v. 4, n. 3, 2024

Journal of Engineering Research

COMPUTATIONAL THINKING IN EDUCATION USE OF SCRATCH IN TEACHING GEOMETRY 2

Gustavo Felipe Meyer Philippsen
http://lattes.cnpq.br/2918361208326304

Sarah Kaefer da Silva
http://lattes.cnpq.br/2600501639398981

Diogo Schropfer
http://lattes.cnpq.br/4537541507705069

Mariele Josiane Fuchs
http://lattes.cnpq.br/9128499961793683

Julhane Alice Thomas Schulz
http://lattes.cnpq.br/6664398749484422

Franciele Meinerz Forigo
http://lattes.cnpq.br/5388223286828360

INTRODUCTION

The advancement of technology provides new ways of facing and solving problems, in which, through the use of computers, it is possible to achieve accurate results much faster. Thus, with the advent of digital technologies, other possibilities for organizing, reasoning, expressing and communicating ideas open up to society (LEVY, 2011) and, consequently, to Education.

In this sense, understanding computational thinking tends to enable the finding of solutions following a pre-established sequence through its four pillars: decomposition, abstraction, pattern recognition and the development of algorithms, making the resolution of problems simpler. a problem. Computational thinking, in addition to creativity, requires a critical and strategic position within the fundamentals of computing, which can be applied in the most diverse areas of knowledge, always seeking to facilitate the resolution of simple or complex problems through a sequence of steps that can be executed by people or by a machine.

In Mathematics, a subject in which most students have difficulty, the development of computational thinking since elementary school has become one of the skills foreseen by the BNCC (BRASIL, 2018). This is because it involves skills such as understanding, definition, analysis, comparison, use of algorithms and other fundamental elements for mathematical literacy. Thus, teaching mathematics through computer programming can become a powerful tool for developing these skills in students.

With the aim of enhancing the teaching and learning of mathematical knowledge focused mainly on Geometry concepts and contributing to the training of teachers and academics in Mathematics, the proposal for a mini-course is presented. This mini-course will make use of the Scratch programming language, exploring through the construction of geometric figures and the calculation of their areas, the development of computational thinking and the creation of algorithms in the *software*.

COMPUTATIONAL THINKING AND SCRATCH

Computational Thinking is closely linked to technology. But contrary to what it seems, it is not just related to computers, it is a way of solving everyday problems more easily and effectively with the use, or not, of technological tools. Using Computational Thinking to solve challenges consists of dividing it into four pillars (parts) to facilitate this process and achieve the objectives.

The first pillar is **decomposition,** where the problem must be divided into smaller parts to promote understanding. Next comes **abstraction,** which means leaving aside what is not important and focusing on what is paramount. The third pillar is **pattern recognition,** recognizing what is repeating itself in the problem, what points are similar. And finally, the fourth pillar is the **algorithm** that proposes a step by step, an order, for solving the problem.

The use of Computational Thinking in teaching develops autonomy, creativity, the construction and improvement of strategies, interdisciplinarity and also mathematical literacy in students. For Barcelos and Silveira (2012), Computational Thinking is related to several areas of Mathematics, such as explanatory and representative models, symbols and codes, establishing relationships and identifying regularities. But one big characteristic they have in common is problem solving. It is possible to notice this similarity, because

> [...] computational thinking is characterized as a problem-solving method that can assist in the development of analytical competence, through computational

> skills and concepts. Its relationship with mathematics can be explored directly through algorithms, data interpretation, logic and algebra, for example, helping to solve various mathematical problems (SILVA; MENEGHETTI, 2019, p. 8).

The study of Mathematics can be carried out using different *software* that are developed through Computational Thinking and that can be worked on in the classroom. In this sense, the study of Geometry on geometric shapes and solids using the GeoGebra or *Scratch software* stands out.

Teaching Mathematics through computer programming can become a powerful tool for students to develop and learn to learn, because when programming is learned it is possible to apply this learning in other areas of knowledge, thus stimulating logical reasoning and development of computational thinking, in addition to being considered a smart way to keep students' attention using digital technology.

Scratch allows its users to develop various *skills,* such as: information skills; of communication; critical reasoning and systemic thinking; identification, formulation and resolution of problems; of creativity and intellectual curiosity; collaboration; self-direction; of accountability and adaptability and social responsibility (SCRATCH, 2011).

