education sciences

Article

On Teaching Programming Fundamentals and Computational Thinking with Educational Robotics: A Didactic Experience with Pre-Service Teachers

João Piedade *, Nuno Dorotea, Ana Pedro and João Filipe Matos

UIDEF, Institute of Education, University of Lisbon, 1649-004 Lisbon, Portugal; nmdorotea@ie.ulisboa.pt (N.D.); aipedro@ie.ulisboa.pt (A.P.); jfmatos@ie.ulisboa.pt (J.F.M.)
* Correspondence: jmpiedade@ie.ulisboa.pt

Received: 26 July 2020; Accepted: 19 August 2020; Published: 20 August 2020

Abstract: This study aims to analyze how pre-service informatics teachers design learning scenarios with robotics to teach programming fundamentals and to promote computational thinking skills. A descriptive and exploratory case study design was implemented with 26 pre-service informatics teachers. Data were collected from the participants using qualitative and quantitative instruments. The main results pointed out the affordances and possibilities of the use of learning scenarios with robotics to teach programming fundamentals and to promote computational thinking skills as well as a strong path to promote the application of contents of the other Science, Technology, Engineering, Arts and Mathematics (STEAM) areas. Another significant finding was the impact of the didactic experience on the level of interest and self-confidence of the pre-service teachers in using robotics for teaching purposes. The results showed the importance of these didactics experiences to the pre-service teachers preparation and to apply the pedagogic approaches they have learned in theory in practical activities and to transfer this knowledge to new pedagogical situations and problems.

Keywords: computational thinking; educational robotics; pre-service teacher education; problem-solving; programming

1. Introduction

This paper presents and discusses a didactic experience developed in a pre-service informatics teacher education course related to programming fundamentals, computational thinking, and robotics.

In terms of teacher training, to become an informatics teacher in primary and secondary schools in Portugal, a master's degree is mandatory. The students (pre-service teachers) of this master's program already hold a first degree in informatics or computer science, and are required to attend a two-year master course in teaching to complete the initial teacher training. During this program, pre-service teachers learn about education, research methods, curricula, learning assessments, and didactics of informatics.

According to the portuguese school curriculum, every student should learn about computer science, programming, and computational thinking as well as other digital technologies during the primary and secondary education [1]. The curricular framework [1] defines that students must develop computational thinking skills through a diversity of pedagogical activities, such as unplugged exercises, block-based programming, and educational robotics problem-solving tasks. Educational robotics is a strong path to promote students' skills through problem-solving tasks, and it is an efficient approach to teaching and learning in different learning styles. These changes evidence the need to analyze the preparation of the computer science pre-teachers to use educational robotics in teaching activities in order to achieve the curricular goals.

This paper reports an experience developed with 26 pre-service teachers of the Master in Teaching Informatics program at the University of Lisbon, organized in learning scenarios. In the course of Didactics of Informatics, pre-service teachers are involved in several activities to learn about computer science, computational thinking, block-based and visual programming languages, tangible object programming, and methodologies and strategies in order to teach these contents to students in primary and secondary education.

The focus of this paper is to analyze how pre-service teachers design learning activities with robotics to teach programming contents and to promote computational thinking skills (concepts to be developed in points 2 and 4). According to that, the following research questions were assumed:

- RQ1—What are the previous experiences of pre-service teachers in the use of robotics and block-based apps for teaching purposes?
- RQ2—What didactics and learning approaches are selected by pre-service teachers to support the future implementation of the learning scenarios with real classes of students?
- RQ3—What multi-curricular articulations are promoted in the developed scenarios?
- RQ4—What programming contents can be taught with the different learning scenarios designed by the pre-service teachers?
- RQ5—What computational thinking (CT) skills are present in the different learning scenarios designed by the pre-service teachers?
- RQ6—What is the impact of the experience on the levels of interest, problem-solving, knowledge, and self-confidence of the pre-service teachers in the use of robotics for teaching purposes?

2. Computational Thinking: From Definition to Skills

Computational thinking is a thematic trend in the educational context and an important topic that has received intense attention from educational researchers around the world. Wing [2] defined the term CT as a way of "solving problems, designing systems, and understanding human behavior, by drawing on the contents fundamental to computer science" and "using abstraction and decomposition when attacking a large complex task or designing a large complex system" (p. 32). She argued that CT is a type of analytical thinking with connections with mathematical, scientific, and engineering thinking. More recently, Wing [3] updated her definition of CT and proposed that it includes "... the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent" (p. 3), be it by humans or machines [4].

After Wing, many other authors pointed out definitions of CT as a set of skills related with problem-solving, understanding problems, defining problems, abstraction, logical thinking, debugging, and pattern recognition [1,2,5], as well as managing information effectively and efficiently with emergent technologies in the Big Data age [6]. According to the literature review, we frame the main definitions of each CT skill in Table 1.

Table 1. Computational thinking skills.

CT Skills	Definition
Abstraction	Abstraction is the process of taking away or removing characteristics from something in order to reduce it to a set of essential characteristics.
Decomposition	Decomposition is about breaking problems down into small parts in order to make them easier to solve.
Generalization	Generalization is transferring a problem-solving process to a wide variety of problems.
Pattern Recognition	Recognizing a pattern or similar characteristics helps break down the problem, build a construct as a path for the solution, and find a set of patterns or similar characteristics that can be generalized.
Algorithms	The algorithm is a practice of writing a step-by-step sequence of instructions for carrying out a solution or process.
Flow Control	Process of using different flow control structures.
Data Representation	Process of selection of the appropriate models for data representation.

The core principles of CT are in line with the main attributes of 21st-century skills, such as (collaborative) problem-solving, critical thinking, creativity, communication, innovation, collaboration [7,8], and digital competences [9].

3. Teaching Programming: Fundamentals, Difficulties, and Tools

Programming is one of the computer science areas that, in addition to promoting the development of CT, also promotes and develops concepts such as: algorithm and logical thinking; understanding the syntax, semantics and complexity of a set of different languages; the mastery of the different programming paradigms; and the identification of problems and the respective design and coding of solutions [10]. This critical computer science area is nowadays integrated into primary and secondary school curricula around the world.

Some international organizations (such as Computing at School Group; Computer Science Teacher Association; or Association for Computing Machinery) have defined curricular guidelines to promote students' computational literacy who should learn the essential contents of programming, like algorithms, sequences, variables, conditionals, loops, synchronism, parallelism, procedures, and debugging, to be able to develop solutions to specific problems.

