SCRATCH AND COMPUTATIONAL THINKING IN HIGHER EDUCATION: A REPORT OF PRACTICE WITH BIOLOGY AND COMPUTING STUDENTS

https://doi.org/10.56238/sevened2025.018-023

Vanessa Stangherlin Machado Paixão-Cortes[1], Cristiane Ellwanger[2] and Regina Barwaldt[3]

ABSTRACT
The training of professionals in Biological Sciences requires the use of computational methods to solve complex problems, which requires an interdisciplinary approach. In this context, computational thinking can contribute significantly to the development of competencies and skills, although there is still a lack of its approach in undergraduate studies. This study reports the experience of an Action Project at School (PAE), carried out in the course (hidden for submission), with the objective of investigating whether programming with games and digital animations focused on programming logic can act as an engagement factor for students of Biological Sciences or if it remains as a privilege of the exact and technological areas. 12 students participated, seven from the biological area and five from the exact sciences, aged between 18 and 30 years old and with varied experiences in programming. The activities involved the development of games and animations in the Scratch language, based on the ARCS motivational model, which guided the analysis from the categories of Attention, Relevance, Trust and Satisfaction. The results indicated that computational thinking can act as a motivational strategy and represent a viable alternative for engagement in teaching-learning in higher education, both for beginners in computing and for students of Biological Sciences.

Keywords: Computational thinking. Scratch programming language. Biological Sciences. Higher education.

[1]Federal University of Rio Grande
Dr. in Computer Science
vanessapaixaocortes@gmail.com
[2]Univ. Fed. do Rio Grande do Sul - UFRGS - Porto Alegre - RS
Dr. in Design
[3]Federal University of Rio Grande
Dr. in Informatics in Education

INTRODUCTION

Teaching programming is not an exclusive practice applied to Computer Science and Engineering courses (Zanetti, Borges, & Ricarte, 2016). In this same context, Denning (2018) corroborates that computing has proven to be productive for scientific advancement in practically all fields of knowledge. For example, the most recent advances in the biological sciences have involved modeling, sequencing, and editing DNA[4] using computational methods and programming.

Currently, there is a need for students in higher education in Biological Sciences to also learn to program and develop their own *software* to solve biological and health problems (Salazar, 2018), and the ability to employ scientific methods in computing and apply methodologies in Software Engineering is essential to successfully design, implement and maintain *software* in scientific settings (Welch, Schwartz, & Lewitter, 2011).

In order to prepare students to face the biological challenges of the twenty-first century, cultivate their interests in scientific careers, and improve their quantitative and critical thinking skills, it is essential to stimulate computational thinking (Qin, 2009). However, studies indicate that despite the interest in working on computational thinking at the elementary and secondary levels, in areas other than Computer Science, there is less effort in research to verify its potential in higher education (Czerkawski & Lyman, 2015). Furthermore, computer training at a higher level in Biological Sciences does not meet the demand for computational knowledge and skills necessary for the effective practice of scientific research in their careers (Salazar, 2018). Teaching focused only on digital literacy is no longer enough, as it is necessary to teach digital proficiency, including programming (Wangenheim, Nunes, & Santos, 2014).

It is in this context that this article presents and discusses the practical experiences of an Action at School Project (PAE), in a postgraduate course (hidden for submission) at the specialization level. The main objective of the PAE was to investigate whether undergraduate students, especially biologists, feel engaged to learn these concepts through games and animations, using Scratch[5]. The sample consisted of twelve students, six participants from the area of Biological Sciences and five students from the areas of Exact Sciences.

The theme for the application of the PAE emerged from the learning obtained in the course (hidden for submission) about the transformations and technological innovations experienced in the new digital society and the educational progress arising from this

[4] DNA
[5] https://scratch.mit.edu/

evolution, which motivated the authors to rethink their pedagogical practices in the classroom, to reflect on the contributions provided by the use of digital technologies in the academic environment and to identify the possibilities of educational changes that could be evidenced in the application of the PAE in undergraduate studies.

The PAE methodology involved data collection obtained through questionnaires and direct observation, where the ARCS (Attention, Relevance, Confidence, Satisfaction) model proposed by Keller (2009) was used to guide the development of the tasks, the design of the instruments and carry out the analysis, which was supported by quantitative and qualitative approaches. As a result, it was realized that Scratch can provide engaging experiences for the development of computational thinking through games and interactive animations for beginner students or in areas such as Biological Sciences. During the analysis, it was noticed that most participants felt motivated, including those who already knew programming languages more deeply.

THEORETICAL FOUNDATION

Currently, biologists need to experience computational practices to perform their analyses. Although implementing your own programs is not a trivial activity and depends on your logical ability and understanding of algorithms and programming languages. Individuals not only have the ability to develop their own systems, but also reinforce adjacent skills, such as abstract thinking (using different levels of abstraction to understand problems and, step by step, solve them), algorithmic thinking (expressing solutions in different steps in order to find the most effective and efficient way to solve a problem), logical thinking (formulation and exclusion of hypotheses) and scalable thinking (decomposition of a large problem into small parts or composition of small parts to formulate a more complex solution) (Phillips, 2009; Resnick, 2008, *apud* Sousa & Lencastre, 2014, p.261).

Authors such as (Czerkawski & Lyman, 2015; Qin, 2009), still approach computational thinking as a way of thinking that uses computing concepts and methodologies, such as the development of logic, algorithmic thinking, recursive thinking, abstraction, parallel thinking, pattern matching, and related processes, to address issues in a wide range of subjects and, as such, offers an important set of skills in the modern sciences.

According to Marques *et al*. (2011) teaching programming from an early age is important because students develop problem-solving and formalization skills, which are useful in their respective areas of knowledge. In this perspective of technological applications, the potential of educational games and animations to motivate and dynamize the teaching and learning process in different contexts is highlighted, as they allow a practical exploration of concepts, playful and motivating learning and serve as a complement to theoretical concepts (Soares & Rodrigues, 2016).

The animations, made from simulations of experiments, stories and interactive activities, make the process of conceptual understanding by students more accessible, which are motivational and stimulating factors in the search for knowledge (Castilho & Ricci, 2006).

Games are efficient instruments because they entertain while motivating, facilitating learning, and increasing the capacity to retain what has been taught, exercising the mental and intellectual functions of students, which reveal autonomy, creativity, originality, and the possibility of simulating and experimenting with situations (Tarouco, Roland, Fabre, & Konrath, 2004). In addition, games are part of the students' daily lives, as highlighted by Almeida e Silva (2011, *p.3*):

> Such technologies have become part of culture, taking place in social practices and resignifying educational relations [...] Among the technological artifacts typical of the current digital culture, with which students interact even outside the school spaces, are electronic games, which instigate immersion in a visual aesthetic of digital culture; the characteristic tools of Web 2.0, such as social media presented in different interfaces; mobile devices, such as cell phones and laptops, which allow access to virtual environments in different spaces and times.

