Journal of Pedagogical Sociology and Psychology
Volume 6, Issue 3, 2024
https://doi.org/10.33902/jpsp.202426267

Research Article

The effect of block-based programming activities on the computational thinking skills of pre-service primary school teachers

Jan Pršala

Department of Applied Cybernetics, Faculty of Science, University of Hradec Králové, Czechia

Correspondence should be addressed to Jan Pršala jprsala@jcu.cz
Received 27 January 2024; Revised 30 April 2024; Accepted 20 May 2024

The aim of this paper was to analyse the level of computational thinking among pre-service primary school teachers and to measure the effect of block-based activities, in a form of programming projects in Scratch, on the computational thinking and programming skills developed by these pre-service teachers. To assess their knowledge, the Beginners Computational Thinking Test was used. The results indicate that pre-service teachers have a high level of understanding of sequences and loops, but low level of understanding of conditionals. A positive statistically significant difference was found in the understanding before and after they used Scratch.

Keywords: Students' gender, school location, correlation, academic achievement in economics

1. Introduction

Computational thinking is increasingly becoming part of 21st century education. The essence of computational thinking is to look at a problem as a computer scientist (Lessner, 2014). It is about formulating problems in such a way that a computer can be effectively used to solve it (Stephens & Buteau, 2023). Thus, computational thinking becomes important for all disciplines, for example, for architects, doctors, or teachers (Wing, 2006). Students must use new skills, often referred to as the components of computational thinking, to solve such problems. These include algorithmization, abstraction, generalization, pattern recognition, and automation (Brennan & Resnick, 2012).

Programming is one of the most effective ways to develop computational thinking (Resnick et al., 2009). However, computational thinking is not just about programming. While the goal of teaching programming is to find a solution and then implement it in a particular programming language, computational thinking seeks to help students understand the basic concepts and mechanisms of digital technologies for formulating and solving problems (Bocconi et al., 2022). Therefore, computational thinking serves as an umbrella concept that encompasses the fundamental intellectual underpinnings necessary for understanding the digital world (Fagerlund et al., 2020). Xia (2017) also defines programming education as supporting students to understand the concepts of programming by problem-solving and their own experience.

In the Czech Republic, a new Framework Curriculum was introduced in 2021 (Ministry of Education, Youth and Sports of the Czech Republic & NPI Czech Republic, 2021), which defines "informatics" as a new area of education that focuses on developing computational thinking. Until now, computer science topics were taught within the area of "information and communication technologies". As this is a new area with new objectives and needs, it is important to ask whether teachers are prepared to teach these new topics. Especially primary school teachers, as they have often never studied computer science and have not had the opportunity to acquire the appropriate knowledge and skills to teach computer science outside of voluntary training.

Some faculties of education are now preparing future primary school teachers for these changes. The aim of this paper is to analyse and describe the Computational thinking of students at the beginning of their studies and what effects block-based activities have on a level and development of Computational thinking of these students. With this goal in mind, we asked the following research questions:

RQ1) What is the level of Computational thinking of the future primary school teachers at the beginning of their professional training?

RQ2) What are the differences in computational thinking skills levels of the future primary school teachers at the beginning of their professional training?

RQ3) What is the effect of block-based activities on the future primary school teachers' computational thinking skills?

In addition to the research questions, three conceptual hypotheses were formulated:

H1: The overall level of computational thinking will be low.

H2: There are significant differences between the levels of measured skills.

H3: Block-based activities produce a significant positive effect on the future primary school teachers' computational thinking skills.

2. Background

2.1. Block-based Programming and Computational Thinking

One of the ways to help develop pupils' Computational thinking is to use programming problem-solving activities (Ramos & Espadeiro, 2014). There are many existing learning tools, such as Scratch, Makecode, Minecraft, Baltík, Kodu, Blockly. Piedade et al. (2019) picked and described 26 of these block-based programming environments and analysed how fit they are for teaching programming, design algorithms and developing Computational thinking.

