




*International Journal of Learning, Teaching and Educational Research*  
Vol. 23, No. 10, pp. 435-452, October 2024  
<https://doi.org/10.26803/ijlter.23.10.21>  
Received Aug 21, 2024; Revised Oct 18, 2024; Accepted Oct 21, 2024

# Investigation of Computational Thinking Skills through Instructional Techniques, Games and Programming Tools

S. Saidah M. Selamat , M. Khalid M. Nasir  and Nor Hafizah Adnan   
Faculty of Education  
Universiti Kebangsaan Malaysia (UKM) Bangi, Malaysia

**Abstract.** Computational thinking (CT) has become a vital approach to problem-solving. This method has been proven successful in various areas, especially science, technology, engineering, mathematics (STEM), and professional development. The integration of CT in education provides a practical approach to solving problems. This concept paper investigated the various uses of instructional techniques and programming tools to enhance CT skills used in previous empirical research. CT consists of four main pillars: decomposition, which refers to breaking down identified problems into smaller parts; pattern recognition, which involves finding occurring patterns that can be seen; abstraction through identification of important details and removing unnecessary information; and algorithm, where a step-by-step procedure is solution is made. By exploring instructional strategies and programming environments, this paper outlines the possible impacts of these instructional strategies, games, and programming environments on improvement in CT skills, thus providing a theoretical basis for future empirical studies. The strength and combination of these approaches provide a comprehensive learning experience. This paper also suggests instructional techniques and programming tools to enhance underexplored CT skills based on research gaps.

**Keywords:** Computational Thinking; Effectiveness; Instructional Techniques; Programming Tools

## 1. Introduction

The definition of Computational Thinking (CT) refers to a systematic approach related to problem-solving that comprises elements such as decomposition, pattern recognition, abstraction, and algorithms to come up with solutions that can solve complex problems across diverse subjects (Wing, 2006; Susanti & Taufik, 2021). It is not just a cognitive skill originating from computer science, but it can be a novel framework in education by providing students with practice in using analytical thinking and creativity to solve real-life problems (Csizmadia et al.,

2019; Butler & Leahy, 2020). CT is derived from the fundamentals of computer science concepts. According to De Jesús and Martínez (2020), CT can be divided into four main components: decomposition, pattern recognition, abstraction, and algorithm, representing an approach to problem-solving. Before designing an algorithm to solve a problem, abstraction is used to eliminate any extraneous information.

CT uses an approach that integrates techniques vital in Science, Technology, Engineering, and Mathematics (STEM) disciplines that require us to demonstrate originality, algorithmic reasoning, analysis, problem-solving, and collaboration (Law et al., 2021). People with excellent CT skills are better at solving complex problems and developing innovative solutions as more sectors use technology (González-González et al., 2021). Therefore, the development of mindset growth is needed to thrive in ever-changing circumstances, and CT must help cultivate just that. Students should prepare to handle global issues and create innovation in the future (Gilchrist et al., 2021). This depends on their ability to improve CT skills, especially considering the increasing need for STEM professions.

As Beecher (2017) referred, CT tactics use strategic problem-solving methods. One approach uses backward working, which involves identifying the results and creating a solution to reach the outcome. This method benefits tasks with clear objectives like puzzle-solving and route planning. One other tactic is through critical thinking. This includes checking the validity and robustness of ideas and conclusions by asking tough questions. This method can identify assumptions and problems with suggested solutions. Beecher stresses that a better grasp of more complex situations can be achieved by applying what has been learned about related difficulties. However, these solutions must also be modified according to their specific context.

Applying CT skills can enhance a better approach to problem-solving, increasing solutions' efficiency and efficacy as stated by Supiarmo et al. (2022), where it acts as a method to solve problems through logical thinking that supports critical evaluation. Kannadass (2023) also pointed out that the connection between CT skills and critical thinking skills was substantial; student's ability to solve problems was connected to these skills. (Kannadass, 2023; Zahroh & Ekawati, 2022). Therefore, this concept paper aims to explore empirical research that has delved into various instructional techniques and programming tools used to enhance CT skills, investigate the effectiveness of different programming environments and tools in promoting CT, and identify gaps and areas for future research in CT education.

## **2. Literature Review**

### **2.1 Theories & Frameworks for Computational Thinking**

Many theories and frameworks are related to computational thinking, and numerous empirical studies have referred to more than one framework. This concept paper discusses three prominent theories used as guidelines for empirical research. Brennan and Resnicks (2012) are some of the most distinguished scholars focusing on expressions, creativity, and empowerment through computational thinking. Their theoretical framework comprises three key components:

computational concepts, practices, and perspectives (Kılıç et al. (2020); Vinnervik & Bungum, 2022). The ideas develop while a programmer completes the code referring to computational concepts. Simultaneously, the skills and habits they obtain while doing tasks such as debugging a programming project are related to computational practices. Finally, computational perspectives refer to the programmer's engagement while handling programming activities (Nayak et al., 2020). This framework uses imagination-based learning, which refers to students' imaginative thinking to solve problems, such as hands-on projects with computational technologies as expressive media for exploring ideas and solving problems (Deng et al., 2022).

Another framework that focuses on CT is Seymour Papert's article that presented the idea of CT in 1980: for students to make mental models when working with computers to enhance learning (Wardana & Pranoto, 2020). According to Papert (2020), an individual's social and educational principles are related to the technological tools they apply, which can influence how they solve a problem. Papert emphasizes the importance of active engagement with computational systems in developing knowledge and comprehension. The learning approach emphasizes learning by doing by utilizing programming languages to create and manipulate artifacts and assessing how the students engage with these languages. It is intended to facilitate learners' tangible investigation of mathematical and computational concepts.

The framework of Papert can be deconstructed into three primary components: constructionism, interdisciplinary learning, and social and affective engagement. Constructionism is related to constructing computational artifacts and underscores the importance of personal creative expression and social engagement in learning (Morado et al., 2021). Interdisciplinary learning refers to utilizing computational activities and programming as instruments to improve cognitive abilities across various fields (Li et al., 2020). Finally, social and affective engagement emphasizes the significance of student engagement in developing computational solutions to promote a better understanding and learning experience (Sharma et al., 2021).