Thus, games and animations favor the creation of more engaging learning experiences, as well as encourage practice and value effort and error as part of learning, important aspects for teaching programming (Gomes, Tedesco, & Melo, 2016).

Finally, for this study, engagement was considered as an area that comprises aspects related to fidelity in the fluency of interactions and motivation of users with the system in general, presenting aspects that objectively establish user involvement through factors such as attention, positive affect, novelty of experiences, control, expectation and interest of the user (Herpich & Tarouco, 2016). Also, engagement is related to satisfaction, attention, confidence, relevance, as well as the way the student successfully performs a task and how this success affects their expectations for the future (Keller, 2009).

RELATED WORKS

In order to provide a view on the research that is being developed within this theme, below are described works that involve the teaching of computational thinking and programming at various educational levels, to then present studies in the scope of Biological Sciences.

Zanetti *et al.* (2016) states that it is possible to find programming teaching practices within elementary and high schools. This author conducted a systematic review of the literature on Computational Thinking in programming teaching, from studies published in Brazilian events from 2012 to 2015. The review was carried out in the annals of important national events: Brazilian Symposium on Informatics in Education (SBIE), Workshop on Informatics at School (WIE) and Workshop on Informatics Education (WEI), Workshop on Teaching in Computational Thinking, Algorithms and Programming (WAlgProg). The author analyzed 16 articles and identified that "unplugged" computing, digital games, programming languages, visual programming language and pedagogical robotics are used, and that some studies use these practices together, and that skills such as data collection, analysis and representation, problem decomposition, abstraction and algorithm development are developed. The tool commonly used in the studies was Scratch because it is widespread in the academic environment and presents positive results for learning and the use of digital games appears as an instrument of motivation to students, due to its playful, challenging and motivating character. Finally, the author mentions that there is a tendency to carry out research in the context of secondary/technical education (Zanetti *et al*., 2016).

In this sense, the studies by Wangenheim *et al*. (2014) and Alves *et al*. (2016) investigated the context of computational thinking at the elementary school level with Scratch. Both researches proposed a multidisciplinary instructional unit for the teaching of computing, inserted in the discipline of History and Social Studies. The instructional units were applied and evaluated in case studies, which involved classes of the first, fifth and seventh grades, in a school in Florianópolis/SC. The implications observed indicate that the instructional unit and the use of Scratch arouse the interest and motivation of students for the area of computing, as well as allow the learning of basic programming concepts, in an effective and fun way by elementary school students (Alves *et al*., 2016).

The studies found that it is interesting that there is a greater engagement of Biological Sciences students in learning programming. For example, the study by Qin (2009) incorporated the teaching of computing into an introductory bioinformatics course for Life Sciences undergraduates, with hands-on lab activities and collaborative peer learning. The methodology involved practices with the use of bioinformatics software, such as MEGA for phylogeny, SWISS PDB Viewer for protein structure analysis and R for gene expression analysis, and the use of Linux servers. The author brings two studies with a total of 39 students and points out that many of these students readily express their interest in improving their computational and quantitative skills, and that, mainly, they agree that there should be a better development of these skills, which are useful for their profession, however few of them actually take computer courses (Qin, 2009). The survey showed that students were positive about their learning experience and that the activities improved their computer knowledge and quantitative skills.

Salazar's work (Salazar, 2018) comprised the development of a project, related to teaching and extension, with the theme of programming and good practices of scientific computing for biologists[6]. Didactic materials for programming were elaborated, involving command line interfaces and programming in Python, accompanied by a semi-structured evaluation questionnaire, including questions about the theme and the use of free *software*. An edition of a mini-course was held, applying the instruments, for 17 participants. The reflections that arise from the results clearly point to the need for research, in the area of education, regarding the teaching of programming to students of Biological Sciences and related areas, and most of the subjects (14 out of 17 participants) agreed that contents like this are important, but mostly absent in the training of biologists in Brazil.

Among the studies that dialogue with the proposal of this article, the research by Santana (2023) stands out, who investigated how undergraduate students in Biological Sciences, at a public university in the Brazilian Northeast, were able to develop skills based on Computational Thinking (PC) through block programming with Scratch. Based on a didactic sequence with seven moments, which integrated biophysics content, PC concepts and practical activities on the platform, the participants developed projects on topics such as structures and functioning of biological membranes. The study showed that the most developed PC skill was algorithmic thinking, and concluded that the use of block programming can favor meaningful learning in interdisciplinary and highly complex content, such as those involving chemistry, physics and biology. The research also highlighted as a limitation the reduced time of the didactic sequence and suggested the creation of a specific discipline on Computational Thinking in the Teaching Degree course, reinforcing the need for teacher training focused on the integration between computing and science teaching.

METHODOLOGY

The methodological nature of the present study was an applied research on a practical and evaluative experience of a tool for teaching computational thinking with a

[6] Activities carried out at the Center for Biological Sciences of the Federal University of Santa Catarina (CCB / UFSC)

focus on undergraduate students with the realization of an Action at School Project (PAE). The proposal of the present work was based on the following premise: there is a lack, in the field of biology and related areas, of programming and scientific computing training, especially at the higher education level (Salazar, 2018).

The Scratch Language was used to implement the objects because its interface is friendly, free, simplified and allows the learning of basic syntax in programming quickly and easily (Alves *et al*., 2016).

The research objective is descriptive, as it uses data collection (questionnaire and direct observation of the use of the system). Specialized questionnaires are used to identify subjective aspects of users, such as their engagement with technology, where questions are only asked at the end of the application (Cybis, Betiol, & Faust, 2015).

And finally, to validate the work, it was taken into account that the research was of a mixed nature, but with a greater focus on qualitative data. According to Preece, Rogers and Sharp (2013), the type of analysis can be influenced by the objectives identified at the beginning of the research and by the data actually collected, and a qualitative or quantitative analysis approach can be identified, or a combination of both.

CHARACTERIZATION OF THE RESEARCH SITE, SAMPLE AND DATA COLLECTION INSTRUMENTS

The sample consisted of twelve students, six participants from the area of Biological Sciences and five students from the areas of Exact Sciences. The theme for the application of the PAE emerged from the learning obtained in the course about the transformations and technological innovations experienced in the digital society and from the educational progress arising from this evolution, who already had previous expertise in the syntax and structures that make up programming languages, such as C, Python, JAVA, JavaScript and PHP, as well as knowledge of logic to implement an algorithm.

To evaluate whether the students felt engaged with the development of games and animations through programming with Scratch, two test instruments were elaborated, questionnaires made in *Google Forms*, which were submitted in two moments: at the beginning of the project and at its conclusion. The steps taken to design the data collection are described in Table 1.

