the Logo language and asked to explore Bee-bots and Pro-bots in groups. In this session, they were briefly shown how to create simple animations in Scratch coding environment and were given online links to the resources that they could use for developing their knowledge of Scratch. In session three, the focus was constructivist game-making. The participants were shown more complex constructs for creating games using the Scratch platform and were asked to discuss the strategies they would use for assessing Scratch games after reading relevant studies. They were then asked to start designing their Scratch games as this was their professional learning task for computing, which they needed to complete in their own time.

Table 3: Session analysis using Technological, pedagogical, and content knowledge (Adapted from Rich, Mason and O'Leary, 2021).

		Technological Knowledge	**Pedagogical Knowledge**	**Content Knowledge**
1	**Introduction to Computing curriculum and CT concepts**	None as the focus was on unplugged activities.	● Collaborative learning ● Cross curricular ● Differentiation ● Planning ● Questioning ● Scaffolding ● Unplugged	● Algorithms ● Computing Curriculum ● Computational Thinking ● Conditionals ● Debugging ● Events ● Loops ● Parallelism ● Sequences
2	**Introduction to Scratch and Logo**	● Logo environment ● Scratch: ● Adding sprites ● Adding sound ● Adding background ● Block types ● Creating sprites ● Creating backgrounds ● Bee-Bot floor robot ● Pro-bot floor robot	● Collaborative learning ● Constructivism ● Constructionism ● Experiential learning ● Metacognition ● Modelling ● Predicting codes ● Questioning ● Testing ● Behaviour management	● Abstraction ● Debugging ● Game mechanics ● Physical computing ● Repetition ● Variables
3	**Creating a**	Scratch:	● Assessment	● Programming

	Scratch game	● Using Operators ● Creating Variables ● My Blocks tab	● Collaborative learning ● Giving feedback ● Modelling ● Planning ● Predicting codes ● Testing	constructs ● Storyboarding ● Abstraction ● Custom codes
4	**Making a LED postcard**	Electrical circuits LEDs	● Assessing project work ● Collaborative learning ● Creating / Making ● Planning /designing ● Testing ● Problem solving ● Behaviour management	● Boolean logic ● Circuit designs ● Materials ● STEM ● STEAM

The final session focused on Boolean logic through puzzles. The participants created an LED postcard and discussed the process they have been through in link to STEM education. A shared forum was placed on the learning management system Moodle for participants to share and discuss anything related to their computing-related activities and tasks, including sharing a link to their finished games. Short videos were created to explain CT concepts and shared on Moodle for participants to refer to whenever they needed.

3.6 Linking the card game to teaching sessions

The “Match it!” card sorting activity was developed to align with the teaching of five core computational thinking (CT) concepts: algorithmic thinking, abstraction, decomposition, patterns & generalisation, and evaluation. These concepts were the focal points of both the teaching sessions and the scenarios and images used in the card-sorting activity. The idea behind using these cards was to create a formative assessment tool that reflected the key concepts from the lessons taught and helped students assess their knowledge of these computational concepts in a structured way.

Each lesson in the program was designed to introduce and explore these CT concepts through a wide range of activities. For example, in the second session, students engaged with tools like Scratch and Logo, platforms known for fostering algorithmic thinking and problem decomposition (Weintrop et al., 2016). These activities allowed students to break down larger programming tasks into smaller, manageable parts (decomposition) and write algorithms (algorithmic thinking) to solve specific problems.

In parallel, the card sorting activity included scenarios and images that mirrored these learning objectives. For instance, the card for algorithmic thinking represented a sequence of instructions for making a toast, while decomposition cards showed travel packs and grocery lists (Appendix 1 of the article).

3.7 Data Analysis

The analysis of the data was performed in two stages. In the first stage, the students' pre-project–post-project scores from the 'Match it!' card sorting activity were evaluated by using a t-test to determine whether there was a significant difference between the mean scores of data. The single sample was tested by using SPSS software with a level of significance of .05. As Pagano (2010) stated, that t-test for a single sample is appropriate when the experimental study has only one sample; the sampling distribution was normal, and the number of the participants were greater than 30.

