# PRE-SERVICE MATHEMATICS TEACHERS' SCRATCH-SUPPORTED MICRO-TEACHING: A TWO-DIMENSIONAL SAMR–COMPUTATIONAL THINKING ANALYSIS

Hatice Akkoç

Marmara University, Türkiye

*This study examines how pre-service mathematics teachers (PMTs) integrate Scratch into their micro-teaching lessons based on the levels of computational thinking (CT) integration. The data of a case study with three PMTs include their micro-teaching lesson videos, lesson plans, and teaching notes. The data analysis is based on the SAMR-based CT integration matrix with an emphasis on a disciplinary approach. Findings showed that participants integrated CT into their lessons at the substitution level only, but also demonstrated some characteristics of the augmentation level. They prioritized Scratch use over fostering CT and approached mathematics and CT as separate domains. The paper discusses the theoretical implications for operationalizing the SAMR-based CT integration.*

## INTRODUCTION

Papert (1980) argued that programming offers students opportunities to develop mathematical ideas and problem-solving skills. Building upon his ideas, Wing (2006) introduced the term computational thinking (CT), emphasizing it as "a fundamental skill for everyone, not just computer scientists" (p. 33), and defined it as the process of formulating problems and their solutions in a manner that a computational agent can execute. There is broad agreement that CT involves *abstraction*, *decomposition*, *pattern recognition*, and *algorithmic thinking*, which have been reported as CT components (Grover & Pea, 2013; Wing, 2006). To analyze how learners structure and control computational processes, Brennan and Resnick (2012) define CT concepts as the core ideas learners engage with during programming activities, including *sequences*, *loops*, *events*, *parallelism*, *conditionals*, *operators*, and *data*. This study addresses only *data*, as it emerged in the findings, and therefore necessitates prior explanation. *Data* refers to information that programs store, manipulate, and represent, typically through variables that function as named containers whose values can change during program execution. *Variables* enable learners to model quantities, track change, and represent mathematical relationships dynamically, thereby supporting abstraction and generalization across computational tasks (Brennan & Resnick, 2012).

The integration of CT into K-12 education is essential for preparing students to survive in an increasingly digital and interdisciplinary world (National Research Council, 2011; Weintrop et al., 2016). Among the various strategies for developing CT in K-12 contexts, block-based or visual programming environments have proven particularly



2026. In H. Krzywacki, M. Hähkiöniemi, E. Haataja, M.S. Hannula & P. Hästö (Eds.). *Proceedings of the 49th Conference of the International Group for the Psychology of Mathematics Education: Research Reports*, pp. 27-34. PME.

effective (Zhang & Nouri, 2019). Scratch, among others, provides a user-friendly, visual interface that enables young learners to grasp complex computational ideas without the cognitive load of syntax errors (Brennan & Resnick, 2012). Previous studies have predominantly focused on in-service teachers (e.g., Kite & Park, 2022; Rich et. al., 2024). However, research specifically addressing how PMTs conceptualize and implement CT during their initial teaching experiences remains limited. Studies indicate that pre-service teachers often face challenges in their pedagogical understanding of CT and require structured support to integrate it meaningfully into their subject teaching (e.g., Yun & Crippen, 2024).

Given the pivotal role of CT in mathematics education and mathematical thinking (MT), it is necessary to explore how PMTs integrate CT into their teaching. In light of this gap, the present study aims to investigate how PMTs integrate Scratch into their micro-teaching lessons and explore this integration through the lens of their CT integration levels, adopting the matrix proposed by Yeni et al. (2024). The following research question was formulated: At what level of CT integration did PMTs use Scratch in their micro-teaching lessons?