Table 1: Steps for performing data collection

Stage	Answers
Definition of the sample of participants	End users: were invited to participate in the application of the PAE of the students of the undergraduate course (hidden for submission). The sample, non-probabilistic by convenience, was composed of 12 undergraduate students from the Exact or Biological Sciences.
Definition of the itinerary	The tasks were carried out according to a pre-established script based on the application of the Project Teaching methodology, where the student had the possibility to choose a theme of interest for the development of his game/animation.
Setting the environment	The tests were applied in a computer lab of the institution where they belonged, supported by presentation media and electronic forms, on computers with internet access.
Constitution of the team of evaluators	The application of the PAE was performed by a teacher/observer, in this research considered as the author.
Definition of the technique for recording events	The records were made by recording images of the users' answers.
Pilot tests	Pre-tests , where a computer science specialist, a specialist in HCI and with experience in teaching and learning, validated the test instrument. Also, the time needed to perform the tasks was measured.
Execution of data collection tests	This stage was the application of the PAE, ethical issues were also addressed, with the Free and Informed Consent Form, and in the case of the Institution, the letter for the knowledge of the person responsible for carrying out the research.

Source: the author. Adapted from Cybis, Betiol, & Faust (2015).

For the development and analysis of the data collection instrument, a model that favors the evaluation process of educational games and animations was used as a theoretical framework. According to Soares & Rodrigues (2016), the ARCS model proposed by Keller (2009) was adequate because it focuses on the evaluation of students' interaction with learning materials and environments, being derived from the expectation theory (probability of an individual succeeding) and is based on value (user satisfaction and motivation).

In this sense, the activities proposed for the development of the PAE and questionnaires sought to raise subsidies that could be related to the motivation of the students in learning. Thus, the tasks and instruments were theoretically based on the ARCS method, as shown in Table 2. The ARCS method is divided into 4 categories, which allow an overview of the main motivational dimensions, especially in the context of motivation in learning, and how to create strategies to stimulate and sustain motivation (Keller, 2009).

Table 2: ARCS method used to guide the tasks and data collection instruments.

ARCS Method	Description	Guiding Questioning	Related Tasks
Attention	Stimulate students' interest, and curiosity to learn	How can I make this learning experience stimulating and interesting?	Application of the pre-test questionnaire Presentation of the Motivating Theme Programming activities with Scratch Development of the Final Project
Relevance	Meet the student's personal needs	How will this learning experience be valuable to my students?	Introducing the Basics Friendship Algorithm

	and goals to effect a positive attitude		Programming activities in the Scratch Language
Confidence	Motivate students to believe and feel that they will succeed	How can I via instruction help students succeed and allow them to control their success?	Programming activities in the Scratch Language Development of the Final Project
Satisfaction	Reinforce students' achievement, with positive feelings after learning experience	What can I do to help students feel good about their experience and desire to continue learning?	Development of the Final Project Application of the post-test questionnaire

Source: the author. Adapted from Keller (2009).

After data collection, observations began to identify patterns, using structured *frameworks* or theories to support the investigation (Preece *et al*., 2013), based on the ARCS method, making use of the experiences lived by the students, talking about the qualitative responses and capturing the interfaces of the projects carried out. This type of analysis was popularized from Bardin (1977) in the interviews and/or observed by the researcher.

In this same line of thought, the dismemberment of the text into analogically grouped categories is supported by a better choice when one wants to observe values, opinions, attitudes, and beliefs through qualitative data (Lidiane Vieira Pozzebon, André Ricardo Theodoro Velho, & Regina Barwaldt, 2019). This study also presents and discusses the results of quantitative responses.

Thus, the data collection served as a basis for discussion on the application of the PAE, verification of student learning and evaluation of the proposed activities to, finally, infer impacts of the action for the group of students.

DESCRIPTION OF ACTIVITIES

For the development of the PAE, two workshops were held, each lasting 3 hours, which totaled 6 hours of activities. Each of the workshops had a different audience, the first with 5 students and the second with 7 students. Both activities followed the same execution protocol.

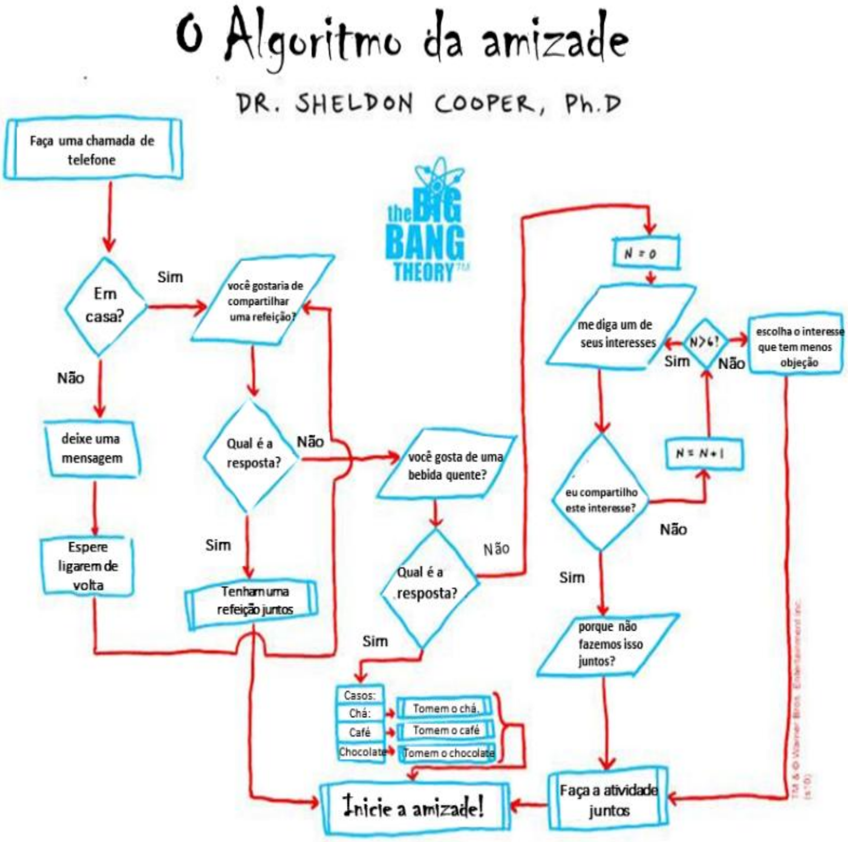
The initial stages involved the presentation of the teacher who taught the workshop and the signing of the Informed Consent Form (ICF), which guided the participants on ethical issues, research objectives and potential risks. Afterwards, the questionnaire (pre-test) was filled out about their profile and experience with programming. For student engagement, the video entitled "Why should everyone learn to program?[7]" and some

[7] https://www.youtube.com/watch?v=mHW1Hsqlp6A

current aspects of technology and the importance of computational thinking were discussed. For example, during the thematic introduction, there was a presentation of the Friendship Algorithm[8] (Figure 1).

