Educational Technology Quarterly, Vol. **2025**, Iss. 3, pp. 330–354 https://doi.org/10.55056/etq.916

Innovative approaches in teaching coding skills to young children: a scoping review

Godwin Pedzisai Dzvapatsva[1], Courage Matobobo[2]

[1]*University of Suffolk, Waterfront Building, 19 Neptune Quay, Ipswich, IP4 1QJ, United Kingdom*
[2]*Walter Sisulu University, Nelson Mandela Drive, Mthatha, Eastern Cape, South Africa*

Abstract. This scoping review explores innovative approaches to teaching coding skills to young children, a crucial competency in our increasingly digital world. For consistency in terminology, the study uses the term young children to describe individuals up to nine years old. The research integrates data from several educational settings and explores approaches such as game-based learning, graphical coding treatments, hands-on robotics, and unplugged activities. The study highlights the effectiveness of tools such as ScratchJr, KIBO robots, Code Karts, and Blink debugging in enhancing young children's computational thinking and problem-solving abilities and increasing interest in coding. A number of studies have shown that coding enhances cognitive development in young children by encouraging computational thinking, which is vital for future careers and other fields such as STEM. Using unplugged approaches promotes communication, motor skills, and socialisation. A key recommendation to educators, which this study suggests, is to design coding programmes to suit the developmental stages of different age groups. There are, however, some limitations to the study, including the use of a single database and the lack of longitudinal studies to estimate long-term impacts. One of the study's limitations is the absence of longitudinal studies to estimate long-term effects. It is important to note that despite these challenges, the findings highlight how innovative teaching approaches could equip young children with the fundamental skills needed to succeed in a technologically advanced society. Future work should focus on collecting empirical data from coding schools for young children.

Keywords: coding skills, computational thinking, unplugged services, innovative teaching approaches, educational technology, young children

1. Introduction and background of the study

In recent years, educational institutions have increasingly prioritised cultivating future-oriented skills at an early age. A popular future-oriented skill for 21st-century workers is coding [12, 90], which has led schools to teach the necessary skills from an early age. The significant investments in infrastructure at preschools, schools, and coding centres highlight the growing emphasis and seriousness placed on teaching young children to code from an early age. What remains lacking are innovative and engaging methods to deliver coding skills to these young children effectively.

Identifying the most effective innovative practices for teaching coding to young children is imperative as coding becomes increasingly relevant in early education. Existing research reflects growing interest in elementary coding [89, 91], with some studies emphasising "learning to code" [104] and others "coding to learn" [37, 67]. However, these studies often differ in focus and do not converge on best practices. While both learning orientations are vital, young children face specific cognitive challenges in grasping coding, an inherently abstract and complex [9]. Therefore,

0000-0001-5534-2647 (G. P. Dzvapatsva); 0000-0002-7125-5989 (C. Matobobo)
g.dzvapatsva@uos.ac.uk (G. P. Dzvapatsva); couragemat@gmail.com (C. Matobobo)
https://www.uos.ac.uk/about/get-in-touch/people-directory/school-of-technology-business-and-arts/godwin-dzvapatsva/ (G. P. Dzvapatsva);
https://scholar.google.com/citations?user=a2Bzs2YAAAAJ (C. Matobobo)

clarity on effective pedagogical approaches is essential. Innovative methods such as computational thinking, visual programming, robotics, and gamified learning have been shown to support age-appropriate cognitive development and digital literacy [13, 21, 62, 77, 91]. Although these methods show promise, their relative effectiveness across diverse contexts and learner profiles remains underexplored. Further, while young children can adapt easily to coding, the complexity of the subject requires thoughtfully designed, creative strategies that align with developmental needs. Despite growing interest in early childhood coding education, there remains a significant gap in the literature concerning specific, innovative teaching approaches that effectively engage young children. This highlights a critical need for further investigation to identify and evaluate methods that foster meaningful and developmentally appropriate coding experiences at this stage.

Given the complexity of teaching coding to young children and the need for targeted, innovative methodologies, educational policies have begun to address this challenge in some countries. For instance, the Ministry of Basic Education in South Africa introduced coding at primary schools to ensure that the young children will have the necessary skills by the time they get to high school and the industry [19, 28]. While many governments have already introduced the coding curriculum in primary and secondary schools [94], the same can not be said for most developing countries.

In order to understand the different strategies used in teaching coding to young children, the study will examine the following key question:

What innovative approaches are emerging as effective strategies for teaching coding skills to young children?

By addressing this key question, the study aims:

1. To identify innovative approaches to teaching young children coding by conducting a systematic literature review.
2. To offer recommendations for educators and policymakers on best practices for integrating innovative coding instruction methods in diverse educational contexts.

Understanding how coding education is implemented across educational levels helps identify effective methods for young children. Coding is increasingly included in early learning curricula, but many studies report that teachers lack adequate teaching skills. For instance, Bjursten, Nilsson and Gumaelius [9] observed that, in some cases, coding was included in the curriculum but had to be taught by "regressed experts" [51] due to a lack of teacher preparation. As noted by Kalogiannakis and Papadakis [38], when programming instruction is delivered by educators who exhibit low confidence, there is a significant risk that young learners may develop unfavourable perceptions of the subject. Adapting established coding methods for young children is challenging. While methods from secondary education have shown effectiveness, their direct application to young learners is limited by developmental differences. Research in developmental psychology confirms that cognitive readiness varies by age, requiring tailored approaches for early learners [76]. For instance, Bjursten, Nilsson and Gumaelius [9] highlights how developmental stages shape young children's capacity to learn coding, and Hill et al. [33] stresses the importance of age-specific design in coding education. Thus, while older students' methods provide useful context, they may not meet the unique needs of younger children. This study addresses the gap in research on innovative, age-appropriate coding approaches for early education.

The rest of the article is structured as follows: section 2 reviews the literature, section 3 outlines the methodology, section 4 presents and discusses the findings, and section 5 concludes with recommendations and future directions.

2. Literature review

This study examines teaching methods for early-stage coding. The literature review explores global childhood coding trends, innovative approaches, and the opportunities and challenges of teaching coding to young children.

2.1. Context of coding

The terms coding and programming are usually used interchangeably, yet in fact, they are different by definition [54]. A notable intersection is that both are found within the computer science domain, with some overlap in each other's sectors. As discussed in Lonati et al. [54], the meaning of coding has shifted from "the narrow part of programming that involves converting the idea for a program into the syntax of a programming language . . . to being used in schools as being equivalent to the whole process of developing a program". Programming, in general, focuses on high-level thinking for full problem-solving and system development. The term "programming" has a broader meaning than "coding." Bell, Duncan and Rainer [6] has examined the evolution of meaning encompassing a broad spectrum of competencies, from technical principles to social practices, including collaborative software development and user-centred design. However, as the discipline has grown, the term coding has become more prevalent, particularly regarding entry-level skills. Additionally, if we adopt Blackwell's [10, p. 218] third definition of programming, "experience in order to develop a more generic understanding of programming activity", we observe that it encompasses our conceptual interpretation of coding. Therefore, since our principal focus in this study is young children, we use the term coding in our methodology.

2.2. Global trends in childhood coding education

This section explores the global trends in childhood coding education. Early pioneers like Alan Turing (1912–1954), who laid the foundation for AI, and Seymour Papert (1928–2016), who developed the Logo programming language, significantly shaped coding education for children. Papert's work in constructivist learning led to global trends in introducing programming to young learners. Building on Papert's vision, the increasing accessibility of technology has made coding more feasible and relevant for young learners. In the 21st century, education systems worldwide have undergone significant changes, influenced by the growing integration of smart devices into young children's lives [16, 114]. As young children are increasingly exposed to technology-rich environments, educators are compelled to adapt to better prepare them for future careers [8, 18], including those in technology. In response, coding education has been progressively introduced, starting as early as kindergarten [9, 33], aligning with a broader push to integrate computing into STEM education. Historically, career-focused education began in high school, allowing students to choose subjects based on their interests or career goals [51]. However, since the early 2000s, coding curricula have become more common for younger students, supporting the development of early problem-solving skills [18, 21] previously absent in primary education, which focused primarily on literacy and numeracy (the "3Rs" of reading, writing, and arithmetic) [27, 33]. As stated by the then Minister of Basic Education for the South African government, Angie Motshekga [22] "The Modern workplace requires learners that can adapt to a fast-changing home and work environments through empowering learners with the skills they develop . . . " and one of the solution is through introducing coding from grade R onwards. Despite these developments, research suggests that educators may limit young children's access to technology as they grapple with other challenges associated with innovation in young children's settings [32, 40]. Other perspectives suggest educators may fail to adapt to technological advances, particularly through the lens of regressed experts [51]. While there have been numerous calls for introducing coding curricula for kids worldwide, which is a positive development, an essential variable of appropriate teaching methods remains underexplored. The existing literature lacks conclusive evidence on which teaching methods are most effective in achieving the Sustainable Development Goal (SDG) of ensuring quality education and fostering a future-ready workforce.

2.3. Innovative teaching methods

Coding is increasingly being taught to young children through various innovative methods, each designed to unlock unique advantages that enhance learning experiences and outcomes. Teachers frequently determine the most suitable methods [14, 41] by leveraging their pedagogical expertise and making informed instructional choices. However, certain countries explicitly mandate the specific methods to be employed when teaching coding to young children. For instance, the South African report on coding curriculum provides a detailed and structured implementation strategy for its coding and robotics curriculum, specifying the necessary software and hardware requirements [22]. This study discusses the following common methods used in teaching coding to young children:

- Play-based learning and coding
- Computational thinking approach
- Gamification in coding education
- Storytelling in teaching coding concepts
- Game-based learning for coding

2.3.1. Play-based learning and coding

A play-based learning approach is a learning approach that introduces young children to coding through interactive and engaging activities that mimic play. According to Vygotsky's seminal theory on play and cognitive development [105], play-based activities are a powerful approach for engaging young children in effectively learning coding concepts. Research by Pollarolo et al. [87], Yu and Roque [113] supports play-based learning and coding for young children as it aligns with their developmental stages and learning styles. This play-based approach's strength is its ability to make abstract coding concepts tangible and comprehensible through interactive and enjoyable activities. Di Lieto et al. [23] demonstrates how learning with robots helps to improve young children's thinking skills. Bers, González-González and Armas-Torres [8], Papadakis [78] show that play-based learning can lay a strong computational foundation in young children, making future learning more manageable and less scary. Coding is incorporated into play-based learning, fostering essential cognitive skills and cultivating early technological interest. Based on the literature reviewed, no conclusive evidence exists about the appropriateness of play-based learning and coding for young children and specific age groups.

