Application of A Teaching Plan for Algorithm Subjects Using Active Methodologies: An Experimental Report

https://doi.org/10.3991/ijet.v17i07.28733

Fabrício Wickey da Silva Garcia(✉), Sandro Ronaldo Bezerra Oliveira, Elielton da Costa Carvalho
Graduate Program in Computer Science (PPGCC), Federal University of Pará (UFPA), Pará, Brazil
fabriciogarcia@ufpa.br

Abstract—Computer programming subjects have high failure rates, their contents are of great importance for the formation of undergraduate students, however their learning is considered challenging. In this sense, the use of approaches that promote active learning on the part of students in algorithmic subjects stand out in the specialized literature, in order to increase their engagement and awaken their interest in the contents covered, making them protagonists of their learning. However, the proposals do not always fully disclose the instruments used, such as the detailing of the teaching plan, which is extremely important material for the good conduct of a subject, since in it the professor is able to articulate his strategies of intervention, the tools used, as well as the learning objectives that must be achieved. Thus, the objective of this work is to discuss and analyze the results of an experiment that consisted in the application of a teaching plan that makes use of multiple active methodologies (Virtual Learning Environments, Coding Dojo, Gamification, Problem Based Learning, Flipped Classroom and Serious Games) in an algorithms subject. We sought to evaluate the learning gains of the proposed approach compared to teaching using the traditional teaching method. A statistical analysis was performed using the Student-t two-tailed approach for independent samples, which showed significant statistical gains when using the proposed approach.

Keywords—active methodologies, teaching plan, algorithms

1 Introduction

Computer programming represents one of the main pillars in the training of students in computing. For [1], programming was one of the areas that became the center of attention of researchers around the world in the last decade. This may reflect the changes brought about by the use of technology, which is present in the daily life of a large part of the population, and increasingly requires that people have skills to use technological resources.

For [2], teaching programming is an extremely important way to develop skills related to the use of technological resources, as well as the development and maturation

of computational thinking. This may allow students to improve skills related to creativity, innovation, autonomy and problem solving.

In Brazil, the National Curriculum Guidelines (DCN) for computing guide that bachelor's degree courses in computing must provide training that includes a solid base of contents that allow the development of skills related to logical reasoning and computer programming [3].

Subjects in computing have high failure rates and their learning is considered complex. In this sense, [4] reinforce that there is a great effort by professors of courses that offer such subjects to raise the approval rates, this is due to the great importance of computer programming in the academic and professional training of students in computing.

Also according to [4], introductory programming courses, such as algorithms, are extremely important, as they represent the foundation of computer programming and it is at this point that basic skills and competences must be stimulated and developed properly.

A study by [5] identified that students who fail introductory programming courses have a higher dropout rate than those who pass. In this sense, it is of great importance to provide ways to overcome the initial programming difficulties, as these efforts can help improve the approval and dropout rates of courses in computing [6].

Based on the difficulties faced, many types of intervention have been tested in classrooms to minimize the barriers to learning in the algorithms subjects. It is clear that new types of intervention need to be investigated to improve student performance. In this sense, several studies recommend the use of active methodologies, which are student-centered approaches, enabling them to increase engagement and make them protagonists of their learning [7], [8], [9].

The use of active methodologies, even with promising results in the literature, requires caution and adequate planning, therefore, it is in the teaching plan that the professor is able to analyze the characteristics of his target audience, define the educational objectives, select and structure the contents that will be studied, define the didactic-pedagogical resources that will be used, as well as define the form of intervention that will be used and its evaluation procedures [10].

This work aims to present the results of the application of a teaching plan for the subject of algorithms for undergraduate students of higher education courses in Computing. The teaching plan was elaborated from good practices extracted from references from the specialized literature and has a teaching strategy that is supported by the use of the following active methodologies Virtual Learning Environments, Coding Dojo, Gamification, Problem Based Learning, Flipped Classroom and Games, which are applied in the teaching units in a gradual way according to the conduction of the subject.

In addition to this introductory section, this article is structured as follows: Section 2 presents an approach to the teaching of algorithms and curriculum references in computing, Section 3 presents the research methodology, detailing its main steps, Section 4 presents some related works to the use of active methodologies in teaching algorithms, Section 5 addresses the strategy used in evaluating the teaching plan, which makes use of active methodologies from an experiment, Section 6 presents the

details of the analysis of the results obtained, Section 7 presents discussions regarding the results obtained in the research, Section 8 contains the details of the threats to validity that have been identified, as well as their mitigation methods, and, finally, Section 9 presents the conclusions, limitations of this study and future work.

2 Teaching algorithms

The teaching algorithms or equivalent subjects should establish a basis for the development of skills and competences related to computer programming, which are strongly related to the correct use of logical and mathematical reasoning with a focus on solving computational problems [11].

Several studies in the specialized literature show that, traditionally, the teaching algorithms is done mechanically, based on the repetition of concepts, students act mainly as passive agents in the learning process, and their learning is usually focused on memorization and fixation of the contents covered in the course [12], [13].

Learning algorithms is considered challenging, there are several difficulties faced by students, such as interpretation of computational problems, low level of mathematical knowledge, lack of understanding of key concepts and lack of motivation [14], [15].

For [16], programming students have difficulties in assimilating content related to abstraction, which can lead to a lack of understanding of a certain key concept, generating difficulties to solve programming problems, even if students know the syntax and semantics of the language being worked on.

When starting to study a programming language, the student is faced with a set of rules, instructions and even mathematical formulas and they need to unify all these new concepts with the skills related to problem abstraction. This becomes a challenge for learning schedule. The authors of [14] reinforce that such difficulties make the development of skills and competences related to computer programming challenging for beginner programmers.

The difficulties faced can be reflected in the high failure rates of subjects in computer programming that, for [12] and [17], usually reach failure averages that are around 40 to 50%. The algorithms subject, as one of the bases for the development of skills related to programming, is usually offered in the initial semesters of courses in computing, usually in the 1st or 2nd semester of courses that have it, and its workload usually varies from 60 to 80 class hours [18], [19] and [20].

For [21], the teaching of algorithms should not follow the molds of traditional teaching, as the programming area is evolving day by day, the contents (teaching units) are constantly changing as well as the tools that are used by programmers. According to [22], with the constant technological evolution the demand for innovative teaching increases, which provides more effective learning experiences for students.

In this sense, curriculum references such as the Brazilian Computer Society (SBC), Association for Computing Machinery (ACM) and Institute of Electrical and Electronic Engineers (IEEE) seek to periodically update practices that can be adopted in

the classroom, as well as the teaching units that must be worked in the subjects of algorithms.

2.1 Reference curriculum (SBC, ACM/IEEE)

The Brazilian Computer Society (SBC) is one of the guiding institutions of computer education in Brazil, actively working in the development of References for Training (RF) in Computing, which seek to assist undergraduate course coordinators in the preparation of projects pedagogical aspects of their courses [23].

The RF are built around the notion of competences, that is, they focus on the development of skills related to "knowing how to do", so that the courses train students who know how to apply the knowledge acquired during their graduation in real situations [24].

In addition, the RF are prepared in accordance with the National Curriculum Guidelines (DCN) for computing, which establish legal standards that guide the structuring of courses in computing in Brazil, as well as defining the profile of students, as well as skills and competences that they must possess.

According to the DCN approved in 2016, computing courses must have a solid and adequate base of contents that allow the development of skills and competences related to logical reasoning and computer programming [3].

In this sense, [24] present some skills that are present in RF and that can be developed with the algorithms subject:

- Identify problems that have an algorithmic solution,
- Solve problems using programming environments,
- Recognize the importance of computational thinking in everyday life and its application in appropriate circumstances and in different domains,
- Recognize the importance of computational thinking in everyday life and its application in appropriate circumstances and in different domains,
- Formulate and solve problems with the application of logical, mathematical and computational reasoning,
- Understand problems and formulate solutions that can be performed by the computer.

Likewise, good practices present in the Computing Curricula 2020 (CC2020) of the Association for Computing Machinery (ACM) and the Institute of Electrical and Electronic Engineers (IEEE) [25] were adopted, extracting the recommendations of the basic knowledge units from the curriculum to key programming fundamentals concepts covering the following topics:

- Create an appropriate algorithm to illustrate iterative and recursive functions, as well as divide and conquer techniques, and use a programming language to implement, test, and debug the algorithm to solve a simple problem,
- Decompose a program for a client that identifies the data components and behaviors of many abstract data types and implement a coherent abstract data type with loose coupling between components and behaviors,

- Design, implement, test, and debug an industry program that uses fundamental programming constructs, including simple I/O and file, standard conditional and iterative structures, function definition, and parameter passing,
- Present the costs and benefits of static and dynamic data structure implementations, choosing the appropriate data structure to model a given problem,
- Apply consistent documentation and programming style standards for a software engineering company that contributes to the readability and maintainability of the software by conducting a personal and small-team code review on the program component using a provided checklist,
- Demonstrate common errors in coding, building and debugging programs using the standard libraries available with a chosen programming language,
- Refactor an industry program, identifying opportunities to apply procedural abstraction.

In addition, the current version of the ACM/IEEE curriculum (CC2020) has adopted the cognitive process levels derived from Bloom's taxonomy to indicate the degree of skill expected in successfully carrying out tasks related to teaching units. According to [26], Bloom's taxonomy has the following cognitive structure:

- Remember: is related to the recognition and reproduction of ideas and content,
- Understand: is related to establishing a link between the new and previously acquired knowledge,
- Apply: is related to the execution or use of a procedure in a given situation, whether new or not,
- Analyze: it is related to the division of information into parts considered relevant and not relevant, in addition to understanding the relationships that exist between the parts,
- Evaluate: is related to making judgments based on qualitative and quantitative criteria and standards or, in some cases, criteria of efficiency and effectiveness standards,
- Create: is related to the insertion of elements with the objective of creating a new vision, making use of previously acquired knowledge and skills.

