Journal of Pedagogical Research
Volume 7, Issue 2, 2023
https://doi.org/10.33902/JPR.202318528

Research Article

Integrating computational thinking into mathematics education through an unplugged computer science activity

Filiz Mumcu[1], Esra Kıdıman[2] and Fatih Özdinç[3]

[1]*Manisa Celal Bayar University, Türkiye (ORCID: 0000-0001-9371-6734)*
[2]*Republic of Türkiye Ministry of National Education, Türkiye (ORCID: 0000-0001-8529-5431)*
[3]*Afyon Kocatepe University, Türkiye (ORCID: 0000-0003-2150-5019)*

Integrating computational thinking (CT) into various disciplines via computer science (CS) methods such as unplugged, block-based, text-based, and physical programming is a trending topic in educational sciences. This study presents a perspective on implementing an unplugged computer science activity to integrate CT into mathematics education. This study aims to examine the integration of CT into mathematics education, its classroom practice, and the opinions of students and pre-service teachers towards the unplugged CS activity. For this purpose, we developed an unplugged CS activity, which includes CT components, computer science and mathematics education achievements. The unplugged CS activity is a problem of transmitting data most efficiently and accurately between transmitter and receiver. The researchers trained twelve pre-service mathematics teachers to implement this activity in real-classroom environments. The pre-service teachers implemented the unplugged CS activity with 80 students in four classes. CT components were considered in the activity design. The students performed data collection, data analysis, decomposition, pattern recognition, algorithm design, and testing and debugging, which are components of CT, during the activity. Middle school students stated that they had enjoyed the activity and that this activity made the mathematics learning process satisfying. They established a relationship between the activity and mathematics education, besides, they needed help connecting it with CS concepts. Pre-service teachers stated that they had challenges in classroom management during the activity. Classroom management should be a part of teacher education in designing and implementing CT-integrated lessons. Courses containing classroom management, CS activity design and implementation should be developed for pre-service teachers based on theory and practice within their discipline.

Keywords: Computational thinking; Unplugged computer science; Mathematics education; Teacher education; Pre-service teachers

Article History: Submitted 14 October 2022; Revised 12 March 2023; Published online 10 June 2023

1. Introduction

Global initiatives are attracting attention to develop digital competencies so that future generations can benefit effectively from computer science (CS) in solving real-world problems they face. As one of these skills, coding (or programming) in commonly used terms stands out. The UK's decision to make computer use compulsory in schools from September 2014 has led to curriculum

Address of Corresponding Author

Filiz Mumcu, PhD, Manisa Celal Bayar University, Faculty of Education, 45900 Demirci, Manisa, Türkiye.

✉ filizkuskaya@gmail.com

How to cite: Mumcu, F., Kıdıman, E., & Özdinç, F. (2023). Integrating computational thinking into mathematics education through an unplugged computer science activity. *Journal of Pedagogical Research, 7*(2), 72-92. https://doi.org/10.33902/JPR.202318528

reforms in other countries that focus on CS concepts, programming skills, and "computational thinking (CT)" (Bocconi, Chioccariello, & Earp, 2018). Various initiatives and policies attract attention to developing CT skills in schools in multiple countries, including the USA, United Kingdom, Lithuania, Finland, Korea, Japan, Singapore, and Turkiye (Seow, Looi, How, Wadhwa, & Wu, 2019). In the European Union's Science on Stage Europe network, the "European Code League" initiative encourages teachers and students to be creative and present innovative coding projects for STEM courses in primary and secondary schools (Science on Stage Europe, 2022a). Again, the European Union's "Coding in STEM Education" movement said, "*What does a robot have to do with environmental protection? How do you remotely control a model ship using a smartphone? How can you program a 'pet'? Information technology is everywhere, and coding offers many possibilities for exciting and relevant courses.*" It calls to popularise computer science education with an interdisciplinary approach (Science on Stage Europe, 2022b). In addition to these, when we examine the European Union reports, it seems that the teaching of digital competencies as a separate, particular subject, such as informatics or CS at lower levels of secondary education is common (European Commission / EACEA / Eurydice, 2019, pp. 28–32).

As education researchers, we focused on integrating computers as a technology into learning and teaching processes until a couple of years ago. However, in recent years we have started discussing integrating CS into education. Disseminating CS education from preschool to higher education and increasing its inclusiveness appears in two ways: (i) How should CS education be from preschool to higher education, (ii) How should CS education be integrated with other courses with an interdisciplinary approach? The first question aims to make CS education a compulsory part of schooling. The second question seeks integrating CS into other courses to improve students' CT skills (Huang & Looi, 2021). The subject of this research is the second question, which deals with mathematics education in particular.

Today, when CS education is mentioned, it is no longer about training field experts. The publication that triggered the adoption of this idea is Wing's (2006) study, which emphasizes the importance and accessibility of CS education for all. This study has increased the recognition of the concept of CT. There has been a linear increase in research articles every year on this topic since 2015 (Ozyurt & Ozyurt, 2023). However, the idea underlying the concept of CT is based on Seymour Papert. He also invented the visual programming language Logo with his book "Mindstorms: Children, Computers and Powerful Ideas" (Papert, 1980). Although early studies in the literature refer to the importance of CS education as a way to explore and think about other disciplines (Guzdial, 1994), it has not been embraced as much as Wing (2006) emphasizes the concept of CT.

Integrating CT into the education of students in different disciplines makes CS accessible to all students (Guzdial, 2008). Classroom practices that deal with CT with an interdisciplinary approach based on real-life problems impact learners' cognitive, social and emotional development (Tsortanidou, Daradoumis, & Barberá, 2021). For these reasons, integrating CT into curricula is advocated (Dickes et al., 2020; Irgens et al., 2020). Although there are efforts at the K12 level, we are still at the beginning of the road to achieving this in the classroom. Teachers' lack of understanding of CT, students' academic unpreparedness, and teachers' professional development (PD) needs are reasons for this (Kite & Park, 2023).

Therefore, studies examining how to integrate CT into learning and teaching processes in real-classroom environments will help create a roadmap for the beginning. CT can be integrated into courses using CS methods such as unplugged, block-based, text-based, and physical programming. This study emphasizes an activity developed to integrate CT into mathematics education through unplugged computer science, how this activity can be applied in the classroom, and how it can be transferred to the school experiences of pre-service teachers.

1.1. Computational Thinking (CT)

The commonly used definition of CT is: "*CT involves solving problems, designing systems, and understanding human behaviour, by drawing on the concepts fundamental to CS. CT includes a range of* *mental tools that reflect the breadth of the field of CS.*" (Wing, 2006, p.33). With the concept becoming popular, Wing (2011, p.1) later introduced a new definition of CT and emphasized that the idea is a thinking process. However, Lye and Koh (2014) argue that "CT" misses necessary research done before the term was popularized. This situation also prevents the formation of a consensus starting from defining the concept. Although there is no commonly accepted definition, the images of "problem analysis" and "problem-solving" are primarily used in the descriptions, followed by understanding computer, programming, and system design (Kalelioğlu et al., 2016; Taşlıbeyaz, Kurşun & Karaman, 2020). In their study, where Zhang and Nouri (2019) draw attention to this complexity, they claim that CT intersects with computing and problem-solving. It indicates that CS is seen as a way of embodying CT. Regarding CT, the first concepts that come to mind are coding and programming. Huang & Looi (2021) also state that programming or coding is used interchangeably, taught as a subset of CS, and is commonly associated with CT.

At this point, it would be helpful to emphasize a profound difference in meaning between programming and coding concepts. Cambridge Dictionary (2022) defines the concept of programming as (i) the process or skill of writing programs for computers, (ii) the instructions that tell a computer what to do; it defines the concept of coding as follows: a language used to program (= give instructions to) computers. Based on these definitions, it is understood that coding uses software languages, and programming is designing the algorithm to solve problems so that computers can operate. Therefore, CS provides a framework for educators to embody CT (Bocconi et al., 2016). In short, CT is more than learning a specific programming language, namely coding (Zha et al., 2020). However, it is a common cliché that CT should be taught with technology and programming because it is practical (Tsortanidou, Daradoumis, & Barberá, 2021). Such knowledge and skills are not yet part of teacher education programs. Nevertheless, unplugged CS activities can be less intimidating for teachers without a CS or programming background and avoid high costs such as teaching coding (e.g. time and PD) or managing hardware (e.g. acquiring resources, troubleshooting technical issues) (Huang & Looi, 2021). Therefore, as a starting point, this study focuses on using unplugged CS to integrate CS education into mathematics education.