In the second stage, the data from semi-structured interviews were analysed to check the students' understanding of computational thinking and the effectiveness of the card-sorting activity as a result. Focusing on the specific research question for this study was useful for analysing the data in a more structured way and making connections between categories and themes that emerged from data and questions in relation to relevant literature.

4. Findings

4.1 The analysis of pre-and post-project scores from 'Match it!' card sorting activity

Pre-project and post-project achievement scores were analysed to examine the students' knowledge of computational concepts prior to this study and evaluate their progress at the end of the project. When the mean scores were examined, the pre-project scores of the participants were 16.58, and the post-project scores were 16.86. This illustrates that there is a slight increase in the post-project scores of the participants. The overview of the mean scores from 'Match it!' card sorting activities is presented in Table 4.

Table 4: Pre-project and post-project achievement scores

Project	**N**	**Mean**	**SD**	**SEM**
Pre-Project	146	16.58	4.27	.35
Post-Project	146	16.86	4.74	.39

Additionally, a t-test was applied to see whether the difference between the pre-project and post-project was significant or not. The results of the t-test are presented in Table 5.

Table 5: T-test Results of the achievement scores

	Mean	**SD**	**MSE**	**t**	**SD**	**p**
Pre and Post Project Results	-.27	6.26	.52	-.528	145	.598

As can be seen in Table 6, there is no significant difference between the pre-project and post-project scores in terms of students' knowledge and understanding of computational concepts.

We then analysed the students’ performance in five computational concepts to make sense of their progress in each theme. The results of this are shown in table 5. The students' pre-project scores in Algorithmic Thinking were 4.01, while post-project scores in Algorithmic Thinking were 4.64. The students' pre-project performances in abstraction and post-project performances remained the same. When looking at the students’ progress in knowledge of decomposition, the pre-project scores of the students were 3.08, while the post-project scores were 3.05. For pattern, the pre-project scores of the students were 3.84, and the post-project scores were 3.16. Finally, the analysis of the scores for students’ knowledge of evaluation showed a decrease in their knowledge at the end of the project: 3.69 and 3.40, respectively.

Table 6: Students Scores of Sub-dimensions of Computational Thinking Skill

		Mean	**N**	**Std. Deviation**	**Std. Error Mean**
Algorithmic Thinking	Pre-project	4.01	146	1.72	.14268
	Post-project	4.64	146	1.95	.16144
Abstraction	Pre-project	1.96	146	1.20	.09811
	Post-project	1.96	146	1.25	.10327
Decomposition	Pre-project	3.08	146	1.41	.11562
	Post-project	3.05	146	1.44	.11935
Patterns & Generalisation	Pre-project	3.84	146	1.28	.10589
	Post-project	3.16	146	1.19	.09818
Evaluation	Pre-project	3.69	146	1.49	.12393
	Post-project	3.40	146	1.51	.12503

The data from Table 5 illustrates that the students' knowledge of algorithmic thinking increased more than other computational concepts that were included in this study. The scores for abstraction remained the same, and for decomposition, patterns & generalisation and evaluation slightly decreased. There might be many reasons for this. For example, the students may not have received input focusing explicitly on other concepts, or they may not have enough time to practise and apply these concepts in different contexts. It can also be suggested that algorithmic thinking can be learned more easily than other concepts, as it is very relevant to students' daily lives.

4.2 The data from semi-structured interviews

For this study, 19 participants were interviewed, 14 of them were female, and five of them were male. The students were asked to reflect on their learning of CT and their experience of using the 'Match it!' card sorting activity to assess their understanding of computational concepts. The questions listed below were asked with more probing questions to clarify the points made by the participants.

- What do you understand when I say computational thinking? Can you explain it?
- Which activities at the university helped you to learn about computational thinking skills? Any examples?
- What are your views on the 'Match it!' card sorting activity as an assessment tool?
- Can you share ideas about how to assess and evaluate children's learning of CT skills?

The interviews were audio recorded and analysed using thematic analysis to search across the interview scripts to identify, analyse and report repeated patterns (Braun and Clarke, 2006). After familiarising with the data, the authors generated initial codes; then, these codes were used for searching themes within the data. The themes were reviewed, defined, and named before producing the final data analysis report. Table 7 shows the list of initial codes and the themes that were defined from the data.

