education sciences

MDPI

Article

Fostering Computational Thinking Skills: A Didactic Proposal for Elementary School Grades

Ricardo Silva [1,2,3,*], Benjamim Fonseca [1], Cecília Costa [1,2] and Fernando Martins [3,4]

[1] Escola de Ciências e Tecnologia, Universidade de Trás-os-Montes e Alto Douro, 5000-801 Vila Real, Portugal; benjaf@utad.pt (B.F.); mcosta@utad.pt (C.C.)
[2] CIDTFF—Centro de Investigação em Didática e Tecnologia na Formação de Formadores, 3810-193 Aveiro, Portugal
[3] Instituto Politécnico de Coimbra, ESEC, NIEFI—PEAPEA, 3030-329 Coimbra, Portugal; fmlmartins@esec.pt
[4] Instituto de Telecomunicações, Delegação da Covilhã, 6201-001 Covilhã, Portugal
* Correspondence: ricardojpratas@gmail.com

Abstract: There is a growing presence of technology in the daily lives of elementary school students, with a recent exponential rise due to the constraints of remote teaching during the COVID-19 pandemic. It is important to understand how the education system can contribute to helping students develop the required skills for technological careers, without neglecting its obligation to create conditions that allow them to acquire transversal skills and to enable them to exercise full citizenship. The integration of Educational Robotics and block programming activities in collaborative learning environments promotes the development of computational thinking and other ICT skills, as well as critical thinking, social skills, and problem solving. This paper presents a theoretical proposal of a didactic sequence for the introduction to educational robotics and programming with Scratch Jr. It is composed of three learning scenarios, designed for elementary school teaching. Its main goal is to create conditions that favour the development of computational thinking in a collaborative learning environment. With increasing complexity and degree of difficulty, all the tasks root from a common problem: How can we create an algorithm that programs the robot/sprite to reach a predetermined position?

Keywords: computational thinking; educational robotics; Scratch Jr; elementary school; collaborative learning

Citation: Silva, R.; Fonseca, B.; Costa, C.; Martins, F. Fostering Computational Thinking Skills: A Didactic Proposal for Elementary School Grades. *Educ. Sci.* **2021**, *11*, 518. https://doi.org/10.3390/educsci11090518

Academic Editors: Sapounidis Theodosios, Michail Kalogiannakis, Nikolaos Fachantidis and Dimitrios Stamovlasis

Received: 4 August 2021
Accepted: 2 September 2021
Published: 8 September 2021

1. Introduction

One of the challenges facing the elementary education system is determining the knowledge and skills that students must acquire since they will finish mandatory education in a future that is difficult to predict [1]. The reality in which we live is constantly changing, influenced by technological advances, access to vast repositories of information, climate change, migratory movements, and social and political tensions. Teaching and learning processes are not alien to these changes and are becoming significantly more complex [1], as seen in the changes implemented by the Portuguese Ministry of Education, in response to the constraints imposed by the COVID-19 pandemic [2]. Furthermore, the growing influence of digital technologies in education systems [1] and the way Digital Natives [3,4] perceive them reinforce the need to understand how to integrate technology in teaching processes [5].

Collaboration, problem-solving, critical thinking, communication, innovation, and creativity are essential skills for the 21st century [6], as is computational thinking [7]. The importance of preparing students for future technological professions is well known [8]. It is important that both teachers and students develop their digital competences [9–11]. However, it is also essential to create conditions for them to acquire transversal skills that will allow them to exercise full citizenship [12]. In response to this issue, more and

more countries are formally integrating programming into the school curriculum, as it contributes to the acquisition and deepening of mathematical skills, problem solving, student engagement in learning, encouragement of peer collaboration [13], and development of computational thinking [7].

At the beginning of the last decade, Benitti [14] stated that most educational robotics (ER) uses focused on the technological aspects (robotics, mechatronics, and programming) or sought curricular links close to these areas, a trend that persists [15,16]. The growing technological development and number of ER platforms available is indisputable. However, there are still obstacles to the implementation of ER in teaching practices: specificity of the technical knowledge involved, lack of pedagogical and didactic material to facilitate its articulation with the curriculum, absence of curriculum guidelines, lack of specific training for the development of teachers' didactic knowledge, and the high cost of most platforms used [17–19].

