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<br>• Cross curricular<br>• Differentiation<br>• Planning<br>• Questioning<br>• Scaffolding<br>• Unplugged | • Algorithms<br>• Computing Curriculum<br>• Computational Thinking<br>• Conditionals<br>• Debugging<br>• Events<br>• Loops<br>• Parallelism<br>• Sequences |
| 2 | **Introduction to Scratch and Logo** | • Logo environment<br>• Scratch:<br>  • Adding sprites<br>  • Adding sound<br>  • Adding background<br>  • Block types<br>  • Creating sprites<br>  • Creating backgrounds<br>• Bee-Bot floor robot<br>• Pro-bot floor robot | • Collaborative learning<br>• Constructivism<br>• Constructionism<br>• Experiential learning<br>• Metacognition<br>• Modelling<br>• Predicting codes<br>• Questioning<br>• Testing<br>• Behaviour management | • Abstraction<br>• Debugging<br>• Game mechanics<br>• Physical computing<br>• Repetition<br>• Variables |
| 3 | **Creating a Scratch game** | Scratch:<br>• Using Operators | • Assessment<br>• Collaborative learning | • Programming constructs |

| | | | | |
|---|---|---|---|---|
| | | • Creating Variables<br>• My Blocks tab | • Giving feedback<br>• Modelling<br>• Planning<br>• Predicting codes<br>• Testing | • Storyboarding<br>• Abstraction<br>• Custom codes |
| 4 | **Making a LED postcard** | Electrical circuits<br>LEDs | • Assessing project work<br>• Collaborative learning<br>• Creating / Making<br>• Planning /designing<br>• Testing<br>• Problem solving<br>• Behaviour management | • Boolean logic<br>• Circuit designs<br>• Materials<br>• STEM<br>• 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.

**Linking the card game to teaching sessions (Yasemin)**

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).

### *3.6 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<br>Vocabulary<br>Concepts<br>Skills<br>Real-life examples | Defining CT<br>Listing CT Concepts<br>Assessing CT | Knowledge of CT |
| Learning theories<br>Activities<br>Ideas<br>Reflections<br>Feedback | Theories for integrating CT<br>What went well?<br>What could be improved? | Session reflections |
| Card sorting activity:<br>Positive aspects<br>Challenges<br>Suggestions | Evaluating Card sorting activity<br>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.

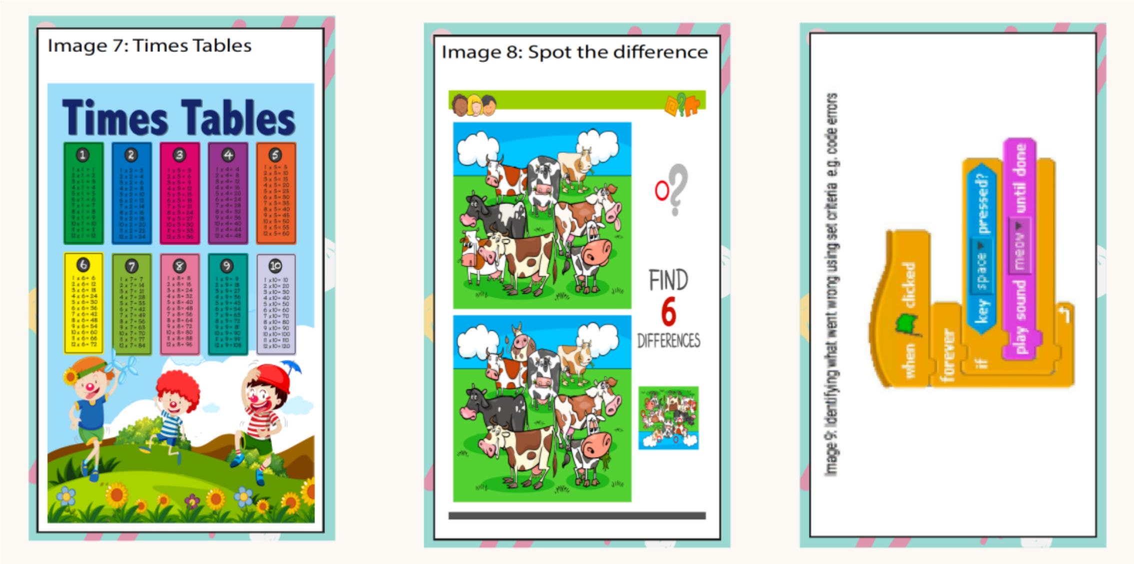
## Appendices

### *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<br><br>Sequence of precise instructions | Re-telling the events from a story<br>Instructional writing<br>Designing a science experiment<br>Creating rules for a game | Image 1: Steps for making a toast.<br>Image 2: Sequences of instructions for controlling the on-screen robot to draw |
| **Abstraction** | Reduce complexity<br><br>Filtering information | Solving word problems<br>Identifying the main theme of a story<br>Summarising the findings of an experiment | Image 3: Following a route on a map<br><br>Image 4: Creating a model of a system e.g., Solar system or computer system. |
| **Decomposition** | Breaking down the problem<br><br>Structuring information | Creating a concept map<br>Making a computer game<br>Labelling parts e.g., Plants, body parts, and computers.<br>Identify the instruments that used in a song | Image 5: Travel pack List<br>Image 6: Grocery List |
| **Patterns & Generalisation** | Common solutions | Recognising the common rules for spelling<br>Using formulae in math problems<br>Looking for patterns of shadows at different times of the day | Image 7: Times tables<br>Image 8: Spot the difference |
| **Evaluation** | Making judgements | Talking about how to improve their work e.g., game design, script, and story. | Image 9: Identifying what went wrong using the set criteria e.g., code errors |

| | | | |
|---|---|---|---|
| | Checking effectiveness and efficiency against the criteria<br>Test and debug | Testing against defined criteria. | |

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




## 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.