3 Research methodology

In order to achieve the objectives of this research, a sequence of steps was established and followed, which made it possible to understand, from scientific studies, the current scenario of teaching algorithms, its difficulties and types of active interventions that are being used in classroom. This made it possible to: (i) elaborate a teaching plan based on the main reference curriculum in computing and which makes use of active methodologies, (ii) technically evaluate the proposed material through peer review, and (iii) carry out experiments to observe the effects of using the teaching plan on algorithmic classes. The steps adopted are detailed in Figure 1 and will be presented in following subsections.

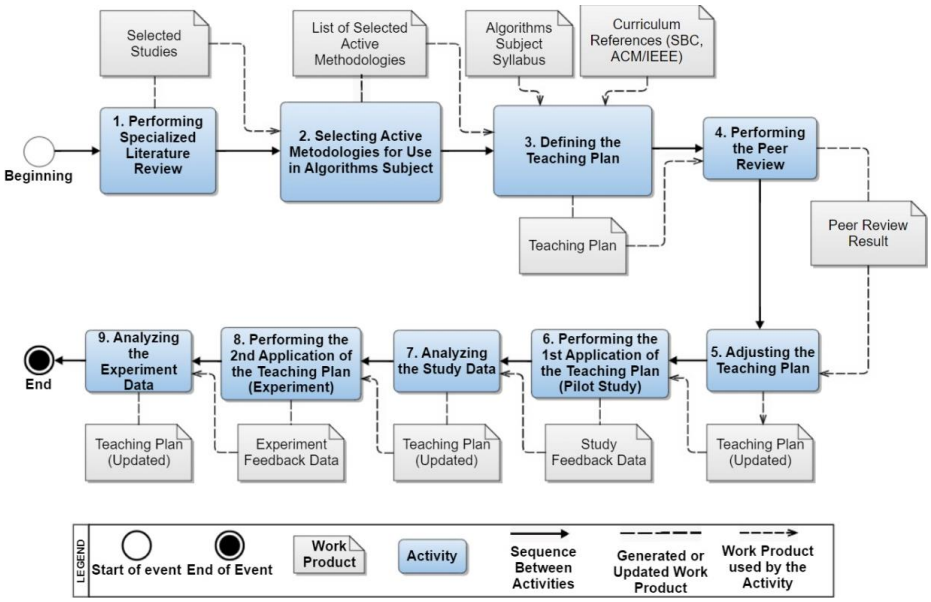

Fig. 1. Study execution steps

3.1 Step I: Literature review

A review of the specialized literature was carried out in order to identify which types of intervention are currently being used in the teaching algorithms that seek to promote more active and autonomous learning in the classroom. The search covered a time period from January 1, 2016 to December 31, 2019, where a total of 1014 scientific studies were analyzed and from these works, 15 active approaches used in the teaching algorithms were identified. The details of the review can be found in [27].

The review adopted good practices from a Systematic Literature Review (RSL), however no comparisons were made between the returned publications, so this type of review is characterized as a quasi-Systematic Literature Review (RqSL) [28].

3.2 Step II: Analysis of active methodologies

With the analysis of the active methodologies identified in [27], the positive and negative points and types of use within subjects were raised. In this way, it was possible to select and establish a strategy for using active methodologies that could be used together within the same subject. Figure 2 presents the selected methodologies and their purposes for use in the algorithms subject.

The way in which the active methodologies were used was evaluated by experts through the peer review process, which allowed for the identification of areas for improvement and the validation of the strategy. Details of the process of analysis and selection of methodologies can be found in [29].

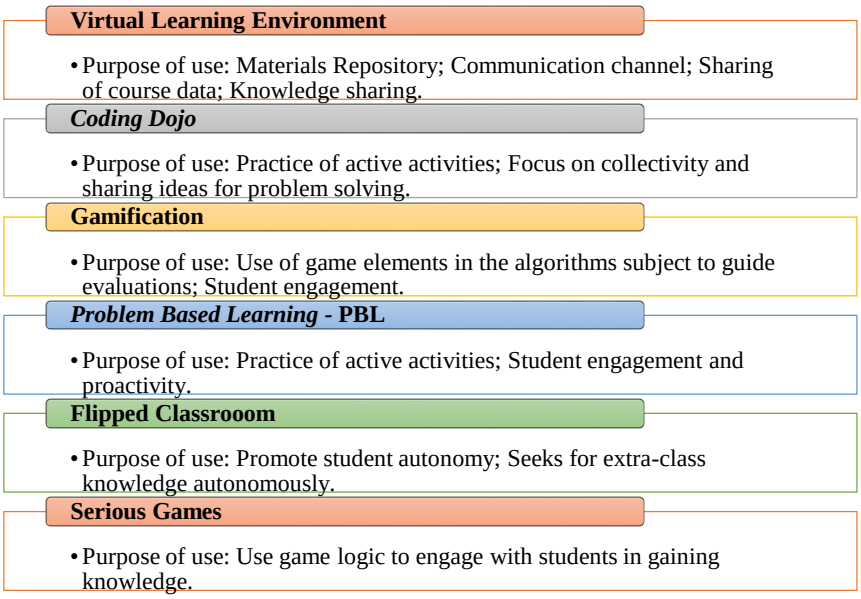

Fig. 2. Selected methodologies and their purposes

3.3 Step III: Definition of the teaching plan

A teaching plan was prepared based on good practices present in the reference curriculum of the Brazilian Computer Society (SBC), Association for Computing Machinery (ACM) and Institute of Electrical and Electronic Engineers (IEEE). The teaching plan has a total workload of 68 class hours and is divided into modules, allowing the teaching units to be worked from a set of active methodologies.

The teaching units of the algorithms subject were related to the active methodologies and were gradually distributed throughout the teaching modules, as shown in Table 1.

The Initial Evaluation Module (AVI) lasts two hours of class and aims to collect feedback related to prior knowledge that students have and that can be matured during the course. The Teaching Modules I, II and III last 8, 24 and 22 class hours, respectively, and seek to work the contents of the algorithms subject through expository and dialogued classes with the support of interventions based on active methodologies. The Final Evaluation Module (AVF) lasts two class hours and seeks to collect feedback related to the teaching process used, in order to identify positive and negative points and opportunities for improvement for the evolution of the proposed approach.

The teaching plan was revised using the peer review technique, in which 3 professors who have been teaching algorithms at the university level, with experience in creating teaching plans and in the use of active methodologies, participated in the review. The evaluation allowed us to identify points of improvement and validate the technical quality of the material. The details of the teaching plan, as well as its review process by experts, can be consulted in [30].

Table 1. Teaching units related to active methodologies

Modules	Teaching Units	Active Methodologies
Initial Evaluation Module	Student Feedback	Not applicable
Teaching Module I	• What is an algorithm? • What is a programming language? • Types to represent an algorithm, • Variables, • Operators (Logical and Arithmetic), • Linearization of expressions.	• Gamification, • Problem Based Learning, • Coding Dojo • Virtual Learning Environment.
Teaching Module II	• Introduction to the Scratch Environment, • Scratch variables, operators and main commands, • Conditional Structures, • Repetition Structures, • Lists (Arrays).	• Problem Based Learning, • Coding Dojo, • Gamification, • Serious Games, • Virtual Learning Environment.
Teaching Module III	• Introduction to C Programming Language, • Variables, operators and main commands of the C programming language, • Conditional structures in C programming language, • Repetition structures in C programming language, • Arrays in C programming language.	• Problem Based Learning, • Coding Dojo, • Gamification, • Flipped Classroom, • Virtual Learning Environment.
Final Evaluation Module	Student Feedback	Not applicable

The contents covered in the algorithms subject were detailed, their expected results were identified and each topic was related to the learning objectives of Bloom's taxonomy, which is also one of the new practices incorporated in the ACM CC2020 curriculum [25].

In this sense, the details of each topic, with the expected results and their respective correlations with the elements from Bloom's taxonomy can be found in Appendix A of this document. The strategy for using each level of cognition in the taxonomy was based on the recommendations for use present in CC2020 [25].

3.4 Step IV: Application of the teaching plan

The application of the teaching plan was divided into two steps: (i) a first application (pilot study) to verify the applicability of the teaching strategy, identify points of improvement and the effects of using active methodologies in a class of algorithms, but without making comparisons with classes that use the traditional teaching approach, (ii) then, a second application was performed using a larger number of participants who were divided into two groups (control and experimental), seeking to evaluate the effectiveness of the approach proposed in the experimental group compared to the traditional approach used in the control group.

With the pilot study, we sought to evaluate the applicability of the teaching plan in an algorithms subject, as well as the effects of its use. The study lasted 68 class hours,

distributed over 17 meetings lasting approximately 4 class hours each, and it was carried out in the first semester of 2021. The target audience was undergraduate students from courses in computing who had not yet attended the course of algorithms. The participation of all those involved in the pilot study was voluntary.

The effectiveness of active methodologies was evaluated based on criteria that were defined, which were classified according to the following indicators presented in Table 2.

Table 2. Summary of the case study effectiveness indicators, adapted from [31]

0 - 50% effectiveness – Critical: This indicator represents Inefficiency and Ineffectiveness in Teaching and Learning using the Active Methodology.
51% - 70% effectiveness – Alert: This indicator represents a possible Efficiency and Effectiveness in Teaching and Learning using the Active Methodology, but with still doubtful gains.
71% - 100% Effectiveness – Satisfactory: This indicator represents Efficiency and Effectiveness in Teaching and Learning using the Active Methodology.

The results observed in the pilot study indicated an effectiveness classified as satisfactory for all active methodologies used in the teaching plan, as they all obtained a score above 70% of effectiveness.

It is noteworthy that in this first application of the teaching plan, there was no comparison of the proposed approach with a control group that uses the traditional approach, as the focus of the pilot study was on the applicability of the teaching plan and its use effects. At the end of the study, some points of improvement were identified and the necessary adjustments were made. The full details of the pilot study can be found in [31].