## THEORETICAL BACKGROUND

### Theoretical Perspectives Informing the Study

To investigate how Scratch is integrated into instruction, a sole focus on levels of technology use risks the evolving needs of integrating CT into disciplinary teaching. Recent research highlights the growing importance of integrating CT into classroom practices, not just technology (Weintrop et al., 2016). Achieving this integration requires pedagogical approaches that meaningfully embed CT practices into content learning, rather than adding isolated technology tasks (e.g., Cabrera et al., 2023). In this direction, we will utilize Yeni et. al.'s (2024) framework, which adopts Puentedura's (2006) SAMR model.

The SAMR model conceptualizes technology integration in four progressive levels: *substitution*, *augmentation*, *modification*, and *redefinition* (Puentedura, 2006). These stages outline how technology transforms instructional practice, ranging from surface-level tool replacement to the creation of entirely new learning tasks and experiences. Technology acts as a direct replacement for traditional tools at the *substitution* level, with no functional change (Puentedura, 2006). It offers convenience or efficiency, but underlying cognitive and pedagogical tasks remain unchanged (Hamilton et al., 2016). At the *augmentation* level, technology begins to add value to the task. Although the task remains unchanged, integrating technology supports more effective or efficient performance (Blundell et al., 2022). At the *modification* level, teachers may adjust the task's structure, process, and goals to leverage the affordances of digital tools. Teachers at this stage begin to reconceptualize pedagogy in light of what technology makes possible. At the *redefinition* level, technology enables the creation of entirely new tasks

and student-centred learning experiences previously inconceivable without digital tools (Puentedura, 2006).

While the SAMR model is widely used to classify levels of technology integration, scholars have often criticized it for being too generic and lacking specificity in discipline-based applications (Blundell et al., 2022; Hamilton et al., 2016). Therefore, some scholars have integrated the SAMR model into other frameworks to enhance its theoretical depth and applicability. In the context of CT integration, Yeni et al. (2024) make a critical contribution by refining the utility of the SAMR model and offering a more nuanced, discipline-specific operationalization of technology integration through a two-dimensional matrix. The first two columns delineate CT integration from a disciplinary perspective. It describes four categories aligned with the SAMR levels (Yeni et. al., 2024). The section below provides a detailed description.

**An Analytical Framework for Examining Scratch-supported Micro-Teaching**

Drawing on the theoretical perspectives outlined above, the current study uses the first two columns of Yeni et al.'s (2024) matrix (which also correspond to the first two columns of Table 1) as an analytical framework to evaluate how Scratch-supported micro-teaching practices evolve across the levels of the SAMR model in relation to CT. The remaining columns in the matrix, although valuable for broader curricular or interdisciplinary analyses (e.g., CT integration across STEM domains), were excluded because they extend beyond the scope of this study.

Table 1: The analytical framework of the study

| SAMR levels | Definitions of CT integration levels (Yeni et al., 2024) | Examples from Scratch-supported lessons |
|---|---|---|
| Substitution | No functional change. (S1)<br>CT concepts and subject concepts are learnt separately in each discipline (Disciplinary). (S2) | Using Scratch to replace a physical manipulative with a visual representation on screen. |
| Augmentation | A direct tool substitute, offering functional improvements. (A1)<br>Making a connection to CT concepts within a common theme (Multi-disciplinary) (A2) | Scratch provides automated feedback or visual interactivity. |
| Modification | A significant task redesign. (M1)<br>A deeper understanding of the subject. (Inter-disciplinary) (M2) | Designing or modifying Scratch projects to explore algebraic relationships. |
| Redefinition | The creation of new tasks that were previously inconceivable. (R1)<br>Two disciplines are connected to solve real-world problems (Trans-disciplinary). (R2) | Students design original Scratch animations to model real-world mathematical problems. |

In the second column of Table 1, there are two separate codes (e.g., S1 and S2) that were operationalized for the current study: the first concerns the SAMR model's levels and the second is related to the disciplinary approach which defines whether CT and MT are linked. The last column describes the author's interpretations of the definitions in the context of Scratch-supported lessons.