After the initial motivation, concepts of programming logic and computational thinking were introduced necessary for the student to understand the context of the activities that would be carried out. These concepts offered an introductory and less complex approach, encompassing concepts such as abstract thinking, decomposition, pattern recognition, algorithms, and programming languages.

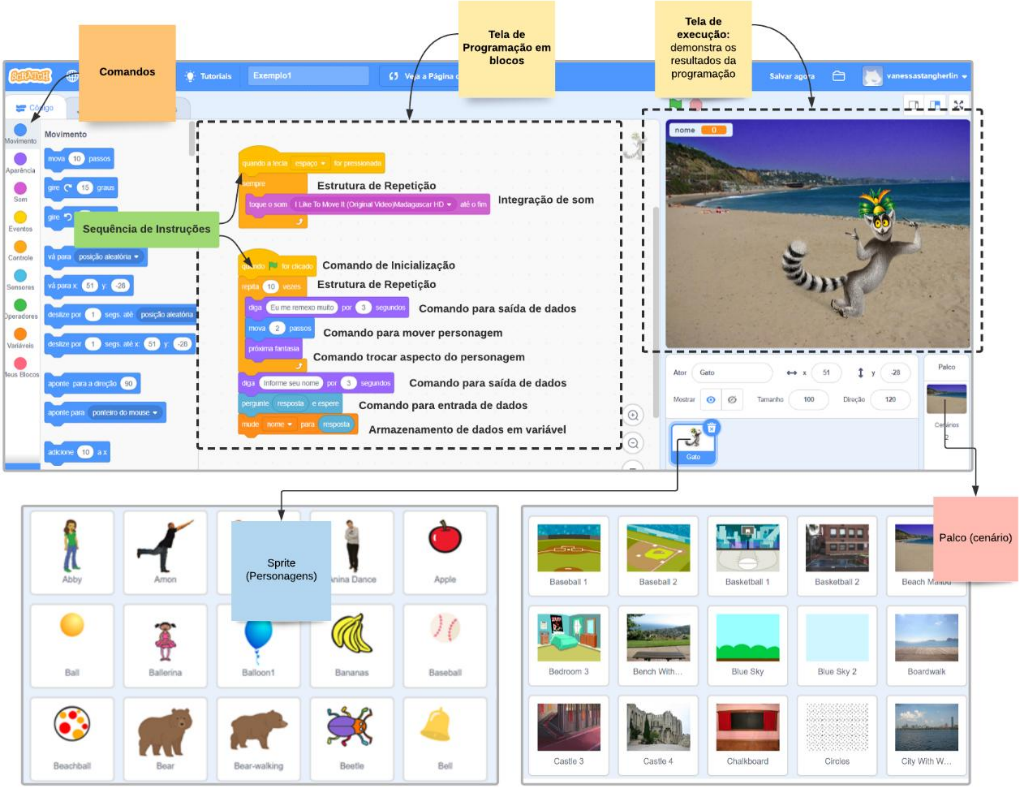
Figure 1: Friendship Algorithm. Adapted from *The Friendship algorithm*[9].



The next step was the presentation of the Scratch interface (Figure 2), with the demonstration of the commands, characters and scenarios. Languages in this format have the differential of providing programming knowledge about: (i) sequences of instructions, (ii) conditionals, (iii) loops, (iv) procedures/functions and (v) parameters (Gomes *et al*., 2016).

[8] From The Big Bang Theory series. In it, the fictional character Sheldon develops an algorithm to befriend and presents it to his friends through a block diagram (or flowchart), which is a way of writing algorithms graphically.
[9] https://link.estadao.com.br/blogs/alexandre-matias/wp-content/uploads/sites/360/2010/08/algoritmo-da-amizade.jpg

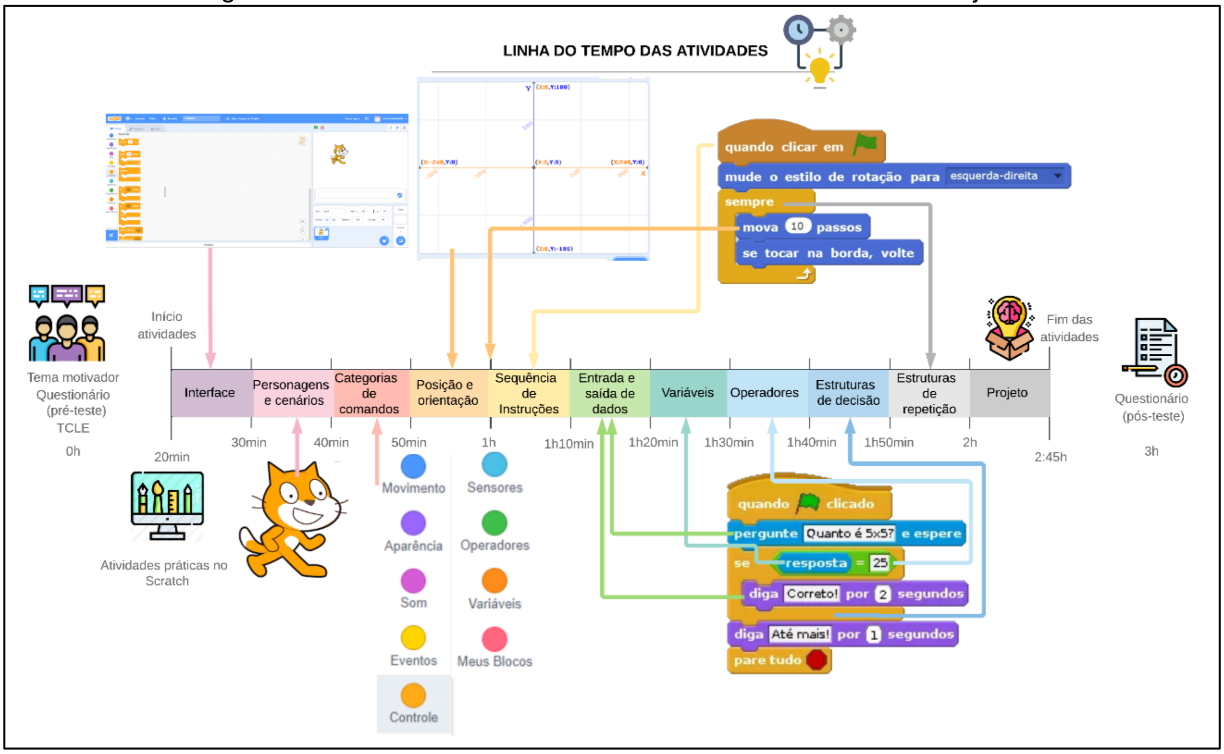
Figure 2: Scratch Interface



The students were able to explore the scratch environment by learning programming logic concepts in the following sequence, as shown in Figure 3:

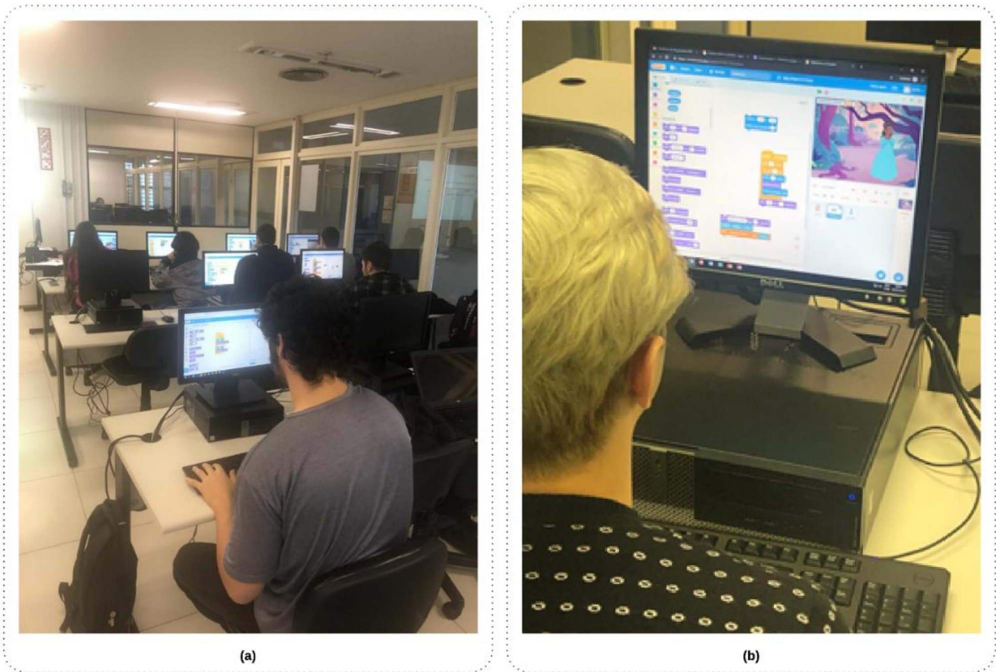
- Exploration of the interface, including opening and saving a project.
- Verification of characters and scenarios.
- Modification of the position and orientation of objects, movement by steps and by degrees, and exploration of the Cartesian plane.
- Interaction with the categories of commands: movement, appearance, sound, control, sensors, operators and variables.
- Use of blocks as a sequence of commands, including starting and stopping the algorithm.
- Employment of data input and output.
- Creation and manipulation of variables.
- Use of logical operators.
- Conceptualization and development of selection and repetition structures;

Figure 3: Timeline of activities carried out in the School Action Project.



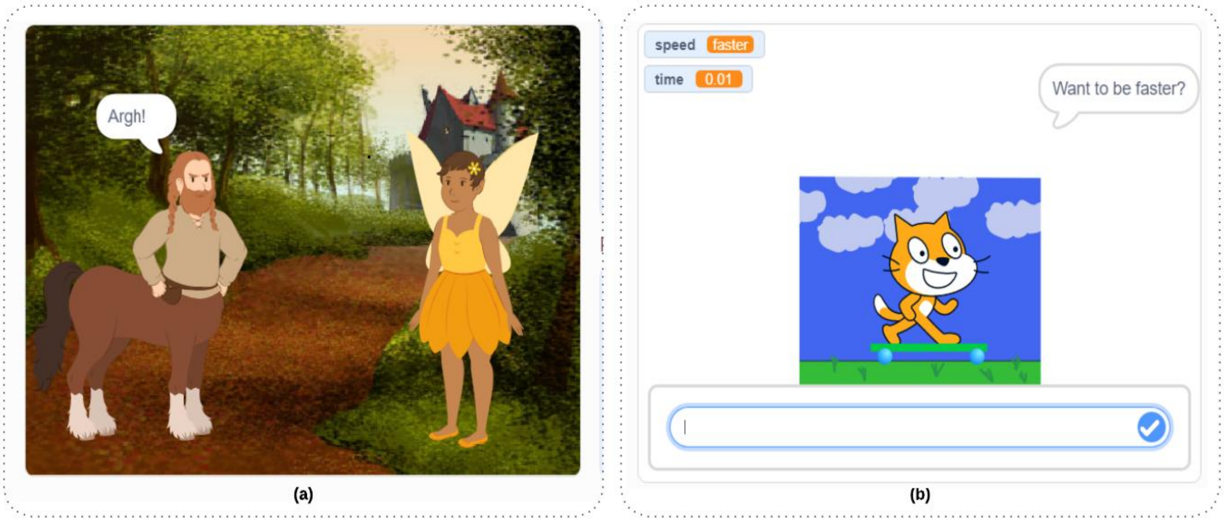
For each participant, they were asked to use the concepts learned in order to develop a project of their own, which involved an animation or game. Figure 4 shows the realization of one of the workshops of a history applied in the project.

Figure 4: - PAE application workshop. (A) Participant in one of the Workshops. (b) Project developed by a participant in the area of biological sciences.



Finally, Figure 5 illustrates two projects developed by the students: a story with dialogue between characters (Figure 5a); and an animation with interaction, where the main character can move according to the information provided by the user (Figure 5b).

Figure 5: - Projects carried out by the students. (A) Story with two characters. (b) Animation with data input to control the character's speed.



