IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 18 July 2024, accepted 19 August 2024, date of publication 23 August 2024, date of current version 3 September 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3449228

RESEARCH ARTICLE

Developing Programming Competencies in School-Students With Block-Based Tools in Chile, Ecuador, and Peru

CRISTIAN VIDAL-SILVA[1], **JESENNIA CÁRDENAS-COBO**[2], **MIGUEL TUPAC-YUPANQUI**[3], **JORGE SERRANO-MALEBRÁN**[4], **AND AURORA SÁNCHEZ ORTIZ**[4]

[1]Escuela de Ingeniería en Desarrollo de Videojuegos y Realidad Virtual, Facultad de Ingeniería, Universidad de Talca, Talca 3460000, Chile
[2]Facultad de Ciencias e Ingenierías, Universidad Estatal de Milagro, Milagro 091050, Ecuado
[3]EAP Ingeniería de Sistemas e Informática, Universidad Continental, Huancayo 12000, Peru
[4]Departamento de Administración, Universidad Católica del Norte, Antofagasta 1240000, Chile

Corresponding author: Cristian Vidal-Silva (cvidal@utalca.cl)

ABSTRACT The information society is a reality nowadays, and computational thinking has become a relevant competence for everybody, regardless of age, social status, and primary activity. Information society is everywhere in contemporary life, and algorithmic thinking represents a significant competency for individuals, irrespective of their educational background and social condition. Developing and applying programming competencies represents a high-value know-how ability. Block-based coding and design tools like Scratch and TinkerCAD Arduino allow people to successfully build programming competencies in online environments regardless of age and social status. This article presents empirical evidence of the positive impact of the block-based programming language Scratch and the design tool TikerCAD Arduino in practical workshops to develop computational thinking with school children, school teachers, and university students. The results permit finding patterns, and almost transversal teaching approaches to build an elementary computational thinking competency applying Scratch and TinkerCAD Arduino, with a block-based approach in both tools and textual programming in the second one. The motivation and wishes of learning in all participants were hegemonic. Those results demonstrate the positive impact of Scratch and TinkerCAD Arduino on developing computational thinking competencies without restrictions. This work shows the application of Scratch and TinkerCAD Arduino in non-WEIRD contexts and, during the pandemic time, demonstrates the relevance of online education. The results show that developing programming competencies with Scratch and TinkerCAD Arduino motivated students' autonomy and motivation for learning regardless of their education level and status. Those results encourage us to continue using Scratch and TinkerCAD Arduino to develop programming competencies without considering age and education level.

INDEX TERMS Programming competencies, school students, children, professors, scratch, TinkerCAD Arduino, online education.

I. INTRODUCTION

In the current information society, computational thinking is a critical competency relevant to people from various fields and ages [1], [2], [3]. As Groover and Pea [4] describe, it involves problem-solving skills and logical thinking patterns that enable individuals to break down complex problems into manageable steps. Likewise, the work of Wing [5] remarks that computational thinking is the "thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that an information-processing agent can effectively carry out." This approach goes beyond programming and can be applied in everyday life.

The associate editor coordinating the review of this manuscript and approving it for publication was Yilun Shang.

© 2024 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

A substantial body of literature highlights the positive impact of hands-on programming activities on academic performance, cognitive skills, and the development of computational thinking [6], [7], [8], [9]. Studies have shown that engaging with tools like Scratch and TinkerCAD Arduino enhances students' problem-solving abilities and fosters critical thinking and creativity [10], [11]. These activities are particularly effective in improving understanding of abstract concepts and developing algorithmic thinking, essential skills in the information society. Educators can significantly boost students' academic achievements and cognitive development by incorporating these practical programming exercises into educational curricula.

Computational thinking is increasingly recognized as a crucial skill for the 21st century, and its development in children has garnered significant attention [12]. This skill involves problem-solving, algorithmic thinking, and the ability to break down complex tasks into smaller, manageable parts [13]. As Wing [14] noted, "Computational thinking is a fundamental skill for everyone, not just for computer scientists." Similarly, Lai [15] remarks that by nurturing computational thinking competencies in children, we usually equip them with problem-solving skills not limited to computer science but extend to a wide range of real-world scenarios [15]. This skill fosters logical reasoning, creativity, and a systematic approach to complex problems, which can be applied in academic and everyday life activities.

Developing computational thinking competencies among school teachers across various subject areas is increasingly relevant in the modern educational landscape [16]. As technology advances, it becomes crucial for educators to not only teach traditional subjects but also equip students with the problem-solving skills associated with computational thinking [17]. This integration would permit preparing students for the challenges of the digital age and foster critical thinking. For example, according to Wing [14], computational thinking involves breaking down complex problems into smaller, manageable parts, which can be applied to subjects like mathematics, science, and even the humanities. In this way, teachers from various fields should possess computational thinking competencies and teach these skills to their students, fostering a multidisciplinary approach to education that transcends traditional boundaries [18], [19].

For computing and information science-related majors university students, overall, for those who did not develop those competencies previously, developing computational thinking is crucial from the first study year because they continue applying that competence in subsequent courses [20].

Nowadays, software technology exists for developing programming competencies in children and adults without great restrictions, regardless of their educational background [2], [21].

The application of computational thinking is becoming increasingly pervasive in a wide range of domains [22]. In engineering [23], mathematics [24], biology [25], economics [26], and social sciences [27], computational thinking helps to analyze data, model systems, and optimize processes [14], [28]. Hence, developing computational thinking is necessary nowadays, and tools exist for that purpose. This work describes workshop sessions by applying Scratch and TinkerCAD Arduino, developing computational thinking, and highlighting the positive results in the diversity of participants. In each workshop session, the participants, school students, school teachers, and university students from different countries continue applying the tools and advancing in their domain, particularly university students.

This study aims to investigate the impact of Scratch and TinkerCAD Arduino on the development of computational thinking among participants spanning diverse age groups. The study population consists of three distinct cohorts: children (under 12 years old), adolescents (between 12 and 18 years old), and adults (over 30 years old) in Chile, Ecuador, and Peru. By examining how these age groups engage with block-based programming tools in various educational environments, we seek to provide insights into practical strategies for promoting programming skills across different developmental stages, cultural conditions, and academic backgrounds.

This article measures and evaluates the development of programming competencies through a validated instrument [29] looking to generate a long-term impact on people who like to develop computational thinking in developing countries by providing a peer learning opportunity to build programming competencies.

A. RESEARCH QUESTIONS

This project represents quantitative quasi-experimental work to answer the following research questions.

- **RQ1** [Impact of block-based tools for developing programming competencies in children from developing countries] How do the utilization of Scratch and TinkerCAD Arduino impact the progression of programming competencies among students in diverse educational settings, particularly focusing on children in developing countries? This study considers workshop experiences with children from Chile and Ecuador, although it can replicated in similar other contexts.
- **RQ2** [Impact of block-based tools for developing programming competencies in school teachers] What is the impact of Scratch and TinkerCAD Arduino in the development of programming competencies of school teachers in developing countries? Although this study considers workshop experiences with school teachers from different schools in Chile, it can be flawlessly performed again in similar scenarios.
- **RQ3** [Impact of block-based tools for developing programming competencies in university students] How practical is the application of block-based tools for developing programming competencies of computer engineering students? Applying an emulator like TinkerCAD Arduino permits university students to use

algorithmic and programming thinking in real scenarios that can motivate their learning process.

RQ4 [Motivation to continue using block-based tools for developing programming competencies] How eager are the workshop participants to continue learning and using more sophisticated programming skills in the future?

This study looks to answer each defined research question considering that the children's participants did not present previous programming knowledge, the teachers' schools were not from technology teaching areas, and the university students were first-year students. Thus, we addresses a critical gap in the existing literature by focusing on the impact of block-based programming tools like Scratch and TinkerCAD Arduino in non-WEIRD contexts. While previous research has extensively explored these tools in Western, Educated, Industrialized, Rich, and Democratic (WEIRD) countries, there is a lack of studies examining their effectiveness in diverse cultural and educational settings. This research contributes to filling this gap by providing empirical evidence of the benefits of these tools in developing countries, offering new insights into their applicability across different age groups and educational backgrounds.

