Improving Computational Thinking in Nursing Students through Learning Computer Programming

Leticia Laura-Ochoa, Norka Bedregal-Alpaca, Elizabeth Vidal
Universidad Nacional de San Agustin de Arequipa
Arequipa, Peru

***Abstract*—Computational thinking is a fundamental skill for problem-solving, it uses computational concepts and other types of thinking such as algorithmic. The experience of improving computational thinking in nursing students using block-based programming environments such as Code.org, Lightbot, and the Python textual programming language is described. The results obtained are analyzed by applying a pre-and post-test of computational thinking to the students. The methodological design is quasi-experimental since it did not work with a control group. The study group was made up of 30 students from the Professional School of Nursing of the National University of San Agustin de Arequipa. The results show that teaching programming allows the understanding of computational concepts and improves computational thinking. It is concluded that block-based programming environments and the Python language facilitate the development of algorithmic thinking and computational thinking.**

Keywords*—Computational thinking; computational thinking assessment; computational thinking test; programming; programming environments

I. Introduction

Computational Thinking (CT) is a fundamental skill that influences different disciplines and professions, not only those related to science and engineering [1][2]. It is considered a transversal competence that goes beyond the use of computers and coding [3] since it includes algorithmic and parallel thinking, which involve different types of thought processes, such as compositional reasoning, pattern matching, procedural thinking, and recursive thinking. [2], which are required by new generations of students in different areas of knowledge.

According to Román-González et al. [4], programming is the main demonstration of the ability of the CT through the use of the computer, because it allows the development of algorithmic thinking, problem-solving, logic, and debugging skills [5]. In this context, textual programming is considered the final educational goal at the end of the K-12 level in most countries [4], because students in adolescence value the higher cognitive load of textual languages. However, text-based programming can make learning difficult for beginning students, overwhelming their cognitive ability [6]. Furthermore, it is considered a difficult task due to the lack of complete development of computational thinking in students [7]. Therefore, it is necessary to select the appropriate didactic tools and strategies that facilitate the teaching of programming with an approach oriented to the development of computational thinking. For [8], block-based programming environments are a good way to introduce beginning students to programming. In addition, in [9] they consider that these environments facilitate the understanding of programming concepts; but both block-based and text-based programming environments allow the development of skills related to computational thinking.

In addition, Tikva and Tambouris [10] have found that teachers face challenges in incorporating TC practices, so more capacity building frameworks and interventions that support teachers for successful integration of TC into their teaching practices are required. They believe that the relationship between tools and TC development should be explored as future work to provide information on which tools support which TC learning strategies.

Consequently, the aim of this work is to show the programming tools that were used in the experience and how they favor the acquisition of computational concepts and the development of computational thinking practices and skills, which can be of help and reference for teachers who need to incorporate computational thinking in their teaching work. This experience was carried out in an online learning environment with higher education students from the professional school of nursing, a career different from science and engineering, where the majority of students are usually women.

In this work, the experience of the use of programming environments based on blocks and text is described, to improve the development of computational thinking skills for beginning students in programming a Basic Computer course, which includes as a learning unit the Introduction to programming. An analysis of the pre- and post-test results of computational thinking applied to students is carried out to verify the improvement of computational thinking. In the experience, Code.org and Lightbot block programming environments were used prior to the Python textual programming language to facilitate the understanding of computational concepts and the acquisition of computational thinking practices.

II. Related Work

In the work of Brackmann et. al. [11], a quasi-experiment was presented in two primary schools in Spain, to develop the computational thinking skills of students through disconnected activities, students' ages are between 10 and 12 years old. Their results show that the students of the experimental group, who participated in the disconnected activities, significantly increased their computational thinking skills, compared to those who did not participate in the disconnected activities, evidencing the effectiveness of the disconnected approach for the development of computational thinking skills. However,

they consider that this approach has limitations, and there may come a point where it loses its effectiveness and the use of computing devices is required to further develop these skills.

