Teaching Computational Thinking to a Student with Attention Deficit Through Programming

Felippe Fernandes da Silva
Department of Informatics
State University of Maringa
Maringa, Brazil
felippefernandes10@yahoo.com.br

Linnyer Beatrys Ruiz Aylon
Department of Informatics
State University of Maringa
Maringa, Brazil
linnyer@gmail.com

Daniela Eloise Flôr
Communication and Information Department
Federal Institute of Paraná
Paranavai, Brazil
daniela.flor@ifpr.edu.br

***Abstract*—This research-to-practice full paper deals with the difficulty that it is possible to teach computational thinking through the teaching of algorithms and programming to a person with ADD. We propose approaches to teach algorithms, knowing that it is a difficult task for many teachers. Likewise, inspiring the student's interest also has its difficulties. The Brazilian Computer Society (SBC) understands that it is fundamental and strategic for Brazil that computer content is taught in basic education. Considering the 4.0 education of a connected generation, the thinking of computer education is relevant to the construction of youth and adult education. When we include people with ADD, this challenge is not trivial. This work presents a case study carried out with a high school student who has ADD and the techniques that were used to teach him to program to maintain his interest in the area. The method used was divided into three stages: Computer Discovery (1); Introduction to computational thinking through programming (2); Learn by Doing, in line with real-world problem solving using pair programming technique and challenge-based learning strategy (3). In order to investigate the effectiveness of the developed strategies, an experimental case study was carried out with a student who has ADD. The student was introduced to the content and approaches for 34 weeks, totaling 204 hours. The student presented a strong indication of learning and will forever have the ability of computational thinking to program and develop their applications.**

***Index Terms*—Computational Thinking, DDA, Active Methodologies, Education 4.0, Teaching Programming.**

I. Introduction

The impact of technological evolution on the life of the contemporary citizen has brought different challenges and uncertainties. This is because known and other unknown skills and abilities will be essential in personal and professional relationships in the near future. Education is the link that unites society with these new demands and Computational Thinking (PC), in turn, is seen as a fundamental skill for the next generations.

Computational Thinking has been popularized, among other ways, through the teaching of algorithms and programming [11], as a strategy that provides necessary cognitive resources for problem solving, across all areas of knowledge. It is one of the contents suggested by the initiatives of insertion of Computing in Basic Education and several works have reported successful educational interventions with students of different levels of education [3].

Thus, an inclusive perspective of this process, to contemplate students with Attention Deficit Disorder (ADD), is essential. People with ADD have insufficient Executive Functions (EF). Executive functions are a set of skills necessary for school success and are transversal to any type of learning.

The understanding of executive functions and how they can be ensured during the knowledge acquisition process of a student with ADD, guided the development of this research. This paper presents a case study where a sequence of methods were applied to teach computational thinking to a high school student who has ADD and this learning will help in other daily routines, considering that the student will better organize his way of thinking as we develop his cognitive dysfunctions [19].

The remainder of this paper is structured to present, in Section II, concepts and related works that supported the research. Section III details the work methodology. Then, Section IV presents the validation of the proposed method. The impact on the student's life is discussed in Section V and the conclusions and future work can be found in Section VI.

II. Related Works

The computer as a learning tool has been explored by Seymour Papert since the 1970s. Over time, several researchers have contributed to strengthening the relationship between education and computing. Jeannete Wing [10], recovered the term Computational Thinking and reframed the idea by correlating cognitive processes and tools common to computer scientists with a set of efficient skills that can be used to solve problems in various contexts. Currently, research groups, social institutions and companies around the world have expanded the discussions and movements that seek the approximation of computing to basic education, including the Brazilian Computer Society (SBC) [6].

There are different approaches to Computational Thinking [18]. In this research, we adopted the perspective that consolidates computer science as an indispensable area of knowledge for future generations. More specifically, we opted for the bias of the contents of algorithms and programming following Denning's [14] interpretation that the skills involved in the involved in the creation of computer programs favor mental training applicable to problems in other contexts.

In addition to the conceptual concern about the PC or the best introduction technique, it is necessary to understand the impact of computing on society and ensure equity and diversity of participation in the educational process, as Blikstein [4] ponders.

Therefore, it is imperative to evaluate, adapt and develop methods that include individuals with executive dysfunctions often associated with students with specific learning difficulties, such as Attention Deficit Disorder, in ways that are guaranteed the same personal opportunities or future professionals.

Fonseca [19] details executive functions as the set of mental tools that are essential for learning to learn. He cites its presence in complex mental processes by which the individual optimizes his cognitive performance, improves his adaptive responses and his behavioral performance in situations that require the operationalization, coordination, supervision and control of cognitive and conative processes, basic and higher.