With the completion of the pilot study, a second application, by means an experiment, was conducted in the second half of 2021, with the participation of 68 students, the strategy of [32], [33] and [34] was adopted, which performs the division of students in groups of the same size (Control and Experimental) with 34 students each, one of them being the control group, which conducted an algorithms subject through the traditional teaching approach, and the second group was the experimental one, which conducted the subject from the approach proposed as part of this work, which is based on the use of active methodologies. The study was carried out remotely, due to restrictions arising from the COVID-19 pandemic, the meetings were weekly and had an approximate duration of 4 class hours each.

The content covered in both groups was divided into 3 parts, which were titled Evaluation 1, 2 and 3 for the control group, and Teaching Module I, II and III for the experimental group, as shown in Figure 3.

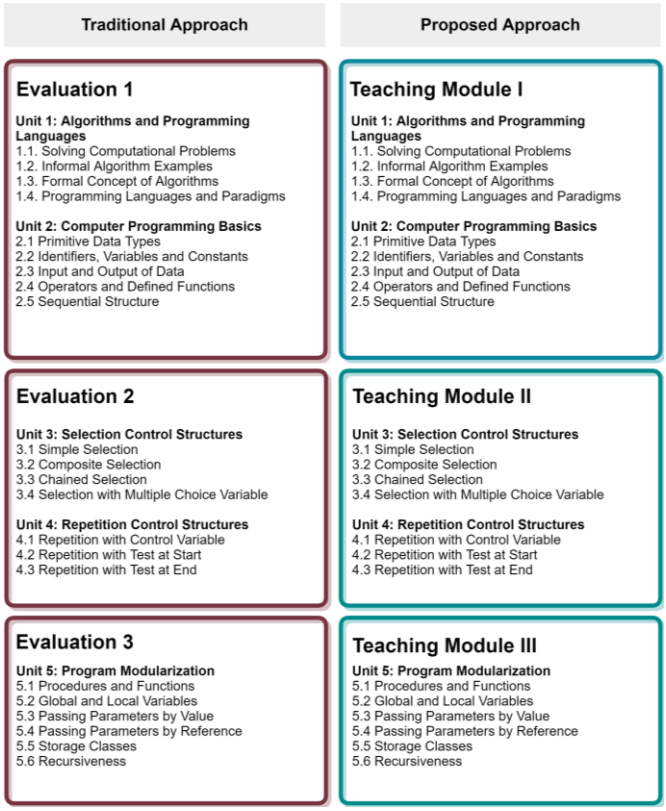

Fig. 3. Contents covered in approaches (Traditional vs. Proposed)

Although both groups approached the same algorithms teaching units, the way of conducting the classes was different in each group. In the control group, which was based on the traditional teaching method, in each part (Evaluation 1, 2 or 3) there were classic lectures, lists of exercises that addressed the content being studied and a test, as illustrated in Figure 4.

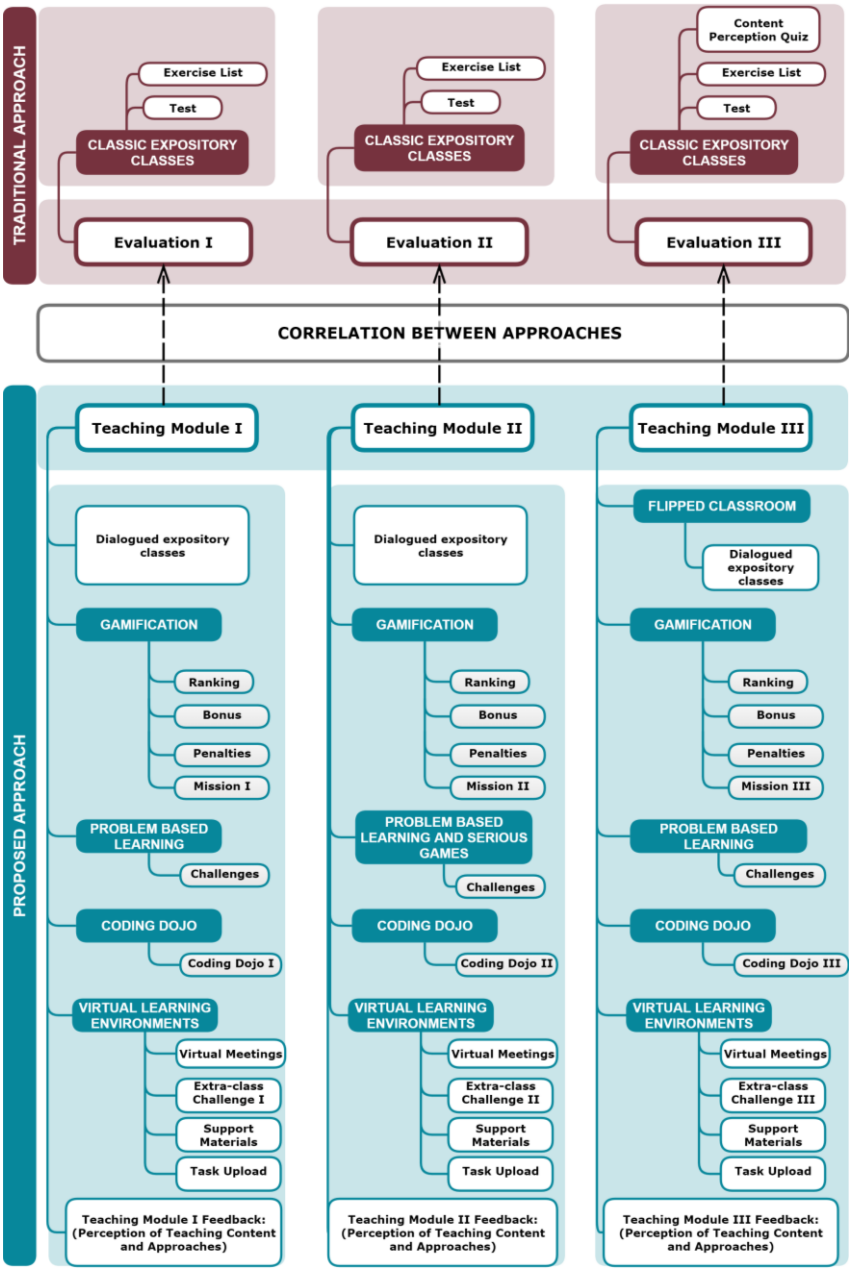

Fig. 4. Teaching structure used in the traditional approach versus the proposed approach

In the experimental group, which used the approach proposed in this research (as can be seen in Figure 4), all classes were expository and dialogued, which were accompanied by practical tasks that were conducted through active methodologies,

which could vary depending on the teaching module that was being worked on. At the end of each part (Teaching Module I, II and III) students provided feedback related to the content covered and the teaching approach used.

4 Related works

[35] present the results of a set of interventions that were carried out in algorithmic subjects at the Federal University of Santa Maria – UFSMA, in Brazil. The authors present an intervention strategy that was created based on the good practices of Problem Based Learning – PBL, applied in the algorithms subject. Preliminary results show an increase in the approval ratings of the classes that participated in the experiment. The authors intend to continue improving the proposal and seek to insert active learning methodologies in all subjects of the Bachelor's Degree in Information Systems course at UFSMA.

[36] carried out a study related to the effects provided by gamification in the teaching algorithms. The authors used a gamified strategy based on online judges to aid the teaching and learning process. A comparative study was carried out and covered 4 academic semesters, making comparisons between the use of the approach based on online judges versus traditional teaching. The results showed that the students who participated in the experimental group had a higher performance than the group that used the traditional approach, so that the pass rates were 81%, while the second group (traditional approach) had an average of approval of 42% of the participants.

The work by [37] proposes a type of intervention for the algorithms subject that applies active methodologies from tutorials. Approximately 16 tutoring meetings were held, where the contents were actively worked out. In addition, the monitoring of students was carried out individually, allowing the identification of the particularities of each student in a more effective way. The proposal by [37] also seeks to evaluate the effectiveness of the proposed approach from an exploratory study with programming students. In this study, we seek to identify, based on feedback from participants, the feasibility and relevance of the approach based on the students' perspective. Preliminary results showed a good acceptance of students with the tutoring and the authors seek in the future to carry out more experiments with the proposal and that the results obtained may alleviate the difficulties faced by students in the algorithms subject.

[38] present a proposal for the use of gamification in algorithmic subjects, in which game elements were created for the teaching units of the subject that focused on homogeneous data structures. The proposal sought to encourage students and increase their engagement in the subject. One study analyzed student participation during the course and the data showed a considerable gain in student grades, so that the concepts of most students in the course rose to the GOOD category, whereas in the traditional approach the average of the concepts is generally classified as REGULAR.

[39] presents an approach based on the use of Problem Based Learning and Flipped Classroom active methodologies applied to the teaching algorithms in order to promote the engagement and autonomy of students in the classroom and, consequently,

improve the performance of students. same in the subject. A comparative study was carried out during six semesters, where the effects of using the proposed approach versus the use of the traditional teaching approach were observed. The results showed that the group that used the proposed approach had a higher learning gain than the group that used the traditional teaching approach.

[40] presents an intervention proposal based on three active methodologies: Flipped Classroom, Collaborative Learning and Gamification. The work addresses the creation of a collaborative environment for teaching and learning programming. An experiment was carried out in which two groups of students were created, the first using the proposed approach in classes and the second using the traditional teaching approach. Preliminary results indicated that the group that used active methodologies obtained higher grades than the group that used the traditional teaching method.

The studies discussed in this section presents significant results, indicating learning gains from the use of active methodologies, but it can be noted that the vast majority is focused on the use of one or two active approaches. In addition, they do not present a support tool to guide the construction of a teaching plan for the algorithms subject, or present a detailed strategy for using each active approach with the contents of the subject, nor do they correlate the teaching units with the cognition levels of Bloom's taxonomy, which is a great strategy to identify and define the learning objectives of a subject, which makes this research present a differential in relation to the others.

5 Teaching plan evaluation strategy

Research questions and hypotheses were defined that allowed observing the degree of learning of the approaches used in the experimental group versus the control group based on Bloom's revised taxonomy [26]. In this sense, we used the score results obtained in the activities that were related to each teaching unit that was involved with the research question that would be analyzed.

