Informatics in Education, 2026, Vol. 25, No. 1, 173–199

DOI: https://doi.org/10.15388/infedu.2506.019

Computational Thinking Across Disciplines: A Taxonomy of Pedagogical Approaches as Reflected in Prospective Teachers' Simulations of Computational Processes

Noa RAGONIS[1] [0000-0002-8163-0199], Orit HAZZAN[2] [0000-0002-8627-0997]

[1]Computer Science Studies and Integrative STEM Education, School of Education, Beit Berl College, Israel

[2]Education in Science and Technology, Technion – Israel Institute of Technology, Israel

e-mail: noarag@beitberl.ac.il, oritha@technion.ac.il

Abstract. Computational thinking (CT) is widely recognized as a key 21st-century competence, yet its integration across disciplines remains unclear for many educators. This study explores how prospective teachers identify and express CT through scripts representing computational processes in school subjects of their choice. The challenge of integrating CT in teacher preparation programs in non-STEM-related fields is also addressed. Using a mixed-methods approach, we analyze projects and accompanying reflective analyses from 375 prospective teachers who created Scratch-based scripts aligned with computational processes in STEM and non-STEM subjects. Data analysis yielded a taxonomy of pedagogical strategies reflecting diverse instructional approaches. The study underscores the value of guided, discipline-specific CT activities in teacher preparation programs and highlights how script development of computational processes fosters both subject-matter understanding and computational thinking. The results suggest holistic lens in evaluating CT integration and offer evidence-based insights for embedding CT meaning-fully into teacher preparation programs across disciplines.

Keywords: Computational thinking, Computational process, Teacher preparation, Prospective teachers, Skills, Taxonomy, Simulations

1. Introduction

Students growing up in today's education system will live their adult lives in a multitasking, multifaceted, technologically driven, and highly diverse environment. It is, therefore, essential that they acquire the skills necessary to navigate such complexity. Computer science is increasingly recognized not only as a field of practical applications but also as a

theoretical and foundational science – one that contributes significantly to the research and development of other disciplines. There is growing consensus that the basic concepts and modes of thinking in computer science are essential for functioning effectively across all domains of life (Boholano, 2017; Harper, 2018).

Over the past two decades, this recognition has led to the emergence of a broader educational approach that asserts that the thinking skills and problem-solving strategies inherent in computer science are important for everyone – not just computer scientists. These skills have been encapsulated in the concept of computational thinking (CT), a term introduced by Wing (2006) and expanded upon in her later publications (Wing, 2011, 2014, 2017). Computational thinking has since been embraced as a set of cognitive and social skills that includes abstraction, decomposition, algorithmic thinking, and generalization and is applicable across a wide range of contexts (Abelson & Kong, 2024; Günbatar, 2019; Wing, 2017).

Awareness of CT's importance in 21st-century life and its relevance beyond computer science and STEM continues to grow, especially in GenAI era. Yet, despite broad recognition, the interpretation of CT varies widely among curriculum developers and researchers. A recent systematic review of 81 academic papers revealed diverse definitions of CT, ultimately converging to one characterization of CT as algorithmic problem solving supported by programming concepts (Ezeamuzie & Leung, 2022). This review also emphasized that while CT is firmly embedded within computer science and STEM education, its adaptation to other disciplines, particularly the social sciences and humanities, remains a significant challenge. Developing effective teaching strategies and robust educator training programs has become a critical need aimed at ensuring that all students can access and apply CT skills in meaningful, contextually relevant ways (Hodhod et al., 2016; Yadav, 2017).

This study explores the 4P4CT – a pedagogical approach aimed at cultivating CT skills – among prospective teachers, equipping them to effectively nurture CT skills in their future students across various teaching disciplines and ages. The main concept imparted in the course is computational process, and its application in all school disciplines and across various fields of knowledge is demonstrated. As it turns out, developing scripts of computational processes within disciplinary domains requires preservice teachers to apply CT skills at a deep, domain-specific level, beyond their use in general problem solving or programming. Specifically, defining a computational process inherently involves advanced cognitive abilities related to CT, a demand further reinforced by the need to represent the computational process definition by a script, since the execution of the script, as either a tutorial or a simulation, is expected to reflect the behavior of the relevant entities or concepts within the domain. In the study, we address this topic from a pedagogical perspective by uncovering a taxonomy that categorizes the pedagogical approaches of prospective teachers, as reflected in the scripts they developed for computational processes in diverse knowledge domains. From a practical perspective, this taxonomy can be a valuable tool for designing teaching materials and assessing teachers' CT skills.

2. Background

2.1 Computational Thinking Skills

It is well agreed that students who grow up in today's education system must acquire the skills needed to function in a multi-tasking, multi-faceted, technology-driven, and highly diverse environment. Many of those skills are encompassed in CT (Haseski et al., 2018; ISTE, 2024). Following Papert (1980, 2020), who planted the seeds of the concept in 1980, the discussion on CT was reopened by Wing in 2006 (Seow et al, 2019; Wing, 2011, 2014, 2017), and has since attracted the interest of educators, policymakers, and researchers. Leveraging CT skills so that learners can design solutions to complex problems has become an important objective (Buitrago et al., 2017; Ezeamuzie & Leung, 2022), which is manifested in wide implementation of CT in curricula worldwide (e.g., Bocconi et al., 2022; Boholano, 2017; CSTA, 2017; Dagienė et al., 2024; Irons & Hartnett, 2020; NGSS, 2013; OECD, 2019; Sabitzer et al., 2014; Seow et al., 2019). Common CT skills include abstraction, pattern recognition, generalization, problem decomposition, and algorithmic thinking (Bocconi et al., 2022; Ezeamuzie & Leung, 2022). Computational thinking is also associated with creative thinking (DeSchryver & Yadav, 2015) and critical thinking (Korkmaz et al., 2017). Although CT is generally accepted as an arsenal of problem-solving skills, the concept does not have one, single definition, but rather, a variety of perspectives and interpretations. While some approaches link CT with algorithms and computers (Barr et al., 2011; CSTA, 2017; Denning, 2009; Li et al., 2020; Sabitzer et al., 2014), others highlight the need to detach CT from technology and computers (Chen et al., 2023; Hu, 2011; Yadav et al., 2017). Our approach, which forms the basis for teacher professional development, is that although CT can be applied in non-technology contexts, its application in technological environments achieves another important goal. Thus, the summative assignment given to students in the course, explored in this paper, is the development of computerized simulations.

Recognizing that CT encompasses multiple skills, dimensions, and practices, and that existing evaluation frameworks conceptualize and define these practices in different ways, past educational research has examined a wide range of approaches for assessing CT. Consequently, its assessment draws on a diverse set of constructs, including:

(a) conceptual understanding, e.g., sequences, conditions, iterations, and abstraction (e.g., Adler et al., 2023; Brennan & Resnick, 2012; Grover, 2015);

(b) practices, e.g, decomposition, debugging, and testing (e.g., Brennan & Resnick, 2012; Hylton et al., 2023; Metcalf et al., 2023);

(c) problem-solving competencies, e.g., modeling, simulation, and generalization (e.g., Adler et al., 2023; Cuny et al., 2010; Weintrop et al., 2016); and,

(d) non-cognitive factors, e.g., self-efficacy, beliefs, and perspectives (e.g., Adler et al., 2023; Brennan & Resnick, 2012; Tang et al., 2020).

While the learning outcomes of the course described below may support the assessment of these aspects, the current study focuses on different dimensions, which are defined explicitly later in the research questions. Specifically, we sought to examine the extent to which prospective teachers were able to apply CT skills across diverse disciplinary domains and age levels. In addition, we investigated whether, and how, their pedagogical orientations were reflected in their execution of the course's final assignment – a dynamic simulation of a computational process within a disciplinary context.