The executive functions belong to nine different axes, namely: attention; perception; working memory; control; ideation; planning and anticipation; cognitive flexibility; metacognition and execution. In turn, the development of algorithms or computational products exercises skills that include abstraction, decomposition, pattern recognition, elaboration of algorithms, generalization, parallelism, simulations, among others. Abstraction is exercised during the interpretation of the problem, identification of inputs and outputs and elimination of unnecessary details. Decomposition is useful for dividing the problem and the solution into smaller parts, favoring the common reuse for pattern recognition. The elaboration of algorithms deals with the formalization and refinement of the solution. The generalization guarantees the expansion of the solution to problems with the same classification. Parallelism organizes resources and performs subtasks concurrently. The simulations deal with the representation of data and processes using models that can even be compared according to criteria.

Due to the similarities between the themes, the method developed in this work sought to base the skills common to a computer professional on the basis of the executive functions mentioned above, inspired by the cognitive behavioral work proposed by Barkley [16]. The technique sought the empowerment of the individual, enabling him to be the agent of change itself. For this, he suggested seven steps for training self-control and finding solutions, whether they are:

- clearly define the problem;
- define the intended objective and the desired alternative for solving the problem;
- list the possible alternatives to solve the problem;
- evaluate from 1 to 10 each option that represents, from the most negative to the most positive;
- select the best option and put it into practice for a week or as long as necessary; if the option does not work, practice the other options listed until the problem can be resolved;
- be open to disagree; if the initial attempt fails, be honest to accept the flaws and look for other options to solve the problem;
- comply with the plan and evaluate the results.

For this reason, this work proposes a method of teaching computational thinking as a stimulus to mastery and management of competences in executive processes that will contribute to the academic, social and professional success of students with attention deficit.

III. Methods

The developed method is divided into three stages: Computer Discovery (1); Introduction to computational thinking through programming (2) [9][20]; Learning by Doing (3) [8][12].

A. Computer Discovery

The initial purpose of the method was to teach computational concepts to someone who has never had contact with programming. In order for the work to be successful, we started by treating primary computation definitions so that the application sequence was as effective as possible. Therefore, the introduction included concepts about hardware and software.

Right at the beginning we found the first challenge, which was to make him understand how a computer works internally and that all components have a function. To attract the interest of the student with attention deficit, it was necessary to take these components apart and show them, explaining each piece individually. Upon receiving the pieces, the student's immediate reaction was to touch them and try to understand how it connected with the others.

After understanding the physical components of a computer, we started to explain the concept of software. Based on the student's express interest in computer games, the approach was initiated by drawing a parallel that they are software being executed inside hardware. By understanding the meaning of hardware and software, the student was able to situate himself as to the components of a computing system and understand that existing problems in the real world can be computationally transposed and treated using software.

The next step dealt with demystifying the concept of software. To this end, a challenging scenario was provided for the student to dedicate himself to abstraction, mental organization and suggested solutions. This exercise was closely related to some executive functions that exercise focus, data selection, treatment of distractors and the exercise of creativity. In a targeted dialogue, the student was asked how he would solve the challenge. Faced with the presented solution, the teacher transcribed the answer. At the end, the answer was reformulated in a pseudocode format so that the student understood the importance of the logical sequence and the reduction of ambiguities.

After a first example was presented and worked on, the student was challenged to solve other situations using pseudocode. Thus, the student's interest in exercising the development of logic was born to find solutions for the proposed exercises. The plurality of examples made the student understand that there was a range of possible ways to find a possible final result.

B. Computational Thinking and Programming

After being introduced to algorithmic logic through pseudocode, the student can understand that it is possible to transfer existing problems in the real world to the computational environment. Thus, it was possible to start the computational thinking approach.

To continue the teaching of computational thinking, it was decided to use programming, which was one of the objectives of this work. For having attention deficit, the student in turn was more interested in the practical part of the application. In order to introduce computational weighing by means of programming, the explanation of some reserved words in the Pascalzim language [13] started. The basic structure of an algorithm and its modeling in the tool was explained to the student.

After learning the reserved words, it was up to the student to develop the logical part of the problem in a given exercise. In principle, the problems provided to the student were the same in which we had already made a pseudocode. With this familiarity with the learning method, the student obtained clarity on how to solve a problem of computational thinking through programming. Then, when acquiring maturity in the specified language, the student was able to think only about the solution, considering that programming was no longer a challenge.

In order to explore the concepts of computational thinking, some activities were thought out so that we could couple with this teaching. The concept of abstraction is acquired during the interpretation of the problem. The student receives the exercise and through it has to understand the statement. Decomposition is the concept in which the student needed to understand the problem and divide it into smaller parts in order to find a solution. The concept of pattern recognition can be absorbed through the repetition of the exercises, as they all had a proximity to each other.

Other concepts of computational thinking were evaluated in this work, such as the development of algorithms, which can be found in the refinement of the solution, where the student can see that his solution was not the best possible, despite obtaining a valid answer [15]. The concept of generalization guarantees a thought acquired also through repetition, since the student was able to use experiences acquired with previous solutions and develop a new one with similar classification exercises. Finally, the parallelism present in computational thinking can be perceived when the student organizes his thinking concurrently with the solution.

