

Review Article

Cite this article: Huertas, Y., Boude, O., & Vargas, A. D. (2026). Curricular Practices for the Development of Computational Thinking in Secondary Education: A Systematic Review. *Educational Process: International Journal*, 22, e2026049.
<https://doi.org/10.22521/edupij.2026.22.49>

Received July 31, 2025

Accepted September 28, 2025

Keywords: Computational Thinking, Secondary Education, Curricular integration, educational technology.

Author for correspondence:

Oscar Boude

✉ oscarbf@unisabana.edu.co

✉ Universidad de La Sabana, Colombia

Curricular Practices for the Development of Computational Thinking in Secondary Education: A Systematic Review

Yaneth Huertas^{id}, Oscar R. Boude^{id}, Ana D. Vargas^{id}

Abstract

Background/purpose. This systematic research examines global curricular methods for integrating computational thinking into secondary school.

Materials/methods. Forty empirical research published from 2008 to 2024 identified five principal categories: curricular practices, instructional models, methods of mainstreaming computational thinking, specific content areas, and evaluation procedures.

Results. The findings highlight a wide range of approaches, including visual and textual programming, instructional robotics, unplugged activities, design and prototyping, and STEAM projects. These practices are implemented through active pedagogical approaches such as project-based learning, problem-solving, and collaborative learning, which promote the development of transversal abilities relevant across a variety of knowledge disciplines. However, substantial obstacles remain due to a lack of curricular articulation, limited teacher participation in practice design, and insufficient systematization of evaluation systems that often focus primarily on final products and ignore learning processes. Furthermore, inequalities in access to technology resources, as well as a lack of studies in contexts such as Latin America and Africa, prevent a thorough global understanding of the phenomena.

Conclusion. This review emphasizes the need to advance more organized, sustainable, and contextually grounded initiatives that incorporate critical thinking as a cross-cutting competency across all levels and domains of the school curriculum, ensuring its accessibility and significance for all students, regardless of their technological or geographical circumstances.



OPEN ACCESS

© The Author(s), 2025. This is an Open Access article, distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>), which permits unrestricted re-use, distribution, and reproduction, provided the original article is properly cited.

1. Introduction

Computational thinking (CT) is recognized as a crucial competency in contemporary society (Zhang & Nouri, 2019). CT is systematically defined as an approach to problem-solving grounded in computer science principles, encompassing key components such as abstraction, decomposition, pattern recognition, and algorithm design (Lu et al., 2022; Shute et al., 2017). These elements are essential for addressing the intricate challenges posed by a world increasingly influenced by automation and digitalization (Wing, 2016; Grover & Pea, 2018; Tedre & Denning, 2021).

Wing (2006) posited that CT should be viewed as a transversal competency, rather than being limited to computer science specialists. This perspective initiated a debate that has progressed through distinct phases: an initial stage centered on the conceptual definition of computational thinking (Wing, 2006, 2016), a subsequent phase addressing its cognitive and pedagogical aspects (Brennan & Resnick, 2012; Grover & Pea, 2013), and a more recent phase focused on delineating its content, methodologies, and assessment within educational contexts (Voogt et al., 2015; Bocconi et al., 2016; Liu et al., 2024). This theoretical trajectory presents a dual perspective on critical thinking: as a cognitive competency that enhances logical reasoning, creativity, and self-regulation, and as a pedagogical domain that demands robust, sustainable curricular practices.

International organizations, including UNESCO (2021) and the OECD (2022), have advocated for the integration of CT into school curricula. The significance of this preparation for students in meeting the academic and professional requirements of a technology-driven society is emphasized (Barr & Stephenson, 2011; Zhao & Shute, 2019). Nonetheless, despite these advancements, current reviews frequently emphasize conceptual, technical, or methodological dimensions (Voogt et al., 2015; Bocconi et al., 2016), overlooking the examination of specific pedagogical practices in secondary education.

Research conducted by Grover and Pea (2013), McClelland and Grata (2018), and Zhang and Nouri (2019) indicate deficiencies in pedagogical frameworks, evaluation strategies, and sustainable curricular experiences. Recent research (Quiroz-Vallejo et al., 2021; Yeni et al., 2024; Liu et al., 2024) highlights the restricted transversal application of CT, the brief duration of implemented initiatives, and the inadequate systematization of its impact in non-technological domains.

Considering this context, innovative proposals have arisen, including game-based learning (Triantafyllou et al., 2024), project-based learning (Saad & Zainudin, 2024), and their application to addressing mathematical problems (Obando-Montoya et al., 2024). However, the development of content, methodologies, assessment, and disciplinary integration is still in its early stages, particularly at the secondary level. This underscores the necessity of developing a pedagogical framework that comprehensively addresses the intricacies and possibilities of computational thinking as both a skill and an area of study.

In this context, it is essential to identify the curricular methods now deployed globally to incorporate computational thinking into secondary education and to assess the pedagogical models, strategies, and content used, along with the evaluation mechanisms implemented. Comprehending these dynamics can provide essential assistance for enhancing its implementation in a systematic, inclusive, and sustainable way.

Therefore, the present review article aims to answer the following research questions:

Q1: What are the curricular practices being developed and integrated to foster computational thinking in secondary education globally?

Q1.1: What specific computational thinking contents are addressed in the curricular practices?

Q2: Which instructional models are utilized in the implementation of computational thinking within curricular practices?

Q3: How is computational thinking related to other subjects?

Q4: How is computational thinking learning assessed in curricular practices?

2. Methodology

This study adopted a systematic review and thematic synthesis design to identify and analyse curricular practices that promote the development of CT in secondary education worldwide. The methodology was based on the PRISMA 2020 guidelines, which ensure transparency and rigor in the identification, selection, evaluation, and synthesis of relevant studies. For data analysis, thematic synthesis was applied following the methodological orientations of Nowell et al. (2017) to code and categorize findings, organizing them into recurring themes. This methodological combination enabled mapping the state of the art and deepening the critical analysis of the evidence, highlighting patterns and gaps.

2.1. Inclusion and Exclusion Criteria

To ensure the relevance and quality of the included studies, specific inclusion and exclusion criteria were established. Table 1 outlines the specific criteria for this systematic literature review, and Figure 1 presents the PRISMA flow diagram used for this study. Consequently, out of the 157 initially retrieved documents, 40 studies that met all established criteria were selected.

Table 1. Inclusion and Exclusion Criteria

| Inclusion Criteria | Exclusion Criteria |
|---|--|
| Empirical studies (quantitative, qualitative, or mixed) | Primary or higher education studies |
| Population: secondary school students | Teacher training without direct student intervention |
| Explicit integration of CT into curricular practices | Theoretical reviews without empirical data |
| Full-text access in English or Spanish | Tool evaluations without classroom application |

The application of exclusion criteria responded to the need to ensure the relevance and coherence of the studies considered in the review. Accordingly, works focused on primary or higher education, as well as those exclusively oriented toward teacher training, were discarded because they do not provide direct evidence of curricular practices at the secondary level, which constitutes the focus of the present review. Likewise, theoretical reviews without empirical data were excluded, as they are mainly oriented toward conceptualization and do not allow for comparative analyses or the extraction of findings applicable to practice (Snyder, 2019). Evaluations of tools without classroom application were also excluded, as they focus solely on technical validation of specific resources rather than their curricular integration, which would limit the scope of the study (Petticrew & Roberts, 2006). Finally, to ensure the quality and reliability of the evidence, documents that had not been peer-reviewed were excluded.

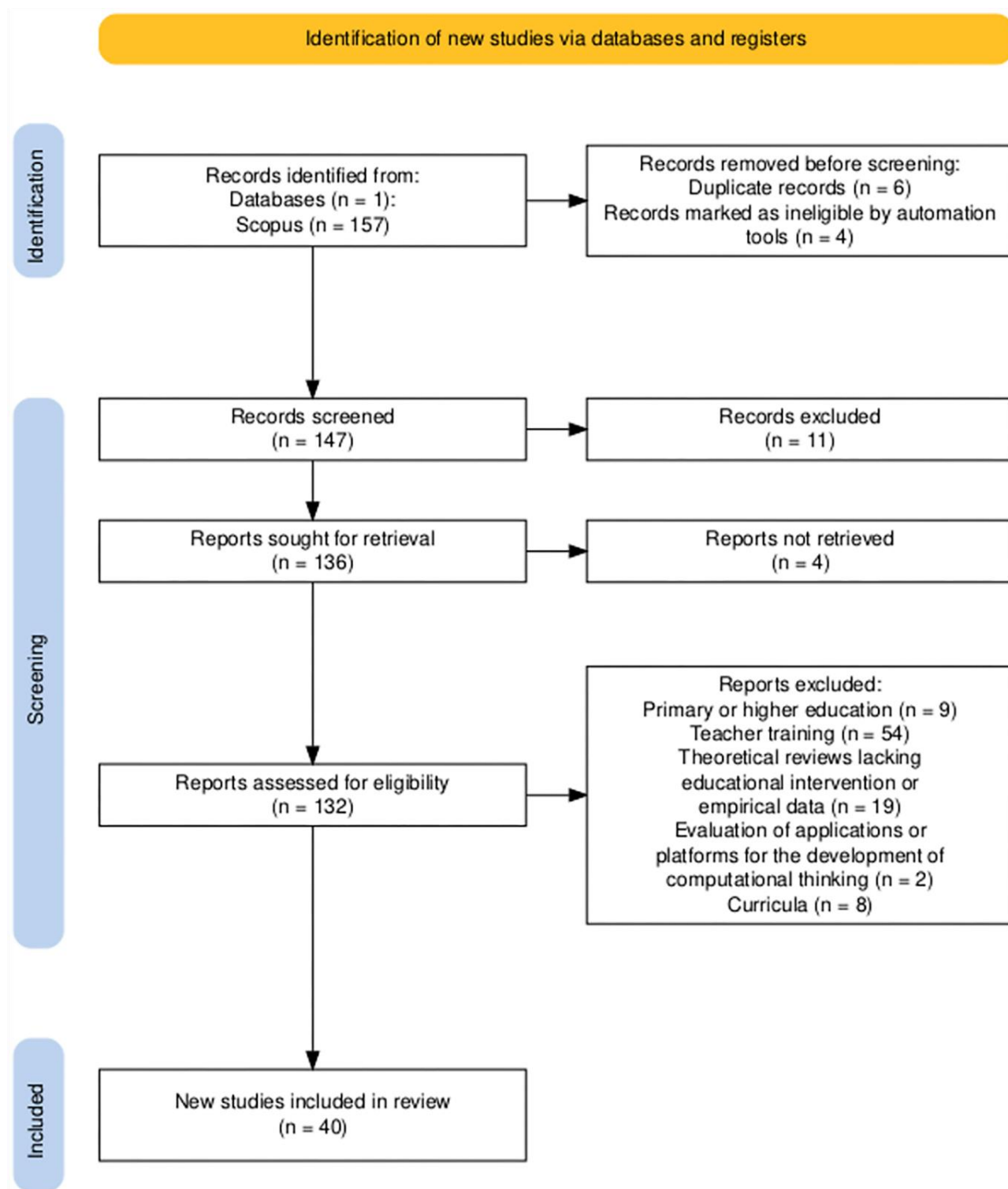


Figure 1. PRISMA Flow Diagram Detailing the Steps in Source Identification and Selection. Source: Haddaway et al., 2022.