The theoretical framework proposed by Jeannette Wing emphasizes fundamental cognitive skills and problem-solving strategies applicable across disciplines. CT encompasses cognitive skills involved in formulating and executing solutions to problems in ways that may be performed by information-processing agents (Fagerlund et al., 2022). Problem-solving skills are essential for reasoning with data and observations, evaluating hypotheses, depicting mathematical expressions, and working well in teams (Cindikia et al., 2020). The learning approach related to this framework is developing computational literacy and problem-solving skills. Wing's framework demonstrates the universal applicability of CT beyond the field of computer science. The framework CT includes how information is represented in digital form, algorithms, which emphasizes that programs convey algorithms and data in a format that can be implemented on a computer; virtual representations, recognizing that digital systems generate virtual representations of both natural and artificial phenomena;

and communication, which involves understanding how digital systems interact using protocols.

## 2.2 Instructional Techniques for CT Development

Instructional techniques refer to the various methods that can be used to enhance learning outcomes. Investigating effective instructional techniques for enhancing CT is one of the significant studies in education and technology. CT integrates problem-solving strategies using computer science elements like algorithms, abstraction, and logical reasoning, as stated by Santosa (2024). Other disciplines can effectively cultivate these computing skills (Avcu & Er, 2020). By investigating how students' computational/critical thinking can be progressively developed, the educator and researcher are better equipped to assist students in realizing those significant competencies needed in today's digital world (ERGIN, 2023). This could be done through various investigations, from coding hands-on projects to more prominent, transdisciplinary efforts to bridge the theory in computation (Ubaidullah et al., 2020). Students must be exposed to multiple learning strategies in today's classroom, which instill the latest innovations and practical techniques to foster the development of CT (Kastner-Hauler et al., 2022).

### 2.2.1 Game-Based Learning.

Current enhancements and innovations of the new technology have led to the call for game-based learning, where these strategies in still games enhance student learning and engagement (Talib et al., 2017; Chan et al., 2021). This approach has been witnessed to make learners learn through a jolly and captivating means (Ramle et al., 2019). As has been echoed by previous empirical studies, it can be effectively used for learners at various stages, such as kindergarten, primary school, middle school, and even undergraduates, to learn and acquire knowledge willingly. These methods ask learners to solve problems innovatively, cultivating CT skills (Gong & Qiao, 2021). Game-based learning can be carried out either through a plugged approach via platforms like Minecraft, CodeCombat, Penguin Go, and Scratch as presented through previous research (Kutay & Öner, 2022; Saritepeci, 2020; Liu & Jeong, 2022) that integrates the use of technology and digital tools to enhance and facilitate the learning process or through unplugged approaches, whereas students formulate solutions in problem-solving by using algorithms and CT steps through non-digital context t (Mumcu et al., 2023). It is a practical approach that can motivate and engage students in classrooms (Zhang, 2024).

Plugged approaches use technological tools such as computer programs to set game-based learning in motion. Unplugged approaches refer to those using other media unrelated to technology. Liu and Jeong (2022) investigated how in-game cognitive supports would affect students' CT skills using a quasi-experimental design with an educational game, Penguin Go. The testing developed CT for CT skills in near and far transfer tasks. Another study by Ye et al. (2023) focused on how middle school students implemented the computational components of decomposition, pattern recognition, abstraction, and algorithm using a game-based learning platform called MaLT2. This integrates affordances for designing 3D dynamic models with computational capabilities. A study by Kutay and Oner

in 2022 implemented these approaches using Minecraft to investigate whether the integration of the said platform could increase computational thinking.

The research by Kutay and Oner (2020) focused on implementing Minecraft into their coding lessons among middle school students to deliver the lesson interactively and engagingly. They have determined whether such a way of delivering the lesson might increase student coding concepts and CT skills. Their findings showed that students' understanding of coding concepts and CT skills increases significantly once this approach is utilized within the classroom. Another study by Zhang et al. (2023) applied game-based learning to improve CT and to involve parents in the educational process for rural students in China. The study has exploited the collaborative features that game-based learning can offer to help improve CT skills, thus demonstrating how students' engagement by sharing their experience in learning can develop a collaborative learning experience and help parents understand and be more involved in their education (Zhang et al., 2023; Nasir & Ngah, 2022). The research results highlighted how crucial parental involvement supported the development of CT in children. It helped students improve in problem-solving, logical reasoning, and creativity through game-based approaches.

### 2.2.2 *Problem-based learning*

They are introducing STEM into the educational setting to enhance critical thinking, collaboration, and problem-solving so that future graduates can solve real-life problems in the work area (Pujiastuti & Haryadi, 2023; Topsakal et al., 2022). Hidayati (2023) defines problem-based learning (PBL) as a strategy through which students solve authentic problems by being motivated and actively engaging in cooperative and critical thinking activities concerning their studies to ensure that the learners can achieve the in-depth understanding that results from experiencing problems and then solving them. A few scholars have conducted empirical research to evaluate the effectiveness of implementing this approach towards the development of CT and have seen potential impact in increasing learners' skills. This section will elaborate on the methods and findings of scholars such as Kwon et al. (2020) and Ma et al. (2021), who have used these problem-based learning approaches to enhance CT skills among learners.

Kwon et al. (2021) have studied the effect of blending problem-based learning into computer science subjects at the elementary education level. The experiment involved a class of students who attended an elementary school. The results illustrated that problem-based learning is very effective in producing improved skill performance in CT and a change of interest in the learner toward the subject. Implication-wise, they put the students at the center of problem-solving, a scenario meant to encourage the advantage of critical thinking.

In a similar study, Ma et al. (2021), guided by the IGGIA framework (Inquiry, Gathering, Generating, Interpreting, Applying) and a problem-solving approach, the study sought to establish that such methods showed a significant improvement in students' CT skills and, likewise, empirically test the view that there are substantial improvements in students' CT skills and self-efficacy

realized. This approach also invited students to deal with the challenges cooperatively, with each student developing problem-solving skills and gaining confidence in CT assignments.

Therefore, it shows the potential that PBL can effectively develop CT and computer science attitudes among students. Problem-solving experience readies students to build significant competencies for the 21st century: critical thinking, collaboration, and growing self-efficacy. Implementing PBL in computer science education makes students experience meaningful learning processes. It allows them to acquire competencies in complex resolution problems at school and in real life. According to Papert, an individual's social and educational principles are related to the technological tools they apply, which can influence how they solve a problem.