2.2 Computational Thinking for All

Several theoretical and practical foundations of computer science (CS) are recognized today as essential knowledge for functioning in all domains of life, not necessarily in connection with CS or programming (Boholano, 2017; Harper, 2018; Lye & Koh, 2014; Wing, 2006, 2011, 2014, 2017). This approach is implemented also in academia (Nichols, 2021; Pollock et al., 2019). From this perspective, CT has been identified as a set of useful skills that can be applied to and enhance learning and teaching across all disciplines (Barr et al., 2011; Lee et al., 2020; Pollock et al., 2019; Seegerer & Romeike, 2018). While there is an ongoing discussion about the nature of CT skills in different fields (Denning & Tedre, 2021), some go as far as to interpret CT as computational literacy that leverages students' skills to improve computational outcomes through the practice of computing (Jacob & Warschauer, 2018). Researchers also highlight the need for targeted professional development for teachers to effectively foster CT skills from both multidisciplinary and interdisciplinary perspectives (Çimsir et al., 2024).

Despite this understanding, it turns out that most of the existing applications of CT in education are limited to STEM fields (Bachtiar, 2023; Hurt et al., 2023). This failure to apply CT in all knowledge domains was revealed in two up-to-date systematic literature reviews. The first reviewed 51 papers and reported that most CT definitions used domain-general definitions, or in other words, did not refer to specific areas of knowledge (Wang et al., 2022). According to this literature review, the most popular instructional model used was problem-based instruction, which is commonly applied in STEM subjects. The second review included 23 papers that focused directly on science classes (Ogegbo & Ramnarain, 2022). The investigation addressed the value of using modelling-based pedagogy to build models of learning, which help students of all ages understand basic ideas, in order to incorporate key CT skills within science instruction. The researchers suggested that educators should deploy effective technology tools to enhance the deductive and inductive teaching of science concepts using a CT framework. Our approach to developing and applying CT skills aligns with this recommendation, emphasizing its relevance across all subjects, not just STEM fields.

2.3 Developing Computational Thinking of Prospective Teachers

The wide implementation of CT curricula has led to extensive educational research across

all ages, from preschool to academia (Lye et al., 2014; Scherer et al., 2019), while considering various pedagogical approaches to the teaching and learning of CT (Hsu et al., 2018). Among these perspectives, the acknowledgement of teachers as key agents is emphasized (OECD, 2019). To help teachers fulfill this role, teacher training strategies are discussed (Armoni, 2019; Hazzan et al., 2025; Hodhod et al., 2016; Mumcu et al., 2023; Ragonis, 2018; Ragonis et al, 2025; Seow et al., 2019; Yadav et al., 2017). As part of this effort, the perceptions and knowledge of pre-service and in-service teachers regarding CT were studied in order to improve professional development programs (Cabrera, 2019; Chang & Peterson, 2018; Lai, 2022; Leung et al., 2022; Ung et al., 2022; Yang et al., 2018; Yilmaz et al., 2018). Findings indicate that teachers acknowledge their need to develop their own CT skills in order to be able to apply them in their teaching, alongside the adaptation of an interdisciplinary perspective (Yilmaz et al., 2018). A very recent study (Macann & Yadav, 2025) that investigated factors influencing elementary teachers' CT learning and CT integration in various fields, asserts that to effectively promote CT integration, professional development processes require a more holistic approach that combines targeted professional development, supportive leadership, and providing teachers with access to appropriate resources. These findings support the teaching methodology that we proposed previously, namely the 4P4CT (Four Pedagogies for CT) (Ragonis et al., 2025). The focus in this framework is on the prospective teachers' acquisition and implementation of CT concepts, that also develop their worldview regarding the necessity of change for the integration of CT in practice. Some of our foundational principles regarding the professional development of teachers were also highlighted as conclusions in a recent systematic literature review (Dong et al., 2024).

3. The Study

3.1 The Research Rationale, Research Target and Research Questions

Our study infrastructure adopts the perspective that CT should be applied across all fields of knowledge and at all ages. Recognizing the importance of working with technology, we emphasize implementation through technological tools. Our effort focuses on cultivating this meaningful perspective among teachers of all disciplines and all ages. Specifically, the target of the research was to explore the outcomes of the course summative assignment (see Section 3.5) in which students use Scratch to create scripts that simulate computational processes, as per their choice, that are grounded in their teaching subject.

Following this approach, we posed the following research questions:

> **RQ1:** Can prospective teachers apply computational processes across various knowledge domains through script development? If so, how?
>
> **RQ2:** What pedagogical approaches do prospective teachers apply when developing scripts of computational processes in their respective fields of knowledge?

3.2 The Research Environment

The research aimed to examine and validate the pedagogical approach applied in an academic CT course for prospective teachers, focusing on the prospective teachers' acquisition of CT concepts and skills, as a foundation for implementation in their future classrooms. The course learning model, namely 4P4CT (Four Pedagogies for CT) (Hazzan et al., 2025; Ragonis, 2018; Ragonis & Hazzan, 2022; Ragonis et al. 2022; Ragonis et al., 2025), implements four learner-centered pedagogies: active learning, project-based learning, product-based learning, and learning in context, which implies constructivism and constructionism (Akin et al., 2023; Harel & Papert, 1991).

3.3 The 4P4CT Pedagogical Framework

The 4P4CT framework is rooted in the well-established learning theories of constructivism and constructionism and encompasses four related pedagogies. Constructivism underscores learners' active engagement in constructing new knowledge based on their existing understanding, promoting involvement and responsibility throughout the learning process (Fosnot, 2013; Piaget, 1973). Constructionism builds upon the constructivist approach, placing greater emphasis on the meaningfulness of learners' understanding when they actively create physical or computational products that represent the concepts they are learning (Harel & Papert, 1991; Papert, 1980, 2020). The significance of these two approaches in acquiring CT is acknowledged in various reports (e.g., Dagienė & Futschek, 2019).

Among the various pedagogies developed based on constructivism and constructionism theories, the 4P4CT framework emphasizes the following four pedagogies:

(a) Active learning. Active learning is an instructional approach that engages students with the learned material through discussions, problem solving, case study analysis, role play, and other methods. In active learning, learners are responsible for their own learning processes and gain opportunities to promote higher-order cognitive skills, such as knowledge application, analysis, and synthesis (Drew & Mackie, 2011).

(b) Project-based learning. A pedagogical approach according to which learners actively develop their knowledge in a process that includes identification and formulation of an authentic issue, question, or problem, search for possible solutions to it, select criteria for choosing among alternative solutions, and develop a solution that solves the problem and meets its requirements and constraints (Dilekli, 2020; Reinholz et al., 2018).

(c) Product-based learning. One possible implementation of the project-based learning approach, which stems from the constructionism theory, emphasizes the idea that people construct new knowledge when they are engaged in constructing meaningful physical products for themselves or for others around them (Ackermann, 1996). In our case, in order to gain meaning during project-based learning, we use

project-based learning in the form of developing a simulation of a computational process in any field of knowledge as per the students' choice.

(d) Context-based learning. A pedagogical approach that emphasizes the idea that knowledge and skills must be learned and built up within a relevant context or domain of life, and not just as a theory (Bennett et al., 2007; Holbrook, 2014; Prins et al., 2018; Stanisavljević et al., 2016; Tal et al., 2021). This pedagogy asserts that both the social context of the learning environment and its concrete context are pivotal to knowledge acquisition and processing (Rose, 2012). Moreover, it is associated with teaching thinking skills (Avargil et al., 2012) and the promotion of motivation (Liem & McInerney, 2020).

The 4P4CT is a conceptual framework for teaching CT to all students in all subject matters. The academic course presented in the following section was designed to fulfil this objective using the 4P4CT framework.