The rest of this paper is organized as follows. Section II describes the main characteristics of computational thinking competencies and block-based programming and design tools. Section III defines the research questions, material, objectives, impact, and applied tools. Section IV gives details and discusses the defined teaching-learning methodology and the academic results obtained by using Scratch and TinkerCAD Arduino for developing computational thinking competencies. Section V reveals the main threats eventually affecting the results. The paper concludes with a summary of the benefits of our educational experience and motivation to continue applying Scratch and TinkerCAD Arduino to develop programming competencies anywhere.

II. BACKGROUND

A. BLOCK-BASED PROGRAMMING AND DESIGN

Block-based programming simplifies coding by representing programming constructs as blocks with a defined algorithmic purpose that can be manipulated and connected to create program code [6]. These blocks often use a drag-and-drop interface, making it accessible to beginners, especially children and people with without or little programming experience [30]. Block-based programming languages, such as Scratch[1] and Blockly,[2] provide a structured way to learn the basics of coding and computational thinking.

As [31], [32] remark, a primary benefit of block-based programming is its ability to foster computational thinking competencies. Computational thinking involves problem-solving techniques that draw from the principles of computer science [7]. Block-based programming helps individuals develop these competencies by breaking down complex problems into smaller, manageable parts [33]. Users must think logically, sequentially, and algorithmically when arranging these blocks to create functional code.

Design tools complement block-based programming by allowing users to create graphical user interfaces (GUIs) for their programs [34]. These tools, like App Inventor[3] or Thunkable,[4] often employ a visual interface that encourages creativity and problem-solving. They enable users to design interactive and user-friendly applications, crucial for teaching user experience (UX) design principles and problem-solving in a real-world context [35]. TinkerCAD Arduino circuits[5] is a mix of a design and block-based programming tool because it permits coding previous Arduino design circuits by using a block-based interface of style-C textual code programming language [10], [36].

Block-based programming and design tools are invaluable in nurturing computational thinking competencies for their user-friendly, visually intuitive approach to coding and design, making them accessible to learners of all ages and backgrounds [37]. As technology becomes increasingly integral to daily life, we can say that these skills are essential for navigating a rapidly changing world. Figure 1 shows the primary examples of using those programming and design tools, like 'Hello World.' Figure 1a shows a Scratch code example in which the main character will move ten steps and then play the sound 'meow.' Figure 1b shows a Blockly Games code example in which the main character will reach the red balloon or goal. Figure 1c shows a TinkerCAD Arduino design and code example to turn on the red LED for three seconds, turn it off for three seconds, and repeat those actions.

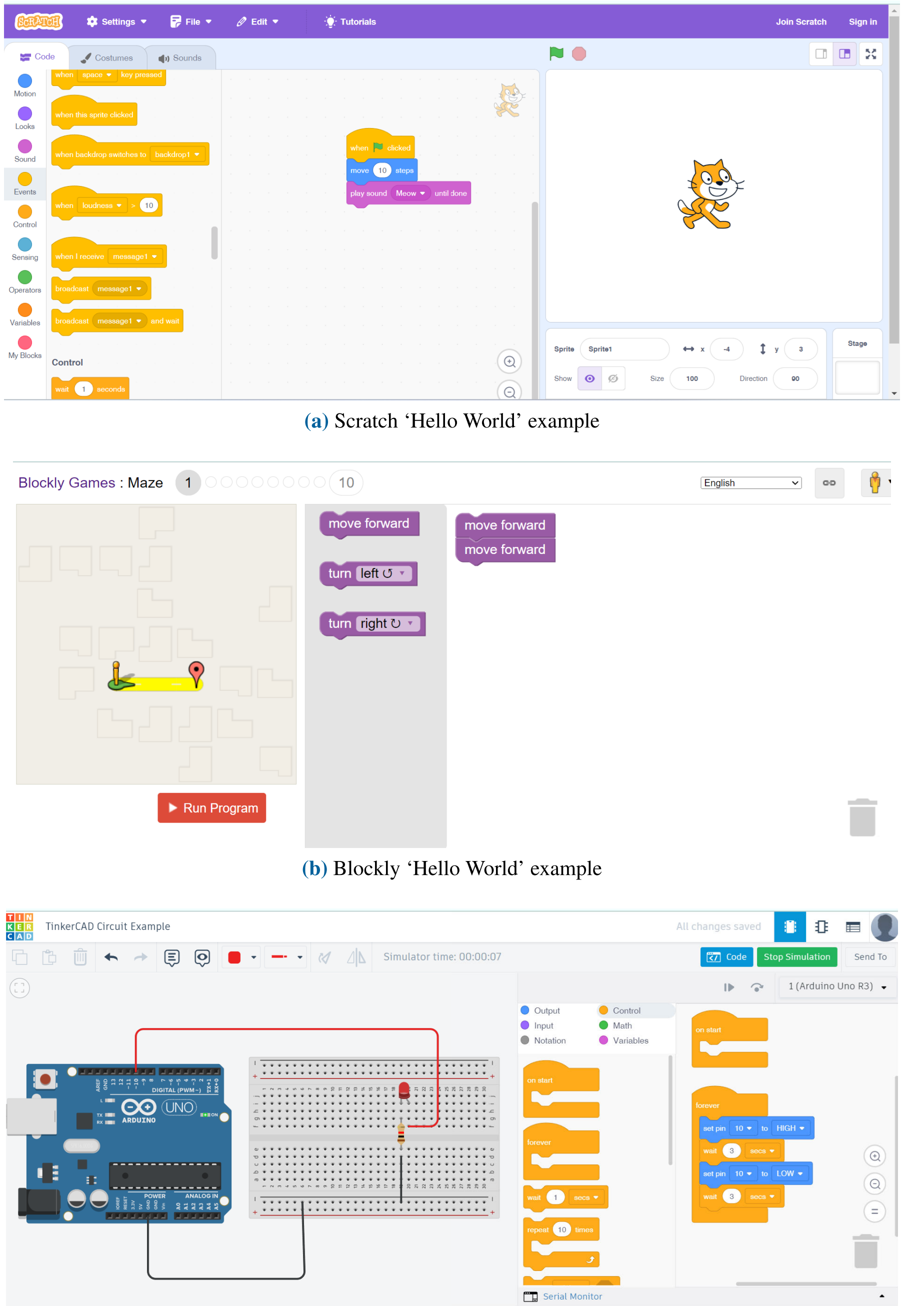
B. COMPUTATIONAL THINKING COMPETENCE

As Lu et al. highlight [20], the computational thinking competency is a foundational skill that transcends programming and computer science. It is an approach to problem-solving that draws from principles rooted in computer science, mathematics, and logic [38]. Computational thinking involves breaking down complex problems into smaller, more manageable parts, identifying patterns, and designing algorithmic solutions [39].

One of the significant benefits of computational thinking competency is its applicability across diverse fields [15], [38]. As Shin et al. [40] describe, in science, computational thinking aids researchers in analyzing complex datasets, simulating experiments, and developing models to understand natural phenomena better. It enhances decision-making processes by enabling data-driven insights invaluable in biology, physics, and social sciences [41].

Shen et al. [42] remark that, in everyday life, computational thinking helps individuals make informed decisions and

[1] https://scratch.mit.edu/
[2] https://blockly.games/
[3] https://appinventor.mit.edu/
[4] https://thunkable.com/
[5] https://www.tinkercad.com/circuits



(a) Scratch 'Hello World' example

(b) Blockly 'Hello World' example

(c) TinkerCAD Arduino 'Hello World' example

FIGURE 1. Basic like 'Hello World' examples of existing block-based programming and design tools.

solve problems efficiently. From optimizing daily routines using algorithms to understanding and evaluating information online, this competency empowers people to navigate the digital age successfully [43]. Additionally, in education, incorporating computational thinking into curricula fosters problem-solving skills, logical reasoning, and creativity among students [7]. For example, Algorithm 1, describes the necessary steps for learning a new topic. Computational thinking equips people with tools and competencies for a rapidly evolving job market that increasingly demands digital items [44].