2.2. Data Collection and Synthesis

The final selection of articles was carried out through the individual evaluation of each article considering the inclusion and exclusion criteria. Data extracted from these 40 selected articles included authors and publication year, participants and academic level, country or region of origin, tools and technologies used, and pedagogical practices implemented. This information was systematized in an Excel spreadsheet, ensuring transparency by following the PRISMA method and organizing studies by categories to facilitate subsequent thematic analysis.

2.3. Quality Assessment

To ensure the quality of the review conducted, after establishing that the established inclusion criteria were met. A thorough review of the texts was conducted to ensure that they contained relevant information to answer the established review questions, as well as to confirm that they were studies with consistent methodological rigor and results that contributed to the objectives of the review. Ultimately, 40 exemplary publications were selected for the current study.

2.4. Data Analysis and Synthesis

The analysis of the studies was carried out thru thematic coding, following Nowell et al. (2017), using descriptive labels that allowed for the organization and grouping of data into recurring categories presented in the 40 selected articles. The information, once the review and discussion process by the researchers was completed, was organized in an Excel spreadsheet (Figure 2) to systematize the data and facilitate the identification of patterns, frequencies, and recurring categories related to curricular practices, pedagogical models, content, integration strategies, and assessment of computational thinking. This systematization facilitated the comparison between studies and allowed the construction of a thematic synthesis that offers an overview of the current state, highlighting strengths and limitations in the integration of computational thinking in secondary education. It is important to note that, for this phase, two researchers segmented the data, identifying developing categories in line with the research questions. Subsequently, the validation of these categories was carried out, followed by an individual examination of each one. After concluding this process, the researchers met to compare their findings and outline the shared results of both investigations, which are documented in this text.

Now, to ensure the validity of the findings, bias control measures were implemented, such as triangulation and peer discussion. Additionally, inter-rater validation was employed, where two researchers independently coded the data to increase reliability. For this, Cohen's Kappa ($k > 0.8$) was calculated.

2.5. Temporal Trends and Geographical Distribution of Research On Computational Thinking in Secondary Education

As shown in Figure 2, the systematic review reveals a notable and growing scholarly interest in the development of computational thinking in secondary education. This upward trend suggests that researchers are progressively exploring a broader spectrum of methodologies and pedagogical tools for its implementation across diverse educational contexts. The increasing volume of publications can also be interpreted as direct evidence of the significant impact computational thinking is exerting on the educational landscape, specifically by fostering the development of targeted pedagogical strategies for its instruction.

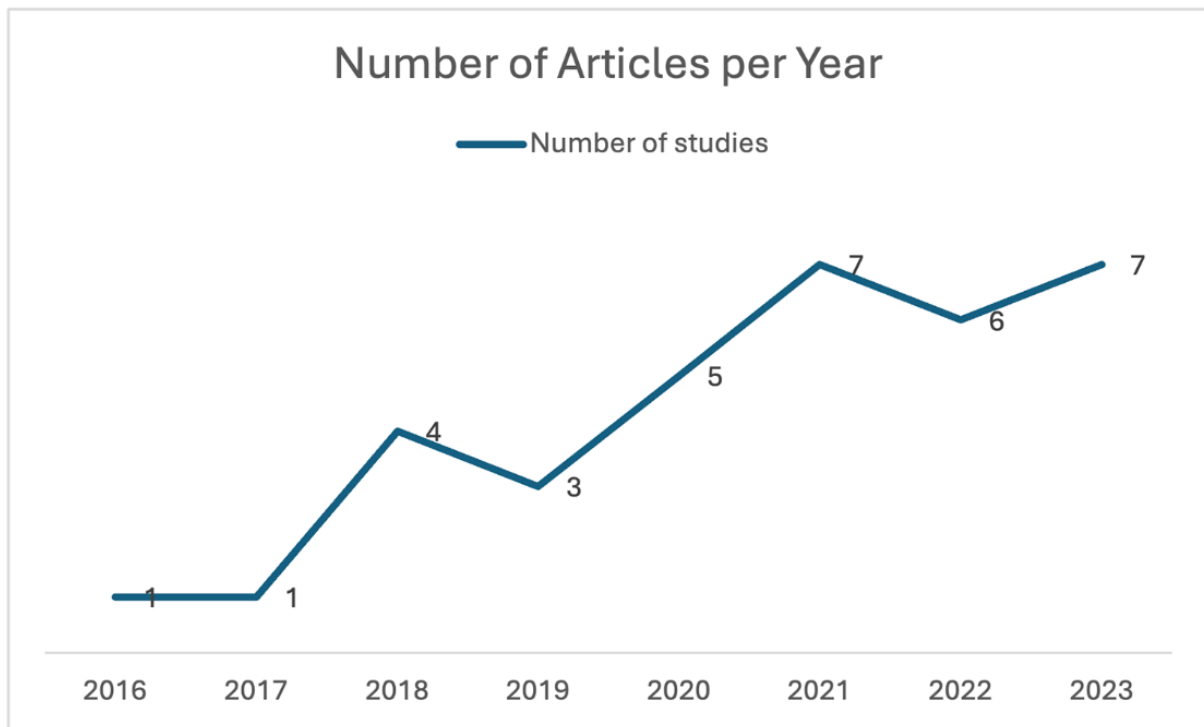


Figure 2. Selected Articles. Source: Authors' Own Elaboration.

Correspondingly, figure 3 illustrates the geographical distribution of the 40 selected articles, disaggregated by continent. A clear predominance of publications originating in Asia is evident, accounting for 50% of the total, indicating that this region currently leads research efforts in this field. This finding may be attributable to specific educational policies that actively promote the development of digital skills and the comprehensive inclusion of computational thinking in school curricula, particularly within computer science domains.

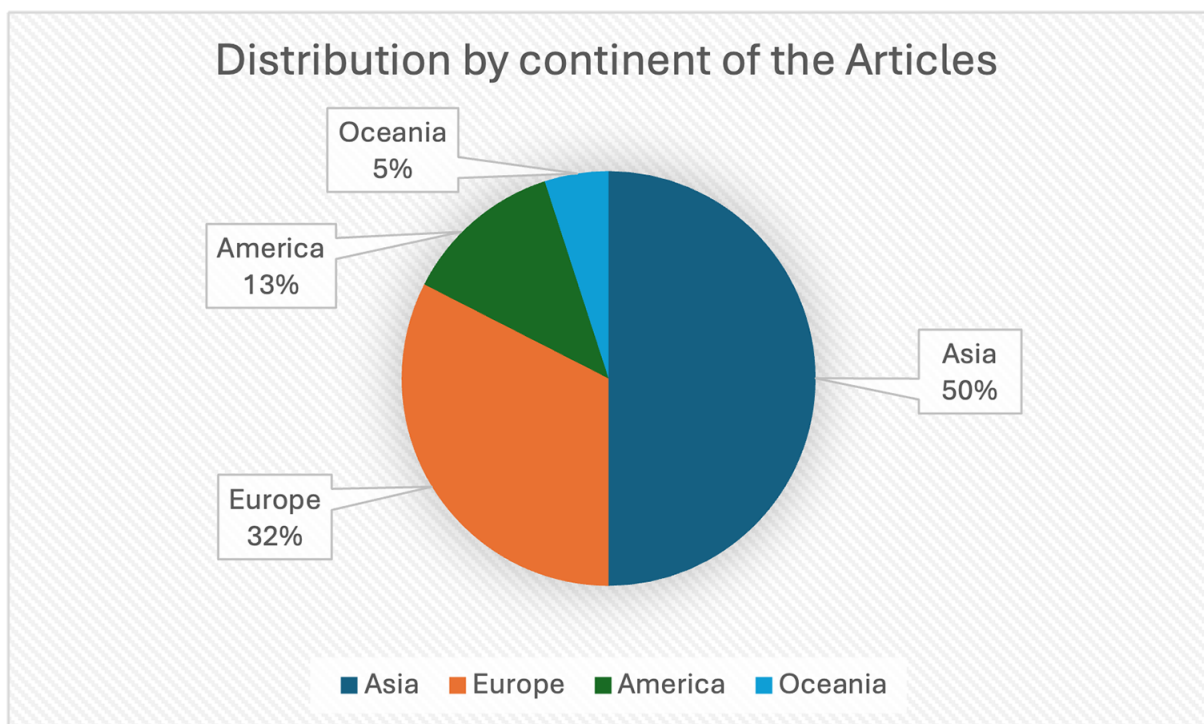


Figure 3. Global Distribution of Literature for Selected Articles. Source: Authors' Own Elaboration.

Europe accounts for 32% of publications, the Americas for 12%, and Oceania for 5%, so underscoring a significant regional gap in global research interest and scientific production. This heterogeneity may be associated with inconsistencies in educational regulations, unequal access to research resources, and differing emphases across regions on the promotion of computational thinking within their educational frameworks. The results obtained from each guiding research question are shown below.

The analysis reveals that, although educational policies and access to technology are essential foundations, understanding regional differences requires considering a broader spectrum of factors. In China and South Korea, mandatory secondary school programming is associated with strengthening analytical thinking (Sun et al., 2021). In Poland, the 2016-2017 reforms included algorithms and programming, though limitations persist due to a lack of teacher training (Yurdakök & Kalelioğlu, 2024). In Malaysia, the subject of Basic Computer Science was introduced, and "unplugged" activities were applied to reduce inequalities (Kucuk & Sisman, 2020).

Sociocultural traditions also condition implementation. In Turkey, cultural and gender factors influence student participation (Durak & Saritepeci, 2018). In the United States, unplugged activities have shown potential to improve scientific understanding and develop algorithmic skills (Peel et al., 2019).