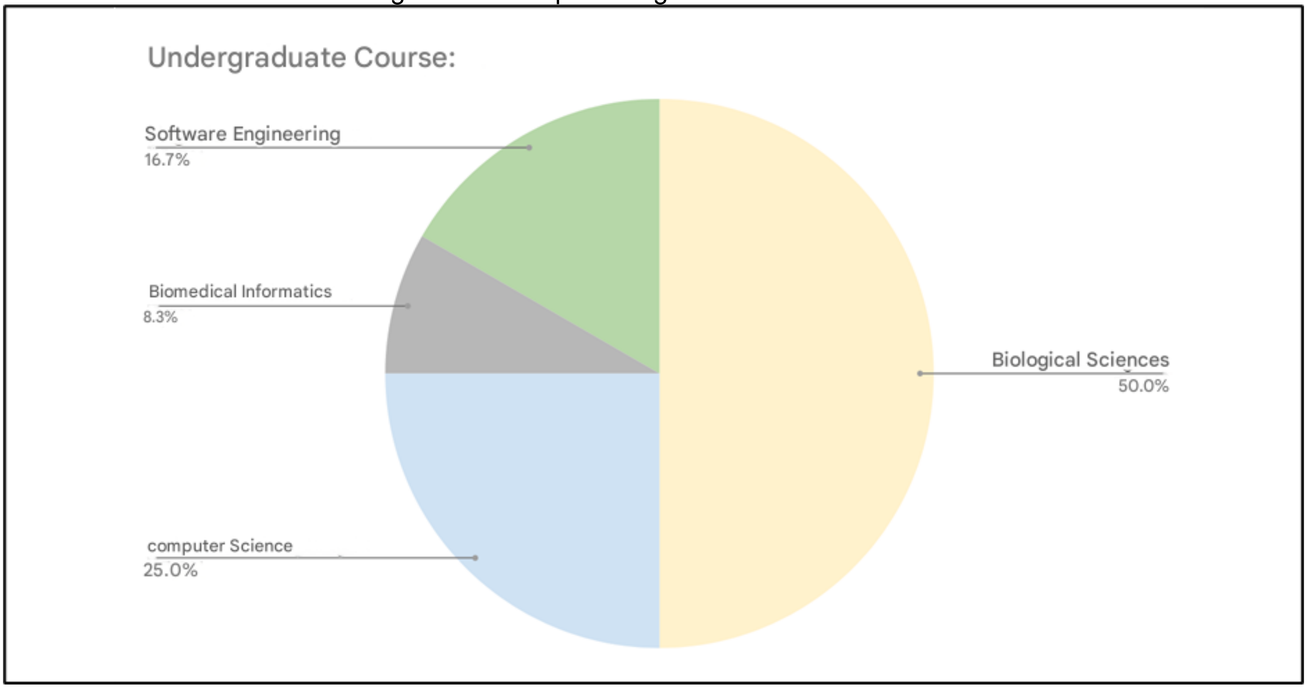
ANALYSIS OF DATA AND RESULTS ACHIEVED

This section reports the results obtained at the end of the practical experience of the PAE through the quantitative and qualitative analysis of the data collection instruments. Quantitative research sought to numerically measure the meanings of the phenomena studied, that is, to translate opinions and information into numbers to classify and analyze them through the use of statistical resources and/or techniques (Schneider, Fujii, & Corazza, 2017), while qualitative research aims to give meaning and significance to the findings, observing and interpreting the reality studied, using diversified methodological procedures, which seek explanations to generate comparability or exemplarity (Pádua, 2019).

In this work, the quantitative results are derived from the answers to the closed questions and the qualitative results are obtained by the critical appreciation of the observation of use, open questions and projects produced by the students.

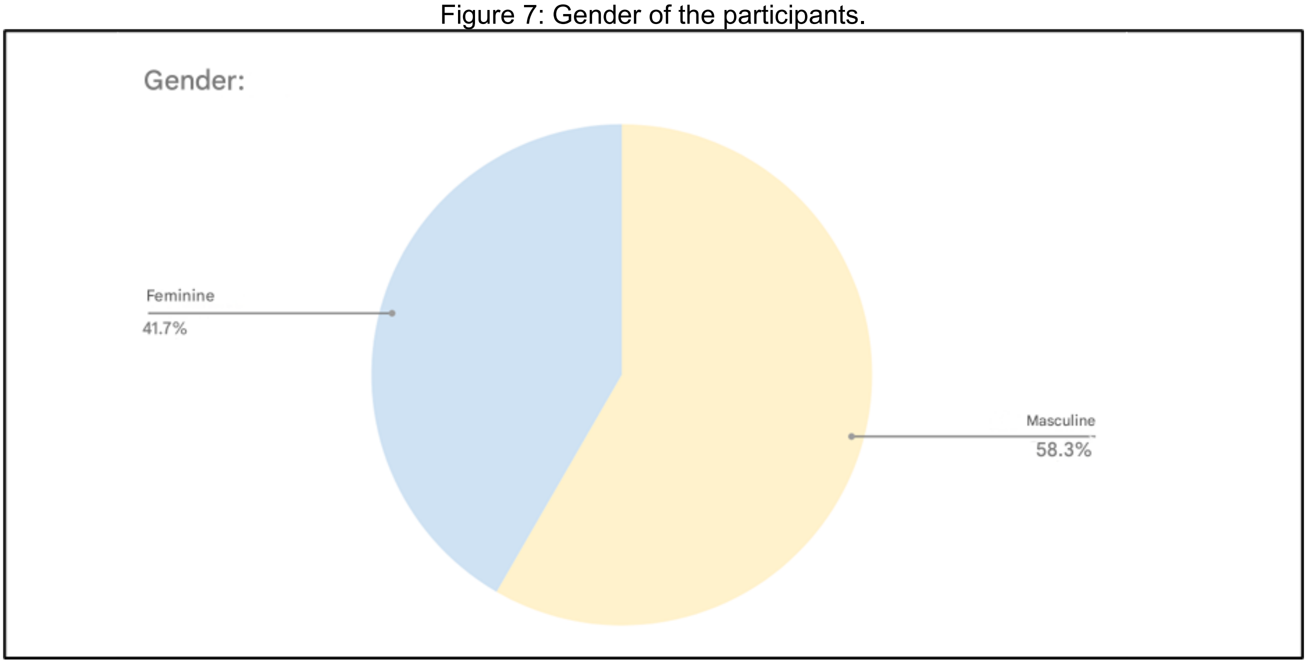
Figures 5,6,7,8 illustrate the profile of the participants and refer to the pre-test questionnaire. In the biological area, seven students participated (six in biological sciences and one in biomedical informatics), while in the exact area, five students participated (three in computer science and two in software engineering), making up a total sample of twelve participants (Figure 6). However, during the course of the study, one participant who, despite having participated and performed the proposed activities, did not complete the post-test questionnaire and could not be included in the analysis stage.

Figure 6: Participants' higher education courses.



Regarding gender, seven participants were male (58.3%) and five participants were female (41.7%) (Figure 7).

Figure 7: Gender of the participants.



Half of the sample was aged between 18 and 21 years and the other half between 22 and 30 years (Figure 8).

All participants declared that they knew some programming language, that is, they had some experience with the content (Figure 9). The languages most known by the students were: Python (100%), Java (58.3%), C (41.7%), JavaScript (33.3%), C++ (25%). C#, PHP, Haskell, Go, Assembly, and Spyder were also mentioned. Python, in addition to being the most mentioned language, is often used by biologists. It is observed that it is still rare, especially in Brazil, to find disciplines in undergraduate biology courses that comprehensively contemplate the teaching of some programming language (Salazar, 2018).

Figure 8: Age group of the participants.