The integration of ER into teaching and learning processes promotes the development of ICT skills and computational thinking [7,20,21], allowing students to participate in systematic tasks of designing code sequences necessary to program a robot and thus to find a solution to a problem [22]. Angeli et al. [23] propose a computational thinking curriculum framework. This approach draws on patterns identified while solving previous tasks to find solutions to new problems. As a result, students develop abstraction and generalisation skills, develop decomposition skills by breaking down complex problems into smaller manageable pieces, draw on algorithmic thinking when structuring a sequence of instructions, and use debugging processes when attempting to correct errors identified in their proposed solutions.

In their systematic review, Conde et al. [24] suggest that the STEAM approach in activities with ER may foster the development of several skills, such as computational thinking, problem-solving, critical thinking, interdisciplinary skills, autonomous learning, negotiating skills, and social skills. Kim et al. [25] suggest that participation in activities that integrate ER positively influence the perception of gender stereotyping associated not only with these types of activities but also with careers within the various STEM fields. Introducing children to programing at early ages is also acknowledged as a strategy that promotes the development of computational thinking [26,27].

As a pedagogical proposal, a didactic sequence for introducing ER and programming with Scratch Jr is presented here. Consisting of three learning scenarios [28,29] and assuming as a main objective the creation of conditions that may foster the development of computational thinking in a collaborative learning environment, the starting question is: How can we create an algorithm that programs the robot/sprite to reach a certain position?

2. Computational Thinking

Papert [12] speculates on the impact of computers and computer culture on the future of education and schools, presenting ideas for the development of computational thinking. Although he does not state the expression, he refers to the application of computational thinking in problem solving, highlighting the use of programming blocks and the recursive principle of identifying bugs and then debugging them, allowing the children to develop skills that can be transferred to other contexts.

Wing [30], p. 33 describes computational thinking as "a fundamental skill for everyone, not just for computer scientists", it "involves solving problems, designing systems, and understanding human behaviour, by drawing on the concepts fundamental to computer science" [30], p. 33. There are different definitions and proposed taxonomies for computational thinking [20,31–33]. Given that the design of the learning scenarios presented here includes ER and block programming, it was deemed appropriate to adopt the computational thinking skills model proposed by Atmatzidou and Demetriadis [20], as seen in the table below (Table 1).

Table 1. Computational thinking skills model [20], p. 664.

CT Skills	Description	Student Skills (The Student Should Be Able to. . .)
Abstraction	Abstraction is the process of creating something simple from something complicated, by leaving out the irrelevant details, finding the relevant patterns, and separating ideas from tangible details.	1. Separate the important from the redundant information. 2. Analyse and specify common behaviours or programming structures between different scripts. 3. Identify abstractions between different programming environments.
Generalisation	Generalisation is transferring a problem-solving process to a wide variety of problems.	Expand an existing solution to a given problem to cover more possibilities/cases.
Algorithm	Algorithm is a practice of writing step-by-step specific and explicit instructions for carrying out a process.	1. Explicitly state the algorithm steps. 2. Identify different effective algorithms for a given problem. 3. Find the most efficient algorithm.
Modularity	Modularity is the development of autonomous processes that encapsulate a set of often used commands performing a specific function that might be used in the same or different problems	Develop autonomous code sections for use in the same or different problems.
Decomposition	Decomposition is the process of breaking down problems into smaller parts that may be more easily solved.	Break down a problem into smaller/simpler parts that are easier to manage.

3. Didactic Sequence

Participation in ER and programming activities fosters the development of computational thinking skills [7,20,31,32,34]. The didactic sequence presented here is composed of three learning scenarios designed for elementary school classrooms: (1) Meet DOC!; (2) Free throw line; and (3) Tell us a story, DOC!.

Seeking to meet the recommendations of the Evaluation Study on the initiation to programming in elementary schools [35] and the Guidelines for Robotics [36] in Portugal, this set of learning scenarios foresees the articulation and collaboration of two teachers in the classroom, the class teacher and a teacher with training in ER and programming. Regarding students, it is suggested that the principles of collaborative learning [37–39] are adopted, forming groups of different sizes in each task, with the number of elements being determined by the potential and constraints of the technological artefact used in the different learning scenarios.

We suggest that the teachers allow themselves to be surprised by students' knowledge and skills and be open to learning from students [40]. By doing so, they can encourage the students to assume an identical posture and thus facilitate collaborative learning. Regarding the students, we suggest that they be allowed to play with the artefacts (robots, story maps, and Scratch Jr template) and collaborate openly with their peers.