Figure 1 summarizes the three foundations of the course design: (1) The course content is CT; (2) The course pedagogy is the 4P4CT framework; and (3) The target population is all learners – both in school (teachers and school students) and outside of school (the general public).

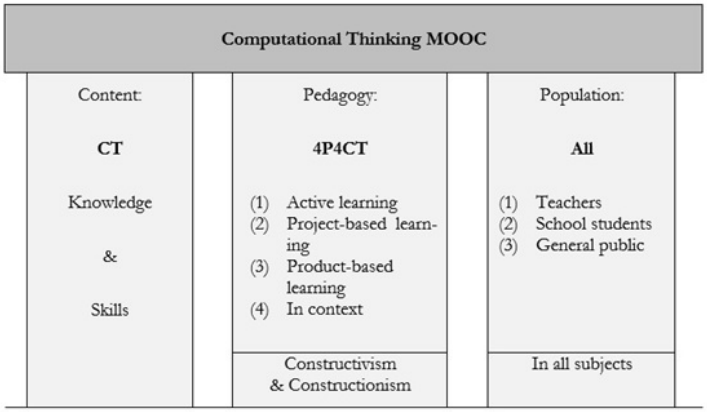


Figure 1. Course design

In the course, the prospective teachers conceptualize CT skills through direct instruction, in addition to practical application thereof. Building on active learning and constructionism (Akin et al., 2023; Harel & Papert, 1991), the course concludes with a summative assignment in which students apply their conceptual and practical knowledge to create simulations of computational processes using Scratch. A computational process is a dynamic visual representation of concepts from any domain of knowledge, realized through an animated script that brings domain-specific entities to life and illustrates their interactions. Scratch was chosen for this task due to its block-based, visual interface and user-friendly

design, making it an ideal educational tool for novice programmers (https://scratch.mit.edu/).

Our pedagogical approach posits that the course infrastructure supports the development and application of CT skills within both the discipline and the computational environment (in coding). The pedagogical rationale behind the summative assignment is that the task fosters the development of CT skills in prospective teachers, while also serving as an example of what they can implement with their future school students. One of the key messages conveyed is that applying CT skills to any knowledge domain enhances the understanding and conceptualization of the domain's content. Additional details about the course structure and content can be found in Ragonis et al. (2025) and in Section 4.1 (specifically, Table 3).

3.4 Course Rationale and Objectives

The course rationale is to develop learners' CT skills while developing their understanding and confidence regarding the use of digital development environments for problem solving. Thus, it is essential to enable course participants to acquire and experience CT in a supportive environment that will maintain their motivation to learn and, in the case of the MOOC, will prevent dropout. The course is therefore based on two main, interwoven principles. The first principle focuses on CT as an object of thought and includes both exposing learners to CT concepts and teaching them CT skills, while combining guided and consistent reflection on their personal understanding of the concepts. The second principle focuses on the implementation of CT by identifying a computational process in some knowledge domain and developing a simulation, in a computerized development environment, that expresses that process.

The course objectives are:

1. Students will spirally develop their understanding of CT as a set of thinking skills.
2. Students will spirally develop their CT skills while applying them to all knowledge domains and to real life.
3. Students will experience an online and dynamic learning environment that enables the application of CT.
4. Students will develop simulations of computational processes across various knowledge domains.
5. Students will become familiar with various applications of CT.
6. If the students are teachers, they will learn how to integrate CT into their teaching of any subject matter at any level.

3.5 The Examined Assignment

One of the messages conveyed/delivered throughout the course was that its main aim is to build the capacity the prospective teachers themselves, both in terms of CT skills and the technological abilities acquired through code development. Therefore, unlike other pedagogical courses that focus primarily on the development of teaching, learning, and assessment models, this course emphasizes students' ability to design simulations of computational processes within their own disciplinary context.

The course concludes with a summative assignment in which the students apply their conceptual and practical knowledge to create simulations of computational processes they choose and define, grounded in their teaching subject, using Scratch. The assignment is performed alone or in groups of 2–4 students. The development of a computational simulation by students is both feasible and serves to contributes to advancing their CT skills, within the discipline and in relation to code development. The assignment design reinforces the core CT skills actively taught throughout the course in preparation for the summative assignment, namely problem decomposition, abstraction, and generalization.

Although the summative assignment was not intended to produce ready-to-use classroom materials, it was made explicit that such an assignment could, per se, be assigned to school students. This pedagogical stance reflects the course's recurring message: school pupils should be regarded not merely as consumers of knowledge, but also as creators of knowledge.

Specifically, the assignment requirements are to submit:

(a) a definition of a computational process within a school subject, highlighting the interaction between relevant concepts;

(b) a dynamic simulation that illustrates the computational process; and

(c) a written reflection on the learning experience.

It is important to note that students are required to submit the computational process they intend to simulate for prior approval by the course instructor. In several cases in our study, multiple iterations were necessary to reach a definition that aligned with the course's conceptualization of a computational process – one that refers to disciplinary concepts and their interrelations rather than to programming structures or the visual rendering of the simulation. Consequently, as mentioned earlier, the study does not evaluate the computational processes themselves but rather the simulations and accompanying pedagogical designs that reflect them.

Since each student is enrolled in a specific teacher education program, the simulation is developed in alignment with this program. It focuses on specific topics drawn from the relevant curriculum, both in terms of the target age group and the subject area. Therefore, the same assignment was given to all course cohorts.

3.6 Population

The research population included 375 prospective teachers studying at two higher education institutions. The institutions were a teachers' college and a science and engineering research university. As mentioned, each student is enrolled in a teacher education program specifically designed for the age group they are preparing to teach—early childhood, elementary, or secondary—and the designated teaching subject. Table 1 shows the distribution of the students' future teaching age groups and fields.

Table 1. Prospective teachers' future teaching age group and teaching fields, N=375

Age group	**%**	**Teaching field**	**%**
Preschool	4%	Exact sciences and life sciences	3%
		Humanities and social sciences	1%
Elementary School	15%	Exact sciences and life sciences	9%
		Humanities and social sciences	6%
Middle and High School	81%	Exact sciences and life sciences	58%
		Humanities and social sciences	23%
Total	100%	Exact sciences and life sciences	70%
		Humanities and social sciences	30%

A little over two-thirds of the students teaching subjects are from the exact and life sciences (263, 70%), while one-third (11 2, 30%) are from the humanities and social sciences.

The study employed a convenience sample, a non-probability sampling method in which participants are selected based on availability. This approach is commonly used in educational research for its practical efficiency, especially when the population is readily reachable (Creswell & Creswell, 2018). Although it may limit generalizability, it is considered appropriate for exploratory studies seeking insights in topics that have not been previously investigated, as is the case in this study (Etikan et al., 2016).

3.7 Data Analysis

The data analysis aimed to assess the prospective teachers' ability to apply the CT skills they acquired throughout the course in the context of their specialized field of knowledge. For this purpose, the analysis focuses on the summative assignment of the course, which, as mentioned above, students completed individually or in teams of 2–4 students. The analysis was conducted on all submissions of the summative assignment received prior to the commencement of the analysis phase, in accordance with the population segmentation described in Table 1. As described in the previous section, students were permitted to proceed with the development of their simulation only after receiving approval that the computational process they had defined was consistent with the course's conceptual

framework. Hence, we did not assess their definitions of computational processes but, as stated in the research questions, examined the pedagogical approaches expressed in the developed simulations. This approach is in line with other research approaches. For example, Cuny et al. (2010) emphasized the CT skills expressed in the thought processes involved in formulating problems, Weintrop et al. (2016) found that computational problem-solving practices are reflected in preparing problems whose solutions are computational solutions, emphasizing the use of modeling practices while developing simulations of scientific phenomena, and Adler et al. (2023) proposed a CT rubric that assesses the transfer of CT skills to new contexts through the criterion of "Generalize to another problem or real-world situation," which involves identifying patterns and applying them to different contexts, corresponding to the Evaluate level of the Revised Bloom's Taxonomy.