We decided to use Scratch as our main tool for assessment of computational thinking skills of pre-service primary school teachers. Scratch is a free block-based programming environment, which was developed by the Scratch Foundation (Scratch, 2023). Several studies were conducted to analyse if Scratch is a good starting point for learning for students with little to no knowledge in programming (e.g. Sáez-López et al., 2016; Sigayret et al. 2022) with positive outcome. Blocks in Scratch are differentiated by colour and shape, which helps pupils to quickly find blocks they need (Weintrop & Wilensky, 2015). There are over 100 blocks divided into 9 categories, but there is also an option to add more blocks via various extensions. For example, Scratch can work with a programmable minicomputer BBC microbit. Each block represents a statement or some programming concept (Montiel & Gomez-Zermeño, 2021), which means pupils don't have to learn and memorize vocabulary specific for a given programming language. This greatly reduces syntax errors tied to wrongly written functions or sequences (Weintrop & Wilensky, 2017). Each block has a clear description of what it does. Scratch allows students to focus on finding a solution to given problems without worrying about the right syntax. But this can also lead to some pupils searching through all available blocks to see if one of them can solve the problem rather than having to think about it (Vaníček, 2019).

Another reason why we picked Scratch was because it is a popular choice among computer science teachers in the Czech Republic and most programming textbooks for primary schools are written for Scratch (Kalaš & Miková, 2020). Therefore, pre-service teachers should get familiar with Scratch during their studies, because they will most likely use it during their teaching.

2.2. Assessment of Computational Thinking Skills

Since computational thinking is a relatively new concept not only in the Czech Republic, but also in the world, there are not many tools measuring the level of computational thinking. Since the interest in Computational Thinking development is growing in recent years, several studies were conducted that focused on defining methods and tools for the evaluation and analysis Computer Thinking skills (Basso et al., 2018; Brennan & Resnick, 2012; Fagerlund et al., 2020; Román-González, 2015; Zapata-Cáceres et al., 2020).

Román-González et al. (2019) defined 7 possible approaches and tools for measuring computational thinking: diagnostic tools, summative tools, formative-interactive tools with automatic feedback, data mining tools, skill transfer tools, perception and attitude scales, and verbal assessments. Poulakis and Politis (2021) defined three ways in which the level of computational thinking can be assessed. These categories are 1) Using a particular programming environment, 2) Using psychometric tools, and 3) Using a combination of the two previous approaches.

Brennan and Resnick (2012) developed a framework for studying and measuring the development of Computational Thinking for Scratch-based tasks. This framework is organized in three dimensions, each of them focusing on different aspects of teaching programming. These dimensions are: Computational Concepts, Computational Practices and Computational Perspectives. Computational Concepts describe the use of programming concepts such as loops, sequences, events, operators and conditionals. Computational Practices are related to practices students develop during solving programming tasks, such as debugging their errors or using their already created code to build something new. Computational Perspectives describes students' perspectives of the computational world around them.

Another popular tool for analysing Scratch projects is Dr. Scratch (Moreno-León & Robles, 2015). It's a web-based application for automatic analysis of Scratch projects. It can identify coding errors and provide suggestions for improvements. It classifies Computational thinking concepts on a three-point scale. Because of this, it has difficulties analysing more complex projects.

In recent years, several tests and psychometric tools have been developed to measure the level of computational thinking. The Computational Thinking Test (Ambrosio et al., 2014) is a test that uses multiple choice questions to analyse the respondent's programming abilities. Román-González (2015) further developed and validated this test on students aged 10-15 years.