This sequence can be implemented in the different years of elementary school or mixed classes, as long as the formation of groups considers each class's specificities. All tasks are presented as a challenge and after the students look for solutions within their group, they will be discussed and optimised with the entire class.

3.1. Meet DOC

For this learning scenario, the DOC robot by Clementoni was chosen. It is a very easy to use ER platform, with reduced programming options and functionalities. These restrictions, associated with its toy-like appearance (Figure 1), make it a suitable option for a learning scenario of an introduction to ER that covers the four years of elementary school.

Figure 1. Robot Doc, retrieved from www.clementoni.com/pt/67285-doc-robo-educativo-falante/ (accessed on 17 August 2021).

Each group of three or four students, depending on the size of the class, will have a workstation where a robot, a board, and playing cards will be available (Figure 2).

Figure 2. Cards and game board.

3.1.1. Challenge 1

Initially, the groups are asked what they think they are going to do. They can explore the resources before answering. Based on the students' answers, the concept of "programming the robot" is discussed with the teachers orchestrating the discussion, so that the students understand the need to use a specific method to communicate with the robot to give instructions that it can execute. In this case, this will be completed by creating a programming sequence, pressing the buttons on the robot head. The next step is to present the challenge: programming the robot according to the sequence given by the deck of cards. This ordering aims to allow students to discover how the robot works to execute basic commands (one step forward; one step back; turn left and turn right) and sequences of commands for the robot to reach a certain position on the board. Throughout this stage, teachers should circulate among the groups, trying to help unblock possible obstacles with questions or hints, avoiding demonstrating the necessary procedures whenever possible.

3.1.2. Challenge 2

After all the groups have managed to execute the commands indicated on the cards, a new challenge is launched: program the robot with a sequence of commands that allows it to reach the butterfly position. This time there is no letter with directions, the groups can decide which path the robot should follow. It is also explained that once everyone has found a solution, they should be able to explain it to the rest of the class. It is expected that different solutions to this problem will emerge or even groups that have not found a solution.

3.1.3. Sharing and Discussion

The phase of presentation and discussion of solutions serves a double purpose: it allows students to understand the importance of a clear and structured communication of the programming carried out, as well as allowing them to experience processes of debugging and optimisation of algorithms.

The learning scenario ends with a brief discussion of the concept of algorithm, starting with what they already know, algorithms of elementary arithmetic operations, so that

students can draw conclusions about the importance of systematically describing specific and explicit instructions to perform a procedure. Next, oral suggestions are requested to construct an algorithm that comprises the necessary steps to leave the classroom. This algorithm will be optimised at the end of each learning scenario. It is expected that students will be able to summarize detailed instructions into simpler blocks that everyone understands.

3.2. Free Throw Line

With an introductory and exploratory approach to programming for this learning scenario, we chose the programming language Scratch Jr, a simplified version of Scratch, employing collaborative learning [37–39], supported by tablets [41] with an Android operating system. This programming language was chosen for being similar to how the DOC robot is programmed and the absence of a coordinate system for movement, a mathematical concept that is not part of the curriculum of the early years of elementary school in Portugal. If there are enough tablets, students should work in dyads.

To create low complexity algorithms for solving specific challenges and problems, this learning scenario starts with a PowerPoint presentation of an algorithm designed for the Game of Catch (Figure 3), giving continuity to the previous learning scenario.