The analysis was conducted using the qualitative document analysis method (Lochmiller, 2021; Morgan, 2022), incorporating the thematic analysis method. We analyzed the complete authentic cohort of student tasks, using a customized coding scheme. Our analysis had two main phases: First, we explored the entire dataset to identify the characteristics of the simulations concerning the pedagogical objectives implemented in the course. Then, we formed a coding scheme that focused on addressing the teachers' pedagogical perceptions as reflected in the implementation of the simulations. The resulting coding scheme includes the teachers' future teaching age group, the simulation's field of knowledge, the definition of the computational process, and indicators of the simulation's pedagogical approach: tutorials designed for either learning or assessment, and simulations intended for demonstration only or for both demonstration and investigation. In addition, we examined the user's interaction with the simulation, distinguishing among three possibilities: (1) no interaction at all, (2) interaction that is merely navigational (selection from a given list of alternatives), (3) interaction designed to actively support exploration of the computational process (user-selected inputs). The data was compiled in Excel, where the analyses were subsequently carried out.

The Findings section begins with a presentation of the taxonomy of scripts representing computational processes, as inferred from the characteristics of the pedagogical approaches adopted by the prospective teachers in the simulations they developed. The taxonomy is described by outlining the indicators of the simulations' pedagogical approaches, accompanied by illustrative case studies. Once the taxonomy was established, all scripts were re-examined and classified. Following the qualitative analysis, a quantitative analysis of the relative frequency of script categories was conducted to provide insights from the data. The findings of the quantitative analysis are also presented in this section.

4. Findings

4.1 Taxonomy of the Pedagogies of Computational Processes

The data analysis revealed two main categories with two sub-categories each, as presented in Figure 2. The first category, "Tutorial", includes scripts that do not demonstrate dynamic

computational processes but serve as instructional tools for learning or assessing the said processes. The second category, "Simulation", represents computational processes within a specific domain that emphasize and demonstrate relationships between concepts. The sub-category "for Demonstration" presents simulations that require no user interaction, while "for Demonstration and Exploration" simulations enable users to alter parameters, influencing the simulations' execution and deepening the users' understanding of the processes being simulated.

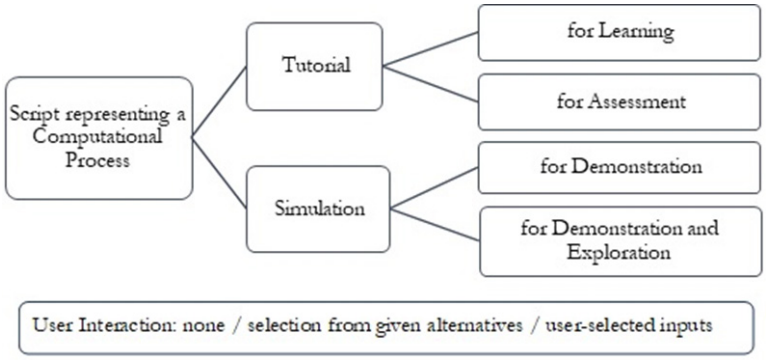


Figure 2. The taxonomy that represents scripts of computational processes

The primary distinction between the "Tutoring" and "Simulation" scripts categories lies in the teaching approaches they reflect. The "Tutoring" category aligns with a traditional teaching model, according to which users engage with multimedia learning software and may optionally be assessed. In contrast, scripts in the "Simulation" involve active domain entities that behave according to domain-specific rules, allowing users to concretize abstract concepts. Simulations that enable exploration aim to further enhance user understanding by allowing interaction and investigation of behavior across different scenarios. As mentioned in the assignment description, the course emphasized simulations over tutorials as its core pedagogical approach and did not address any requirements regarding user interaction or exploration.

Table 2 presents examples of scripts that illustrate the taxonomy: two examples for each sub-category. The initials of the students who developed the simulations are noted in parentheses.

Table 2. Examples of scripts that demonstrate the taxonomy

Type of script	Age group	Subject	Computational Process (CP)	Type of interaction	Screenshot
Tutoring for Learning	Elementary School	Environmental education, Recycling Students: [SL, KG, LN, GG]	*Sorting types of waste into the appropriate recycling bins:* Different types of waste appear on the screen, along with a set of six recycling bins, each designated for a different type of waste. The user clicks on an item of waste, and the script moves it into the appropriate.	Selection	
	Middle and High school	Geography, Map scale Students: [TY, FAO, DL]	*Illustrating map scales:* The script presents an image of a map, displaying the scale bar, and provides a textual explanation of its meaning. The tutorial lacks, however, a visual representation, such as a specific map segment using varied scale measurements.	None	
Tutoring for Assessment	Middle and High school	Mathematics, Probability Student: [LY]	*Probability calculation:* Given a bag with a random number of three types of items, the user calculates the probability of random sampling with and without replacement and receives feedback regarding the accuracy of the answer.	Numerical answer	
	Middle and High school	Citizenship, Civil rights Students: [DS,LY,ER] [DS,LY,ER]	*Decision making regarding various aspects of civil rights when presenting a civil case:* The script presents multiple-choice questions on civil rights alongside alternative answers. The user chooses an answer and receives feedback regarding its correctness.	Selection	
Simulation for Demonstration	Preschool	Nature studies Students: [DR, NL, SO]	*Depicting a bird's life cycle:* The simulation illustrates a bird in the process of building a nest, laying eggs, and hatching chicks.	None	
	Middle and High school	Physics, Wave theory Students: [OY, KO]	*Illustrating the motion of a one-dimensional wave:* The simulation operates according to the laws of wave propagation in space, allowing a choice between wave motion with a fixed end or a free end.	Choice between alternatives (free the wave, fix right side, cat sounds)	

Simulation for Demonstration and Exploration	Middle and High school	Psychology Students: [NS, EA, BL]	*Demonstrating the cognitive development of a toddler according to Piaget, ages 0 through 18 months:* The stages of development are explained and illustrated using scenes depicting toddler behavior.	Option to advance the presentation of the script	
	Middle and High school	Physics, Free throw Students: [MO, GZ, AT]	*Demonstrating the trajectory of a free throw:* The trajectory of a free throw in a baseball game is animated, whereby the position of the ball at successive points along the timeline is calculated based on its previous position.	Choice of Vx and Vy speeds	

Additional examples can be found in Ragonis et al. (2022), and Ragonis et al. (2025). As can be observed, variation exists among the different examples which stems from differences in disciplinary domains and age groups; however, in all cases, the activity represents a simulation of a computational process whose development required the use of computational thinking skills. Furthermore, based on the examples presented in Table 2, we demonstrate the premise that the development of a simulation of a computational process necessarily requires the use of the CT skills emphasized in the course, through two illustrative cases: one developed by prospective teachers preparing to teach in early childhood education, and another by prospective teachers preparing to teach high school physics. These cases were selected to reflect diversity in both disciplinary context and target age group. For both cases, we elaborate on the application of these CT skills in both the subject domain and script development. See Table 3.