2.3.2. Computational thinking approach

In this study, we uphold Hsu, Chang and Hung [34] the definition of computational thinking (CT) as an educational approach designed to teach young children how to think like computers, focusing on key skills such as simplification, embedding, transformation, and simulation. This method emphasises analytical and problem-solving abilities over traditional basics like reading, writing, and arithmetic. The concept of computational thinking was introduced by Seymour Papert in the 1960s [82], but it did not gain traction until Jeannette Wing popularised it in the 1990s [17, 53]. CT aims to develop cognitive skills through aesthetic environments. Most countries, such as the United Kingdom, Germany, and Japan, have already adopted the CT approach, which is vital, especially for Science, Technology, Engineering, Arts, and Mathematics (STEM) courses [52]. Similar approaches have also been adopted in developing countries such as South Africa [22], Zimbabwe [26], and Ethiopia [42] with the introduction of coding in the curriculum, but very little is known about young children in early education levels [56, 95]. Although some critics of CT thinking use the argument that it can not replace traditional cognitive thinking models [20, 60], those in favour of the approach see it as complementing the traditional ones [34, 52, 81] to make the process quicker and accommodate the tech-savvy generation. The fundamental components of CT include decomposition, pattern recognition, abstraction, and algorithm design. These elements are utilised in traditional and CT teaching approaches to enhance problem-solving skills and should be seen as competing approaches. CT remains one of the approaches essential in developing thinking skills using the unplugged approaches [98], necessary for application in future learning or workplaces. Unplugged approaches are very theoretical ways of teaching programming, for instance, using pen, paper, and logic games to represent concepts such as algorithms and loops [24, 92, 100]. Teaching coding through CT allows students to grasp foundational concepts in artificial intelligence (AI), including pattern recognition, data manipulation, and automation [109]. While CT is often introduced to young children as abstract concepts without strong connections to real-world applications, it can potentially cultivate computational fluency, equipping students with essential skills for understanding AI [96].

2.3.3. Gamification in coding for young children

Gamification is an effective strategy for teaching coding to young children. Studies show it boosts engagement, creativity, critical thinking, and problem-solving by leveraging intrinsic motivation [5, 30]. Young children learn best through active, enjoyable experiences, which games naturally provide. For example, Drăgănoiu et al. [25] found that embedding coding in game mechanics helped K-12 students understand math and programming more deeply, with similar benefits observed in university settings [72]. While generally effective, gamification must be well-structured to avoid risks like distraction or game addiction [44]. When designed with clear learning goals, these risks can be mitigated. Game-based coding education nurtures young children's interest in technology and supports skill development, though further research is needed to clarify best practices for early education settings.

2.3.4. Game-based learning

Game-based learning (GBL), closely related to gamification, takes a more hands-on approach aligned with experiential learning theory, using actual games as a medium for learning through doing. Game-based learning is ideal for introducing coding to young children, as it engages them through an interactive environment that simplifies complex programming concepts in a play-like environment [73]. Examples of these include serious games, simulations, and video games. A study in [5] demonstrated the popularity of game-based learning and how it fosters direct acquisition of knowledge/skills through gameplay. Examples of such games can include Minecraft in teaching geometry concepts. Research supports this, showing that students relate coding concepts more intuitively within games. According to Mathrani, Christian and Ponder-Sutton [59], students could connect programming with gameplay. Mathew, Malik and Tawafak [58] found that this approach promotes a clearer understanding of programming fundamentals. Papadakis [79] similarly observed that game-based learning leads to positive outcomes in coding skills for young children. While gamification motivates with rewards, game-based learning immerses students directly in coding tasks, fostering greater engagement. However, many studies focus on young children with prior programming knowledge, suggesting that, for beginners, game-based learning may be best introduced alongside foundational lessons.

2.3.5. Storytelling approach

Storytelling integrates coding lessons within a narrative framework featuring characters, settings, and plots, making learning engaging and relatable for young children. Young children naturally enjoy storytelling and folktales, which can be leveraged to enhance their learning experience. According to Piaget's cognitive development stages, the early stages of sensorimotor, preoperational, concrete operational, and formal operational reasoning, understanding programming should be sequenced accordingly [101]. A good example of a program used to teach coding to young children is ScratchJr. As students progress to middle school, tools like Alice can help them understand and appreciate the foundational concepts of object-oriented programming, fostering their ability to build key programming constructs. By manipulating objects and characters, learners can see immediate visual feedback of their programming decisions, which enhances their understanding and motivation to explore further. Among other techniques, Yang, Ng and Su [110] found that Story-Inspired Robot and Tablet Programming improved computational thinking. Understanding the strengths and limitations of such methods is key to developing effective coding approaches, as discussed next.

2.4. Opportunities and challenges in teaching coding to young children

Pedagogical approaches that yield positive economic and developmental impacts are best developed at early stages [8] and adjusted to suit learners as they grow. Teaching coding to young children from a young age fosters early skill development, which is crucial for future jobs [87]. Although the initial costs of setting up coding programs for young children are high [91], it is essential to note that developing these skills in young children is associated with lower long-term costs and more substantial long-term effects [8, 31]. Given the general trend that curricula usually lag behind the actual skills in demand, it is more reasonable to start developing early skills even though they may need upgrading later. Nevertheless, it is worthwhile to begin developing early skills for the highly tech-centric environment towards which society is heading.

Additionally, teaching early skills such as coding helps students who might later pursue other STEM disciplines [36, 52]. Similarly, teachers are more likely to include such innovative approaches in teaching other science subjects when they use these approaches because of the perceived ease of use [38]. A technology-driven future demands critical thinkers and skilled problem-solvers, traits that can be effectively cultivated in young children through early exposure to approaches like CT. However, while young children are growing up in a technology-rich environment, teaching coding is often more complex than it may initially seem, requiring thoughtful, developmentally appropriate strategies to ensure effective learning.

The complexity is further compounded by the infrastructure challenges associated with introducing innovative coding approaches, which often involve significant costs for hardware, software, and teacher training [29, 91]. Schools in low-income areas often prioritise basic needs such as water, food, and shelter, and coding initiatives seem utopian. In some communities where governments assist, the unequal distribution of resources exacerbates the problem [46]. A common problem, especially witnessed in poor communities, is that trained teachers prefer to work in towns where the standards of living are better, leaving the rural areas without trained teachers who, in most cases, lack confidence and the means to implement innovative approaches in teaching the subject. Further, some early school curricula are packed with traditional subjects focusing on the 3 R's. Without careful planning with curriculum balance and resources, innovative approaches to delivering coding remain a challenge to implement in associated areas. Investment and policy support are needed to close the digital divide [43, 70].

In some regions, a mismatch of priorities between national and provincial governments is often seen in areas where political directives influence the direction of teaching and learning initiatives. Examples can be found in South Africa, where the Western Cape province is governed by the Democratic Alliance. At the same time, the national government is run by the African National Congress. As much as the national governments oversee the general education policies, the provincial governments have autonomy in disbursing funding for the different projects.

2.5. Theoretical framework

The constructionism theory, developed by Seymour Papert, serves as the foundational framework for this study. The theory posits that involving young children in hands-on activities where they create tangible artefacts can significantly enhance their learning experiences [71]. Constructionism maintains that creating things is a powerful means of education and that constructing tangible, shareable objects is the most effective method to develop knowledge [2, 93]. According to the applied theory, young children build their cognitive tools and external realities. Constructionism sheds light on how people's ideas get formed and transformed when expressed through different media, actualised in particular contexts, and worked out by individual minds [83]. The theory centres on young children's discourse with their objects to think with, artefacts, or representations [2]. Constructionism focuses more on the art of learning and the significance of making things in learning to construct new knowledge [1]. The constructionism theory offers a robust framework for analysing innovative approaches to teaching coding skills to young children, emphasising active learning through creating tangible projects that align seamlessly with coding education.

3. Methodology

A scoping literature review was conducted to identify innovative approaches to teaching coding to young children. A scoping review maps the existing literature on a particular topic, identifies key concepts, and summarises the available evidence [68, 85]. This approach is particularly well suited to the topic due to the extensive body of published articles in this field, enabling a comprehensive synthesis of diverse research studies. The Preferred Reporting Items for Systematic Reviews and Meta-Analyses Extension for Scoping Reviews (PRISMA-ScR) standards were adhered to to locate relevant articles [61, 102]. The PRISMA-ScR principles were followed with adaptations to emphasise literature mapping rather than synthesis, as done in systematic reviews, ensuring thoroughness, repeatability, and rigour [74, 75]. The inclusion and exclusion criteria guided article selection.

The following inclusion criteria were used:

- Publication type: Peer-reviewed journal articles, conference proceedings, and book chapters.
- Focus: Studies on coding skills development in children aged 0 to 9.
- Scope: Innovative methods, tools, or approaches to teaching coding skills.
- Publication window: 2014 and 2024 (last 10 years).
- Language: Full text publications in English.

The following exclusion criteria were used:

- Irrelevant scope: Studies do not focus on coding skills.
- Age mismatch: Studies targeting children over 9 years old.

- Outdated research: Publications prior to 2014.
- Language barrier: Non-English publications.
- Access limitations: Publications that are not available online.
- Non-peer-reviewed work: Publications that are not peer-reviewed.