Pedagogical traditions bring relevant differences. In Austria, the use of Poly-Universe integrated computational thinking into biology and physical education (Totan & Korucu, 2023). In Spain, Polya's heuristics strengthened mathematical comprehension and coding skills (Molina et al., 2020). In Finland, platforms such as EarSketch and TunePad promoted positive attitudes toward programming (Sinclair & Patterson, 2018).

Finally, teacher training appears as a critical factor. In Poland, the shortage of specialized teachers limits implementation (Yurdakök & Kalelioğlu, 2024). In China, the teaching of artificial intelligence faces difficulties due to a lack of materials and methodological approaches (Sun et al., 2021). In Malaysia, the need to train teachers in diversified pedagogies that address student heterogeneity is highlighted (Kucuk & Sisman, 2020).

In conclusion, the review shows that the development of computational thinking in secondary education has experienced sustained growth, though with notable regional differences. While Asia leads in production and integration policies, Europe, America, and Oceania show advances linked to specific pedagogical traditions and sociocultural contexts. The findings underscore that the consolidation of this field depends not only on policies and technological access but also on the ability to adapt pedagogical approaches and strengthen teacher training as key elements to ensure its effective implementation.

3. Results

As an integral component of the analytical process for the studies encompassed within this review, a qualitative methodology was employed to meticulously examine the content of the selected articles. This involved an initial, manual coding phase, utilizing descriptive labels. The primary objective was to systematically categorize the extracted information into five principal domains: curricular practices, pedagogical models, modalities of CT mainstreaming/curricular frameworks, specific content, and evaluation strategies.

This standardized coding technique enabled the systematic structuring of data, aided by an Excel spreadsheet (Figure 4). This tool was important for aggregating information from the articles using standardized codes, thereby facilitating the identification of links and patterns within the data. As a result, this greatly improved the thematic analysis and inter-study comparisons.

| Aspect | 1 | 4 | 9 | 9 |
|--|---|--|--|--|
| APA Reference | Atmatzidou, S., & Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. <i>Robotics and Autonomous Systems</i> , 75, 661–670. | Durak, H. Y., & Saritepeci, M. (2018). Analysis of the relation between computational thinking skills and various variables with the structural equation model. <i>Computers and Education</i> , 116, 191–202. | Peel, A., Sadler, T. D., & Friedrichsen, P. (2019). Learning natural selection through computational thinking: Unplugged design of algorithmic explanations. <i>Journal of Research in Science Teaching</i> , 56(7), 983–1007. | Durak, H. Y., Karaoglan, Y.F. & Yilmaz R. (2019). Computational thinking, programming self-efficacy, problem solving and experiences in the programming process conducted with robotic activities. <i>Contemporary</i> |
| General Information of the Study: | Authors: Soumela Atmatzidou and Stavros Demetriadis Year: 2016 Country: Greece Research design: Quasi-experimental with pretest/posttest. 164 students participated (89 from secondary education and 75 from vocational training). | Authors: Mustafa Saritepeci, Hatice Yildiz Durak Year: 2018 Country: Turkey Design: Relational screening model (structural equations) with 156 grade. | Authors: Amanda Peel, Troy D. Sadler, Patricia Friedrichsen Year: 2019 Country: United States Design: Mixed methods study, explanatory sequential design (quantitative + qualitative). | Authors: Ramazan Yilmaz, Fatma Gizem Karaoglan Yilmaz, Hatice Yildiz Durak Year: 2019 Country: Turkey Design: Sequential explanatory mixed methods. 55 students participated from 6th and 7th grade. |
| Curricular Practices | The curricular practices focus on the development of computational thinking (CT) skills through structured educational robotics (ER) activities, guided by an instructional approach based on the constructionist model. | Curricular Practices: It addressed Robotics intervention design, but rather as part of the analysis of variables that explain the development of computational thinking. | Curricular practices focus on the use of CT as a strategy to promote learning about natural selection. Modeling practices, algorithm construction, and step-by-step sequences are integrated to explain scientific processes. | Curricular practices focus on the development of computational thinking through programming self-efficacy, and reflective thinking aimed at problem-solving through programming and robotics activities. |
| Forms of Transversalization of Computational | The study promotes the transversal integration of CT by embedding it in a learning environment that combines | Computational thinking is considered transversal to disciplines such as mathematics, science, and technology. | Computational thinking is integrated into the science domain, especially in biology, to support the understanding of natural | It was developed within the course 'Information Technologies and Software' Although scientific subjects were not |

Figure 4. Excel Spreadsheet for Collected Data Registration. Source: Authors' Own Elaboration.

Collectively, this comprehensive process not only afforded a structured classification of the extracted information but also enabled the precise identification of prevailing trends, predominant approaches, and emergent practices in the integration of computational thinking within secondary education. Subsequent to this, the frequency of occurrence for each category was meticulously reviewed based on its unified code.

Q1: What are the curricular practices being developed and integrated to foster computational thinking in secondary education globally?

Numerous studies have consistently demonstrated that the use of visual programming environments, such as Scratch, MakeCode, or App Inventor, alongside textual languages like Python or JavaScript, constitutes an effective strategy for cultivating CT in secondary school students. These platforms empower students to design sequences, identify intricate patterns, develop robust control structures, and efficiently debug errors—all fundamental skills critical for the advancement of algorithmic thinking and logical problem-solving. As articulated by Peel, Sadler, and Friedrichsen (2021), Çakiroğlu and Selçuk (2024), and Yurdakök and Kalelioğlu (2024), these languages not only facilitate a progressive introduction to CT but also remain highly accessible to students without prior programming experience.

Polat et al. (2021) particularly emphasize that Scratch actively promotes an engaging and motivating learning experience by enabling students to experiment directly with sequencing, conditionals, and loops. Concurrently, Atmatzidou and Demetriadis (2016) and Durak, Karaoglan, and Yilmaz (2019) have demonstrated that the judicious combination of programming with hands-on activities, such as educational robotics, significantly enhances comprehension of complex concepts, including debugging and algorithm design.

From an inclusive pedagogical perspective, Pellas and Peroutseas (2017) provided evidence that visual languages like App Inventor and Scratch are adaptable to diverse cognitive styles, thereby facilitating learning for students with varied educational needs. At a disciplinary level, Sun et al. (2021) and Looi et al. (2023) successfully integrated Python into mathematics classes, observing marked improvements in students' logical thinking, problem structuring, and solution generalization.

In more advanced instructional settings, Campina López et al. (2024) and Çakiroğlu and Selçuk (2024) have illustrated how textual programming enables the exploration of data modeling, machine

learning, and algorithmic problem-solving, thereby extending computational thinking into specialized scientific domains. Furthermore, Yurdakök and Kalelioğlu (2024) demonstrated that integrating programming with artificial intelligence fortifies algorithmic decision-making and deepens the understanding of complex data structures.

Conversely, Polat et al. (2021) employed visual environments within gamified propositions, conclusively demonstrating that challenges incorporating game mechanics significantly augment student participation, refine algorithmic comprehension, and cultivate persistence in the face of errors. These authors, however, prudently caution that the ultimate effectiveness of gamification depends on meticulous instructional design.

Complementarily, a multitude of studies have solidified educational robotics as one of the most effective strategies for fostering the development of computational thinking in secondary education. This practice encompasses investigations that leverage physical or simulated robots as pedagogical instruments to enhance critical skills such as programming, problem-solving, logical reasoning, and abstraction. Works by Atmatzidou and Demetriadis (2016), Looi et al. (2018, 2023), and Sun et al. (2021, 2023) report significant experiences where robotics acts as a direct mediator of computational thinking, facilitating students' development of competencies including problem-solving and algorithmic thinking. In this regard, robotics is presented as a highly motivating and tangible resource that promotes experimentation with algorithms, control structures, and iterative trial-and-error processes—all pivotal aspects for the robust strengthening of computational thinking (Atmatzidou & Demetriadis, 2016).

Similarly, Sun et al. (2024) underscore that the integration of robotics into the curriculum fosters active learning, stimulates peer collaboration, and strengthens the nexus between scholastic knowledge and real-world challenges. These experiential engagements, wherein students systematically design, program, and evaluate the behavior of robots, concurrently cultivate metacognitive skills and advanced problem-solving abilities within authentic or simulated contexts.

Furthermore, studies such as those by Durak, Karaoglan, and Yilmaz (2019) and Lin et al. (2024) broaden this perspective, conclusively demonstrating how educational robotics, when applied in programming projects and sensor control, enhances self-efficacy and cultivates high-level computational competencies. Analogously, Lee and Lee (2021) implemented hands-on activities involving robotic prototypes within STEAM environments, enabling students to seamlessly integrate computational logic with creative and interdisciplinary skills. Kucuk and Sisman (2020) also reported marked improvements in students' attitudes toward programming and in skills such as decomposition and algorithm design through the consistent weekly utilization of educational robots.

Correspondingly, Campina López et al. (2024) and Çakiroğlu and Selçuk (2024) illustrate how robotics-based learning environments facilitate the integration of computational thinking into subjects like mathematics and science, thereby fortifying collaborative work and the resolution of complex problems. Within the Asian context, Looi et al. (2018, 2023) and Sun et al. (2021, 2024) have pioneered structured programs where robotics is strategically positioned as a mediator of learning, with a pronounced emphasis on 21st-century competencies.

From an alternative vantage point, Yurdakök and Kalelioğlu (2024) and Huang and Qiao (2022) expand this panorama by incorporating robotics into artificial intelligence projects. This integration encompasses algorithmic decision-making, data analysis, and optimization, consequently extending the scope of computational thinking towards advanced domains of computation.

In a similar vein, unplugged activities have emerged as a highly efficacious pedagogical alternative for introducing the foundational principles of CT without necessitating technological infrastructure. These activities encompass logic games, dramatizations, manipulative exercises,

physical simulations, and paper-and-pencil problem-solving, all of which enable students to experientially engage with concepts such as algorithms, decomposition, abstraction, and solution evaluation (Sinclair & Patterson, 2018; Schmidthaler et al., 2023).

Numerous studies unequivocally demonstrate that these activities not only facilitate the conceptual appropriation of CT but also robustly promote educational equity, particularly pertinent in contexts characterized by limited connectivity or lack of access to technological resources. Peel, Sadler, and Friedrichsen (2019) meticulously designed a sequence of unplugged activities to teach natural selection through algorithms represented without digital technology, thereby evidencing their potent capacity to integrate CT into fields such as natural sciences. Analogously, Montes-León et al. (2020) and Çelik and Özdener (2023) conducted comparative studies of unplugged and digital activities, demonstrating that both modalities positively foster the development of computational skills.