1.2. CS Unplugged Activities for CT Integration

The concept of unplugged CS, which started as a collection of 20 activities published under the name "CS Unplugged: Off-line Activities and Games for All Ages," was put forward by its authors Bell, Witten, and Fellows (1998). The plan, initiated by Mike Fellows and colleagues in the 1980s, was closely concerned with mathematics and CS education trends. The goal was to develop materials based on modern research in CS and mathematics and to use these materials to make CS and mathematics education more exciting and interesting (Bell, Rosamond, & Casey, 2012). These ideas included many of the pillars of CS research, such as NP-completeness and automata. While the initial material was used in various projects and groups (such as "Family math" and "Mathmania"), it evolved and led to the "Mega-Mathematics" project and the internationally widely used "Unplugged CS" project. The initial purpose of CS Unplugged was to serve as a means of reaching out to young students and teaching them about CS, without requiring them to first acquire programming skills (Bell & Lodi, 2019).

Huang and Looi (2021, p. 84) express their understanding of CS, CT, and unplugged CS as follows:

- CS is an academic discipline encompassing principles such as algorithms, data structures, programming, computer architecture, and computer organization.
- CT is a broadly applicable set of problem-solving skills that includes abstraction, decomposition, pattern recognition, and algorithmic thinking.
- CS unplugged activities teach CS or CT without using computers (machines).

The unplugged CS approach emphasizes practical experiences to teach complex CS concepts and encourages a constructivist approach, where the goal is for students to discover interesting concepts worthy of study (Bell & Lodi, 2019). Unplugged CS is very effective in introducing programming ideas and algorithms before students put them into practice on a computer (Bell & Henderson, 2022). In this study, we use a CS unplugged activity as a means of engaging students with CT without using computers (based on Bell, 2021). The role of unplugged CS activities can be considered as a "preparatory" step to help students understand the algorithmic steps before writing code (Huang & Looi, 2021). Although it is mainly used at preschool and primary school levels, it is possible to prepare and implement unplugged CS activities that appeal to all age levels (for example, Özdinç et al. (2022) designed an unplugged CS activity for a PD course for STEM teachers in which they integrate CT into STEM activities). The essential advantages of unplugged CS activities are that learners cannot stay behind computers and communicate face-to-face with their teachers and peers. Moreover, computer access is not a problem; game elements can be included in the learning and teaching processes.

1.3. CT in Mathematics Education

Students should think computationally to solve real-life problems using CS concepts (Lye & Koh, 2014). The components of CT, abstraction, decomposition, pattern recognition, algorithm design, and debugging enable students to develop their solutions to real-life problems using CS possibilities. Therefore, the fields of science and mathematics education create a natural context for the integration of CT (Lee & Malyn-Smith, 2020; Waterman et al., 2020; Weintrop et al., 2016), and CT makes science and mathematics education more suitable with current and future professional practices (Weintrop et al., 2016).

Including CT skills in K-12 curricula presents challenges for mathematics educators while providing new perspectives on overcoming difficulties in mathematics teaching and familiar contexts for students. Feurzeig, Papert, and Lawler (2011, p. 490) describe the advantages of this relationship as follows:

1. The programming activity encourages an experimental approach to problem-solving.
2. Using a programming language provides students with a natural framework, a standard vocabulary, and personal experiences discussing mathematics.
3. Programming can give students precise information about key concepts, such as variables and functions.
4. Formal knowledge is only a part of what is tried to be conveyed to students in mathematics teaching. Without heuristic knowledge of problem-solving, this knowledge is of no value. Programming experience is precious for students in this sense.
5. The richness and extensibility of non-numerical examples open to programming can be used to help students develop essential and seemingly complex mathematical ideas. For example, using iterative procedures in programming to handle mathematical induction requires using the concept of recursion, a generalization of induction. The concept of recursion can be presented in simple programming examples that are easily accessible to children, even for those who are not yet ready to understand how to solve problems by induction on integers. For students prepared in this way, the principle of mathematical induction will seem like a small and natural extension of a very familiar concept.

There is a significant overlap between the practices gathered under CT and mathematical mind habits (Pei, Weintrop, & Wilensky, 2018). With the popularity of CT, the tendency towards studies that deal with the relationship between mathematics education and programming and examine it within the CT framework has increased in recent years. For instance, In Kong (2019)'s study examined the development of CT in primary school mathematics education. He asked students to design and program an algorithm to find the factors of a given number to learn composite and prime numbers. However, studies on CT in mathematics education primarily focus on teaching programming skills and rarely deal with concepts in mathematics content knowledge (Hickmott, Prieto-Rodriguez, & Holmes, 2018). There is a need for studies that will reveal the connections between mathematics education and CT, especially in classroom practices. This study aims to examine the integration of CT into mathematics education, its classroom practice, and the opinions of students and pre-service teachers towards an unplugged CS activity developed by the researchers. The research questions are:

RQ 1) How do the students develop their programming solutions according to the algorithms they design?

1a) What are the error rates that occur as a result of programming?

1b) What are the characteristics of the best algorithm?

1c) What is the amount of data used in programming?

RQ 2) What are the students' opinions on the unplugged CS activity?

2a) What are the difficulties and facilitators of the activity? What are the suggestions?

2b) What are the opinions on the relationship of the activity with mathematics education and CS education?

RQ 3)What are pre-service teachers' opinions on the unplugged CS activity?

3a) What are the difficulties and facilitators of the activity? What are the suggestions?

2. Methodology

2.1. Research Model

This study is modeled as a Type 1 product design and development study, one of the design development studies. Design development research examines specific components of the process and the impact of design efforts on a development process as a whole (Richey & Klein, 2014). Design and development research is an umbrella term for a wide range of studies that employ various traditional quantitative and qualitative research methods and strategies.

Most design and development studies employ multi-methods that combine qualitative and quantitative methods, with the qualitative approach taking precedence (Richey & Klein, 2007). Critical design and development processes are frequently described using case study methods in type 1 studies. Techniques such as interviews, observations, and document analysis can collect case study data and document the processes used and the conditions under which they are used (Richey & Klein, 2005).

Hevner et al. (2004) created a 6-phase model that included: a) identifying the problem motivating the research; b) describing the objectives; c) designing and developing the artefact; d) subjecting the artefact to testing; e) evaluating the results of testing; and f) communicate those results. We used this approach as the structure for this study.

This study designed and applied how CT, one of the current phenomena in the context of education, can be integrated into mathematics education in a real classroom environment. The results were examined regarding the activity, students, and pre-service teachers.

2.2. Context and Participants

The participants of this study were middle school students and pre-service mathematics teachers. 80 middle school students attending the 5th and 7th grades and 12 pre-service teachers participated in the research. This study was carried out in classrooms where pre-service teachers were assigned to teaching internships in the Spring semester of the 2021-2022 academic year. A convenient sample method was applied.

The authors developed the unplugged CS activity implemented in this study. Pre-service teachers applied the activity to middle school students in a real-classroom environment within the scope of the "Teaching Practice" course in which they were enrolled. This course enables pre-service teachers to experience the knowledge and skills they have acquired in their education by participating in educational activities at schools, thus improving their teaching knowledge and skills. Before applying the activity, pre-service teachers were trained on how to implement the activity. Pre-service teachers also took a 14-week "Algorithm and Programming" course in the Fall 2021-2022 academic year as part of CS education. Within the scope of this course, the pre-service teachers gained knowledge and skills about basic CS concepts, creating algorithms, unplugged CS, block-based programming, and STEM applications.

The pre-service teachers implemented the activity in small groups of four or five students. The implementation was carried out in four classes, with three pre-service teachers working in one class (see Table 1). Each activity lasted between 40-60 minutes. Before the implementation of the activity, pre-service teachers observed and taught mathematics in this class for 12 weeks.