Vasquez and Luján [12] carried out an evaluation of aptitude level on the development of computational thinking in students of the basic level of secondary school. They identified the need to strengthen skills related to computational thinking such as analysis, algorithm design, and data abstraction, because their average scores did not exceed 50% of the total number of questions evaluated. Likewise, they found that the maturity of the students and the cognitive development according to the academic degree do not establish the level of development of computational thinking skills. Since their cultural environment must also be considered. The authors considered that the results of the computational thinking test can be used to design and develop a computational thinking course that allows strengthening skills that require it. Montes-León et al. [7] describe their experience of the application of computational thinking activities that positively influences the improvement of learning in a course of fundamentals of programming. They carried out an analysis of the results of a pre and post test of computational thinking applied to secondary students divided into control and experimental groups, to evaluate the improvement of computational thinking. The ages of the participating students were between 15 and 16 years old. They also analyzed the results of the evaluation applied in the course. The activities that were carried out in the experimental group were some exercises from the international Bebras contest, mathematical problems, exercises from a university entrance test and mental games.

Laura-Ochoa and Bedregal-Alpaca [13] found that the incorporation of computational thinking practices allowed students to improve their performance on the first Python programming course, where they used support tools such as PSeInt, CodingBat and the turtle graphic library for the development of skills related to computational thinking, conducted an analysis of the grades obtained by the students of the control and experimental group in the midterms and final average of the programming course, where the students of the experimental group showed an improvement in their learning results. As future work, they suggest the application of computational thinking measurement assessments by following a practice-oriented programming teaching approach to computational thinking.

III. Methodology

The methodological design used was quasi-experimental, since it did not work with a control group. In the study group, there were 30 students enrolled in the Basic Computer course (groups B and F) of the academic semester 2020-B of the Professional School of Nursing of the National University of San Agustin de Arequipa (Peru).

The Basic Computer Science course at the Professional School of Nursing of the National University of San Agustin de Arequipa - Peru, is given in the second academic semester. It is developed for 17 weeks. It has 4 practice hours a week, lasting 50 minutes each, equivalent to 2 credits.

The students of the experiment were 30 women (100%), who participated in the pre- and post-test of computational thinking, as well as in the development of the third learning unit: Introduction to Programming. This unit is developed during five weeks, with two weekly sessions of 2 hours each.

In the practice hours, the students experimented with the use of visual programming environments: Code.org, Lightbot and Python textual programming language. The method used in the class sessions was expository-participatory.

The computational thinking test developed by Román-González et al. [14] was used to measure the level of development of computational thinking in students. Its test is consistent with other computational thinking tests aimed at middle/high school students [15], it is mainly aimed at Spanish students between 12 and 14 years old (7th and 8th grade of primary school), but it can be used in lower and higher grades. It is aligned with some CT computational concepts defined by Brennan and Resnick [16] and partially aligned with some computational practices.

The students who participated in the experience completed the computational thinking test [14] in the first week of classes on the subject and at the end of the third learning unit: Introduction to programming, accessing an online test through a Google Apps form.

To measure the improvement of computational thinking, a comparison of the scores obtained by the students in the pre-test and post-test of computational thinking was made, to check if the activities carried out in the third learning unit: Introduction to Programming allowed the acquisition of computational concepts and development of skills related to computational thinking.

IV. Description of the Experience

Block-based visual programming environments (Code.org, Lightbot) and Python programming language were selected for teaching the introduction to programming and development of skills related to computational thinking in the third learning unit of the Basic Computer course taught to students of the nursing professional career in the period 2020-B.

Table I shows the programming topics that were developed using the selected programming environments, where the students learned computational thinking concepts considered in the work of Luo, Antonenko and Davis [17], such as sequential instructions, conditionals, and loops.

Students began learning the block-based visual programming language using Code.org by completing exercises in the "hour of code" tutorials that enabled them to understand concepts of sequential statements, loops, and functions. An example of block programming on Code.org is shown in Fig. 1, where students understood the utility of iterative statements by replacing repetitive blocks of code (left) with loops (right) to draw the geometric figure of the square, in which the students acquired some computational thinking practices such as iteration and abstraction through the identification of repetitive patterns, which allowed them to acquire the ability to reduce unnecessary details and propose new solutions.