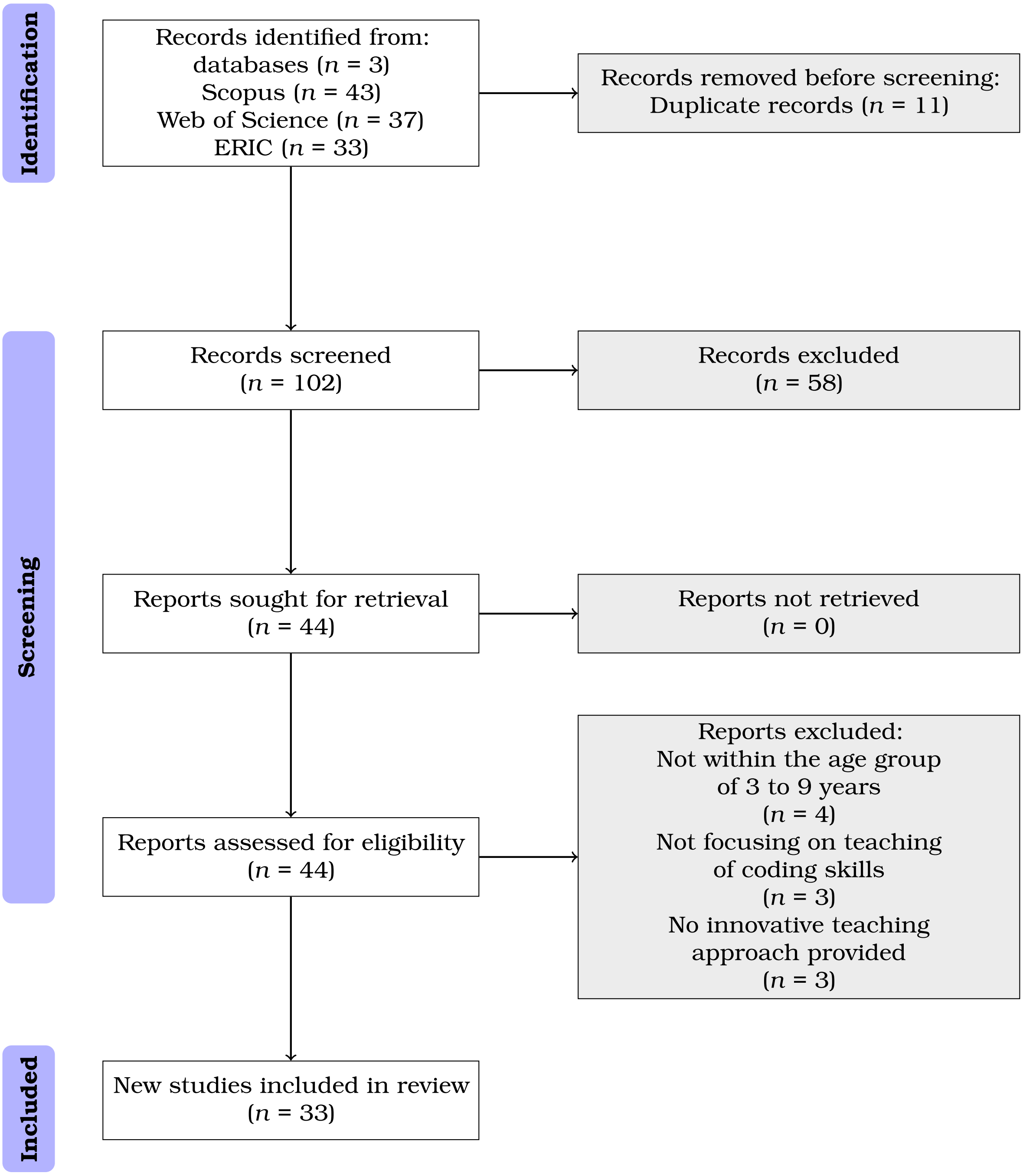

Figure 1: Screening process.

The study used three (3) databases, Scopus, Web of Science (WoS), and the ERIC database, to look for relevant information. The eligibility criteria were studies that focused on young children between the ages of 3 and 9. These three databases enabled the researchers to access relevant and high-quality research since they offer extensive coverage of peer-reviewed literature in various academic areas. The authors used the following search strings:

- Scopus database: *Teaching coding skills to young children*
  The search was conducted on November 6 2024, and forty-three (43) articles were retrieved.
- Web of Science database: *"coding skills" AND children*
  The search was conducted on May 6 2025, and thirty-seven (37) articles were retrieved.
- ERIC database: "coding skills" AND children
  The search was also conducted on May 6, 2025, and thirty-three (33) articles were retrieved.

The study had a total of 113 articles that were retrieved from the three databases. After checking for duplicate articles, 11 were found and excluded. Fifty-eight (58) items were not included in the first screening. The articles' titles and abstracts were the basis for the first screening. Forty-four (44) articles were selected for thorough screening and analysis after articles with titles and abstracts unrelated to the topic were removed. Eleven (11) articles were removed from the thorough analysis due to their failure to fulfil the inclusion criteria. After all the criteria for inclusion and exclusion were satisfied, 33 articles were analysed.

We used the Critical Appraisal Skills Programme (CASP) Checklist for Qualitative Research to assess the methodological quality of the included articles. Each study was evaluated against the 10 CASP criteria, focusing on research aims, methodology, data collection and analysis, ethical considerations, and findings. The authors independently appraised the studies, and discrepancies were resolved through discussion.

4. Findings and discussion

This section presents the research findings in table 1. The table presents the paper's focus, the innovative method incorporated into teaching coding skills, and the points taken from the studies. Table 1 outlines how each article met the inclusion criteria, with analysis guided by key principles of constructionism theory.

Table 1: Characteristics of included articles

#	Authors	Age group	Innovative approach	Focusing on coding skills	Written in English
1	Metin et al. [65]	3-6 years	Design-based digital story (DBDS) program	Yes	Yes
2	Wu, Zheng and Huang [107]	Kindergarten	Programmable robotics	Yes	Yes
3	Leung, Wu and Li [50]	3-6 years	Unplugged digital activities	Yes	Yes
4	Sun, Guo and Zhou [97]	Kindergarten	ScratchJr	Yes	Yes
5	Lee, Yunus and Lee [48]	3-4 years	Programmable robots (KIBO)	Yes	Yes
6	Law et al. [47]	Preschool	ScratchJr	Yes	Yes
7	Yildiz and Çengel [112]	Kindergarten	U-bot starter package	Yes	Yes
8	Papadakis and Kalogiannakis [80]	4-6 years	Bee-Bot educational robotics	Yes	Yes

Continued on next page

Table 1 – continued from previous page

#	Authors	Age group	Innovative approach	Focusing on coding skills	Written in English
9	Relkin, de Ruiter and Bers [90]	5-9 years	KIBO robot and developmentally appropriate coding curriculum	Yes	Yes
10	Kritzer and Green [45]	Preschool	Code-a-pillar, Bee-Bot, and Kubo	Yes	Yes
11	Angeli and Valanides [3]	5-6 years	Bee-Bot with scaffolding strategies	Yes	Yes
12	Rehmat, Ehsan and Cardella [88]	5-7 years	Unplugged and plugged instructional strategies	Yes	Yes
13	Bers [7]	Young children	KIBO robot and ScratchJr	Yes	Yes
14	Bers, González-González and Armas-Torres [8]	3-5 years	KIBO robotics kit	Yes	Yes
15	Marinus et al. [57]	3-6 years	Robot-based tool (Cubetto)	Yes	Yes
16	Kalyenci, Metin and Başaran [39]	5-7 years	Assessment tool for coding skills	Yes	Yes
17	Hu, Li and Gao [35]	6-8 years	Integrated STEM program with AR, VR, and a 3D printer coding	Yes	Yes
18	Yang et al. [111]	7-8 years	Block-based visual programming (CAL-ScratchJr) curriculum	Yes	Yes
19	Louka and Papadakis [55]	4-6 years	ScratchJr integration	Yes	Yes
20	Arfé, Vardanega and Ronconi [4]	6-7 years	Code.org platform	Yes	Yes
21	Lennon et al. [49]	M = 5.16	Digital game-based learning	Yes	Yes
22	Pila et al. [86]	M = 5.15	Tablet apps: Daisy the Dinosaur and Kodable	Yes	Yes
23	Murcia, Cross and Lowe [69]	Kindergarten	Tangible coding device (Cubetto)	Yes	Yes
24	Metin [63]	5 years	Activity-based unplugged coding	Yes	Yes
25	Montuori et al. [66]	Preschoolers	Combined unplugged and educational robotics	Yes	Yes
26	Blake-West, Alrawashdeh and Bers [11]	5-8 years	Creative coding rubric validation	Yes	Yes
27	Yang, Yang and Bers [108]	5-8 years	Computer science curriculum (K-2) with CAL-ScratchJr	Yes	Yes
28	Papadakis [79]	5-7 years	ScratchJr systematic review	Yes	Yes

Continued on next page

Table 1 – continued from previous page

#	Authors	Age group	Innovative approach	Focusing on coding skills	Written in English
29	Papadakis [78]	Preschool and preprimary	Mobile apps	Yes	Yes
30	Pelizzari et al. [84]	4 years	Storytelling and the Cubetto robot	Yes	Yes
31	Metin, Basaran and Kalyenci [64]	5 years	Examining coding skills assessment	Yes	Yes
32	Uğraş et al. [103]	3-8 years	Pedagogical approaches, tools, and frameworks for teaching coding	Yes	Yes
33	Wang et al. [106]	5-8 years	Game-based collaborative learning and cognitive control strategy integration	Yes	Yes

From this study, the following themes were generated:

- Learning through the creation of tangible artefacts
- Role of technology as a tool of construction
- Social interactions and collaborations
- Scaffolding and Teacher Facilitation

4.1. Learning through the creation of tangible artefacts

Creating tangible artefacts provides young learners with hands-on, engaging experiences that support the development of coding and CT skills. The Design-Based Digital Story Program (DBDS) enabled children aged 3–6 to embed CT skills within the design process by creating digital stories, aligning with constructionism's emphasis on learning through meaningful, shareable artefacts [64]. Similarly, Wu, Zheng and Huang [107] found that children used KUBO robots to simulate real-world traffic scenarios, constructing knowledge through project-based, interactive activities. Yildiz and Çengel [112] showed that U-Bot robots facilitated hands-on programming tasks, while Bee-Bot robots have also been widely used to teach coding through tangible interaction [3, 35, 45, 80]. KIBO robots support learning by enabling children to program behaviours like movement and sound response, reinforcing coding principles through physical engagement [8, 8, 48, 90]. Other studies have used Cubetto kits, coding boards, and directional markers to help children understand sequences, loops, and commands through block-based programming and problem-solving activities [57, 63, 84]. These activities included storytelling, navigation tasks, and puzzle-solving, enhancing both cognitive and creative development. Leung, Wu and Li [50] further demonstrated that animation-making using geometric boards and stop-motion techniques supports problem-solving and storytelling through hands-on, project-based learning.