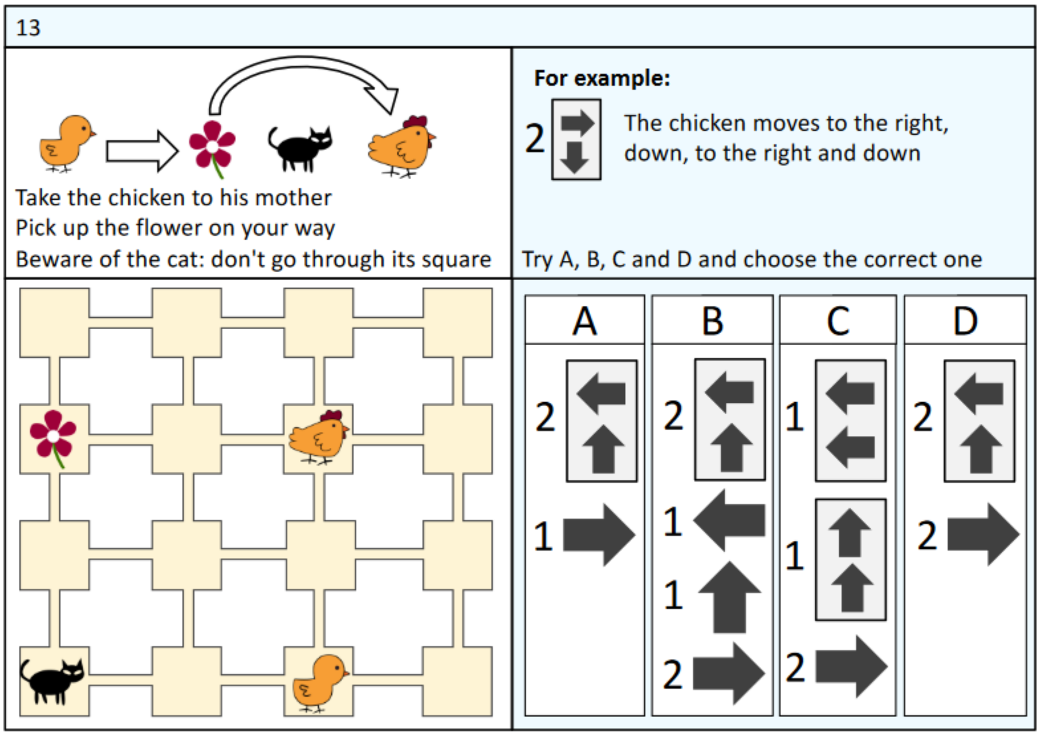
For our purposes, we chose to use the Beginners Computational Thinking Test (BCTt) (Zapata-Cáceres et al., 2020). The test contains 25 questions, which are divided into 6 categories that are closely related to programming and computational thinking - Sequences, Loops, Nested Loops, If-Then, If-Then-Else, and While. Table 1 shows the absolute and relative representations of the questions of each category in the test. As you can see, most of the questions in the test focused on Loops and Nested Loops.

Table 1
The distribution of BCTt test questions by category

Category	*N*	*%*
P1	4	16
P2	4	16
P3	7	28
P4	2	8
P5	2	8
P6	6	24

This test does not assume any prior knowledge of computer science or programming and is not dependent on any particular programming environment. The tasks are presented either in the form of a maze, in which respondents must help a chick to get to a chicken, or in the form of a canvas on which they must draw a given picture. There are always 4 possible answers, but only one of them is correct. It should be noted that the BCTt does not assess competencies, but rather allows us to measure the students' knowledge and skills regarding concepts associated with Computational thinking (Piedade & Dorotea, 2023). We'll refer to them as computational thinking skills in this paper. Figure 1 shows an example of a BCTt task.

Figure 1
An exmaple of a BCTt task (Zapata-Cáceres et al., 2020)



This test was primarily used to measure the level of computational thinking skills in 4th grade students in elementary school. But since our lectures were done using tasks originally designed for 4th and 5th grade of elementary school, we found it most appropriate to use this test to also measure the level of computational thinking skills of pre-service primary school teachers. The test was translated into Czech language.

4. Methodology

4.1. Research Design

Due to the nature of the research and the impossibility of dividing the students into control and experimental groups, we chose a quasi-experimental pre-test post-test research design. The students' level of Computational thinking was first measured at the beginning of their studies, before they took any of the mentioned classes. Then, students took a programming class, which spanned four months, from February to May. We then again measured their Computational thinking levels and analysed the results.

4.2. Participants

Our research sample was students of the study primary school education programme at the Faculty of Education of the University of South Bohemia in České Budějovice. They were first year students who had not yet encountered informatics during their studies. In the first three years of their studies, students in this field have compulsory informatics lessons - programming in the 1st year, robotics in the 2nd year and didactics of informatics and practice in the 3rd year. A total of 53 students participated in the research, 2 of them male and the rest female.