TABLE I. PROGRAMMING TOPICS

Tools	*Sequential Instructions*	*Conditional Instructions*	*Repetitive Instructions*	*Functions*
Code.org	X		X	X
Lightbot	X		X	X
Python	X	X	X	

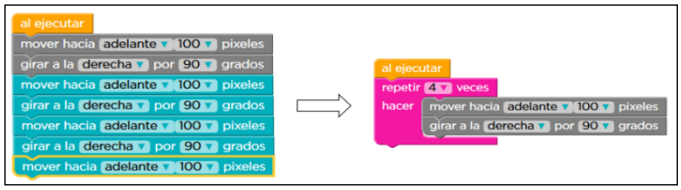

Fig. 1. Replacing Repetitive Blocks of Code with Loops at Code.org.

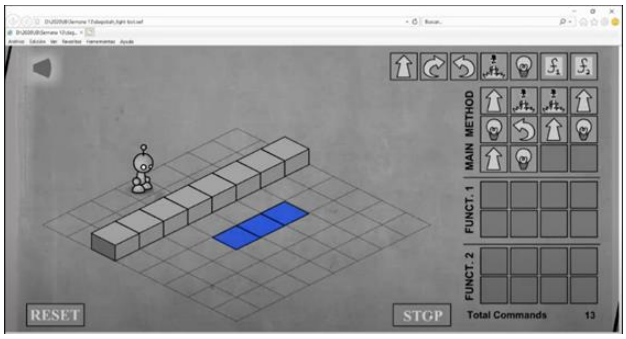

Fig. 2. Sequential Instructions with Repeating Patterns.

The Lightbot video game was used to reinforce algorithmic thinking skills in students by thinking in sequence of instructions for problem solving. Fig. 2 shows an example of the Lightbot third level solution using only sequential instructions (move, jump, turn left) in the MAIN METHOD for the robot to move and turn on all blue blocks, which is the objective of the video game, where repetitive patterns such as moving forward and turning on are identified.

In Fig. 3, a second solution for the third level of Lightbot is shown, where students, through generalization (identification of repetitive patterns go forward and turn on), abstraction (reduction of unnecessary details) and decomposition of the program using one of the functions (FUNCT. 1), acquire the ability to find better solutions to the problem and make use of code reuse.

In addition, the students practiced the Lightbot video game from Code.org, where they used recursion to create loops in the PROC1 procedure (Fig. 4).

After experience with the block-based programming environments, the students learned the syntax and semantics of the Python textual programming language using the Google Collaboratory environment for the creation and execution of code, with which they reinforced their understanding of computational concepts, such as sequential statements, conditionals, and loops. Fig. 5 shows some examples of codification of single and double selection structures; but there was also done double selection instructions (nested).

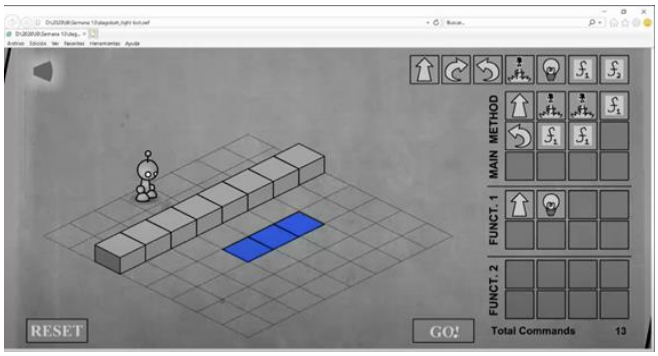

Fig. 3. Program Decomposition and Code Reuse with Functions.