Table 7: Thematic analysis of the interview data

Initial codes	**Themes**	**Reviewed themes**
Definition Vocabulary Concepts Skills Real-life examples	Defining CT Listing CT Concepts Assessing CT	Knowledge of CT
Learning theories Activities Ideas Reflections Feedback	Theories for integrating CT What went well? What could be improved?	Session reflections
Card sorting activity: Positive aspects Challenges Suggestions	Evaluating Card sorting activity Benefits	Benefits and limitations of Card sorting activity

4.2.1 Knowledge of CT

Most of the students were able to provide a simple definition of computational thinking. However, only a few of them could refer to the relevant literature that was discussed during teaching sessions. They mainly explained CT as a problem-solving approach, and many mentioned the word ‘automation’ but necessarily in the correct context. All the participants reported that the computing activities that they attended helped them to make sense of the terminology related to CT and understand how CT concepts can be taught through hands-on activities. 15 participants out of 19 indicated that they were unfamiliar with CT concepts as they were not taught about these when they were in primary and secondary education. There were many comments about how completing a Scratch game as a task was very beneficial for checking their understanding, as they had to make sure that they included specific programming constructs in their Scratch games for their professional learning tasks.

The students shared many ideas about how CT skills should be assessed and evaluated at the primary level. Many students expressed that using questioning as a formative assessment tool would be very useful for assessing children's learning of CT skills. Some of their comments are listed below:

> *"I mean talking to them, I suppose, finding out. From them. What, they have learned or so."*
>
> *"I think when those like scratch and stuff more like questioning of like getting them to add to it explain what they're doing and why they're doing it, so maybe to get them to explain the steps that they need to take in order to get that outcome because a lot of that I think for a lot of children is also a kind of like it is a game."*
>
> *"How would I check, questioning, surely? Questioning. How would I check that move up, just sort of just questioning and doing things that maybe sorting it out, things on the table to match them up or something?"*

Some students mentioned the use of open-ended tasks for assessing CT skills that allow children to use their creativity. They mentioned including a deliberate error in a task where the students had to debug it and then ask them to talk about how they solved the problem. Student 1 expressed this as:

> *'Setting themselves up a task and saying how not necessarily how quickly, but how well they complete it. I think it really brings out the creativity in children, and it's helping them to think with a different part of their mind, which is different to most lessons.'*

Student 5 reported as:

> *'So maybe even put in a deliberate mistake in the code and then see if they can fix it. I think deliberate mistake modelling is a fabulous method.'*

Observing children whilst working on their games was also mentioned as a method for assessing CT skills, but the students were worried about how time-consuming this could be. Student 15 expressed this as:

> *'Maybe they could debug something. You need to observe them. It is time-consuming and difficult.'*

Student 18 explained this as:

> *'I think through observation and seeing how being with friends when they tried to solve something and the way they did and the questions they ask. That type of thing helped you understand what they understand so far. When they come across a barrier, and then you see them solve or try to get in different types of ways, through observation might be the correct way. It might be more time consuming than just looking back.'*

4.2.2 Session reflections

Many students had a positive view of the sessions and emphasised how these made computing look less scary. They suggested that this first session, where the Computing Curriculum was discussed, and computational thinking concepts were briefly introduced via unplugged activities, was necessary and made them feel excited about teaching computing. Some of them mentioned having a very different experience with computing in school as a student, and they thought this first session gave them an overview of what the curriculum looked like and what computational concepts they should focus on as future teachers to plan and teach children. A few participants mentioned activities helping them not only with developing their subject knowledge but also pedagogical content knowledge. One participant expressed this as:

> *"It was interactive and very engaging. I learnt how to code with scratch. I also learnt interesting activities to share with the children in my class. It definitely has influenced my pedagogy".*

Another participant mentioned:

> *"I think the best part for me was to learn about computing pedagogy. You can really find out about subject knowledge on the Internet etc, but pedagogy is not very straightforward, especially when teaching coding".*

Hands-on activities, having fun, interactive and learning collaboratively were the main elements that were mentioned by participants in sessions two and four reflections, where they participated in theoretical discussions about CT concepts and also explored the floor robots (Bee-bots and Pro-bots) through hands-on activities. This highlights the value of adopting a constructivist approach when designing teaching activities. One participant reported this:

> *"Really enjoyed this session - one of my favourites of the year. Really great to be able to be interactive and have a go at using floor robots. I would like to use this in SE3 if there is the opportunity".*

Another participant explained:

> *"This was a great practical session! I really enjoyed having the practical application of the bee-bots and practising so we can feel what it would be like for children in the classroom. It was great for applying our knowledge and thinking about pedagogy. Kind of social constructivism approach really".*

In session two, the participants briefly used the Scratch coding application to create a simple animation. One participant shared the following comment:

> *"This was a fun session and made computing and coding feel more approachable, especially ideas for working with younger children".*

Many participants noted that even though it was modelled well, they found using Scratch hard and that they would benefit from written instructions that they could take with them to practice it later. Reflecting on the final session, the participants emphasised the cross-curricular element of the activities, where they solved problems using Boolean logic and created an LED postcard. There were remarks about 'learning by doing' and STEM activities that can be used for teaching concepts from different subjects in an integrated way. One participant commented:

> *"It was a hands-on and engaging session. This gave me some nice ideas for ways that STEAM can be done in the classroom with some fun resources".*

Another participant reported:

> *"My favourite session. Puzzles were good challenges. Creating the card showed how circuits and switches work. Learned by doing. Easier to understand the scientific concepts when you have a go yourself, made errors that improved thinking skills & troubleshooting, using trial and error. Can see how KS2 would love it, the more able children in KS1 too and others with modelling/pair work".*

Some students mentioned the usefulness of having face-to-face sessions and video clips, which provided information about the main computing concepts and modelling of the Scratch coding environment. This shows that blended learning was valued by the students. They also discussed how using different strategies and tools helped them to engage with the activities. One student expressed this as:

> *"I really liked that program, you know, Padlet, the one you just write together with your friends. Erm, I guess I could use it in KS 2 class, right? I am an active learner, so I learn better when I work with others".*

This shows that blended learning is not only about using technology but also implementing different tools and strategies when facilitating the sessions. The students expressed that they learn better when they actively work with others. By including opportunities for participants to work collaboratively, the tutor was able to accommodate the participant's learning needs.

4.2.3 Benefits and limitations of the card sorting activity

Many students highlighted the 'fun' element of completing the 'Match it!' card sorting activity if it was played in a group rather than alone. Student 3 reported this as:

> *'I can see how it would be fun if it was played like a game in groups.'*

Student 5 also shared a similar point:

> *'It would be fun to do with a partner I guess.'*

Student 6 commented:

> *"It was fun, but it could be like a game, and we could play with friends.'*

A few students explained that they found the physical paper version of the activity very useful for sorting and learning in the process, as it felt like playing a game rather than completing a standard classroom assessment. They also mentioned that the activity could be used for peer assessment or as a tool for learning in groups. Student 13 reported this as:

> *'I thought using the cards was a lot better, easier to see and move around. It was like playing a card game. I think it could be used for peer assessment or maybe even a learning tool in groups. You know, you could sort it out with friends, talk about it, etc.'*

Student 14 explained:

> *'It is great for learning as well, like you could play and have discussions with your friends. Then you could check to see if you got it.'*

Student 13 expressed this as:

> *'I guess you could use it in many ways. Like learning through play, in groups or with a partner. Then assessing each other?'*

One interesting point was made by a few students who expressed that they found the tool very difficult; they thought that some images and scenarios would fit under many scenarios.

Student 1 reported this as:

> *'The images seem to be fitting under many headings. Like this one (A script written using Scratch code blocks) could be algorithms or decomposition.'*

Student 3 explained as:

> *'Images were a bit confusing, like some of them could mean two things, right?'*

Student 15 shared:

'Do you think we needed that many options? Because it took time to complete. I guess it is fine for playing as a game but for assessment purposes, maybe better to have fewer options.'

Some students mentioned that the activity had too many items to sort and suggested that it should be shortened. Student 4 reported this as:

'I thought it was very long. I got a bit stressed as I felt like I should know these. Should have less things than we have more time to think.'

Student 9 made an interesting point by suggesting that including too many items and too much text can disengage some learners.

'I would shorten and include less scenarios and text. Some kids don't like reading.'