Sun et al. (2021) underscore that these activities function as crucial cognitive mediators preceding formal programming, assisting students in internalizing logical structures before engaging with languages like Python. Schmidthaler et al. (2023) assert that these practices democratize access to CT by obviating the need for costly devices, rendering them exceptionally valuable in rural or marginalized environments.

However, Yurdakök and Kalelioğlu (2024) caution that, for optimal effectiveness, these activities must be seamlessly integrated into a structured curricular framework that coherently links analog concepts with digital applications, thus ensuring appropriate cognitive progression. In this same vein, Durak and Saritepeci (2018) and Çakiroğlu and Selçuk (2024) extend the discussion by illustrating how unplugged practices can be judiciously complemented with textual programming and machine learning, thereby elevating CT development to more advanced levels. Collectively, the empirical evidence substantiates that unplugged activities represent a potent, accessible, and adaptable pedagogical strategy capable of fostering fundamental computational thinking skills, provided they are meticulously planned, coherently articulated, and expertly mediated by the educator.

The majority of reviewed studies consistently concur that active "doing" constitutes the central unifying axis of pedagogical proposals aimed at fostering CT development. Within this framework, the design and construction of prototypes have gained substantial prominence as a pedagogical strategy, empowering students to conceive, fabricate, and evaluate physical, digital, or hybrid artifacts to resolve concrete problems. This practice synergistically integrates technical and creative knowledge, cultivates iterative thinking, and promotes collaborative engagement within active and meaningful learning environments (Lin et al., 2024; Huang & Qiao, 2022).

This category encompasses projects where students, prompted by a specific challenge or need, design and construct functional solutions that necessitate both physical development and the integration of programming and automation. Sinclair and Patterson (2018) demonstrated that working with tangible materials, without the need for digital interfaces, enables students to develop computational capacities such as iteration, conditional logic, and solution evaluation through hands-on manipulation and practical design.

Lin et al. (2024) and Huang and Qiao (2022) further elaborate on this approach, providing compelling evidence that the prototyping process significantly propels the development of skills such as problem decomposition, abstraction of functionalities, automation through sensors and programming, and iterative solution evaluation. Notably, Lin et al. (2024) reported that, through the design of autonomous robotic vehicles, students effectively consolidated key CT competencies including problem analysis, algorithm construction, and real-time solutioning. Concurrently, Huang and Qiao (2022) integrated artificial intelligence into these processes, thereby empowering students to make algorithmic decisions predicated on data, consequently broadening the scope of CT towards

predictive modeling and physical computing. Table 2 provides a summary of the most commonly used curricular practices in the development of CT.

Table 2. Thematic Coding Table of Identified Curricular Practices.

| Main Categories | Descriptive Code | Unified Code | Frequency |
|----------------------|--|-------------------------|-----------|
| Curricular Practices | Educational Robotics | Robotics | 22 |
| | Block-based Programming (Scratch, Blockly) | Programming | 24 |
| | Text-based Programming (Python, Java) | | |
| | Unplugged Activities | Unplugged Activities | 17 |
| | Design and Construction of Artifacts (prototypes, simple machines) | Design and Construction | 13 |
| | STEAM Projects (integration of Science, Technology, Engineering, Art, and Mathematics) | STEAM | 2 |
| | Gamification applied to computational thinking development | Gamification/Games | 15 |
| | Curricular integration of programming in subjects (mathematics, science, technology) | Curricular Integration | 2 |

The integration of CT within secondary education has materialized through a diverse array of pedagogical practices that judiciously combine methodological approaches with both technological and analog tools. The systematic analysis of the reviewed studies reveals a predominant trend towards the utilization of educational robotics tools, particularly kits such as Lego Mindstorms NXT and EV3. These are primarily employed for the design, programming, and construction of prototypes, thereby effectively promoting problem-solving competencies (Atmatzidou & Demetriadis, 2016; Durak et al., 2019; Kucuk & Sisman, 2020; Çınar & Tüzün, 2021). Such tools facilitate student interaction with sensors, motors, and physical structures that concretely represent algorithms and computational processes, thereby enabling the transfer of abstract concepts into tangible scenarios.

Another notable cluster of identified tools comprises programming platforms, encompassing both text-based and block-based environments like Scratch and App Inventor. These platforms empower students to cultivate computational logic skills through engaging creative coding activities (Durak & Saritepeci, 2018; Montes-León et al., 2020; Lapawi & Husnin, 2020; Çelik & Özdener, 2023; Theodoropoulos, 2022). These accessible platforms concurrently foster motivation, particularly when seamlessly integrated into interdisciplinary projects within subjects such as science, mathematics, and the humanities.

The strategic deployment of unplugged tools, including physical balances, manual spreadsheets, or simple tangible materials, has been underscored in studies by Peel et al. (2019) and Looi et al. (2018). These non-electronic strategies are instrumental in cultivating algorithmic thinking without requiring digital devices.

Furthermore, several studies advocate for the application of mathematical models and simulations, such as structural equation models or simulations of natural processes, to establish a robust connection between computational thinking and the comprehension of scientific phenomena

(Durak & Saritepeci, 2018; Peel et al., 2019). These tools actively promote data analysis, modeling, and outcome prediction, thereby strengthening both scientific and computational competencies in an integrated manner.

The curricular practices also encompass the development of interdisciplinary projects under STEM and STEAM frameworks. In these initiatives, students synergistically combine principles from science, technology, engineering, art, and mathematics to address real-world problems, often utilizing tools such as robotics kits, programming software, and construction materials (Looi et al., 2018; Durak et al., 2019). These integrated strategies not only cultivate technical skills but also foster crucial soft skills, including teamwork, creativity, and effective communication.

In summary, the analysis of curricular practices (see Table 2) reveals that the learning of CT in secondary education focuses on programming, developing technical skills such as sequencing, debugging, and algorithm design, but with the risk of being reduced to an instrumental competency. Although educational robotics and programming offer relevant pedagogical experiences, they require specific resources and teacher training, which are often absent in vulnerable contexts. This can limit the promotion of metacognition, interdisciplinarity, and creative problem-solving, thereby restricting CT to a technical skill rather than a mode of thinking.

The diversity of tools used to strengthen the CT can be seen in Table 3.

Table 3. Selected Studies, Tools, and Practical Interventions.

| Author and Year | Tools | Interventions—Related Practices |
|-----------------------------------|---|---|
| Altanis et al. (2018) | Lego Mindstorms NXT 2.0 educational robotics kit, sensors (touch, sound, light, ultrasonic), motors, Lego NXT-G programming environment, worksheets, questionnaires | Robotics, Programming, Curricular Integration, Design & Construction, STEM |
| Atmatzidou and Demetriadis (2016) | Structural equation models, computational thinking skills scale | Programming, Curricular Integration, STEM, Unplugged Activities |
| Çakiroğlu and Selçuk (2024) | Algorithmic explanations without devices (unplugged) | Unplugged Activities, Curricular Integration |
| Campina López et al. (2024) | Educational robotics activities with sensors, motors, Scratch for Arduino, Arduino | Robotics, Programming, Design & Construction, Curricular Integration, STEM |
| Cárdenas-Sainz et al. (2022) | Unplugged activities with scales, worksheets, pseudocode, flowchart, Python, video recordings | Unplugged Activities, Curricular Integration, Programming |
| Çelik and Özdener (2023) | Lego WeDo 2.0, Lego Mindstorms EV3, tablets, design guides | Robotics, Programming, Curricular Integration, STEM, Design & Construction |
| Chan et al. (2021) | Unplugged programming (CS Unplugged), physical activities, logic games, cards, interactive whiteboards, Python | Unplugged Activities, Curricular Integration, Gamification, Programming, STEM |

| Author and Year | Tools | Interventions—Related Practices |
|-----------------------------|---|--|
| Chen et al. (2023) | Geometer's Sketchpad (GSP), geometric machine design activities | Programming, Curricular Integration, Design & Construction, STEM, Unplugged Activities |
| Çınar and Tüzün (2021) | Scratch 3.0, coding projects, CTt (Computational Thinking Test), CTLS (CT Perception Scale) | Programming, Curricular Integration, Gamification, Design & Construction, STEM, Unplugged Activities |
| Durak et al. (2019) | Scratch, Kinect2Scratch, Microsoft Kinect, tangible gesture cards, rubrics, questionnaires, storyboards | Gamification, Programming, Curricular Integration, Design & Construction, STEM, Unplugged Activities |
| Durak and Saritepeci (2018) | Scratch, game-based environment | Programming, Gamification, Design & Construction, STEM |
| Fojtík et al. (2023) | Scratch4SL, Second Life, 2D and 3D visual coding tools | Curricular Integration, Design & Construction, Programming, Gamification, STEM |
| He et al. (2021) | AI projects, prototypes, presentations | Curricular Integration, STEAM, Design & Construction, Programming |
| Huang and Qiao (2022) | Object-oriented programming (C#), robotic programming with Lego Mindstorms NXT 2.0 and RobotC | STEM, Curricular Integration, Programming, Design & Construction, Robotics |
| Kucuk and Sisman (2020) | Unplugged activities, mathematical exercises, logic exercises, platforms like Scratch and Lightbot | Gamification, Unplugged Activities, Programming, Curricular Integration, Design & Construction, STEM |
| Lapawi and Husnin (2020) | Unplugged activities, spreadsheets, diagrams, worksheets | Unplugged Activities, Programming, Curricular Integration |
| Lee and Lee (2021) | Scratch, Code.org, unplugged activities (CS Unplugged and educational board games) | Programming, Unplugged Activities, Gamification, Curricular Integration |
| Lin et al. (2024) | Programmable Logic Controllers (PLC) | Curricular Integration, Programming, Design & Construction, STEM |
| Looi et al. (2023) | Hands-on programming activities, algorithm development | Programming, Curricular Integration |