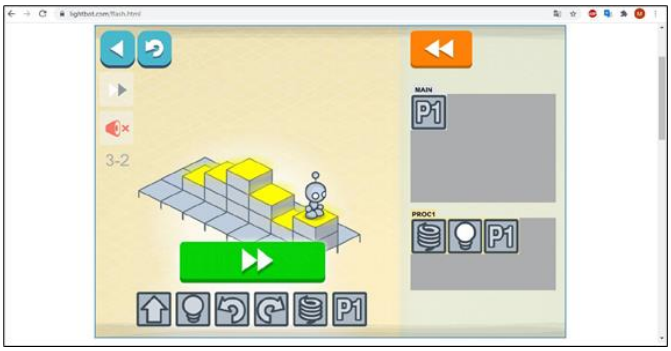

Fig. 4. Looping using Recursive Calls in Lightbot from Code.org.

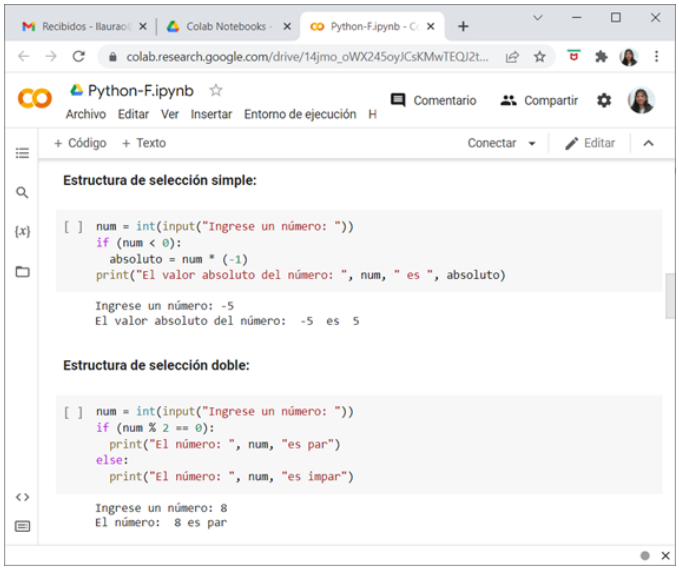

Fig. 5. Coding with Single and Double Select Structures using Python.

With the programming environments used in the experience, computational thinking skills are used, such as automation and debugging for the execution of programs and logical analysis for the verification of the results.

Therefore, students acquired computational thinking practices such as abstraction, decomposition, iteration, and debugging.

V. Result and Discussion

The computational thinking test [14] was applied at the beginning of the Basic Computer Science course before taking Unit 3 of Introduction to programming, serving as a pre-test for the evaluation of said unit.

Fig. 6 shows the percentage of correct answers per question in the pre-test. The regression line shows the progressive difficulty of the test. The average correct percentage of the 28 questions was 64.64%.

At the end of the third learning unit, the computational thinking test was applied again [14].

Fig. 7 shows the percentage of correct answers per question in the post-test. The regression line shows an improvement in terms of the progressive difficulty of the pre-test. The average correct percentage of the 28 questions was 80.6%.

Table II shows some descriptive statistical data related to the total scores obtained by the students in the pre and post-test. The total scores are evaluated from 0 to 28.

Fig. 8 and 9 show histograms with the distribution of said total scores, where the improvement in the total, mean, median and mode scores of the post-test are evidenced. In the post-test, the median and mode have the same value of 23.

Fig. 10 shows box plots for the scores obtained through computational thinking pre- and post-test. In the post-test, an atypical value is observed that corresponds to a student who obtained a total score of 15 points. The median, maximum value, minimum value, quartiles are higher in the post-test.

Table III shows the averages of the percentages of correct answers of the questions for the computational concepts that are addressed in the computational thinking test, obtained by the students in the pre and post-test.

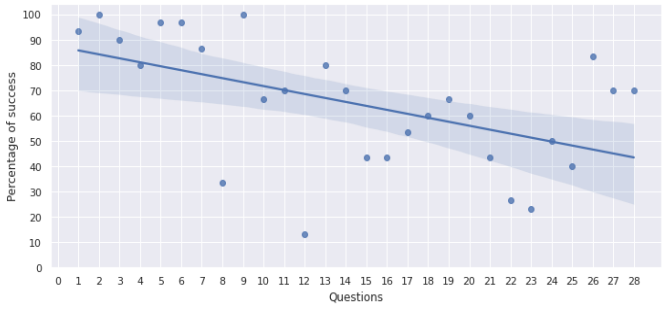

Fig. 6. Success Percentage per Question in the Pre-test.