## **2.2 Scaffolding learning approaches**

Scaffolding refers to providing structured support while students learn new skills or consciousnesses. This approach helps each learner acquire the requisite skills and knowledge. According to Zhou et al. (2023), scaffolding strategies within CT's educational system have different objectives. Some may steer students through complex ideas or activities that gradually reduce the assistance as learners' competence grows with increasing independence. This section investigates empirical studies using scaffolding approaches to highlight their effectiveness in computational thinking.

Leveraging activities related to the development of a game among primary school students conducted by Kukul and Çakır (2020), the use of the Kodu Lab Game was implemented to investigate its effectiveness in enhancing CT skills with the assistance of scaffolding approaches. The observed CT skills components were decomposition, abstraction, generalization, testing debugging, and evaluation. The results indicated that at a novice level, students required intense support through scaffolding to enhance their CT skills since prior experience within the group was limited, leading to lower developments in algorithmic thinking, for instance. Scaffolding strategies, especially modeling and peer assistance, proved very effective in helping the students deconstruct the tasks, enhancing their overall learning outcomes. These scaffolding activities also made possible not only the development of technical skills but also the increase in the motivation and self-confidence of students, underlining that tailored support plays a significant role in educational contexts. Kukul & Cakir (2020).

Additionally, scaffolding techniques showed potential in developing abstraction skills faster but minimal progress in generalization testing and debugging skills.). Therefore, this study highlighted the potential of scaffolding techniques to enhance CT skills. A-Mazed is a game developed by Blockly Games. It implements the study by Tivka and Tambouris (2023), which uses scaffolding approaches. The researcher adapted this method to a controlled and experimental group to check on two outcomes: CT skills and attitude toward computational thinking.

Scaffolding approaches apply specific guidance and support to the students concerning their learning needs (Lee et al., 2023). Such design would progress students through complex programming tasks only to focus on acquiring CT skills in algorithmic thinking, abstraction, and debugging (Udvaros et al., 2023). Gradually removing the scaffolds as students became proficient showed the presence of structured support mechanisms that appropriately scaffold CT learning and foster more profound engagement with computational concepts. The results showed that this approach increased CT concepts among them and their problem-solving skills (Zhou et al., 2023).

Thus, previous studies' findings showed the importance of scaffolding as an instructional strategy in CT education. With appropriate guidance and support, students could be encouraged to solve complex problems, developing higher-order thinking to foster a favorable disposition toward computational learning. Scaffolding is, in strategy, embedded within the framework of constructivism, emphasizing active learning and student-centered approaches as learners acquire meaningful CT skills.

### **2.3 Design-Based Learning Approaches**

The instructional approach of design-based learning (DBL) in education means enhancing problem-solving skills, generating creativity, and creating more meaningful learning experiences (Kerimbayev et al., 2023). It is a pedagogical strategy aimed at equipping learners with 21st-century competencies through practical activities that are project-based in nature and promote CT skills (Shin et al., 2021). This paper, therefore, aims to bring several studies to light that describe how DBL fosters CT skills and leads to better student outcomes.

One exemplary study by Saritepeci (2020) investigated the impact of DBL activities on high school students' CT skills. Saritepeci (2020) used a design-based learning method that integrated programming tasks and showed how this integrated approach significantly enhanced students' CT skills. The research found that design-based activities and programming challenges are closely connected; it is argued that learners involved in authentic designing develop skills associated with computational thinking. Jun, Han, and Kim (2017) conducted a similar study on the DBL approach in 2017. They focused on how it affected learners' self-efficacy and CT skills compared to traditional instructional strategies. The results showed that DBL significantly enhanced students' self-efficacy, self-interest, and self-CT. This research used CT skills and encouraged better knowledge about computing ideas, hence putting more emphasis on collaborative and project-based learning.

Students develop critical problem-solving abilities, creativity, and confidence in applying computational principles to real-world contexts by engaging in hands-on design challenges and programming tasks. However, for regions that do not have similar access to digital sources, this approach could be used using low-tech solutions or unplugged solutions such as building blocks for prototyping, therefore creating experiential learning and promoting creativity. DBL is entirely consonant with constructivist learning theories, which connects the fact that students must be actively engaged, explore, and work collaboratively during

learning experiences (Wang et al., 2022; Weng et al., 2022). These are crucial in the development of students' 21st-century skills. The two studies provided empirical findings that integrating activities using this scheme could significantly enhance students' CT skills.

#### **2.4 Collaborative and Constructionist Learning Approaches.**

Any approach utilized for instilling cognitive ability in learners comprises collaborative and constructionist methodologies. According to a constructionist perspective (Papert, 1980), learning indicates that people generate understanding when they involve themselves in practical activities accompanied by tangible artifacts. It seeks to provide students with an opportunity for self-directed problem-solving, which enhances their intellectual level instead of ordinary memorization associated with the schooling system (Harini, 2023; Kim & Seo, 2021). In contrast, collaborative learning modes allow students to directly engage with their peers, instructors, and course materials to construct knowledge. It, therefore, capitalizes on socialization to enhance learning achievement through teamwork, opinion sharing, and knowledge building from other people's experiences. For instance, a study by Girvan and Savage (2019) identified that students involved with problem-based learning performed better in solving problems than those who integrated traditional methods. This proves the efficiency of collaborative approaches.

According to Saritepeci (2020), design-based learning activities are the constructionist approaches to enhancing CT among high school students. During these hands-on projects and programming tasks, students actively construct solutions for problems and work in teams to apply computational concepts that relate to challenges in the real world (Harlizius-Klück & McLean, 2021). Students gain a deeper understanding of CT principles while learning key communication and teamwork skills by working in groups. This approach fosters a constructivist approach through which students build their comprehension of the computational concepts and deeply understand the subject matter (Guaman-Quintanilla et al., 2022).

Herro et al. (2021) have identified that the activities that are conducted in collaboration significantly enhance CT. Students, during such activity actively and independently develop comprehension of computational ideas through the creation and iterative testing of their ideas with their codes. Thus, these approaches provide them with problem-solving skills along with creativity. The underlying attribute of such an approach is creation; hence, learning is collaborative and experiential. Therefore, these constructionist and cooperative learning activities effectively build computational skills and improve teacher education. Educators can provide significantly different classroom learning experiences by integrating these methods (Saad & Zainudin, 2022; Morado et al., 2021).