Many students expressed their disappointment of not receiving immediate feedback and suggested that an integrated scoring system or a paper-based score sheet that would show them the areas they need to work on would be very beneficial. Student 8 expressed this as:

'Is there a sheet to show the correct answers? That would be useful and less stressful.'

Student 12 reported:

'I couldn't see if my answers were correct, is there a scoring system? That would be nice to have, then you know what you need to work on.'

5. Discussion and conclusion

The 'Match it! Card sorting activity was developed to assess student teachers' understanding of five computational concepts: algorithmic thinking, abstraction, decomposition, patterns & generalisation, and evaluation. The data analysis of participants' pre- and post-project scores from the pilot study showed that there was a slight increase in the post-project scores of the participants; there was no significance between the pre-project and post-project scores in terms of students' knowledge of CT concepts.

There might be many reasons for this; CT concepts are very new for many students; therefore, they may not have had enough experience to learn and develop their understanding of these concepts outside of these limited sessions. In teaching sessions, these concepts were taught very briefly with a few examples, which didn't enable the students to deepen their understanding. The more explicit teaching of CT concepts through practical tasks could help students with their learning of these concepts. The observed increase in Algorithmic Thinking suggests that students may have grasped the foundational elements of this concept better than the others, possibly due to the specific examples and activities related to algorithmic processes that were integrated into the curriculum. For instance, the hands-on nature of algorithmic tasks, such as sequencing and step-by-step problem-solving, may have facilitated a clearer understanding and practical application of this concept compared to more abstract CT concepts. Conversely, the declines in scores for the other CT concepts indicate a potential misalignment between the teaching methods used and the cognitive demands of those concepts. The complexity and abstract nature of concepts like abstraction and pattern recognition may require more intensive and varied instructional strategies to support student learning. It is possible that the brief exposure to these concepts and the limited examples provided did not sufficiently enable participants to engage deeply with the material, resulting in reduced understanding and retention.

There were also issues around the design of the evaluation tool. The card sorting activity is designed to focus on recognising and categorising concepts rather than applying them in real-world problem-solving contexts. As noted in the study, the card sorting activity primarily measured whether students could match scenarios and images to specific CT concepts like algorithmic thinking or decomposition. However, computational thinking involves more than just recognising these concepts; it requires the ability to apply them to formulate solutions to problems, often in creative and dynamic ways (Wing, 2010; Grover & Pea, 2013).

One of the key issues raised by participants was related to some of the images and scenarios. Students reported that certain images could be categorised under multiple CT concepts, leading to confusion (Student 1, Student 3). For example, an image showing Scratch code could be categorised as both algorithmic thinking and decomposition, depending on how the participant interpreted it. This confusion suggests that the cards were not always clearly aligned with a single concept, which is crucial for accurate assessment (Shute, Sun, & Asbell-Clarke, 2017). The lack of clarity in the card content made it difficult to ensure that students were correctly assessed on their understanding of individual CT concepts, further weakening the reliability of the assessment tool.

The core of computational thinking is the ability to apply concepts such as algorithmic thinking, pattern recognition, and abstraction in problem-solving situations. According to Brennan and Resnick (2012), CT assessment frameworks should not only test students' understanding of concepts but also their ability to apply these concepts in practice. For instance, assessing algorithmic thinking requires students to develop a sequence of instructions to solve a problem, not just to recognize that a sequence exists. Similarly, evaluating decomposition skills involves breaking down complex problems into simpler parts, which goes beyond merely recognizing that a task involves decomposition. The card sorting activity, by focusing on categorization rather than application, did not measure these deeper skills.

Another significant limitation is the reliance on concept recognition rather than task-based assessment. Many CT assessment tools, such as portfolio-based evaluations or interactive problem-solving tasks, focus on students' ability to apply CT in a hands-on context (Lui et al., 2018; Grover, 2017). The card sorting activity, by contrast, offered no opportunity for students to engage in active problem-solving. As Shute et al. (2017) argue, the value of CT lies in its application, and assessment tools should be designed to measure this application, not just conceptual recognition.

Another problem raised by students was the lack of immediate feedback. The students suggested that including a score sheet or integrating a scoring system into the online version of the sorting activity would help them identify the concepts that they need to work on directly after they complete the activity. This also highlights the importance of immediate feedback for learning. Providing a rubric at the end or access to the electronic version of the tool with a built-in scoring mechanism could provide students with immediate feedback.