Table 3. Manifestation of the three foundational CT principles in the developed simulations

	Decomposition: Breaking a problem into sub-problems	**Abstraction: Omitting or postponing unnecessary details**	**Abstraction: For the purpose of generalization**
Case A: Depicting a bird's life cycle			
In the subject domain	Bird flying back and forth to collect leaves for its nest; egg laying; egg incubation; egg hatching; chick flying	Bird species; specific nest-building materials; duration of incubation	Can be done similarly with twigs instead of leaves

In the script development	Independent script developed for each entity (e.g., bird, leaf, egg)	Simulation timed via message-passing rather than more complex mechanisms	Single leaf entity can be duplicated, with changes made only to its "costume"; each time a leaf is collected (same action) it can be placed in a different nest location
Case B: Demonstrating the trajectory of a free throw			
In the subject domain	Velocity, vertical and horizontal directions	Ignoring friction and weight	At each moment, the ball has a velocity component on both the X and Y axes
In the script development	Independent script developed for each entity (e.g., player, stick, ball)	Only one player; no ball catcher	Repeated reference to ball position on screen; option of showing/hiding velocity arrows; interactive input enabling behavior exploration

4.2 Frequencies of the Taxonomy Categories

Table 4 shows the relative frequencies of the developed scripts according to the defined taxonomy and the types of user interactions involved. The interactions are categorized as either none, navigation (selection from given alternatives), or exploration (user-selected inputs).

Table 4. Frequencies of script types and user interaction categories, N=375

Category and sub-category	Frequency	Type of user interaction		
	%	None	Navigation	Exploration
Tutoring for learning	5%	1%	4%	0%
Tutoring for assessment	3%	0%	3%	0%
Simulation solely for demonstration	46%	26%	13%	7%
Simulation for demonstration and exploration	46%	0%	0%	46%
Total % (N)	100% (375)	27% (100)	20% (74)	53% (201)

As clarified in the summative assignment description, the final assignment was not intended to produce ready-to-use classroom materials, but rather to demonstrate the students' ability to design a computerized simulation that reflects the essence of a computational process. Consequently, no specific pedagogical approach was required or expected. At first glance, one might therefore assume that all submissions would naturally fall under the

category of "Simulation for Demonstration." However, our analysis revealed meaningful variation in the ways students approached and conducted the assignment, reflecting different underlying pedagogical orientations, even though such orientations were not explicitly required.

The data, presented in Table 5 (rows 3 and 4), shows the prevalence of simulations for demonstration, with and without exploration (92%), with exactly half of these simulations incorporating options for exploration, thus supporting active learning by the intended users, namely school pupils. On the other hand, only 8% of the scripts reflect a tutoring pedagogical model, of which over a third were developed specifically for the assessment of user knowledge. It can therefore be said that most of the students implemented the pedagogical approach emphasized in the course: development of dynamic simulations for concepts within different domains. Moreover, almost half (46%) of the simulation developers applied this pedagogical approach by designing simulations that allow for inquiry-based learning of the concepts and their interrelationships.

Table 5 presents the relative frequencies of the developed scripts according to the defined taxonomy and the prospective teachers' future teaching age group and school subject.

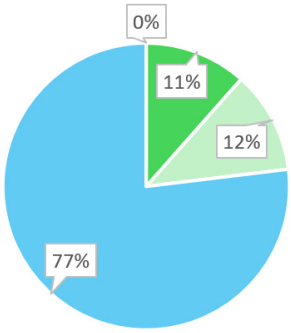
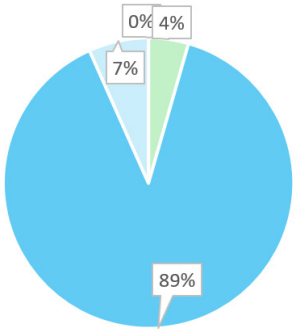
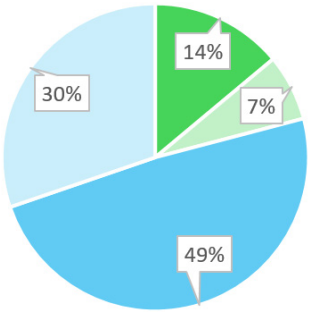
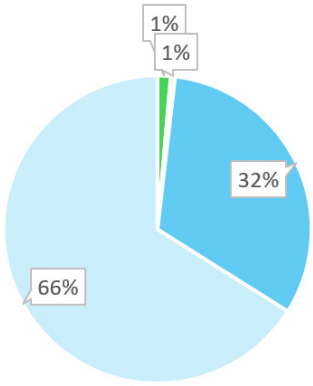
Table 5. Teaching age groups and script topics by taxonomy categories, N=375

Categories and sub-categories	**Prospective teachers' future teaching age group and school subject**					
	Preschool		**Elementary School**		**Secondary and High School**	
	Exact Sciences and Life Sciences	Humanities and Social Sciences	Exact Sciences and Life Sciences	Humanities and Social Sciences	Exact Sciences and Life Sciences	Humanities and Social Sciences
Tutoring for learning	0%	0%	0%	1%	1%	3%
Tutoring for assessment	0%	0%	0%	1%	0%	2%
Simulation solely for demonstration	3%	1%	8%	4%	19%	11%
Simulation for demonstration and exploration	0%	0%	1%	0%	38%	7%
Total % (N)	3% (11)	1% (5)	9% (34)	6% (21)	58% (218)	23% (86)

For all age groups, the "Simulation" category, reflecting simulations with or without exploration, is the most common category. Two main observations can be noted. Firstly, all preschool prospective teachers developed simulations; secondly, out of all analyzed scripts, 75% were simulations developed by middle and high school prospective teachers in both content domains. Furthermore, within the "Simulation for demonstration and

exploration" category, simulations developed by prospective teachers for middle and high school are notably more prevalent, with a relative frequency of 45% across both content domains than those developed for elementary school and preschool. The majority of these simulations focus on exact sciences or life sciences as opposed to the humanities and social sciences - 38% vs. only 7%, respectively. Examining the relationship between student's fields of specialization and the development of inquiry-based simulations, the Pearson Chi-Square test revealed a significant difference ($X2=20.05$; $p<0.001$), with a very strong effect size (Cramer's $V=0.21$). This indicates that inquiry-based simulations were significantly more likely to be developed in the fields of exact and natural sciences compared to those developed in the humanities and social sciences (57%, 30%, respectively). It may therefore be said that the pedagogical worldview of teachers in the exact sciences and life sciences is oriented towards inquiry-based learning.

Table 6. Distribution of categories by age group and by subject area

	Humanities and Social Sciences	Exact Sciences and Life Sciences
	N=26	N=45
Pre-school and Elementary School		
	N=86	N=218
Middle School And High School		
Legend	Tutoring for learning Tutoring for assessment	Simulation for demonstration Simulation for demonstration and exploration

The distribution across the four pedagogical categories presented in Table 6, by age group and subject domain, reveals clear differences between the different prospective teacher populations. Although the majority of simulations submitted aligned with the assignment requirement of demonstrating a computational process (dark blue and light blue), some notable patterns emerged:

- With respect to all subject matters, while only a small percentage (if at all) of prospective pre-school and elementary school teachers developed simulations for demonstration and exploration (light blue), these percentages increased among the prospective middle and high school teacher population, from 0% to 30% in the humanities and social sciences and from 7% to 66% in the exact sciences and life sciences.
- As for the prospective middle and high school teacher population level, a difference is observed between the two groups of subject matters: while 30% developed "for demonstration and exploration" simulations on topics from the humanities and social sciences, 66% created such simulations in the exact sciences and life sciences.
- More simulations incorporated inquiry-based learning (i.e., simulation for demonstration and exploration; denoted in light blue) at the secondary school level compared with the preschool and elementary school level. This is especially notable with respect to the science-related subjects.

These observations suggest that both age group and disciplinary contexts play a meaningful role in shaping the pedagogical orientation adopted during the simulation design.

5. Discussion

Building on the taxonomy developed in the study described in this paper, we answer now the research questions.

Regarding the first research question,

> **RQ1:** Can the concept of computational processes be applied effectively across various fields of knowledge through script development? If so, how?