#### **2.5 Programming Environments and Tools**

Previous empirical studies have investigated various programming environments and tools to enhance CT skills (Cetin & Otu, 2023; Dikkartin et al., 2022; Moon et al., 2022; Piedade & Dorotea, 2023) that focus on 12 students and undergrad students. The previous researchers have implemented several tools and



environments in the educational setting, including object-based, text-based, educational games, and virtual reality. This section will investigate what has been studied and the outcome.

### *2.5.1 Block-based Tools and Environments.*

Block-based environments refer to visual blocks that can be used to teach the concepts of programming to students, especially at the early stages of programming introduction for novice learners, where the fundamentals of programming concepts can be learned without having to deal with complex issues such as syntax errors that occur in text-based programming (Lin & Weintrop, 2021). Some standard block-based programming tools, Blockly, eduBlocks, mBlock, and Scratch, are widely used to enhance learning (Chao et al., 2023; Tikva & Tambouris, 2023). However, Scratch is one of the most prominent tools used in research to evaluate CT (Mladenović et al., 2020; Piedade & Dorotea, 2023). Its visual features allow students with no prior knowledge of programming background to learn coding without worrying about syntax errors. Previous studies have applied this tool to teach students coding and investigated its impact on computational thinking. The effectiveness of block-based environments in improving CT abilities among students has been demonstrated by research works like those done by Kutay and Öner (2022) and Piedade and Dorotea (2023). In engaging activities and interactive tutorials, learners can develop problem-solving, algorithmic thinking, and logical reasoning skills needed for computational thinking.

### *2.5.2 Text-based Programming Languages*

Text-based programming languages like Python, JavaScript, and Swift offer a more advanced pathway for intermediate learners to deepen their understanding of coding principles and syntax (Cetin & Otu, 2023; Cheng & Chen, 2021). While these languages may present a steeper learning curve than block-based environments, they provide learners greater flexibility and power to create complex programs. Research studies by Saritepeci (2020) and Cheng and Chen (2021) have explored the impact of text-based programming environments on CT skill development, showcasing how structured curricula and interactive platforms like Swift Playgrounds can enhance logical thinking and reasoning skills among students. Text-based programming languages prepare learners for more advanced programming tasks and foster a deeper understanding of computational concepts by immersing learners in real-world coding scenarios and projects.

## **2.6 Integration with Educational Games and Virtual Reality (VR)**

Minecraft, CodeCombat, and CodeWars are educationally inspired games and virtual reality experiences that support novice, intermediate, and advanced learners and help take advantage of the engagement from games and simulations to produce better learning outcomes (Dabbous et al., 2022; Dabbous, 2023). The tools showed the increased CT skills in a rather engagingly immersive and interactive way. Previous works by Liu and Jeong (2022) and Agbo et al. (2023) opened possibilities for applying these tools to teaching and learning situations. Their findings showed that implementing educational games and virtual reality simulation impacted CT competency. These tools supported the students toward a

better understanding of the concept of coding and problems in an engaging manner through hands-on approaches.

Instead, teachers can integrate game-based activities that include immersive technologies; with this, students will become curious, creatively and critically thoughtful, and problem hands-on solvers and flexible thinkers in our contemporary computer-based environment (Smiderle et al., 2020). Minecraft or CodeCombat are educational games that require students to explore, build, and solve problems in a virtual world; therefore, the learning process is triggered through curiosity and creativity as the student navigates and interacts with the game environment. Besides, challenges that include coding or logical puzzles can establish their critical thinking and problem-solving skills (Wong & Nasir, 2024). Such task difficulty levels can be gradually increased by teachers to eventually render the students flexible thinkers who can adapt to any new diverse problem-solving scenario.

CT education programming settings and tools include trajectories for learners to explore computational ideas, get coding skills, and develop a mindset crucial to triumphing in our contemporary digital society (Jääskä et al., 2021). This ranges from block-based platforms to text-based languages and even includes immersive experiences to engage educators and learners in meaningful practices involving coding principles to deepen an understanding of CT (GUO, 2024). Further research in this field will, without any doubt, pay benefits back to the future of CT education, especially in such innovative teaching methods and technology-enhanced learning environments that have opened a wide range of possibilities and insight into this now-altered scenery of learning computer science.

### **2.7 Effectiveness of different techniques and tools in enhancing CT skills**

The previously conducted research has provided helpful information on how different teaching methods and tools help learners develop CT (CT) skills (Kutay & Öner, 2022; Saritepeci, 2020; Zhang et al., 2023). To achieve this aim, we need to look at different themes and trends from various combinations of methods and resources to get a picture of CT education. Various instructional strategies, such as integrating GBL with PBL, have been reported to be effective (Kwon et al., 2021; Ma et al., 2021). Through authentic problem-solving scenarios incorporating game elements, students can create stimulating and immersive learning experiences promoting critical thinking and creativity. Similarly, CT development has benefited from applying scaffolding strategies alongside design-based learning (DBL) activities (Saritepeci, 2020). Through hands-on design challenges and programming tasks, learners receive learning approaches tailored to their needs, facilitating obtaining CT skills while promoting deeper conceptual understanding. Collaborative and constructionist learning strategies have also been found to enhance CT skills, with collaborative-making activities and constructionist approaches fostering active engagement and peer interaction in the learning process (Herro et al., 2021; Nasir et al., 2018).

A repeating theme across these instructional techniques is the emphasis on engagement immersion and real-world application of CT skills. Through educational games, project-based learning tasks, or collaborative activities,

learners actively engage in meaningful learning experiences that bridge the gap between theory and practice. Moreover, structured support and guidance are essential for effective CT skill development (Tikva & Tambouris, 2023). Techniques such as scaffolding and collaborative learning underscore the importance of educators in providing targeted assistance and feedback to learners, guiding them through challenging concepts and tasks while fostering independence and confidence.