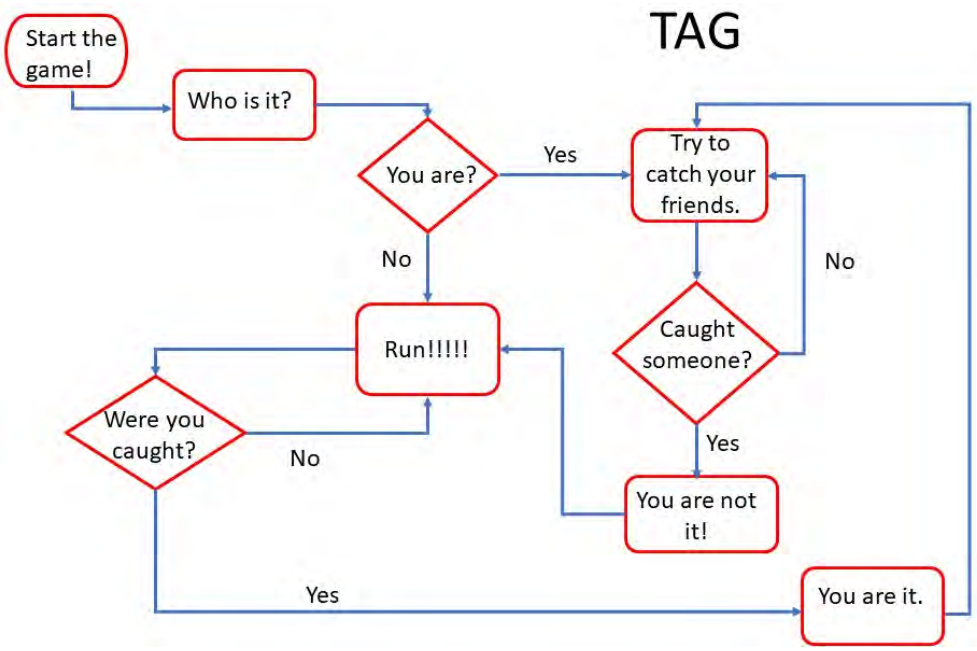

Figure 3. Algorithm for the game of Tag.

3.2.1. Challenge 1

The first challenge is presented to the students: explore the software interface and figure out how to move the sprite one step to the right (Figure 4), without touching it. The students have at their disposal a Portuguese translation of the guide available at https://www.scratchjr.org/pdfs/scratchjr-interface-guide.pdf (accessed on 17 August 2021).

After all groups have discovered which block of code to put in the programming track, they are asked to describe the procedure, so that the teacher can reproduce it on the interactive whiteboard. This sequence is repeated for the discovery of how to move the sprite one step up.

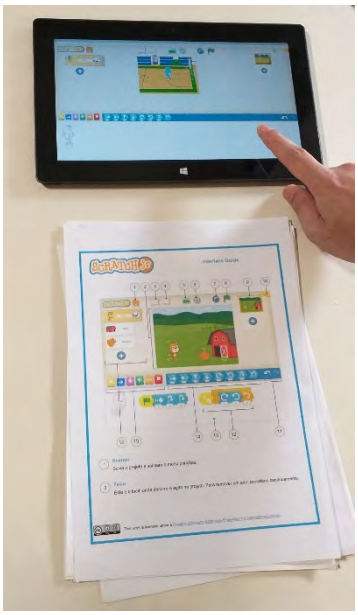

Figure 4. Exploring the interface.

3.2.2. Challenge 2

For the next challenge, students must create an algorithm that allows the sprite to reach a specific position in the scenario (free throw line). If they do not succeed, the teachers try to help through questioning, and may also use the interactive whiteboard to exemplify a possible discovery process (for example, try to run a code sequence of two blocks and see what happens to the sprite). As in the previous tasks, the groups orally share the discovered algorithms reproduced on the interactive whiteboard by the teacher (Figure 5). This suggestion is related to the class time management. Ideally, the students should be allowed the opportunity to share their solutions using the interactive whiteboard.

Figure 5. Algorithm on the interactive whiteboard.

3.2.3. Sharing and Discussion

None of the algorithms created by the students will achieve the expected result when implemented on the whiteboard. The sprite's position in the whiteboard template is intentionally different from the one on the students' tablets, serving as a trigger for discussion around the importance of adapting algorithms to specific contexts and problems. Since the difference is quite noticeable, it is expected that the students will understand why the algorithms do not produce the intended results and suggest the necessary corrections. At this stage students are encouraged to use the interactive whiteboard to demonstrate and share their ideas with the rest of the class.

It is expected that most or all groups will create algorithms consisting of a block of code for each movement of the sprite (Figure 6). Starting from a solution found by one group, building an algorithm that implements fifty steps in the programming track is discussed. The discussion is orchestrated by the teachers in order to allow the students to draw conclusions about the necessity to optimise the code, reducing the number of blocks used for its creation to the minimum possible. Since only the blocks "Move up" and "Move right" are worked on, the groups are asked to create an algorithm that uses only one block of each of these movements. If students cannot figure out a solution, teachers should encourage them to look closely at each block and describe what they see. It is expected that by observing the presence of a number field in each block, they will try to see what happens if they change the value of that field. The solutions found (Figure 7) are again shared with the class via the interactive whiteboard.