| Author and Year | Tools | Interventions—Related Practices |
|------------------------------|---|--|
| Looi et al. (2018) | Micro:bit, Microsoft MakeCode (blocks/JavaScript), MicroPython | Programming, Robotics, Curricular Integration, Design & Construction, Unplugged Activities, STEM |
| Love and Asempapa (2022) | Sonic Pi for musical composition | Curricular Integration, Programming, Design & Construction, STEM, STEAM, Unplugged Activities |
| Molina et al. (2020) | Small Basic, practical programming activities, oral presentations | Curricular Integration, Programming, Design & Construction, STEM |
| Montes-León et al. (2020) | Unplugged activities for natural selection, bacterial laboratory, Lightbot | Unplugged Activities, Curricular Integration |
| Panskyi et al. (2019) | Physical computing (microcontrollers, sensors) | Programming, Curricular Integration, Unplugged Activities, Design & Construction, Robotics, STEM |
| Peel et al. (2019) | MIT App Inventor, Tynker Debugger Game | Programming, Gamification, Curricular Integration, Design & Construction, STEM |
| Peel et al. (2021) | THINKMOTION (interactive learning environment), Leap Motion Controller, visual block programming, 3D physical simulations, Babylon.js | Programming, Curricular Integration, Design & Construction, STEM |
| Pellas and Peroutseas (2017) | Scratch 3.0, printed modules, animations, visualization activities | Programming, Curricular Integration |
| Petrie (2022) | EarSketch, TunePad, Python programming, block editor, virtual DAW stations, digital music composition tools | Programming, Curricular Integration, Design & Construction |
| Petrie (2023) | BBC micro:bit, block programming, motion sensors | Programming, Design & Construction, STEM |
| Polat et al. (2021) | Explanatory videos, mathematical modeling activities, video production | Curricular Integration, Design & Construction |
| Sanabria et al. (2020) | Poly-Universe (analog game), printed materials, worksheets, digital animations, GeoGebra platform | Unplugged Activities, STEAM, Gamification, Curricular Integration |

| Author and Year | Tools | Interventions—Related Practices |
|--------------------------------|--|---|
| Schmidthaler et al. (2023) | Blockly platform, block-based programming | Programming, Curricular Integration, Gamification, Design & Construction |
| Shahin et al. (2022) | Scratch and Polya's technique for arithmetic problem-solving | Programming, Curricular Integration, Unplugged Activities, Design & Construction, Gamification, STEM |
| Sinclair and Patterson (2018) | Scratch, Code.org, mBot, mBlock (based on Scratch and Python) | Programming, Robotics, Unplugged Activities, Curricular Integration, Design & Construction, STEAM, STEM |
| Sun et al. (2024) | Micro:bit, Python, Bebras challenges | Programming, Robotics, Curricular Integration, Design & Construction, STEM |
| Sun et al. (2021) | Code4all, PyCharm, CoffeeScript, Python | Programming, Curricular Integration, Design & Construction, STEM |
| Sun et al. (2024) | Code4all (Pencil Code-based platform), CoffeeScript (programming language), Python | Programming, Curricular Integration, Design & Construction |
| Theodoropoulos (2022) | AI learning, structural equation models | Curricular Integration, STEM, Unplugged Activities |
| Totan and Korucu (2023) | Machine Learning for Kids, Scratch | Programming, Curricular Integration, Unplugged Activities, STEM, STEAM |
| Yurdakök and Kalelioğlu (2024) | Micro:bit, digital pH sensor, MakeCode (block or JavaScript programming), modeling materials | Curricular Integration, Programming, Design & Construction, STEM, Robotics |

Q1.1: What specific computational thinking contents are addressed in curricular practices?

Computational thinking has been consolidated as an essential competency in secondary education, which has generated a growing interest in identifying the curricular practices that integrate it worldwide. Therefore, it is necessary to focus research on recognizing the specific CT content addressed in these practices. This articulation between pedagogical approaches and conceptual content facilitates the identification of educational trends and guides curricular implementation in diverse secondary school contexts.

In relation to the specific contents of CT, the reviewed studies consistently concur that its development is fundamentally articulated through core concepts such as abstraction, decomposition, algorithmic thinking, sequencing, debugging, and branching. These abilities are broadly recognized as indispensable for effective problem-solving and the construction of efficient solutions across diverse educational contexts.

Authors such as Looi et al. (2018, 2023), Sun et al. (2021, 2023), and Çakiroğlu and Selçuk (2024) particularly emphasize that these contents are explicitly addressed through activities that strategically combine programming with educational robotics. These experiential engagements not only enable students to encode solutions but also to experiment directly with the logic of algorithms in concrete scenarios, thereby facilitating the comprehension of control structures and iterative trial-and-error cycles.

From a more contextualized perspective, Molina et al. (2020) and Sanabria et al. (2020) demonstrate that these contents can be transversally integrated into the resolution of authentic, real-world problems. Their investigations illustrate how students applied skills such as decomposition and sequencing to resolve challenges linked to natural sciences and environmental education, thereby showcasing CT's potential to enhance learning in non-technological domains.

The research by Altanis et al. (2018) further elucidates the value of game-based interactive activities (without physical contact) in developing algorithmic and logical reasoning skills through bodily movement, underscoring that CT can also be addressed via creative and accessible strategies. Complementarily, Fojtík et al. (2023) propose leveraging the BBC micro:bit to foster statistical reasoning, thus connecting CT with applied mathematics and data interpretation.

Additionally, Chen et al. (2023) provide evidence that applying cognitive and metacognitive scaffolding in technical training improves students' mastery of algorithms for programmable logic controllers, extending CT into the industrial sphere. Similarly, He et al. (2021) highlight the importance of incorporating critical reflection as an integral part of CT development, fostering students' capacity to analyze the broader impact of their solutions beyond mere technical functionality.

Other studies, including those by Durak and Saritepeci (2018) and Polat et al. (2021), underscore the pedagogical importance of teaching students to debug their algorithms and optimize their solutions. This process not only strengthens technical competencies but also cultivates cognitive resilience, as it teaches students to perceive errors as opportunities for continuous learning and refinement.

Furthermore, Lee and Lee (2021) and Huang and Qiao (2022) advance these contents to a more sophisticated level by incorporating data modeling, algorithmic analysis, and artificial intelligence. In these environments, students engage with sensors and automated decision-making systems, applying algorithms to resolve complex problems that necessitate data analysis and context-driven decision-making.

Lastly, Looi et al. (2023) and Sun et al. (2024) emphasize that the development of these contents is demonstrably more effective when articulated within active pedagogical approaches such as project-based learning and interdisciplinary integration. This holistic approach not only facilitates technical comprehension but also significantly enhances motivation and the transfer of learning to authentic contexts.

The analysis of the Content category in CT curricular practices within secondary education reveals a focus on abstraction, decomposition, algorithms, and debugging (see Table 4). While these skills are pertinent for foundational programming education, there is a notable lack of emphasis on additional topics such as logical reasoning, modeling, simulation, and artificial intelligence. The restricted presence diminishes the opportunities to integrate CT with advanced reasoning processes, scientific experimentation, and the evolving challenges of the digital society.

The different topics addressed by researchers in developing CT in secondary education are shown in Table 4.

Table 4. Thematic Coding Table of Identified Contents.

| Main Categories | Descriptive Code | Unified Code | Frequency |
|-----------------|---|-------------------------|-----------|
| Contents | Problem Decomposition | Decomposition | 21 |
| | Abstraction | Abstraction | 28 |
| | Algorithms | Algorithms | 16 |
| | Debugging | Debugging | 15 |
| | Pattern Recognition | Patterns | 13 |
| | Logical Thinking | Logical Thinking | 3 |
| | Modeling and Simulation | Modeling | 2 |
| | Simulation | Simulation | 1 |
| | Sensor Programming/Micro:bit and Actuator Programming (in robotics and STEM/STEAM projects) | Sensors | 8 |
| | Introduction to Artificial Intelligence (AI) | Artificial Intelligence | 4 |

Q2: What teaching model is utilized in the implementation of computational thinking in curricular practices?

The analysis of the pedagogical models category shows that the reviewed studies adopt a wide variety of theoretical approaches in teaching computational thinking. The results reveal a high frequency of instructional models based on tasks and projects, and on problem-solving. In this vein, Looi et al. (2023) and Sun et al. (2021) applied programming activities linked to mathematical problem-solving, while Durak, Karaoglan, and Yilmaz (2019) demonstrated that instructional projects strengthen students' self-regulation and algorithmic structuring.

On the other hand, constructionism and constructivism are consolidated as key frameworks. From the constructivist tradition, Piaget (1972) conceives of learning as an active process of cognitive reorganization; this perspective is reflected in experiences that integrate programming into mathematics and sciences (Sun et al., 2021). Papert's (1980) constructionism, for its part, emphasizes "learning by doing" through the creation of artifacts, an approach applied in educational robotics by Atmatzidou and Demetriadis (2016) and in block-based programming by Polat et al. (2021).

Likewise, collaborative learning and student-centered learning, inspired by the sociocultural contributions of Vygotsky (1978) and Bandura (1986), were integrated into studies such as that of Durak and Saritepeci (2018), who showed that collaboration and social interaction favor problem-solving and the acquisition of metacognitive skills in programming environments.

Similarly, the inquiry-based and experiential learning approach, founded on Kolb's (1984) proposal, appears in studies such as those by Peel, Sadler, and Friedrichsen (2019), who designed unplugged activities to promote scientific inquiry, and Lin et al. (2024), who showed how direct experience with robotic prototypes fosters critical reflection and knowledge transfer.

Finally, the socio-cognitive approach, albeit infrequently, appears in works such as Looi et al. (2023), where peer interaction in collaborative problem-solving environments reflects the social and cultural mediation of learning.

In conclusion, the results of the Pedagogical Models category (see Table 5) indicate a high recurrence of problem-solving-based and collaborative learning, highlighting the practical and social orientation of CT. However, less frequent approaches like inquiry, student-centered learning, or constructionism offer critical and reflective dimensions that are rarely explored. The true educational challenge is to consolidate CT as a form of critical and creative reasoning, where problems are not only solved but also formulated and analyzed in a way that enriches learning and strengthens its applicability in secondary education.

The diversity of pedagogical models used by the academic community to strengthen CT development in secondary education is shown in Table 5.

Table 5. Thematic Coding Table of Identified Pedagogical Models.

| Main Categories | Descriptive Code | Unified Code | Frequency |
|--------------------|--|------------------------|-----------|
| Pedagogical Models | Constructivism | Constructivism | 8 |
| | Constructionism | Constructionism | 11 |
| | Sociocognitive Approach | Sociocognitive | 1 |
| | Problem-Based Learning (PBL) | Problem-Based Learning | 2 |
| | Inquiry-Based Learning | Inquiry | 5 |
| | Problem-Solving Based Learning | Problem-Solving | 12 |
| | Collaborative / Cooperative Learning | Collaborative | 8 |
| | Student-Centered Learning | Student-Centered | 5 |
| | Task and Project-Based Instructional Model | Instructional Model | 12 |
| | Experiential Learning | Experiential | 5 |

Q3: How is computational thinking related to other subjects?