Despite these limitations, the interview data showed that the ‘Match it!’ card sorting activity enabled students to reflect on their understanding of computational concepts. This was supported by Fincher and Tenenberg (2005) as they discussed how placing concepts into categories can help people reflect their mental representation of these concepts. There were many occasions where the students described the card sorting activity as a learning tool. This shows that, as mentioned by Rugg and McGeorge (2005), the sorting activity could be used for both assessment and learning purposes in different contexts. In a group, it can be played as a game, which can provide students the opportunity to discuss and learn about concepts. In pairs, again, it could be used as a learning tool but also for peer assessment purposes.

As mentioned before, Liu and Chen (2013) reported that having a physical version of activity pieces allows players to communicate with their peers and learn in the process. The students did not have the opportunity to complete the activity in groups during this study. It would be interesting to include this aspect in future studies and investigate whether completing the game in groups would help the students with their anxieties related to being assessed.

In conclusion, the card sorting activity did not accurately measure CT skills because the images and text on the cards focused on the surface-level recognition of CT concepts rather than their application. The lack of clarity in some of the card scenarios, coupled with the absence of immediate feedback and practical problem-solving tasks, further undermined its effectiveness as an assessment tool for computational thinking. To improve the measurement of CT skills, future iterations of the activity should incorporate more applied tasks, clearer categorization of scenarios and images, and a mechanism for immediate feedback, aligning more closely with the cognitive processes involved in computational thinking (Wing, 2010). As Rao and Bhagat (2024) discussed, if a CT assessment is easy to implement and accurately reflects the learning outcomes of a particular curriculum, it has the potential to be highly effective. Aligning the card sorting task with specific objectives that have been taught would increase its effectiveness.

References

Aho, A. V. (2012). Computation and Computational thinking. *The Computer Journal, 55*(7), 832-835.

Black, P. & Wiliam, D., (1998). Assessment and classroom learning. *Assessment in Education: principles, policy & practice, 5*(1), 7-74.

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

Brennan, K., & Resnick, M. (2012). New Frameworks for Studying and Assessing the Development of Computational Thinking. In *Proceedings of the 2012 Annual Meeting of the American Educational Research Association*, Vancouver, Canada, 1-25.

Cooke, N. J. (1994). Varieties of knowledge elicitation techniques. *International journal of human-computer studies, 41*(6), 801-849.

Creswell, J. W. (2003). *Research design: Qualitative, quantitative, and mixed methods approaches* (2nd ed.). Sage Publications.

Cuny, J., Snyder, L., & Wing, J. M. (2010). Demystifying Computational Thinking for Non-Computer Scientists. [Unpublished Manuscript] Retrieved from http://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf Accessed January 21, 2021

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*, 277–296.

Department for Education. (2013). The National Curriculum in England, Framework Document. Retrieved from www.Education.Gov.Uk/Nationalcurriculum_ Accessed January 21, 2021

Dorn, B. and Guzdial, M., (2010, April). Learning on the job: characterizing the programming knowledge and learning strategies of web designers. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 703-712).

Eli, J.A., Mohr-Schroeder, M.J. and Lee, C.W., (2011). Exploring mathematical connections of prospective middle-grades teachers through card-sorting tasks. *Mathematics Education Research Journal, 23*(3), p.297.

Fincher, S. and Tenenberg, J., (2005). Making sense of card sorting data. *Expert Systems, 22*(3), 89-93.

Friedrichsen, P.M. and Dana, T.M., (2003). Using a card-sorting task to elicit and clarify science-teaching orientations. *Journal of Science Teacher Education*, *14*(4), 291-309.

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., Cooper, S., & Pea, R. (2014). Assessing computational learning in K-12. In *Proceedings of the 2014 conference on Innovation & technology in computer science education* (pp. 57–62). ACM, New York.

Grover, S., Pea, R., & Cooper, S. (2015). Systems of assessments" for deeper learning of computational thinking in K-12. In *Proceedings of the 2015 annual meeting of the American educational research association* (pp. 15-20).

Grover, S., (2017). Assessing algorithmic and computational thinking in K-12: Lessons from a middle school classroom. *In Emerging research, practice, and policy on computational thinking*, 269-288.