## METHODOLOGY

This study employed a multiple case study design (Yin, 2014) to investigate how PMTs integrate Scratch into micro-teaching. The researcher conducted this study in the fourth year of a teacher preparation program at a state university in İstanbul, Türkiye. The program qualifies its participants to teach mathematics at the lower secondary level to students aged 12–15. The study was situated in a selective course called "Computational Thinking". One of the course requirements was to design a mathematics lesson that integrates Scratch into micro-teaching. Each PMT planned and implemented a micro-teaching lesson of 20–30 minutes integrating Scratch as an instructional tool for mathematics teaching in a simulated classroom environment with peer learners. PMTs taught their peers as if they were students in a lower-secondary classroom, and their peers worked individually or in pairs using their own devices. The lesson plans and teaching notes of course participants were analysed using the analytical framework. Most of the course participants aimed to demonstrate pre-prepared Scratch project links. However, to ensure theoretical diversity (Yin, 2014) and to examine CT integration from different perspectives, PMTs who also used tasks requiring the design of a Scratch project were selected. Among them, three PMTs volunteered to participate in the study. While the cases are the three micro-teaching lessons, the unit of analysis is the level of CT integration demonstrated in each micro-teaching lesson.

A systematic analysis of each PMT's lesson plan and lesson video identified the independent tasks, marking their beginning when a new mathematical or computational learning goal was introduced and their conclusions. The video segments that included Scratch-supported tasks were analysed according to the analytical framework's CT integration levels (See Table 1). To increase the validity of the findings, the author re-coded the data one month after its initial analysis.

## FINDINGS

Rather than descriptively reporting each case, this section will report findings obtained from the cross-case analysis. Table 2 presents the findings for each code of the observed levels. There were no codes for the *modification* and *redefinition* levels.

The data analysis showed that all three PMTs integrated CT into their lessons at the *substitution* level. Although there were aspects of the SAMR model's *augmentation* level (A1), A2 code was absent since CT and MT were not linked in any of the lessons. However, the data did not indicate a completely identical integration at the *substitution* level. Three cross-case issues implied three types of this level.

Table 2: Findings regarding CT integration levels (—: absent; √: exists)

| CT integration levels | Murat | Mehmet | Harun |
|---|---|---|---|
| S1: No functional change | — | — | — |
| S2: CT concepts and subject concepts are learnt separately in each discipline. (Disciplinary) | √ | √ | √ |
| A1: A direct tool substitute, offering functional improvements. | √ | √ | √ |
| A2: Making a connection to CT concepts within a common theme. (Multi-disciplinary) | — | — | — |

The subsections below present findings regarding each type.

**Substitution type 1: Demonstrating a pre-prepared Scratch project**

All three participants used pre-prepared Scratch projects for demonstration purposes. For example, Murat opened the Scratch project that aimed to find rectangles with the same areas but different perimeters. The sprite prompts the user to enter the short and long sides of two rectangles, draws them, provides feedback on whether the areas are equal, and prompts the user to "try again" when the answer is incorrect. In this case, the tool replaces traditional ones by enabling interaction and providing feedback, resulting in a functional improvement (A1). However, there is no place for learning CT concepts since his peers did not design the Scratch project (S2). The integration was at the *substitution* level, as it does not satisfy the A2 code of the *augmentation* level.

The other two PMTs also used the Scratch projects for demonstration purposes. For example, Mehmet opened a pre-prepared Scratch project and asked his peers to move the ray of an angle to obtain different angle values, using the Scratch project as a protractor, which implied a functional improvement (A1). However, there is no place for learning CT concepts since his peers did not design the Scratch project (S2).

**Substitution type 2: designing a Scratch project with verbal pseudocode**

This type of *substitution* was observed in the cases of Murat and Mehmet. Murat distributed pseudocode worksheets to his peers and asked them to create the Scratch project he demonstrated earlier. This project required them to draw rectangles with the same perimeter but different areas. He told his peers they had to specify eight variables, but did not give them enough time to develop an algorithm. While his peers were working on the Scratch project, Murat provided feedback on where to find the code blocks in the Scratch menu. Therefore, the task promoted learning to use Scratch rather than mathematical or CT concepts (S2).