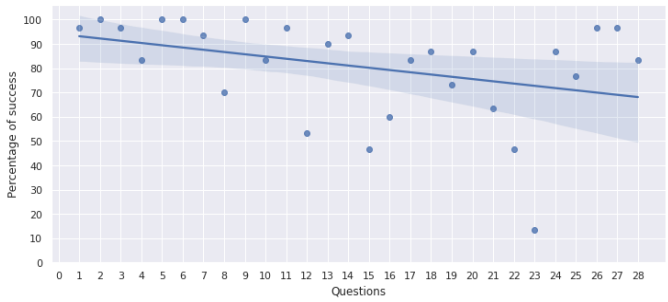

Fig. 7. Success Percentage per Question in the Post-test.

TABLE II. Descriptive Statistical Data of the Total Scores

	Pre-Test	Post-Test
Minimum	11	15
Maximum	26	27
Mean	18.1	22.567
Median	17.5	23.0
Mode	16	23
Standard deviation	3.791	2.885

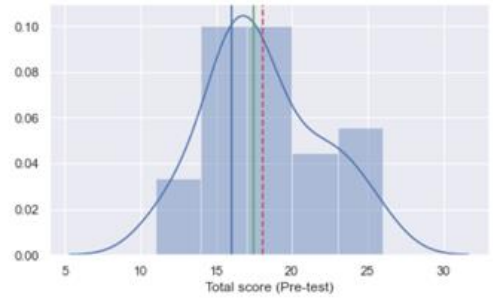

Fig. 8. Histogram with the Distribution of Total Scores (Pre-test).

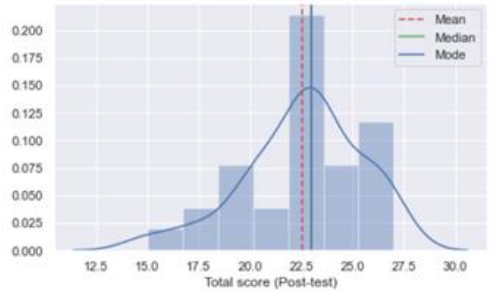

Fig. 9. Histogram with the Distribution of Total Scores (Post-test).

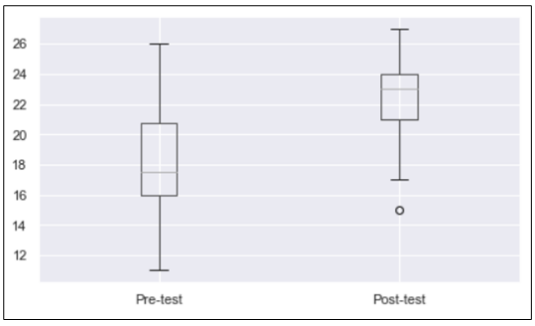

Fig. 10. Box Plots of the Scores Obtained in the Pre and Post Test.

TABLE III. Success Percentages for Computational Concepts

	Pre-Test (%)	Post-Test (%)
Basic directions and sequences	91	94
Loops 'Repeat Times'	78	91
Loops 'Repeat Until'	63	83
Simple Conditional 'if'	59	73
Complex conditional 'if/else'	60	83
While conditional	36	53
Simple functions	66	88

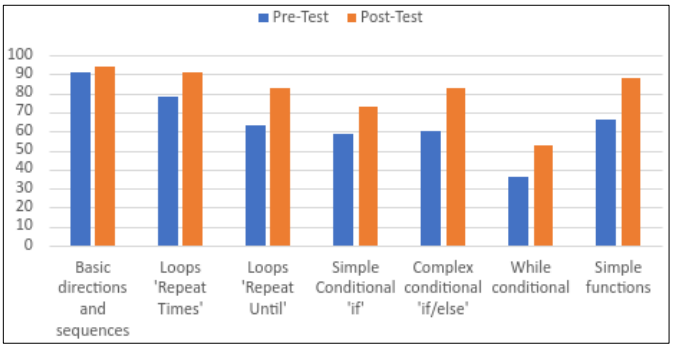

Fig. 11. Success Percentage for Computational Concepts.