Despite the progress in understanding the effectiveness of various instructional techniques and tools in promoting CT (CT) skills, several significant gaps persist in the literature. Among the literature gap was the long-term effects of these interventions on students' CT skill development. While many studies have demonstrated immediate improvements in CT skills following the implementation of instructional strategies such as game-based learning, problem-based learning, scaffolding, and collaborative learning (Kukul & Çakır, 2020; Herro et al., 2021; Kwon et al., 2021; Liu & Jeong, 2022), there is little research examining the sustainability of these gains over time. For instance, Kukul and Çakır (2020) presented the gain of CT skills after game-based learning; their research had no longitudinally observed results. Herro et al. 2021 identified significant gains in CT skills following instruction in collaborative learning activities, but their design did not involve longitudinal measurement. Kwon et al. 2021 stated enhancements in problem-solving and creativity from problem-based learning; however, the research focused on short-term outcomes. Liu and Jeong (2022) reported higher CT skills using scaffolding techniques but did not track improvements in these skills. Understanding whether these efforts last beyond the initial learning setting is crucial. This helps guide educational methods and ensures that CT skills stay solid and applicable in real-life situations.

Studies have shown that block-based programming languages, game-based environments, and unplugged approaches can enhance CT skills; for instance, studies conducted by Moon, H., Cheon, J., & Kwon, K. (2022) showed Scratch activities among novice learners capable of enhancing CT skills. However, very few findings have been found that relate text-based programming languages like Python, JavaScript, and Swift to fostering skills in CT. Although these languages would be considered essential in the real world of coding, offering even more flexibility to create complex programs, research works are limited to exploring their impact on developing CT skills, especially in comparison to block-based and game-based approaches. Therefore, empirical studies are called to test the extent to which text-based programming languages bear on developing students' CT skills (Hongquan et al., 2021). The findings in this line of research may be expected to provide insights into instructional strategies and aspects of curriculum design, along with learning environments that could better elicit the use of text-based programming languages in the service of building skills in CT (Sun, 2024). Once this research gap is filled, it shall go a long way to assist educators and policymakers in upgrading their efforts to equip students with the required computational skills in the modern digital world.

Previously conducted research provided valuable information regarding how different teaching methods and tools helped learners develop CT skills. The paper

aims to help one achieve the goal by examining emerging themes and trends through instructional strategies and resources such as scaffolding techniques, game-based learning, and collaborative projects in CT education. Other modes of instruction, like integrating game-based learning with problem-based learning (Kutay & Öner, 2022; Tikva & Tambouris, 2023)., have also shown great promise. In this way, students create engaging and immersive learning experiences for the promotion of critical thinking and creativity through problem-solving scenarios that are authentic and enriched by game elements (Piedade & Dorotea, 2023). In the same line of reasoning, scaffolding strategies have been applied fruitfully together with design-based learning activities in CT development (Saritepeci, 2020; Liu & Jeong, 2022). They used project-based design challenges and programming activities among learners to customize learning methods for CT skills and help them achieve a much deeper level of conceptual knowledge (Sun et al., 2021). Collaborative and constructionist learning strategies have also been proven effective in developing CT skills (Cetin & Otu, 2023).

In collaborative making and constructionist approaches, active participation and interaction with peers during learning are further accentuated. One of the common themes in such instructional methods is that CT skills must be applied, deployed, and used in life. For instance, through educational games or project-based learning tasks, learners become self-initiated toward meaningful learning experiences, bridging the gap between theory and practice (Liu & Jeong, 2022). Effective development of CT skills requires structured support and guidance. Techniques supporting scaffolding and collaborative learning provide educators with tools to guide and inform learners. As Tikva & Tambouris (2023) and Fanchamps et al. (2024) explain, while there is considerable advancement regarding how different instructional methods and tools effectively teach CT skills, critical gaps exist in the literature. One of the gaps was the investigation of the long-term effectiveness of these interventions on the development of CT skills in students. Whereas many studies have established short-term gains in the CT skills of students upon deployment of several instructional techniques, including game-based learning, problem-based learning, scaffolding, and collaborative learning, little, if any, consideration has been given to whether such benefits can be sustained (Tikva & Tambouris, 2023; Piedade & Dorotea, 2023; Saritepeci, 2020). In other words, one would be keen to know if such efforts would prove relevant after a couple of initial years or outside the learning environment. It guides education methods and makes the CT skills sound and applicable in real life (Kutay & Öner, 2022). Other research studies have also illustrated the effectiveness of block-based programming

There is, however, a visible lack of literature concerning the application of text-based programming languages, such as Python, JavaScript, and Swift, in the development of CT skills. While these languages form the basis for nearly all current real-world coding scenarios and make the expression of complex programs easier, only a few studies examine their potential influence on the development of CT skills, let alone in direct comparison to block-based and game-based approaches. This brings us to the real need for empirical research on how text-based programming languages can be used to support learners in improving

their CT skills. Therefore, such research may offer beneficial insights into pedagogical strategies, educational design, and learning settings that efficiently integrate text-based programming languages to enhance CT skill development. This will also help educators and policymakers flesh out CT competencies for success in a digital world, thus losing another research gap (Liu & Jeong, 2022; Zhang et al., 2023).

### **3. Conclusions**

Previous empirical investigations have given promising approaches to improving CT in the classroom. The following studies, for example, report how the game-based learning environment, like Minecraft, develops the principles of students' CT: by Kutay and Öner (2022) and Zhang et al. (2023). Similarly, problem-based learning links CT with critical thinking in real-life problems while scaffolding Systematically develops problem-solving skills.

On the other hand, Saritpeci (2020) explains that design-based learning will create or develop the student's creativity and motivation for computational problems because these problems set the students to engage in practical design challenges. Abilities in CT develop by emphasizing dialog and hands-on experience within the approaches of collaborative and constructionist pedagogies. Scratch and Blockly allow easy programming for beginners, but text-based languages such as Python, JavaScript, and Swift add complexity for advanced learners to further deepen logical reasoning skills. Immersive learning through educational games and VR is also growing, whereby the development of skills in CT was highlighted by Liu and Jeong (2022) and Agbo et al. (2023). However, more significant gaps remain, especially regarding the long-term effects and role of text-based programming languages in CT education. Evidence supporting the potential of using visual drag-and-drop tools like Turtle Python against game-based or block-based solutions is still scant. This includes future research on how text-based programming languages can enhance CT skills, including comparisons with block-based and game-based approaches. Long-term studies are called for that follow young people to see if the development of CT skills is sustained over more extended periods and different educational settings. This involves, moreover, determining what resources educators need to teach these programming languages and the related pedagogical strategies effectively. Meeting these research needs will contribute to educators and policymakers putting in place more efficient instructional strategies that develop strong CT skills; such students would then be better prepared to succeed in the digital world.