Arfé, Vardanega and Ronconi [4], Law et al. [47], Louka and Papadakis [55], Papadakis [79], Relkin, de Ruiter and Bers [90], Sun, Guo and Zhou [97], Yang et al. [111] examined how young children used ScratchJr to develop coding projects, manipulate digital objects, create sequences, and engage in problem-solving. ScratchJr was noted for its effectiveness in cultivating CT and coding proficiencies within young children [55]. Rehmat, Ehsan and Cardella [88], Uğraş et al. [103] investigated unplugged and plugged activities, such as Puppy Play Space and Computing for the Critters, that can be introduced to young learners as a stand-alone subject or integrated into their curriculum. The Puppy Play Space activity allowed young children to design and build a play area using foam blocks, making abstract concepts like problem decomposition and pattern recognition concrete and accessible. Similarly, Computing for the Critters involved young children creating algorithms to navigate a robot, further reinforcing computational thinking through tangible interactions. Kalyenci, Metin and Başaran [39], Wang et al. [106] investigated unplugged coding (physical cards, story-based activities) and robotic coding (using the "My School Bus" robotic kit) as tangible tools for teaching coding to young children. The study emphasised activity-based learning, where children manipulate physical objects to internalise coding logic. A study explored how young children learn coding skills by creating tangible artefacts in digital games such as Daisy the Dinosaur and Kodable [49]. The study showed that young children who played the game Daisy the Dinosaur more independently learned more coding skills. The results show that young children can learn foundational coding skills using apps when they are appealing to children [86].

The studies discussed in this section demonstrate the power of hands-on, project-based learning in young children's education. By engaging with tangible artefacts, young children develop problem-solving skills, logical reasoning, and a deeper understanding of computational thinking principles, fundamental to the constructionist learning framework.

4.2. Role of technology as a tool of construction

The development of the DBDS program incorporated technological tools such as computers and Web 2.0 applications to facilitate the creation of digital stories [65]. A virtual construction tool like Code.org was used to improve CT and executive functions in children, boosting their planning ability and inhibiting premature reactions [4]. This approach aligns with constructionism theory, which emphasises using technology as a medium for knowledge construction.

The KUBO robotics kit was an interactive play and experimentation tool [107]. It allowed young children to explore concepts through hands-on engagement rather than passive learning. Similarly, Lee, Yunus and Lee [48], Relkin, de Ruiter and Bers [90] demonstrated how young children used tangible programming blocks (e.g., "Forward", "Backward", and "Wait for Clap") to manipulate physical objects and create programs. This hands-on approach made abstract concepts like sequencing and conditional logic more concrete and accessible.

The U-Bot robot functioned similarly, enabling young children to manipulate objects to construct programs physically [112]. Additionally, Leung, Wu and Li [50] found that young children used storyboarding and stop-motion animation techniques to create tangible artefacts, enhancing their CT skills through active engagement.

Law et al. [47], Louka and Papadakis [55], Papadakis [79], Sun, Guo and Zhou [97], Yang et al. [111] highlighted how applications (such as ScratchJr, Kodable, and Lightbot Jr) provided an interactive and visual programming platform that allowed young children to explore coding principles in a graphical environment. Technology was a dynamic tool for conceptual learning that facilitated creative exploration and problem-solving.

Furthermore, in studies by Marinus et al. [57], Murcia, Cross and Lowe [69], the Cubetto robot acted as a technology-mediated learning tool. It enabled young children to experiment with basic coding structures without relying on screen-based programming skills, reinforcing hands-on learning principles within a tangible, interactive environment. Technologies such as AR, VR, 3D printers, coding robotic tools (e.g., "My School Bus" kit) are framed as constructive tools that help children translate abstract commands into physical outcomes (e.g., moving a robot) [35, 39].

These studies demonstrate that technology is a powerful instrument for constructionist learning. Technology allows young children to manipulate digital and physical elements, bridging the gap between abstract coding principles and hands-on, experiential learning, fostering deeper conceptual understanding and problem-solving skills.

4.3. Social interactions and collaborations

The DBDS program fostered a collaborative learning environment where young children worked in groups to design, illustrate, and refine their digital stories [65]. This approach promoted knowledge sharing and the development of a collaborative culture.

The KUBO robotics kit facilitated group-based activities that encouraged young children to collaborate, learn from each other, and co-construct knowledge [107]. Similarly, the U-Bot robots provided opportunities for young children to engage in joint problem-solving and shared learning experiences [112].

A study by Hu, Li and Gao [35], Leung, Wu and Li [50] highlighted collaborative learning in animation-making activities, where young children worked in groups to design and create animations like an "ideal" Ocean Park map and negotiate Bee-Bot routes. Tangible artefacts, like Cubetto robotic kit and coding boards, enabled educators to conduct group activities (treasure hunts, collaborative storytelling), promoted peer learning and teamwork [66, 84]. This group-based approach enabled young children to exchange ideas, support one another, and co-construct knowledge.

Law et al. [47], Sun, Guo and Zhou [97], Yang et al. [111] demonstrated that young children working together on ScratchJr projects could modify and create coding sequences collaboratively through dialogue and mutual understanding, as emphasised in Habermas' communicative action theory. Collaboration helps to immerse young children in authentic learning experiences. This process facilitated idea-sharing and peer feedback, enhancing computational and problem-solving skills through active social engagement.

These studies underscore the importance of collaboration in learning environments, where social interactions enhance individual understanding and contribute to collective knowledge-building and problem-solving.

4.4. Scaffolding and teacher facilitation

In the DBDS program, teachers acted as facilitators, guiding young children through the design process, providing scaffolding, and encouraging reflection [65]. Young children were supported and empowered through their involvement in their projects, allowing them to take ownership of them.

In the KUBO project, an introductory session was conducted to familiarise young children with the robotics kit, followed by child-led exploration with ongoing teacher guidance [107]. This approach balanced structured support with opportunities for independent discovery.

VR/AR and robots such as U-Bot, Bee-Bot, Cubetto, and KIBO enabled teachers to provide structured guidance throughout the programming process, ensuring that young children could progressively build their skills with appropriate levels of support [35, 69, 80, 90, 111, 112]. The use of tangible artefacts such as the Cubetto robotic kit, BeeBot robots, and physical grids enables teachers to provide scaffolding to young children, like gradual progression from unplugged to educational robotics activities and narrative contexts to support understanding [69, 79, 80]. Teachers also use non-digital tools like coding mats to scaffold computational thinking [64].

Sun, Guo and Zhou [97] emphasised the teacher's role as a facilitator in introducing young children to coding concepts. Teachers structured activities and assisted when young children encountered challenges, fostering confidence and independence in coding.

Overall, these studies highlight the importance of teacher facilitation in constructionist learning environments. By offering guidance, structured activities, and scaffolding, teachers help young children develop problem-solving skills, confidence, and autonomy in their learning journey.

4.5. Discussion

This study thoroughly examined the emerging innovative approaches to teaching coding skills to young children. The imperative to integrate coding as a foundational subject in education is unequivocal, as it equips young children with the critical skills necessary to thrive in a rapidly evolving, technology-driven landscape, ensuring they are not merely participants but innovators and leaders in the digital economy of the future. The contemporary educational landscape increasingly emphasises the importance of young children emulating computer-like problem-solving and task completion, underscoring why coding is a valuable skill and essential for navigating and excelling in a technology-centric world.

A key finding from our scoping literature analysis is identifying activities significantly enhancing computational thinking skills and offering actionable insights for optimising educational practices in this domain. This aligns closely with the principles of constructionism, which emphasise that learners achieve deeper and faster comprehension by creating and interacting with tangible artefacts, such as robots or digital art, as demonstrated in unplugged learning environments that promote active engagement and conceptual mastery. These play-based learning environments offer invaluable opportunities for young children to develop cognitive skills early [15, 48, 50, 107]. Some reviewed studies demonstrated that computational thinking extends beyond coding, enhancing understanding in areas such as road safety through storytelling applications like Alice and Scratch. Additionally, young children indirectly learn object-oriented programming by animating objects with actions.

Coding has been shown to significantly foster cognitive development in young children. Unplugged approaches promote socialisation, communication, and motor skills [98]. Through games, young children engage their entire bodies, including the nervous system, as many games involve physical movements. Bers, González-González and Armas-Torres [8] found that Computer-Aided Learning (CAL) facilitates young children's progression through various stages of coding. Our findings highlight that CT provides the foundational framework for young children to navigate coding complexities. Combined with CAL, it effectively guides their progression from basic problem-solving to advanced algorithmic thinking.

It is noteworthy that unplugged approaches are increasingly being implemented globally, a trend that benefits developing countries facing infrastructure challenges and is equally embraced by developed countries. Although critics argue that elements like algorithm design and decomposition are too complex for young children [37, 60, 99], our review indicates that aspects such as pattern recognition are easily taught through play and game-based learning. For instance, applications like Alice and ScratchJ effectively introduce concepts such as looping and conditionals, which are crucial in coding.

A key common concern about unplugged services in teaching coding is their limited realism, making young children struggle to transfer the taught concepts into real-world coding environments. Similarly, approaches such as robots are resource-intensive and often unsuitable for under-resourced environments, where addressing underlying infrastructural challenges is essential to avoid exacerbating equity and access gaps.