Learning computer programming is a challenging and complex process for many students [11,12], as it requires both a theoretical understanding and a practical application of the syntax and semantics of specific languages, as well as algorithmic thinking and programming skills [10]. According to Robbins et al. (2003), cited by Cheng [13], "one of the major challenges facing novice programming pupils was that they knew the syntax and semantics of each single statement in the programming language, but they did not understand how to combine the statements to create a valid computer program" (p. 362). For this reason, teachers try to understand which type of tool is the most advantageous for their students in learning programming: applications with a higher pedagogical nature, such as block-based applications, or professional text-based language tools. Professional text-based language tools can support the learning of programming, because they are authentic technological tools [14,15] that are used in professional software development; however, text-based languages are also usually very complex for many students [11]. Sáez-López et al. [16], quoting Maya et al. [17], state that "some teachers believe that the only computing experiences are those related to programming languages, such as Java or C++" (p. 3).

Several studies developed in recent years highlight the use of block-based programming tools in primary schools as a way to teach and learn programming, develop CT, or design algorithms to answer specific problems. To help overcome difficulties in the initial learning of programming, the use of block-based programming applications are an excellent alternative instead of traditional text-based tools. [18,19]. Piedade et al. [10] analyzed the core characteristics of 26 block-based apps and pointed out that the majority of the applications are excellent solutions for teaching essential programming contents and supporting computational thinking activities.

4. Educational Robotics as Pedagogic Strategy

Educational robotics could be a secure path to introducing CT and programming in early education through activities that involve students in systematic tasks in order to implement a step-by-step sequence of code needed to program a robot to solve a problem [20], or part of it. Furthermore, educational robotics is a powerful approach to teaching and learning that inspires students to construct and program robots using a specific programming language.

Papert [21] refers that learning is most effective and contextualized when students experiment and construct things by themselves through interaction with objects like computers and robots. According to Bers, Flannery, Kazakoff, and Sullivan [22], young students can use simple block-based programming tools and robotics to develop coding and CT skills. In fact, Papert's [21] constructionist principles, as well as the core patterns of CT proposed by Wing [2] and defined in some CT frameworks [5,23], provide a rationale for learning activities with robotics. Furthermore, educational activities based on the use of robotics can help students to assume a more active role in their learning process, to develop many mental skills, and to create new knowledge. Educational robotics activities can draw from the core principles of Piaget's constructivism, Papert's constructionism, Vygostsky's collaborative learning, and Bruner's discovery learning. Many studies indicate that educational activities have a positive effect on the development of critical thinking, problem-solving, and meta-cognitive skills [24], as well as on the learning of core programming contents [25].

During the learning process of constructing, programming, or interacting with a robot, students apply CT contents, such as abstraction, decomposition, pattern recognition, logical thinking, and debugging [5,26,27]. Additionally, students' achievements and outcomes of their computational practices connect to the problem definition and decomposition, the algorithm designed, the testing and debugging programs, and the robot's performance when solving the problem.

5. Research Methods

According to the research questions, a descriptive and exploratory case study design [28] was organized, and a set of qualitative and quantitative instruments were used to collect and analyze data from participants. According to [28], it is important to utilize both quantitative and qualitative processes and methods that help interpret and manage information to convey clarity and applicability to the results. Furthermore, the ethical guidelines of American Educational Research Association (AERA) and British Educational Research Association (BERA) for educational research were respected, as well as the recommendations of the ethical commission of the Institute of Education of the University of Lisbon. All the participants were informed about the main goals and gave their informed consent in order to participate in the study.

5.1. Pre-Service Computer Science Teacher Education

This case study was implemented in the pre-service teacher education program at the University of Lisbon. The Portuguese model of pre-service teacher education was designed according to the Bologna Process, an intergovernmental cooperation of 48 European countries in the field of higher education. The regulatory Law no. 74/2006 makes it mandatory to have a master's degree in teaching in order to become a teacher in preschool, primary, and secondary education. In addition, Portuguese Law no. 43/2007 defined both the guidelines and the framework for the initial teacher education courses and created master's degrees in teaching in different subject areas (mathematics, science and biology, arts, geography, primary education, informatics, and others). The initial teacher education framework is organized in four mandatory training components: General education, specific didactics, scientific teaching area, and professional practice initiation. Each training component is structured in several curricular units over the two years of training. For example, in specific didactics, a pre-service teacher needs to attend five curricular units of didactics of informatics during the course. These curricular units cover a diversity of areas and themes of computer science teaching, such as teaching hardware, programming languages, databases, robotics, information and communications technologies, computational thinking, or the internet of things. Additionally, the students attend other curricular units for general education, scientific teaching area, and professional practice initiation.

However, the didactic experience reported in this case was developed in the first curricular unit of didactics of informatics of the master's course.

5.2. Participants

This study involved 26 pre-service teachers of the Master in Teaching Informatics program who attended the course of Didactics of Informatics in the last two years. The group of 26 participants was predominantly female (57.6%), and they were between 22 and 56 years old. All participants had a previous bachelor's degree in informatics and were attending the master's program in order to become professional teachers in primary and secondary schools. Through an initial interview, it was possible to identify that 38.4% had some previous experience in teaching programming as non-regular teachers.

The cases were selected based on the research purpose and question, and for what they could reveal about the phenomenon or topic of interest [28].

5.3. Learning and Didactic Experience

The third topic of the didactics of informatics course aims to lead pre-service teachers to explore the programming of tangible objects—such as robots, drones, and mobile devices—as pedagogic strategies to promote the development of CT skills and the learning of programming fundamentals by their future K–12 students.

Two tasks were proposed to the pre-service teachers: (a) An analysis of the core characteristics and pedagogic potentialities of different types of robots (to be developed in groups of students or individually); and (b) the design and implementation of a learning scenario including robots as mediating tools to teach computer science contents to the K–12 students. In this task, each group should choose one of the curricula for the fifth to ninth grade, define a multidisciplinary problem to solve, and design a learning scenario with the robots that they analyzed in task 1.

The two tasks were implemented during the five weeks of the semester. At the end of this period, each group, or individual, presented the learning scenario in the class, as well as the physical models of the scenarios (Appendix A) with robots in action and the possible algorithms and programs (Appendix B) that students should develop to solve the problem. A set of 14 learning scenarios (Appendix A, Appendix B, and Appendix C) was produced by the pre-service teachers, with different types of robots to solve different problems with different levels of complexity.

5.4. Instruments

Data were collected and organized according to qualitative and qualitative techniques. Thus, different types of instruments were used:

(a) Initial focus-group interview to collect the pre-service teachers' experience and knowledge about educational robotics and its use for learning purposes. In this open-ended interview, each participant should share their beliefs and experience concerning educational robotics and computational thinking activities.

(b) Analysis of the learning scenario template designed by the pre-service teachers to analyze the curricular adequacy for the selected grade, the learning approach proposed, the complexity of the solutions, and the curricular articulation with other curricular disciplines. To do that, a cross-analysis of each grade's curriculum, learning approach, and proposed solutions was made.

(c) Rubric for the assessment of programming fundamentals and CT skills presented in the program or programs (problem solutions) proposed by the pre-service teachers. This instrument was organized into five performance descriptors, corresponding to a five-point scale (1 = "unsatisfactory"; 2 = "quite satisfactory"; 3 = "satisfactory"; 4 = "very satisfactory"; 5 = "excellent").