### **4. Implications in Education and Future Recommendations**

The identification of previous studies and gaps mentioned in the literature review inquires about the knowledge of longitudinal effects due to instructional interventions towards the development of students' CT skills, which has been viewed as educationally significant. Therefore, the proposed study is concerned with determining how different instructional strategies affect the CT skills of students over time, mainly through text-based programming languages. It is imperative that educators, during their design of interventions aimed at informing CT, should focus on sustainable types of interventions so that the immediate gains among students translate to continuous improvement of CT

skills to sustain them into the future for further academic or professional advancement.

The proposed empirical research investigation's methodological framework will efficiently avoid confirmatory and publication biases. This includes the application of Design and Development Research (DDR), which allows iterative refinement of instructional modules and ensures that findings from this research are based on a set of diversified evidence-based practices. The authors also apply the principles of random sampling and control groups to reduce selection biases, and multiple data sources are used to enhance the reliability of results.

Measurable variables within the study will be students' performance in the CT tasks, which could be measured through standardized assessment and project-based evaluation. These metrics will provide quantifiable data to compare the effectiveness of text-based programming languages in enhancing CT skills. The results will be compared with those of students in block-based languages to gather knowledge on the unique value each possesses.

Most existing studies have targeted block-based programming languages and game-based learning environments for novices (Sun et al., 2021). However, more research is necessary to consider the importance of text-based programming languages to real-world programming. These languages are much more cognitively demanding since they express more profound concepts of syntax and coding principles, which may improve CT skills (Sun, 2023).

Moreover, such an understanding of learning can be technologically enhanced using visual aids in Turtle Python, for example. This will let them graphically interact with students when running code, making it possible to understand how to embed knowledge of algorithmic thinking through experimentation creatively. Examples could be Turtle Python drawing geometric shapes or simulating real-world situations to understand programming principles further and enhance their algorithmic thinking skills.

These oversights, therefore, mean that future studies should bridge the gaps by emphasizing the use of text-based programming languages in the delivery of CT education; such results will inform significant decisions on instructional strategy and curriculum design for educators and policymakers.

## 5. References

- Agbo, F. J., Oyelere, S. S., Suhonen, J., et al. (2023). Design, development, and evaluation of a virtual reality game-based application to support computational thinking. *Education Tech Research Dev*, 71(2), 505–537. <https://doi.org/10.1007/s11423-022-10161-5z>
- Avcu, Y. E., & Er, K. O. (2020). Developing an instructional design for the field of ICT and software for gifted and talented students. *International Journal of Educational Methodology*, 6(1), 161-183. <https://doi.org/10.12973/ijem.6.1.161>
- Beecher, K. (2017). *Computational thinking: A beginner's guide to problem-solving and programming*. BCS Learning & Development Ltd.
- Butler, D., & Leahy, M. (2020). Using classroom practice as “an object to think with” to develop preservice teachers’ understandings of computational thinking. *IFIP*

- Advances in Information and Communication Technology*, 56-65. [https://doi.org/10.1007/978-3-030-59847-1\\_6](https://doi.org/10.1007/978-3-030-59847-1_6)
- Çakır, H., Bahadır, H., & Tüfekci, A. (2021). Assessment of information and communication technology competencies in design-based learning environments. *Designs for Learning*, 13(1), 55-70. <https://doi.org/10.16993/dfl.160>
- Cetin, I., & OTU, T. (2023). The effect of the modality on students' computational thinking, programming attitude, and programming achievement. *International Journal of Computer Science Education in Schools*, 6(2). <https://doi.org/10.21585/ijcses.v6i2.170>
- Chao, Q., Liu, Y., & Zhang, H. (2023). Scratch versus Lego robots: Which engages undergraduates more in programming education? *Journal of Computer Assisted Learning*, 39(3), 935-953. <https://doi.org/10.1111/jcal.12778>
- Cheng, G. M., & Chen, C. P. (2021). Processing Analysis of Swift Playgrounds in a Children's Computational Thinking Course to Learn Programming. *Computers*, 10, 68. <https://doi.org/10.3390/computers10050068>
- Cindikia, M., Achmadi, H. R., Prahani, B. K., & Mahtari, S. (2020). Profile of students' problem solving skills and the implementation of assisted guided inquiry model in senior high school. *Studies in Learning and Teaching*, 1(1), 52-62. <https://doi.org/10.46627/silet.v1i1.22>
- Csizmadia, A., Standl, B., & Waite, J. (2019). Integrating the constructionist learning theory with computational thinking classroom activities. *Informatics in Education*, 18(1), 41-67. <https://doi.org/10.15388/infedu.2019.03>
- Dabbous, M., Kawtharani, A., Fahs, I., Hallal, Z., Shouman, D., Akel, M., ... & Sakr, F. (2022). The role of game-based learning in experiential education: Tool validation, motivation assessment, and outcomes evaluation among a sample of pharmacy students. *Education Sciences*, 12(7), 434. <https://doi.org/10.3390/educsci12070434>
- Dabbous, M., Sakr, F., Safwan, J., Akel, M., Malaeb, D., Rahal, M., ... & Kawtharani, A. (2023). Instructional educational games in pharmacy experiential education: A quasi-experimental assessment of learning outcomes, students' engagement and motivation. *BMC Medical Education*, 23(1). <https://doi.org/10.1186/s12909-023-04742-y>
- De Jesús, S., & Martinez, D. (2020). Applied Computational Thinking with Python: Design algorithmic solutions for complex and challenging real-world problems. *Packt Publishing Ltd.*
- Deng, W., Guo, X., Cheng, W., & Zhang, W. (2022). Embodied design: a framework for teaching practices focused on the early development of computational thinking. *Computer Applications in Engineering Education*, 31(2), 365-375. <https://doi.org/10.1002/cae.22588>
- Dikkartın Övez, F. T., & Acar, İ. G. (2022). The effect of block-based game development activities on the geometry achievement, computational thinking skills and opinions of seventh-grade students. *Journal of Educational Technology & Online Learning*, 5(4), 1106-1121.
- Ergin, H., & Arikan, Y. D. (2023). The effect of project-based learning approach on computational thinking skills and programming self-efficacy beliefs. *AJIT-e: Academic Journal of Information Technology*, 14(55), 320-334. <https://doi.org/10.5824/ajite.2023.04.001.x>
- Fagerlund, J., Leino, K., Kiuru, N., & Niilo-Rämä, M. (2022). Finnish teachers' and students' programming motivation and their role in teaching and learning computational thinking. *Frontiers in Education*, 7. <https://doi.org/10.3389/educ.2022.948783>
- Fanchamps, N., van Gool, E., Slangen, L., et al. (2024). The effect on computational thinking and identified learning aspects: Comparing unplugged smartGames with SRA-Programming with tangible or on-screen output. *Education and*