Haines, S., Krach, M., Pustaka, A., Li, Q., & Richman, L. (2019). The Effects of Computational Thinking Professional Development on STEM Teachers' Perceptions and Pedagogical Practices. *Athens Journal of Sciences, 6* (2), 97-122.

Hattie, J. & Timperley, H., (2007). The power of feedback. *Review of educational research, 77*(1), 81-112.

Hennissen, P., Beckers, H. and Moerkerke, G., (2017). Linking practice to theory in teacher education: A growth in cognitive structures. *Teaching and Teacher Education, 63*, 314-325.

Kafai, Y. B., & Burke, Q. (2015). Constructionist Gaming: Understanding the Benefits of Making Games for Learning. *Educational Psychologist, 50* (4), 313-334.

Kim, B., Kim, T., & Kim, J., (2013). Paper-and-pencil programming strategy toward computational thinking for non-majors: Design your solution. *Journal of Educational Computing Research 49*, 437–459.

Lu, J. J., & Fletcher, G. H. (2009). Thinking About Computational Thinking. *ACM SIGCSE Bulletin, 41*(1), 260-264.

Meerbaum-Salant, O., Armoni, M., & Ben-Ari, M. (2013). Learning computer science concepts with scratch. *Computer Science Education, 23*(3), 239–264.

Moreno-León, J., & Robles, G. (2015). Dr. Scratch: A web tool to automatically evaluate Scratch projects. In *Proceedings of the Workshop in Primary and Secondary Computing Education* (pp. 132–133).

Mühling, A., Ruf, A., & Hubwieser, P. (2015). Design and first results of a psychometric test for measuring basic programming abilities. In *Proceedings of the Workshop in Primary and Secondary Computing Education* (pp. 2–10).

Pagano, R. R. (2010). Understanding statistics in the behavioral sciences (9th ed.). Belmont, CA; Australia: Wadsworth Cengage Learning.

Papert, S. (1980). *Mindstorms: Children, Computers and Powerful Ideas.* NY: Basic Books.

Rao, T. S. S., & Bhagat, K. K. (2024). Computational thinking for the digital age: a systematic review of tools, pedagogical strategies, and assessment practices. *Educational technology research and development*, 1(32), 1893-1924.

Rich, P. J., Mason, S. L., & O'Leary, J. (2021). Measuring the effect of continuous professional development on elementary teachers’ self-efficacy to teach coding and computational thinking. *Computers & Education*, 168, 104196.

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.

Rugg, G. and McGeorge, P., (2005). The sorting techniques: a tutorial paper on card sorts, picture sorts and item sorts. Expert Systems, 22(3), pp.94-107.

Selby, C., & Woollard, J. (2014). 'Refining an Understanding of Computational Thinking.' Author's Original, 1-23. Retrieved from https://Eprints.Soton.Ac.Uk/372410/1/372410understdct.Pdf Accessed January 21, 2021

Shute, V.J., Sun, C. & Asbell-Clarke, J., (2017). Demystifying computational thinking. *Educational Research Review*, *22*, 42-158.

Spencer, D. (2009). *Card sorting: Designing usable categories*. Rosenfeld Media.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., et al., (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology 25* (1), 127–147.

Werner, L., Denner, J., Campe, S., & Kawamoto, D. C. (2012). The Fairy Performance Assessment: Measuring Computational Thinking in Middle School. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education, ACM*, pp. 215-220.

Werner, L., Denner, J., & Campe, S. (2014). Using computer game programming to teach computational thinking skills. *Learning, education and games*, 37.

Wiliam, D., & Thompson, M. (2008). Integrating assessment with learning: What will it take to make it work? Routledge.

Wing, J. (2010). Computational Thinking: What and Why? Retrieved from www.Cs.Cmu.Edu/~Compthink/Resources/Thelinkwing.Pdf

Yadav, A., Zhou, N., Mayfield, C., Hambrusch, S., & Korb, J. T. (2011, March). Introducing computational thinking in education courses. In *Proceedings of the 42nd ACM technical symposium on Computer science education* (pp. 465-470).

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., Korb, J. T., (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education (TOCE) 14* (1), 1–16.