Table 1
Research process

Team	*Number of pre-service teachers*	*Number of students*	*Grade*	*Number of activity groups*	*Duration*
1	3	19	5	4	60′
2	3	21	5	4	40′
3	3	19	7	4	40′
4	3	21	7	4	60′

2.3. Design of the Unplugged CS Activity

We adopted Bell and Lodi's (2019, p. 345) understanding to design the activity: *the unplugged CS approach should strike a balance between structured guidance and exploration, where the teacher is a facilitator who helps students construct their knowledge rather than an expert delivering knowledge*. The activity is to harness the power of unplugged CS to integrate CS education with mathematics education. In the design of the activity, attention was paid to the students' readiness. The activity aimed to raise students' awareness of sending and receiving data, collecting and analyzing data, creating a pattern, breaking the problem into small parts, designing an algorithm to solve a problem, and testing and debugging. The pilot implementation was conducted with ten 5th-grade students in the CS course. Field notes were taken, and observations were made during the pilot implementation. Then the activity was revised by the researchers. The final version of the activity is explained in detail in the "Implementation of the Unplugged CS Activity" section.

The researchers took decisions after the pilot implementation are as follows:

- The activity is suitable for the level of 5th-grade students and above.
- Forms consisting of 4x4 squares extend the implementation duration. Instead, 3x3 will suffice.
- The lottery method was used to form the groups. Choosing the task by lot has created excitement.
- If it is not possible to make colour printouts, a task can also be given from digital media.
- No difficulties were encountered regarding classroom management. There is no need for a warning for silence.
- The students struggled mentally during the activity and developed a solution by trying different ways.
- Student's ability to work in groups and express themselves can be observed during the activity.
- Students enjoyed the activity. They were willing to try again.
- Students thought that the activity was a mathematical study.
- Students were highly motivated by the activity. They conveyed their requests like "Can we do other activities like this?", "Let's try again next week".

The activity consists of encrypting the data according to a particular system, transmitting this encrypted data to the recipient in a secure environment, and deciphering the transmitted message (cryptanalysis). The activity can be applied to middle school students aged 11-14. Unplugged CS can be utilized through role-playing, manipulating real-world objects, and the physical actions of the body, among others (Aranda & Ferguson, 2018). The materials needed for the activity are:

- Forms consisting of 3x3 squares
- Coloured task sheets of 3x3 squares
- Pencil, yellow, blue, red, and green pencil
- A4 paper

2.4. Implementation of the Unplugged CS Activity

Bell and Lodi (2019) argues that an optimally guided approach, where the teacher has a path in mind but the students construct the knowledge for themselves, is the most effective way to implement the unplugged CS approach. For this purpose, an implementation guideline for the final version of the activity was prepared. The researchers held an online meeting with the pre-service teachers before the implementation. Pre-service teachers were first informed about the activity at this meeting, and the activity guideline was explained. Then, it was shown how the activity would be applied to the pre-service teachers from scratch. All the questions of the pre-service teachers regarding the activity were clarified. For example, a pre-service teacher asked, "*How will unpainted boxes be reflected in the scoring?*". All possible problems were discussed, and the issues were handled.

Students were divided into groups of four. Each group was given forms consisting of 3x3 coloured squares, blank A4 papers, and four coloured pencils (See Appendix 1). One of the students took on the receiver, while the other took on the transmitter's role. The group should develop an algorithm on how to send the data. Each group could only interact by making a clicking sound using a pencil, or something like that. For example, two clicks for blue, one-click for green. The following instruction was read to the students:

> "We have a table/field of three rows and three columns. You can think of every cell as a pixel. We have 12 pre-made coloured images. There will be four coloured pencils in groups, red, yellow, blue, and green. As a group, you must select a receiver among you. You will sit back to back with your friend who plays the receiver role. Then you will send the data on the task sheet to the recipient using only the pen click sound. Accordingly, the receiver will paint the 3x3 blank table given to her/him with the data she/he receives. So, you should develop a strategy as a group on how to send the data to the receiver. You have 10 minutes for this task. It would be best if you wrote your strategy for communication on a piece of paper. Try to understand the strategies of other groups during their task implementation. Each pen clicks you make to transmit data during the task will be counted as a +1 point. When finished, the similarity between the visual drawn by the distant group member and the task sheet given to the group will be checked. Each different cell between the task sheet and the paper painted by the receiver will be added to your score of +5 points. The group that completes the task with the lowest score will be deemed to have developed the best strategy."

Finally, if the students had questions about the activity, they were answered, and 10 minutes were started for the working time of the groups. Then, the student who assumed the role of the receiver and the other students sat with their backs to each other. A randomly selected task sheet was given to the group. Other groups counted and recorded the number of pens' click sounds. Students waited until the task was completed. After all, groups had completed the task and they were asked to explain their communication strategies to the other groups. Images of the implementation of the activity are given in Figure 1.

2.5. Data Collection and Analysis

The data regarding the activity implementation were collected in three different forms: *the pre-service teachers' reflective reports, the pre-service teachers' group activity form,* and *the students' group activity form* (see Appendix 2).

The thematic analysis framework of Braun & Clarke (2006) was used to analyse qualitative data: familiarizing with data, generating initial codes, developing initial themes, reviewing themes, defining and naming themes, and producing results. The second author created the initial codes and themes after reading the data from scratch. Then, the first and second authors reviewed and named the themes together. At the last stage, the third author was asked to repeat the thematic analysis by placing the codings on the themes created. Coding reliability was checked among the authors. In presenting the findings, the qualitative data collected from pre-service teachers and students were evaluated together and presented, thus ensuring context integrity.

Figure 1
Images from the implementation of one of the activities

In addition, the following data was collected during the implementation of the unplugged CS activity:

- Number of clicks,
- Correctly coded cells and the number of errors, and
- The solutions obtained by the groups as a result of the activity.

The quantitative data were used to calculate the points of the groups and analyzed by descriptive statistics.

3. Findings

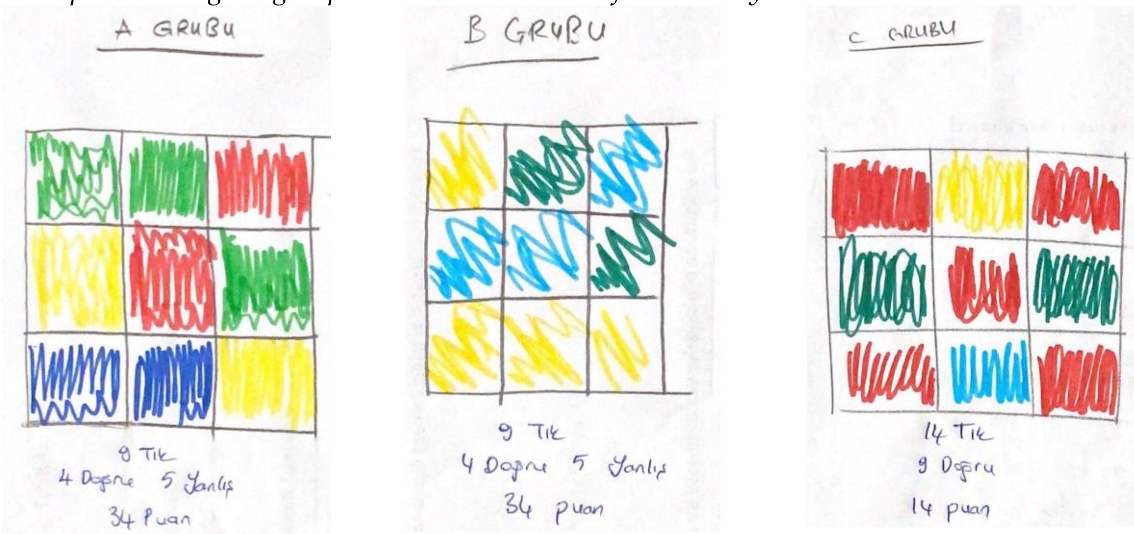
The findings were given as follows: the programming solutions developed by the students and the opinions of the students and pre-service teachers about the activity, respectively.