Unsurprisingly, innovative approaches such as robots might be destructive, as young learners from underdeveloped environments might become more interested in the mechanics of the tools and miss the intended coding goals to be achieved. The unplugged approach remains powerful and addresses equity easily, while robotics remains ideal for deep engagement, which will require experienced educators.

The study results underscore the progression of cognitive skills through both unplugged and plugged coding approaches. These innovative methods not only make learning coding accessible and engaging for young children but also lay a robust foundation for their cognitive and developmental growth. A compelling conclusion from our scoping review is that the most effective approach to teaching coding to young children prioritises not the mechanics of coding but rather the foundational development of critical skills and a deep understanding of underlying principles, ultimately serving as the bedrock for long-term computational proficiency.

5. Conclusion, recommendations, limitations, and future work

This comprehensive analysis of innovative approaches to teaching coding has highlighted the importance of exposing young children to computational thinking and programming concepts in their early stages of life. The study explored various approaches and tools, including programmable robotics, game-based learning, debugging tools, graphic coding interventions, and unplugged activities. In each of the approaches, traces of the CT approach are recognisable. Further, all these methods align with constructionism theory, which emphasises learning through hands-on experiences and creative problem-solving.

The findings from the study highlighted the effectiveness of these innovative approaches in improving young children's coding skills. More importantly, unplugged activities offer several advantages, particularly in under-resourced environments. Tools such as Blink for Scratch and ScratchJr have provided immediate feedback and fostered an engaging learning environment. Programmable robotics, such as KIBO and Bee-Bot, has significantly benefited from teaching fundamental programming concepts through tangible and playful interactions. Game-based learning apps like Code Karts have successfully promoted classroom engagement and interaction, while unplugged activities have proven valuable in developing computational thinking without the need for technology.

Our findings highlight several pedagogical affordances of unplugged activities, including the concrete learning of abstract concepts, flexibility, increased engagement, and accessibility. However, the study did not address the inclusivity aspect. Since unplugged activities require no digital devices, they offer pedagogical benefits equally in under-resourced and well-resourced environments. As young children engage with these innovative approaches, they also develop valuable soft skills, such as teamwork and perseverance. Further, teachers can adapt and deliver instruction according to age, which is crucial in incremental cognitive development. What did not emerge explicitly is how innovative approaches such as unplugged services would cater to various modalities for young children with developmental challenges. According to constructionism theory, as was expected, innovative approaches provide learners with situational learning opportunities and critical instructional elements.

The practical application of these methods in various educational settings has revealed that age-appropriate, interactive, and engaging approaches are crucial for successfully teaching coding skills to young children. Integrating these methods into the curriculum has improved young children's coding competence and fostered essential skills such as problem-solving, creativity, collaboration, and critical thinking.

Innovative approaches to teaching coding skills to young children offer significant potential to equip the next generation with the necessary skills to thrive in a digital world. By adopting and continually enhancing these techniques, educators may create a dynamic and inclusive learning environment that encourages a lifetime interest in technology and computational thinking. This analysis provides the foundation for future study and development in young children's coding education, opening the door for a generation of kids who will be more creative and technologically savvy.

The study recommends the following:

- A key recommendation to the educators, which this study suggests, is designing coding programmes to suit the developmental stages of the different age groups. Incorporating visual programming tools like block-based coding using Scratch can make learning more engaging and motivating. Text-based programming can be incrementally introduced, and real-world problem-solving projects that enhance critical thinking and prepare students for advanced coding challenges.
- Effective implementation of innovative teaching methods requires continuous teacher training and access to adaptable resources, including unplugged activities like puzzles and role-play, particularly in under-resourced settings where digital tools are limited. Numerous studies have highlighted the critical importance of teacher training in effectively integrating digital tools in education [38, 80].
- The use of game-based learning approaches and storytelling must be advanced to connect young children's interests and develop computational thinking.
- Policymakers should integrate coding into early digital literacy curricula and allocate dedicated funding for teacher training in innovative teaching methods.

5.1. Limitations of the study

The study has the following limitations:

- Only three databases were used in this study.
- The study's findings were collected in various educational settings, such as countries, socio-economic contexts, and school systems. This, in a way, can enrich the study findings and, at the same time, pose a challenge in generalising the findings to all contexts.
- The studies analysed included children of different age ranges and developmental stages, from preschoolers to young children.

Despite the identified limitations, we are convinced that our scoping review helped to map the breadth of the field and laid the foundation for further discussion on innovative approaches.

5.2. Future studies

Future research should prioritise gathering empirical data from coding schools for young children, thereby establishing a solid foundation for advancing the debate on innovative approaches and strengthening the existing literature across various databases.

Future studies could also consider more narrowly defined age groups to provide more precise recommendations and conduct longitudinal studies to assess the long-term impacts of these teaching approaches on young children. Further, future studies should investigate how innovative coding approaches can support underrepresented groups, such as rural children, girls, and children with disabilities, to bridge the inequalities predominantly in these areas. Additionally, research should assess the impact of emerging technologies such as artificial intelligence-driven tools in early coding instruction.

Author contributions: All authors contributed equally to this article. They have read and approved the final manuscript.

Conflicts of interest: The authors declare no conflict of interest.

References

[1] Ackermann, E., 2001. Piaget's Constructivism, Papert's Constructionism: What's the difference? Available from: https://learning.media.mit.edu/content/publications/EA.Piaget%20_%20Papert.pdf.

[2] Ackermann, E.K., 2004. Constructing knowledge and transforming the world. In: M. Tokoro and L. Steels, eds. *A learning zone of one's own: Sharing representations and flow in collaborative learning environments*. Amsterdam, Berlin, Oxford, Tokyo, Washington, DC: IOS Press, vol. 1, chap. 2, pp.15–37. Available from: https://www.researchgate.net/publication/263087010.

[3] Angeli, C. and Valanides, N., 2020. Developing young children's computational thinking with educational robotics: An interaction effect between gender and scaffolding strategy. *Computers in Human Behavior*, 105, p.105954. Available from: https://doi.org/10.1016/j.chb.2019.03.018.

[4] Arfé, B., Vardanega, T. and Ronconi, L., 2020. The effects of coding on children's planning and inhibition skills. *Computers & Education*, 148, p.103807. Available from: https://doi.org/10.1016/j.compedu.2020.103807.

[5] Behnamnia, N., Kamsin, A., Ismail, M.A.B. and Hayati, S.A., 2023. A review of using digital game-based learning for preschoolers. *Journal of Computers in Education*, 10(4), pp.603–636. Available from: https://doi.org/10.1007/s40692-022-00240-0.

[6] Bell, T., Duncan, C. and Rainer, A., 2017. What is coding? In: S. Humble, ed. *Creating the Coding Generation in Primary Schools*. London: Routledge, pp.3–21. Available from: https://doi.org/10.4324/9781315545813-2.

[7] Bers, M.U., 2019. Coding as another language: a pedagogical approach for teaching computer science in early childhood. *Journal of Computers in Education*, 6(4), pp.499–528. Available from: https://doi.org/10.1007/s40692-019-00147-3.

[8] Bers, M.U., González-González, C. and Armas-Torres, M.B., 2019. Coding as a playground: Promoting positive learning experiences in childhood classrooms. *Computers & Education*, 138, pp.130–145. Available from: https://doi.org/10.1016/j.compedu.2019.04.013.

[9] Bjursten, E.L., Nilsson, T. and Gumaelius, L., 2023. Computer programming in primary schools: Swedish Technology Teachers' pedagogical strategies. *International Journal of Technology and Design Education*, 33(4), pp.1345–1368. Available from: https://doi.org/10.1007/s10798-022-09786-7.

[10] Blackwell, A.F., 2002. What is Programming? In: J. Kuljis, L. Baldwin and R. Scoble, eds. *14th annual psychology of programming interest group (ppig)*. pp.204–218. Available from: https://www.ppig.org/files/2002-PPIG-14th-blackwell.pdf.

[11] Blake-West, J., Alrawashdeh, G. and Bers, M., 2024. Validating a Creative Coding Rubric through expressive activities for elementary grades. *Journal of Research on Technology in Education*. Available from: https://doi.org/10.1080/15391523.2024.2398502.

[12] Boz, T. and Allexsaht-Snider, M., 2022. How do elementary school teachers learn coding and robotics? A case study of mediations and conflicts. *Education and Information Technologies*, 27(3), pp.3935–3963. Available from: https://doi.org/10.1007/s10639-021-10736-4.

[13] Çakır, R., Korkmaz, Ö., İdil, Ö. and Uğur Erdoğmuş, F., 2021. The effect of

robotic coding education on preschoolers' problem solving and creative thinking skills. *Thinking Skills and Creativity*, 40, p.100812. Available from: https://doi.org/10.1016/j.tsc.2021.100812.

[14] Carlborg, N., Tyrén, M., Heath, C. and Eriksson, E., 2019. The scope of autonomy when teaching computational thinking in primary school. *International Journal of Child-Computer Interaction*, 21, pp.130–139. Available from: https://doi.org/10.1016/j.ijcci.2019.06.005.

[15] Carvalho, R.M., Castro Andrade, R.M. de, Oliveira, K.M. de, Sousa Santos, I. de and Bezerra, C.I.M., 2017. Quality characteristics and measures for human–computer interaction evaluation in ubiquitous systems. *Software Quality Journal*, 25(3), pp.743–795. Available from: https://doi.org/10.1007/s11219-016-9320-z.

[16] Chaika, V.M., Kuzma, I.I., Yankovych, O.I., Binytska, K.M., Pysarchuk, O.T. and Ivanova, T.V., 2024. Implementing media educational technology in Ukrainian preschool institutions. *Educational Technology Quarterly*, 2024(1), p.1–19. Available from: https://doi.org/10.55056/etq.57.

[17] Chalmers, C., 2018. Robotics and computational thinking in primary school. *International Journal of Child-Computer Interaction*, 17, pp.93–100. Available from: https://doi.org/10.1016/j.ijcci.2018.06.005.