Figure 6. Algorithm with block repetition.

Once the task is concluded, the algorithm for leaving the classroom is redesigned, seeking improvements in its efficiency and simplicity of implementation.

3.3. Tell Us a Story, DOC!

As with the ER learning scenario present in Section 3.1, this one will also be supported by the robot DOC. Each group of three or four students, depending on the size of the class, will have a workstation where a robot will be available.

3.3.1. Challenge

This time the challenge is to create a story map and use the robot DOC to tell it, i.e., the robot must scroll through the map according to the algorithm created. The groups can create maps for original stories or for adaptations of stories they know.

Before some examples of story maps are projected/shown (Figure 8), some characteristics of the robot are pointed out: each movement forward or backward corresponds to a displacement of 15 cm and rotations to the left and to the right correspond to a quarter turn.

Figure 7. Optimised algorithm.

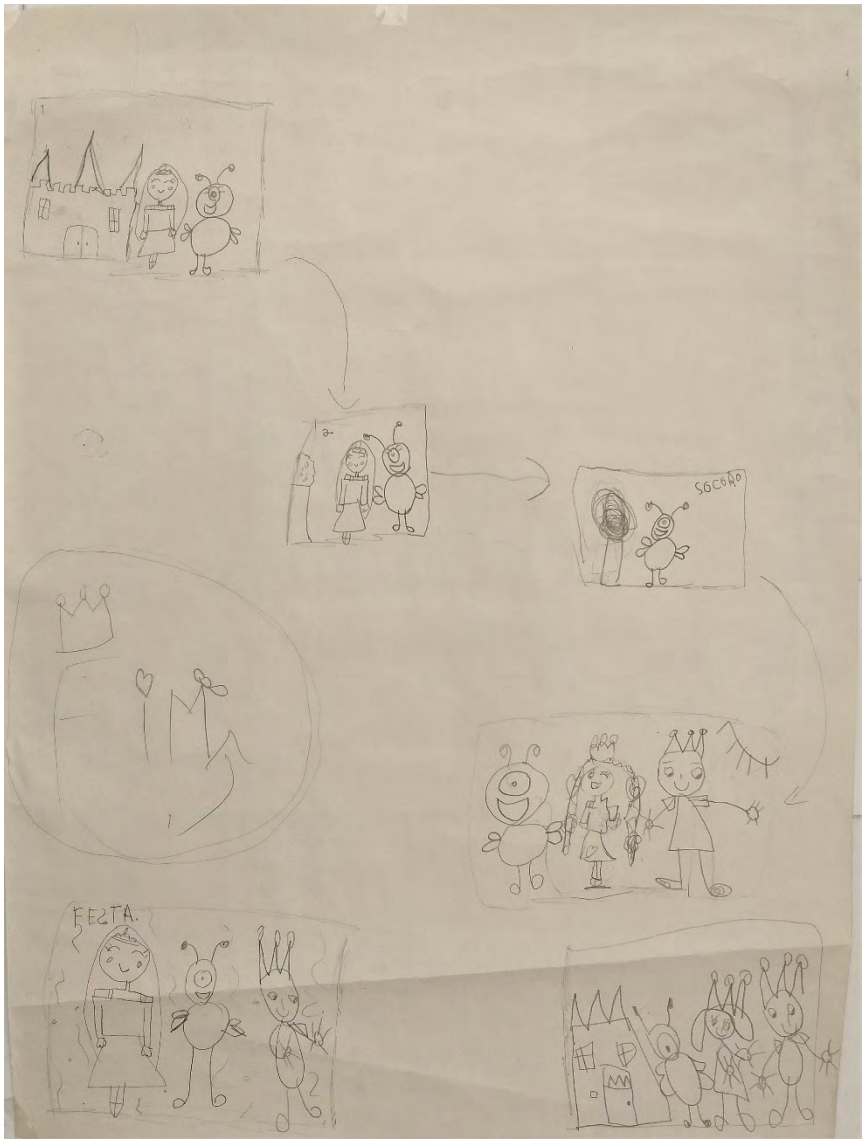

Figure 8. Example of a story map created by students.

The students are encouraged to decorate their robot, for which there should be a stand with various materials available (paper, tape, streamers, etc.). For the creation of the map, there should be different paper formats (A4, A3, A2, or a roll of drawing paper), and the students can also use the floor to create their map. During this phase, teachers should accompany the groups, trying to help them overcome any obstacles that may arise.