Table 3 details the study objectives with their respective research questions and hypotheses, as well as the evaluation instruments used in the algorithm teaching units.

Table 3. Study objectives, research questions and hypotheses

Study objective 1
Research question 1 (RQ1): What is the effectiveness of learning in Teaching Module I when the proposed approach of using active methodologies for teaching Algorithms is adopted in relation to the traditional approach?
Hypothesis H01: In Teaching Module I, there will be no difference between the scores obtained by the Experimental and Control groups at the Create level.
Variables
DES1 – Challenges 1 DES2 – Challenges 2 DES3 – Challenges 3 D.EX1 – Extra class Challenge 1 COD 1 – Coding Dojo 1 M1 – Mission 1
Formulation: Ma > Mb, where:

1. a = Experimental Group 2. b = Control Group 3. Experimental Group scores: $Na_i = \frac{((DES1*0,25)+(DES2*0,25)+(DES3*0,25)+(D.EX1*1)+(COD1*0,25)+(M1*8))}{10}$, where i is a student of Group a 4. Control Group scores: Nb_i = M1, where i is a student of Group b 5. Average of the scores of students in Group a: $Ma_i = \frac{\Sigma_{i=1}^{n} * Nai}{n}$, where n is the number of students in Group a 6. Average of the scores of students in Group b: $Mb_i = \frac{\Sigma_{i=1}^{n} * Nbi}{n}$, where n is the number of students in Group b
Instruments: Challenges, Extra class Challenge, Coding Dojo and Mission.
Study objective 2
Research question 2 (RQ2): What is the effectiveness of learning in Teaching Module II when the proposed approach of using active methodologies for teaching Algorithms is adopted in relation to the traditional approach?
Hypothesis H02: In Teaching Module II, there will be no difference between the scores obtained by the Experimental and Control groups at the Create level.
Variables
DES4 – Challenges 4 DES5 – Challenges 5 DES6 – Challenges 6 DES7 – Challenges 7 D.EX2 – Extra class Challenge 2 COD2 – Coding Dojo 2 M2 – Mission 2
Formulation: Ma > Mb, where: 1. a = Experimental Group 2. b = Control Group 3. Experimental Group scores: Na_i = , where i is a student of Group a 4. Control Group scores: Nb_i = M1, where i is a student of Group b 5. Average of the scores of students in group a: $Ma_i = \frac{\Sigma_{i=1}^{n} * Nai}{n}$, where n is the number of students in Group a 6. Average of the scores of students in group b: $Mb_i = \frac{\Sigma_{i=1}^{n} * Nbi}{n}$, where n is the number of students in Group b
Instruments: Challenges, Extra class Challenge, Coding Dojo and Mission.
Study objective 3
Research question 3 (RQ3): What is the effectiveness of learning in Teaching Module III when the proposed approach of using active methodologies for teaching Algorithms is adopted in relation to the traditional approach?
Hypothesis H03: In Teaching Module III, there will be no difference between the scores obtained by the Experimental and Control groups at the Create level.
Variables
DES8 – Challenges 8 DES9 – Challenges 9 DES10 – Challenges 10 DES11 – Challenges 11 D.EX3 –Extra class Challenge 3 COD3 – Coding Dojo 3 M3 – Mission 3

Formulation: Ma > Mb, where: 1. a = Experimental Group 2. b = Control Group 3. Experimental Group scores: $Na_i = \frac{((DES8*0{,}25)+(DES9*0{,}25)+(DES10*0{,}25)+(DES11*0{,}25)+(D.EX3*1)+(COD3*0{,}25)+(M3*7{,}75))}{10}$, where i is a student of Group a 4. Control Group scores: Nb_i = M1, where i is a student of Group b 5. Average of the scores of students in group a: $Ma_i = \frac{\Sigma_{i=1}^{n} * Nai}{n}$, where n is the number of students in Group a 6. Average of the scores of students in group a: $Mb_i = \frac{\Sigma_{i=1}^{n} * Nbi}{n}$, where n is the number of students in Group b
Instruments: Challenges, Extra class Challenge, Coding Dojo and Mission.
Study objective 4
Research question 4 (RQ4): What is the learning efficiency of the proposed approach in the experimental group.
Hypothesis H04: There will be no difference between the results obtained in the pre-test and post-test activities carried out in the Experimental group.
Variables
Pre-Test Post-Test
Formulation: A qualitative analysis was performed based on the following criteria: • Finite Instruction Sequence, • Ordering of instructions, • Unambiguous, • Problem resolution.
Instruments: Pre-Test and Post-Test.

The experiment was carried out based on a schedule that guided the classes of the experimental and control groups (as can be seen in Table 4), which presents the details of the teaching units that were worked in each class, as well as the intervention strategies used in both groups.

Table 4. Experiment schedule

Days	Control Group	Experimental Group
Day 1 Inaugural Class	Presentation of the subject, how it is conducted and the teaching plan used. Availability of support material.	Presentation of the subject and the functioning of the teaching plan used [30]. Availability of support material.

Days 1 to 5	**Classic lectures:** About the topics of teaching units 1 and 2 present in Figure 3. **Test:** Evaluation activity with multiple choice and discursive questions about the content taught in teaching units 1 and 2.	**Dialogue lectures:** about teaching module I – Development of the Programming Logic: With the topics of Unit 1 and Unit 2 present in Figure 3, supported by active teaching methodologies: **Gamification:** Using game elements to guide the scoring of course tasks and activities. **Problem Based Learning:** Dynamics focused on the active learning of students, which guide the conduct of Challenges. **Virtual Environment:** Tool where class materials are made available, in addition to serving as a central place for students' virtual meetings to share knowledge and send out activities. **Coding Dojo 1:** Pseudocode construction challenge using the VisuAlg Tool. All students participate through alternating roles: pilot, copilot and audience. The grade given to the participants takes into account the amount of challenges performed and their correctness. **Extra class Challenge 1:** Problems that involve the subjects present in the content of all classes of the current teaching unit, these must be carried out outside the classroom. **Challenges:** Problems that involve the subjects present in the content of each class of the current teaching unit and that are carried out after the dialogued expository classes. **Mission 1:** Evaluation activity carried out at the end of teaching module I and which includes all the content covered in the current teaching unit.
Days 6 to 11	**Classic lectures:** About the topics of teaching units 3 and 4 present in Figure 3. **Test:** Evaluation activity with multiple choice and discursive questions about the content taught in teaching units 3 and 4.	**Dialogue lectures:** about teaching module II – Software Construction with Scratch: With the topics of Units 3 and 4 present in Figure 3, supported by active teaching methodologies: **Gamification:** Using game elements to guide the scoring of course tasks and activities. **Problem Based Learning:** Dynamics focused on the active learning of students, which guide the conduct of Challenges. **Serious Games:** Use of the theme of games with a focus on learning the content covered in the current module. The theme of serious games will be inserted in the activities of the current module: Challenges, Extra class Challenge 2, Coding Dojo 2 and Mission 2. **Virtual Environment:** Tool where class materials are made available, in addition to serving as a central place for students' virtual meetings to share knowledge and send out activities. **Coding Dojo 2:** Block programming activity aimed at creating simplified games that involve the contents covered in the current teaching module using Scratch. All students participate through alternating roles: pilot, copilot and audience. The grade given to the participants takes into account the amount of challenges performed and their correctness. **Extra class Challenge 2:** Problems with the theme of serious games that involve the subjects present in the content of all classes of the current teaching unit, these must be carried out outside the classroom. **Challenges:** Problems with the theme of serious games that involve the subjects present in the content of each class of the current teaching unit and that are carried out after the dialogued expository classes. **Mission 2:** Evaluation activity based on the theme of serious games, carried out at the end of teaching module II and which includes all the content covered in the current teaching unit.

Days 12 to 17	**Classic lectures:** About the topics of teaching unit 5 present in Figure 3. **Test:** Evaluation activity with multiple choice and discursive questions about the content taught in teaching unit 5.	**Dialogue lectures:** on teaching module III - Software Construction with programming language: With the topics of Unit 5 present in Figure 3, supported by active teaching methodologies: **Gamification:** Using game elements to guide the scoring of course tasks and activities. **Problem Based Learning:** Dynamics focused on the active learning of students, which guide the conduct of Challenges. **Flipped Classroom:** Dynamics focused on the autonomous learning of students, which will guide the acquisition of knowledge of the teaching units worked on in the current module. **Virtual Environment:** Tool where class materials are made available, in addition to serving as a central place for students' virtual meetings to share knowledge and send out activities. **Coding Dojo 3:** Code construction activity using the C programming language, involving the contents covered in the current teaching module. All students participate through alternating roles: pilot, copilot and audience. The grade given to the participants takes into account the amount of challenges performed and their correctness. **Extra class Challenge 3:** Problems that involve the subjects present in the content of all classes of the current teaching unit, these must be carried out outside the classroom. **Challenges:** Problems that involve the subjects present in the content of each class of the current teaching unit and that are carried out after the dialogued expository classes. **Mission 3:** Evaluation activity carried out at the end of teaching module III and which includes all the content covered in the current teaching unit.
Day 17 Feedback	Content Perception Questionnaire.	Content Perception Questionnaire. Questionnaire on teaching approaches

Each group (experimental and control) had a specific professor, who was responsible for conducting the subject from beginning to end. Both professors were supported by a team of evaluators consisting of 3 members with experience in teaching algorithms, who were responsible for correcting the activities carried out in both groups, so that, in this way, there was no interference in the evaluation grades by the professors who were carrying out the interventions in the classroom and, consequently, the bias in the results could be minimized.

6 Data analysis

In this section, the data obtained from the execution of the experiment presented in this work are presented. The analysis will be carried out from the research questions RQ1, RQ2, RQ3 and RQ4 defined in this work.