Scratch programming is done through command blocks, with a simple, intuitive and interactive *interface* for beginner programmers. It is possible to create programs with sounds, movements, graphics, music and text, covering different contents. This way, *Scratch* is "ideal for teachers who want to build their own digital materials, including creating educational games in a simple way" (ZOPPO, 2016). In Geometry, *Scratch* enables more dynamic ways of learning angles, geometric shapes, area and perimeter calculations through the *software's Computational Thinking.*

METHODOLOGICAL PROCEDURES: SEQUENCE OF ACTIVITIES

Participants' interaction with these tools will be carried out in a practical way. Thus, it will be possible to experiment with different possibilities of use, especially those related to learning mathematical concepts in Geometry Teaching. In this sense, the mini-course will involve the construction of flat figures, such as: square, rectangle, equilateral triangle, trapezoid and rhombus. In addition, their areas will be calculated.

Scratch tools, activities will be developed in the form of a tutorial. The material will be made available to participants, containing activities and steps to resolve them.

The step-by-step activities proposed to participants of the mini-course will be described here. Since the *Scratch software may* not be known to everyone, the first moment will be intended for the first access and knowledge of the *software,* while the second moment will be for the construction of geometric figures and the calculation of their areas.

FIRST ACCESS AND PROGRAM FEATURES

Initially, participants must access the *Scratch platform* at https://scratch.mit.edu/ and register for the first time if they do not have an account. To do this, they must inform the country in which they live, date of birth, gender and an email address, as well as their username and password. The *Scratch* platform used in this mini-course is online, requiring you to be connected to the internet, without the need to download the program.

Afterwards, some characteristics, functions and possibilities of the program present in the platform creation area will be presented. The stage (Figure 1) is the place where the execution of events/projects can be seen.

In this way, it is characterized as the space where the objects are inserted and in it we can visualize the scenario, which can be chosen by carrying out a search in *Scratch itself,* drawing or *downloading* an image from the computer.

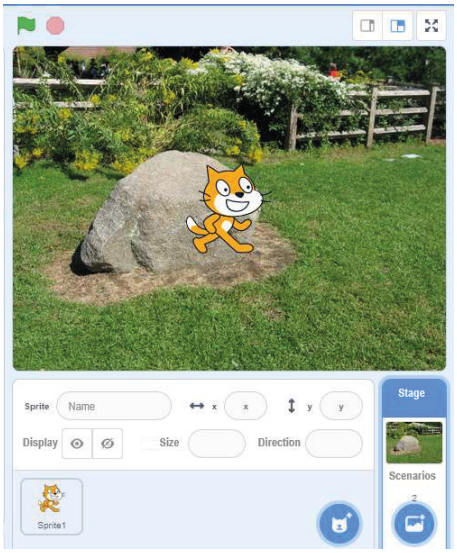

Figure 1: Stage

Source: (SCRATCH, 2011)

Actors are characters used to give action to the game being developed. They can be defined in the same way as when choosing the stage. To do this, simply click on the "select an actor" option and choose the option you prefer (Figure 2).

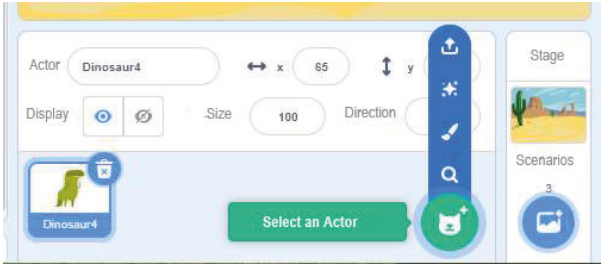

Figure 2 : Actors

Source: (SCRATCH, 2011)

Codes, costumes and sounds are command categories used for editing in *Scratch*. Each category has its own command blocks used to drive programming actions.

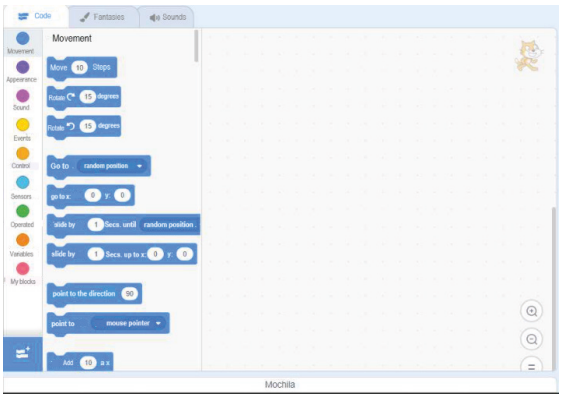

Figure 3: Codes, fantasies and sounds

Source: (SCRATCH, 2011)

Codes are the blocks that, together, create algorithms. In other words, they create a sequence of commands that will be followed by the actors. The blocks work like a puzzle, as the desired pieces fit together and the algorithm is formed. The codes are divided into nine groups, namely: Movement, Appearance, Sound, Events, Control, Sensors, Operators, Variables and My blocks (Figure 3). Groups can be identified by colors and different fitting formats.