(d) A self-reported scale, adapted from [29], to measure the interest, self-confidence, and knowledge in using educational robotics to teach CT and programming. The scale was structured in 33 items divided into four dimensions (Appendix D): Interest, problem-solving, educational robotics knowledge, and self-confidence. The analysis of the scale's metric quality revealed a high level of internal consistency, with a Cronbach α reliability coefficient of 96 [30]. All 33 items have a good level of sensitivity, with values between −3 and 3 for skewness and −7 and 7 for kurtosis. To analyze the factorial structure of the 33 items, an exploratory factor analysis was made based on the maximum likelihood extraction method. The Kaiser–Meyer–Olkin measure of sample adequacy had a satisfactory level (.57). The result of Bartlett's test of sphericity was significant (χ^2 = 418.34 df = 190, $p < 0.00$), and the four factors explained 73.88% of the variance of the scale.

The data collection process was organized in three phases. Firstly, at the beginning of the semester, two focus-group interviews were done with two groups of participants (11 and 15, respectively).

The participants were asked about their: (a) Previous experiences and knowledge about using robotics to teach programming and CT; (b) expertise about some learning approaches; (c) interest and self-confidence in using robotics in future classroom activities. Secondly, after the didactic experience, an assessment of each learning scenario and solution proposal was made using the cross-analysis and rubric instruments. Finally, at the end of the semester, after the final presentations, pre-service teachers were invited to answer an online self-report scale to measure the impact of the didactic experience on their interest, knowledge, problem-solving, and self-confidence with respect to using robotics in classroom activities.

6. Results

The analysis of the participants' opinions in the initial focus group allowed us to understand their experiences with education robotics programming and its use in classroom activities (RQ1). It was possible to identify that only 23% of pre-service teachers had previous experience in designing and implementing programming activities with robotics in the classroom. Furthermore, all the participants were familiar with core computational thinking aspects, and 31% of them had previous experience "with unplugged computational thinking activities." Almost all of the participants believed that "educational robotics is a good solution to introduce programming into learning activities with young students" because "when kids program the robot to do something, the results of this action are immediate, and it is very important in the learning process." Most of them "expect to learn about pedagogical approaches to support the future learning activities with robotics" and to learn how they "can assess the students' outcomes in robotics activities." Both groups were familiar with the set of learning approaches, which they learned in another curricular course, particularly project-based learning, problem-based learning, game-based learning, flipped classrooms, and learning theories like Constructivism, Constructionism, Behaviorism, Cognitivism, and Connectivism. The results of this open-ended interview were very relevant in deciding the typology of the learning activities that pre-service teachers should be involved in in this experience.

The second research question (RQ2) aimed to analyze the learning approaches selected by pre-service teachers to support the future implementation of the learning scenario with real classes of students. The pre-service teachers were free to select learning approaches to support the implementation of the learning scenario as well as the learning activities. According to the cross-analysis represented in Appendix E, it was possible to identify that five learning scenarios were designed to be implemented in the seventh grade, three to both the fifth and sixth grades, two to the ninth grade, and one to the eighth grade. The learning scenarios were planned with the support of constructionist and constructivist learning theories, particularly using the learning approaches of problem-based (7), project-based (4), or challenge-based (3) learning. To analyze the adequacy of the learning scenarios and the solutions, the national curricular guidelines for each grade, the problem definition, the learning approach, and other resources were produced by the pre-service teachers. This analysis revealed solutions with excellent (3), very good (2), good (4), and satisfactory (5) adequacy. Furthermore, the solutions with higher adequacy were the solutions with more complex problems to solve, which involved the design of complex physical scenarios.

The task's guidelines suggested that pre-service teachers should design multicurricular problems and design educational robotics activities to teach programming contents in combination with other curricular areas. Through the analysis, it was noticed that pre-service teachers followed the recommendation and design activities with multiple articulations (RQ3). To do the articulations, pre-service teachers had to analyze the curricular guidelines of other areas like Portuguese, Visual Arts, Math, or Science.

The assessment of the computational thinking dimensions (RQ4) presented in each educational robotics scenario proposed by the pre-service teachers is presented in Appendix F. This analysis did not intend to assess the pre-service teachers' CT skills, but rather to assess the skills that the scenario may promote in future K–12 students. To accomplish this, the CT dimensions presented in the background topic were used. According to Appendix F, all the CT dimensions were presented in each proposal with different intensity. The "SOS TiNoNi", "Intelligent Forest", and "Two Unlikely Friends" scenarios had the highest scores (4.8, 4.6, and 4.10, respectively), and on the other hand, the "Auto Drive Car", "Pitcher and Batter", and "Moving and Rotate" scenarios had the lowest scores (2.50).

The assessment of essential programming contents (RQ5) applied in each coding solution is presented in Appendix G. All essential contents were covered in the solutions except loops, procedures, parallelism, and synchronization. This analysis did not intend to assess the pre-service teachers' programming knowledge, but rather to assess the programming contents that can be learned and applied by future K–12 students. The results showed that only the "TiNoNi" and the "Intelligent Forest" solutions covered all the contents with excellent quality (5.00 and 4.80, respectively). In fact, these proposals have a high complexity level, solve more complex problems, and use more advanced programming concepts. The "Intelligent Forest" solution uses synchronization and parallelism to implement a fire detection system and alert messages as well as to synchronize the action of two robots. The "TiNoNi" solution uses a set of procedures to implement a diverse set of actions, parallelism, and synchronization to define the communication between three robots based on an emergency sound captured by sound sensors.

To analyze the levels of interest, problem-solving, educational robotics knowledge, and self-confidence presented by pre-service teachers (RQ6), after the didactic experience, a matrix of descriptive scores of each item was constructed (Appendix D). The mean scores observed in each dimension are represented in Table 2. Pre-service teachers had high scores in the four dimensions; however, the "Interest" and "Problem-Solving" dimensions presented higher scores in comparison with the others (M = 4.44, SD = 0.55; M = 4.22, SD = 0.57, respectively), and the "Educational Robotics Knowledge" had the lowest score (M = 3.67, SD = 0.72). According to the Likert-type scale (between 1 and 5), scores between 1 and 2.4 are weak, between 2.5 and 3.4 are moderate, and between 3.5 and 5 are high.

Table 2. Mean and standard-deviation (SD) of each dimension (n = 26).

Dimensions	Mean	SD
Interest	4.44	0.55
Problem-solving	4.22	0.57
Educational robotics knowledge	3.67	0.72
Self-confidence	3.93	0.73

To analyze the statistical significance of the correlation between the dimensions, an analysis of the Pearson correlation was made, and the results are presented in Table 3.

Table 3. Pearson's correlation coefficient between the scale's dimensions (n = 26).

	Interest	Problem Solving	Educational Robotics Knowledge
Problem-solving	0.84 **		
Educational robotics knowledge	0.65 **	0.53 **	
Self-confidence	0.69 **	0.66 **	0.82 **

(** Significant $p < 0.001$).