6.1 Analysis of research question 1

In RQ1, "What is the effectiveness of learning in Teaching Module I when the proposed approach of using active methodologies for teaching Algorithms is adopted in relation to the traditional approach?", evidence was sought to refute H01 "In Teaching Module I there will be no difference between the scores obtained by the Experimental

and Control groups at the Create level", by comparing the experimental and control groups in Teaching Module I versus Evaluation 1. The data normality, variance and the objective of evaluating the difference between two populations with treatment conditions and two samples (treatments), therefore, we chose the Student-t two-tailed test for independent samples.

The experimental group scored 7.79±1.25 in the evaluation, while the control group scored 6.44±1.72. Thus, there is a real difference between the groups, where the experimental score is Δ=1.35 higher than the control, indicating a possible increase in learning in this group.

To perform the Student-t test, data were submitted to normality assumption tests, which were verified using the Shapiro-Wilk test, with a result of P = 0.096. To verify the variance, the Equal Variance Test (Brown-Forsythe) was used, with a result of P = 0.159.

From the Student-t two-tailed test (α=0.05), there were significant differences between the analyzed groups (Experimental – Teaching Module I versus Control – Evaluation 1), thus the test for the two independent samples showed that there is an effect of using active methodologies in the experimental group, as can be seen in the result of the Student-t two-tailed test with P-value = 0.000451 < 0.05. Thus, the significance level derived from the test provided statistical evidence to reject H01. Table 5 summarizes the results obtained for RQ1.

Table 5. Comparison of learning effectiveness between participating groups (student-t) in teaching module I x evaluation 1

Variables	**Experimental Group**	**Control Group**
	Evaluation	***Evaluation***
Sample Size	34	34
Minimum	5,4	2
Maximum	9,6	9,3
Sum of Points	265,1	219,2
Median	7,64	6,75
First Quartile	6,85	5,50
Third Quartile	8,96	8,00
Average	7,79	6,44
Standard Deviation	1,256	1,726

6.2 Analysis of research question 2

In RQ2, "What is the effectiveness of learning in Teaching Module II when the proposed approach of using active methodologies for teaching Algorithms is adopted in relation to the traditional approach?", evidence was sought to refute H02 "In Teaching Module II, there will be no difference between the scores obtained by the Experimental and Control groups at the Create level". In the comparison between the experimental and control groups in Teaching Module II versus Evaluation 2, the standard of comparison performed in the analysis of RQ1 was followed.

The experimental group scored 7.81±1.25 in the evaluation, while the control group scored 6.35±2.00. Thus, there is a real difference between the groups, where the experimental score is Δ=1.46 higher than the control, indicating a possible increase in learning in this group.

To perform the Student-t test, data were submitted to normality assumption tests, which were verified using the Shapiro-Wilk test, with a result of P = 0.226. To verify the variance, the Equal Variance Test (Brown-Forsythe) was used, with a result of P = 0.116.

From the Student-t two-tailed test (α=0.05), there were significant differences between the analyzed groups (Experimental – Teaching Module II versus Control – Evaluation 2). Thus, the test for two independent samples showed that there is an effect of using active methodologies in the experimental group, as can be seen in the result of the Student-t two-tailed test with P-value = 0.000594 < 0.05. Thus, the significance level derived from the test provided statistical evidence to reject H02. Table 6 summarizes the results obtained for RQ2.

Table 6. Comparison of learning effectiveness between participating groups (student-t) in teaching module II x evaluation 2

Variables	Experimental Group	Control Group
	Evaluation	***Evaluation***
Sample Size	34	34
Minimum	6,200	1,200
Maximum	9,900	9,700
Sum of Points	265,680	216,020
Median	7,555	6,450
First Quartile	6,720	5,135
Third Quartile	9,057	7,625
Average	7,814	6,354
Standard Deviation	1,250	2,002

6.3 Analysis of research question 3

In RQ3, "What is the effectiveness of learning in Teaching Module III when the proposed approach of using active methodologies for teaching Algorithms is adopted in relation to the traditional approach?", evidence was sought to refute H03 "In Teaching Module III, there will be no difference between the scores obtained by the Experimental and Control groups at the Create level". In the comparison between the experimental and control groups in Teaching Module III versus Evaluation 3, the standard of comparison performed in the analysis of RQ1 was followed.

The experimental group scored 8.46±1.05 in the evaluation, while the control group scored 6.46±1.52. Thus, there is a real difference between the groups, where the experimental score is Δ=2.00 higher than the control, indicating a possible increase in learning for this group.

To perform the Student-t test, data were submitted to normality assumption tests, which were verified using the Shapiro-Wilk test, with a result of P = 0.448. To verify the variance, the Equal Variance Test (Brown-Forsythe) was used, with a result of P = 0.057.

From the Student-t two-tailed test (α=0.05), there were significant differences between the analyzed groups (Experimental – Teaching Module III versus Control – Evaluation 3). Thus, the test for two independent samples showed that there is an effect of using active methodologies in the experimental group, as can be seen in the result of the Student-t two-tailed test with P-value = 0.0000000295< 0.05. Thus, the significance level derived from the test provided statistical evidence to reject H03. Table 7 summarizes the results obtained for RQ3.

Table 7. Comparison of learning effectiveness between participating groups (student-t) in teaching module III x evaluation 3

Variables	**Experimental Group**	**Control Group**
	Evaluation	***Evaluation***
Sample Size	34	34
Minimum	6,350	4,000
Maximum	10,00	9,500
Sum of Points	287,650	219,800
Median	8,590	6,500
First Quartile	7,580	9,287
Third Quartile	5,400	7,500
Average	8,460	6,465
Standard Deviation	1,052	1,523

6.4 Analysis of research question 4

In RQ4, "What is the learning efficiency of the proposed approach in the experimental group?", a qualitative analysis was carried out at the beginning and at the end of the experiment, taking into account the following criteria: (i) Sequence of finite instructions, (ii) Ordering of instructions, (iii) Unambiguousness and (iv) Problem resolution.

Two logic tests were carried out, the first at the beginning (pre-test) and the second at the end of the experiment (post-test), where the tests consisted in the elaboration of a sequence of steps to solve a specific problem. With the completion of the tests, we sought to analyze in the solutions provided by the students whether they had a sequence of finite, ordered, unambiguous instructions and whether the students' answers were able to solve the proposed problems.

Pre-test data indicated that students, in general, sought to solve the problem using steps or steps, but without presenting a logical step-by-step of actions. It was noticed that the students had difficulty in developing a strategy to define a sequence of consistent steps that would enable them to adequately solve the problem. The answers generally had some incomplete steps, as well as not using commands characteristic of

pseudocode or programming language. It was noticed that the students had difficulties in visualizing/understanding what was sought to be solved and this may justify the initial difficulty in mapping the actions that would make it possible to reach the resolution of the problem, which was considered normal at this initial stage, considering that the themselves still lacked knowledge and skills that would be acquired during the experiment.

The post-test data showed a significant evolution of the students, as most of the solutions presented contained well-structured algorithms, presenting a well-defined sequence of steps, as well as the correct use of the syntax derived from the programming language that was used in the experiment. In general, students were able to structure programs in C language that managed to achieve the objective of the problem presented in the post-test. It was noticed that the students had less difficulty in understanding the proposed problem and defining a consistent strategy for its resolution. This can be explained by the fact that in the post-test the students had a base of knowledge and skills that were developed during the experiment and that this helped them to elaborate the presented solutions.

In general, with RQ4 it could be seen that there was an evolution in student learning, as the results showed that they were able to develop structured algorithms through unambiguous sequences of instructions that made it possible to solve the proposed problems, which is indicative of efficiency and effectiveness of the proposed approach that was used in the experimental group, therefore the H04 was rejected.

7 Discussion

The results obtained in this study suggest that the learning effectiveness of the approach proposed in this work is superior to that obtained using a methodology of traditional classes, since hypotheses H01, H02, H03 and H04 were rejected, so that the average grades achieved by the experimental group during the evaluations were significantly higher when compared to the control group.

The positive results attributed to the experimental group may reflect the use of active methodologies and their adopted practices, which are mainly focused on students, seeking to promote their engagement and autonomy in the learning process. The results are similar to the statements of several authors in the specialized literature, who work with types of active intervention focused on the development of student autonomy in programming subjects [7], [9], [41], [42].

It is noteworthy that this work sought to compare only the effects of using active methodologies in the teaching algorithms, with no analysis of student performance being carried out. We only sought to work with students who had not yet taken the algorithms course, so prior knowledge that could be related to the course content was considered low.

From the qualitative feedbacks of the students, it was possible to perceive that they consider the teaching experience based on active methodologies very useful and that it can contribute positively if it is adopted in their courses, as the students reported

participating more actively in the classes. This corroborates the analysis made by authors of [22], who observed in their research that the use of forms of intervention focused on the student allows them to actively participate in classes, unlike when using the traditional teaching method. Participants also consider that other subjects could benefit if they adopted types of intervention similar to the one used in the experimental group.

With regard to the content used in the teaching units of the experimental group, which was based on the strategy adopted in [27], students consider that they are adequate and sufficient to promote the learning of the initial contents of computer programming. The distribution of contents was positively evaluated by the students, as well as the activities that were carried out in the experimental group, which always sought to promote engagement, autonomy and the exchange of knowledge by the students.

The control group that used an approach based on the traditional teaching methodology had its contents distributed in 3 periods (AV1, AV2 and AV3), which had a theoretical workload much higher than the practical one, with some exercise lists and a type of evaluation at the end of each cycle.

The theoretical workload greater than the practical one may have impacted on the averages of the students, which were lower than those obtained in the control group, as can be seen in the analysis of research questions and hypotheses in this work. This may be a reflection of the difficulty in learning certain key contents of the subject, which could be appropriated with less difficulty, with fewer lectures and more practical activities, which would stimulate student autonomy, exchange knowledge and consequently increase their engagement. This corroborates some existing investigations in the specialized literature that state that the concentration of students in lectures lasts approximately 20 minutes [43], [44].

With regard to the weaknesses identified in the experiment, it can be highlighted that the experimental group was conducted remotely, as a result of the COVID-19 pandemic. This atypical period made it even more challenging to hold the attention of students in the classroom (virtual), considering a number of factors that can arise with remote teaching, such as internet connection issues, power outages, or other hardware and software issues.