3.3.2. Sharing and Discussion

The algorithm created by the students should allow the robot to autonomously scroll through the map according to the sequence of the story. Once the process of designing and testing the created algorithms and maps is completed, each group should prepare how they will "tell" their story to the class. This stage is quite important, sufficient time must be allotted for its completion [42]. The initial positioning of the robot must be aligned with placement of the events in the story map, otherwise it will deviate from the intended trajectory. The teachers, if necessary, should help the students to manage frustration that might arise from the trial and error runs and help them to understand if there is an issue with the algorithm or the positioning of the robot. The teachers must also be on standby to mediate conflicts between the students that may occur during the distribution of the different roles (who programs the robot, tells the story, explain the ideas, answer questions, or any other role that the students consider necessary).

After each presentation, the other groups have the opportunity to comment on the work of their colleagues or suggest improvements. Students should be encouraged to offer constructive criticism. They may address algorithm optimization, debugging, graphic design of the story map, storytelling, or any other aspect that they consider to be important.

At the end of the learning scenario, the algorithm for leaving the room has its last optimization, seeking improvements in its efficiency and simplicity of implementation.

4. Final Considerations

The didactic sequence presented here was designed to allow students to develop the computational thinking skills identified in the model of Atmatzidou and Demetriadis [20]: abstraction, generalisation, algorithm, modularity, and decomposition. Although its design targets the Portuguese formal education context, it can be adapted to other contexts.

Regarding the abstraction skill, the learning scenarios are expected to provide situations that allow students to identify patterns in the different programming created with the robot DOC and with Scratch Jr and look for solutions to a common problem regardless of the type of programming used.

The stages of sharing and improving the created algorithms were included to create conditions that enable students to be able to generalise solutions and explain different steps of the algorithms, as well as to identify different algorithms that enable solving the same problem and finding the most efficient one.

The work presented here is a theoretical proposal. We hope that its implementation may improve the design, particularly regarding the characteristics that promote collaborative learning [43,44], time allotted to each stage of the tasks, and ascertaining the effectiveness of the learning scenarios.

Author Contributions: Conceptualization, R.S. and B.F.; validation B.F.; resources, F.M.; writing—original draft prepara-tion, R.S.; writing—review and editing, R.S., B.F., C.C., F.M.; visualization, C.C.; supervision, C.C, F.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by FCT/MCTES through national funds and when applicable cofunded EU funds under the project UIDB/50008/2020. The work was financially supported by National Funds through FCT — Fundação para a Ciência e a Tecnologia, I.P., under the project UIDB/00194/2020. This work is also financed by national funds through FCT — Fundação para a Ciência e a Tecnologia, I.P., under the doctoral scholarship 2020.06821.BD.The APC was funded by IPC/i2A.