3.1. Groups' Solutions

First, the students decided how to digitise data so the transmitter could send the data (*data collection*). Then they determined how the receiver would receive the data (*data analysis*). After that, they had to select how to move between cells and how long they would wait while sending the data, speed, etc. (*decomposition*). They also identified patterns representing each colour's signal (*pattern recognition*). They later brought all these together and revealed their algorithmic designs (*algorithm design*). After the students implemented the activity as a group, they noticed and debugged their mistakes, so they discussed the opportunities to improve their algorithms (*testing & debugging*). The following formula was used to calculate the score of each group: Point = Number of clicks + (Number of errors x 5). Accordingly, the group with the lowest score is the most successful. Examples showing the groups' solutions as a result of the activity implementation are given in Figure 2.

Programming solutions of groups related to the same activity that four different groups of pre-service teachers did with 16 student groups; the accuracy rates were examined based on algorithm designs and the amount of data used.

Figure 2
Examples showing the group' solutions as a result of the activity



Nine clicks
Four correct and five wrong clicks
34 points

Nine clicks
Four correct and five wrong clicks
34 points

14 clicks
Nine correct and zero wrong clicks
14 points

3.1.1. Accuracy rate

The findings regarding the click and accuracy of the programming solutions developed by the groups are given in Table 2. Four of the 16 teams participating in the event could transmit the message without error. The number of clicks of the teams without errors was at least 14 and at most 29. Risky methods (such as trying to convey all the colours with a single click to reduce the number of clicks or determining colour and waiting for it without making a sound) increased the error rates. Students also tried to develop additional solutions to make up for the shortcomings of these risky methods. One of the pre-service teachers conveyed his observation as follows: "Some students in the group tried to solve the loudness with one click using different pencils, but they made many mistakes". In addition, the students noticed the mistakes that might occur while watching the solution of other groups and developed their solutions. As an illustration of this situation, a pre-service teacher stated her observation:

> "The first groups could not adjust the time between clicks because they clicked quickly. There was a communication breakdown between the transmitter and the receiver. That caused them to discolour or leave it blank. Other groups have concluded that there is enough time and that they should not make the mistake of their friends".

Table 2 shows that for the unplugged CS activity, students can work in groups, produce more than one algorithmic solution, develop different algorithm designs for each group, and test and debug their algorithms. 5th grade used at least 9, a maximum of 27 clicks, 7th grade used a minimum of 4, and a maximum of 29 clicks. The mean of correctly coded cells in the 5th grade is 4.88, and in the 7th grade is 6.12. The error mean of the 5th grade is 4.12, and the 7th grade is 2.88. Regarding the point averages, 36.13 is for the 5th grade and 33.38 for the 7th grade. While the mean of clicks of the 7th grade (mean=19.00) is higher than the 5th grade (mean=15.50), 7th graders completed the task more successfully than the 5th grade in terms of points and correctly coded cells. Table 2 makes it possible to see that there are different solutions to the same problem and that the groups' performances are different. If each group had come up with the same solution and algorithmic design, that would show researchers to revise their activity design.

Table 2
Click and accuracy results of programming solutions developed by the groups

Group Name	*Number of students*	*Grade*	*Number of Clicks*	*Correctly Coded Cells*	*Number of Errors*	*Points*
Group C	5	5	14	9	0	14
Stars	4	7	15	9	0	15
Skull	5	7	4	6	3	19
Rubik's Cube	5	7	21	9	0	21
Anonymous 2	5	7	12	7	2	22
Mind Cubes	5	5	21	8	1	26
Anonymous	5	7	29	9	0	29
Group D	6	5	11	5	4	31
Group A	5	5	9	4	5	34
Group B	4	5	9	4	5	34
Winners	5	5	27	7	2	37
Finishing	5	7	13	3	6	43
Those Who Shine	5	5	16	1	8	56
Smart People	5	5	17	1	8	57
Stars 2	6	7	29	3	6	59
Flying Seals	5	7	29	3	6	59

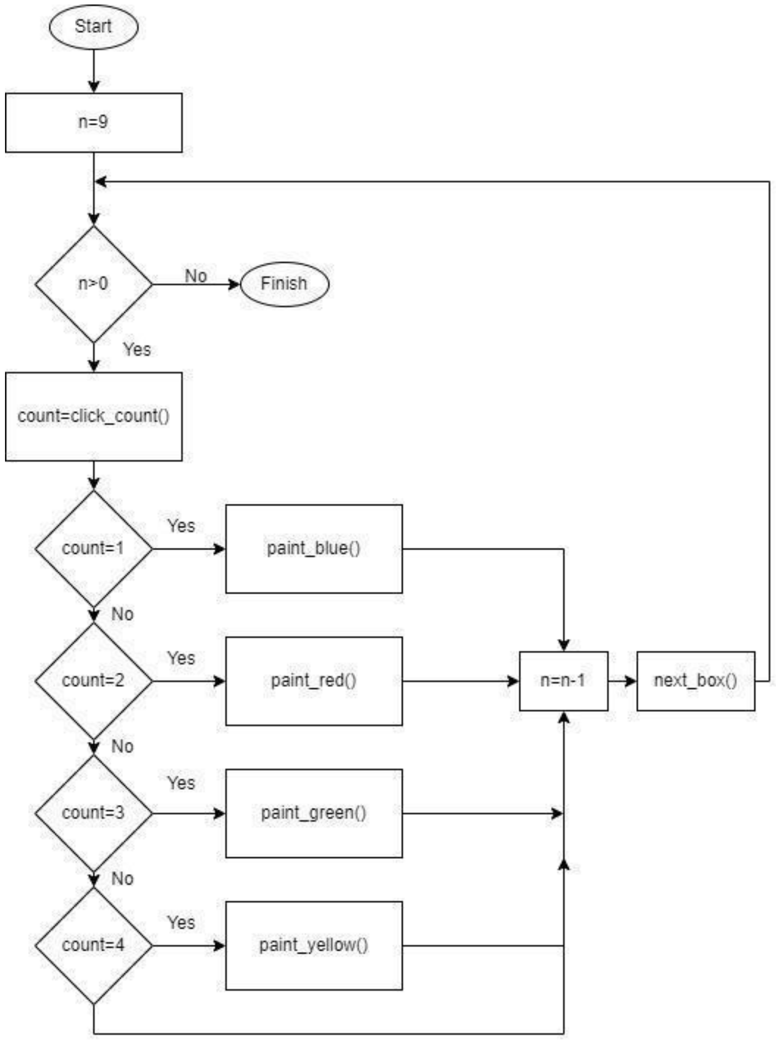
3.1.2. The algorithm design

All groups started with defining the direction of progress as from right to left, from top to bottom. Considering the strategies developed by the students, they generally used two basic solution approaches. The first and most preferred strategy is associating each colour with a click count. Colours are usually associated with numbers between 1 and 4 by students. One group tried not to make a sound of their blue colour. Another group used two clicks for both blue and green colours. This group has changed the tone for the colour blue. The second preferred strategy is to create a separate sound for each colour. Some groups tried to make different sounds from the pen, and some attempted to develop a solution using objects that made different sounds (elbow, coin). The algorithm represents the path that leads to the solution of a problem. However, different paths may have advantages and disadvantages over each other. Table 2 shows the best solutions of the groups are error-free ones and the other ones with the least clicks. The groups which prefer the first strategy (associating each colour with a click account) have higher points than the groups which prefer the second strategy. Each solution created by the groups corresponds to an algorithm design. However, since the students did not have prior knowledge of CS, they were not expected to prepare flowcharts for the task. The flowchart of the algorithm design created by the group Rubik's Cube is given in Figure 3. In the algorithm, a loop is repeated for nine cells. The number of clicks in each loop is colour-coded according to the condition that provides the relationship between its format and colour. After the loop is repeated nine times, the algorithm is terminated.