4.3. Data Analysis

Our quantitative data were analysed using The R Project for Statistical Computing and Microsoft Excel. Several statistical tests were applied during the analysis. We checked the reliability of our data using Pearson correlation coefficient and Cronbach's Alpha. To answer our first research question, we calculated descriptive statistics of our pre-test data. To answer the second research question, we used a non-parametric Kruskal-Wallis test. To answer our third research question, we used a parametric test paired t-test. We also used the Shapiro-Wilk test to check if our data follow the normal distribution. Before analysing our data and answering our research questions, we first analysed the reliability of our data.

4.4. Reliability

To test the reliability of our data, we calculated Pearson correlation coefficient. The results indicate high correlation between our pre-test and post-test scores (ρ=0.67). We also calculated Cronbach's Alpha coefficient to measure internal consistency of the BCTt scores. The analysis of the Cronbach's Alpha coefficients of the test from the pre-test and post-test scores also revealed very good internal consistency (Table 2). These results are similar to the reliability of the original BCTt calculated by the authors (α=.82; N=299).

Table 2
Internal consistency of pre-test and post-test samples

Sample	*Cronbach's Alpha coefficient*
Pre-test	0.703
Post-test	0.789

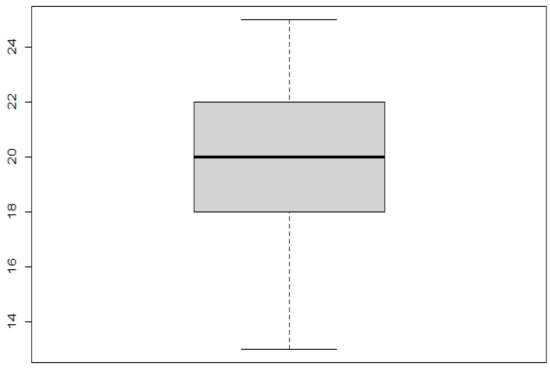
Note. N=53 for both tests.

5. Results

5.1. The Level of Computational Thinking of the Future Primary School Teachers

To answer our first research question, we analysed the results of the pre-test. Table 3 shows a statistical analysis of the results. The overall average is high (M=20.17, Me=20). Three students achieved the maximum number of points. The minimum number of points was 13. Only one student scored this low.

Table 3
The Descriptive Statistics for the pre-test results of the pre-service primary school teachers

Descriptive Statistics	*Statistics*	*Boxplot of mean score*
Mean	20.17	
Median	20	
Variance	6.99	
Standard Deviation	2.64	
Minimum	13	
Maximum	25	

5.2. The Differences in Computational Thinking Skills Levels of the Future Primary School Teachers

Next, we focused on analysing the results by category. We then performed a Kruskal-Wallis test to see if these differences are statistically significant. The results indicate that there is a significant difference between categories (p = .00). Table 4 shows the statistical analysis for each category. Students were most successful in the first 3 categories - Sequences, Loops and Nested Loops. They performed best on questions in the Loops category. Students were more successful in solving these types of problems compared to the If-Then and If-Then-Else categories. Students performed the worst in the While category. These results, together with the results of Kruskal-Wallis test, confirm our hypothesis H2 that there are significant differences between the levels of measured skills.

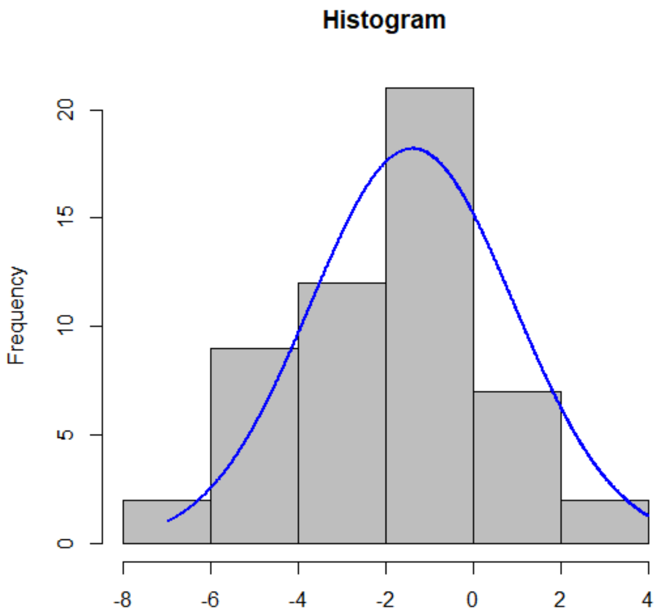
Table 4
The demographics of the recruited study participants