The experimental group, despite having a higher quantity of practices than the control group, still uses expository and dialogued classes to teach certain contents of the teaching plan, as presented by [30]. However, it was noticed that when classes lasted more than one class hour, students began to become more dispersed, losing some of their focus, corroborating the statements of [43] and [44], about the duration of classes. Students reported that the experiment could be further improved if dialogued expository classes were reduced to fit into a time interval of at most one class-hour per meeting. Therefore, it is intended to carry out further studies to better determine the appropriate duration for the realization of each expository and dialogued class for future experiments.

It is noteworthy that the experiment was carried out over a period of 17 meetings lasting approximately 4 hours of classes in both groups, totaling 68 hours of classes, with the aim of covering all teaching units planned for the algorithms subject. It was

observed that initially the students in the experimental group had a certain difficulty in interacting with other members during practical activities. Students reported that initial shyness made them more introspective during the first activities, but as the course progressed, the dynamics provided by the use of active methodologies made them overcome the initial limitations. The increase in interactions between students is similar to the reports of [45], which reinforces that the correct use of active methodologies makes students feel more motivated to carry out classroom activities.

It can be noted that the activities that most aroused the interest of students were those associated with the serious games approach, where students sought, through practical activities, to build simplified games that met the demands defined in the classroom.

8 Threats to validity

The results obtained in scientific research must be analyzed and interpreted with some caution, especially with regard to the generalization of results. For [46, p.4], "the validity of an experiment is related to the level of confidence that one can have in the process of experimental investigation as a whole". Therefore, some threats that can influence the results were identified, which follow the usual classification of threats to validity (internal, external, construction and conclusion). In the next subsections they will be presented, as well as some actions that could be taken to mitigate them. Therefore, it is recommended that the results presented in this article should be interpreted within the limits created by the threats presented below.

8.1 Internal validity

Internal validity helps to define whether the results obtained in a study conceive a truth for the observed population, that is, with this type of validity, we seek to verify whether the conclusions are adequate based on the results obtained from the studied samples [47].

The experimental and control groups had students from higher education courses in computing who had not yet taken the algorithms subject or its equivalent in their undergraduate courses. The participation of students was voluntary and everyone involved filled out a registration form that contained a consent form to carry out the research. The distribution of participants in the groups was done randomly so that there was no structuring of each group, this made it possible to reduce the confounding factor and make the groups have a great similarity.

Regarding a possible threat of internal validity related to maturation, the study did not limit the participants' search for knowledge to external materials (which were not used in the experiment), therefore the existence of this threat is possible. To reduce this threat, all materials used in both groups addressed the same contents (teaching units) and were previously made available to students. In addition, the professors in each group had great availability to assist students and answer questions outside the hours of the experiment.

An internal threat related to instrumentation was mitigated through the use of experts who were responsible for correcting the evaluations of both groups. Thus, the evaluation process was independent and was not influenced by the professors. The data that would be evaluated were passed on to the experts in an unidentified way, so they did not have any information about the student who was being evaluated.

About a possible threat involving the level of knowledge of the professors who participated in the experiment, the instrumentation used in each group, despite being different, addressed the same contents (teaching units). Likewise, the evaluations sought to work with the contents of each cycle, so both professors had a direction of the base content that should be taught at each stage, and corrections were made independently by experts.

8.2 External validity

External validity is characterized by the results obtained from the analysis of a sample and whether the conclusions are applicable to a broader population from which the sample was drawn, that is, it seeks to identify whether the results can be generalized to larger populations, that have the same characteristics as the sampling [47]. Thus, this study sought to use a context focused on undergraduate students from higher education courses in computing who had not yet attended the algorithms subject, so it is recommended that the results be generalized through this academic context.

The experiment was carried out with a sample of 34 students in each group (experimental and control), so the sample size is considered small and generalizations should be viewed with caution due to their limitations. Furthermore, the study was not tested in other populations, taking into account factors such as different academic levels of the participants. The study was also not used within a subject of a regular undergraduate course, the same occurred in the format of extension courses involving participants (professors and students) who had an active relationship with a university. The decision to use extension courses was made with a focus on evaluating the applicability together of active methodologies in a controlled scenario without causing damage to the absorption of knowledge by students in the undergraduate courses to which they are linked.

8.3 Construction validity

With the construction validity or construct validity, the aim is to demonstrate that the instruments of a study can measure or evaluate what is proposed, so that it is possible to determine characteristics that can explain possible dispersions of the results (scores) with the study average [48].

In this sense, the research questions that were defined in this study with their respective results were analyzed and one of the most important validity of the construction of this research is based on the efficiency and effectiveness of the use of active methodologies in the experimental group compared to the group of control. Thus, the

distribution of contents and activities in the experimental group was planned based on the levels of cognition extracted from Bloom's revised taxonomy [26].

It is important to highlight that the number of participants may not be enough to provide concrete evidence that allows the evaluation of the learning level obtained by students at each learning level in Bloom's revised taxonomy. Therefore, statements or generalizations should be viewed with caution, for example, it cannot be said that students who participated in the study in the experimental group will have success (approval) when taking the courses of algorithms and others related to programming in their courses, even having indications from the results of the experiment.

8.4 Conclusion validity

For [49], the validity of the conclusion seeks to reach the conclusion of a study based on the relationship between the treatment used and its final result. Therefore, this process is related to the proper use of statistical analyzes and assertive interpretations of the results obtained, as well as the use of reliable measures and the correct implementation of treatments [46].

With this study, it was noticed that the data collected from the samples of the experimental and control groups are small and, therefore, statements and generalizations of the results for larger populations must be interpreted with caution. To circumvent this possible threat of low statistical power in the distribution of samples and ensure greater reliability of the conclusions obtained in the study, a technique disseminated in some studies such as [50], [51], [52], [53], who used a strategy based on the adoption of a more robust statistical test that accepted small samples, with low statistical power.

9 Conclusion

This work presented the results of the application of an algorithms teaching plan that makes use of multiple active methodologies. The proposed approach presented significant results when compared to the use of the traditional teaching method, reaching relevant statistical gains that indicate greater teaching effectiveness when using the proposed approach.

It was presented the details of the adopted teaching strategy, as well as the correlation of the algorithm teaching units with the cognitive structures of Bloom's taxonomy.

The preliminary results obtained in this work are positive and considered statistically significant, however caution is recommended when making generalizations, as the target audience reached by the study is still considered low, and the teaching plan can still be replicated in other scenarios, such as for example in face-to-face classes.

As future work, it is intended to continue evolving the teaching plan and replicating the experiment in classroom classes of courses in computing, considering that the applications of the teaching plan were all made remotely due to the restrictions caused by the COVID-19 pandemic. In addition, it is also intended to analyze the perspective

of professors using the proposed approach. In the long term, it is intended to expand the use of the intervention strategy to other advanced programming subjects in higher education courses in computing, considering that the algorithms subject is considered introductory in this area.

10 References

[1] Papadakis, S. (2020). Evaluating a Teaching Intervention for Teaching STEM and Programming Concepts Through the Creation of a Weather-Forecast App for Smart Mobile Devices. In Handbook of Research on Tools for Teaching Computational Thinking in P-12 Education (pp. 31-53). IGI Global. https://doi.org/10.4018/978-1-7998-4576-8.ch002

[2] Chen, P., & Huang, R. (2017). Design thinking in app inventor game design and development: a case study. In Advanced Learning Technologies (ICALT), 2017 IEEE 17th International Conference on, (pp.139–141). IEEE. https://doi.org/10.1109/ICALT.2017.161

[3] Brasil. (2016). Ministério da Educação. Diretrizes Curriculares Nacionais para os cursos de Computação. Brasilia: MEC. Disponível em: http://portal.mec.gov.br/docman/novembro-2016-pdf/52101-rces005-16-pdf/file

[4] Vieira, C., Lima Junior, J., Vieira, P. (2015). Dificuldades no Processo de Aprendizagem de Algoritmos: uma Análise dos Resultados na Disciplina de AL1 do Curso de Sistemas de Informação da FAETERJ–Campus Paracambi. Cadernos UniFOA, v. 10, n. 27, p. 5–15. https://doi.org/10.47385/cadunifoa.v10.n27.293

[5] Hoed, R. (2016). Análise da evasão em cursos superiores: o caso da evasão em cursos superiores da área de Computação (Dissertação de Mestrado). Universidade de Brasília. Brasilia.

[6] Abu-Oda, G., El-Halees, A. (2015). Data mining in higher education: university student dropout case study. In: International Journal of Data Mining & Knowledge Management Process, v. 5, n. 1. https://doi.org/10.5121/ijdkp.2015.5102

[7] Giraffa, L. & Müller, L. (2017). Metodologia baseada em sala de Aula invertida e Resolução de Problemas relacionado ao cotidiano dos estudantes: uma proposta para ensinar programação para iniciantes. International Journal on Computational Thinking (IJCThink). https://doi.org/10.14210/ijcthink.v1.n1.p52

[8] Vasconcelos, B. (2016). Importância da validade externa na pesquisa científica. Revista de Cirurgia e Traumatologia Buco-maxilo-facial, 16(2), 04-05.

[9] Fonseca, J. & Brito, C. (2021). Percepção dos alunos do curso técnico em desenvolvimento de sistemas após vivências com o método de ensino peer instruction. Research, Society and Development, v. 10, n. 3, p. e10710312382-e10710312382. https://doi.org/10.33448/rsd-v10i3.12382

[10] HAYDT, R. C. Avaliação do processo ensino- aprendizagem. São Paulo: Áca, 1995.