3.1.3. The amount of data usage

In this activity, the number of clicks by the researchers was associated with the "data used". Some groups focused on delivering the message with fewer clicks. Groups generally tried four different sound strategies for four messages. The students connected the algorithm they developed with the data they used. At the end of the activity, the groups stated that they could achieve this with fewer clicks. The statement of one of the students is as follows: *"We only developed logic, but we could also develop different strategies and solutions"*. One student emphasised the importance of collaboration by saying, *"Our strategy has been strengthened with group work"* for the algorithm they designed.

Figure 3
Flowchart of the algorithm design created by the group Rubik's Cube



3.2. Students' Opinions

Students' opinions about the activity were examined in terms of the difficulty of the activity, suggestions for its implementation, and the integration of unplugged CS activity into mathematics education.

3.2.1. The difficulty of the unplugged CS activity and suggestions

The students' opinions are that the activity is generally simple. However, there were a few points where they had difficulties. The first was to remember the matches students made between clicks and colours. In addition, both listening and drawing at the same time forced the students. For students habituated to learning mathematics with traditional course contents and activities, participating in such an activity for the first time in their education life caused them to be excited, act hastily, and mix their strategies. Some opinions of the students about the difficulty of the game are as follows:

> "It was oblivion."
>
> "I had a hard time memorising sounds. Because time was limited."
>
> "I am confused by the sounds."
>
> "I wasn't forced; I was worried."
>
> "Simple and beautiful."

The activity was appropriate and applicable regarding students' age, class, and knowledge level. Some students saw the activity as a game and suggested that they should make the game more challenging. In the students' recommendations for the following activity implementations, the ideas about group work came to the fore. Some suggestions are as follows:

> "I recommend playing the event in different lessons and free activity times by adding something new."
>
> "Groups must have leaders."
>
> "Let the race be done."
>
> "Teams get along well."

3.2.2. Integrating unplugged CS activities into mathematics education

Most students associated the activity with the mathematics class's counting, geometry, and operations. A few students said they did not find it related to the mathematics class. Some students' statements are as follows:

> "It was as if we had fixed a point."
>
> "Our multiplication, addition and coding have improved."
>
> "I associated with geometric shapes and numbers."

The students also answered, *"intelligence and memory have improved"* and *"our mind has become stronger"* regarding different skills. The most important reason is that they establish a relationship between the mathematics course and these skills.

Most students could not establish a relationship between CS-related subjects and the activity. The most important reason for this is the Information Technology and Software Course, the only course that includes CS education in Turkiye, is offered only in the 5th and 6th grades and in limited subjects. The unplugged CS activity that students see most associated with is coding. In addition, some students said that they saw the activity as related to painting on the computer. Examples of their connection to coding are:

> "The computer works with the codes we entered; we worked with the codes."
>
> "It is like the codes of a computer."

3.3. Pre-service Teachers' Opinions

The pre-service teachers' opinions about the activity were examined regarding the difficulties and suggestions for implementing it and the student's participation.

3.3.1. The difficulties experienced during the activity

Classroom management was one of the most prominent problems experienced by pre-service teachers. Pre-service teachers stated they had difficulty maintaining silence in some classes, so click sounds could not be heard during the activity. They tried to solve this problem by frequently warning students. Some pre-service teachers' statements on this subject are as follows:

> "Classroom silence is essential."
>
> "The students understood the activity as a game and made a lot of noise."
>
> "We did not experience any difficulties in general. The students were compatible. A small incident would not be difficult; one of the students wanted to confuse the students by clicking the pencil 1-2 times during the other group activity. We warned the student in a nice language so that he would abandon the behaviour and not repeat it."

Pre-service teachers stated that some students could not understand the activity when they presented it. The solutions they developed for this situation are as follows:

> "Some students could not understand the original purpose. We helped them understand the event by making other friends watch it."
>
> "We had problems such as students' understanding of the activity and students' inexperience with this activity. We explained clearly the activity to the students and showed them how to be applied with materials."

3.3.2. Recommendations for future teachers who will implement the activity

Pre-service teachers generally found the activity joyful and explicit. We can group the suggestions of the pre-service teachers into two groups, for creating groups and implementing the activity. Recommendations for group formation emerged as groups consisting of fewer students and combining different strategies while forming groups. The statements of some of the pre-service teachers regarding this are as follows:

> "Some students were not chosen by their other friends when forming the group. That's why we had problems with group formation. We got over it by talking in a way that helped students match up."

> "The number of groups can be increased, and the number of students in the group can be decreased. It is possible to form groups of two and ensure that each student is active during the activity."

All pre-service teachers were able to carry out the activity successfully. The statement of a pre-service teacher regarding this is as follows: *"I was fortunate; the students didn't have any difficulties anywhere. Naturally, I didn't have any difficulties either because the students listened to the activity silently and focused, so we completed the activity without any problems"*. Of course, the fact that three pre-service teachers acted together and carried out the activity was also influential. The suggestions of the pre-service teachers in terms of their experience after applying the activity are as follows:

> "The classroom must be quiet."

> "The time can be extended. Three lesson hours is more suitable."

> "Competitions can be made with other classes."

> "Some groups had trouble understanding the activity. An explanation can be made with a pilot test."

> "The activity can be deepened by making the task more difficult."

Some of the suggestions of the pre-service teachers to the teachers/pre-service teachers who will implement this activity in the future are as follows:

> "My advice to teachers who will implement this activity; is to ensure silence in the classroom environment, motivate students towards the activity, and keep them motivated. After these conditions are met, there will be no problems."

> "Being the first group to tell is disadvantageous. But it reinforces the strategies of other groups by observing."

3.3.3. Observations on students' participation

Pre-service teachers found the activity generally joyful for students. Although some difficulties were mentioned above in the implementation of the activity, the students were eager to repeat the activity. Examples of the statements of some pre-service teachers are as follows: *"The students wanted to play the game again"*, *"They enjoyed winning and competing"*, and *"The students enjoyed the activity very much. Although there were problems in terms of time, the students unanimously gave extra time from the break times because they wanted to re-apply the activity"*. Pre-service teachers also stated that this activity improved students' creativity, reasoning, etc. skills, developed unique solutions, and positively affected their attitudes toward the mathematics lesson. The statements of the pre-service teachers regarding these observations are as follows:

> "This activity improves students' creativity; I think that such activities should be included more frequently in learning environments because it is an activity that is different from the course and the curriculum but will develop the student in different ways."

> "I enjoyed observing each student's strategies, such as slow, fast, and side-clicking the pen. I saw that every student is unique."

> "I think it helps students' reasoning."

> "The students interacted with each other as a group.... They improved their ability to focus, communicate with friends, and be in a group."

> "They also developed strategies to solve the problems they experienced during the event and to reduce the error."

> "They compared their solution with the solution of others. They also added new solutions that came to mind while watching the solution of others."
>
> "They gave positive feedback about the math class."

This study was carried out by being aware that the CT phenomenon is new concerning educational sciences, the lack of visibility of practical examples, the difficulties in integrating due to the tight structure of the curriculum, and the educational needs of teachers and students in the academic context. The study focused on how CT can be integrated into mathematics education and how CT can be penetrated the teaching experiences and educational processes of pre-service mathematics teachers. In this context, the researchers designed an unplugged CS activity, and the pre-service teachers implemented it in real-classroom environments.

4. Results and Discussion

This study examines how CT can be integrated into mathematics education through an unplugged CS activity. We developed the unplugged CS activity and examined its implementation and the opinions of students and pre-service teachers regarding it. The research findings are discussed regarding integrating CT into mathematics education and what needs to be done for mathematics teachers and pre-service teachers to integrate CS activities into their classrooms.

According to Aho's (2012) definition, CT is a thinking process for formulating solutions to problems using algorithms and computational steps. This unplugged CS activity is a problem of transmitting data most efficiently and accurately between transmitter and receiver. The students performed data collection, data analysis, decomposition, pattern recognition, algorithm design, and testing and debugging, which are components of CT, during the activity. CT components were considered in the activity design, but a tool measuring CT components was not used to evaluate the effectiveness of the activity. Babazadeh and Negrini (2022) reviewed studies conducted between 2016 and 2020 on CT assessment tools and components evaluated through a systematic literature review. They concluded that there is no assessment tool that fully measures the components of CT, but studies argue that the components of CT generally measure programming skills. Moreover, we adopted implicit learning, not focused on students' explicit knowledge of CS concepts. Bell and Lodi (2019, p. 345) stated that CS concepts in an unplugged approach are made accessible with practical experiences. Furthermore, the fact that each group developed a solution should be considered proof of their engagement with these CS concepts. In future studies, teaching about how students are exposed to various CS concepts should be examined closely (Luo, Israel, & Gane, 2022).