C. Learning by Doing: Problems of Interest

We always try to follow a continuous line of reasoning with the student. For that, we thought of proposing that the student solve real-world problems where it was possible to obtain a better understanding of the problem. With a real example the student could think better about the solution of the exercise and strive to find a correct answer.

When presented with problems where the application was not practical in everyday life or that did not stimulate the student's interest, it was noticed that he took more time to understand and organize his ideas. Realizing this, we changed our teaching approach. An example of an applied exercise was a problem for calculating the arithmetic mean between four integers.

Starting the development of this exercise, the student had some difficulty in visualizing the problem and transposing it to the computational environment. After resolving the issue (which took a considerable amount of time), we opted for an adaptation, in the following class we present a similar example but directed to a problem situation in the student's daily life. The exercise consisted of: ”Given four grades by a student at school, calculate the average between them and report whether the student passed or failed”. It was then realized that the student was able to think about the problem more quickly and efficiently because he was able to apply it to his reality, which motivated him to find the correct answer.

One of the strategies in teaching algorithms is repetition, which can cause disinterest on the part of the learner. Therefore, we chose to explore the student's automotive interest, identified through interviews. In this way, we designed exercises where the word ”cars” was constant. For example, instead of proposing an exercise such as: ”Given a circumference, calculate the radius and diameter of the figure”, we propose a problem where: ”A car of a certain brand is on display. Calculate the radius and diameter of the wheel.”

When applying this approach, we found that the student was more committed to solving the exercise and reaching a solution. In this way, we were intertwining computational thinking; the problems to be solved; and the student's interest.

D. Learning by Doing: Pair Programming

At the beginning of our activities, when the student did not yet have formalized computational thinking and more acute logic, his low familiarity with coding ended up hindering the development of the work. Although the programming language and its structure of use have been explained, the difficulty in starting to develop the challenges was still notorious.

Informally, it was possible to verify that the apprentice understood the problem and had a clear line of reasoning to find the solution. However, there was a certain inertia to be overcome, to carry out the transposition of the idea into the code, starting the implementation. In order to assist in this step we employ the concept of pair programming.

As the student did not have extensive experience with program development, we decided to give an *insight* at the beginning of the resolution. For this, the teacher started the code and structured some steps of the solution. Questions were made so that the student could organize himself mentally and fix the instructions, practicing various skills during the classes. After the teacher molded the algorithmic structure, the student took control of the computer and developed the logical part of the problem.

Over the weeks the student gained independence in the development of the algorithms and with that the classes started to flow better, and in this stage the discussions related only to the logical solution of the challenge.

Paired programming started to be used sporadically, usually when the student had some difficulty with more complex exercises. In this case, the teacher's intervention continued to occur according to the technique, after the outline of a proposed solution developed by both and after the beginning of the coding by the teacher, the student felt confident and took back control of the computer to develop the rest of the solution. With this teaching dynamic, we managed to hold the student's attention.

E. Learning by Doing: Challenge-Based Learning

In conversations with the student during classes, seeking a better understanding of his interests, it was noticed that the student had an enormous interest in games. At first, racing games and MineCraft. Seeking to insert the student in the computational universe involving programming, we made him aim for the final result of a problem. Due to his attention deficit, it was often difficult to lie centered on the development of the problem offered. That way we would need to find a way out.

Then, the way out to estimulate the student´s interest in solving a problem was to use the challenge-based learning strategy. In it the student would have the challenge of finding the final result and this is similar to recurring challenges in computer games.

When we applied this technique, the student recognized that the approach was challenging and started to have a greater incentive to solve the indicated problems. He realized that his performance and focus were greater when he was entertained in finding the answer to a proposed problem, along with other techniques applied previously, such as problems applied to the real world and his personal interests.

IV. Validation

To validate the proposed method, we will discuss about: The student's profile (A); the evaluation to obtain results (B); and the results of the applied test followed by discussions about the student's performance (C) [1][5].

A. Student Profile

The student selected for the development of this work was chosen because he is part of our target audience. Based on this principle, there was an opportunity to spread the teaching of computational thinking through programming. For that, it is necessary to understand that teaching programming requires a high level of logical reasoning and mathematical knowledge. In this concept, our high school student seemed to fit the mentioned profile. Although the student's mathematical knowledge is not as advanced, it was possible to develop our method normally.

Our student was fifteen years old at the beginning of this case study and was in his first year of high school. The student in question receives educational support at a psychopedagogy clinic and is diagnosed with Attention Deficit. According to this support clinic, the attention deficit manifested in our student consists of the main characteristics of this disorder, such as inattention and hyperactivity.

The work developed had a great challenge, because the student never had contact with the programming. At the school where he studies, there are no algorithm classes, only basic education subjects. That way, the student had willingness to learn how to program, as he has always involved with online games and had a great interest for technology. The student reported this interest in the clinic he attends and through this interest the opportunity to developing this work arose. On the other hand, additional programming classes were included only for the student's knowledge, with no intention of obtaining better grades at school.