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

References

1. Organisation for Economic Co-Operation and Development. *Education Policy Outlook 2019: Working Together to Help Students Achieve their Potential*; OECD Publishing: Paris, France, 2019.
2. Flores, M.A.; Gago, M. Teacher education in times of COVID-19 pandemic in Portugal: National, institutional and pedagogical responses. *J. Educ. Teach.* **2020**, *46*, 507–516. [CrossRef]
3. Prensky, M. Digital Natives, Digital Immigrants Part 1. *Horizon* **2001**, *9*, 1–6. [CrossRef]
4. Paterson, R. The Power of EMPs: Educational Multimedia Projects. In *Didactics of Smart Pedagogy*; Springer: Cham, Switzerland, 2019; pp. 393–414.
5. Mishra, P. Considering Contextual Knowledge: The TPACK Diagram Gets an Upgrade. *J. Digit. Learn. Teach. Educ.* **2019**, *35*, 76–78. [CrossRef]
6. Binkley, M.; Erstad, O.; Herman, J.; Raizen, S.; Ripley, M.; Miller-Ricci, M.; Rumble, M. Defining twenty-first century skills. In *Assessment and Teaching of 21st Century Skills*; Griffin, P., McGraw, B., Care, E., Eds.; Springer: Dordrecht, The Netherlands, 2012; pp. 17–66. [CrossRef]
7. Piedade, J.; Dorotea, N.; Pedro, A.; Matos, J.F. On Teaching Programming Fundamentals and Computational Thinking with Educational Robotics: A Didactic Experience with Pre-Service Teachers. *Educ. Sci.* **2020**, *10*, 214. [CrossRef]
8. Soulé, H.; Warrick, T. Defining 21st century readiness for all students: What we know and how to get there. *Psychol. Aesthet. Creat. Arts* **2015**, *9*, 178–186. [CrossRef]
9. INTEF. Digital Docente. 2017. Website: Aprendeintef. Available online: https://aprende.intef.es/sites/default/files/2018-05/2017_1020_Marco-Com%C3%BAn-de-Competencia-Digital-Docente.pdf (accessed on 17 August 2021).
10. International Society for Technology in Education. *ISTE Standards for Students: A Practical Guide for Learning with Technology*; ISTE: Washington, DC, USA, 2017.
11. Redecker, C. *European Framework for the Digital Competence of Educators: DigCompEdu*; Publications Office of the European Union: Luxembourg, 2017.
12. Papert, S. *Mindstorms Children, Computers, and Powerful Ideas*, 2nd ed.; Basic Books, Inc.: New York, NY, USA, 1980; Volume 1.
13. Freeman, A.; Becker, S.A.; Cummins, M. *NMC/CoSN Horizon Report: 2017 K-12 Edition*; The New Media Consortium: Austin, TX, USA, 2017.
14. Benitti, F.B.V. Exploring the educational potential of robotics in schools: A systematic review. *Comput. Educ.* **2012**, *58*, 978–988. [CrossRef]
15. Angeli, C.; Jaipal-Jamani, K. Preparing Pre-service Teachers to Promote Computational Thinking in School Classrooms. In *Computational Thinking in the STEM Disciplines*; Springer Science and Business Media LLC: New York, NY, USA, 2018; pp. 127–150.
16. Luciano, A.P.G.; Fusinato, P.A.; Gomes, L.C.; Luciano, A.; Takai, H. The educational robotics and Arduino platform: Constructionist learning strategies to the teaching of physics. *J. Phys. Conf. Ser.* **2019**, *1286*, 012044. [CrossRef]
17. Alimisis, D. Robotics in Education & Education in Robotics: Shifting Focus from Technology to Pedagogy. In Proceedings of the 3rd International Conference on Robotics in Education, Prague, Czech Republic, 13–15 September 2012; Charles University in Prague: Prague, Czech Republic, 2012; pp. 7–14.
18. Kuhl, P.K.; Lim, S.-S.; Guerriero, S.; Van Damme, D. *Developing Minds in the Digital Age*; OECD Publishing: Paris, France, 2019.
19. Zhong, B.; Xia, L. A Systematic Review on Exploring the Potential of Educational Robotics in Mathematics Education. *Int. J. Sci. Math. Educ.* **2018**, *18*, 79–101. [CrossRef]
20. Atmatzidou, S.; Demetriadis, S. Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. *Robot. Auton. Syst.* **2016**, *75*, 661–670. [CrossRef]
21. Angarita, M.G.; Deco, C.; Collazos, C.C. Robotics Based Strategies to Support Computational Thinking: The Case of the Pascual Bravo Industrial Technical Institute. *J. Comput. Sci. Technol.* **2017**, *17*, 59–67.
22. Chalmers, C. Robotics and computational thinking in primary school. *Int. J. Child-Comput. Interact.* **2018**, *17*, 93–100. [CrossRef]
23. Angeli, C.; Voogt, J.; Fluck, A.; Webb, M.; Cox, M.; Malyn-Smith, J.; Zagami, J. A K-6 computational thinking curriculum framework: Implications for teacher knowledge. *Educ. Technol. Soc.* **2016**, *19*, 47–57.
24. Conde, M.Á.; Rodríguez-Sedano, F.J.; Fernández-Llamas, C.; Gonçalves, J.; Lima, J.; García-Peñalvo, F.J. Fostering STEAM through challenge-based learning, robotics, and physical devices: A systematic mapping literature review. *Comput. Appl. Eng. Educ.* **2021**, *29*, 46–65. [CrossRef]
25. Kim, C.; Yuan, J.; Gleasman, C.; Shin, M.; Hill, R.B. Preparing pre-service early childhood teachers to teach mathematics with robots. *Comput. Collab. Learn. Conf. CSCL* **2017**, *2*, 617–620.
26. Muñoz, R.F.Z.; Alegría, J.A.H.; Collazos, C.A.; Fardoun, H. ChildProgramming evolution, a method to increase the computational thinking skills in school. *Commun. Comput. Inf. Sci.* **2019**, *847*, 57–69. [CrossRef]
27. Zhang, L.; Nouri, J. A systematic review of learning computational thinking through Scratch in K-9. *Comput. Educ.* **2019**, *141*, 103607. [CrossRef]
28. Matos, J.F. *Princípios Orientadores Para o Design de Cenários de Aprendizagem*; Instituto de Educação: Lisboa, Portugal, 2014.
29. Pedro, A.; Piedade, J.; Matos, J.F.; Pedro, N. Redesigning initial teacher's education practices with learning scenarios. *Int. J. Inf. Learn. Technol.* **2019**, *36*, 266–283. [CrossRef]
30. Wing, J.M. Computational thinking. *Commun. ACM* **2006**, *49*, 33–35. [CrossRef]