As Lodi and Martini [45] remark, computational thinking is an interdisciplinary skill with profound implications for the future of human life, science, and education. It equips individuals with the ability to approach problems systematically and devise innovative solutions, making it a crucial competency in the modern world [28]. Further research and ongoing integration of computational thinking in education

Algorithm 1 Algorithm for Learning a New Topic

Require: *Define the topic*
Ensure: *Material of the topic*
  Get a fist picture
  Determine scope
  Find resources
  Create a learning plan
  Define learning success
  while *not success* **do**
    Filter resources
    Review resources
    Play around
    Apply new learning ▷ If it were possible
    Teach new learning ▷ If it were possible
  end while

are essential for maximizing its benefits across various domains.

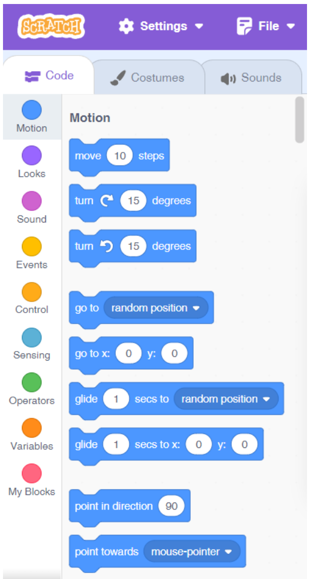
C. SCRATCH & TinkerCAD ARDUINO

1) SCRATCH

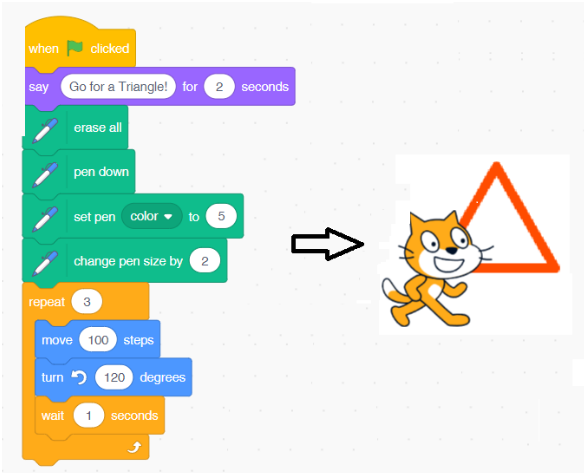
The Scratch platform proposes a programming language based on blocks with visual grammar and combination rules that have the same role as the syntax in text-based programming languages such as C, Java, or Python [36], [46]. As indicated by [47], the original goal of Scratch was to develop a programming approach that would attract people, regardless of age, social background, or educational background, to develop algorithmic solutions. Without the complexities of syntax and semantics of traditional programming languages, Scratch is a language for programming interactive stories, games, animations, and simulations accessible to all its users, who can also share their creations with others. Thus, the primary goal of Scratch is not to prepare people for professional or technical careers in programming but to nurture a new generation of creative and systematic thinkers using programming to express their ideas. Scratch is a current platform that motivates collaborative work [48].

As Vidal et al. [49] noted, Scratch was developed in 2003 by Dr. Michael Resnick and a team of researchers at the Lifelong Kindergarten group within the MIT Media Laboratory. This project received support from various institutions, including the National Science Foundation, Intel Foundation, Microsoft, MacArthur Foundation, LEGO Foundation, Code-to-Learn Foundation, Google, Dell, Fastly, Inversoft, and the MIT Media Lab research consortium [50].

Initially, the objective of Scratch was to create a programming environment that would appeal to individuals of all ages, backgrounds, and educational levels. Scratch aimed to empower programmers to develop algorithmic solutions without the complexities of syntax and semantics found in traditional programming languages [50]. Scratch is designed to operate in multiple languages, such as Spanish, English, French, and Portuguese, enabling users



(a) Categories of actions in Scratch 3



(b) Scratch code for drawing a triangle

FIGURE 2. Scratch actions and code for drawing a triangle.

to create interactive stories, simple games, animations, and simulations. Additionally, Scratch developers can share their creations on the Scratch web platform [51].

The primary goal of Scratch is to cultivate a new generation of innovative and systematic thinkers who can use programming to express their ideas effectively [52]. In Scratch, users can employ the main character, Scratch, utilize existing characters, or create new ones using photos or graphical design effects. Actions for each character can be defined algorithmically using blocks. Scratch categorizes these blocks into colored groups based on their functions, such as motion (dark blue), looks (dark purple), sound (light purple), events (yellow), control (light orange), sensing (light blue), operators (green), variables (dark orange), and user-defined blocks (rose). Figure 2a illustrates these blocks categorized in the current Scratch web platform [51].

The experiments in this article utilize Scratch 3.0 to introduce new blocks and enhance existing ones, ensuring compatibility with earlier programming blocks [51]. Figure 2b illustrates the steps of a simple Scratch program to draw a triangle.

2) TinkerCAD ARDUINO

Tinkercad Arduino is a user-friendly, web-based 3D design, electronics, and coding platform, particularly in computing-electronic solutions using Arduino [53]. Arduino, created in 2005 in Italy, sought to integrate computing and electronics easily, especially for students within open-hardware, cost-effective contexts [54]. Today, many low-cost, Arduino-compatible hardware and software components are available.

As Vidal et al. [55] highlight, Tinkercad encourages and facilitates the development of solutions using Arduino, providing a designated area for hardware components within the solution, a variety of hardware components to add, and a code section for programming. Figure 3 depicts a hardware solution designed in Tinkercad, illustrating turning on a red LED during t1 seconds and turning off that light during t2 seconds. Arduino controls the positive and ground signals. Incorporating programming allows for the cyclic activation and deactivation of the LED for specified durations (t1 and t2). Programming Arduino solutions. As Vidal et al. [55] remark, an exceptional feature of Tinkercad is its ability to generate C/C++ code for Arduino from block-structured code solutions, making it an excellent starting point for learning computing-electronic solutions based on Arduino.

D. RELATED WORK

This article summarizes workshop experiences in different times, contexts, and participants from various countries: children from Milagro, Ecuador [2]; children from Valparaíso, Chile [9]; school teachers from Antofagasta, Chile; and university students from Huancayo, Perú [11], [56].

Regarding the main focus of developing programming competencies in children, the work of Cardenas [57] applied Scratch in a non-WEIRD context. The study by Hsu et al. [58] underscores the significant progress in developing programming competencies within educational settings in recent years. However, educators face the challenge of determining practical pedagogical approaches for teaching these skills. Just the work of Cardenas et al. [2] present updates and effectiveness evaluation of using a recommender system for assisting the development of programming competencies for school children in non-WEIRD scenarios.

Zamir et al. [59] studied school children in Malaysia and Australia who participated in a week-long programming camp where they were introduced to Scratch programming. The study found that these children thoroughly enjoyed learning Scratch, which was fun and easy to grasp. They expressed a keen interest in furthering their understanding of the language and desired to master it eventually.

Regarding the development of computational thinking, Kunz et al. [60] also support the use of block-based programming, finding the enhancement of students' computational thinking, arithmetic skills, and non-verbal visuospatial reasoning following the transition to text-based programming. Importantly, this shift did not lead to a significant decrease in students' motivation for programming. These findings suggest that there could be an effective transition from block-based programming to text-based programming while maintaining students' motivation and nurturing computational thinking skills.

In exploring the advantages and challenges associated with utilizing Scratch for the enhancement of programming competencies, as highlighted by Rodríguez et al. [61], Scratch emerges as a freely accessible online programming language facilitating the acquisition of mathematical concepts and the development of programming skills. Schools can review their primary activities to incorporate and apply computational thinking across core subjects, necessitating adjustments to teaching curricula to encompass programming competencies and block-based environments tailored for children, such as Scratch and TinkerCAD [36]. Various fields incorporate computational thinking into their research endeavors, suggesting that developing programming competencies can be a rewarding and engaging pursuit [10].