The analysis of Pearson's correlation coefficient revealed a statistically significant correlation between all dimensions ($0.53 < r < 0.84$, $p < 0.001$). A high level of correlation was found between "Problem-Solving" and "Interest" ($r = 0.84$, $p < 0.001$), as well as between "Self-Confidence" and "Educational Robotics Knowledge" ($r = 0.82$, $p < 0.001$). Accordingly, the linear regression model was made between these dimensions.

The linear regression of the "Interest" score as the predictor of the "Problem-Solving" score suggests that the pre-service "Interest" can explain 70% of the variance in the "Problem-Solving", score and the regression model predicts a significantly high problem-solving level ($F(1,24) = 56.47$, $p < 0.001$; $r^2 = 0.70$). The results show that each one-point increase in the "Interest" score also increases the "Problem-Solving" score by 0.81 ($b1 = 0.81$, $t = 7.52$, $p < 0.001$).

The linear regression of the "Educational Robotics Knowledge" score as the predictor of the "Self-Confidence" score suggests that the pre-service "Educational Robotics Knowledge" can explain 67% of the variance in the "Self-Confidence" score, and the regression model predicts a significantly high self-confidence level ($F(1,24) = 48.98$, $p < 0.001$; $r^2 = 0.67$). The results show that each one-point increase in the "Educational Robotics Knowledge" score also increases the "Self-Confidence" score by 0.81 ($b1 = 0.81$, $t = 6.99$, $p < 0.001$).

7. Implications for Pre-Service Teacher Education

This research has some methodological limitations. Firstly, the sampling process and the small size of the sample do not allow the generalization of the results; secondly, it was not possible to analyze the implementation of these pedagogical activities with real classes of K–12 students. To prove the results, in the next academic year, the pre-service teachers will implement some of these learning scenarios with real classes of students in their supervised pedagogical practice in schools.

Despite the limitations, a set of relevant results were systematized about how pre-service teachers put the aspects of the theory they have learned in the preparation of the learning scenarios with robotics into practice. The design of learning scenarios, supported in active learning approaches, is a regular practice in pre-service teacher education at the University of Lisbon [31,32]. For Tetchueng, Garlatti, and Laube [33], learning scenarios are a powerful tool to plan and describe learning activities to acquire knowledge domains and know-how to solve particular problems. Along the same line, Misfeldt [34] defines a learning scenario in education as a "newly developed framework or approach to understanding educational situations building on scenarios, understood as real or artificial situations that are used to create context, experience of relevance, and immersion in educational situations" (p. 183). This study shows that learning scenarios are an important tool to design and plan learning tasks with robotics, anticipate problems, think about solutions for these problems, and establish bridges with other curricular areas. The first implication of this study is the possibility of the extension of the scenario-based strategy to other master's programs in teaching and also to in-service training initiatives.

Other relevant results are related to the use of educational robotics as a strategy to introduce programming to young students and to develop CT skills. In educational robotics activities, students can learn and apply programming fundamentals and observe the tangible results of the programs they have coded immediately. This principle is coined by Papert [21] as constructionist, meaning that learning is most effective and contextualized when students experiment and construct things (both physical as well as conceptual) by themselves through the interaction with objects like computers and robots. Through the learning process of constructing and programming a robot to solve a problem, students deal with computational thinking skills, such as abstraction, decomposition, logical thinking, pattern recognition, design of algorithms, and debugging [26,28]. Furthermore, programming robotics with blocked-based apps saves the students from the difficulties of traditionally complex text-based languages [18,19]. The pedagogical strategy of using educational robotics has been referred to in many studies as a powerful approach to teaching and learning programming, to developing CT skills, to developing 21st-century skills [9,20], and to developing reading and writing literacy [35]. The main results of this study are totally aligned with that perspective.

The second implication of the study is the possibility of using these examples as formative "good practice" resources with new pre-service teachers to consolidate the practice of using educational robotics in the initial teacher training programs. The initial teacher training courses are the place where pre-service teachers should design, plan, experiment, and test learning tasks and pedagogical approaches without the stress of the real classroom. This contact with good practices is an essential factor for the training of pre-service teachers.

Finally, the results of the self-report scale analysis revealed high levels of interest and self-confidence in using educational robotics as well as high levels of knowledge about robotics and problem-solving skills. Considering that only 23% of the pre-service teachers of the sample had previous experience in designing and implementing programming activities with robotics in the classroom, and 31% had previous contact with unplugged CT activities, the results of the pedagogical experience reported here have an important impact on the four dimensions of CT. The linear regression analysis showed a strong correlation between "Interest" and "Problem-Solving" as well as between the "Educational Robotics" and "Self-Confidence" dimensions. Thus, the third implication of the study refers to the importance of the development of these dimensions in an integrated way. Pre-service teachers should be challenged with collaborative problem-solving learning activities in order to develop or improve their skills in each dimension.

When pre-service teachers are involved in planning, designing, and implementing learning scenarios with robots and thinking about the solutions, they rethink all the possible pedagogic approaches that they learned in theory and are able to transfer and use knowledge in new situations and problems.

8. Conclusions

Computational thinking has received intense attention as an essential skill that all 21st-century citizens should develop. According to that, CT, programming, and robotics have been integrated into schools' curricula in many countries around the world. In Portugal, the national curriculum for primary and secondary education include the learning of programming, computational thinking, and educational robotics as mandatory for every student between fifth and ninth grade.

According to the national regulation, in order to become an informatics teacher in primary and secondary education, it is compulsory to obtain a master's degree in informatics (computer science) education.

The results of the scenario-based didactic experience developed and researched with 26 pre-service informatics teachers enrolled in the master's program in teaching informatics at the University of Lisbon pointed out the affordances and potential of the use of learning scenarios with robotics to teach programming fundamentals as well as to promote computational thinking skills. Furthermore, a strong path to promote the use and application of the contents of the other STEAM areas becomes apparent. The impact of the reflection of pre-service teachers on the use of scenario-based teaching strategies that include robotics is also clear at the level of interest and self-confidence of the pre-service teachers in the use of robotics for teaching purposes. Thus, the results of the study contribute to the consolidation of the recommendation of using robotics in initial teacher education programs to promote better pedagogical preparation for future teachers of informatics and computer science in both primary and secondary schools.

As a consequence of this small-sample case study, it would be essential to design and implement a large-scale study involving other pre-service STEAM teachers in order to analyze the impact of the robotics learning scenarios on pre-service teachers' and K–12 students' skills. The development of CT skills in K–12 students has been pointed out in many international reports as crucial, and to promote that would be important to recognize the importance of the pre-service and in-service teacher training experiences.