Unplugged CS activities can be enjoyable for students to engage in mathematics and CS education. Benton et al. (2018) stated that carefully designed and sequenced CT activities and appropriate teacher support help students understand difficult math questions. The students who participated in the study stated they established a relationship between the activity and mathematics education, which they associated with intelligence, creativity, and other skills. This relationship is promising regarding the reproduction and implementation of such activities for mathematics education. Besides, the students needed help connecting with CS concepts. Information Technology and Software Course is compulsory in the 5th and 6th grades of middle schools and the 9th grades of Science High Schools in Turkiye. It is elective or extra-curricular in other levels and institutions. So, the students in this study did not have enough academic knowledge of CS. Sun, Hu, and Zhou (2021) found that students' mathematics performance and previous programming experience would significantly affect their CT skills.

Polat and Yılmaz (2022), comparing unplugged and plugged-in activities, stated that unplugged activities were more effective on CT and that unplugged activities were perceived as more educational but not problematic and tiresome. In this study, although the students had some difficulties understanding the activity because they had such an experience for the first time, they were not bored and wanted to repeat it. However, unplugged CS activities should enable collaborative experiences (Jun, 2018). The activity developed in this study supported collaboration among the students and allowed students to take different roles (basically transmitter and receiver). The groups also were generally compatible and did not experience any problems during the activity. Busuttil and Formosa's (2020) study shows that unplugged computing is an effective pedagogical strategy for increasing student engagement and involvement while also encouraging teamwork and collaboration. Jun (2018) stated that an unplugged activity should be fun for the students. Curzon (2015) also implied that unplugged CS activities are fun for all levels in gaining CS concepts. This study supports that the students liked the unplugged CS activity and had fun.

Although the pre-service mathematics teachers who carried out the activity implementation had prior knowledge of algorithms and programming, they did not have experience integrating CS activities into their teaching. Learning the skills necessary to include programming and CT in compulsory K-12 classrooms is often not part of a teacher's formal education (Hickmott, Prieto-Rodriguez, & Holmes, 2018). However, the recent emphasis on integrating CT into K-12 education indicates the need to prepare future teachers properly (Alqahtani, Hall, Leventhal, & Argila, 2022). In this study, pre-service mathematics teachers received introductory training in CS and then gained classroom experience in integrating CS with mathematics education. Such an experience was the first for pre-service teachers and caused some difficulties in classroom management. Classroom management should be a part of teacher education in designing and implementing CT-integrated lessons (Yadav et al., 2014). Munasinghe, Bell and Robins's (2021) research reveals that the nature of the teacher's knowledge of computational concepts can be either that the computational meaning is unknown, that the computational context is unclear, or that their applicability is uncertain.

In preparing pre-service teachers to integrate CT into their teaching, they must receive training in the context of their disciplines, as well as stand-alone courses (Yadav et al., 2022). In this study, while pre-service teachers received training on CT practices and CT tools in the context of their disciplines in the same semester, they also experienced the knowledge and skills they acquired in the classroom environment. It is as essential for pre-service teachers to gain the skills to integrate CT into their teaching as it is for them to gain CT skills. The first step in enabling them to do this in their future teaching is to be able to design and implement such activities. In Switzerland, scalable game design is one of the pillars of a 14-week CS education course designed for primary pre-service teachers (Lamprou & Repenning, 2018). In this course, each week is divided into two parts, theory and activity, for pre-service teachers to gain skills to integrate CT into their teaching. In this study, we did not design a compulsory CS education for pre-service teachers. Still, we tried to show how they can benefit from CS education in mathematics education regarding unplugged CS activity design and implementation. We hope that this effort sets an example for future studies. Because; courses benefit the understanding of computational thinking concepts, tools, and practices among pre-service teachers (Lamprou & Repenning, 2018; Mouza et al., 2017; Yadav et al., 2022). However, researches show that integrating CT meaningfully into lessons with disciplinary content and pedagogy is not easy for pre-service teachers (Mouza et al., 2017). These studies have important implications for developing teacher education programs that equip pre-service teachers with the necessary competencies to integrate CT into their future teaching and learning environment.

5. Conclusion and Implications

Unplugged CS activities in mathematics education make teaching and learning more joyful, active, engaging, challenging, and motivating for students. Students must master concepts from both fields and establish a link between that discipline and CS in integrating CT into different disciplines. In this study, the student's difficulties in connecting the activity and the CS concepts stem from the fact that they did not have sufficient training on this subject. In addition, the teacher of the relevant discipline and the CS teacher can collaborate in overcoming the difficulties encountered in such activities. Such collaborations significantly differ in the design of learning and teaching processes (Mumcu, Uslu, & Yıldız, 2023). The priority in integrating disciplines is the collaboration of colleagues from different disciplines (Mumcu, Uslu, Özdinç, & Yıldız, 2022).

It is necessary to consider the content of teacher education programs and the PD needs of teachers in integrating CT into different disciplines. Nevertheless, PD programs effectively teach teachers from different disciplines to include CT activities in their curriculum (Jocius et al., 2019; Yadav et al., 2017). We need to develop teacher education courses which develop pre-service teachers' competencies in designing and implementing CT-integrated lessons to engage their students with CT ideas. It should be encouraged to design and implement such activities, especially in courses that enable pre-service teachers to gain real school experience, such as teaching practice.

Furthermore, the CS course should be one of the compulsory courses, such as mathematics and science, in the K12 curriculum. Today, the sciences are increasingly computational, which is necessary to develop such skills in our students and integrate them with other disciplines. Thus, it will be easier to highlight and implement the interdisciplinary dimension of CS. Future research topics include how these activities will be designed, implemented, and evaluated, how CS concepts, practices, and tools can be embedded into other disciplines and which competencies teachers and pre-service teachers should have.

6. Limitations

This activity was carried out in classes with an average of 20 students. Since the activity is based on sound, it is necessary to carry out the activity in a broader area where students can hear the click sounds to make the process more efficient and not strain the classroom management skills of the pre-service teachers. This activity was applied in classes between 40 and 60 minutes. This duration is not enough. It would be beneficial to allocate more time to the activity.

Regarding the key components of CT, there is a need for an elaborate analysis of the groups' solutions. For this purpose, CT assessment strategies of the activity should be revised.

Author contributions: All authors are agreed with the results and conclusions.

Declaration of interest: None of the authors have a conflict of interest in this project.

Ethics declaration: In this study, all scientific ethical rules were followed. Informed consent was obtained from all participants before the study.

Funding: No funding source is reported for this study.

References

Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal, 55*(7), 832-835. https://doi.org/10.1093/comjnl/bxs074

Alqahtani, M. M., Hall, J. A., Leventhal, M., & Argila, A. N. (2022). Programming in mathematics classrooms: Changes in pre-service teachers' intentions to integrate robots in teaching. *Digital Experiences in Mathematics Education*, *8*(1), 70-98. https://doi.org/10.1007/s40751-021-00096-6

Aranda, G., & Ferguson, J. P. (2018). Unplugged Programming: The future of teaching computational thinking?. Pedagogika, *68*(3), 279-292. https://doi.org/10.14712/23362189.2018.859

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), 3-19. https://doi.org/10.21585/ijcses.v5i4.138

Bell, T. (2021, October 18-20). Computational thinking: online and offline, plugged and unplugged. In *The 16th Workshop in Primary and Secondary Computing Education (WiPSCE '21)* [Virtual presentation]. https://doi.org/10.1145/3481312.3488711

Bell, T., & Henderson, T. (2022). Providing a CS unplugged experience at a distance. *ACM Inroads, 13*(4), 26-31. http://dx.doi.org/10.1145/3571093

Bell T. & Lodi M. (2019) Constructing computational thinking without using computers. *Constructivist Foundations 14*(3), 342–351.