Category	*Number of questions*	*Mean*	*Median*
Sequences	4	3.91	4
Loops	4	3.96	4
Nested Loops	7	6.85	7
If-Then	2	1.01	1
If-Then-Else	2	1.49	2
While	6	2.86	2

5.3. The effect of Block-based Activities on the Future Primary School Teachers' Computational Thinking Skills

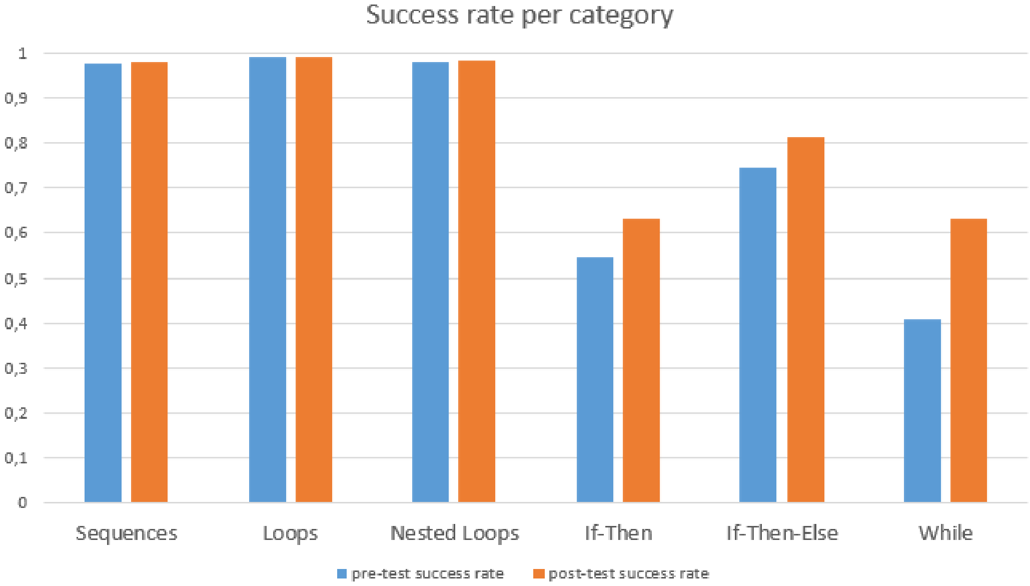
The analysis of the effect of block-based activities on the computational thinking skills of future primary school teachers started with calculating the differences between pre-test and post-test scores. We analysed the normality of data using the Shapiro-Wilk test. The results indicate that the BCTt scores differences follow a normal distribution ($W = 0.96$, $p = .070$). Figure 2 shows the histogram of our data with the respective normality curve.

Figure 2
Histogram of the scores differences



We then used paired t-test to analyse the statistical significance of these differences. The results ($p = .040$) revealed that there is a significant difference between pre-test and post-test results. After that we compared the percentage of correct answers of each category during pre-test and post-test. The Figure 4 shows that there is a visible difference between pre-test and post-test percentage of correct answers. The success rate for sequences, loops and nested loops stayed almost the same, since they were near 100% from the beginning, but there is a huge difference in success rate of the following categories. The biggest change is in the while category, in which the success rate got up 22%. These results confirm our hypothesis H3 about the positive effect of block-based programming activities on the pre-service primary school teachers' computational thinking skills.

Figure 4
Success rate per each category



6. Discussion and Conclusion

The study sought to describe the impact of block-based programming activities on the computational thinking skills of primary school pre-service teachers at the beginning of their studies. In addition, we examined their computational thinking at the start of their studies, before any formal computer science teaching had occurred.

The analysis of the results of the pre-test showed that students have a good understanding of sequences and loops, but have a poor understanding of conditionals. Their understanding of the concepts of sequences and loops were almost perfect, which resulted in the high scores from the test. The biggest problem was caused by the tasks involving the If-Then statement, especially by the tasks with the right answer having a condition that wasn't met. For example, if the chicken was standing on heart and the answer at that moment said "if the chicken is standing on the star, then", then some students became confused and couldn't answer properly. Another problem was involving a while statement. Some students couldn't decide when the while statement ends and that lead to the wrong answers.