Author Contributions: Conceptualization, J.P., N.D., A.P. and J.F.M.; Investigation, J.P., N.D., A.P. and J.F.M.; Methodology, J.P., N.D., A.P. and J.F.M.; Writing—original draft, J.P., N.D., A.P. and J.F.M.; Writing—review & editing, J.P. and J.F.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest

Appendix A

Table A1. List of robotics learning scenarios.

Learning Scenario	Problems to Solve	Robot	Programming Apps
LS1: Intelligent Forest (a group of 2 elements)	How can we make forests more intelligent? Implementing automatic fire detection and fireman activation.	m-Bot	mBlock
LS2: A Walk through the Forest (a group of 2 elements)	Implement a robotic quiz about forest diversity and biodiversity.	Lego NXT	Lego Mindstorms NXT 2.0
LS3: Two Unlikely Friends (a group of 3 elements)	Create stories with unlikely friends. Synchronization of two different robots.	m-Bot and Zowi	mBlock and bitbloq
LS4: STEM Things (individual)	Using m-Bot robot to implement different math and science learning activities.	m-Bot	mBlock
LS5: Auto-Drive Car (individual)	Using Dash & Dot robot to simulate some problems of cars without drivers.	Dash & Dot	Blockly
LS6: Mathematical Things (individual)	Using Anprino robot to learn areas and perimeters in math activities.	Anprino	Ardublockly
LS7: SOS TiNoNi (a group of 2 elements)	Simulation of the building's emergency evacuation.	m-Bot	mBlock
LS8: Robotic Story (a group of 2 elements)	Create a story with robots. Synchronization of two different robots.	m-Bot and Codey Rocky	mBlock
LS9: Solve the Maze (a group of 2 elements)	Solving a maze starting from different points.	m-Bot	mBlock
LS10: EcoRobot (a group of 2 elements)	Implementation of a waste-sorting system.	Codey Rocky	mBlock
LS11: Solve the Maze II (a group of 2 elements)	Solving a simple maze through the user's inputs (right or left).	Zowi	Bitbloq
LS12: Fortnite and Lion King (a group of 2 elements)	Create a story based on the Fortnite game and Lion King film. Synchronization of two different robots.	Dash and Zowi	Bitbloq and Blockly
LS13: Pitcher and Batter (a group of 2 elements)	Simulation of baseball pitcher and batter.	Lego Mindstorm NXT	Lego Mindstorms NXT 2.0
LS14: Moving and Rotate (Individual)	Simple movements (front, rear, left, right, rotate, etc.).	Lego Mindstorm NXT	Lego Mindstorms NXT 2.0

Appendix B

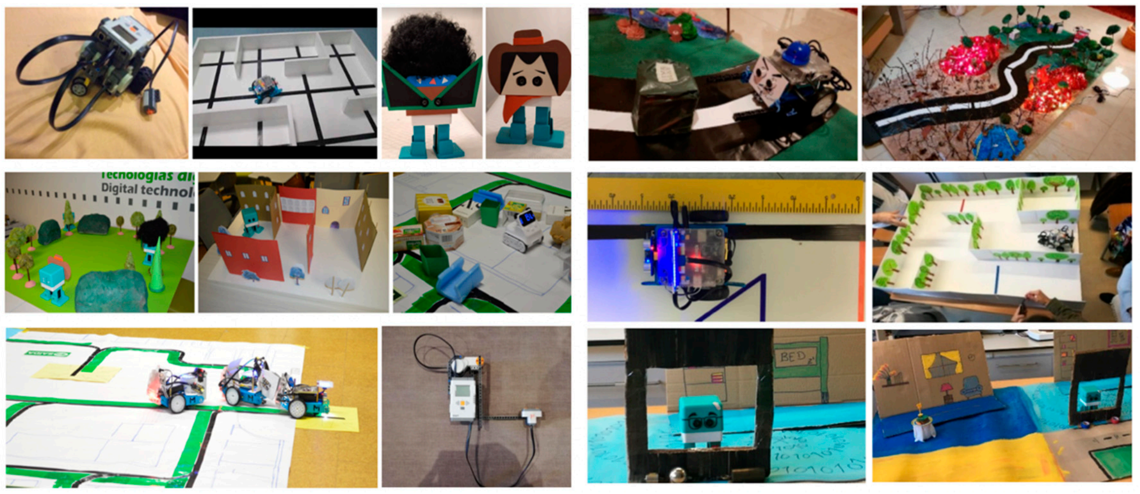
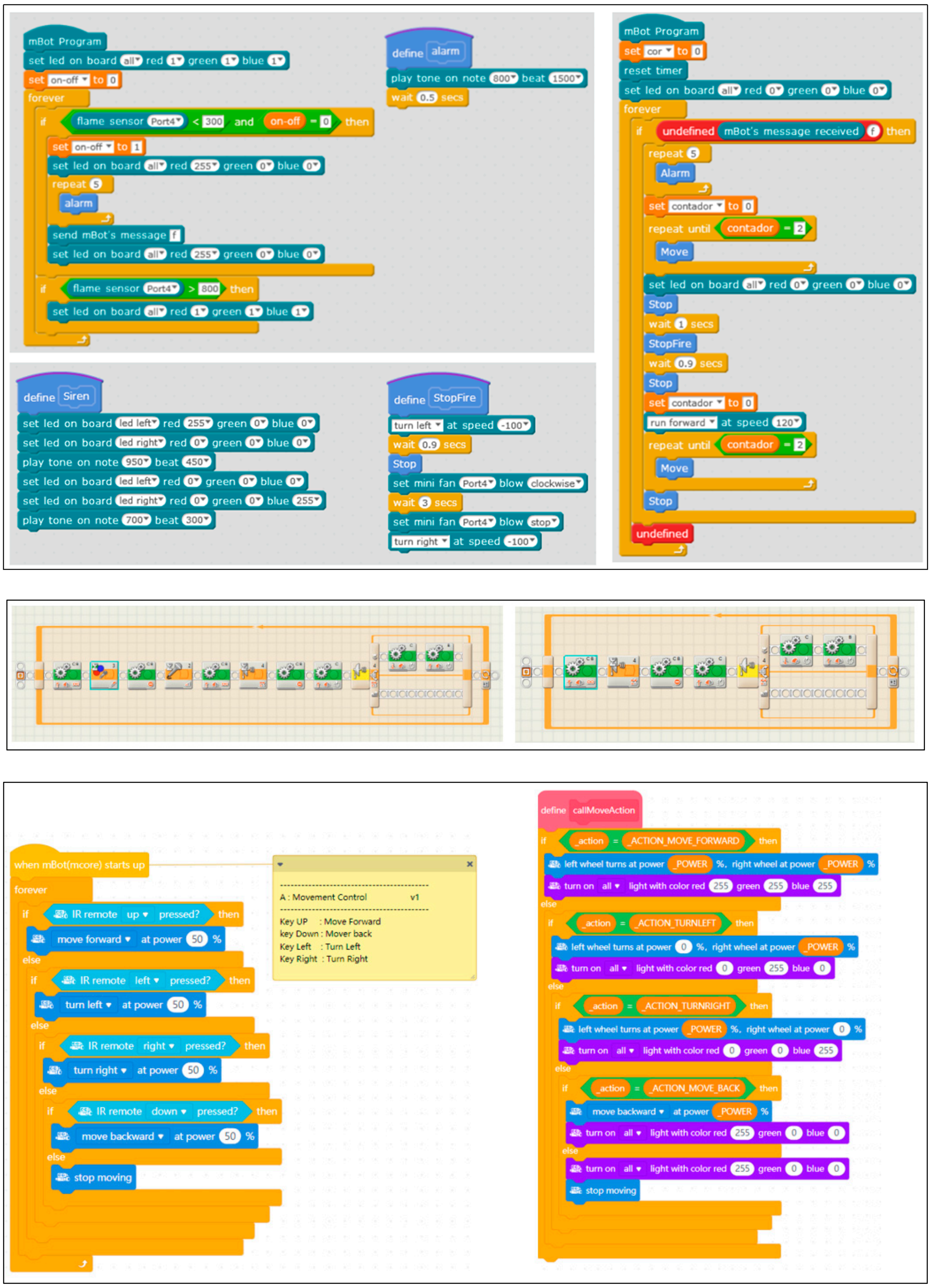