CONSTRUCTION OF GEOMETRIC FIGURES AND AREA CALCULATION

The first geometric figure to be constructed is the square and to do this, participants must, based on decomposition and pattern recognition, analyze the figure and think of an algorithm that allows its creation.

A square is a geometric figure that has four congruent sides and internal angles equal to 90°, so it is necessary to move the character a certain distance, change its direction by 90° and travel the same distance again, thus forming the second side of the figure. It is observed that to complete the drawing of the image, the actor must carry out the movement of traveling a certain distance and turning 90° four times. This observation is about pattern recognition, an essential characteristic in computational thinking. In this way, using the

pen and the movement and control blocks, the algorithm (Figure 4) is created to represent the square.

It is noteworthy that in this activity, as well as in other activities that involve the representation of geometric figures, the blocks "events", "control", "appearance" and "pen" will be the most used. In this case, the "pen" block must be added from the "Add an extension" menu at the bottom of the control panel.

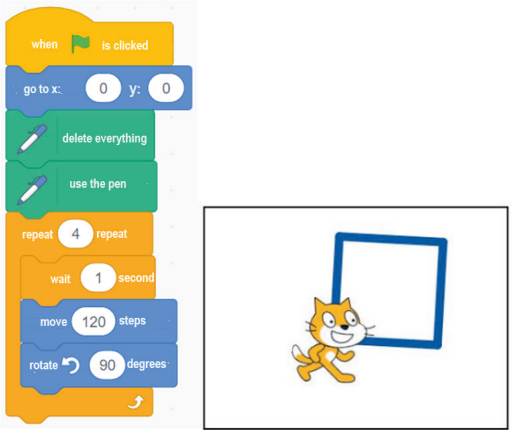

Figure 4: Algorithm for representing a square

Source: (SCRATCH, 2011)

Once the representation is done, we start to create an algorithm for calculating the area using blocks of "sensors" and "operators". To do this, you need to know that the area of a square is given by multiplying two sides. Therefore, with the sensor blocks the user can enter the value of the square's edge and the operator blocks will calculate the area.

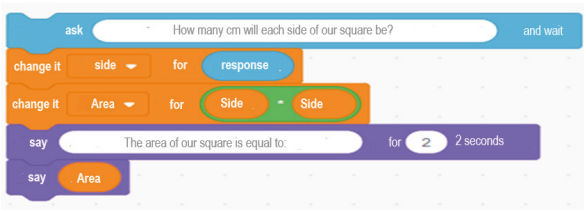

Figure 5: Algorithm for calculating the area of a square.

Source: (SCRATCH, 2011)

The next figure to be constructed is the rectangle which, like the square, has four right angles. The difference is that opposite sides must be congruent, not requiring that all sides have the same measurement. Therefore, the character must travel a certain distance, turn 90°, travel another distance different from the first and rotate 90° again to represent two sides of the rectangle. This way, just repeat this sequence again to complete the representation.

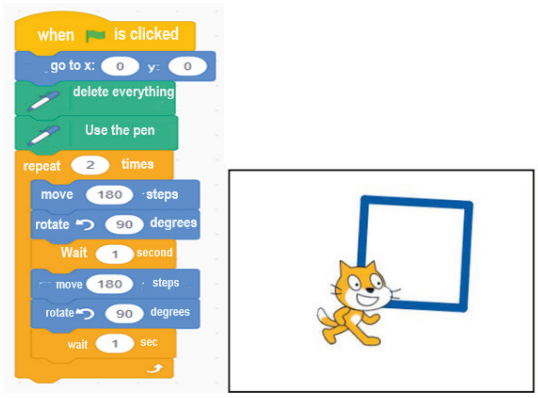

Figure 6: Algorithm for representing a rectangle.

Source: (SCRATCH, 2011)

To calculate the area, unlike the square, it is necessary to enter two measurements for the sides, since the drawn rectangle has a longer side (length) and a smaller side (width). Having both values, simply multiply between the two.

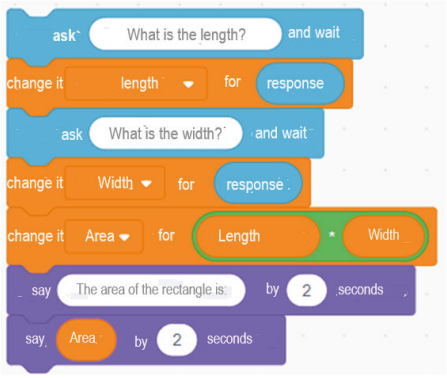

Figure 7: Algorithm for calculating the area of a rectangle.