- Information Technologies*, 29, 2999-3024. <https://doi.org/10.1007/s10639-024-10893-3>
- Gilchrist, P. O., Alexander, A. B., Green, A., Sanders, F. E., Hooker, A. Q., & Reif, D. M. (2021). Development of a pandemic awareness STEM outreach curriculum: Utilizing a computational thinking taxonomy framework. *Education Sciences*, 11(3), 109. <https://doi.org/10.3390/educsci11030109>
- Girvan, C., & Savage, T. (2019). Virtual worlds: A new environment for constructionist learning. *Computers in Human Behavior*, 99, 396-414. <https://doi.org/10.1016/j.chb.2019.03.017>
- Gong, X., & Qiao, A. (2021). The impact of game-based experiential learning on computational thinking. *Modern Education Technology*, 31(11), 119-126.
- González-González, C. S., Caballero-Gil, P., García-Holgado, A., García-Peñalvo, F. J., Molina, J. C. F., Castillo-Olivares, J. M. D., ... & Ramos, S. (2021). Coedu-in project: An inclusive co-educational project for teaching computational thinking and digital skills at early ages. *2021 International Symposium on Computers in Education (SIIE)*. <https://doi.org/10.1109/siie53363.2021.9583648>
- Guaman-Quintanilla, S., Everaert, P., Chiluíza, K., & Valcke, M. (2022). Fostering teamwork through design thinking: Evidence from a multi-actor perspective. *Education Sciences*, 12(4), 279. <https://doi.org/10.3390/educsci12040279>
- Guo, P. (2024). Approaches to enhance game-based teaching literacy for kindergarten major students. *Pacific International Journal*, 6(4), 120-124. <https://doi.org/10.55014/pij.v6i4.489>
- Harini, E., Nurul Islamia, A., Kusumaningrum, B., & Singgih Kuncoro, K. (2023). Effectiveness of e-worksheets on problem-solving skills: A study of students' self-directed learning in the topic of ratios. *International Journal of Mathematics and Mathematics Education*, 150-162. <https://doi.org/10.56855/ijmme.v1i02.333>
- Harlizius-Klück, E., & McLean, A. (2021). The Penelope project: A case study in computational thinking. *Algorithmic and Aesthetic Literacy*, 59-80. <https://doi.org/10.3224/84742428.04>
- Herro, D., Quigley, C., Plank, H., & Abimbade, O. (2021). Understanding students' social interactions during making activities designed to promote computational thinking. *The Journal of Educational Research*, 114(2), 183-195. <https://doi.org/10.1080/00220671.2021.1884824>
- Hidayati, T. and Purwaningsih, D. (2023). The effect of applying a problem-based learning model on students' critical thinking ability in science subjects in grade vs. elementary school. *JPI (Jurnal Pendidikan Indonesia)*, 12(3), 576-585. <https://doi.org/10.23887/jpiundiksha.v12i3.55235>
- Hongquan, B., Wang, X., & Zhao, L. (2021). Effects of the problem-oriented learning model on middle school students' computational thinking skills in a Python course. *Frontiers in Psychology*, 12. <https://doi.org/10.3389/fpsyg.2021.771221>
- Jääskä, E., Aaltonen, K., & Kujala, J. (2021). Game-based learning in project sustainability management education. *Sustainability*, 13(15), 8204. <https://doi.org/10.3390/su13158204>
- Jun, S., Han, S., & Kim, S. (2017). Effect of design-based learning on improving computational thinking. *Behaviour & Information Technology*, 36(1), 43-53. <https://doi.org/10.1080/0144929X.2016.1188415>
- Kannadass, P., Hidayat, R., Siregar, P. S., & Husain, A. P. (2023). Relationship between computational and critical thinking towards modelling competency among pre-service mathematics teachers. *TEM Journal*, 1370-1382. <https://doi.org/10.18421/tem123-17>
- Kastner-Hauler, O., Tengler, K., Sabitzer, B., & Lavicza, Z. (2022). Combined effects of block-based programming and physical computing on primary students'