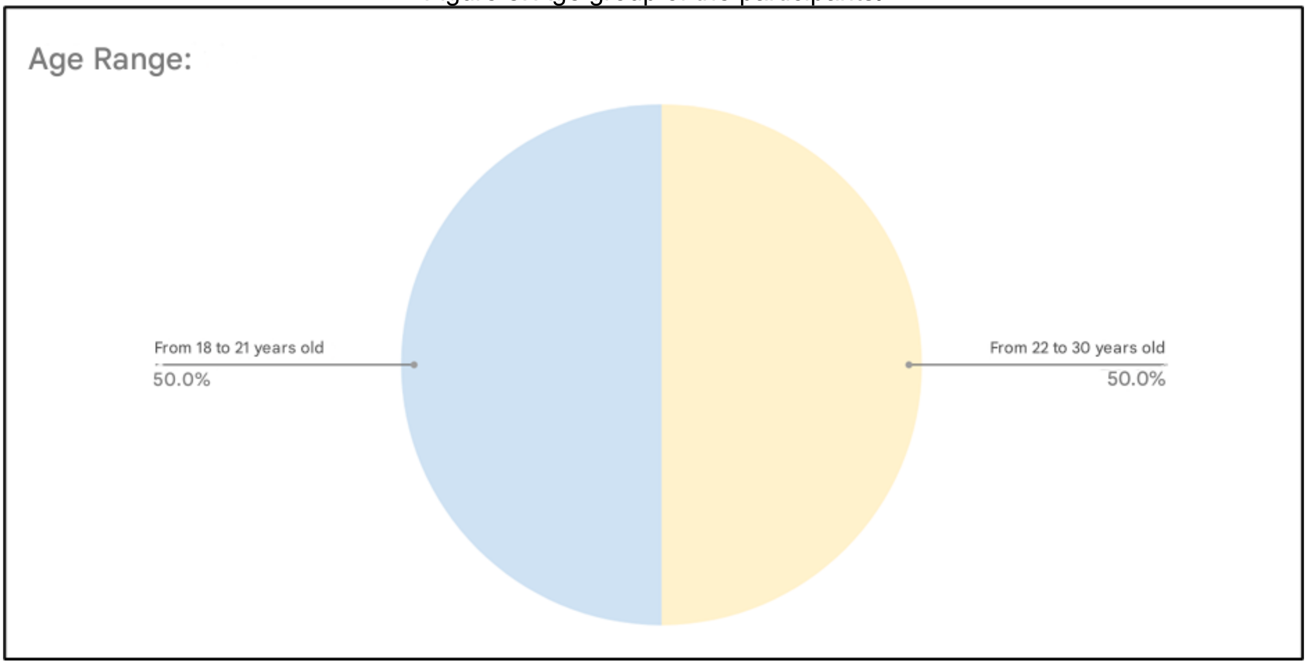
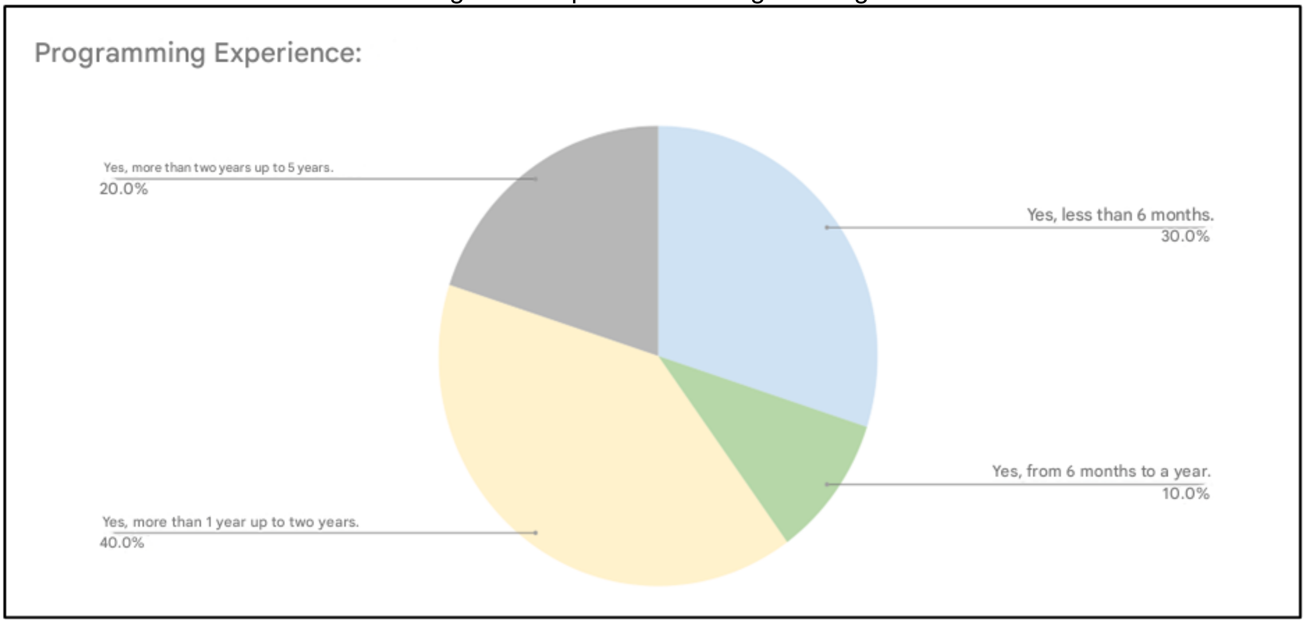


Figure 9: Experience in Programming.



After performing the tasks, the students answered quantitative questions about their experience, which they considered Scratch suitable for teaching programming to beginners, 90% of the students answered that they agreed, and one software engineering student believes that it could be used in basic education. Students in the area of biological sciences gave positive answers about the language being suitable for undergraduate students and that they use Scratch to continue the practice. However, two students from the technological area stated that they would not continue using this language, as well as did not know how to give an opinion on whether Scratch could be used by students from other areas, while another student stated that the language should not be used.

Regarding the qualitative analysis, the data that emerged were grouped and later discussed according to the categories of the ARCS method. The 4 categories could be evidenced, as described in Table 3.

Table 3. Categories evidenced during the PAE

ARCS Method	Guiding Concept
Attention	This category can be verified because the focus of the students during the execution of practical activities was perceived, in which the students were able to perform the tasks. They also completed the project in the expected time.
Relevance	Based on the quantitative data, the results suggest that the Scratch programming language is relevant for students who are not in computer areas.
Confidence	Motivation was noted as the students perceived their progress in the execution of programming commands and felt comfortable exploring the resources. Thus, the category emerged due to the ease of students demonstrating confidence, being able to perform tasks and build their own project.
Satisfaction	During the practical activities, it was noticed that the students expressed their feelings, showing joy and surprise when interacting with Scratch. They also demonstrated creativity and imagination to propose the development of the projects, noting the engagement and satisfaction of the participants.

From the analysis of the information contained in the table, we can establish relationships with the scientific literature to detect and interpret points of divergence or convergence. The analysis of the qualitative data suggests that the Scratch programming language can be used in higher education, especially for beginner students.