Source: (SCRATCH, 2011)

For the next figure, we have the representation of the equilateral triangle, which must have the three congruent sides, as well as its internal angles. However, as we refer to the character when we enter the degree value, the value we must rotate will be 120º so that the representation of the equilateral triangle can be completed. This way, our character must move a certain distance, rotate 120º and repeat this same process two more times, totaling three at the end.

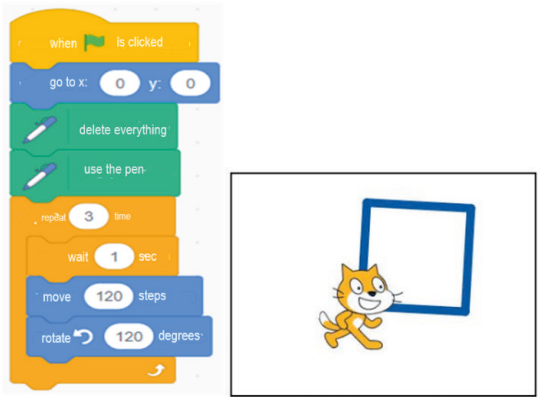

Figure 8: Algorithm for representing an equilateral triangle.

Source: (SCRATCH, 2011)

To calculate its area, we have the value of the base multiplied by the height and then divide by two. Therefore, the equation must be constructed involving two variables that must be informed by the user.

Figure 9: Algorithm for calculating the area of an equilateral triangle.

Source: (SCRATCH, 2011)

At this point, the trapezoid will be constructed and unlike the figures already constructed, the trapezoid does not have congruent sides. In this way, for its representation, we will work with the decomposition of the figure into parts, since there will be no patterns to be repeated. Therefore, the algorithm for representing the trapezoid is presented in Figure 10.

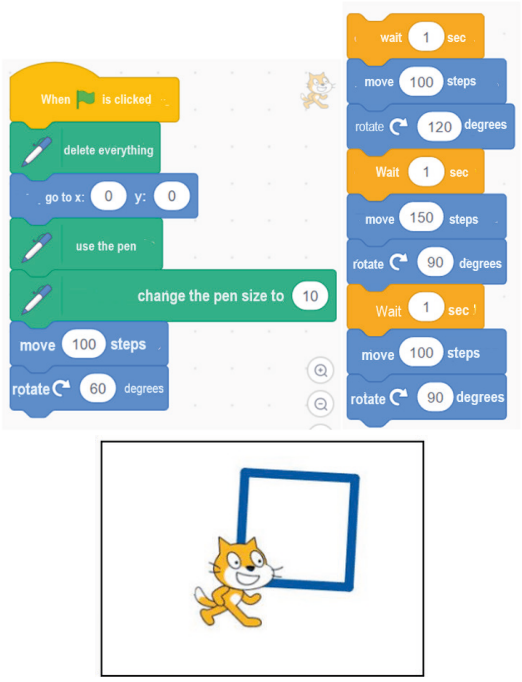

Figure 10: Algorithm for representing a trapezoid.

Source: (SCRATCH, 2011)

The calculation of the area of the trapezoid will involve three variables, namely: value of the largest base; lower base value; and trapezoid height. With this, simply add the value of the larger base with that of the smaller base, multiply by the height and then divide the result by two, similar to what was done in the representation of the triangle.

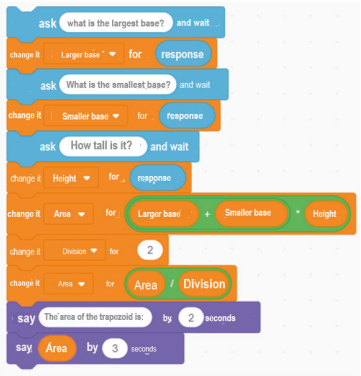

Figure 11 : Algorithm for calculating the area of a trapezoid.

Source: (SCRATCH, 2011)

The last figure to be constructed will be the rhombus. In it, participants must carry out the same process of decomposition, abstraction, pattern recognition and creation of the algorithm for representing the figure and calculating the area. Figure 12 presents the algorithm for representing the diamond.