As described above, the course emphasizes the conceptual understanding of computational processes within any disciplinary domain and how they are defined by the application of the selected cognitive computational skills. Student achievement is therefore reflected in their ability to define such a process within their domain of knowledge and to translate it into a script that represents the relationships among the relevant concepts. Tables 4 and 5 reveal that the course approach effectively supported the students' ability to identify computational processes across various fields of knowledge, including the humanities and social sciences. Table 2 presents some examples of the variety of the developed scripts. Note that although all students developed scripts that addressed a computational process within their disciplinary domain, as demonstrated in Table 3, the assignment requirements specifically emphasized the design of a script that demonstrates a computational process rather than teaches it or assesses the learners' knowledge of it. Thus, 8% of the students did

not fully meet this assignment requirement.

The fact that all students were able to identify and define a computational process within their disciplinary domain, including in the social sciences and humanities, strongly validates the course's pedagogical approach. It is important to emphasize that even in cases where the final product was not dynamic, the ability to accurately define the computational process and successfully develop a corresponding script is itself of significant pedagogical value.

Hence, the answer to RQ1 is: Yes.

Regarding the second research question,

> **RQ2:** What pedagogical approaches do prospective teachers apply when developing scripts of computational processes in their respective fields of knowledge?

The analysis of script execution revealed pedagogical approaches that prospective teachers employ when designing scripts of computational processes within their respective domains of knowledge. These approaches are organized into/encompassed in a taxonomy comprising two main categories, each with two subcategories.

While we might have expected all students to develop a dynamic simulation of the computational process, as the instructions intended, other approaches emerged, forming the basis for the taxonomy presented in Section 4.1. On one end of the spectrum, several students produced learning module-type scripts that addressed the computational process but did not represent it dynamically. This reflects a tendency to adopt the conventional, traditional teaching mode –learning materials (5%) or assessment tools (3%)– despite the specific orientation of the course, which is inspired by the 4P4CT. On the other end of the spectrum, half of the students who did create dynamic simulations developed (46% of the entire population) "for demonstration and exploration" simulations, i.e., extended their simulation by adding an inquiry-oriented dimension. These students approached the assignment through an explicitly constructivist perspective, in accordance with the 4P4CT pedagogical approach. Figure 3 presents this pedagogical interpretation, which explores the taxonomy, highlighting the data presented in Table 4.

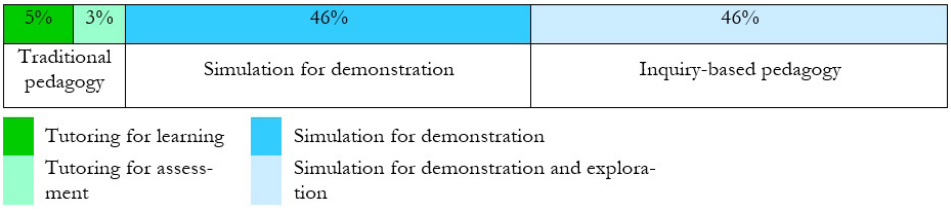


Figure 3. Pedagogical approaches reflected in the simulations

It is important to note that, in all cases, the students applied CT within their domain of knowledge. Even when developing tutorial simulations, the domain content had to be analyzed and structured by applying CT skills such as problem decomposition, abstraction, and generalization. Additionally, since the knowledge was presented as a computerized visual simulation, all students also developed CT skills related to code implementation. The difference between the "Tutorial" approach implementation and the "Demonstration" approach implementation lies in the fact that tutorial represent the content in a more traditional, static manner as opposed to a dynamic demonstration of the relationships between concepts.

6. Summary

This study presents a taxonomy that interprets the assimilation of computational thinking (CT) in the identification by prospective teachers of computational processes in various fields of knowledge. The research findings show that it is indeed feasible to apply CT skills across all fields of knowledge and support the pedagogical implementation model used in the course. Moreover, the findings demonstrate that learners in any domain can construct knowledge through the development of computerized scripts, with computational processes playing a pivotal role in line with the principle of learning in context, as outlined in the 4P4CT pedagogical framework. This approach can be implemented with school pupils, as well as with academic students, including prospective teachers.

The implementation of the 4P4CT pedagogical model is particularly crucial for student populations that are confronted with technology-based code development for the first time. Expanding the integration of CT equips prospective teachers with the ability to cultivate interdisciplinary and problem-solving skills among their diverse learners, promotes equitable access to essential 21st-century competencies.

We conclude by presenting the research limitations, recommendations for teacher preparation programs, and our planned future research.

Research Limitations.

The study employed a convenience sample, a non-probability sampling method in which participants are selected based on availability. This sample yielded an unbalanced population of prospective teachers of exact and life sciences vs. humanities and social sciences. Therefore, although the taxonomy was validated and can be used for different pedagogical purposes, generalization must be applied very carefully. In addition, as in any qualitative data analysis process, biases are possible, although categorizing the students' work into one of the four categories we identified using our coding scheme was very clear cut.

Recommendations for teacher education.

During the course, the prospective teachers faced several challenges:

(a) They found the course challenging, as its primary goal was to cultivate their own thinking skills rather than to simply deliver content;

(b) It took the students time to understand that the terms "computational thinking" and "computational process" do not refer exclusively to calculations, mathematics, or science but rather to problem-solving skills applicable in any field of knowledge;

(c) Consequently, prospective teachers, particularly those preparing to teach preschool or elementary students and those specializing in the humanities and social sciences, struggled at first to identify and define computational processes within their fields. Iterative guidance played a critical role in supporting them during this stage;

(d) Even students with experience in problem solving within the sciences had to recognize the importance of formally conceptualizing CT skills, both for themselves for future incorporation in their teaching.

Based on these observations, as well as on the taxonomy presented in this paper, we recommend incorporating CT skills into all teacher preparation programs, even those dedicated to elementary school teachers, and for all subject matters. This recommendation is largely derived from a recognition of the importance of CT skills in the GenAI era. Although GenAI systems can, among other things, automate coding and problem solving, CT provides the cognitive framework that enables humans to understand, evaluate, and guide these automated processes. CT cultivates skills such as decomposition, abstraction, pattern recognition, and algorithmic design, not merely to facilitate programming, but in order to structure complex problems in any domain. In an era in which AI models generate solutions that often seem correct and sophisticated, CT empowers learners to reason critically about how those solutions are produced, and to identify biases and limitations. Thus, rather than diminishing the relevance of CT, GenAI elevates its importance: understanding computation conceptually becomes one of the central foundations of meaningful human–AI collaboration (Erez et al., 2024).

Future Research.

Future research will evaluate the application of CT skills in the development of students' script codes and explore the potential expansion of the proposed taxonomy. In addition, building directly on the current study's results, future studies will examine the depth of the CT expressed in prospective teachers' code in their respective developed simulations, the application of the 4P4CT model in school classrooms, and accordingly, the effectiveness of the 4P4CT pedagogical model in developing CT competencies among K–12 pupils. A longitudinal study can be conducted to explore the sustainability of teachers' CT practices over time.

References

Abelson H, Kong S C (Eds.). (2024). Computational Thinking Curricula in K–12: International Implementations. MIT Press.

Ackermann, E. (1996). Constructionism in practice: Designing, thinking, and learning in a digital world. Routledge.

Adler, R. F., Hibdon, J., Kim, H., Mayle, S., Pines, B., & Srinivas, S. (2023). Assessing computational thinking across a STEM curriculum for pre-service teachers. Education and Information Technologies, 28(7), 8051–8073.

Akin, V., Braley, E., & Bookman, J. (2023). Modeling Active Learning in Professional Development for Teaching. The Journal of Faculty Development, 37(3), 28–39.