Regarding the forms of curricular transversalization of computational thinking (CT), there is clear evidence of a growing trend towards its integration across diverse knowledge domains, beyond technology, particularly within mathematics, natural sciences, biology, and the arts. This strategy not only promotes the development of technical skills but also fosters meaningful learning, knowledge transfer, and prepares students to navigate complex digital environments (Sun et al., 2023; Looi et al., 2023; Campina López et al., 2024).

Kucuk and Sisman (2020) and Polat et al. (2021) document successful experiences where CT is applied in mathematics through programming and problem-solving activities. Similarly, Sanabria et al. (2020) and Campina López et al. (2024) demonstrate how educational robotics facilitates the learning of biology and natural science content. Other authors, such as Chan et al. (2021) and Lapawi and Husnin (2020), highlight the inclusion of CT in mathematical and scientific activities, while Cárdenas-Sainz et al. (2022) and Petrie (2022, 2023) explore its application in interactive environments and musical creation, respectively.

Furthermore, Sun et al. (2023) demonstrate that integrating CT into biology and mathematics through robotics not only strengthens disciplinary learning but also cultivates skills such as decomposition and automation. In this context, the STEM/STEAM approach emerges as an ideal framework for enhancing these practices. By synergistically combining science, technology,

engineering, art, and mathematics, it enables students to address real-world problems from multifaceted perspectives, thereby fostering creativity, innovation, and computational thinking (Looi et al., 2023; Huang & Qiao, 2022).

Concurrently, Sun et al. (2024) emphasize that project-based STEAM environments enhance motivation and meaningful learning by allowing students to apply computational concepts in authentic contexts. Similarly, Looi et al. (2023) report improvements in decomposition, abstraction, and solution evaluation through projects integrating robotics and coding. However, when mainstreaming through programming and educational robotics, questions arise as to whether the development of higher-order thinking processes like analysis, abstraction, or evaluation is being fostered, or if technological tools are simply being used to make traditional content more appealing.

An analysis of the results from the Computational Thinking Integration/Curricular Models category (see Table 6) shows a considerable degree of integration, especially in areas not directly linked to STEM and oriented toward technology, which constitutes a significant step forward. Nevertheless, the lack of a focus on instructional design can limit experiences to isolated activities, restricting their development as a meaningful transversal competency.

Table 6. Thematic Coding Table of Identified Forms of CT Mainstreaming/Curricular Models.

| Main Categories | Descriptive Code | Unified Code | Frequency |
|---|--|-------------------------------|-----------|
| Forms of Computational Thinking Mainstreaming/Curricular Models | Integration into STEM subjects (mathematics, science, technology) | STEM | 3 |
| | Integration into non-computer science subjects (biology, physics, music) | Transversalization | 20 |
| | Integration into formal computer science and technology courses | Computer Science & Technology | 15 |
| | Phased instructional design (introduction, development, evaluation) | Instructional Design | 1 |
| | Integration of computational thinking through virtual environments | Virtual Environments | 1 |

Q4: How is computational thinking learning assessed in curricular practices?

The evaluation strategies reported in the reviewed studies reflect a comprehensive combination of quantitative and qualitative approaches aimed at assessing the impact of pedagogical practices implemented to foster CT in research settings. Among the most prevalent strategies, pre- and post-intervention questionnaires, as utilized by Durak and Saritepeci (2018), prominently feature. These instruments effectively measure shifts in students' computational competencies and perceptions before and after the intervention. Complementarily, Sinclair and Patterson (2018) leverage written artifacts to analyze how students represent and articulate their problem-solving processes, while Looi et al. (2018) meticulously collect performance logs to evaluate the level of engagement and execution during robotics activities.

Sanabria et al. (2020), conversely, propose the analysis of final products generated by students as empirical evidence of their mastery of skills such as decomposition, sequencing, and debugging. Similarly, Yurdakök and Kalelioğlu (2024) and Totan and Korucu (2023) integrate questionnaires with performance analysis, thereby yielding both objective and observational data concerning the efficacy of didactic strategies. Sun et al. (2024) extend this by incorporating mixed methods to compare

behaviors and skills across various programming modalities, offering a more expansive perspective on learning processes.

Some studies introduce innovative assessment approaches by integrating observations of playful and manipulative activities. For instance, Altanis et al. (2018) and Fojtík et al. (2023) focus evaluation on active participation and the application of computational concepts through tools such as physical games or the micro:bit, emphasizing the importance of assessing learning within tangible and meaningful contexts.

On a deeper dimension, Chen et al. (2023) and He et al. (2021) underscore the critical relevance of incorporating the assessment of metacognitive processes. These authors employ reflective journals and qualitative questionnaires to prompt students to critically analyze their decisions, strategies, and learning experiences, consequently strengthening competencies such as critical thinking and informed decision-making.

Furthermore, Çakiroğlu and Selçuk (2024) integrate the evaluation of functional prototypes and algorithmic solutions, considering criteria of efficiency, optimization, and applicability in real-world contexts. This approach allows for the assessment not only of the final solution but also of the qualitative aspects of the design and development process.

In conclusion, the findings for the Computational Thinking Evaluation Strategies category (see Table 7) are varied, yet they reveal a tension between the measurement of immediate results and the authentic development of thinking skills. Diagnostic tests and post-tests are predominant, whereas formative methods such as rubrics, observation, and self-evaluation are less frequent. In this sense, a key concern remains: to what extent does the current emphasis on prioritizing immediate outcomes in evaluation limit the evidence of the development of computational thinking processes, including metacognition?

The diversity of ways in which CT is assessed in secondary education is shown in Table 7.

Table 7. Thematic Coding Table of Identified Evaluation Strategies. Source: Authors' Own Elaboration

| Main Categories | Descriptive Code | Unified Code | Frequency |
|-----------------------|---|-----------------|-----------|
| Evaluation Strategies | Performance Rubrics | Rubrics | 8 |
| | Project Self-Assessment | Self-Assessment | 1 |
| | Product-Based Assessment (prototypes, programs, final projects) | Products | 3 |
| | Competency-Based Assessment | Competencies | 4 |
| | Diagnostic and Final Knowledge Tests on Computational Thinking (pre/post) | Tests | 25 |
| | Systematic Observation of Processes and Participation | Observation | 7 |
| | Recording of Interaction and Collaboration in Group Activities | Interaction Log | 2 |

4. Discussion

This systematic review provides a critical examination of the integration of CT in secondary education, considering it from both empirical and cognitive perspectives, as well as its broader formative implications. The literature underscores its significance as a core competency in modern education (Zhang & Nouri, 2019; Wing, 2006), demonstrating its cross-disciplinary applicability in fields including mathematics, science, arts, and humanities. Nonetheless, a contradiction is evident: while concepts such as abstraction, decomposition, and algorithms are prevalent, the methodology is frequently influenced by programming and robotics. This influence enhances engaging learning experiences but simultaneously constrains the breadth of computational thinking to its practical applications. Tedre and Denning (2021) caution that lacking critical, ethical, and reflective dimensions hinders the conception of a holistic process that integrates cognitive and social aspects.

This limitation is intensified by the fragmentation of the curriculum. Numerous interventions are ephemeral, fragmented, and fail to establish a coherent connection among content, methodologies, and evaluation criteria (Grover & Pea, 2013; McClelland & Grata, 2018). This results in students performing tasks without engaging in computational thinking, thereby missing the chance to strengthen essential metacognitive processes, including analysis, self-reflection, and knowledge transfer.

International organizations such as UNESCO (2021) and the OECD (2022) advocate for the incorporation of computational thinking as a means to enhance higher-order reasoning; however, empirical evidence reveals inconsistencies in its application. Limited research addresses the relationship between computational skills and other knowledge domains, as well as the assessment of their development over time (Atmatzidou & Demetriadis, 2016; Peel et al., 2019; Campina López et al., 2024). The absence of continuity and systematicity undermines the potential for computational thinking to develop as a sustainable and transferable competency.

Ultimately, although active pedagogical models such as project-based learning and problem-solving present a promising framework (Çakiroğlu & Selçuk, 2024; Lin et al., 2024), the review questions the degree to which these approaches succeed in transcending procedural limitations. Their limited pedagogical contextualization restricts their transformative potential. This raises an important question: should we focus on training students to master programming tools and languages, or should we prioritize developing citizens who can think critically, reflect on their learning processes, and apply their knowledge to real-world issues? Integrating practices into the formal curriculum, with active teacher participation (Liu et al., 2024), presents a significant challenge for consolidating computational thinking as a transversal and metacognitive approach rather than merely a temporary technical training.

Despite progress in evaluating CT via diagnostic tests, rubrics, and observations, the review indicates a continued bias toward assessing final products (projects or prototypes) rather than the underlying thought processes (Durak & Saritepeci, 2018; Sun et al., 2024). This tendency prompts a critical inquiry: are we cultivating students who can reflect on their own learning processes, or are we merely developing technical skills that result in tangible outcomes? Restricting assessment to the product diminishes the opportunity to comprehend how students engage in analysis, abstraction, and the application of knowledge to real-world contexts. Chen et al. (2023) emphasize the necessity of adopting instruments that assess both outcomes and the processes involved, integrating reflective and metacognitive aspects.

The findings suggest the necessity for a more structured, equitable, and sustainable curricular framework for computational thinking in secondary education. This suggests the need to develop evaluative strategies that emphasize cognitive processes rather than solely outcomes, while also examining their applicability in low-tech environments, incorporating critical, ethical, and cultural

considerations. This approach will enable the consolidation of computational thinking not merely as a skill, but as a cognitive process that integrates technique with metacognition, thereby transforming the educational experience significantly.

5. Conclusion

Currently, a diverse array of curricular practices is being implemented in secondary education to promote the development of computational thinking (CT). This systematic review has effectively identified and analyzed these practices. Five principal categories were identified as a result of 40 empirical studies: curricular practices, pedagogical models, modalities of CT transversalization, specified contents, and evaluation strategies. These categories include curricular integration, gamification, STEAM projects, design and construction of prototypes, educational robotics, disconnected activities, and block-based and textual programming.