[18] Chen, G., Shen, J., Barth-Cohen, L., Jiang, S., Huang, X. and Eltoukhy, M., 2017. Assessing elementary students' computational thinking in everyday reasoning and robotics programming. *Computers & Education*, 109, pp.162–175. Available from: https://doi.org/10.1016/j.compedu.2017.03.001.

[19] Cheruiyot, K.C. and Chepngetich, K.J., 2023. Culture in Design of Coding Toolkits for Young Learners in Developing Economies in Africa: A Review. *Current Journal of Applied Science and Technology*, 42(24), pp.43–50. Available from: https://doi.org/10.9734/cjast/2023/v42i244177.

[20] Clements, D.H. and Gullo, D.F., 1984. Effects of computer programming on young children's cognition. *Journal of Educational Psychology*, 76(6), pp.1051–1058. Available from: https://doi.org/10.1037/0022-0663.76.6.1051.

[21] Dagienė, V., Jevsikova, T., Stupurienė, G. and Juškevičienė, A., 2022. Teaching Computational Thinking in Primary Schools: Worldwide Trends and Teachers' Attitudes. *Computer Science and Information Systems*, 19(1), pp.1–24. Available from: https://doi.org/10.2298/CSIS201215033D.

[22] Department: Basic Education, Republic of South Africa, 2021. Proposed Amendments to the Curriculum and Assessment Policy Statement (CAPS) to Make Provision for Coding and Robotics Grades. Available from: https://www.eccurriculum.co.za/Circulars/2021-Circulars/20%20Grade%207-9%20Coding%20and%20Robotics%20Draft%20CAPS%20FINAL%2019Mar2021.pdf.

[23] Di Lieto, M.C., Inguaggiato, E., Castro, E., Cecchi, F., Cioni, G., Dell'Omo, M., Laschi, C., Pecini, C., Santerini, G., Sgandurra, G. and Dario, P., 2017. Educational Robotics intervention on Executive Functions in preschool children: A pilot study. *Computers in Human Behavior*, 71, pp.16–23. Available from: https://doi.org/10.1016/j.chb.2017.01.018.

[24] Dietz, G., Le, J.K., Tamer, N., Han, J., Gweon, H., Murnane, E.L. and Landay, J.A., 2021. StoryCoder: Teaching Computational Thinking Concepts Through Storytelling in a Voice-Guided App for Children. *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery, CHI '21, p.54. Available from: https://doi.org/10.1145/3411764.3445039.

[25] Drăgănoiu, R., Moldoveanu, F., Morar, A. and Moldoveanu, A., 2024. DigiMathArt: a game-based approach to learning Mathematics and Program-

ming. *Interactive Learning Environments*, 32(9), pp.4831–4856. Available from: https://doi.org/10.1080/10494820.2023.2205903.
[26] Dudu, W.T. and Mashoko, D., 2023. Implementation of ICT integration in STEM curriculum: A case of secondary schools and higher education institutions (HEIs) in Zimbabwe . In: U. Ramnarain and M. Ndlovu, eds. *Information and Communications Technology in STEM Education*. London: Routledge, chap. 4, pp.50–67. Available from: https://doi.org/10.4324/9781003279310-4.
[27] Gafoor, A.K., 2011. Elementary Competencies In 3R's Among Upper Primary Pupilsof Kerala: A Secondary Analysis. *Innovations and Researches in Education*, 1(1), pp.51–68. Available from: https://www.researchgate.net/publication/262924134.
[28] Gorrah, A.R., 2022. *Teachers' perception of gender stereotypes with the teaching of coding and robotics in primary schools*. Master's thesis. University of Johannesburg. Available from: https://hdl.handle.net/10210/503429.
[29] Guncaga, J., Korenova, L., Záhorec, J. and Ostradicky, P., 2024. Innovative Approach on Teaching and Learning with Technical Aids for STEM Education at the Primary Level. *Education Sciences*, 14(7), p.682. Available from: https://doi.org/10.3390/educsci14070682.
[30] Hartt, M., Hosseini, H. and Mostafapour, M., 2020. Game On: Exploring the Effectiveness of Game-based Learning. *Planning Practice & Research*, 35(5), pp.589–604. Available from: https://doi.org/10.1080/02697459.2020.1778859.
[31] Heckman, J.J. and Masterov, D.V., 2007. The Productivity Argument for Investing in Young Children. *Applied Economic Perspectives and Policy*, 29(3), pp.446–493. https://jenni.uchicago.edu/papers/Heckman_Masterov_RAE_2007_v29_n3.pdf, Available from: https://doi.org/10.1111/j.1467-9353.2007.00359.x.
[32] Highfield, K., Paciga, K.A. and Donohue, C., 2018. Supporting Whole Child Development in the Digital Age. In: S.J. Danby, M. Fleer, C. Davidson and M. Hatzigianni, eds. *Digital Childhoods: Technologies and Children's Everyday Lives*. Singapore: Springer Singapore, *International Perspectives on Early Childhood Education and Development*, vol. 22, pp.165–182. Available from: https://doi.org/10.1007/978-981-10-6484-5_11.
[33] Hill, C., Dwyer, H.A., Martinez, T., Harlow, D. and Franklin, D., 2015. Floors and Flexibility: Designing a Programming Environment for 4th-6th Grade Classrooms. *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*. New York, NY, USA: Association for Computing Machinery, SIGCSE '15, p.546–551. Available from: https://doi.org/10.1145/2676723.2677275.
[34] Hsu, T.C., Chang, S.C. and Hung, Y.T., 2018. How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers & Education*, 126, pp.296–310. Available from: https://doi.org/10.1016/j.compedu.2018.07.004.
[35] Hu, X., Li, H. and Gao, M., 2025. Integrating coding education into early STEM programme: the changing roles of children and integrative learning context. *Early Child Development and Care*. Available from: https://doi.org/10.1080/03004430.2025.2479600.
[36] Huang, X. and Qiao, C., 2024. Enhancing Computational Thinking Skills Through Artificial Intelligence Education at a STEAM High School. *Science & Education*, 33(2), pp.383–403. Available from: https://doi.org/10.1007/s11191-022-00392-6.
[37] Kafai, Y., Proctor, C. and Lui, D., 2020. From theory bias to theory dialogue: embracing cognitive, situated, and critical framings of computational thinking in K-12 CS education. *ACM Inroads*, 11(1), p.44–53. Available from: https:

//doi.org/10.1145/3381887.

[38] Kalogiannakis, M. and Papadakis, S., 2019. Pre-service kindergarten teachers' acceptance of 'ScratchJr' as a tool for learning and teaching Computational Thinking and science education. *The Journal of Emergent Science*, 15, pp.31–34. Available from: https://www.researchgate.net/publication/336022829.

[39] Kalyenci, D., Metin, Ş. and Başaran, M., 2022. Test for assessing coding skills in early childhood. *Education and Information Technologies*, 27(4), pp.4685–4708. Available from: https://doi.org/10.1007/s10639-021-10803-w.

[40] Kambona, W., 2024. Tutors' perspectives on the integration of information and communication technology in early-grade teacher colleges in Tanzania. *Educational Technology Quarterly*, 2024(4), p.434–455. Available from: https://doi.org/10.55056/etq.795.

[41] Kaplancali, U.T. and Demirkol, Z., 2017. Teaching Coding to Children: A Methodology for Kids 5+. *International Journal of Elementary Education*, 6(4), pp.32–37. Available from: https://doi.org/10.11648/j.ijeedu.20170604.11.

[42] Kassa, E.A. and Mekonnen, E.A., 2022. Computational thinking in the Ethiopian secondary school ICT curriculum. *Computer Science Education*, 32(4), pp.502–531. Available from: https://doi.org/10.1080/08993408.2022.2095594.

[43] Kasusse, M., 2005. Bridging the digital divide in Sub-Saharan Africa: The rural challenge in Uganda. *The International Information & Library Review*, 37(3), pp.147–158. Available from: https://doi.org/10.1080/10572317.2005.10762676.

[44] Konok, V., Liszkai-Peres, K., Bunford, N., Ferdinandy, B., Jurányi, Z., Ujfalussy, D.J., Réti, Z., Pogány Ákos, Kampis, G. and Miklósi Ádám, 2021. Mobile use induces local attentional precedence and is associated with limited socio-cognitive skills in preschoolers. *Computers in Human Behavior*, 120, p.106758. Available from: https://doi.org/10.1016/j.chb.2021.106758.

[45] Kritzer, K.L. and Green, L., 2021. Playing With Code: An Innovative Way to Teach Numeracy and Early Math Concepts to Young DHH Children in the 21st Century. *American Annals of the Deaf*, 166(3), pp.409–423. Available from: https://doi.org/10.1353/aad.2021.0027.

[46] Kuteesa, K.N., Akpuokwe, C.U. and Udeh, C.A., 2024. Theoretical perspectives on digital divide and ICT access: Comparative study of rural communities in Africa and the United States. *Computer Science & IT Research Journal*, 5(4), pp.839–849. Available from: https://doi.org/10.51594/csitrj.v5i4.1045.

[47] Law, C.Y., Goh, K.O., Ooi, S.Y., Chong, L.Y., Ing, C.Y. and Yong, Y.Y., 2023. A Pilot Study of Learning Programming with ScratchJr. In: K. Arai, ed. *Advances in Information and Communication*. Cham: Springer Nature Switzerland, *Lecture Notes in Networks and Systems*, vol. 651, pp.785–793. Available from: https://doi.org/10.1007/978-3-031-28076-4_57.

[48] Lee, J., Yunus, S. and Lee, J.O., 2025. Investigating Children's Programming Skills Through Play with Robots (KIBO). *Early Childhood Education Journal*, 53(1), pp.109–117. Available from: https://doi.org/10.1007/s10643-023-01563-y.

[49] Lennon, M., Pila, S., Flynn, R. and Wartella, E.A., 2022. Young children's social and independent behavior during play with a coding app: Digital game features matter in a 1:1 child to tablet setting. *Computers & Education*, 190, p.104608. Available from: https://doi.org/10.1016/j.compedu.2022.104608.