Armoni, M. (2019). Computing in Schools. On the Knowledge of CS Teachers' Educators. ACM Inroads, 10(2), 10–13.

Avargil, S., Herscovitz, O., & Dori, Y. J. (2012). Teaching thinking skills in context-based learning: Teachers' challenges and assessment knowledge. Journal of science education and technology, 21, 207–225.

Bachtiar, A. M. (2023). Computational Thinking: The Essential Skill for being Successful in Knowledge Science Research. International Journal of Informatics, Information System and Computer Engineering (INJIISCOM), 4(1), 11–22.

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

Bennett, J., Lubben, F., & Hogarth, S. (2007). Bringing science to life: A synthesis of the research evidence on the effects of context-based and STS approaches to science teaching. Science education, 91(3), 347–370.

Brennan, K., & Resnick, M. (2012, April). Using artifact-based interviews to study the development of computational thinking in interactive media design. In annual American Educational Research Association meeting, Vancouver, BC, Canada (pp. 1–25).

Bocconi, S., Chioccariello, A., Kampylis, P., Dagienė, V., Wastiau, P., Engelhardt, K., ... & Stupurienė, G. (2022). Reviewing computational thinking in compulsory education: State of play and practices from computing education.

Boholano, H. (2017). Smart social networking: 21st century teaching and learning skills. Research in Pedagogy, 7(1), 21–29.

Buitrago, F., Casallas, R., Hernández, M., Reyes, A., Restrepo, S., & Danies, G. (2017). Changing a generation's way of thinking: Teaching computational thinking through programming. Review of Educational Research, 87(4), 834–860.

Cabrera, L. (2019). Teacher preconceptions of computational thinking: A systematic literature review. Journal of Technology and Teacher Education, 27(3), 305–333.

Chang, Y. H., & Peterson, L. (2018). Pre-service teachers' perceptions of computational thinking. Journal of Technology and Teacher Education, 26(3), 353–374.

Chen, P., Yang, D., Metwally, A. H. S., Lavonen, J., & Wang, X. (2023). Fostering computational thinking through unplugged activities: A systematic literature review and metaanalysis. International Journal of STEM Education, 10(1), 47.

Çimsir, S., Kalelioglu, F., & Gülbahar, Y. (2024). Perceptions of Primary School Teachers on Interdisciplinary Computational Thinking Skills Training. Informatics in Education, 23(3), 507–524.

Creswell, J. W., & Creswell, J. D. (2018). Research Design: Qualitative, Quantitative, and Mixed Methods Approaches (5th ed.). SAGE.

CSTA (2017). Computer science standards. https://www.doe.k12.de.us/cms/lib/

DE01922744/Centricity/Domain/176/CSTA%20Computer%20Science%20Standards%20Revised%202017.pdf

Cuny, J., Snyder, L., & Wing, J. M. (2010). Demystifying computational thinking for non-computer scientists. Unpublished manuscript in progress, referenced in http://www. cs. cmu. edu/~ CompThink/resources/TheLinkWing. pdf.

Dagienė, V., & Futschek, G. (2019). On the Way to Constructionist Learning of Computational Thinking in Regular School Settings. Constructivist Foundations, 14(3), 231–233.

Dagienė, V., Kampylis, P., Giannoutsou, N., Engelhardt, K., Malagoli, C., & Bocconi, S. (2024). Fostering computational thinking in compulsory education in Europe: a multiple case study. Baltic journal of modern computing., 12(2), 189–221.

Denning, P. J. (2009). The profession of IT Beyond computational thinking. Communications of the ACM, 52(6), 28–30.

Denning, P. J., & Tedre, M. (2021). Computational thinking: A disciplinary perspective. W in Education, 20(3), 361.

DeSchryver, M. D., & Yadav, A. (2015). Creative and computational thinking in the context of new literacies: Working with teachers to scaffold complex technology-mediated approaches to teaching and learning. Journal of Technology and Teacher Education, 23(3), 411–431.

Dilekli, Y. (2020). Project-based learning. In Paradigm shifts in 21st century teaching and learning (pp. 53–68). IGI Global.

Dong, W., Li, Y., Sun, L., & Liu, Y. (2024). Developing pre-service teachers' computational thinking: A systematic literature review. International Journal of Technology and Design Education, 34(1), 191–227.

Drew V, Mackie L (2011) Extending the constructs of active learning: implications for teachers' pedagogy and practice. Curriculum Journal, 22(4), 451–467

Etikan, I., Musa, S. A., & Alkassim, R. S. (2016). Comparison of convenience sampling and purposive sampling. American Journal of Theoretical and Applied Statistics, 5(1), 1–4

Erez, Y., Mike, M. and Hazzan, H. (2024). Leveraging Computational Thinking in the Era of Generative AI, Communication of the ACM, https://cacm.acm.org/blogcacm/leveraging-computational-thinking-in-the-era-of-generative-ai/.

Ezeamuzie, N. O., & Leung, J. S. (2022). Computational thinking through an empirical lens: A systematic review of literature. Journal of Educational Computing Research, 60(2), 481–511.

Fosnot, C. T. (2013). Constructivism: Theory, perspectives, and practice. Teachers College Press.

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

Günbatar M (2019). Computational thinking within the context of professional life: Change in CT skill from the viewpoint of teachers. Education and Information Technologies, 1–24. DOI: https://doi.org/10.1007/s10639-019-09919-x

Harel, I. E., & Papert, S. E. (1991). Constructionism. Ablex Publishing.

Harper, B. (2018). Technology and teacher–student interactions: A review of empirical research. Journal of Research on Technology in Education, 50(3), 214–225.

Haseski, H. İ., İlic, U., & Tuğtekin, U. (2018). Defining a new 21st century skill-computational thinking: Concepts and trends. International Education Studies, 11(4), 29–42.

Hazzan, O., Ragonis, N., & Lapidot, T. (2025). Computational Thinking. Guide to teaching computer science: An activity–based approach – chapter 5 (4rd edition). London, UK: Springer.

Hodhod, R., Khan, S., Kurt-Peker, Y., & Ray, L. (2016, February). Training teachers to integrate computational thinking into K-12 teaching. In Proceedings of the 47th ACM Technical Symposium on Computing Science Education (pp. 156–157)

Holbrook J (2014) A context-based approach to science teaching. Journal of Baltic Science Education, 13(2), 152–154

Hsu, T. C., Chang, S. C., & Hung, Y. T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. Computers & Education, 126, 296–310.

Hu, C. (2011, June). Computational thinking: what it might mean and what we might do about it. In Proceedings of the 16th annual joint conference on Innovation and technology in computer science education (pp. 223–227).

Hurt, T., Greenwald, E., Allan, S., Cannady, M. A., Krakowski, A., Brodsky, L., ... & Dorph, R. (2023). The computational thinking for science (CT-S) framework: Operationalizing CT-S for K–12 science education researchers and educators. International Journal of STEM Education, 10(1), 1.

Hylton, D., & Sung, S. H., & Ding, X., & Van Vleet, M. J. (2023, June), Board 196: A Framework to Assess Debugging Skills for Computational Thinking in Science and Engineering. In 2023 ASEE Annual Conference & Exposition, Baltimore , Maryland. 10.18260/1–2–42591

Irons, J., & Hartnett, M. (2020). Computational thinking in junior classrooms in New Zealand. Journal of Open, Flexible and Distance Learning, 24(2), 28–42.

ISTE. (2024). ISTE® International Society for Technology in Education, 2024. ISTE Computational Thinking Competencies. iste.org. https://iste.org/standards/computational-thinking-competencies

Jacob, S. R., & Warschauer, M. (2018). Computational thinking and literacy. Journal of Computer Science Integration, 1(1), 1–21. https://doi.org/10.26716/jcsi.2018.01.1.1

Korkmaz, Ö., Çakir, R., & Özden, M. Y. (2017). A validity and reliability study of the computational thinking scales (CTS). Computers in human behavior, 72, 558–569.