Mladenović, Boljat, and Žanko [62] studied programming concepts, mainly focusing on loops, comparing their use in Scratch, Logo, and Python. The study found that most misconceptions about loops are significantly reduced when they use block-based programming languages, such as Scratch, rather than text-based programming languages. The study also found that while students were equally successful in completing tasks in Logo and Python, issues arose predominantly with Python when nested loops were involved. These findings underscore the suitability of visual programming languages like Scratch for teaching programming to K-12 novices, mainly through game development. By eliminating syntax obstacles, students can concentrate on grasping fundamental algorithms. Moreover, compared to text-based languages, visual programming significantly minimizes misconceptions.

III. METHODOLOGY AND RESEARCH DESIGN

The research was conducted with students and teachers from elementary and secondary schools and universities in Chile, Ecuador, and Peru. This study adopted a quasi-experimental design, which is quantitative and descriptive in nature. Participants were divided into groups based on age, educational level, and demographic characteristics. This design allows for comparisons between different demographic groups. The study utilizes an instrument to assess the achievement of programming competencies after the intervention.

This study considers four workshop experiences from 2019 to 2022. The following lines detail each of them.

1) Workshop of 4 sessions of two hours with primary school students from Milagro, Ecuador using

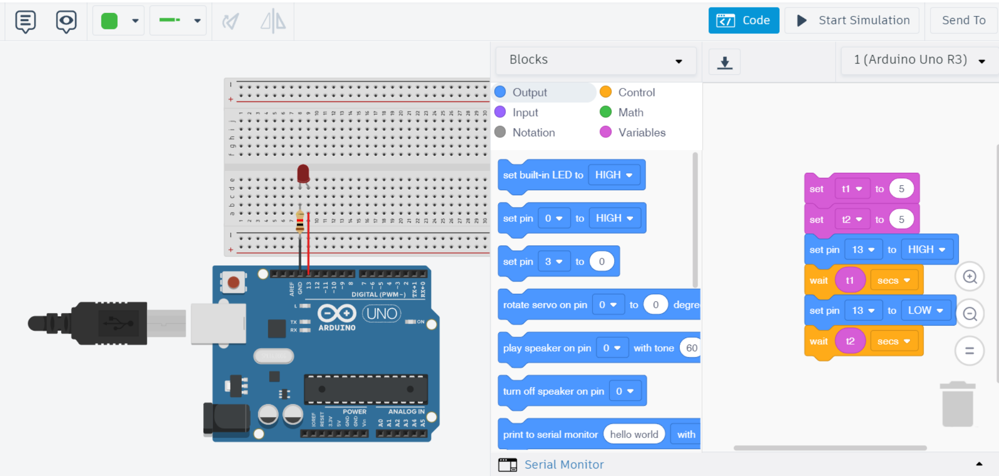


FIGURE 3. TinkerCAD Arduino code example to turn on and off a red LED during t1 and t2 seconds.

Scratch [2]. The workshop took place for one week in May of 2020. We named its participants as Group 1.

2) Workshop of 4 sessions of two hours with primary school students from Valparaíso, Chile using Scratch and TinkerCad Arduino [9], [10]. The workshop took place in September 2019, one session per week. We named its participants as Group 2.
3) Workshop of 4 sessions of two hours with school teachers from Antofagasta, Chile, using Scratch and TinkerCad Arduino. The workshop took place in November 2019, one session per week. We named its participants as Group 3.
4) Academic experience of sessions per week of two hours each, with University students from the Continental University from Huancayo, Perú using TinkerCad Arduino [11]. The activity occurred from March to June 2019, with two weekly sessions. We named its participants as Group 4.

By performing the workshops, children, school teachers, and young people like university students can access software technology and develop programming and computing competencies without significant restrictions, mainly the internet connection that is usually present nowadays. Tables 1, 2, 3 and 4 describe each workshop participants.

TABLE 1. Workshop participants from Milagro, Ecuador.

Focus	Description	Beneficiaries
Gender	Male	236
	Female	192
	Total	428
Age	Under 8 years of age	426
	From 8 to 12 years old	2
	From 12 years to 18	0
	From 18 years to 30	0
	From 30 years to more	0
	Total	428

Each workshop considered practical classes with code examples and the following topics.

TABLE 2. Workshop participants from Valparaíso, Chile.

Focus	Description	Beneficiaries
Gender	Male	6
	Female	3
	Total	9
Age	Under 8 years of age	0
	From 8 to 12 years old	2
	From 12 years to 18	7
	From 12 years to 18	0
	From 18 years to 30	0
	From 30 years to more	0
	Total	9

TABLE 3. Workshop participants from Antofagasta, Chile.

Focus	Description	Beneficiaries
Gender	Male	9
	Female	6
	Total	15
Age	Under 8 years of age	0
	From 8 to 12 years old	0
	From 12 years to 18 years old	0
	From 18 years to 30	0
	From 30 years to more	15
	Total	15

TABLE 4. Workshop participants from Huancayo, Perú.

Focus	Description	Beneficiaries
Gender	Male	95
	Female	33
	Total	128
Age	Under 8 years of age	0
	From 8 to 12 years old	0
	From 12 years to 18	0
	From 18 years to 30	128
	From 30 years to more	0
	Total	128

- Sequential actions. Understanding and exemplifying the nature of sequential actions in daily life and computing programming and design tools.
- Conditional actions. Reviewing and practicing conditional situations in daily life, considering logical situations, and exemplifying their equivalence in computing programming and design tools.

- Repetitive actions. Reviewing and practicing repetitive actions in daily life to introduce control flow actions in computing programming and design tools.
- Grouping related actions. Understanding and exemplifying groups of actions and their context considering conditional or control flow actions in computing programming and design tools.

IV. DATA ANALYSIS AND RESULTS

Figure 4 summarizes the obtained results in previously described workshops in developing programming competencies. First, we can appreciate that applying sequential actions represents better programming competence results. On the other hand, applying grouping-related actions is the programming competence with the lowest results. Applying conditional and iterative actions for the control-flow actions represents similar results.

- Each subfigure of Figure 4 permits answering RQ1 positively; that is, block-based programming and design tools Scratch and TinkerCAD Arduino permit developing programming competencies in children from developing countries like Chile and Ecuador. That motivates us to continue working on defining new experiments and workshops in similar primary school contexts to massify the beauty of programming in children.
- Each subfigure of Figure 4 permits answering RQ1 positively and also permits answering RQ2; that is, Scratch and TinkerCAD Arduino can undoubtedly assist in the programming competencies development of school teachers from an in-developing country like Chile. That motivates us to continue defining new experiments and workshops with primary and secondary school teachers so they can know the applicability of programming for developing competencies and knowledge in their matters.
- Each subfigure of Figure 4 permits answering RQ1 positively and also permits positively answering RQ3; block-based design and programming tools like TinkerCAD Arduino can positively assist in the programming competencies development of university students from an in-developing country like Perú. That motivates us to continue defining new experiments and workshops with first-year students in other universities from in-developing countries, regardless of their professional major, to develop their algorithmic and programming knowledge for applying them in different academic contexts.
- With the previously mentioned results, the paper authors want to define free and open-access online courses for developing programming competencies in their respective universities and countries.

Figure 5 summarizes successful results in each workshop. We can appreciate that Group 3, the group of school teachers, obtained better results. That can be explained by the fact that they are possibly more familiar with technology and because they are more disposable for learning. It is crucial that the teacher participant can use these programming competencies in their subject and teaching to attract and motivate students to learn. The impact of computing technology on children can positively contribute to children's learning.

Groups 1 and 2, school children from Milagro-Ecuador and Valpaíso-Chile, are the second and third groups with better results. Regarding Group 1, children participated in their workshop as an academic activity in the technology course. Group 3, on the other hand, assisted in this workshop as a recognition for being good students in their primary school. Considering university students of Group 4, as [11] also highlights, they were motivated to interact with physical devices (although emulated), and their academic results were acceptable concerning the number of students that did not pass the course.

In general, the workshop participants declared no previous computer programming knowledge, so the sessions were a success. The obtained results positively answer the established research questions. We can also perceive that the most straightforward topic in each programming workshop was the application of sequential actions. In our opinion, those results endorse the sequential nature of human activities [63]. The topics of conditional and grouping actions resulted in more complex issues in each workshop, although the success results were over 65%. Concerning interactive actions, we perceived participants avail that human beings seem familiar with repetitive actions by nature [64].