The results clearly indicate that CT is becoming a fundamental and cross-disciplinary ability in secondary education. The examined methods exhibit many methodologies that transcend solely technological fields, integrating smoothly into areas such as mathematics, sciences, arts, and music. These experiences, grounded on active pedagogical paradigms such as project-based learning, problem-solving, and collaborative work, enhance learning and fortify competencies essential for addressing the problems of an increasingly digital and linked society.

However, persistent challenges remain, primarily related to inadequate curricular integration, limited teacher engagement, and insufficient systematization of assessment strategies. These limitations are particularly exacerbated in contexts characterized by low technological availability. Therefore, it is imperative to advance towards more structured, sustainable, and contextually sensitive proposals that integrate CT as a transversal competency across all levels and areas of the curriculum.

The findings of this review indicate the necessity for a structured, equitable, and sustainable curricular approach to computational thinking in secondary education. Three priority lines of research are identified to contribute to achieving this goal: (1) the development of comprehensive curricular frameworks through national guidelines and scalable instructional sequences; (2) the design of evaluative strategies that focus on thinking processes, utilizing rubrics, learning journals, and contextualized assessments; and (3) the promotion of contextualized practices in low-tech environments, emphasizing unplugged activities and real-world application problems. A comprehensive vision, supported by strategic and practical actions, is essential for providing meaningful, transformative, and contextualized computational education for all students.

Declarations

Author Contributions. Yaneth Huertas conceived and designed the study, performed the material preparation, data collection, and analysis, and wrote the manuscript. Oscar Boude participated in the analysis, wrote, reviewed, and edited the initial versions, and along with Ana Vargas, he reviewed and approved the final version. Ana Vargas reviewed and approved the final version of the manuscript

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

Funding. This study is part of the Educational Innovation Observatory project, which aims to determine the effects of integrating educational innovation on student learning, educational institutions, pedagogical practices, and teacher training in the country. It received funding from the University of La Sabana under project number EDU-94-2025.

Ethical Approval. The authors declare no conflicts of interest in this research.

Data Availability Statement. Raw data supporting this study's findings were generated during the research process and are available from the author upon reasonable request. Derived data that underpin the results of this study can also be provided upon request.

Generative AI Usage. During the preparation of this work the author(s) used [GEMINI] in order to [IMPROVE THE WORDING OF THE TEXT AND THE USE OF ACADEMIC LANGUAGE]. The author(s) declare that they reviewed and edited the final output as needed and take(s) full responsibility for the content of the published article.

References

- Altanis, I., Retalis, S., & Petropoulou, O. (2018). Systematic design and rapid development of motion-based touchless games for enhancing students' thinking skills. *Education Sciences*, 8(1), 18. <https://doi.org/10.3390/educsci8010018>
- Atmatzidou, S., & Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. *Robotics and Autonomous Systems*, 75, 661–670. <https://doi.org/10.1016/j.robot.2015.10.008>
- Bandura, A. (1986). *Social foundations of thought and action: A social cognitive theory*. Prentice Hall. <https://moretech.technion.ac.il/files/2015/07/Bandura-Theory.pdf>
- Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12. *ACM Inroads*, 2(1), 49–54. <https://doi.org/10.1145/1929887.1929905>
- Bocconi, S., Chiocciariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., & Punie, Y. (2016). *Developing computational thinking in compulsory education: Implications for policy and practice*. European Commission, Joint Research Centre. <https://doi.org/10.2791/792158>
- Çakiroğlu, Ü., & Selçuk, V. (2024). Machine learning meets secondary school classrooms: Using hands-on activities to advance computational thinking. *Education and Information Technologies*. Advance online publication. <https://link.springer.com/article/10.1007/s10639-024-13196-8>
- Cárdenas-Sainz, B. A., Barrón-Estrada, M. L., Zatarain-Cabada, R., & Ríos-Félix, J. M. (2022). Integration and acceptance of Natural User Interfaces for interactive learning environments. *International Journal of Child-Computer Interaction*, 31, 100381. <https://doi.org/10.1016/j.ijcci.2021.100381>
- Çelik, A. K., & Özdener, N. (2023). A comparison of plugged and unplugged tools in teaching algorithms at the K-12 level for computational thinking skills. *Technology, Knowledge and Learning*, 28(4), 1485–1513. <https://doi.org/10.1007/s10758-021-09585-4>
- Chan, S.-W., Looi, C.-K., Ho, W. K., Seow, P., & Wu, L. (2021). Learning number patterns through computational thinking activities: A Rasch model analysis. *Heliyon*, 7(9), e07922. <https://doi.org/10.1016/j.heliyon.2021.e07922>
- Chen, C.-H., Liu, T.-K., & Huang, K. (2023). Scaffolding vocational high school students' computational thinking with cognitive and metacognitive prompts in learning about programmable logic controllers. *Journal of Research on Technology in Education*, 55(3), 527–544. <https://doi.org/10.1080/15391523.2021.1983894>
- Çınar, M., & Tüzün, H. (2021). Comparison of object-oriented and robot programming activities: The effects of programming modality on student achievement, abstraction, problem solving, and motivation. *Journal of Computer Assisted Learning*, 37(2), 370–386. <https://doi.org/10.1111/jcal.12495>
- Durak, H. Y., & Saritepeci, M. (2018). Analysis of the relation between computational thinking skills and various variables with the structural equation model. *Computers and Education*, 116, 191–202. <https://doi.org/10.1016/j.compedu.2017.09.004>

- Durak, H. Y., Karaoglan, Y. F., & Yilmaz, B. R. (2019). Computational thinking, programming self-efficacy, problem solving and experiences in the programming process conducted with robotic activities. *Contemporary Educational Technology, 10*(2), 173–197.
- Fojtík, M., Cápaj, M., Medová, J., & Valovičová, L. (2023). Activities with BBC micro as a foundation for statistical reasoning of lower-secondary students. *Mathematics, 11*(14), 3206. <https://doi.org/10.3390/math11143206>
- Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher, 42*(1), 38–43. <https://doi.org/10.3102/0013189X12463051>
- Grover, S., & Pea, R. (2018). Computational thinking: A competency whose time has come. In S. Sentance, E. Barendsen, & C. Schulte (Eds.), *Computer science education: Perspectives on teaching and learning in school* (pp. 19–38). Bloomsbury Academic. <http://dx.doi.org/10.5040/9781350057142.ch-003>
- Haddaway, N. R., Page, M. J., Pritchard, C. C., & McGuinness, L. A. (2022). PRISMA2020: PRISMA2020: An R package and Shiny app for producing PRISMA 2020-compliant flow diagrams, with interactivity for optimised digital transparency and open synthesis. *Campbell Systematic Reviews, 18*, e1230. <https://doi.org/10.1002/cl2.1230>
- He, Z., Wu, X., Wang, Q., & Huang, C. (2021). Developing eighth-grade students' computational thinking with critical reflection. *Sustainability (Switzerland), 13*(20), 11192. <https://doi.org/10.3390/su132011192>
- Huang, X., & Qiao, C. (2024). Enhancing computational thinking skills through artificial intelligence education at a STEAM high school. *Science & Education, 33*(2), 383–403. <https://doi.org/10.1007/s11191-022-00392-6>
- Kucuk, S., & Sisman, B. (2020). Students' attitudes towards robotics and STEM: Differences based on gender and robotics experience. *International Journal of Child-Computer Interaction, 23–24*, 100167. <https://doi.org/10.1016/j.ijcci.2020.100167>
- Lapawi, N., & Husnin, H. (2020). The effect of computational thinking module on achievement in science. *Science Education International, 31*(2), 164–171.
- Lee, M., & Lee, J. (2021). Enhancing computational thinking skills in informatics in secondary education: The case of South Korea. *Educational Technology Research and Development, 69*(5), 2869–2893. <https://doi.org/10.1007/s11423-021-10035-2>
- Lin, X.-F., Zhou, Y., Shen, W., Xian, X., & Pang, B. (2024). Modeling the structural relationships among Chinese secondary school students' computational thinking efficacy in learning AI, AI literacy, and approaches to learning AI. *Education and Information Technologies, 29*(5), 6189–6215. <https://link.springer.com/article/10.1007/s10639-023-12029-4>
- Liu, Z., Gearty, Z., Richard, E., Orrill, C. H., Kayumova, S., & Balasubramanian, R. (2024). Bringing computational thinking into classrooms: A systematic review on supporting teachers in integrating computational thinking into K-12 classrooms. *International Journal of STEM Education, 11*(51). <https://doi.org/10.1186/s40594-024-00510-6>
- Looi, C.-K., Chan, S.-W., Wu, L., Huang, W., Kim, M. S., & Sun, D. (2024). Exploring computational thinking in the context of mathematics learning in secondary schools: Dispositions, engagement and learning performance. *International Journal of Science and Mathematics Education, 22*(4), 993–1011. <https://doi.org/10.1007/s10763-023-10419-1>