[50] Leung, S.K.Y., Wu, J. and Li, J.W., 2024. Children's knowledge construction of computational thinking in a play-based classroom. *Early Child Development and Care*, 194(2), pp.208–229. Available from: https://doi.org/10.1080/03004430.2023.2299405.

[51] Liberman, N., Kolikant, Y.B.D. and Beeri, C., 2012. “Regressed experts” as a new state in teachers’ professional development: lessons from Computer Science teachers’ adjustments to substantial changes in the curriculum. *Computer Science Education*, 22(3), pp.257–283. Available from: https://doi.org/10.1080/08993408.2012.721663.

[52] Lin, S.Y., Chien, S.Y., Hsiao, C.L., Hsia, C.H. and Chao, K.M., 2020. Enhancing Computational Thinking Capability of Preschool Children by Game-based Smart Toys. *Electronic Commerce Research and Applications*, 44, p.101011. Available from: https://doi.org/10.1016/j.elerap.2020.101011.

[53] Lodi, M. and Martini, S., 2021. Computational Thinking, Between Papert and Wing. *Science & education*, 30(4), pp.883–908. Available from: https://doi.org/10.1007/s11191-021-00202-5.

[54] Lonati, V., Brodnik, A., Bell, T., Csizmadia, A.P., De Mol, L., Hickman, H., Keane, T., Mirolo, C. and Monga, M., 2022. What We Talk About When We Talk About Programs. *Proceedings of the 2022 Working Group Reports on Innovation and Technology in Computer Science Education*. New York, NY, USA: Association for Computing Machinery, ITiCSE-WGR ’22, p.117–164. Available from: https://doi.org/10.1145/3571785.3574125.

[55] Louka, K. and Papadakis, S., 2024. Enhancing computational thinking in early childhood education through ScratchJr integration. *Heliyon*, 10(10), p.e30482. Available from: https://doi.org/10.1016/j.heliyon.2024.e30482.

[56] Macrides, E., Miliou, O. and Angeli, C., 2022. Programming in early childhood education: A systematic review. *International Journal of Child-Computer Interaction*, 32, p.100396. Available from: https://doi.org/10.1016/j.ijcci.2021.100396.

[57] Marinus, E., Powell, Z., Thornton, R., McArthur, G. and Crain, S., 2018. Unravelling the Cognition of Coding in 3-to-6-year Olds: The development of an assessment tool and the relation between coding ability and cognitive compiling of syntax in natural language. *Proceedings of the 2018 ACM Conference on International Computing Education Research*. New York, NY, USA: Association for Computing Machinery, ICER ’18, p.133–141. Available from: https://doi.org/10.1145/3230977.3230984.

[58] Mathew, R., Malik, S.I. and Tawafak, R.M., 2019. Teaching Problem Solving Skills using an Educational Game in a Computer Programming Course. *Informatics in Education*, 18(2), pp.359–373. Available from: https://doi.org/10.15388/infedu.2019.17.

[59] Mathrani, A., Christian, S. and Ponder-Sutton, A., 2016. PlayIT: Game Based Learning Approach for Teaching Programming Concepts. *Journal of Educational Technology & Society*, 19(2), pp.5–17. Available from: https://drive.google.com/file/d/1Z8tpXBV1hFX25QSpfG5Ih1Ia9LVFlrsh/view.

[60] Mayer, R.E., Dyck, J.L. and Vilberg, W., 1986. Learning to program and learning to think: what’s the connection? *Communications of the ACM*, 29(7), p.605–610. Available from: https://doi.org/10.1145/6138.6142.

[61] McGowan, J., Straus, S., Moher, D., Langlois, E.V., O’Brien, K.K., Horsley, T., Aldcroft, A., Zarin, W., Garitty, C.M., Hempel, S., Lillie, E., Tunçalp, Ö. and Tricco, A.C., 2020. Reporting scoping reviews—PRISMA ScR extension. *Journal of clinical epidemiology*, 123, pp.177–179. Available from: https://doi.org/10.1016/j.jclinepi.2020.03.016.

[62] Melliti, M. and Henchiri, M., 2024. Advancing autonomy in Tunisian higher education: Exploring the role of technology in empowering learners. *Educational Technology Quarterly*, 2024(4), p.456–483. Available from: https://doi.org/10.55056/etq.780.

[63] Metin, S., 2022. Activity-based unplugged coding during the preschool period.

International Journal of Technology and Design Education, 32(1), pp.149–165. Available from: https://doi.org/10.1007/s10798-020-09616-8.

[64] Metin, S., Basaran, M. and Kalyenci, D., 2023. Examining coding skills of five-year-old children. *Pedagogical Research*, 8(2), p.em0154. Available from: https://doi.org/10.29333/pr/12802.

[65] Metin, Ş., Kalyenci, D., Başaran, M., Relkin, E. and Bilir, B., 2024. Design-Based Digital Story Program: Enhancing Coding and Computational Thinking Skills in Early Childhood Education. *Early Childhood Education Journal*. Available from: https://doi.org/10.1007/s10643-024-01728-3.

[66] Montuori, C., Pozzan, G., Padova, C., Ronconi, L., Vardanega, T. and Arfé, B., 2023. Combined Unplugged and Educational Robotics Training to Promote Computational Thinking and Cognitive Abilities in Preschoolers. *Education Sciences*, 13(9), p.858. Available from: https://doi.org/10.3390/educsci13090858.

[67] Moreno-León, J., Robles, G. and Román-González, M., 2016. Code to Learn: Where Does It Belong in the K-12 Curriculum? *Journal of Information Technology Education: Research*, 15, pp.283–303. Available from: https://www.jite.org/documents/Vol15/JITEv15ResearchP283-303Moreno2233.pdf.

[68] Munn, Z., Peters, M.D.J., Stern, C., Tufanaru, C., McArthur, A. and Aromataris, E., 2018. Systematic review or scoping review? Guidance for authors when choosing between a systematic or scoping review approach. *BMC Medical Research Methodology*, 18(1), p.143. Available from: https://doi.org/10.1186/s12874-018-0611-x.

[69] Murcia, K., Cross, E. and Lowe, G., 2025. Young children's computational thinking: educator pedagogy fostering children's play and learning with a tangible coding device. *The Australian Educational Researcher*, 52(2), pp.1261–1279. Available from: https://doi.org/10.1007/s13384-024-00762-9.

[70] Murris, K., Scott, F., Thomsen, B.S., Dixon, K., Giorza, T., Peers, J. and Lawrence, C., 2023. Researching digital inequalities in children's play with technology in South Africa. *Learning, Media and Technology*, 48(3), pp.542–555. Available from: https://doi.org/10.1080/17439884.2022.2095570.

[71] Negrini, L., Giang, C. and Bonnet, E., 2021. Designing Tools and Activities for Educational Robotics in Online Learning. In: N. Eteokleous and E. Nisiforou, eds. *Designing, Constructing, and Programming Robots for Learning*. Hershey, PA: IGI Global Scientific Publishing, chap. 10, pp.202–222. Available from: https://doi.org/10.4018/978-1-7998-7443-0.ch010.

[72] Ngandu, M.R., Risinamhodzi, D., Dzvapatsva, G.P. and Matobobo, C., 2023. Capturing student interest in software engineering through gamification: a systematic literature review. *Discover Education*, 2(1), p.47. Available from: https://doi.org/10.1007/s44217-023-00069-4.

[73] Nolan, J. and McBride, M., 2014. Beyond gamification: reconceptualizing game-based learning in early childhood environments. *Information, Communication & Society*, 17(5), pp.594–608. Available from: https://doi.org/10.1080/1369118X.2013.808365.

[74] Okoli, C., 2015. A Guide to Conducting a Standalone Systematic Literature Review. *Communications of the association for information systems*, 37. Available from: https://doi.org/10.17705/1CAIS.03743.

[75] Okoli, C., 2022. Developing Theory from Literature Reviews with Theoretical Concept Synthesis: Topical, Propositional and Confirmatory Approaches. Available from: https://doi.org/10.2139/ssrn.3452134.

[76] Papadakis, S., 2020. Apps to Promote Computational Thinking Concepts and Coding Skills in Children of Preschool and Pre-Primary School Age. In: S. Papadakis and M. Kalogiannakis, eds. *Mobile Learning Applications in Early Child-*

hood Education. Hershey, PA: IGI Global Scientific Publishing, chap. 6, pp.101–121. Available from: https://doi.org/10.4018/978-1-7998-1486-3.ch006.

[77] Papadakis, S., 2020. Evaluating a game-development approach to teach introductory programming concepts in secondary education. *International Journal of Technology Enhanced Learning*, 12(2), p.127–145. Available from: https://doi.org/10.1504/ijtel.2020.106282.

[78] Papadakis, S., 2022. Apps to Promote Computational Thinking and Coding Skills to Young Age Children: A Pedagogical Challenge for the 21st Century Learners. *Educational Process: International Journal*, 11(1), pp.7–13. Available from: https://doi.org/10.22521/edupij.2022.111.1.

[79] Papadakis, S., 2024. Can Preschoolers Learn Computational Thinking and Coding Skills with ScratchJr? A Systematic Literature Review. *International Journal of Educational Reform*, 33(1), pp.28–61. Available from: https://doi.org/10.1177/10567879221076077.

[80] Papadakis, S. and Kalogiannakis, M., 2020. Learning Computational Thinking Development in Young Children With Bee-Bot Educational Robotics. In: M. Kalogiannakis and S. Papadakis, eds. *Handbook of Research on Tools for Teaching Computational Thinking in P-12 Education*. Hershey, PA: IGI Global Scientific Publishing, chap. 11, pp.289–309. Available from: https://doi.org/10.4018/978-1-7998-4576-8.ch011.

[81] Papert, S., 1972. Teaching Children Thinking. *Programmed Learning and Educational Technology*, 9(5), pp.245–255. https://dailypapert.com/wp-content/uploads/2020/07/Papert-Teaching-Children-Thinking-AIM-247.pdf, Available from: https://doi.org/10.1080/1355800720090503.

[82] Papert, S., 1980. *Mindstorms: Children, Computers, and Powerful Ideas*. New York: Basic Books. Available from: https://worrydream.com/refs/Papert_1980_-_Mindstorms,_1st_ed.pdf.