The analysis of performance by age group is depicted in Figure 6, illustrating that the most favorable area for school students and professors was the Application results of iterative actions, while for university students, the optimal area was the Application results of sequential actions.

Our experience resulted in an innovative pedagogical approach for teaching programming competencies to individuals across diverse age groups, using hands-on learning experiences with Arduino and Scratch. Regarding future research directions, it is essential to continue delving into specific contexts, educational levels, and contents. For example, Arduino-based STEM education can increase students' entrepreneurial skills and positively affect their attitudes toward STEM [65], [66]. In this context, multiple areas and teaching experiences can be conducted in Physics [67], [68], Chemistry [69], [70] or Mathematics [71].

V. DISCUSSION

This article considers the following threats that could distort the obtained results.

- Although this research was conducted in different places with different participants, the results are not completely comparable because participants differ in age and previous knowledge. We cannot compare the obtained university students with school children or

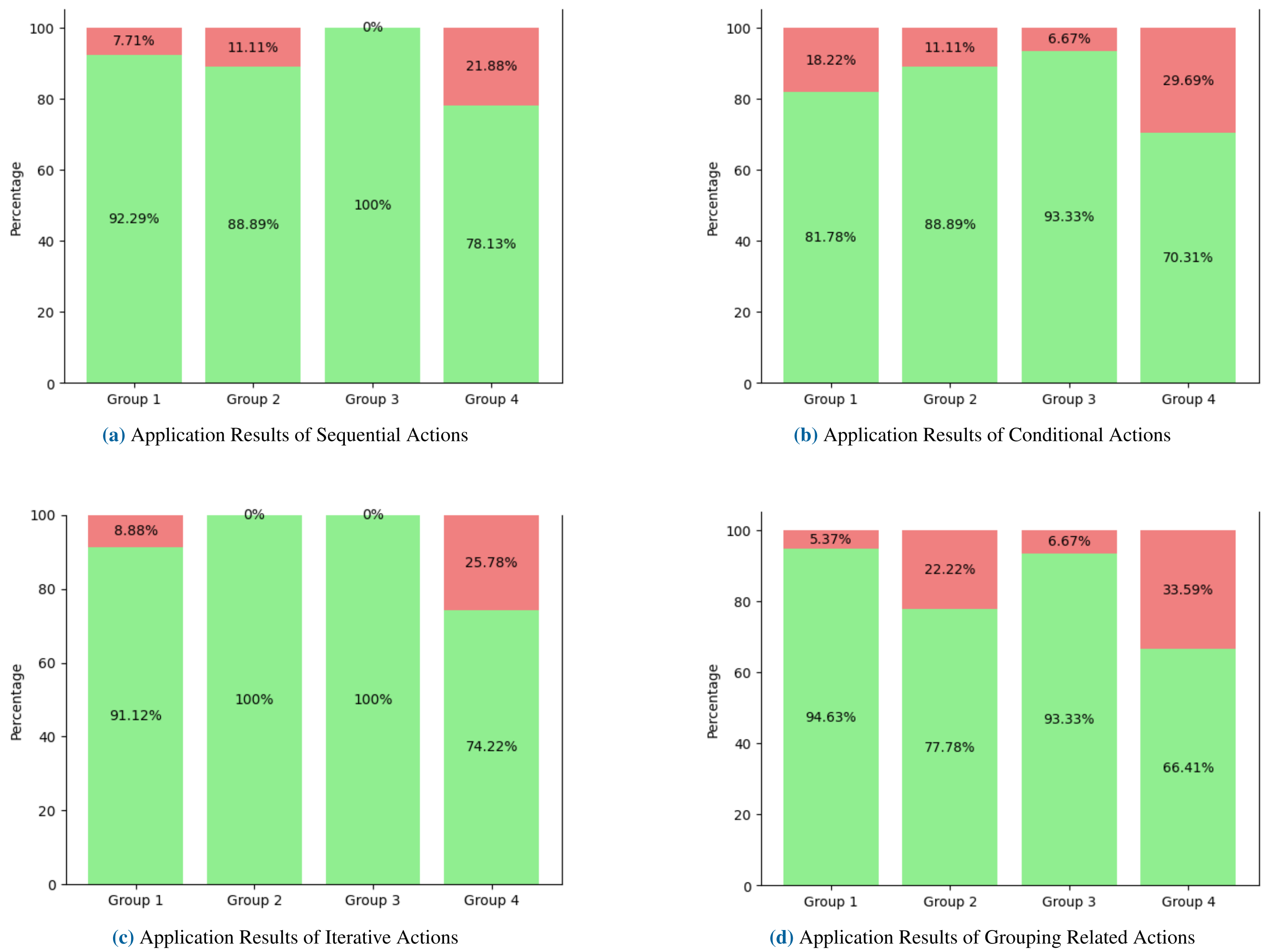


FIGURE 4. **Results for developing programming competencies Workshops - color for Success and color for Failure.**

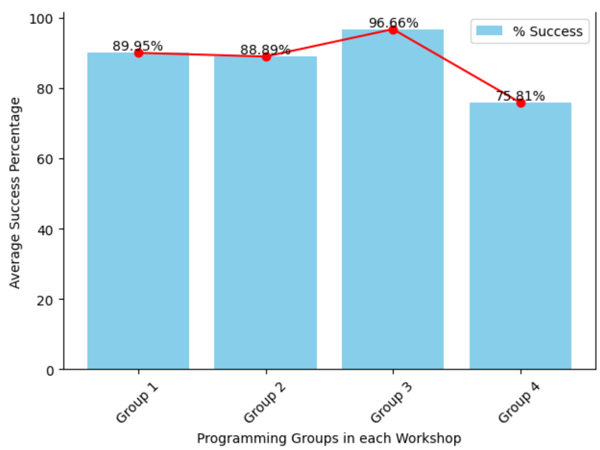


FIGURE 5. **Average of success in workshops for developing programming competencies.**

teachers with school students, although all of them are positive concerning the main goal of developing computational thinking.

- Experiments were performed at different times during the same year. Although experiments were conducted in the same year, other times and weather conditions may affect the participants' performance.
- To avoid the bias implied by the screening process, this work carried out a fully transparent evaluation process using multiple-choice questions with only one correct option.

Despite the promising outcomes demonstrated in this study, several challenges and limitations must be considered. One significant challenge is access to technology, which can vary greatly depending on the socio-economic context. Additionally, the effective use of Scratch and TinkerCAD Arduino relies heavily on the adequate training of teachers, who may require additional support to integrate these tools into their teaching practices. Another potential limitation is the risk of developing an over-reliance on visual programming environments, which might hinder the transition to more advanced, text-based programming languages. Addressing these challenges is crucial for maximizing the benefits of block-based programming tools.

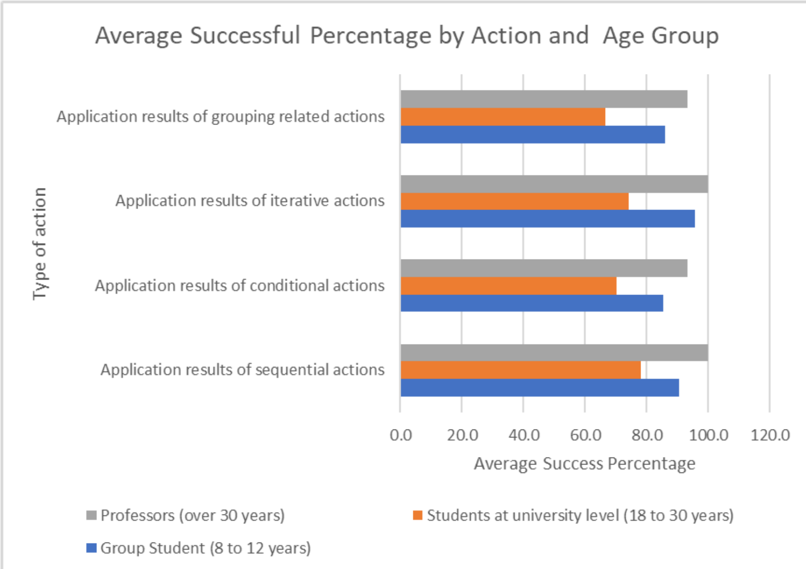


FIGURE 6. Average of success in workshops for developing programming competencies by age group.