- Looi, C.-K., How, M.-L., Longkai, W., Seow, P., & Liu, L. (2018). Analysis of linkages between an unplugged activity and the development of computational thinking. *Computer Science Education, 28*(3), 255–279. <https://doi.org/10.1080/08993408.2018.1533297>
- López, A. C. C., Marín, A. A. L., De las Heras Pérez, M. Á., & Stepanovic, M. B. (2024). The concept of pH and its logarithmic scale. *Journal of Science Education, 29*(6), 771–803.
- Love, T. S., & Asempapa, R. S. (2022). A screen-based or physical computing unit? Examining secondary students' attitudes toward coding. *International Journal of Technology and Design Education, 32*(2), 1173–1194. <https://doi.org/10.1016/j.ijcci.2022.100543>
- Lu, C., Macdonald, R., Odell, B., Kokhan, V., Epp, D. C., & Cutumisu, M. (2022). A scoping review of computational thinking assessments in higher education. *Journal of Computing in Higher Education, 34*(2), 416–461. <https://doi.org/10.1007/s12528-021-09305-y>
- McClelland, K., & Grata, L. (2018). A review of the importance of computational thinking in K-12. *Proceedings of the Tenth International Conference on Mobile, Hybrid, and On-line Learning (eLmL 2018)*, 32–34. <https://www.researchgate.net/publication/344714364>
- Molina, Á., Adamuz, N., & Bracho, R. (2020). La resolución de problemas basada en el método de Polya usando el pensamiento computacional y Scratch con estudiantes de Educación Secundaria. *Aula Abierta, 49*(1), 83–90. <https://doi.org/10.17811/rifie.49.1.2020.83-90>
- Montes-León, H., Hijón-Neira, R., Pérez-Marín, D., & Montes-León, S. R. (2020). Improving computational thinking in secondary students with unplugged tasks. *Education in the Knowledge Society, 21*, eks20202124. <https://doi.org/10.14201/eks.23002>
- Nowell, L. S., Norris, J. M., White, D. E., & Moules, N. J. (2017). Thematic analysis: Striving to meet the trustworthiness criteria. *International Journal of Qualitative Methods, 16*(1). <https://doi.org/10.1177/1609406917733847>
- Obando-Montoya, J. D., Valencia-Cardenas, M., Romero-Díaz, C. H., & Reyes-Alvarado, S. (2024). Categorías y prácticas implicadas con el pensamiento computacional para la mejora de las habilidades en la resolución de problemas matemáticos en secundaria. *Aibi revista de investigación, administración e ingeniería, 12*(2), 173-181. <https://doi.org/10.15649/2346030X.4408>
- Organisation for Economic Co-operation and Development (OCDE). (2022). *The state of the field of computational thinking in early childhood education*. OECD Publishing. <https://www.oecd.org/publications/the-state-of-the-field-of-computational-thinking-in-early-childhood-education-3354387a-en.htm>
- Panskyi, T., Rowinska, Z., & Biedron, S. (2019). Out-of-school assistance in the teaching of visual creative programming in the game-based environment: Case study: Poland. *Thinking Skills and Creativity, 34*, 100593. <https://doi.org/10.1016/j.tsc.2019.100593>
- Papert, S. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*. New York: Basic Books. https://worrydream.com/refs/Papert_1980_-_Mindstorms,_1st_ed.pdf
- Peel, A., Sadler, T. D., & Friedrichsen, P. (2019). Learning natural selection through computational thinking: Unplugged design of algorithmic explanations. *Journal of Research in Science Teaching, 56*(7), 983–1007. <https://doi.org/10.1002/tea.21545>
- Peel, A., Sadler, T. D., & Friedrichsen, P. (2021). Using unplugged computational thinking to scaffold natural selection learning. *American Biology Teacher, 83*(2), 112–117. <https://doi.org/10.1525/abt.2021.83.2.112>

- Pellas, N., & Peroutseas, E. (2017). Leveraging Scratch4SL and Second Life to motivate high school students' participation in introductory programming courses: Findings from a case study. *New Review of Hypermedia and Multimedia*, 23(1), 51–79. <https://doi.org/10.1080/13614568.2016.1152314>
- Petrie, C. (2022). Interdisciplinary computational thinking with music and programming: A case study on algorithmic music composition with Sonic Pi. *Computer Science Education*, 32(2), 260–282. <https://doi.org/10.1080/08993408.2021.1935603>
- Petrie, C. (2023). Design and use of domain-specific programming platforms: Interdisciplinary computational thinking with EarSketch and TunePad. *Computer Science Education*, 34(4), 645–678. <https://doi.org/10.1080/08993408.2023.2240657>
- Petticrew, M., & Roberts, H. (2006). *Systematic reviews in the social sciences: A practical guide*. Blackwell Publishing. <https://doi.org/10.1002/9780470754887>
- Piaget, J. (1972). *The principles of genetic epistemology*. New York: Basic Books.
- Polat, E., Hopcan, S., Kucuk, S., & Sisman, B. (2021). A comprehensive assessment of secondary school students' computational thinking skills. *British Journal of Educational Technology*, 52(5), 1965–1980.
- Quiroz-Vallejo, D. A., Carmona-Mesa, J. A., Castrillón-Yepes, A., & Villa-Ochoa, J. A. (2021). Integración del pensamiento computacional en la educación primaria y secundaria en Latinoamérica: Una revisión sistemática de literatura. *RED. Revista de Educación a Distancia*, 21(68), Artículo 7. <https://doi.org/10.6018/red.485321>
- Saad, A., & Zainudin, S. (2024). A review of teaching and learning approach in implementing Project-Based Learning (PBL) with Computational Thinking (CT). *Interactive Learning Environments*, 32(10), 7622–7646. <https://doi.org/10.1080/10494820.2024.2328280>
- Sanabria, E., Rodríguez, N., Zerpa, A. E., Prieto, P. L., & Alonso, M. Á. (2020). Computational thinking: A new way to train working memory? | El pensamiento computacional: ¿Una nueva forma de entrenar la memoria de trabajo? *Revista de Educación a Distancia*, 20(63), 2.
- Schmidthaler, E., Schalk, M., Schmollmüller, M., Lavicza, Z., & Sabitzer, B. (2023). The interdisciplinary implementation of poly-universe to promote computational thinking: Teaching examples from biological, physical, and digital education in Austrian secondary schools. *Frontiers in Psychology*, 14, 1139884.
- Shahin, M., Gonsalvez, C., Whittle, J., Li, L., & Xia, X. (2022). How secondary school girls perceive computational thinking practices through collaborative programming with the micro. *Journal of Systems and Software*, 183, 111107.
- Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, 22, 142–158. <https://doi.org/10.1016/j.edurev.2017.09.003>
- Sinclair, N., & Patterson, M. (2018). The dynamic geometrisation of computer programming. *Mathematical Thinking and Learning*, 20(1), 54–74. <https://doi.org/10.1080/10986065.2018.1403541>
- Snyder, H. (2019). Literature review as a research methodology: An overview and guidelines. *Journal of Business Research*, 104, 333–339. <https://doi.org/10.1016/j.jbusres.2019.07.039>

- Sun, D., Looi, C.-K., Li, Y., Zhu, C., & Cheng, M. (2024). Block-based versus text-based programming: A comparison of learners' programming behaviors, computational thinking skills, and attitudes toward programming. *Educational Technology Research and Development*, 72(2), 1067–1089. <https://link.springer.com/article/10.1007/s11423-023-10328-8>
- Sun, D., Ouyang, F., Li, Y., & Zhu, C. (2021). Comparing learners' knowledge, behaviors, and attitudes between two instructional modes of computer programming in secondary education. *International Journal of STEM Education*, 8(1), 54. <https://doi.org/10.1186/s40594-021-00311-1>
- Sun, D., Zhu, C., Xu, F., Ouyang, F., & Cheng, M. (2023). Transitioning from introductory to professional programming in secondary education: Comparing learners' computational thinking skills, behaviors, and attitudes. *Journal of Educational Computing Research*, 62(3), 647–674. <https://journals.sagepub.com/doi/10.1177/07356331231204653>
- Tedre, M., & Denning, P. J. (2021). Computational thinking: A professional and historical perspective. In A. Yadav & U. D. Berthelsen (Eds.), *Computational thinking in education: A pedagogical perspective* (1st ed.). Routledge. <https://doi.org/10.4324/9781003102991>
- Theodoropoulos, A. (2022). Participatory debugging in computing education: Engaging students through game design. *International Journal of Child-Computer Interaction*, 34, 100525.
- Totan, H. N., & Korucu, A. T. (2023). The effect of block based coding education on the students' attitudes about the secondary school students' computational learning skills and coding learning: Blocky sample. *Participatory Educational Research*, 10(1), 443–461. <https://doi.org/10.17275/per.23.24.10.1>
- Triantafyllou, S. A., Sapounidis, T., & Farhaoui, Y. (2024). Gamification and computational thinking in education: A systematic literature review. *Salud, Ciencia y Tecnología – Serie de Conferencias*, 3, 659. <https://doi.org/10.56294/sctconf2024659>
- United Nations Educational, Scientific and Cultural Organization (UNESCO). (2019). *Directrices para la educación en ciencias de la computación*. UNESCO Publishing. <https://unesdoc.unesco.org/ark:/48223/pf0000385091>
- Voogt, J., Fisser, P., Good, J., Mishra, P., & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies*, 20(4), 715–728. <https://doi.org/10.1007/s10639-015-9412-6>
- Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes*. Harvard University Press. <https://saberrespsi.files.wordpress.com/2016/09/vygostki-el-desarrollo-de-los-procesos-psicolc3b3gicos-superiores.pdf>
- Wing, J. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35. <https://www.cs.cmu.edu/~15110-s13/Wing06-ct.pdf>
- Wing, J. (2016). Computational thinking, 10 years later. *Communications of the ACM*. <https://cacm.acm.org/blogcacm/computational-thinking-10-years-later/>
- Yeni, S., Grgurina, N., Saeli, M., Hermans, F., Tolboom, J., & Barendsen, E. (2024). Interdisciplinary integration of computational thinking in K-12 education: A systematic review. *Informatics in Education*, 23(1), 223–278. <https://doi.org/10.15388/infedu.2024.08>
- Yurdakök, E. A., & Kalelioğlu, F. (2024). The effect of teaching physical programming on computational thinking skills and self-efficacy perceptions towards computational thinking. *Journal of Educational Computing Research*, 62(3), 785–815. <https://journals.sagepub.com/doi/10.1177/07356331231220313>

- Zhang, L. C., & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K-9. *Computers and Education*, 141, 103607. <https://doi.org/10.1016/j.compedu.2019.103607>
- Zhao, W., & Shute, V. J. (2019). Can playing a video game foster computational thinking skills? *Computers & Education*, 141, 103633. <https://doi.org/10.1016/j.compedu.2019.103633>

About the Contributor(s)

Yaneth Huertas, PhD student in Education, belonging to the research line of transformation and pedagogical practices with ICT. She has published in leading international journals and authored chapters on education.

Email: yanethhuca@unisabana.edu.co

Oscar Boude, PhD, is Professor of Educational Technology at the University of La Sabana in Colombia. He is also the director of the "Technologies for Academia" research group and director of the "Transformation of Pedagogical Practices with ICT" research line. His main research interests include the development of educational technology, the design of video games and gamified environments, as well as game-based learning. He has published extensively in leading international journals and authored books and chapters on education.

Email: oscarbf@unisabana.edu.co

ORCID: <https://orcid.org/0000-0002-7414-2664>

Ana Vargas, PhD, is Professor of Educational Technology at the University of La Sabana in Colombia. She is also the Head of Faculty for the Faculty of Education. Her main research interests are school peace and coexistence. She has published extensively in leading international journals and authored books and chapters on education.

Email: ana.vargas@unisabana.edu.co

ORCID: <https://orcid.org/0000-0002-5633-0901>

Publisher's Note: *The opinions, statements, and data presented in all publications are solely those of the individual author(s) and contributors and do not reflect the views of Universitepark, EDUPIJ, and/or the editor(s). Universitepark, the Journal, and/or the editor(s) accept no responsibility for any harm or damage to persons or property arising from the use of ideas, methods, instructions, or products mentioned in the content.*