Figure A1. Example of the physical models of learning scenarios.

Appendix C



```
mBot Program
set led on board all red 1 green 1 blue 1
set on-off to 0
forever
  if flame sensor Port4 < 300 and on-off = 0 then
    set on-off to 1
    set led on board all red 255 green 0 blue 0
    repeat 5
      alarm
    send mBot's message f
    set led on board all red 255 green 0 blue 0
  if flame sensor Port4 > 800 then
    set led on board all red 1 green 1 blue 1
```

```
define alarm
play tone on note 800 beat 1500
wait 0.5 secs
```

```
define Siren
set led on board led left red 255 green 0 blue 0
set led on board led right red 0 green 0 blue 0
play tone on note 950 beat 450
set led on board led left red 0 green 0 blue 0
set led on board led right red 0 green 0 blue 255
play tone on note 700 beat 300
```

```
define StopFire
turn left at speed -100
wait 0.9 secs
Stop
set mini fan Port4 blow clockwise
wait 3 secs
set mini fan Port4 blow stop
turn right at speed -100
```

```
mBot Program
set cor to 0
reset timer
set led on board all red 0 green 0 blue 0
forever
  if undefined mBot's message received f then
    repeat 5
      Alarm
    set contador to 0
    repeat until contador = 2
      Move
    set led on board all red 0 green 0 blue 0
    Stop
    wait 1 secs
    StopFire
    wait 0.9 secs
    Stop
    set contador to 0
    run forward at speed 120
    repeat until contador = 2
      Move
    Stop
  undefined
```

```
when mBot(mcore) starts up
forever
  if IR remote up pressed? then
    move forward at power 50 %
  else
    if IR remote left pressed? then
      turn left at power 50 %
    else
      if IR remote right pressed? then
        turn right at power 50 %
      else
        if IR remote down pressed? then
          move backward at power 50 %
        else
          stop moving
```

```
------------------------------------------
A : Movement Control              v1
------------------------------------------
Key UP    : Move Forward
key Down : Mover back
Key Left   : Turn Left
Key Right  : Turn Right
```

```
define callMoveAction
if _action = _ACTION_MOVE_FORWARD then
  left wheel turns at power _POWER %, right wheel at power _POWER %
  turn on all light with color red 255 green 255 blue 255
else
  if _action = _ACTION_TURNLEFT then
    left wheel turns at power 0 %, right wheel at power _POWER %
    turn on all light with color red 0 green 255 blue 0
  else
    if _action = _ACTION_TURNRIGHT then
      left wheel turns at power _POWER %, right wheel at power 0 %
      turn on all light with color red 0 green 0 blue 255
    else
      if _action = _ACTION_MOVE_BACK then
        move backward at power _POWER %
        turn on all light with color red 255 green 0 blue 0
      else
        turn on all light with color red 255 green 0 blue 0
        stop moving
```

Figure A2. Example of pieces of code.

Appendix D

Table A2. Robotics interest questionnaire (adapted from Jaipal-Jamani and Angeli, 2017).

Robotics Interest	**Mean (n = 26)**	**SD**
I1. I like learning about new technologies like robotics.	4.62	0.57
I2. I like using scientific methods to solve problems.	4.42	0.76
I3. I like using mathematical formulas and calculations to solve problems.	4.23	0.95
I4. I think careers in science, technology, engineering, or math are interesting.	4.62	0.57
I5. I would like to learn more about careers that involve science, technology engineering, and mathematics.	4.27	0.92
I6. I find it interesting to learn about robots or robotics technology.	4.62	0.57
I7. I would like to use robotics to learn mathematics or science.	4.15	0.73
I8. I would use robotics in my classroom teaching.	4.62	0.70
Total Score:	4.44	0.55
Problem-Solving		
P1. I use a step-by-step process to solve problems.	4.42	0.64
P2. I make a plan before I start to solve a problem.	4.00	0.75
P3. I try new methods to solve a problem when one does not work.	4.31	0.74
P4. I carefully analyze a problem before I begin to develop a solution.	4.04	0.96
P5. In order to solve a complex problem, I break it down into smaller steps.	4.19	0.85
P6. I like listening to others when trying to decide how to approach a task or problem.	4.31	0.62
P7. I like being part of a team that is trying to solve a problem.	4.42	0.76
P8. When working in teams, I ask my teammates for help when I run into a problem or do not understand something.	4.42	0.64
P9. I am confident that I could learn how to make a robot do something that I had not done before today.	3.96	0.92
P10. I believe that I could work with a robot in a science investigation.	4.15	0.93
P11. I believe that I could fix a software problem if I needed to do so.	4.08	0.98
P12. I like to work with others to complete projects.	4.38	0.64
Total Score:	4.22	0.57
Educational Robotics Knowledge		
K1. I have sufficient knowledge about robotics for use in teaching and learning activities.	3.65	0.98
K2. I have sufficient knowledge of coding as it applies to robotics.	3.62	0.90
K3. I have sufficient knowledge of the engineering and design process as it applies to robotics.	3.54	1.03
K4. I have sufficient knowledge to select the most appropriate robot for teaching and learning according to students' ages.	3.50	0.86
K5. I have sufficient knowledge to analyze the pedagogical potentialities of different types of robots.	3.77	0.81
K6. I have sufficient knowledge about block-based programming apps that can be used to teach programming concepts.	3.92	0.69
Total Score:	3.66	0.72
Self-Confidence		
S1. I feel confident that I have the necessary skills to use robotics for classroom instruction.	3.85	0.83
S2. I feel confident that I can engage my students to participate in robotics-based projects.	4.00	0.85
S3. I feel confident that I can help students when they have difficulties with robotics.	3.85	0.93
S4. I feel confident that I can plan and design learning scenarios with robotics.	3.96	0.87
S5. I feel confident about teaching computer science with different types of robotics.	3.73	0.87
S6. I feel confident that I can assess students' outcomes in robotics learning activities.	3.73	1.00
S7. I feel confident that robotics is a good strategy to teach computer science concepts.	4.42	0.50
Total Score:	3.93	0.73

Appendix E

Table A3. Cross-analysis of the learning scenarios.