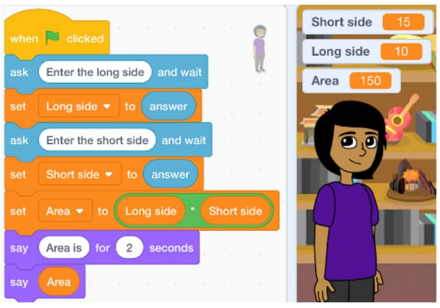
Similarly, Mehmet implemented a task to prepare a Scratch project. He provided the pseudocode for drawing a 90° angle in Scratch. This task requires considering the

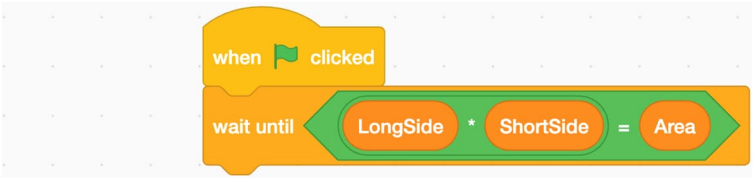
amount and direction of rotation and relating them to the concept of angle. However, Mehmet provided the pseudocode and did not let his peers design projects on their own. Furthermore, he did not consider the angle's direction and used the code "turn right" instead of "turn left" (which is counter-clockwise). There was no link between MT and CT (S2). Therefore, this section of the lesson was coded as the *substitution* level, although it provides functional improvement (A1).

**Substitution type 3: designing a Scratch project without a given algorithm**

Harun was the only participant who used a task that required designing a Scratch project without a given algorithm. He wanted his peers to create a Scratch project that would ask for the short and long sides of a rectangle, calculate its area, and display the result. Although he guided his peers in creating the project on their own devices, the integration level was coded as *substitution* since he focused on mathematical and CT concepts separately (S2). The Scratch task aimed to calculate the areas of rectangles using the formula previously discovered via a unit-square approach, using a paper-and-pencil method. In other words, he used Scratch for reinforcement.

Figure 1(a) Scratch project by Harun ; Figure 1(b) Invalid code block by a peer




In the Scratch task, the most critical CT concept is *variables*. However, Harun focused on only its mathematical meaning. Furthermore, he presented the necessary *variables* for writing the Scratch codes without giving his peers a chance to think them through first. He expressed the area as a *relation*, indicating that he was focusing on the relationship between a dependent variable and its independent variables. In this context, Harun concentrated solely on the mathematical meaning of the variable concept, without explaining it as a CT concept. As the lesson progressed, the most challenging aspect for his peers was defining the *variables* (long side, short side, and area) and assigning values to them. One should assign the answer values to the *variables* using the "answer" block (See the correct code blocks in orange colour in Figure 1a). Instead, one of his peers prepared a code block that represented a condition (Long side × Short side = Area), as shown in Figure 1b. Harun's explanations of the concept of *variables* were insufficient in the context of CT. He just told his peers to assign the answer to the *variables*. He did not explain how Scratch stores the answers using variables to help them overcome their difficulties. In other words, there was no emphasis on the meaning of *variables* (as a CT concept) as containers. Overall,

feedback on the technical knowledge of using Scratch was more dominant than sufficient explanations of CT concepts, and there was no link between MT and CT.