VI. CONCLUSION

We can draw the following conclusions based on the results obtained in each workshop for the development of programming competencies.

1) According to our workshops using Scratch and TinkerCAD Arduino to develop programming competencies in different countries and participants' backgrounds, considering the obtained results, we conclude that both tools represent excellent options regardless of the participant's location.
2) Our work successfully answers the four research questions.
   a) Scratch and TinkerCAR Arduino positively affect children's programming competence development process.
   b) Scratch and TinkerCAR Arduino are great tools for developing programming competencies in school teachers from in-developing countries.
   c) TinkerCAR Arduino positively affects university students in developing programming competencies in academic scenarios.
   d) We want to motivate the use and continue using Scratch and TinkerCAD Arduino to develop programming competencies in all contexts.
3) We realized the high impact of tools and technology in attracting children and adults to develop new competencies such as programming and design circuits, two relevant topics for developing problem-solving, critical thinking, creativity, and logical reasoning. That motivates us to continue developing programming competencies in our countries and other countries in South America. Results demonstrate the tools' application effectiveness and the current impact of developing computational thinking in people regardless of their social conditions, age, and academic background.
4) Regarding the school teacher experiences, obtained results and perception of their satisfaction and motivation for learning motivate us to continue working with similar school teacher participants from different social contexts in our countries.

Our work accomplished its primary goal of applying block-based programming and design tools to successfully develop programming competencies in children, school teachers, and university students from developing countries.

Future research should explore the long-term effects of using Scratch and TinkerCAD Arduino in various educational contexts, particularly in developing countries. Additionally, investigating the integration of these tools with other educational technologies could provide a more comprehensive understanding of their potential in enhancing computational thinking. Further studies should also examine strategies to overcome the challenges identified in this study, such as ensuring equitable access to technology and providing adequate teacher training to support the effective use of block-based programming tools.

REFERENCES

[1] U. Sarı, H. M. Pektas, Ö. F. Sen, and H. Çelik, "Algorithmic thinking development through physical computing activities with Arduino in STEM education," *Educ. Inf. Technol.*, vol. 27, no. 5, pp. 6669–6689, Jun. 2022.
[2] J. Cárdenas-Cobo, C. Vidal-Silva, L. Arévalo, and M. Torres, "Applying recommendation system for developing programming competencies in children from a non-weird context," *Educ. Inf. Technol.*, vol. 29, no. 8, pp. 9355–9386, Jun. 2024.
[3] S. Gökçe and A. A. Yenmez, "Ingenuity of scratch programming on reflective thinking towards problem solving and computational thinking," *Educ. Inf. Technol.*, vol. 28, no. 5, pp. 5493–5517, May 2023.
[4] S. Grover and R. Pea, "Computational thinking: A competency whose time has come," *Comput. Sci. Educ., Perspect. Teaching Learn.*, vol. 19, no. 1, pp. 19–38, 2018.

[5] J. M. Wing, "Computational thinking: What and why?" *Mag. Varnegie Mellon Univ. School Comput. Sci.*, vol. 49, no. 3, pp. 1–6, 2010.

[6] Y. Lin and D. Weintrop, "The landscape of block-based programming: Characteristics of block-based environments and how they support the transition to text-based programming," *J. Comput. Lang.*, vol. 67, Dec. 2021, Art. no. 101075.

[7] K. Kwon, A. T. Ottenbreit-Leftwich, T. A. Brush, M. Jeon, and G. Yan, "Integration of problem-based learning in elementary computer science education: Effects on computational thinking and attitudes," *Educ. Technol. Res. Develop.*, vol. 69, no. 5, pp. 2761–2787, Oct. 2021.

[8] A. Chatzopoulos, M. Kalogiannakis, M. Papoutsidakis, S. Psycharis, and D. Papachristos, "Measuring the impact on student's computational thinking skills through STEM and educational robotics project implementation," in *Advances in Early Childhood and K-12 Education*. Hershey, PA, USA: IGI global, 2020, pp. 238–288.

[9] C. Vidal-Silva, J. Serrano-Malebran, and F. Pereira, "Scratch and Arduino for effectively developing programming and computing-electronic competences in primary school children," in *Proc. 38th Int. Conf. Chilean Comput. Sci. Soc. (SCCC)*, Nov. 2019, pp. 1–7.

[10] P. Rojas-Valdés, C. Vidal-Silva, and C. d. L. Fuente, "Successful development of problem-solving and computing programming competences in children using Arduino," in *Proc. Int. Symp. Meas. Control Robot. (ISMCR)*, Sep. 2022, pp. 1–6.

[11] M. Tupac-Yupanqui, C. Vidal-Silva, L. Pavesi-Farriol, A. S. Ortiz, J. Cardenas-Cobo, and F. Pereira, "Exploiting Arduino features to develop programming competencies," *IEEE Access*, vol. 10, pp. 20602–20615, 2022.

[12] X. Tang, Y. Yin, Q. Lin, R. Hadad, and X. Zhai, "Assessing computational thinking: A systematic review of empirical studies," *Comput. Educ.*, vol. 148, Apr. 2020, Art. no. 103798.

[13] M. Chevalier, C. Giang, A. Piatti, and F. Mondada, "Fostering computational thinking through educational robotics: A model for creative computational problem solving," *Int. J. STEM Educ.*, vol. 7, no. 1, pp. 1–18, Dec. 2020.

[14] J. M. Wing, "Computational thinking," *Commun. ACM*, vol. 49, no. 3, pp. 33–35, 2006.

[15] R. P. Y. Lai, "Beyond programming: A computer-based assessment of computational thinking competency," *ACM Trans. Comput. Educ.*, vol. 22, no. 2, pp. 1–27, Jun. 2022.

[16] L.-L. Ung, J. Labadin, and F. S. Mohamad, "Computational thinking for teachers: Development of a localised E-learning system," *Comput. Educ.*, vol. 177, Feb. 2022, Art. no. 104379.

[17] J. Nouri, L. Zhang, L. Mannila, and E. Norén, "Development of computational thinking, digital competence and 21stcentury skills when learning programming in K-9," *Educ. Inquiry*, vol. 11, no. 1, pp. 1–17, Jan. 2020.

[18] M. University, M. Bower, L. Wood, M. University, J. Lai, M. University, C. Howe, R. Lister, R. Mason, K. Highfield, M. University, J. Veal, and M. University, "Improving the computational thinking pedagogical capabilities of school teachers," *Austral. J. Teacher Educ.*, vol. 42, no. 3, pp. 53–72, Mar. 2017.

[19] A. Saad and S. Zainudin, "A review of project-based learning (PBL) and computational thinking (CT) in teaching and learning," *Learn. Motivat.*, vol. 78, May 2022, Art. no. 101802.

[20] C. Lu, R. Macdonald, B. Odell, V. Kokhan, C. D. Epp, and M. Cutumisu, "A scoping review of computational thinking assessments in higher education," *J. Comput. Higher Educ.*, vol. 34, no. 2, pp. 416–461, Aug. 2022.

[21] G. K. W. Wong, "Amplifying children's computational problem-solving skills: A hybrid-based design for programming education," *Educ. Inf. Technol.*, vol. 29, no. 2, pp. 1761–1793, Feb. 2024, doi: 10.1007/S10639-023-11880-9.

[22] A. Pears, M. Tedre, T. Valtonen, and H. Vartiainen, "What makes computational thinking so troublesome?" in *Proc. IEEE Frontiers Educ. Conf. (FIE)*, Oct. 2021, pp. 1–7.

[23] V. Dolgopolovas and V. Dagiene, "Computational thinking: Enhancing STEAM and engineering education, from theory to practice," *Comput. Appl. Eng. Educ.*, vol. 29, no. 1, pp. 5–11, Jan. 2021.