The analysis of the effect of block-based programming activities was also positive. The differences in the pre-test and post-test results are statistically significant. The usage of block-based programming activities had a positive effect on the pre-service teachers' results. Activities involving problem-solving or creation of projects like simple games or animations have an evident effect on the level of the computational thinking skills not only promote the development of computational thinking skills but also contribute to the application of knowledge from other curricular areas (Piedade & Dorotea, 2023).

Thus, we can argue that regular project development, problem-solving, and game creation using block-based programming languages are crucial for student interest and success. These exercises support the application of information from other subject areas in addition to fostering the growth of computational thinking abilities. Primary school programming could encourage initiatives that imitate or gamify other subject areas (Fagerlund et al., 2020). According to Brennan and Resnick (2012), Scratch has shown to be a great resource for creating projects that support computational principles, especially for younger pupils.

The test selected for this study (Zapata-Cáceres et al., 2020), despite being developed and tested on 4th grade students, proved to be adequate for pre-service teachers, because their understanding of computational thinking skills is similar. The results regarding the reliability of the test proved to be similar to the results of the original authors' validation of the test.

Future research should assess pre-service primary school teachers' practices and perspectives on computational thinking in addition to examining their basic understanding of computational ideas. An enhanced comprehension of the abilities and expertise obtained by aspiring primary school teachers will be possible through the evaluation of these three dimensions as suggested by Brennan and Resnick (2012) in their framework. In order to achieve this, it will be crucial to organize challenges and problems that let students investigate computational techniques like testing and debugging, reusing and remixing, abstracting, and expressing their computational viewpoints on how computing affects daily life. In primary school classrooms, teachers and educators are increasingly in need of evidence-based pedagogical expertise to assist kids' CT learning through programming.

Declaration of interest: No conflict of interest is declared by author.

Funding: No funding was received for undertaking this study.

References

Ambrosio, A. P., Almeida, L. da S., Franco, A., & Macedo, J. (2014). *Exploring core cognitive skills of computational thinking*. The Psychology of Programming Interest Group-University of Liverpool.

Basso, D., Fronza, I., Colombi, A., & Pahl, C. (2018). Improving assessment of computational thinking through a comprehensive framework. In M. Joy & P. Ihantola (Eds.), *ACM Proceedings of the 18th Koli Calling International Conference on Computing Education Research* (pp. 1-5). Association for Computing Machinery. https://doi.org/10.1145/3279720.3279735

Bocconi, S., Chioccariello, A., Kampylis, P., Dagienė, V., Wastiau, P., Engelhardt, K., Earp, J., Horvath, M.A., Jasutė, E., Malagoli, C., Masiulionytė-Dagienė, V. & Stupurienė, G. (2022). *Reviewing Computational thinking in compulsory education*. Publications Office of the European Union. https://doi.org/10.2760/126955

Brennan, K., & Resnick, M. (2012). *New frameworks for studying and assessing the development of computational thinking*. Annual Meeting of the American Educational Research Association.

Fagerlund J, Häkkinen P, Vesisenaho M, & Viiri J. (2020). Computational thinking in programming with Scratch in primary schools: A systematic review. *Computer Applications in Engineering Education, 29*(1), 12-28. https://doi.org/10.1002/cae.22255

Kalaš, I., & Miková, K. (2020). *Základy programování ve Scratch pro 5. ročník zš* [Basics of programming in Scratch for the 5th year of elementary school]. Jihočeská univerzita v Českých Budějovicích, České Budějovice.

Lessner, D. (2014). Analysis of the meaning of the term "Computational thinking”. *Journal of Technology and Information Education*, *6*(1), 71-88. https://doi.org/10.5507/jtie.2014.006

Ministry of Education, Youth and Sports of the Czech Republic & NPI Czech Republic. (2021). *Nová informatika v RVP ZV* [New informatics in RVP ZV]. Author.