- computational thinking skills. *Frontiers in Psychology*, 13. <https://doi.org/10.3389/fpsyg.2022.875382>
- Kerimbayev, N., Nurym, N., Akramova, A., & Abdykarimova, S. (2023). Educational robotics: Development of computational thinking in collaborative online learning. *Education and Information Technologies*, 28(11), 14987-15009. <https://doi.org/10.1007/s10639-023-11806-5>
- Kılıç, S., Gökoğlu, S., & Öztürk, M. (2020). A valid and reliable scale for developing programming-oriented computational thinking. *Journal of Educational Computing Research*, 59(2), 257-286. <https://doi.org/10.1177/0735633120964402>
- Kim, J., & Seo, J. (2021). Analysis of structural relationships among college freshmen's self-understanding, self-direction, learning competency, and problem-solving ability. *Journal of Human-Centric Research in Humanities and Social Sciences*, 2(1), 25-34. <https://doi.org/10.21742/jhrhss.2021.2.1.02>
- Kukul, V. and Çakır, R. (2020). Exploring the development of primary school students' computational thinking and 21st century skills through scaffolding: voices from the stakeholders. *International Journal of Computer Science Education in Schools*, 4(2), 36-57. <https://doi.org/10.21585/ijcses.v4i1.84>
- Kutay, E., & Öner, D. (2022). Coding with Minecraft: The Development of Middle School Students' Computational Thinking. *ACM Transactions on Computing Education*, 22(2), 1-19. <https://doi.org/10.1145/3471573>
- Kwon, K., Ottenbreit-Leftwich, A. T., Brush, T. A., et al. (2021). Integration of problem-based learning in elementary computer science education: effects on computational thinking and attitudes. *Education Tech Research Dev*, 69, 2761-2787. <https://doi.org/10.1007/s11423-021-10034-3>
- Law, K. E., Karpudewan, M., & Zaharudin, R. (2021). Computational thinking in stem education among matriculation science students. *Asia Pacific Journal of Educators and Education*, 36(1), 177-194. <https://doi.org/10.21315/apjee2021.36.1.10>
- Lin, Y., & Weintrop, D. (2021). The landscape of block-based programming: Characteristics of block-based environments and how they support the transition to text-based programming. *Journal of Computer Languages*, 67, 101075. <https://doi.org/10.1016/j.cola.2021.101075>
- Liu, Z., & Jeong, A. C. (2022). Connecting learning and playing: the effects of in-game cognitive supports on the development and transfer of computational thinking skills. *Education Tech Research Dev*, 70, 1867-1891. <https://doi.org/10.1007/s11423-022-10145-5>
- Ma, H., Zhao, M., Wang, H., et al. (2021). Promoting pupils' computational thinking skills and self-efficacy: a problem-solving instructional approach. *Education Tech Research Dev*, 69, 1599-1616. <https://doi.org/10.1007/s11423-021-10016-5>
- Mladenović, M., Žanko, Ž., & Mladenović, S. (2020). Impact of used programming language for k-12 students' understanding of the loop concept. *International Journal of Technology Enhanced Learning*, 12(1), 79. <https://doi.org/10.1504/ijtel.2020.10024760>
- Moon, H., Cheon, J., & Kwon, K. (2022). Difficult concepts and practices of computational thinking using block-based programming. *International Journal of Computer Science Education in Schools*, 5(3), 3-16. <https://doi.org/10.21585/ijcses.v5i3.129>
- Nasir, M. K. N., Mansor, A. Z., & Rahman, M. J. A. (2018). Measuring Malaysian online university student social presence in online course offered. *Journal of Advanced Research in Dynamical and Control Systems*, 10(12), 1442-1446.
- Nasir, M. K. M., & Ngah, A. H. (2022). The sustainability of a community of inquiry in online course satisfaction in virtual learning environments in higher education. *Sustainability*, 14(15), 9633. <https://doi.org/10.3390/su14159633>
- Nayak, J., Keane, T., & Seemann, K. (2020). Mapping computational thinking and programming skills using technacy theory. *IFIP Advances in Information and*

- Communication Technology, 24-32. [https://doi.org/10.1007/978-3-030-59847-1\\_3](https://doi.org/10.1007/978-3-030-59847-1_3)
- Papert S A. *Mindstorms: Children, computers, and powerful ideas* [M]. Basic books, 2020
- Piedade, J., & Dorotea, N. (2023). Effects of Scratch-based activities on 4th-grade students' computational thinking skills. *Informatics in Education*, 22(3), 499–523. <https://doi.org/10.15388/infedu.2023.19>
- Santosa, E. B., & Sukmawati, F. (2024). Level of computational thinking and technological literacy skills to improve pre-service teacher learning innovation. *Jurnal Kependidikan: Jurnal Hasil Penelitian Dan Kajian Kepustakaan Di Bidang Pendidikan, Pengajaran Dan Pembelajaran*, 10(1), 338. <https://doi.org/10.33394/jk.v10i1.10872>
- Saritepeci, M. (2020). Developing Computational Thinking Skills of High School Students: Design-Based Learning Activities and Programming Tasks. *Asia-Pacific Edu Res*, 29, 35–54. <https://doi.org/10.1007/s40299-019-00480-2>
- Talib, N., Yassin, S. F. M., & Nasir, M. K. M. (2017). Teaching and Learning Computer Programming Using Gamification and Observation through Action Research. *International Journal of Academic Research in Progressive Education and Development*, 6(3), 1-11. <http://dx.doi.org/10.6007/IJARPED/v6-i3/3045>
- Tikva, C., & Tambouris, E. (2023). The effect of scaffolding programming games and attitudes towards programming on developing Computational Thinking. *Educ Inf Technol*, 28, 6845–6867. <https://doi.org/10.1007/s10639-022-11465-y>
- Ubaidullah, N. H., Mohamed, Z., Hamid, J., & Sulaiman, S. (2020). The use of Delphi technique in validating a teaching and learning model for enhancing students' computational thinking skills. *Universal Journal of Educational Research*, 8(12B), 8201-8213. <https://doi.org/10.13189/ujer.2020.082624>
- Veenman, K., Tolboom, J., & Beekun, O. v. (2022). The relation between computational thinking and logical thinking in the context of robotics education. *Frontiers in Education*, 7. <https://doi.org/10.3389/feduc.2022.956901>
- Vinnervik, P. and Bungum, B. (2022). Computational thinking as part of compulsory education: how is it represented in swedish and norwegian curricula?. *Nordic Studies in Science Education*, 18(3), 384-400. <https://doi.org/10.5617/nordina.9296>
- Wardana, M. A. and Pranoto, I. (2020). Case study: developing computational thinking skill during pandemic situation. *Southeast Asian Mathematics Education Journal*, 10(2), 97-104. <https://doi.org/10.46517/seamej.v10i2.111>
- Wong, K. E., & Nasir, M. K. M. (2024). Exploring the Challenges and Academic Performance of Online Learning Students in Flipped Classrooms: A Case Study. *International Journal of Academic Research in Progressive Education and Development*, 13(3), 1264–1277. <http://dx.doi.org/10.6007/IJARPED/v13-i3/21873>
- Wen, Z. and Li, Y. (2022). A study of the effectiveness of peer collaboration in improving college students' English writing ability. *Proceedings of the 2022 6th International Seminar on Education, Management and Social Sciences (ISEMSS 2022)*, 928-936. [https://doi.org/10.2991/978-2-494069-31-2\\_108](https://doi.org/10.2991/978-2-494069-31-2_108)
- Xiangling Zhang, Ahmed Tlili, Junhong Guo, David Griffiths, Ronghuai Huang, Chee-Kit Looi & Daniel Burgos (2023) Developing rural Chinese children's computational thinking through game-based learning and parental involvement, *The Journal of Educational Research*, 116(1), 17-32, <https://doi.org/10.1080/00220671.2023.2167798>
- Zhang, Y. and Deng, P. (2024). Research on educational evaluation paths under the perspective of game-based learning. *Advances in Education, Humanities and Social Science Research*, 10(1), 28. <https://doi.org/10.56028/aehtsr.10.1.28.2024>