To carry out the activities, classes were held at the student's or teacher's residence. The choice of the location for the development of the work was made so that the student felt in a safe environment and that his initial rigidity was overcome. Due to school commitments, classes were often rescheduled, however, this obstacle was resolved with the replacement of the class on a day when the student was available.

B. Evaluation Methods

In order to evaluate the developed method, we propose classes for 34 weeks, totaling 204 hours. Before each class, a review of the content learned by the student was carried out, where questions about the previous class were answered. At the end of a theoretical explanation of the content, the student was subjected to exercises on the subject learned in class. This is also a strategy adopted by professional developers in an approach adapted from the agile methodology called Scrum. The idea is that in fifteen minutes of conversation the pair will connect with what they are doing, that is, with the challenge they are facing, discuss expectations and plan the actions of that meeting.

To evaluate the knowledge learned at the end of all classes, a test was applied in order to consolidate that the student's learning was carried out. This test was divided into three parts. In the first part (i) the teacher selected five different problems and listed them between 1 and 5 gradually, with 1 being the easiest exercise and 5 the most difficult. Questions such as: ”Find the sum of all the numbers in the interval between them”, ”Make a program that checks whether a word is ambiguous or not”, ”Calculate the overall average of a student in 4 exams”, ”Countdown from a number to x ”,” Check if a number is prime”. Then the student should find the solution by programming these exercises and the correctness to find the final solution would be evaluated. The second part (ii) consists of asking the student to list from 1 to 5 what levels of difficulty he found in the exercises. Finally, in the third part (iii) the student was asked to use the same evaluation criteria to inform which types of questions he most liked to solve.

In the first stage (i) the student solved the programming exercises without the help of the teacher, using only his logic and what was learned in the classroom. Then he presented the solved exercise and the teacher checked its correctness. In the second (ii) and third stages (iii) he evaluated the questions without knowing the level of difficulty and also without the professor's external interference.

C. Results and Discussion

For the analysis of results it was expected that the student would find the correct answer in most of the questions. This would demonstrate whether the knowledge absorbed was sufficient for the learning of computational thinking and the effectiveness of programming. To determine confidence in the correctness of our data, expect the student to have reached more than 80% of the questions, or to be considered high due to his attention deficit. The obtained result reached the total precision of the correct answers, however the percentage was very close, totaling 84% and exceeding our expectations.

We present the figure 1, that shows the student's correct answers for each question. The questions presented in the first part of the evaluation involve the logic expected to solve the exercises in the second part. In this way, similar questions were asked between the two steps so that there was a stimulus of thought already carried out by the student and that we could overcome his lack of attention. The values in the table represent the correction of each problem, in which the answers vary between ”0.0” (totally incorrect) and ”5.0” (totally correct). Following the evaluation method, problem 1 is easier and question 5 is more difficult, gradually. To correct the exercises, we divided the evaluation into three points of correction: Logic, worth ”2.0” points; Development, worth ”1.0” point e; Solution, worth ”2.0” points.

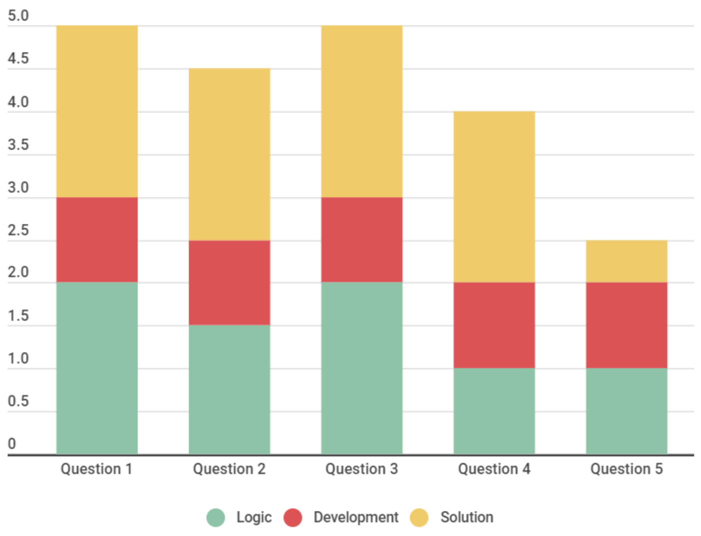

Fig. 1. Exercise evaluation chart.

The correction points were evaluated as follows: The logic was evaluated as being the student's reasoning to organize his ideas and understand the problem. Thus, after this modeling, it was up to the student to send this knowledge to the algorithm. In this stage we have the development, which was evaluated as the student's ability to transpose his ideas to the computational environment, organizing his reasoning and showing the mastery of language. Finally, the solution was assessed as being how close the student was to the expected result.