Yadav, A., Krist, C., Good, J., & Caeli, E. N. (2018). Computational thinking in elementary classrooms: measuring teacher understanding of computational ideas for teaching science. *Computer Science Education*, 28(4), 371-400.

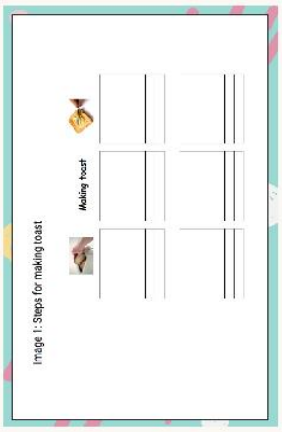
Appendices

Appendix 1: List of words, scenarios and images that are included in 'Match it!' Card sorting task.

	Word	**Scenario**	**Image**
Algorithmic thinking	Achieve a specific task Sequence of precise instructions	Re-telling the events from a story Instructional writing Designing a science experiment Creating rules for a game	Image 1: Steps for making a toast. Image 2: Sequences of instructions for controlling the on-screen robot to draw
Abstraction	Reduce complexity Filtering information	Solving word problems Identifying the main theme of a story Summarising the findings of an experiment	Image 3: Following a route on a map Image 4: Creating a model of a system e.g., Solar system or computer system.
Decomposition	Breaking down the problem	Creating a concept map Making a computer game	Image 5: Travel pack List Image 6: Grocery List

	Structuring information	Labelling parts e.g., Plants, body parts, and computers. Identify the instruments that used in a song	
Patterns & Generalisation	Common solutions	Recognising the common rules for spelling Using formulae in math problems Looking for patterns of shadows at different times of the day	Image 7: Times tables Image 8: Spot the difference
Evaluation	Making judgements Checking effectiveness and efficiency against the criteria Test and debug	Talking about how to improve their work e.g., game design, script, and story. Testing against defined criteria.	Image 9: Identifying what went wrong using the set criteria e.g., code errors

Appendix 2: Images that are included in the 'Match it!' card sorting task



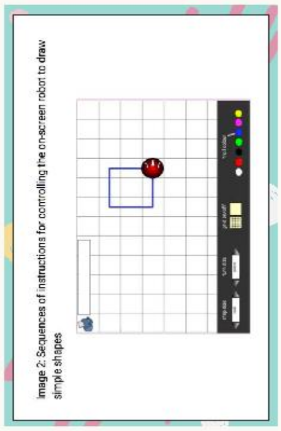
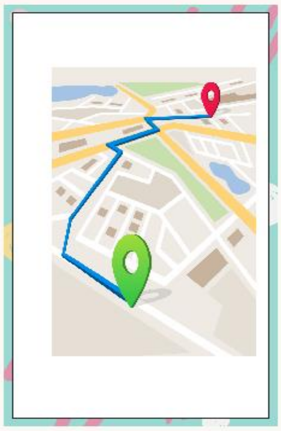
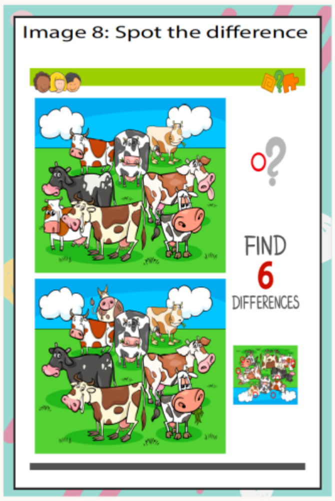


Image 2: Sequences of instructions for controlling the on-screen robot to draw simple shapes





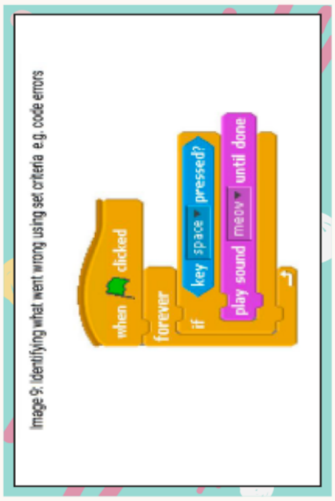


Image 9: Identifying what went wrong using set criteria e.g. code errors
when clicked
forever
if key space pressed?
play sound meow until done