Fig. 11 shows that the percentages of success in the post test are higher for each of the computational concepts addressed in the test.

Therefore, it can be affirmed that the performance of the students has improved in the post-test, evidencing an improvement in computational thinking, after the development of the third learning unit, in which the introduction to programming was made using visual programming environments based on Code.org blocks, Lightbot and the Python textual programming language.

According to the experience described, the programming environments used allowed the acquisition of computational practices such as abstraction, decomposition, iteration and debugging, which correspond to the computational thinking practices defined by Brennan and Resnick [16] and adapted in the work of Luo, Antonenko and Davis [17]. In addition, students developed skills such as algorithmic thinking, generalization, automation, and debugging, which are part of the computational thinking skills identified in five featured articles in the work of [18]. It was achieved that students had a means to acquire methodological tools, deepen knowledge, and cultivate skills [19].

We agree with a previous work [20] that learning text-based programming is important for the development and professional performance of students, because in addition to reinforcing concepts and practices of computational thinking, they can expand their learning for the creation of applications, analysis or visualization of data.

In programming courses aimed at beginners, we consider it important to carry out activities related to computational thinking at the beginning of the course to improve their learning, where visual programming environments based on blocks can be used. Likewise, in [8] they consider block-based programming environments a good way to introduce beginning students to programming.

Likewise, we agree with Zapata-Cáceres et al. [21], in that computational thinking is not limited only to the activities of computer scientists; it is applied in daily life and in different areas of knowledge, so it is a necessary skill to adapt to the future.

VI. Conclusion

This article has presented the experience of developing computational thinking skills through a block and text-based programming activities to improve students' computational thinking. We examined the total scores obtained by the students in the pre-test and post-test, there is evidence of an improvement in the scores in the post-test with the programming environments used in the learning unit of introductory programming, which indicates the effectiveness of programming for understanding computational concepts addressed in the test. We concluded that the visual programming environments based on blocks in combination with the textual programming language using Python allow the student to acquire computational concepts and computational thinking practices such as abstraction, decomposition, iteration, and debugging in introductory programming courses directed mainly to beginning students of non-computer related careers.

Acknowledgment

The authors' thanks are expressed to the National University of San Agustin de Arequipa for the support received in the realization of the proposal and the results are expected to benefit the institution.

References

[1] J. M. Wing, “Computational thinking,” Communications of the ACM, vol. 49, no. 3, pp. 33-35, 2006.

[2] J. M. Wing, “Computational thinking: What and why. The Link,” News from the School of Computer Science at Carnegie Mellon University, 2011.

[3] J. Acevedo-Borrega, J. Valverde-Berrocoso and M. D. C. Garrido-Arroyo, “Computational Thinking and Educational Technology: A Scoping Review of the Literature,” Education Sciences, vol. 12, no. 1, pp. 39, 2022.

[4] M. Román-González, J. C. Pérez-González, J. Moreno-León and G. Robles, “Can computational talent be detected? Predictive validity of the Computational Thinking Test,” International Journal of Child-Computer Interaction, vol. 18, pp. 47-58, 2018.

[5] F. Buitrago Flórez, R. Casallas, M. Hernández, A. Reyes, S. Restrepo and G. Danies, “Changing a generation’s way of thinking: Teaching computational thinking through programming,” Review of Educational Research, vol. 87, no. 4, pp. 834-860, 2017.

[6] C. Chen, P. Haduong, K. Brennan, G. Sonnert and P. Sadler, “The effects of first programming language on college students’ computing attitude and achievement: a comparison of graphical and textual languages,” Computer Science Education, vol. 29, no. 1, pp. 23-48, 2019.