Analyzing figure 1, we observed that for problems 1 and 3 the student had no difficulties and completely met what was expected. In problems 2 and 3, we see that the student scored ”4.5” and ”4.0”. These grades were given because, despite having found the final solution, the logic used was not the one expected by the classes taught. For this reason, the decrease of some points. Finally, problem 5 can be seen that the student obtained regular logical reasoning and, unlike the other exercises, he was unable to reach the final solution completely. For the trivial case, the student found the solution and therefore the partiality of the grade in the evaluation.

We can also highlight that, after the organization of ideas and logical reasoning, the student had no difficulty in developing the algorithm, showing then a mastery of the programming language. It stands out that the biggest difficulty was the logical part.

In the second part of our evaluation, some exercises of general scope were separated so that the student could evaluate them. It was preferred not to choose exercises modified by the student's interest, as it could influence his individual assessment. For this reason, the table I shows what these problems are.

TABLE I
PROBLEMS PROPOSED FOR STUDENT ASSESSMENT

Problem	Question Description	Difficulty
1	Hello World	1
2	Sum numbers	1
3	Sort numbers	2
4	Show even numbers	2
5	Sum all numbers using FOR	2
6	Calculates salary	3
7	Calculates average students with FOR	3
8	Banking system	3
9	Countdown	4
10	Calculates and identifies triangle	4
11	Calculates prime number	5
12	Calculates factorial of a number	5

Looking at the table I, we can say that the problems are presented in increasing order according to their level of difficulty. Therefore, each of these problems was presented to the student and the figure 2 presents the student's evaluation.

When analyzing the figure 2 we noticed that the results were similar. For problems where there is a difference in level according to the student, we can explain that the student's lack of experience has influenced the assessment and for him, using his logical reasoning, it was easier.

We can highlight that problems 1, 3, 7, 8, 9 and 11 had the same evaluations, both for the teacher and the student, corresponding to 50% equivalence. Problems 2, 4, 5 and 6 were evaluated as being more difficult for the student. Problem 4 stands out, which was the only one in which the student diverged ”2.0” points from the teacher's assessment. Finally,

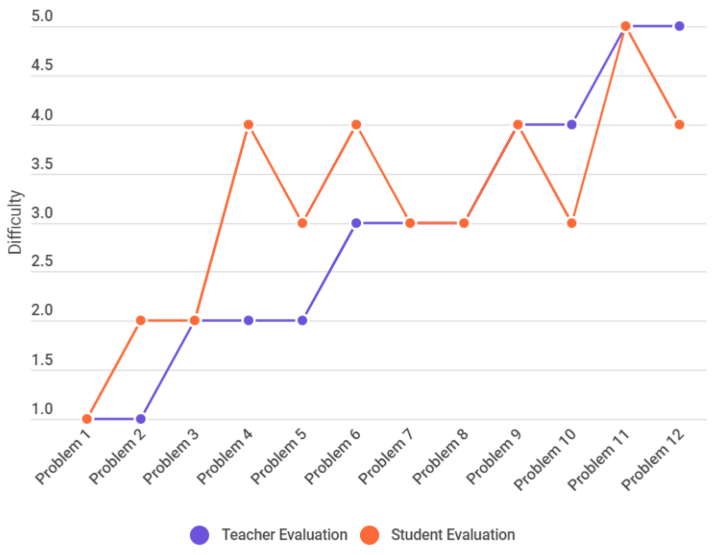

Fig. 2. Problem difficulty graph.

exercises 10 and 12 were rated as being easier than the assessment proposed by the mentor.

One of the reasons that explain this difference may be the student's familiarity with a certain problem, which is a mathematical basis provided by the school. The subject may also be more recent in the student's life.

Finally, the third step was applied in order to recognize the student's profile and understand what types of problems he developed a greater interest in solving. For this, the same 12 previous problems were passed and the student should evaluate from 1 to 5 which exercises he most liked to develop. The figure 3 shows the student's assessment.

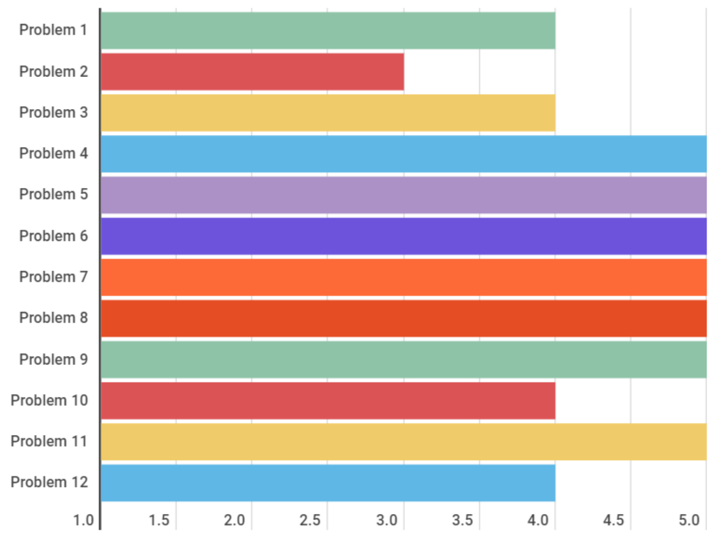

Fig. 3. Student interest rating chart.