Lai, R. P. (2022). Teachers' ontological perspectives of computational thinking and assessment: A text mining approach. Journal of Educational Computing Research, 60(3), 661–695.

Lee, I., Grover, S., Martin, F., Pillai, S., & Malyn-Smith, J. (2020). Computational thinking from a disciplinary perspective: Integrating computational thinking in K-12 science, technology, engineering, and mathematics education. Journal of Science Education and Technology, 29, 1–8.

Leung, L. L., Labadin, J., & Mohamad, F. S. (2022). Computational thinking for teachers: Development of a localised Elearning system. Computers & Education, 177, 104379.

Li, Y., Schoenfeld, A. H., diSessa, A. A., Graesser, A. C., Benson, L. C., English, L. D., & Duschl, R. A. (2020). Computational thinking is more about thinking than computing. Journal for STEM Education Research, 3, 1–18.

Liem, G. A. D., & McInerney, D. M. (2020). Culturally relevant and responsive educational interventions. Promoting Motivation and Learning in Contexts: Sociocultural Perspectives on Educational Interventions, 1.

Lochmiller, C. R. (2021). Conducting thematic analysis with qualitative data. The qualitative report, 26(6), 2029–2044.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through rogramming: What is next for K-12?. Computers in human behavior, 41, 51–61. Macann, V., & Yadav, A. (2025). Factors influencing elementary teacher's CT learning and CT integration. Education and Information Technologies, 1–32.

Macann, V., & Yadav, A. (2025). Factors influencing elementary teacher's CT learning and CT integration. Education and Information Technologies, 1–32.

Metcalf, S. J., Reilly, J. M., Jeon, S., Wang, A., Pyers, A., Brennan, K., & Dede, C. (2023). Assessing computational thinking through the lenses of functionality and computational fluency. In Assessing Computational Thinking (pp. 87–111). Routledge.

Morgan, H. (2022). Conducting a qualitative document analysis. The qualitative report, 27(1), 64–77.

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(3), 3443–3474.

NGSS (2013). Next generation science standards: For states, by states. https://www.nap.edu/catalog/18290/next-generation-science-standards-for-states-by-states

Nichols, J. (2021). Measuring General Education Community College Faculty's Beliefs on Computational Thinking Skills Correlated with Curriculum Integration Strategies in Higher Education (Doctoral dissertation, University of Florida).

OECD (2019). Computer Science and PISA 2021. OECD Education and Skills Today. https://oecdedutoday.com/computer-science-and-pisa-2021/

Ogegbo, A. A., & Ramnarain, U. (2022). A systematic review of computational thinking in science classrooms. Studies in Science Education, 58(2), 203–230.

Papert, S. (1980). Children, computers, and powerful ideas (Vol. 10, pp. 978–3). Eugene, OR, USA: Harvester.

Papert, S. A. (2020). Mindstorms: Children, computers, and powerful ideas. Basic books.

Piaget, J. (1973). To Understand is to Invent: The Future of Education. Penguin Books.

Pollock, L., Mouza, C., Guidry, K. R., & Pusecker, K. (2019, February). Infusing computational thinking across disciplines: Reflections & lessons learned. In Proceedings of the 50th ACM Technical Symposium on Computer Science Education (pp. 435–441).

Prins G T, Bulte A M, Pilot A (2018) Designing context-based teaching materials by transforming authentic scientific modelling practices in chemistry. International Journal of Science Education, 40(10), 1108–1135

Ragonis, N. (2018). Computational thinking: Constructing the perceptions of pre–service teachers from various disciplines. In S. Pozdniakov & V. Dagienė (Eds.) Informatics in Schools–Fundamentals of Computer Science and Software Engineering. ISSEP 2018. Lecture Notes in Computer Science, vol 11169, (pp. 167–179). Springer, Cham.

Ragonis, N., & Hazzan, O. (2022). A MOOC on computational thinking for all: pedagogical principles, challenges, and their application. In 2022 International Conference on Computational Science and Computational Intelligence (CSCI), IEEE, 1943–1949. Las Vegas, NV, USA, 14–16 December 2022.

Ragonis, N., Bukai, A., & Hazzan, O. (2022). Selecting examples for CS courses: The case of a

computational thinking MOOC. ACM Inroads, 13(3), 22–28.

Ragonis, N., Rosenberg–Kima R, & Hazzan O. (2025). Computational thinking course for all pre-service K-12 teachers: implementing the four pedagogies for developing computational thinking (4P4CT) framework. Educational Technology Research and Development, 73(1), 301–329.

Reinholz D, Slominski T, French T A, Pazicni S. Rasmussen C, McCoy B (2018) Good Problems within and Across Disciplines. Journal of Research in STEM Education, 4(1), 37–53.

Rose, D. E. (2012). Context-based learning. In Encyclopedia of the sciences of learning (pp. 799-802). Springer, Boston, MA.

Sabitzer, B., Antonitsch, P. K., & Pasterk, S. (2014, November). Informatics concepts for primary education: preparing children for computational thinking. In Proceedings of the 9th Workshop in Primary and Secondary Computing Education (pp. 108–111).

Scherer, R., Siddiq, F., & Sánchez Viveros, B. (2019). The cognitive benefits of learning computer programming: A meta-analysis of transfer effects. Journal of Educational Psychology, 111(5), 764–792.

Seegerer, S., & Romeike, R. (2018, October). Computer science as a fundamental competence for teachers in other disciplines. In Proceedings of the 13th Workshop in Primary and Secondary Computing Education (pp. 1–2). https://doi.org/10.1145/3265757.3265787

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. Computational thinking education, 345–361.

Stanisavljević J D, Pejčić M G, Stanisavljević L Ž (2016) The Application of Context-Based Teaching in the Realization of the Program Content "The Decline of Pollinators." Journal of Subject Didactics, 1(1), 51–63

Tal, M., Herscovitz, O., & Dori, Y. J. (2021). Assessing teachers' knowledge: Incorporating context-based learning in chemistry. Chemistry Education Research and Practice, 22(4), 1003–1019.

Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. Computers & Education, 148, 103798.

Ung, L. L., Labadin, J., & Mohamad, F. S. (2022). Computational thinking for teachers: Development of a localised E-learning system. Computers & Education, 177, 104379.

Wang, C., Shen, J., & Chao, J. (2022). Integrating computational thinking in STEM education: A literature review. International Journal of Science and Mathematics Education, 20(8), 1949–1972.

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.

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

Wing, J. (2011). Research notebook: Computational thinking—What and why. The link magazine, 6, 20–23.

Wing, J. M. (2014). Computational thinking benefits society. 40th anniversary blog of social issues in computing, 2014, 26.

Wing, J. M. (2017). Computational thinking's influence on research and education for all. Italian Journal of Educational Technology, 25(2), 7–14.

Yadav, A., Stephenson, C., & Hong, H. (2017). Computational thinking for teacher education. Communications of the ACM, 60(4), 55–62.

Yang, H., Mouza, C., & Pan, Y. (2018). Examining Pre-service Teacher Knowledge Trajectories of Computational Thinking through a Redesigned Educational Technology Course. In Kay, J. and Luckin, R. (Eds.) Rethinking Learning in the Digital Age: Making the Learning Sciences Count, 13th International Conference of the Learning Sciences (ICLS) 2018, Volume 1. London, UK: International Society of the Learning Sciences.

Yılmaz, F. G. K., Yılmaz, R., & Durak, H. Y. (2018). A review on the opinions of teachers about the development of computational thinking skills in K-12. Teaching computational thinking in primary education, 157–181.