[24] M. Kallia, S. P. van Borkulo, P. Drijvers, E. Barendsen, and J. Tolboom, "Characterising computational thinking in mathematics education: A literature-informed delphi study," *Res. Math. Educ.*, vol. 23, no. 2, pp. 159–187, May 2021.

[25] D. Christensen and D. Lombardi, "Understanding biological evolution through computational thinking: A K-12 learning progression," *Sci. Educ.*, vol. 29, no. 4, pp. 1035–1077, Aug. 2020.

[26] R. Venkatachalam and S. G. Wang, "Computational thinking in economics and finance: Introductory remarks," in *Understanding Complex Systems*. Cham, Switzerland: Springer, 2023, pp. 1–12.

[27] J. Acevedo-Borrega, J. Valverde-Berrocoso, and M. D. C. Garrido-Arroyo, "Computational thinking and educational technology: A scoping review of the literature," *Educ. Sci.*, vol. 12, no. 1, p. 39, Jan. 2022.

[28] A. A. Ogegbo and U. Ramnarain, "A systematic review of computational thinking in science classrooms," *Stud. Sci. Educ.*, vol. 58, no. 2, pp. 203–230, Jul. 2022.

[29] Ñ. Román-González, "Codigoalfabetización y pensamiento computacional en educación primaria y secundaria: validación de un instrumento y evaluación de programas," Tesis, Universidad Nacional de Educación a Distancia UNED, España. Escuela Internacional de Doctorado, Programa de Doctorado en Educación, 2016.

[30] L. Carlos Begosso, L. Ricardo Begosso, and N. Aragao Christ, "An analysis of block-based programming environments for CS1," in *Proc. IEEE Frontiers Educ. Conf. (FIE)*, Oct. 2020, pp. 1–5.

[31] O. Kastner-Hauler, K. Tengler, B. Sabitzer, and Z. Lavicza, "Combined effects of block-based programming and physical computing on primary students' computational thinking skills," *Frontiers Psychol.*, vol. 13, Jun. 2022, Art. no. 875382.

[32] C.-H. Chen and H.-Y. Chung, "Fostering computational thinking and problem-solving in programming: Integrating concept maps into robot block-based programming," *J. Educ. Comput. Res.*, vol. 62, no. 1, pp. 406–427, Mar. 2024.

[33] R. Vardasca, D. Silva, J. Fonseca, M. Tereso, F. Bento, and D. Martinho, "Easy-programming: Towards a web collaborating algorithmic and programming aid for early apprentices," in *Proc. Int. Conf. Adv. Res. Technol., Inf., Innov. Sustainability*, 2023, pp. 243–254.

[34] C. Brady, B. Broll, G. Stein, D. Jean, S. Grover, V. Cateté, T. Barnes, and Á. Lédeczi, "Block-based abstractions and expansive services to make advanced computing concepts accessible to novices," *J. Comput. Lang.*, vol. 73, Dec. 2022, Art. no. 101156.

[35] E. Mailani, D. Tarigan, and A. K. Sitanggang, "Development of a learning tool using the cooperative integrated reading and composition (CIRC) model based on thunkable," *QALAMUNA, Jurnal Pendidikan, Sosial, Dan Agama*, vol. 16, no. 1, pp. 31–48, Feb. 2024.

[36] M. Tupac-Yupanqui, C. L. Vidal-Silva, A. Sánchez-Ortiz, and F. Pereira, "Experiencias y beneficios del uso de Arduino en un primer curso de programacion," *Formación Universitaria*, vol. 14, no. 6, pp. 87–96, Dec. 2021.

[37] S. Rijo-García, E. Segredo, and C. León, "Computational thinking and user interfaces: A systematic review," *IEEE Trans. Educ.*, vol. 65, no. 4, pp. 647–656, Nov. 2022.

[38] Y. Li, A. H. Schoenfeld, A. A. Disessa, A. C. Graesser, L. C. Benson, L. D. English, and R. A. Duschl, "Computational thinking is more about thinking than computing," *Proc. J. STEM Educ. Res.*, vol. 1, pp. 1–18, May 2020.

[39] K. Srinivasa, M. Kurni, and K. Saritha, "Computational thinking," in *Learning, Teaching, and Assessment Methods for Contemporary Learners: Pedagogy for the Digital Generation*. Cham, Switzerland: Springer, 2022, pp. 117–146.

[40] N. Shin, J. Bowers, S. Roderick, C. McIntyre, A. L. Stephens, E. Eidin, J. Krajcik, and D. Damelin, "A framework for supporting systems thinking and computational thinking through constructing models," *Instructional Sci.*, vol. 50, no. 6, pp. 933–960, Dec. 2022.

[41] D. Helbing, S. Mahajan, R. H. Fricker, A. Musso, C. I. Hausladen, C. Carissimo, D. Carpentras, E. Stockinger, J. Argota Sanchez-Vaquerizo, J. C. Yang, M. C. Ballandies, M. Korecki, R. K. Dubey, and E. Pournaras, "Democracy by design: Perspectives for digitally assisted, participatory upgrades of society," *J. Comput. Sci.*, vol. 71, Jul. 2023, Art. no. 102061.

[42] J. Shen, G. Chen, L. Barth-Cohen, S. Jiang, and M. Eltoukhy, "Connecting computational thinking in everyday reasoning and programming for elementary school students," *J. Res. Technol. Educ.*, vol. 54, no. 2, pp. 205–225, Mar. 2022.

[43] K. Kanaki and M. Kalogiannakis, "Assessing algorithmic thinking skills in relation to age in early childhood STEM education," *Educ. Sci.*, vol. 12, no. 6, p. 380, May 2022.

[44] A. Yadav and U. Berthelsen, *Computational Thinking in Education: A Pedagogical Perspective*. Evanston, IL, USA: Routledge, 2021.

[45] M. Lodi and S. Martini, ''Computational thinking, between papert and wing,'' *Sci. Educ.*, vol. 30, no. 4, pp. 883–908, Aug. 2021.

[46] C. Cabo, ''Student progress in learning computer programming: Insights from association analysis,'' in *Proc. IEEE Frontiers Educ. Conf. (FIE)*, Oct. 2019, pp. 1–8.

[47] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, and B. Silverman, ''Scratch: Programming for all,'' *Commun. ACM*, vol. 52, no. 11, pp. 60–67, 2009.

[48] C. V. Silva and L. P. Farriol, ''Desarrollo de un sistema de adquisición y tratamiento de señales electrocardiográficas,'' *Revista Facultad de Ingeniea-Universidad de Tarapacá, vol. 13, no. 1, pp. 39–46, 2005.*, vol. 13, no. 1, pp. 39–46, 2005.

[49] C. L. Vidal, C. Cabezas, J. H. Parra, and L. P. López, ''Experiencias prácticas con el uso del lenguaje de programación scratch para desarrollar el pensamiento algorítmico de estudiantes en Chile,'' *Formación Universitaria*, vol. 8, no. 4, pp. 23–32, 2015.

[50] A. Sweigart, *Scratch Programming Playground: Learning to Program By Making Cool Games*. San Francisco, CA, USA: Starch Press, 2016.

[51] *Scratch—Imagine Program Share*. Accessed: Feb. 24, 2024. [Online]. Available: https://scratch.mit.edu/

[52] L. Zhang and J. Nouri, ''A systematic review of learning computational thinking through scratch in K-9,'' *Comput. Educ.*, vol. 141, Nov. 2019, Art. no. 103607.

[53] *Tinkercad—Autodesk Tinkercad—Circuits*. Accessed: Mar. 24, 2024. [Online]. Available: https://www.tinkercad.com/dashboard?type=circuits&collection=designs

[54] M. Lineros, B. Bastías, F. Muñoz, K. Aravena, M. Figueroa, L. Rodríguez, B. Villegas, F. Hinojosa, B. Gutiérrez, A. Guerra, O. Bernales, C. Román, S. Collantes, C. Vidal, and R. Villarroel, ''Electronics for everybody: Student practical experiences using Arduino,'' in *Proc. 37th Int. Conf. Chilean Comput. Sci. Soc. (SCCC)*, Nov. 2018, pp. 1–8.