## DISCUSSION

Findings showed that all three PMTs integrated CT into their micro-teaching lessons only at the *substitution* level. Although in one case, the task required writing pseudocode for the Scratch project (third *substitution* type), its implementation did not link MT and CT. In other words, the mathematical content served solely as a context for coding practice. This approach aligns with a disciplinary rather than a multi-disciplinary approach, according to Yeni et al. (2024). Harun could not address his peers' difficulties with CT concepts, such as *variables*, and instead focused solely on its mathematical meaning. Zhang and Nouri's (2019) review of the related literature indicates that students also have a poor understanding of *variables* in the context of CT. Therefore, future research is needed to explore how PMTs or teachers address the relationship between its different meanings in MT and CT. These findings highlight a dual gap: the overemphasis on Scratch as a tool and the absence of coherent connections between CT and mathematics instruction. Other studies confirm similar difficulties that mathematics and science pre-service or in-service teachers face (Kite & Park, 2022; Waterman et al., 2019).

Providing or requesting pseudocode was a cross-case issue that distinguishes the second and third levels of *substitution*. This finding has practical implications for teachers' pedagogical choices when to do so. It may be more pedagogically effective to determine which instructional approach best fits a given context than to judge whether CT integration levels are high or low. Likewise, prior research emphasizes that CT practices should be flexibly aligned with disciplinary goals, and partial engagement with computational elements can be pedagogically sound depending on the learning objectives (Grover & Pea, 2013; Weintrop et al., 2016).

The study's findings have crucial theoretical implications for the shortcomings of the SAMR model, namely that it provides insufficient accounting for context (Hamilton et al., 2016). Utilizing Yeni et al.'s (2024) SAMR-based CT integration matrix with binary codes led to explanations in which some approaches were classified simultaneously at two levels, such as S2 and A1. Although all three participants used Scratch in ways that the SAMR model would classify as *augmentation* (A1), the conceptual framework of this study positioned their approaches at the *substitution* level (S2), as the participants did not establish connections between the subject (mathematics) concepts and CT. In other words, the study's findings resonate with existing criticisms of the SAMR model for overlooking contextual factors (Hamilton et al., 2016), thereby suggesting the potential value of Yeni et al.'s (2024) conceptualization. However, future research is needed to explore higher levels (*modification* and *redefinition*) that were not observed in this study.

## References

Blundell, C. N., Mukherjee, M., & Nykvist, S. (2022). A scoping review of the application of the SAMR model in research. *Computers and Education Open*, *3*, 100093.

Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 Annual Meeting of the American Educational Research Association* (pp. 1–25). Vancouver, Canada.

Cabrera, L., Ketelhut, D. J., Mills, K., Killen, H., Coenraad, M., Byrne, V. L., & Plane, J. D. (2023). Designing a framework for teachers' integration of computational thinking into elementary science. *Journal of Research in Science Teaching*, *61*(6), 1326–1361.

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

Hamilton, E. R., Rosenberg, J. M., & Akcaoglu, M. (2016). The Substitution Augmentation Modification Redefinition (SAMR) model: A critical review and suggestions for its use. *TechTrends*, *60*(5), 433–441.

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.

National Research Council. (2011). *Report of a workshop on the pedagogical aspects of computational thinking*. The National Academies Press.

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

Puentedura, R. R. (2006). *Transformation, technology, and education*. Hippasus.

Rich, K. M., Yadav, A., & Fessler, C. J. (2024). Computational thinking practices as tools for creating high cognitive demand mathematics instruction. *Journal of Mathematics Teacher Education*, *27*(2), 235–255.

Waterman, K. P., Goldsmith, L., & Pasquale, M. (2019). Integrating computational thinking into elementary science curriculum. *J. of Science Education and Technology*, 29, 53–64.

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, 127–147.

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

Yeni, S., Grgurina, N., Saeli, M., Hermans, F., Tolboom, J., & Barendsen, E. (2024). Interdisciplinary integration of computational thinking in K–12 education: A systematic review. *Informatics in Education*, *23*(1), 223–278.

Yin, R. K. (2014). *Case study research: Design and methods* (5th ed.). SAGE.

Yun, M., & Crippen, K. J. (2024). Computational Thinking Integration into Pre-Service Science Teacher Education: A Systematic Review. *Journal of Science Teacher Education*, 36(2), 225–254.

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