[7] H. Montes-León, R. Hijón-Neira, D. Pérez-Marín and S. R. Montes-León, “Mejora del Pensamiento Computacional en Estudiantes de Secundaria con Tareas Unplugged,” Education in the knowledge society (EKS), no. 21, pp. 24, 2020.

[8] Z. Xu, A. D. Ritzhaupt, F. Tian and K. Umapathy, “Block-based versus text-based programming environments on novice student learning outcomes: a meta-analysis study,” Computer Science Education, vol. 29, no. 2-3, pp. 177-204, 2019.

[9] L. Laura-Ochoa and N. Bedregal-Alpaca, “Análisis de entornos de programación para el desarrollo de habilidades del pensamiento computacional y enseñanza de programación a principiantes,” Revista Ibérica de Sistemas e Tecnologias de Informaçao, no. E43, pp. 533-548, 2021.

[10] C. Tikva and E. Tambouris, “Mapping computational thinking through programming in K-12 education: A conceptual model based on a systematic literature Review,” Computers & Education, vol. 162, pp. 104083, 2021.

[11] C. P. Brackmann, M. Román-González, G. Robles, J. Moreno-León, A. Casali and D. Barone, “Development of computational thinking skills through unplugged activities in primary school,” in Proceedings of the 12th workshop on primary and secondary computing education, pp. 65-72, 2017.

[12] A. J. O. Vasquez and B. I. S. Luján, "Evaluación del nivel de aptitud desarrollo de pensamiento computacional en jóvenes de nivel básico secundaria," RECIE. Revista Electrónica Científica de Investigación Educativa, vol. 4, no. 2, pp. 1151-1163, 2019.

[13] L. Laura-Ochoa and N. Bedregal-Alpaca, "Incorporation of Computational Thinking Practices to Enhance Learning in a Programming Course," International Journal of Advanced Computer Science and Applications(IJACSA), vol. 13, no. 2, 2022, DOI: 10.14569/IJACSA.2022.0130224.

[14] M. Román-González, J. C. Pérez-González and C. Jiménez-Fernández, "Test de Pensamiento Computacional: diseño y psicometría general," in III congreso internacional sobre aprendizaje, innovación y competitividad (CINAIC 2015), pp. 1-6, 2015.

[15] M. Román-González, J. C. Pérez-González and C. Jiménez-Fernández, "Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test," Computers in human behavior, vol. 72, pp. 678-691, 2017.

[16] K. Brennan and M. Resnick, "New frameworks for studying and assessing the development of computational thinking," in Proceedings of the 2012 annual meeting of the American Educational Research Association, 2012.

[17] F. Luo, P. D. Antonenko and E. C. Davis, "Exploring the evolution of two girls' conceptions and practices in computational thinking in science," Computers & Education, vol. 146, pp. 103759, 2020.

[18] S. Bocconi, A. Chioccariello, G. Dettori, A. Ferrari and K. Engelhardt, "Developing computational thinking in compulsory education - Implications for policy and practice," in JRC Science for Policy Report, 2016.

[19] N. Bedregal-Alpaca, "Virtual tutoring and blended-learning in the postgraduate course: Orientations and results of an experience," Proceedings of the 17 LACCEI international Multi-conference for Engineering, Education and Technology, 2019, DOI 10.18687/LACCEI2019.1.1.220.

[20] L. Laura-Ochoa and N. Bedregal-Alpaca, "Development of Computational Thinking Skills: An Experience With Undergraduate Students," in 2021 XVI Latin American Conference on Learning Technologies (LACLO), IEEE, pp. 112-117, 2021, DOI 10.1109/LACLO54177.2021.00070.

[21] M. Zapata-Cáceres, E. Martín-Barroso and M. Román-González, "Computational thinking test for beginners: Design and content validation," in 2020 IEEE Global Engineering Education Conference (EDUCON), IEEE, pp. 1905-1914, 2020.