[55] C. Vidal-Silva, M. I. Lineros, G. E. Uribe, and C. J. Olmos, ''Electrónica para todos con el uso de Arduino: Experiencias positivas en la implementación de soluciones hardware-software,'' *Información tecnológica*, vol. 30, no. 6, pp. 377–386, Dec. 2019.

[56] M. Tupac-Yupanqui, C. L. Vidal-Silva, A. Sánchez-Ortiz, and F. Pereira, ''Experiencias y beneficios del uso de Arduino en un curso de programación de primer año,'' *Formación Universitaria*, vol. 14, no. 6, pp. 87–96, Dec. 2021.

[57] J. Cardenas-Cobo, ''Enhancing the learning of programming using scratch: A recommender-systems-based approach in non WEIRD communities,'' Ph.D. dissertation, Softw. Eng., Univ. Seville, Seville, Spain, 2020.

[58] T.-C. Hsu, S.-C. Chang, and Y.-T. Hung, ''How to learn and how to teach computational thinking: Suggestions based on a review of the literature,'' *Comput. Educ.*, vol. 126, pp. 296–310, Nov. 2018.

[59] A. R. Zamir, A. Sax, W. Shen, L. Guibas, J. Malik, and S. Savarese, ''Taskonomy: Disentangling task transfer learning,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3712–3722.

[60] K. Kunz, K. Moeller, M. Ninaus, U. Trautwein, and K. Tsarava, ''Making the transition to text-based programming: The pilot evaluation of a computational thinking intervention for primary school students,'' in *Proc. 18th WiPSCE Conf. Primary Secondary Comput. Educ. Res.*, Sep. 2023, p. 20, doi: 10.1145/3605468.3609770.

[61] J. A. Rodríguez-Martínez, J. A. González-Calero, and J. M. Sáez-López, ''Computational thinking and mathematics using scratch: An experiment with sixth-grade students,'' *Interact. Learn. Environments*, vol. 28, no. 3, pp. 316–327, Apr. 2020, doi: 10.1080/10494820.2019.1612448.

[62] M. Mladenović, I. Boljat, and Ž. Žanko, ''Comparing loops misconceptions in block-based and text-based programming languages at the K-12 level,'' *Educ. Inf. Technol.*, vol. 23, no. 4, pp. 1483–1500, Jul. 2018, doi: 10.1007/S10639-017-9673-3.

[63] K. Li and Y. Fu, ''Prediction of human activity by discovering temporal sequence patterns,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 8, pp. 1644–1657, Aug. 2014.

[64] K. Bacharidis and A. Argyros, ''Exploiting the nature of repetitive actions for their effective and efficient recognition,'' *Frontiers Comput. Sci.*, vol. 4, Mar. 2022, Art. no. 806027.

[65] U. Sari, H. Çelik, H. M. Pektas, and S. Yalçın, ''Effects of STEM-focused Arduino practical activities on problem-solving and entrepreneurship skills,'' *Australas. J. Educ. Technol.*, vol. 38, pp. 135–149, May 2022.

[66] T. Topcubasi and A. Tiryaki, ''The effect of Arduino-based e-stem education on students' entrepreneurial skills and stem attitudes,'' *J. Sci. Learn.*, vol. 6, no. 4, pp. 424–434, 2023.

[67] C. A. Petry, F. S. Pacheco, D. Lohmann, G. A. Correa, and P. Moura, ''Project teaching beyond physics: Integrating Arduino to the laboratory,'' *Technol. Appl. Electron. Teaching (TAEE)*, vol. 1, pp. 1–6, Jun. 2016.

[68] U. Sari and T. Kirindi, ''Using Arduino in physics teaching: Arduino-based physics experiment to study temperature dependence of electrical resistance,'' *J. Comput. Educ. Res.*, vol. 7, no. 14, pp. 698–710, Oct. 2019.

[69] A. V. Cordova-Huaman, V. R. Jauja-Ccana, and A. La Rosa-Toro, ''Low-cost smartphone-controlled potentiostat based on Arduino for teaching electrochemistry fundamentals and applications,'' *Heliyon*, vol. 7, no. 2, Feb. 2021, Art. no. e06259.

[70] N. Papadimitropoulos, K. Dalacosta, and E. A. Pavlatou, ''Teaching chemistry with Arduino experiments in a mixed virtual-physical learning environment,'' *J. Sci. Educ. Technol.*, vol. 30, no. 4, pp. 550–566, Aug. 2021.

[71] D. Herceg and D. Herceg, ''Arduino and numerical mathematics,'' in *Proc. 9th Balkan Conf. Informat.*, Sep. 2019, pp. 1–26, doi: 10.1145/3351556.3351590.

CRISTIAN VIDAL-SILVA received the B.S. degree in information engineering from the Faculty of Engineering, Catholic University of Maule, Talca, Chile, in 2003, the M.S. degree in computer science from the University of Concepcion, Concepción, Chile, in 2007, the M.S. degree in computer science from Michigan State University, East Lansing, MI, USA, as a Fulbright Scholar, and the Ph.D. degree in software engineering from the University of Seville, Seville, Spain.

From 2004 to 2007, he was a Research Assistant with the Engineering Faculty, University of Concepcion, and a part-time Professor with the Catholic University of Maule. From 2008 to 2013, he was a Professor with the Economy and Business Faculty, University of Talca, Talca. From 2014 to 2018, he was a Professor and a Researcher in the software engineering and computer science areas with the University of Playa Ancha, University of Viña del Mar, and Autónoma University of Chile, Chile. From 2019 to 2021, he was a Professor with the Administration Department, Catholic University of the North, Antofagasta, Chile. He is currently a full-time Professor with the Videogame Development and Virtual Reality Engineering (VDVRE) School, Engineering Faculty, University of Talca. He is also a Professor and a Researcher with the VDVRE School. He is the main author of more than 30 articles in research areas, such as feature-oriented and aspect-oriented software engineering, machine learning, digital circuits, and programming. He was a reviewer of WOS and Scopus journals.

JESENNIA CÁRDENAS-COBO received the B.S. degree in information systems from the Escuela Politecnica del Litoral (ESPOL), Ecuador, the Diploma degree in higher education by competences from the Technical University of Ambato, Ecuador, and the M.Sc. degree in business administration from the Business Technological University of Guayaquil, Ecuador. She is currently pursuing the Ph.D. degree in software engineering with the University of Seville, Spain. She is also a full-time Professor and the Dean of the Faculty of Engineering Sciences, State University of Milagro (UNEMI, today). She has more than 20 years of professional experience in higher education. Her main research interests include software products and artificial intelligence applied to engineering education.

MIGUEL TUPAC-YUPANQUI received the master's degree in systems engineering with a major in business systems management from the National University of the Center of Peru. He is currently a full-time Teacher with Continental University, Peru. He is also an Electronics Engineer with the Ricardo Palma University of Lima, Peru. Since 2005, he has been the Academic Director in charge of the Professional Academic School (EAP) of Systems Engineering and Informatics, Continental University, contributing with academic and administrative management. He has contributed to teaching and research.

JORGE SERRANO-MALEBRÁN received the B.S. and M.S. degrees in information systems from the Universidad Católica del Norte, Chile, in 2013, and the Ph.D. degree in strategic management and international business from the University of Seville, Spain, in 2022. Since 2013, he has been an Assistant Professor with the Business Management Department, Universidad Católica del Norte. He has authored research articles published in JCR journals. His research interests include information systems, social media, and consumer behavior.

AURORA SÁNCHEZ ORTIZ received the M.Sc. degree in industrial engineering from the Universidad de Chile, Chile, and the Ph.D. degree in information science, with a minor in management information systems from The University of North Texas, USA. She is currently an Associate Professor in information systems and the Director of the Information Technology Management Research Center, Business School, Universidad Católica del Norte, Chile. Her research interests include digital transformation, e-government, and smart cities. She has been published in journals, such as *Heliyon*, IEEE ACCESS, *Journal of Theoretical and Applied Electronic Commerce Research*, *Sustainability*, and *Journal of Technology Management & Innovation*. She received the Fulbright Scholarship Award for the Ph.D. degree.