Analyzing the figure 3 we can say that seven exercises were interesting to be developed by the student. Another four exercises were rated ”4.0”, which also inspiring his interest. The only question on which the student was less dissatisfied was in problem 2, where he assessed it as a grade of ”3.0”. This may have occurred due to the repetition around the problem in order to find the solution. This problem was assessed by the teacher with the least degree of difficulty. Due to the lack of challenges, the student then assessed the problem with a grade below the others. It should be noted that even with only a single exercise with a score of ”3.0”, it was still above half of the possible assessment. In this way, we can conclude that the student aroused an interest in programming and problem solving.

We can see that the problems proposed to the student were solved with an efficiency of 84%. In order to confirm this percentage, an assessment of the teacher and the student was made for 12 selected questions. As the results obtained approached the evaluations, it can be consolidated that the student obtained an understanding of the proposal of teaching computational thinking using programming. It can also be said, through the third stage evaluation, that the student developed an interest in solving these problems and learning more about programming. These results show that our method is promising and contagious, where the barriers of attention deficit have been resolved and a strong indicator of learning has been shown.

V. Impact on Student Life

In order to understand the real impact of learning on the student's life, a questionnaire [2] was applied with questions distributed in three dimensions: (i) applied methodology; (ii) experiences in relation to the content and; (iii) future expectations. The questions are recorded in the table II, containing numerical identification and description.

The form consisted of questions with options for graded answers from 1 to 5, multiple choice and open questions to value the student's opinions and suggestions. It is worth emphasizing the seriousness with which the student participated in the interview, considering that even questions to mark were justified, even if not asked.

The questionnaire was applied 12 weeks after the end of classes, waiting for the consolidation of knowledge and reflection on the experience. The answers provided were very positive and allowed us to better understand the difficulties and interests that interfere in the learning process. The answers will be described below.

- **Q1:** *I believe so. In math and chemistry I did better. In the part of chemistry that involved more logic, my grade went up. In math it helped a lot.*
- **Q2:** *Graphic design, architecture, chemistry or science in general. Almost everything. Because most of them use logic or a computer, and knowing how it works is good. Like graphic design, where most of it is computing.*
- **Q3:** *Let's talk about less common cases than computing. For example, a teacher, in which he can build a program to help him when giving grades, in addition to logic, of course. And you can also develop a program as needed.*

TABLE II
QUESTIONNAIRE APPLIED TO STUDENT

Number	Question Description
Q1	Do you believe that computational thinking, through programming classes, helped you in school activities? If so, give some examples.
Q2	What types of university courses do you believe computational thinking or programming can be useful in?
Q3	In what other areas or professions is the knowledge learned used?
Q4	To what extent do you think paired programming helped you learn? ( )5 ( )4 ( )3 ( )2 ( )1
Q5	To what extent has your expectation of learning about computational thinking been met? ( )5 ( )4 ( )3 ( )2 ( )1
Q6	To what extent are you interested in more advanced subjects about computational thinking? ( )5 ( )4 ( )3 ( )2 ( )1
Q7	What characteristics of computational thinking have made you find the best solution to a problem?
Q8	What kind of exercise did you most enjoy doing? ( ) With conditional structures ( ) With repetition structures ( ) With homogeneous data set ( ) With arithmetic operations
Q9	Can learning about computational thinking influence your choice for an university course or your professional choice? If so, what is your current interest?
Q10	Do you believe that other young people may be interested in computational thinking? What are the clues for your opinion?
Q11	To what extent do you believe that you have a competitive advantage for the knowledge acquired? ( )5 ( )4 ( )3 ( )2 ( )1
Q12	Check and tick which other topics you would like to learn: ( ) Autonomous things (robotics, drones, cars) ( ) Smart environments (homes, cities, schools, universities) ( ) Cloud computing ( ) Game development ( ) Immersive experiences (virtual reality, augmented or mixed reality) ( ) Artificial intelligence ( ) Others
Q13	Are your choices in the previous question related to the computational thinking course you took?
Q14	What do you suggest to make the next classes more attractive?

For games it is also important to use programming, like Minecraft for example.