Learning Scenario	Grade	Learning Approach	Solution Adequacy	Curricular Articulation
LS1. Intelligent Forest	7th	Constructionism Problem-based Learning	Excellent	Physics and Chemistry, Visual Arts, Science, Math
LS2. A Walk Through the Forest	7th	Project-based Learning	Good	Citizenship, Visual Arts, Portuguese, Math
LS3. Two Unlikely Friends	5th	Project-based Learning	Excellent	Portuguese, Visual Arts
LS4. STEM Things	6th	Problem-based Learning	Very Good	Math, Science
LS5. Auto Drive Car	7th	Challenge-based Learning	Satisfactory	Citizenship, Visual Arts
LS6. Mathematical Things	7th	Constructivism Problem-based Learning	Satisfactory	Math
LS7: SOS TiNoNi	9th	Problem-based Learning	Excellent	Math, Science, Physic and Chemistry, Citizenship, Visual Arts
LS8: Robotic Story	5th	Problem-based Learning	Good	Sciences, Math and Music Education
LS9: Solve the Maze	6th	Problem-based Learning	Good	Visual Arts
LS10: EcoRobot	9th	Project-Based Learning	Good	Math, Science, Physic and Chemistry, Citizenship, Visual Arts
LS11: Solve the Maze II	7th	Challenge-based Learning	Satisfactory	Math and Visual Arts
LS12: Fortnite and Lion King	5th	Constructionism Challenge-based Learning	Very Good	Visual Arts and Sciences
LS13: Pitcher and Batter	8th	Project-based Learning	Satisfactory	Math
LS14: Moving and Rotate	6th	Problem-based Learning	Satisfactory	undefined

Appendix F

Table A4. Assessment of computational thinking dimensions (1 = "unsatisfactory"; 2 = "quite satisfactory"; 3 = "satisfactory"; 4 = "very satisfactory"; 5 = "excellent").

Computational Thinking Skills	LS 1	LS 2	LS 3	LS 4	LS 5	LS 6	LS 7	LS 8	LS 9	LS 10	LS 11	LS 12	LS 13	LS 14
Abstraction	4.00	3.00	4.00	3.00	3.00	3.00	5.00	4.00	4.00	3.00	3.00	3.00	3.00	3.00
Decomposition	5.00	4.00	5.00	4.00	3.00	3.00	5.00	4.00	4.00	4.00	4.00	3.00	3.00	3.00
Generalization	5.00	4.00	4.00	4.00	3.00	3.00	5.00	4.00	4.00	4.00	3.00	3.00	3.00	3.00
Pattern Recognition	3.00	3.00	4.00	3.00	2.00	2.00	4.00	3.00	4.00	3.00	4.00	2.00	2.00	2.00
Algorithms	5.00	4.00	5.00	4.00	3.00	3.00	5.00	4.00	4.00	4.00	5.00	3.00	3.00	3.00
Flow Control	5.00	4.00	4.00	4.00	2.00	2.00	5.00	4.00	3.00	4.00	4.00	3.00	2.00	2.00
Data Representation	4.00	4.00	4.00	5.00	2.00	2.00	4.00	4.00	3.00	4.00	4.00	2.00	2.00	2.00
Parallelism	5.00	2.00	3.00	3.00	2.00	3.00	5.00	3.00	3.00	2.00	2.00	2.00	2.00	2.00
Synchronization	5.00	2.00	3.00	3.00	2.00	2.00	5.00	3.00	4.00	4.00	2.00	2.00	2.00	2.00
Testing and Debugging	5.00	4.00	5.00	4.00	3.00	3.00	5.00	4.00	4.00	4.00	4.00	3.00	3.00	3.00
Mean	4.60	3.40	4.10	3.70	2.50	2.60	4.80	3.70	3.70	3.60	3.50	2.60	2.50	2.50

Appendix G

Table A5. Assessment of programming contents (1 = "unsatisfactory"; 2 = "quite satisfactory"; 3 = "satisfactory"; 4 = "very satisfactory"; 5 = "excellent").

Programming Fundamentals	LS 1	LS 2	LS 3	LS 4	LS 5	LS 6	LS 7	LS 8	LS 9	LS 10	LS 11	LS 12	LS 13	LS 14
Algorithm	5.00	4.00	5.00	4.00	3.00	3.00	5.00	4.00	4.00	4.00	3.00	3.00	3.00	3.00
Sequences	5.00	4.00	5.00	4.00	3.00	3.00	5.00	4.00	4.00	4.00	4.00	3.00	3.00	3.00
Input/Output Operations	5.00	3.00	4.00	4.00	2.00	2.00	5.00	4.00	4.00	4.00	3.00	3.00	3.00	3.00
Arithmetic, Relational, and Logical Operators	5.00	4.00	5.00	4.00	3.00	2.00	5.00	4.00	4.00	3.00	4.00	3.00	3.00	3.00
Variables and Constants	4.00	3.00	4.00	4.00	2.00	3.00	5.00	4.00	2.00	4.00	2.00	3.00	3.00	2.00
Conditional Structures	5.00	4.00	4.00	4.00	3.00	3.00	5.00	4.00	4.00	4.00	4.00	2.00	3.00	2.00
Loops	5.00	4.00	4.00	4.00	2.00	2.00	5.00	3.00	4.00	4.00	4.00	3.00	3.00	3.00
Procedures	4.00	1.00	1.00	2.00	1.00	1.00	5.00	3.00	2.00	1.00	1.00	2.00	1.00	1.00
Parallelism	5.00	2.00	3.00	3.00	2.00	3.00	5.00	4.00	4.00	3.00	1.00	2.00	1.00	2.00
Synchronism	5.00	2.00	3.00	3.00	2.00	2.00	5.00	4.00	4.00	4.00	4.00	3.00	3.00	3.00
Testing and Debugging	5.00	4.00	5.00	4.00	3.00	3.00	5.00	4.00	4.00	4.00	4.00	3.00	3.00	3.00
Mean	4.82	3.18	3.91	3.64	2.36	2.45	5.0	3.81	3.64	3.64	3.00	2.72	2.64	2.55