[11] Barros, D. & Santos, J. (2018). Técnicas de estudo e gestão do tempo no auxílio a aprendizagem de fundamentos de algoritmos e lógica aplicada a computação. CIMATech, v. 1, n. 5. https://doi.org/10.21452/issn2447-5378.v1i5.2018p.1154

[12] Ramos, V., Wazlawick, R., Galimberti, M., Freitas, M., & Mariani, A. C.,A (2015). Comparação da Realidade Mundial do Ensino de Programação para Iniciantes com a Realidade Nacional: Revisão sistemática da literatura em eventos brasileiros. In: Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE).. p. 318. https://doi.org/10.5753/cbie.sbie.2015.318

[13] Amaral, E., Camargo, A., Gomes, M., Richa, C. H., & Becker, L., (2017). ALGO+ Uma ferramenta para o apoio ao ensino de Algoritmos e Programação para alunos iniciantes. In:

Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE). p. 1677. https://doi.org/10.5753/cbie.sbie.2017.1677

[14] Shi, N., Cui, W., Zhang, P. e Sun, X. (2018). Shi, N., Cui, W., Zhang, P. e Sun, X. (2018),. Evaluating the effectiveness roles of variables in the novice programmers learning. Journal of Educational Computing Research, Vol. 56, No. 2, pp. 181-201. https://doi.org/10.1177/0735633117707312

[15] Morais, c., Neto, F., & Osório, A. (2020). M., Dificuldades e desafios do processo de aprendizagem de algoritmos e programação no ensino superior: uma revisão sistemática de literatura. Research, Society and Development, v. 9, n. 10, p. e9429109287-e9429109287. https://doi.org/10.33448/rsd-v9i10.9287

[16] Chiu, CF (2014). Use of problem-solving approach to teach scratch programming for adult novice programmers. In Proceedings of the 45th ACM Technical Symposium on Computer Science Education, (pp.710– 711). ACM. https://doi.org/10.1145/2538862.2544284

[17] Sousa, R., Leite, F., Guimarães, Á., & Oliveira, A. (2020). Pré-Algoritmos–Ações de Apoio à Melhoria do Ensino de Graduação. Brazilian Journal of Development, v. 6, n. 3, p. 12625-12635. https://doi.org/10.34117/bjdv6n3-213

[18] Falkembach, G., Amoretti, M., Tarouco, L., (2003). Uma experiência de resolução de problemas através da estratégia ascendente: Ambiente de aprendizagem adaptado para algoritmos (a4). In challenges. international conference of information and communication technologies in education and 5th siie-international symposium in educational computing, 3., Proceedings. 2003.

[19] Saito, D., Washizaki, H., Fukazawa, Y., (2015). Work in progress: A comparison of programming way: Illustration-based programming and text-based programming. In: 2015 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE). IEEE, p. 220-223. https://doi.org/10.1109/TALE.2015.7386047

[20] Bulcão, J. (2017). Uma análise curricular das disciplinas de algoritmo e lógica computacional em cursos de licenciatura em informática e computação. Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte – IFRN. https://doi.org/10.5753/cbie.wie.2018.548

[21] Blikstein, P. (2018) Pre-College Computer Science Education: a Survey of the Field, Google LLC, Mountain View, CA [online] https://goo.gl/gmS1Vm (aces-sado em 15 de janeiro de 2019).

[22] Papadakis, S., & Kalogiannakis, M. (2019). Evaluating the ef-fectiveness of a game-based learning approach in modifying students' behavioural outcomes and competence, in an introduc-tory programming course. A case study in Greece. International Journal of Teaching and Case Studies, 10(3), 235-250. https://doi.org/10.1504/IJTCS.2019.102760

[23] Calsavara, A., Serra, A., Zampirolli, F., Carvalho, L., Jonathan, M., & Correia, R. (2018) Método baseado nos Referenciais de Formação da SBC para reestruturação de descritivos de disciplinas de Ciência da Computação em conformidade com as DCN de 2016. In Anais do XXVI Workshop sobre Educação em Computação. SBC. https://doi.org/10.5753/wei.2018.3517

[24] Zorzo, A., Nunes, D., Matos, E., Steinmacher, I., Leite, J., Araujo, R., Correia, R., Martins, S. (2017). Referenciais de Formação para os Cursos de Graduação em Computação. Sociedade Brasileira de Computação (SBC). 153p, ISBN 978-85-7669-424-3.

[25] ACM/IEEE (2020). Computing Curricula 2020 - CC2020. Paradigms for Global Computing Education. December 31, 2020.

[26] Ferraz, A. & Belhot, R. (2010). Taxonomia de Bloom: Revisão Teórica e Apresentação das Adequações do Instrumento para Definição de Objetivos Instrucionais. Gestão &

Produção. Handbook I, Cognitive Domain: Longmans. https://doi.org/10.1590/S0104-530X2010000200015

[27] Garcia, F., Carvalho, E., & Oliveira, S. (2020) "A survey of teaching methods for a programming subject: A literature review". 17th CONTECSI, Brasil.

[28] D. Ramos, E. Oliveira, I. Monteverde, and K. Oliveira, "Trilhas de Aprendizagem em Ambientes Virtuais de Ensino-aprendizagem: Uma Revisão Sistemática da Literatura". Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação - SBIE), 26(1), 338, 2020.

[29] Garcia, F., Carvalho, E., & Oliveira, S. (2021). "A strategy for the application of active teaching methodologies for the algorithms subject: an exploratory study on a way to use the methodologies". 18th CONTECSI, Brasil.

[30] Garcia, F., Carvalho, E., & Oliveira, S. (2021) "Use of active methodologies for the development of a teaching plan for the algorithms subject". 51th Annual Frontiers in Education – FIE'21. Lincoln, Nebraska - USA.

[31] Garcia, F., Carvalho, E., Oliveira, S. (2021). "Application an algorithm teaching plan using active methodologies: a case study report". 18th CONTECSI, Brasil.

[32] Levanova, E. A., Galustyan, O. V., Seryakova, S. B., Pushkareva, T. V., Serykh, A. B., & Yezhov, A. V. (2020). Students' Project Competency within the Framework of STEM Education. International Journal of Emerging Technologies in Learning (iJET), 15(21), pp. 268–276. https://doi.org/10.3991/ijet.v15i21.15933

[33] Setyarini, T., Mustaji, M., & Jannah, M. (2020). The Effect of Project-Based Learning Assisted PANGTUS on Creative Thinking Ability in Higher Education. International Journal of Emerging Technologies in Learning (IJET), 15(11), pp. 245-251. https://doi.org/10.3991/ijet.v15i11.12717

[34] Tezer, M., Orekhovskaya, N. A., Shaleeva, E. F., Knyazeva, S. A., & Krokhina, J. A. (2021). The Effectiveness of STEM Education Applied with a Distance Education Approach. International Journal of Emerging Technologies in Learning (iJET), 16(19), pp. 180–192. https://doi.org/10.3991/ijet.v16i19.26061

[35] Bigolin, N., Silveira, S., Bertolini, C., Almeida, I., Geller, M., Parreira, F., & Macedo, R. (2020). "Metodologias Ativas de Aprendizagem: um relato de experiência nas disciplinas de programação e estrutura de dados". Research, Society and Development, 9(1), e74911648-e74911648. https://doi.org/10.33448/rsd-v9i1.1648

[36] Sousa, R. & Leite, F. (2020). Usando gamificação no ensino de programação introdutória. Brazilian Journal of Development, 6(6), 33338-33356. https://doi.org/10.34117/bjdv6n6-043

[37] Freire, L., Coutinho, J., Lima, V., & Lima, N. (2019). Uma Proposta de Encontros de Tutoria Baseada em Metodologias Ativas para Disciplinas de Programação Introdutória. In Anais dos Workshops do Congresso Brasileiro de Informática na Educação (Vol. 8, No. 1, p. 298). https://doi.org/10.5753/cbie.wcbie.2019.298

[38] Quaresma, J., Eliasquevici, M., Oliveira, S. (2018). Gamificação e Avaliação de um Framework de Ensino e Aprendizagem para a Disciplina Algoritmos: Um Estudo de Caso. TISE – Conferência Internacional sobre Informática na Educação - Nuevas Ideas en Informática Educativa, Volumen 14, p. 60 - 69. Santiago de Chile.

[39] Drini, M. (2018). Using new methodologies in teaching computer programming. IEEE Integrated STEM Education Conference (ISEC), Princeton, NJ, 2018, pp. 120-124. https://doi.org/10.1109/ISECon.2018.8340461

[40] Silva, G. (2017). Flipped Classroom, aprendizagem colaborativa e Gamification: conceitos aplicados em um ambiente colaborativo para ensino de programação. Master's thesis, Universidade Federal de Pernambuco.

[41] Marinho, C., Moreira, L., Coutinho, E., Paillard, G., & de Lima, E. T. (2016). Experiências no uso da metodologia coding dojo nas disciplinas básicas de programaçao de computadores em um curso interdisciplinar do ensino superior. In Anais dos Workshops do Congresso Brasileiro de Informática na Educação (Vol. 5, No. 1, p. 1097). https://doi.org/10.5753/cbie.wcbie.2016.1097

[42] Oliveira, M., Neves, A., Lopes, M., Medeiros, H., Andrade, M., & Reblin, L. (2017). Um curso de programação a distância com metodologias ativas e análise de aprendizagem por métricas de software. RENOTE, 15(1). https://doi.org/10.22456/1679-1916.75143

[43] Branstetter BF, Faix LE, Humphrey AL, Schumann JB. Preclinical medical stu-dent training in radiology: the effect of early exposure. AJR Am J Roentgenol, 2007 Jan; 188(1): W9-14. ISSN 1546-3141. Disponível em: https://doi.org/10.2214/AJR.05.2139

[44] El-Ali A, Kamal F, Cabral CL, Squires JH. Comparison of traditional and webbased medical student teaching by radiology residents. J Am Coll Radiol, 2019; 16(4):492-495. ISSN 1546-1440. Disponível em: https://doi.org/ 10.1016/j.jacr.2018.09.048

[45] Hartwig, A. K., Silveira, M., Fronza, L., da Silveira, H. U. C., Mattos, M., & de Araújo Kohler, L. P. (2019, November). Metodologias ativas para o ensino na graduação na área de Computação. In Anais do Workshop de Informática na Escola (Vol. 25, No. 1, pp. 1134-1138). https://doi.org/10.5753/cbie.wie.2019.1134

[46] Lima, V, Neto, A., & Emer, M. (2014). Investigação experimental e práticas ágeis: ameaças à validade de experimentos envolvendo a prática ágil Programação em par. Revista Eletrônica de Sistemas de Informação, 13(1). https://doi.org/10.21529/RESI.2014.1301005

[47] Vasconcelos, A., Andrade, G., Brainer, S., SOARES, R., Santos, L., & Souza, P., (2019) As estratégias de ensino por meio das metodologias ativas. Brazilian Journal of Development, 5(5), 3945-3952.