31. Hsu, T.-C.; Chang, S.-C.; Hung, Y.-T. How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Comput. Educ.* **2018**, *126*, 296–310. [CrossRef]
32. Dede, C.; Mishra, P.; Voogt, J. Working Group 6: Advancing computational thinking in 21. *Int. Summit ICT Educ.* **2013**, 1–6. Available online: https://ris.utwente.nl/ws/files/6168377/Advancing_computational_thinking_in_21st_century_learning.pdf (accessed on 14 July 2021).
33. Brennan, K.; Resnick, M. New frameworks for studying and assessing the development of computational thinking. In Proceedings of the 2012 Annual Meeting of the American Educational Research Association, Vancouver, BC, Canada, 13–17 April 2012.
34. National Research Council. *Report of a Workshop on the Pedagogical Aspects of Computational Thinking*; National Academies Press: Washington, DC, USA, 2011.
35. Ramos, J.L.P.; Espadeiro, R.G. Iniciação à Programação no 1o Ciclo do Ensino Básico. Estudos de Avaliação. 2016. Available online: https://www.erte.dge.mec.pt/sites/default/files/estudos_avaliacao_ip1ceb.pdf (accessed on 14 July 2021).
36. Associação Nacional de Professores de Informática. Linhas Orientadoras para a Robótica. DGE—Direção Geral de Educação. 2016. Website: DGE—Direção-Geral da Educação. Available online: https://www.erte.dge.mec.pt/sites/default/files/linhas_orientadoras_para_a_robotica.pdf (accessed on 12 July 2021).
37. Dillenbourg, P. Introduction: What do you mean by 'collaborative learning'? In *Collaborative Learning: Cognitive and Computational Approaches*; Elsevier: Oxford, UK, 1999.
38. Chan, C.K.K. Co-regulation of learning in computer-supported collaborative learning environments: A discussion. *Metacognition Learn.* **2012**, *7*, 63–73. [CrossRef]
39. Stahl, G.; Koschmann, T.; Suthers, D. Computer-supported collaborative learning: An historical perspective. In *Cambridge Handbook of the Learning Sciences*; Sawyer, R.K., Ed.; Cambridge University Press: Cambridge, UK, 2006; pp. 409–426.
40. Alimisis, D. Teacher Training in Educational Robotics: The ROBOESL Project Paradigm. *Technol. Knowl. Learn.* **2019**, *24*, 279–290. [CrossRef]
41. Jeong, H.; Hmelo-Silver, C.E. Seven Affordances of Computer-Supported Collaborative Learning: How to Support Collaborative Learning? How Can Technologies Help? *Educ. Psychol.* **2016**, *51*, 247–265. [CrossRef]
42. Kucuk, S.; Sisman, B. Behavioral patterns of elementary students and teachers in one-to-one robotics instruction. *Comput. Educ.* **2017**, *111*, 31–43. [CrossRef]
43. Soller, A.; Martínez, A.; Jermann, P.; Muehlenbrock, M. From mirroring to guiding: A review of state of the art technology for supporting collaborative learning. *Int. J. Artif. Intell. Educ.* **2005**, *15*, 261–290.
44. Delgado, V.A.; Collazos, C.A.; Fardoun, H.M.; Safa, N. Collaboration Increase Through Monitoring and Evaluation Mechanisms of the Collaborative Learning Process. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*; Meiselwitz, G., Ed.; Springer: Cham, Switzerland, 2017; Volume 10283, pp. 20–31.