References

1. DGE. *Aprendizagens Essenciais para da Disciplina de TIC*; Direção-geral da Educação. Ministério da Educação de Portugal: Lisboa, Portugal, 2017.
2. Wing, J.M. Computational thinking. *Commun. ACM* **2006**, *49*, 33–35. [CrossRef]
3. Wing, J.W. Computational thinking. In Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing, Pittsburgh, PA, USA, 18–22 September 2011; p. 3.
4. Gotlieb, C.C.; Borodin, A. Computational Thinking Benefits Society. In *Social Issues in Computing*; Wing, J.M., Ed.; Academic Press: New York, NY, USA, 2014.
5. 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.
6. Burke, Q.; O'Byrne, W.I.; Kafai, Y.B. Computational participation: Understanding coding as an extension of literacy instruction. *J. Adolesc. Adult Lit.* **2016**, *59*, 371–375. [CrossRef]
7. 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., McGaw, B., Care, E., Eds.; Springer: Dordrecht, The Netherlands, 2012; pp. 17–66. [CrossRef]
8. Repenning, A.; Webb, D.; Koh, K.H.; Nickerson, H.; Miller, S.B.; Brand, C.; Horses, I.H.M.; Basawapatna, A.; Gluck, F.; Grover, R.; et al. Scalable Game Design: A strategy to bring systemic computer science education to schools through game design and simulation creation. *ACM Trans. Comput. Educ.* **2015**, *15*, 1–31. [CrossRef]
9. Juškevičienė, A.; Dagienė, V. Computational Thinking Relationship with Digital Competence. *Inf. Educ.* **2018**, *17*, 265–284. [CrossRef]
10. Piedade, J.; Dorotea, D.; Sampaio, F.F.; Pedro, A. A Cross-analysis of Block-based and Visual Programming Apps with Computer Science Student-Teachers. *Educ. Sci.* **2019**, *9*, 181. [CrossRef]
11. Martins, S.W.; Mendes, A.J.; Figueiredo, A.D. Diversifying Activities to Improve Student Performance in Programming Courses. *Commun. Cogn.* **2013**, *46*, 39–58. [CrossRef]
12. Jenkins, T. On the difficulty of learning to program. In Proceedings of the 3rd Annual Conference of LTSN-ICS, Loughborough, UK, 23 August 2002.
13. Cheng, G. Exploring factors influencing the acceptance of visual programming environment among boys and girls in primary schools. *Comput. Hum. Behav.* **2019**, *92*, 361–372. [CrossRef]
14. Navarrete, C.C. Creative thinking in digital game design and development: A case study. *Comput. Educ.* **2013**, *69*, 320–331. [CrossRef]
15. Webb, H.C.; Rosson, M.B. Exploring careers while learning Alice 3D: A summer camp for middle school girls. In Proceedings of the 42nd ACM Technical Symposium on Computer Science Education, Dallas, TX, USA, 9–12 March 2011; ACM: New York, NY, USA, 2011; pp. 377–382. [CrossRef]

16. López, J.M.S.; Roman-Gonzalez, M.; Vázquez-Cano, E. Visual programming languages integrated across the curriculum in elementary school: A two year case study using "Scratch" in five schools. *Comput. Educ.* **2016**, *97*, 129–141. [CrossRef]
17. Israel, M.; Pearson, J.N.; Tapia, T.; Wherfel, Q.M.; Reese, G. Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Comput. Educ.* **2015**, *82*, 263–279. [CrossRef]
18. Franklin, D.; Skifstad, G.; Rolock, R.; Mehrotra, I.; Ding, V.; Hansen, A.; Weintrop, D.; Harlow, D. Using upper-elementary student performance to understand conceptual sequencing in a blocks-based curriculum. In Proceedings of the 2017 ACM SIGCSE Technical Symposium Computer Science Education, Seattle, WA, USA, 8–11 March 2017; ACM: New York, NY, USA, 2017; pp. 231–236. [CrossRef]
19. Wilson, A.; Moffat, D.C. Evaluating Scratch to Introduce Younger Schoolchildren to Programming. 2010. Available online: http://scratched.gse.harvard.edu/sites/default/files/wilson-moffat-ppig2010-final.pdf (accessed on 20 August 2020).
20. Chalmers, C. Robotics and computational thinking in primary school. *Int. J. Child-Comput. Interact.* **2018**, *17*, 93–100. [CrossRef]
21. Papert, S. *Mindstorms: Children, Computers, and Powerful Ideas*; Basic Books: New York, NY, USA, 1980.
22. Bers, M.U.; Flannery, L.; Kazakoff, E.R.; Sullivan, A. Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Comput. Educ.* **2014**, *72*, 145–157. [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. *Int. J. Robot. Auton. Syst.* **2016**, *59*, 371–375.
24. Atmatzidou, S.; Demetriadis, S. Evaluating the Role of Collaboration Scripts as Group Guiding Tools in Activities of Educational Robotics: Conclusions from Three Case Studies. In Proceedings of the 2012 IEEE 12th International Conference on Advanced Learning Technologies, Rome, Italy, 4–6 July 2012; pp. 298–302. [CrossRef]
25. Alimisis, D. *Teacher Education on Robotics-Enhanced Constructivist Pedagogical Methods*; School of Pedagogical and Technological Education: Heraklion, Greece, 2009.
26. Chalmers, C.; Nason, R. Systems Thinking Approach to Robotics Curriculum in Schools. In *Robotics in STEM Education: Redesigning the Learning Experience*; Kline, M., Ed.; Springer: Dordrecht, The Netherlands, 2017; Volume 70.
27. Kazakoff, R.E.; Bers, M. Put your robot in, put your robot out: Sequencing through programming robots in early childhood. *J. Educ. Comput. Res.* **2014**, *50*, 553–573. [CrossRef]
28. Merriam, S.B. *Qualitative Research: A Guide to Design and Implementation*, 2nd ed.; Jossey-Bass: San Francisco, CA, USA, 2009.
29. Jaipal-Jamani, K.; Angeli, C. Effect of Robotics on Elementary Preservice Teachers' Self-Efficacy, Science Learning, and Computational Thinking. *J. Sci. Educ. Technol.* **2016**, *26*, 175–192. [CrossRef]
30. Field, A. *Discovering Statistics Using SPSS*, 3rd ed.; SAGE: London, UK, 2009.
31. Pedro, A.; Piedade, J.; Matos, J.F. Cenários de Aprendizagem na Formação Inicial de Professores de Informática. *Revista Lusófona Educação* **2019**, *45*, 219–234. [CrossRef]
32. 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]
33. Tetchueng, J.; Garlatti, S.; Laube, S. A context-aware learning system based on generic scenarios and the theory in didactic anthropology of knowledge. *Int. J. Comput. Appl.* **2008**, *5*, 71–87.
34. Misfeldt, M. Scenario based education as a framework for understanding students engagement and learning in a project management simulation game. *Electron. J. e-Learn.* **2015**, *13*, 181–191.
35. Hassenfeld, Z.R.; Govind, M.; De Ruiter, L.E.; Bers, M.U. If You Can Program, You Can Write: Learning Introductory Programming Across Literacy Levels. *J. Inf. Technol. Educ. Res.* **2020**, *19*, 65–85. [CrossRef]