[48] Raymundo, V. (2009). Construção e validação de instrumentos: um desafio para a psicolinguística. Letras de hoje, 44(3).

[49] Travassos, G., Gurov, D., Amaral, E. (2002). Introdução à engenharia de software experimental. Rio de Janeiro: [49] COPPE/UFRJ.

[50] Furtado, J. (2020). Uma Abordagem para o Ensino do Controle Estatístico de Processos em Cursos Superiores de Computação; orientador, Sandro Ronaldo Bezerra Oliveira. - 2020. 177f. Tese (Doutorado) - Universidade Federal do Pará. Instituto de Ciências Exatas e Naturais. Programa de Pós-Graduação em Ciência da Computação. Belém.

[51] Chaves, R., Wangenheim, C., Furtado, J., Oliveira, S., Santos, A. & Favero, E. (2015). Experimental Evaluation of a Serious Game for Teaching Software Process Modeling. IEEE Transactions on Education, vol. PP, no. 99. https://doi.org/10.1109/TE.2015.2411573

[52] Wangenheim, C., Thiry, M., Kochanski, D. (2009). Empirical evaluation of na educational game on software measurement. Empirical Software Engineering, Kluwer Academic Publishers Hingham, MA, USA,v. 14, n. 4, p.418-452. https://doi.org/10.1007/s10664-008-9092-6

[53] Pfahl, D., Laitenberger, O., Dorsch, J., Ruhe, G. (2003). An externally replicated experiment for evaluating the learning effectiveness of using simulations in software project management education. Empirical Software Engineering, Kluwer Academic, The Netherlands, v.8, p.367-395. https://doi.org/10.1023/A:1025320418915

11 Authors

Fabrício Wickey da Silva Garcia, Doctoral student in Computer Science with emphasis in Software Engineering from Graduate Program in Computer Science (PPGCC) at Federal University of Pará (UFPA). Currently he is professor and researcher at the Federal Rural University of the Amazon (UFRA). His research areas are: Algorithms, Software Engineering, Software Quality and Informatics Education (email: fabriciogarcia@ufpa.br).

Sandro Ronaldo Bezerra Oliveira, PhD in Computer Science with emphasis in Software Engineering and did his postdoctoral internship at the Informatics Center, Federal University of Pernambuco. Currently he is professor and researcher at the Faculty of Computing (FACOMP) and Graduate Program in Computer Science (PPGCC) at Federal University of Pará (UFPA). He is the Lead Coordinator of the SPIDER research project, which has won many scientific awards and has already graduated many doctoral, master, graduate and scientific initiation students in Computer Science. He is consultant, appraiser and instructor of the MPS.BR and CMMI software and service quality models. His research areas are: Informatics in Education, Software Engineering and Software Process Improvement (email: srbo@ufpa.br).

Elielton da Costa Carvalho, Master student in Computer Science with emphasis in Software Engineering from Graduate Program in Computer Science (PPGCC) at Federal University of Pará (UFPA). His research areas are: Project Management, Software Engineering and Software Quality (email: elielton.carvalho@icen.ufpa.br).

Article submitted 2021-12-08. Resubmitted 2022-02-10. Final acceptance 2022-02-12. Final version published as submitted by the authors.

12 Appendix A

Table 8. Details of the content of algorithms used in the teaching plan versus Bloom's taxonomy

Modules	Topics	Activities	Expected Results	Learning Levels
Initial Evaluation Module	Presentation of the teaching plan student profile identification and placement test	Application of initial level evaluation questionnaire	Identify the profile of each student, as well as the skills and prior knowledge they have	Not applicable
Teaching Module I	Content: Initial Concepts of Algorithms Units: 1.1, 1.2 e 1.3	Theoretical and practical class	The student must know the basic contents of algorithms	Remember / Factual
		Challenge 1	The student must be able to correlate the basic contents of algorithms with possible problems and computational limits	Understand / Procedural
		Extraclass challenge 1	The student must be able to solve problems that involve initial concepts of algorithms	Apply / Procedural
	Content:	Theoretical and	The student must be able to under-	Remember /

	Problem solving algorithmically Units: 1.4, 2.1 e 2.2	practical class	stand how real problems can be solved algorithmically	Conceptual
		Challenge 2	The student must be able to solve problems algorithmically	Understand / Procedural
	Content: Problem solving algorithmically (Continued) Units: 2.3, 2.4 e 2.5	Theoretical and practical class	The student must be able to understand how real problems can be solved algorithmically	Remember / Conceptual
		Challenge 3	The student must be able to solve problems algorithmically	Understand / Procedural
	Coding Dojo Practice (All content in current module)	Coding Dojo 1	The student must be able to analyze the problems and solve them based on their knowledge of algorithms and pseudocode	Analyze / Conceptual
			The student must be able to create solutions that solve the problems using algorithms and pseudocodes	Create / Procedural
			The student must be able to evaluate the implemented solutions and make decisions based on their knowledge of the Teaching Module I	Evaluate / Factual
	Evaluation (All content in current module)	Mission 1	The student must be able to apply the knowledge acquired during Teaching Module I to solve problems that involve all the content taught	Apply / Metacognitive
		Delivery of Extra-class Challenge 1	The student must deliver the Extra-class Challenge 1 resolution through the Virtual Learning Environment	Not applicable
Teaching Module II	Content: Introduction to Scratch - variables and operators Units: 3.1 e 3.2	Theoretical and practical class	The student must know Scratch and how the contents of variables and operators are worked in this environment	Remember / Factual
		Challenge 4	The student must be able to solve problems about variables, operators and selection structures using Scratch	Understand / Procedjural
		Extraclass challenge 2 available	The student must be able to solve algorithmic problems using Scratch	Apply / Procedural
	Content: Conditional and Control Structures in Scratch Units: 3.3 e 3.4	Theoretical and practical class	The student must know the conditional and control structures through Scratch	Remember / Factual
		Challenge 5	The student must be able to solve problems involving condition and control structures using Scratch	Understand / Procedural
	Content: Repetition Structures in Scratch Units: 4.1 e 4.2	Theoretical and practical class	The student must know the repetition structures through Scratch	Remember / Factual
		Challenge 6	The student must be able to solve problems involving repetition structures through Scratch	Understand / Procedural
	Content: Lists	Theoretical and	The student must know what repeti-	Remember /

	(Arrays) in Scratch Units: 4.2 e 4.3	practical class	tion structures and vectors in Scratch are and how they work	Factual
		Challenge 7	The student must be able to solve problems involving repetition structures and vectors through Scratch	Understand / Procedural
	Coding Dojo Practice (All content in current module)	Coding Dojo 2	The student must be able to analyze the problems and solve them based on their knowledge of Scratch	Analyze / Conceptual
			The student must be able to create solutions that solve problems using Scratch	Create / Procedural
			The student must be able to evaluate the implemented solutions and make decisions based on their knowledge of the Teaching Module I and II	Evaluate / Factual
	Evaluation (All content in current module)	Mission 2	The student must be able to apply the knowledge acquired throughout Teaching Module II to solve problems that involve all the content taught.	Apply / Metacognitive
		Delivery of Extra-class Challenge 2	The student must deliver the Extra-class Challenge 2 resolution through the Virtual Learning Environment	Not applicable
Teaching Module III	Content: Introduction to C language Units: 5.1 e 5.2	Theoretical and practical class	The student must know how the C programming language works	Remember / Factual
		Challenge 8	The student should be able to solve basic programming problems (declaring variables, data input and output, logical operators, functions) using C programming language	Understand / Procedural
		Extraclass challenge 3 available	The student must be able to solve algorithmic problems using the C programming language	Apply / Procedural
	Content: Conditional and Control Structures in C Units: 5.3 e 5.4	Theoretical and practical class	The student should know how conditional, control and parameter passing structures are developed using the C programming language	Remember / Factual
		Challenge 9	The student must be able to solve problems that determine the conditional and control structures with parameter passing through the C programming language	Understand / Procedural
	Content: Repetition Structures in C Units: 5.4 e 5.5	Theoretical and practical class	The student must know how repetition structures are developed and how the parameters of the C programming language are passed	Remember / Factual
		Challenge 10	The student must be able to solve parameter passing problems using repetition structures through the C programming language	Understand / Procedural
	Content: C language vectors Units: 5.5 e 5.6	Theoretical and practical class	The student should know how storage classes and arrays are developed using the C programming language	Remember / Factual
		Challenge 11	The student must be able to solve	Understand /

			problems that make use of storage classes and vectors through the C programming language	Procedural
	Coding Dojo Practice (All content in current module)	Coding Dojo 3	The student must be able to analyze the problems and solve them based on their knowledge of the C programming language	Analyze / Conceptual
			The student must be able to create solutions that solve the problems using a C programming language	Create / Procedural
			The student must be able to evaluate the implemented solutions and make decisions based on their knowledge acquired throughout Teaching Modules I, II and III	Evaluate / Factual
	Evaluation (All content in current module)	Mission 3	The student must be able to apply the knowledge acquired throughout Teaching Module III to solve problems that involve all the content taught	Apply / metacognitive
		Delivery of Extra-class Challenge 3	The student must deliver the Extra-class Challenge 3 resolution through the Virtual Learning Environment	Not applicable
Final Evaluation Module	Final Evaluation of the Teaching Plan	SWOT (Strengths, Weakenesses, Opportunities and Threats) Analysis	Identify the main points related to areas, weaknesses, proposals and opportunities from the application of the teaching plan	Not applicable