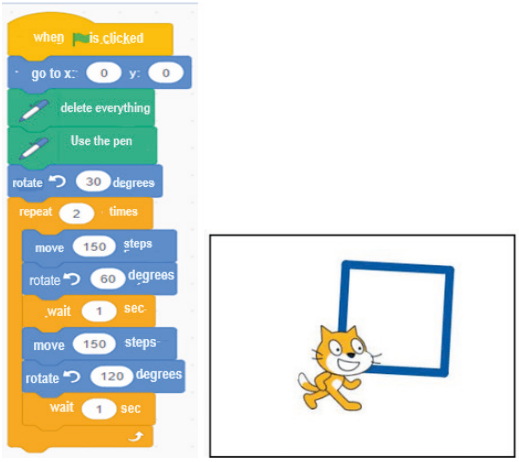

Figure 12: Algorithm for representing a diamond.

Source: (SCRATCH, 2011)

The calculation of its area will involve two variables that must be entered by the user, namely the largest diagonal and the smallest diagonal. By multiplying the smaller diagonal by the larger one and then dividing the result by 2 we will obtain the result of the area.

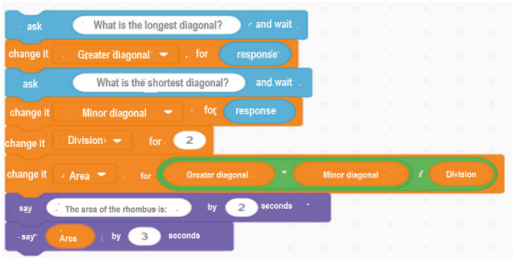

Figure 13 : Algorithm for calculating the area of a rhombus.

Source: (SCRATCH, 2011)

After completing the construction of these geometric figures and calculating their area, participants will be able to share their projects created by the *Scratch platform itself.*

CONCLUSION / RECOMMENDATIONS

The lack of continuing training for Basic Education teachers who address the use of computer programming strategies with mathematical pedagogical practices still appears timidly in the initial and continuing training of teachers. In this sense, through this mini-course it will be possible to show that the Mathematics teacher has alternatives to teach Geometry in a more dynamic way using the Scratch *software,* such as in the construction of geometric figures and the calculation of area using computational language.

Based on this assumption, it is clear that it is important that teachers and undergraduates are trained and prepared to integrate Scratch *effectively* into their classes, exploring the potential of this tool for teaching and learning Geometry, enabling Mathematics classes to become more meaningful and enjoyable for the student.

This mini-course aims to contribute to the continued training of Mathematics teachers regarding the didactic use of the *Scratch platform* in their classes, realizing the benefits it provides. By proposing essential activities for understanding geometric concepts, in a

dynamic and interactive way, the teacher has the opportunity to put students in contact with programming logic, developing computational thinking. Therefore, it is understood that this mini-course is the initial step for participants to learn about and interact with the possibilities of the *Scratch environment,* which can be deepened according to each person's interest.

It is clear that digital games and programming strategies are increasingly present in students' daily lives, and it is important that teachers use digital technologies to spark interest and motivation in Mathematics classes. In this sense, *Scratch* can contribute to training in which the student is placed as a protagonist in the construction of their knowledge.

REFERENCES

BARCELOS, Thiago Schumacher; SILVEIRA, Ismar Frango. **Pensamento Computacional e Educação Matemática**: Relações para o Ensino de Computação na Educação Básica. In: WORKSHOP SOBRE EDUCAÇÃO EM COMPUTAÇÃO, 2012, Curitiba. Anais. Curitiba: UFPR, 2012. Disponível em: < http://www2.sbc.org.br/csbc2012/anais_csbc/eventos/wei/artigos/Pensamento%20Computacional%20e%20Educacao%20Matematica%20Relacoes%20para%20o%20Ensino%20de%20Computacao%20na%20Educacao%20Basica.pdf>. Acesso em: 15 jun. 2023.

BRASIL. Ministério da Educação. **Base Nacional Comum Curricular**. Brasília, 2018.

BRASIL. Secretaria de Educação Fundamental. **Parâmetros curriculares nacionais**: Matemática/ Secretaria de Educação Fundamental. Brasília: MEC/SEF, 1998.

LEVY, Pierry. **Tecnologias da Inteligência**: o futuro do pensamento na era da informática. Rio de Janeiro: Ed. 34, 2011.

SCRATCH**.** Site oficial. 2011. Disponível em: <http://scratch.mit.edu>. Acesso em: 10 jun. 2023.

SILVA, Fernanda Martins da Silva; MENEGHETTI, Renata Cristina Geromel. **Matemática e o Pensamento Computacional**: uma análise na pesquisa brasileira. In: XIII Encontro Nacional de Educação Matemática (ENEM), 2019, Curitiba.

ZOPPO, Beatriz Maria. **O uso do Scratch no ensino da matemática.** Disponível em: <http://www.ebrapem2016.ufpr.br/wp-content/uploads/2016/04/gd6_beatriz_zoppo.pdf> Acesso em: 11 jun. 2023.