Bell, T., Rosamond, F., & Casey, N. (2012). Computer science unplugged and related projects in math and computer science popularization. In H. Bodlaender, R. Downey, F. Fomin, & D. Marx (Eds.), *The multivariate algorithmic revolution and beyond* (pp. 398–456). Springer.

Bell, T., Witten, I. H., & Fellows, M. (1998). *Computer Science Unplugged: Off-line activities and games for all ages*. https://classic.csunplugged.org/documents/books/english/unplugged-book-v1.pdf

Benton, L., Saunders, P., Kalas, I., Hoyles, C., & Noss, R. (2018). Designing for learning mathematics through programming: A case study of pupils engaging with place value. *International Journal of Child-Computer Interaction*, *16*, 68-76. https://doi.org/10.1016/j.ijcci.2017.12.004

Busuttil, L., & Formosa, M. (2020). Teaching computing without computers: Unplugged computing as a pedagogical strategy. *Informatics in Education, 19*(4), 569-587.

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). *Developing computational thinking in compulsory education-Implications for policy and practice* (No. JRC104188). Joint Research Centre.

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

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*(2), 77–101. https://doi.org/10.1191/1478088706qp063oa

Cambridge Dictionary (2022). www.dictionary.cambridge.org

Curzon, P. (2015). Unplugged computational thinking for fun. In T. Brinda, N. Reynolds, R. Romeike & A. Schwill (Eds.), *Key Competencies in Informatics and ICT* (pp. 15-27). Universitätsverlag Potsdam.

Dickes, A. C., Farris, A. V., & Sengupta, P. (2020). Sociomathematical norms for integrating coding and modeling with elementary science: A dialogical approach. *Journal of Science Education and Technology, 29*(1), 35-52. https://doi.org/10.1007/s10956-019-09795-7

European Commission / EACEA / Eurydice (2019). *Digital Education at School in Europe. Eurydice Report*. Publications Office of the European Union.

Feurzeig, W., Papert, S. A., & Lawler, B. (2011). Programming-languages as a conceptual framework for teaching mathematics. *Interactive Learning Environments*, *19*(5), 487-501. https://doi.org/10.1080/10494820903520040

Guzdial, M. (1994). Software-realized scaffolding to facilitate programming for science learning. *Interactive Learning Environments*, *4*(1), 1-44. https://doi.org/10.1080/1049482940040101

Guzdial, M. (2008). Education Paving the way for computational thinking. *Communications of the ACM*, *51*(8), 25-27. https://doi.org/10.1145/1378704.1378713

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). *Design Science in Information Systems Research. MIS Quarterly, 28*(1), 75–105. https://doi.org/10.2307/25148625

Hickmott, D., Prieto-Rodriguez, E., & Holmes, K. (2018). A scoping review of studies on computational thinking in K–12 mathematics classrooms. *Digital Experiences in Mathematics Education*, *4*(1), 48-69. https://doi.org/10.1007/s40751-017-0038-8

Huang, W., & Looi, C. K. (2021). A critical review of literature on "unplugged" pedagogies in K-12 computer science and computational thinking education. *Computer Science Education*, *31*(1), 83-111. https://doi.org/10.1080/08993408.2020.1789411

Irgens, G. A., Dabholkar, S., Bain, C., Woods, P., Hall, K., Swanson, H., ... & Wilensky, U. (2020). Modeling and measuring high school students' computational thinking practices in science. *Journal of Science Education and Technology*, *29*(1), 137-161. https://doi.org/10.1007/s10956-020-09811-1

Jocius, R., Joshi, D., Albert, J., Barnes, T., Robinson, R., Cateté, V., ... & Andrews, A. (2021, March). The virtual pivot: Transitioning computational thinking PD for middle and high school content area teachers. *In Proceedings of the 52nd ACM Technical Symposium on Computer Science Education* (pp. 1198-1204). https://doi.org/10.1145/3408877.3432558

Jun, W. (2018, October 17-19). *A study on development of evaluation standards for unplugged activity* [Paper presentation]. International Conference on Information and Communication Technology Convergence (ICTC). http:// doi.org/10.1109/ICTC.2018.8539505

Kalelioglu, F., Gulbahar, Y., & Kukul, V. (2016). A framework for computational thinking based on a systematic research review. *Baltic J. Modern Computing, 4*(3), 583-596.

Kite, V., & Park, S. (2023). What's computational thinking?: Secondary science teachers' conceptualizations of computational thinking (CT) and perceived barriers to ct integration. *Journal of Science Teacher Education, 34*(4), 391-414. https://doi.org/10.1080/1046560X.2022.2110068

Kong, S. C. (2019). Learning composite and prime numbers through developing an app: An example of computational thinking development through primary mathematics learning. In S. C. Kong & H. Abelson (Eds.), *Computational thinking education* (pp. 145-166). Springer.

Lamprou, A., & Repenning, A. (2018). Teaching how to teach computational thinking. *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education*, 69–74. https://doi.org/10.1145/3197091.3197120

Lee, I., & Malyn-Smith, J. (2020). Computational thinking integration patterns along the framework defining computational thinking from a disciplinary perspective. *Journal of Science Education and Technology*, *29*(1), 9-18. https://doi.org/10.1007/s10956-019-09802-x

Luo, F., Israel, M., & Gane, B. (2022). Elementary computational thinking instruction and assessment: A learning trajectory perspective. *ACM Transactions on Computing Education*, *22*(2), 1-26. https://doi.org/10.1145/3494579

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

Mouza, C., Yang, H., Pan, Y.-C., Ozden, S. Y., & Pollock, L. (2017). Resetting educational technology coursework for pre-service teachers: A computational thinking approach to the development of technological pedagogical content knowledge (TPACK). *Australasian Journal of Educational Technology*, *33*(3), 3. https://doi.org/10.14742/ajet.3521

Mumcu, F., Uslu, N. A., & Yıldız, B. (2023). Teacher development in integrated STEM education: Design of lesson plans through the lens of computational thinking. *Education and Information Technologies, 28*, 3443–3474. https://doi.org/10.1007/s10639-022-11342-8

Mumcu, F., Uslu, N. A., Özdinç, F., & Yıldız, B. (2022). Exploring teacher development courses in the lens of integrated STEM education: A holistic multiple case study. *International Journal of Contemporary Educational Research, 9*(3), 476-491. https://doi.org/10.33200/ijcer.1035464

Munasinghe, B., Bell, T., & Robins, A. (2021, February). Teachers' understanding of technical terms in a computational thinking curriculum. In *Proceedings of the 23rd Australasian Computing Education Conference* (pp. 106-114). https://doi.org/10.1145/3441636.3442311

Ozyurt, O., & Ozyurt, H. (2023). A large-scale study based on topic modeling to determine the research interests and trends on computational thinking. *Education and Information Technologies*, *28*(3), 3557-3579. https://doi.org/10.1007/s10639-022-11325-9

Özdinç, F., Kaya, G., Mumcu, F., & Yıldız, B. (2022). Integration of computational thinking into STEM activities: an example of an interdisciplinary unplugged programming activity. *Science Activities – Projects and Curriculum Ideas in STEM Classrooms, 59*(3), 151-159. https://doi.org/10.1080/00368121.2022.2071817

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

Pei, C., Weintrop, D., & Wilensky, U. (2018). Cultivating computational thinking practices and mathematical habits of mind in lattice land. *Mathematical Thinking and Learning*, *20*(1), 75-89. https://doi.org/10.1080/10986065.2018.1403543

Polat, E., & Yilmaz, R. M. (2022). Unplugged versus plugged-in: examining basic programming achievement and computational thinking of 6th-grade students. *Education and Information Technologies, 27*(7), 9145-9179. https://doi.org/10.1007/s10639-022-10992-y

Richey, R. C., & Klein, J. D. (2005). Developmental research methods: Creating knowledge from instructional design and development practice. *Journal of Computing in Higher Education, 16*(2), 23-28. https://doi.org/10.1007/BF02961473

Richey, R. C., & Klein, J. D. (2007). *Design and Development Research: Methods, Strategies, and Issues*. Routledge.