- **Q4:** *(X)5 ( )4 ( )3 ( )2 ( )1*
I think it works better. If it were a class format with several students, there would always be someone who wouldn't know and it would be more difficult for everyone to learn. It is always good to have someone who already knows how to help.
- **Q5:** *( )5 (X)4 ( )3 ( )2 ( )1*
I was imagining that it would be creating programs that are real-world problems, as we do. I did not expect them to be passed in the form of challenges, as you do. I also thought that it would be something more specific to the content, that you would start from something more advanced and not from scratch.
- **Q6:** *( )5 (X)4 ( )3 ( )2 ( )1*
I am not going to give it a 5 because I have other interests besides computing. However, it is my main focus of work, because any place in the world where I want to go has a job vacancy related to that. I really want to go to England or the USA. The other interests are architecture or car design.
- **Q7:** *Plan before any action. Start basic ideas in my head to make it more complex later. Create more than one path to a solution.*
- **Q8:** *(X) With conditional structures*
(X) With repetition structures
FOR and IF was a lot of fun! In the case of exercises using them, the ones I liked the most are the ones that have real use because it is easier to imagine a situation to find an answer.
- **Q9:** *Yes. I intend to give more interest in this area. I basically got an idea of what I'm going to do and follow if I go to that area. My current interests are architecture, car design and computing.*
- **Q10:** *Definitely! For those who are interested in having a job in the future, almost certainly they will be interested in it, because those who know this will have a greater advantage in the future. Because it will be used more. It is like knowing how to speak another language.*
- **Q11:** *( )5 ( )4 (X)3 ( )2 ( )1*
It depends. If the idea is to pass the entrance exam, it will help me little. But if I go into computing, I will have a little advantage.
- **Q12:** *(X) Autonomous things (robotics, drones, cars)*
(X) Smart environments (homes, cities, schools, universities)
(X) Game development
(X) Immersive experiences (virtual reality, augmented or mixed reality)
(X) Artificial intelligence
(X) Others: Anything related to cars.
- **Q13:** *I think not. I was interested before. The course helps me to get closer to that.*
- **Q14:** *I believe that more exercises that can be applied in real life and more exercises involving vehicles.*

After analyzing the answers offered by the student, we highlight points where we consider important for this work and which correspond to the impact on the student's life, in addition to the effectiveness of the proposed method. The responses of *Q4, Q5, Q6 and Q8* are related to the applied methodology (i).

In the question *Q4* the student points out that the programming in pairs was efficient for his learning. When there was any difficulty related to the programming or the elaboration of the beginning of the algorithm, the student emphasized the importance of someone with more experience to assist him in the exercise. On the other hand, the student recognized that applying the programming in pairs in a format with several students in the same class, would be complicated.

In *Q5* the student explains that he was surprised by the challenge-based learning. He believed that we would use real-world problems and try to translate into computational language. Finally, he confirmed that he expected classes to start in a more specific way, not that he would teach from the principles of computing.

The questions *Q6* and *Q8* show that the computational thinking learned in class aroused an interest in more advanced subjects in the student. The justification for this is due to the types of exercises you enjoyed doing. For him, these problems allow a better way to find the answer and think of a solution.

Some experiences regarding the content (ii) can be found in the questions *Q1*, *Q2*, *Q3*, *Q7* and *Q11*. In the first question *Q1* the student affirms that the programming classes had an impact on the daily life of high school. Disciplines such as mathematics and chemistry, which are part of STEM, have improved in terms of logic [7][17]. The influence was so great that the student's grades in these subjects rose.

In questions *Q2* and *Q3*, we can verify that the student has acquired a wide knowledge of the dimension in which computational thinking can be useful in other areas. He also says he helped him with online games, which is part of his entertainment.

The Question *Q7* is the clearest example of how your experiences with the content have been useful in the student's life. When asked, he said that now whenever he encounters a problem, he plans before taking an action thinking about an initial solution and improving his thinking to something more complex, creating several possibilities.

In the question *Q10* the student understands that other young people can definitely be interested in computational thinking. He justifies that the technological labor market is on the rise and that anyone planning to enter this market will be interested in computational thinking.

With the student's answers to questions about future expectations (iii), it is possible to see what the student expects from his future and from this learning. In question *Q9*, for example, he says he intends to continue with the interest inspiring in the area, considering that he understood the range of possibilities if he chooses to continue in that area.

When asked about the question *Q11*, the student justifies that the competitive advantage for the knowledge acquired, will depend on what he chooses for his future. If he chooses to give continuity to computing, he will have a small advantage over other students. On the other hand, he thinks that if he chooses another choice, it will help little.

In the question *Q12*, the student must indicate which other subjects he would like to learn. He said he would like to learn more about subjects involving computing and any subject related to cars. In order to consolidate the answers to this question, in *Q13*, he states that the classes did not interfere with the answers given and that interest in these themes was prior to the course.

Finally, in the question *Q14*, the student was asked what suggestions he would give to make the classes more attractive. He replied that continuing with applications involving the real world can be a huge draw. As a personal interest, he said that exercises involving vehicles would be more attractive to him.

With the analysis of the responses we can trace the student's profile. The profile of this high school student is directed to his interest in technology and computer learning. We can highlight other interests such as cars and games. As far as we can extract, the student will continue with his work focused on technology in general and this includes a good principle of computational thinking.

VI. Conclusion and Future Work

In this paper, a method of teaching computational thinking through algorithms was proposed to assist in the learning process of a student with attention deficit. This method presents a series of steps that can be adapted according to the student's development.