[83] Papert, S. and Harel, I., 1991. Situating Constructionism. *Constructionism*. Ablex Publishing Corporation, vol. 36, pp.1–11. Available from: http://www.papert.org/articles/SituatingConstructionism.html.

[84] Pelizzari, F., Marangi, M., Rivoltella, P.C., Peretti, G., Massaro, D. and Villani, D., 2023. Coding and childhood between play and learning: Research on the impact of coding in the learning of 4-year-olds. *Research on Education and Media*, 15(1), pp.9–19. Available from: https://doi.org/10.2478/rem-2023-0003.

[85] Peters, M.D.J., Marnie, C., Colquhoun, H., Garritty, C.M., Hempel, S., Horsley, T., Langlois, E.V., Lillie, E., O'Brien, K.K., Tunçalp, Ö., Wilson, M.G., Zarin, W. and Tricco, A.C., 2021. Scoping reviews: reinforcing and advancing the methodology and application. *Systematic Reviews*, 10(1), p.263. Available from: https://doi.org/10.1186/s13643-021-01821-3.

[86] Pila, S., Aladé, F., Sheehan, K.J., Lauricella, A.R. and Wartella, E.A., 2019. Learning to code via tablet applications: An evaluation of *Daisy the Dinosaur* and *Kodable* as learning tools for young children. *Computers & Education*, 128, pp.52–62. Available from: https://doi.org/10.1016/j.compedu.2018.09.006.

[87] Pollarolo, E., Papavlasopoulou, S., Granone, F. and Reikerås, E., 2024. Play with Coding Toys in Early Childhood Education and Care: Teachers' Pedagogical Strategies, Views and Impact on Children's Development. A Systematic Literature Review. *Entertainment Computing*, 50, p.100637. Available from: https://doi.org/10.1016/j.entcom.2024.100637.

[88] Rehmat, A.P., Ehsan, H. and Cardella, M.E., 2020. Instructional strategies to promote computational thinking for young learners. *Journal of Digital Learning in Teacher Education*, 36(1), pp.46–62. Available from: https://doi.org/10.1080/21532974.2019.1693942.

[89] Relkin, E., de Ruiter, L.E. and Bers, M.U., 2021. Learning to code and the acquisition of computational thinking by young children. *Computers & Education*, 169, p.104222. Available from: https://doi.org/10.1016/j.compedu.2021.104222.

[90] Relkin, E., Ruiter, L.E. de and Bers, M.U., 2021. Learning to code and the acquisition of computational thinking by young children. *Computers & Education*, 169. Available from: https://doi.org/10.1016/j.compedu.2021.104222.

[91] Rich, P.J., Bartholomew, S., Daniel, D., Dinsmoor, K., Nielsen, M., Reynolds, C., Swanson, M., Winward, E. and Yauney, J., 2024. Trends in tools used to teach computational thinking through elementary coding. *Journal of Research on Technology in Education*, 56(3), pp.269–290. Available from: https://doi.org/10.1080/15391523.2022.2121345.

[92] Sigayret, K., Tricot, A. and Blanc, N., 2022. Unplugged or plugged-in programming learning: A comparative experimental study. *Computers & Education*, 184, p.104505. Available from: https://doi.org/10.1016/j.compedu.2022.104505.

[93] Stager, G.S., 2016. Seymour Papert (1928–2016). *Nature*, 537(7620), pp.308–308. Available from: https://doi.org/10.1038/537308a.

[94] Sterling, L., 2015. An education for the 21st century means teaching coding in schools. *The Conversation*. Available from: https://theconversation.com/an-education-for-the-21st-century-means-teaching-coding-in-schools-42046.

[95] Su, J., Yang, W. and Li, H., 2023. A Scoping Review of Studies on Coding Curriculum in Early Childhood: Investigating Its Design, Implementation, and Evaluation. *Journal of Research in Childhood Education*, 37(2), pp.341–361. Available from: https://doi.org/10.1080/02568543.2022.2097349.

[96] Su, J. and Zhong, Y., 2022. Artificial Intelligence (AI) in early childhood education: Curriculum design and future directions. *Computers and Education: Artificial Intelligence*, 3, p.100072. Available from: https://doi.org/10.1016/j.caeai.2022.100072.

[97] Sun, L., Guo, Z. and Zhou, D., 2024. Measuring development of young students' coding ability through a graphical teaching intervention: further explanation of the effect of coding experience and coding interest. *Interactive Learning Environments*, 32(7), pp.3732–3755. Available from: https://doi.org/10.1080/10494820.2023.2190356.

[98] Sung, W., Ahn, J. and Black, J.B., 2017. Introducing Computational Thinking to Young Learners: Practicing Computational Perspectives Through Embodiment in Mathematics Education. *Technology, Knowledge and Learning*, 22(3), pp.443–463. Available from: https://doi.org/10.1007/s10758-017-9328-x.

[99] Tedre, M. and Denning, P.J., 2016. The long quest for computational thinking. *Proceedings of the 16th Koli Calling International Conference on Computing Education Research*. New York, NY, USA: Association for Computing Machinery, Koli Calling '16, p.120–129. Available from: https://doi.org/10.1145/2999541.2999542.

[100] Tengler, K., Kastner-Hauler, O. and Sabitzer, B., 2021. Enhancing Computational Thinking Skills using Robots and Digital Storytelling. *International Conference on Computer Supported Education, CSEDU - Proceedings*. INSTICC, SciTePress, vol. 1, pp.157–164. Available from: https://doi.org/10.5220/0010477001570164.

[101] Thompson, R. and Tanimoto, S., 2016. Children's Storytelling and Coding: Literature Review and Future Potential. *PPIG 2016 - 27th Annual Workshop*. Available from: https://www.ppig.org/papers/2016-ppig-27th-thompson/.

[102] Tricco, A.C., Lillie, E., Zarin, W., O'Brien, K.K., Colquhoun, H., Levac, D., Moher, D., Peters, M.D., Horsley, T., Weeks, L., Hempel, S., Akl, E.A., Chang, C., McGowan, J., Stewart, L., Hartling, L., Aldcroft, A., Wilson, M.G., Garritty,

C., Lewin, S., Godfrey, C.M., Macdonald, M.T., Langlois, E.V., Soares-Weiser, K., Moriarty, J., Clifford, T., Tunçalp, Ö. and Straus, S.E., 2018. PRISMA Extension for Scoping Reviews (PRISMA-ScR): Checklist and Explanation. *Annals of Internal Medicine*, 169(7), pp.467–473. Available from: https://doi.org/10.7326/M18-0850.

[103] Uğraş, M., Güneş, H., Uğraş, H., Papadakis, S. and Kalogiannakis, M., 2025. Coding activities in early childhood: a systematic review. *International Journal of Technology Enhanced Learning*, 17(1), pp.45–80. Available from: https://doi.org/10.1504/IJTEL.2025.143512.

[104] Videnovik, M., Vlahu-Gjorgievska, E. and Trajkovik, V., 2021. To code or not to code: Introducing coding in primary schools. *Computer applications in engineering education*, 29(5), pp.1132–1145. Available from: https://doi.org/10.1002/cae.22369.

[105] Vygotsky, L.S., 1976. Play and its Role in the Mental Development of the Child. In: J.S. Bruner, A. Jolly and K. Sylva, eds. *Play: Its Role in Development and Evolution*. Penguin Books Ltd., pp.863–895. Available from: https://www.researchgate.net/publication/328486788.

[106] Wang, L., Shi, D., Geng, F., Hao, X., Chanjuan, F. and Li, Y., 2022. Effects of cognitive control strategies on coding learning outcomes in early childhood. *The Journal of Educational Research*, 115(2), pp.133–145. Available from: https://doi.org/10.1080/00220671.2022.2074946.

[107] Wu, Z., Zheng, L. and Huang, L., 2024. Learning to code and coding to learn: A robotics curriculum integrating tangible programming and road safety education for young children. *Education and Information Technologies*, 29(17), pp.23049–23087. Available from: https://doi.org/10.1007/s10639-024-12757-1.

[108] Yang, D., Yang, Z. and Bers, M.U., 2025. The efficacy of a computer science curriculum for early childhood: evidence from a randomized controlled trial in K-2 classrooms. *Computer Science Education*, 35(2), pp.321–341. Available from: https://doi.org/10.1080/08993408.2023.2279198.

[109] Yang, W., 2022. Artificial intelligence education for young children: Why, what, and how in curriculum design and implementation. *Computers and Education: Artificial Intelligence*, 3, p.100061. Available from: https://doi.org/10.1016/j.caeai.2022.100061.

[110] Yang, W., Ng, D.T.K. and Su, J., 2023. The impact of story-inspired programming on preschool children's computational thinking: A multi-group experiment. *Thinking Skills and Creativity*, 47, p.101218. Available from: https://doi.org/10.1016/j.tsc.2022.101218.

[111] Yang, Z., Blake-West, J., Yang, D. and Bers, M., 2025. The impact of a block-based visual programming curriculum: Untangling coding skills and computational thinking. *Learning and Instruction*, 95, p.102041. Available from: https://doi.org/10.1016/j.learninstruc.2024.102041.

[112] Yildiz, E.P. and Çengel, M., 2022. An Experimental Study in Determining Basic Robotic Coding Skills of Pre-School Students with Intellectual Disabilities. *International Journal of Interactive Mobile Technologies (iJIM)*, 16(23), p.84–98. Available from: https://doi.org/10.3991/ijim.v16i23.36077.

[113] Yu, J. and Roque, R., 2019. A review of computational toys and kits for young children. *International Journal of Child-Computer Interaction*, 21, pp.17–36. Available from: https://doi.org/10.1016/j.ijcci.2019.04.001.

[114] Zhao, L., Liu, X., Wang, C. and Su, Y.S., 2022. Effect of different mind mapping approaches on primary school students' computational thinking skills during visual programming learning. *Computers & Education*, 181, p.104445. Available from: https://doi.org/10.1016/j.compedu.2022.104445.