Richey, R. C., & Klein, J. D. (2014). *Design and development research*. In J. M. Spector, M. D. Merrill, J. Elen & M. J. Bishop (Eds.), *Handbook of Research on Educational Communications and Technology*, (pp. 141-150). Springer. https://doi.org/10.1007/978-1-4614-3185-5_12

Science on Stage Europe (2022a). *European Code League*. https://www.science-on-stage.eu/european-code-league

Science on Stage Europe (2022b). *Coding in STEM Education*. https://www.science-on-stage.eu/material/coding-in-stem-education

Seow, P., Looi, C. K., How, M. L., Wadhwa, B., & Wu, L. K. (2019). Educational policy and implementation of computational thinking and programming: Case study of Singapore. In S. C. Kong & H. Abelson (Eds.), *Computational thinking education* (pp. 345-361). Springer. https://doi.org/10.1007/978-981-13-6528-7_19

Sun, L., Hu, L., & Zhou, D. (2021). Improving 7th-graders' computational thinking skills through unplugged programming activities: A study on the influence of multiple factors. *Thinking Skills and Creativity, 42*, 100926. https://doi.org/10.1016/j.tsc.2021.100926

Taşlıbeyaz, E., Kurşun, E., & Karaman, S. (2020). How to develop computational thinking: a systematic review of empirical studies. *Informatics in Education*, *19*(4), 701-719.

Tsortanidou, X., Daradoumis, T., & Barberá, E. (2021). A K-6 computational thinking curricular framework: pedagogical implications for teaching practice. *Interactive Learning Environments*, 1-21. https://doi.org/10.1080/10494820.2021.1986725

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*, 53-64. https://doi.org/10.1007/s10956-019-09801-y

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, *25*(1), 127-147. https://doi.org/10.1007/s10956-015-9581-5

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

Wing, J. M. (2011). Computational thinking: What and why. *The Link Magazine*, *6*. http://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why

Yadav, A., Caeli, E. N., Ocak, C., & Macann, V. (2022). Teacher Education and Computational Thinking: Measuring Pre-service Teacher Conceptions and Attitudes. In *Proceedings of the 27th ACM Conference on on Innovation and Technology in Computer Science Education, 1*, 547–553. https://doi.org/10.1145/3502718.3524783

Yadav, A., Gretter, S., Hambrusch, S., & Sands, P. (2017). Expanding computer science education in schools: Understanding teacher experiences and challenges. *Computer Science Education, 26*, 235–254. https://doi.org/10.1080/08993408.2016.1257418

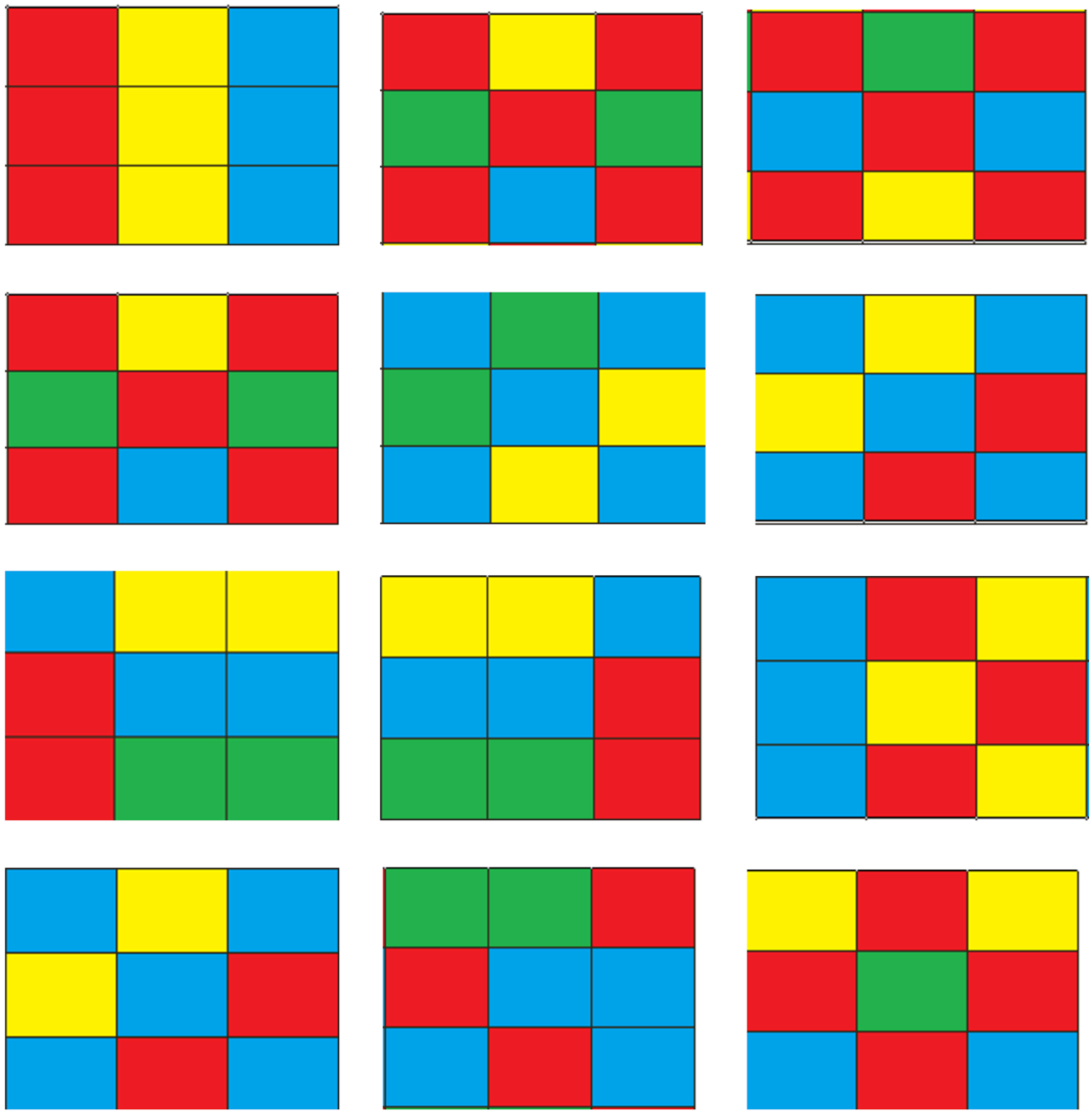
Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational Thinking in Elementary and Secondary Teacher Education. *ACM Transactions on Computing Education*, *14*(1), 1-16. https://doi.org/10.1145/2576872

Zha, S., Jin, Y., & Moore, P., & Gaston, J. (2020). Hopscotch into coding: Introducing pre-service teachers computational thinking. *Techtrends, 64,* 17-28. https://doi.org/10.1007/s11528-019-00423-0

Zhang, L., & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K-9. *Computers & Education, 141*, 103607. https://doi.org/10.1016/j.compedu.2019.103607

Appendix 1.
Blank 3x3 Tables

Sample Task Tables



Appendix 2.

Reflective Report Form for Pre-Servive Teachers

Did you enjoy the activity? What did you like?

Where did you have difficulties while implementing the activity? Where did your students struggle?

Did you observe that this activity impacted the students? How?

How did you overcome the problems you experienced during the activity?

What would you advise teachers who will implement this activity? What kind of arrangement would you suggest for the activity design?

Group Activity Form for Pre-Service Teachers

Grade:

Class size:

How many groups were formed?

How were the groups created?

The difficulties experienced during the event, how they were overcome, and the solutions offered:

Recommendations for educators who will implement the activity:

Your observations about the student participation in the activity (did the students enjoy the activity, did they want to do the activity again, where did they have difficulties, etc.):

Group Activity Form for Students

Did you enjoy the event? What did you like?

Where did you have difficulties during the activity?

What would you suggest to other students and teachers who will implement this activity?

How did you connect the activity with the mathematics lesson?

How did you associate the activity with CS?

What strategy did you develop as a group?