This work contributed to the spread of the teaching of computational thinking in basic education by discussing relevant factors of this thinking, such as abstraction, decomposition, pattern recognition, algorithm development, generalization and parallelism. In addition to aligning itself with the Brazilian Computer Society [6] that proposes this teaching through a law of basic guidelines. On the other hand, the most important contribution of our work was a study of the application of teaching techniques to an individual with attention deficit, considering the scarce literature found in the area.

We proposed a method that, at the end of the experiments, the student said was didactic and dynamic, because each class presented a new challenge for each exercise. After the experience and the analysis of the results, it was possible to observe that the student had a good percentage of correct answers in the exercises, suggesting its effectiveness and indicating that the method is promising and needs to be used more frequently, since the result was satisfactory.

The proposed method faced challenges, the first of which was to break the difficulty of solving the student's exercise in the first minutes of all meetings. The use of a methodology in pairs helped to overcome this barrier, since the teacher started solving the exercise and the student finished. This occurred until the student obtained a certain independence in the face of the proposed challenges.

Another challenge faced was to maintain student attendance, even when the routine school commitments prevented him from participating in classes. For the lack of time of the student we always overcome with the rescheduling or replacement of classes. Despite not obtaining a continuous flow of classes, it was still possible to extract strongly positive results.

As future work, we suggest that the application of this method be carried out at other times so that we can affirm our results and effectiveness.

Acknowledgment

We thank the Brazilian Agencies for supporting Research: CAPES - Coordination for the Improvement of Higher Education Personnel and CNPq - National Council for Scientific and Technological Development for financial support. We would also like to thank the State University of Maringa for the opportunity to develop this project.

References

[1] A. C. T. Klock, et al. ”Integration of learning analytics techniques and gamification: An experimental study.” 2018 IEEE 18th International Conference on Advanced Learning Technologies (ICALT). IEEE, 2018.

[2] A. Lamprou, and A. Repenning. ”Teaching how to teach computational thinking.” Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education. 2018.

[3] A. L. S. Araujo, et al. ”Exploring computational thinking assessment in introductory programming courses.” 2017 IEEE Frontiers in Education Conference (FIE). IEEE, 2017.

[4] A. Raabe, A. F. Zorzo, and P. Blikstein. ”Computação na Educação Básica: Fundamentos e Experiências.” Penso Editora.

[5] A. Yadav, et al. ”Computational thinking in teacher education.” Emerging research, practice, and policy on computational thinking. Springer, Cham, 2017. 205-220.

[6] ”DCEB - Diretrizes para ensino de Computação na Educação Básica”, Brazilian Computer Society, 2020, access Date: 05 January, 2020. [Online]. Available: https://www.sbc.org.br/educacao/diretrizes-para-ensino-de-computacao-na-educacao-basica

[7] D. Weintrop, et al. ”Defining computational thinking for mathematics and science classrooms.” Journal of Science Education and Technology 25.1 (2016): 127-147.

[8] F. Gossen, et al. ”Computational thinking: learning by doing with the Cinco adventure game tool.” 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC). Vol. 1. IEEE, 2018.

[9] I. Corradini, M. Lodi, and E. Nardelli. ”Computational Thinking in Italian Schools: Quantitative Data and Teachers’ Sentiment Analysis after Two Years of” Programma il Futuro”.” Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education. 2017.

[10] J. M. Wing. ”Computational thinking.” Communications of the ACM 49.3 (2006): 33-35.

[11] K. Eiselt, and P. Carter. ”Integrating Social Skills Practice with Computer Programming for Students on the Autism Spectrum.” 2018 IEEE Frontiers in Education Conference (FIE). IEEE, 2018.

[12] M. Quinson, and G. Oster. ”A Teaching System to Learn Programming: the Programmer’s Learning Machine.” Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education. 2015.

[13] “Pascalzim: A pascal compiler,” University of Brasilia, 2020, access Date: 05 January, 2020. [Online]. Available: http://pascalzimbr.blogspot.com.br/

[14] P. J. Denning. ”The profession of IT Beyond computational thinking.” Communications of the ACM 52.6 (2009): 28-30.

[15] P. Sengupta, et al. ”Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework.” Education and Information Technologies 18.2 (2013): 351-380.

[16] R. A. Barkley, and L. S. Roizman. ”Transtorno de déficit de atenção/hiperatividade (TDAH).” Artmed, 2002.

[17] R. S. Rodrigues, W. L. Andrade, and L. M. S. Campos. ”Can Computational Thinking help me? A quantitative study of its effects on education.” 2016 IEEE Frontiers in Education Conference (FIE). IEEE, 2016.

[18] V. Barr, and C. Stephenson. ”Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community?.” Acm Inroads 2.1 (2011): 48-54.

[19] V. Fonseca. ”Papel das funções cognitivas, conativas e executivas na aprendizagem: uma abordagem neuropsicopedagógica.” Revista Psicopedagogia 31.96 (2014): 236-253.

[20] Y. Li. ”Teaching programming based on Computational Thinking.” 2016 IEEE Frontiers in Education Conference (FIE). IEEE, 2016.