Montiel, H., & Gomez-Zermeño, M. (2021). Educational challenges for computational thinking in K–12 education: A systematic literature review of "Scratch" as an innovative programming tool. *Computers, 10*, 69. https://doi.org/10.3390/computers10060069

Moreno-León, J., & Robles, G. (2015). Dr. Scratch: A web tool to automatically evaluate Scratch projects. *ACM International Conference Proceeding Series, 15*, 132–133. https://doi.org/10.1145/2818314.2818338

Piedade, J., & Dorotea, N. (2023). Effects of Scratch-based activities on 4th-grade students’ computational thinking skills. *Informatics in Education*, *22*(3), 19. https://doi.org/10.15388/infedu.2023.19

Piedade, J., Dorotea, N., Ferrentini, F.S., & Pedro, A. (2019). A cross-analysis of block-based and visual programming apps with computer science student-teachers. *Education Sciences*, *9*(3), 181. https://doi.org/10.3390/educsci9030181

Poulakis, E., & Politis, P. (2021). Computational thinking assessment: literature review. In T. Tsiatsos, S. Demetriadis, A. Mikropoulos, & V. Dagdilelis (Eds.), *Research on E-Learning and ICT in Education* (pp. 111-128). Springer. https://doi.org/10.1007/978-3-030-64363-8_7

Ramos, J. L., & Espadeiro, R. G. (2014). Os futuros professores e os professores do futuro. Os desafios da introdução ao pensamento computacional na escola, no currículo e na aprendizagem [Future teachers and teachers of the future. The challenges of introducing computational thinking in school, in the curriculum and in learning]. *Educação, Formação e Tecnologias, 7*(2), 4–25. http://hdl.handle.net/10174/14227

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., & Kafai, Y. (2009). Scratch: programming for all. *Communications of the ACM, 52,* 60-67. https://doi.org/10.1145/1592761.1592779

Román González, M. (2015, July). *Computational thinking test: Design guidelines and content validation* [Paper presentation]. 7th International Conference on Education and New Learning Technologies, IATED Academy, Barcelona, Spain.

Román González, M., Moreno Léon, J., & Robles, G. (2019). Combining assessement tools for a comprehensive evaluation of computational thinking interventions. In S. C. Kong, & H. Abelson (Eds.), *Computational thinking education* (pp. 79-98). Springer. https://doi.org/10.1007/978-981-13-6528-7_6

Sáez-López, J.-M., Román-González, M., Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using "Scratch" in five schools. *Computers & Education*, *97*, 129–141. https://doi.org/10.1016/j.compedu.2016.03.003

Scratch. (n.d.). *About Scratch*. Author. https://scratch.mit.edu/

Sigayret, K., Tricot, A., & Blanc, N. (2022). Unplugged or plugged-in programming learning: A comparative experimental study. *Computers & Education, 184*, 104505. https://doi.org/10.1016/j.compedu.2022.104505

Stephens, M., & Buteau, C. (2023). Introduction to the special issue on "Computational thinking and mathematics teaching and learning". *Journal of Pedagogical Research, 7*(2), 1-4. https://doi.org/10.33902/JPR.202313362

Vaníček, J. (2019). Early programming education based on concept building. *Constructivist Foundation, 14*(3), 360–372.

Weintrop, D., & Wilensky, U. (2015). Using commutative assessments to compare conceptual understanding in blocks-based and text-based programs. In B. Dorn & J. Sheard (Eds.), *ICER '15: Proceedings of the eleventh annual International Conference on International Computing Education Research* (pp.101-110). Association for Computing Machinery. https://doi.org/10.1145/2787622.2787721

Weintrop, D., & Wilensky, U. (2017). How block-based languages support novices: a framework for categorizing block-based affordances. *Journal of Visual Languages and Sentient Systems*, *3,* 92-100. https://doi.org/10.18293/VLSS2017-006

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49,* 33-35. https://doi.org/10.1145/1118178.1118215

Xia, B. S. (2017). A pedagogical review of programming education research: what have we learned. *International Journal of Online Pedagogy and Course Design, 7*(1), 33–42. https://doi.org/10.4018/IJOP-CD.2017010103

Zapata Cáceres, M., Martín Barroso, E., & Román González, M. (2020). Computational thinking test for beginners: Design and content validation. In A. Cardoso, G. R. Alves, & M. T. Restivo (Eds.), *Proceedings of the IEEE Global Engineering Education Conference (EDUCON 2020)* (pp. 1905-1914). Educon. https://doi.org/10.1109/EDUCON45650.2020.9125368