In the pre-test questionnaire, students reported that learning through computational thinking and programming provides a better understanding of current technologies (programs and their operation) and that their understanding allows for easier implementation of programs. The students also demonstrated that programming is an important basis for several issues, both for the present and for their future, which demonstrates the adequacy of relevance and satisfaction.

From the point of view of learning, students believe that computational thinking is an important resource for cognitive development, in addition to enabling insertion in the job market and the development of skills that can help them in academic work, which corroborates Wangenheim, Nunes and Santos (2014) when they state that this is a necessary knowledge for your insertion in the job market, regardless of your final area of study or chosen professional career. In this sense, Valente (2002) states that these skills and competencies must be built by each learner in interaction with objects and with people who cohabit their daily lives.

In this context, the Scratch programming language becomes an instrument with the potential to engage students in learning computational thinking, which has a strong resonance with the activities in which users are interested, such as the creation of animated stories, games and interactive presentations (Valente, 2016). This engagement can be observed in the participants' testimonies:

- "*Scratch uses the same language components as other forms of programming, such as Python. This makes it easier for a person with no experience of Scratch to another language medium, with a base acquired in a relaxed way*" – Relevance;
- "*It was a simpler way to learn the commands and that involves those who are learning*" – Relevance;
- "I believe that the use of Scratch at the beginning of learning is more intuitive and pleasurable for the student" – Satisfaction;
- "*Currently, coded programming languages can be too complex for beginners, scary on first impressions. Scratch allows for friendlier and more interactive learning*" – Relevance and trust;
- "*The Scratch tool provides the possibility of visualizing what is happening, so that the user can follow and better understand what each command does*" – Attention;

- "*With Scratch, I had a less stressful experience, compared to another programming language itself, obtaining knowledge in structures in an easy way*" – Relevance;
- "*The use of Scratch helps a lot in understanding the initial concepts of programming, not to mention that it ends up being a fun and dynamic method of learning to program. Unlike the traditional method, in which the first contact of the biological sciences student with the programming language is a little intimidating and ends up becoming, in most cases, a dull and difficult subject*" – Relevance, confidence and satisfaction;
- "*Scratch abstracts all the coding part that in traditional learning can scare and make the person give up learning. In addition, in the tool it is possible to quickly identify the effect of each programmed step and identify what must be done to reach the final goal and this is not so clear when we start learning with a coded language" –* Attention;
- "*I believe it is a more didactic way to introduce concepts that are often difficult to assimilate for those who have no experience with the area*" – Relevance;
- "*The advantage is that the student gets more involved*" – Relevance;
- "*Scratch has a user-friendly interface and simplifies concepts that are initially difficult to understand. The abstraction of coding through colored blocks and the use of characters and scenarios helps to make learning fun*" – Relevance and satisfaction;
- "*The didactics, the program presents the basic concepts of programming in a simple and fun way; small commands that, if explained together with a theoretical basis, where there is a professional able to teach properly, can become useful at the first moment of immersion in this field of study*" – Relevance and confidence;
- "*The possibility of being able to visualize the actions for which the commands are associated, allowing people with no experience to start programming*" – Relevance.

The participants are in accordance with the definitions of Valente (2016) who states that computational thinking is a process of problem solving and that it has the following characteristics: formulation of problems in a way that allows the use of a computer and other tools to help solve them; logical organization and data analysis; automation of solutions through algorithmic thinking (the series of ordered steps); identification, analysis and implementation of possible solutions in order to achieve the most efficient and effective combination of steps and resources; and generalization and transfer of this problem-solving process to a wide variety of problems.

This is important because the objects and activities must be stimulating so that the student can be involved, allow them to be explored and increase the quality of the interaction (Valente, 2002).

Regarding learning, students demonstrate that Scratch allows the understanding of basic programming concepts in an interactive way and simplifies concepts that are initially difficult to understand, facilitates the understanding of algorithms and code structures, such as variables, and helps in the analysis of results. Without any formal rigor, characteristic of traditional programming, concepts are learned more easily and direct feedback on actions, via graphic representation, makes the interaction more relaxed and fun.

Asked about the disadvantages, only one student pointed out negative points in its use in higher education: "*Perfect for teaching logical structures to children about programming, for adults I think it is not efficient and productive, even so, it is always good to learn with practice in other areas, it can be more advantageous, because it does not show the codes of each command/block".*

The students suggested that it would be interesting if the interaction allowed them to follow the programming language that is linked to each action block and that there could be more advanced levels of programming in the language. For example, a biology student stated that "*the insertion of blocks within blocks allows you to execute everything at once, but to execute only one part (to see what it does) you have to remove it from the largest block, which makes it difficult to understand what each part is doing. In addition, building something that needs many blocks within other blocks makes the construction too laborious and the concepts and understanding as a whole (which should be the main thing) is left aside. It would be interesting for the tool to present structures such as vectors and matrices (which are basic and important structures)*".

Finally, at the end of the application of the PAE, the students report the following:

- "*Programming is important regardless of career, as it helps to get a sense of how tools we use every day work. In Biology, several areas use programming, such as in the creation of scripts and the like, such as in areas of Bioinformatics. Programming does not only allow us to work in applied areas, but to exercise logic and thinking";*
- *"It is important to obtain more knowledge and programming base";*
- *"Learning programming for people of any age is important for the development of logical reasoning and the formulation of strategies to solve problems";*
- *I still have the same opinion that learning programming is essential today. But I understood a little better why this is important";*

- *"After participating in the workshop, I believe that learning programming, in addition to providing greater possibilities of acting in the job market, provides stimulation for solving everyday problems";*
- *"Programming is a fundamental pillar for my career, as it is necessary to understand the process involving purely so-called coding, in addition to developing the logical reasoning that software demands, to seek to implement more elegant solutions";*
- *"The learning of programming language by biologists is of paramount importance, as it is very present nowadays due to the growing technological development and, in addition, plays a significant role in biotechnology, data analysis, etc.";*
- *"I still think that learning to program is extremely important to solve problems, but not only that. Learning to program helps in the process of thinking, schematizing and seeking various solutions to a given problem or simply to avoid rework";*
- *"In the area I follow it is very important, but in a broader sense, the academic area involves a lot of computational technique that can be facilitated with learning in a computational language".*

FINAL CONSIDERATIONS

With the advent of digital technologies in the educational sphere, it has become essential to reflect on the aspects related to users' interactions with tools aimed at education and their pedagogical materials (Herpich & Tarouco, 2016). In this sense, in order to contribute to a greater engagement of undergraduate students to learn computational logic, this article reported the practical experiences of the application of an Action at School Project that involved the use of the Scratch programming language, through games and animations, by higher education students.

The participants' reports suggest that most students consider the Scratch language suitable for teaching programming, especially to beginners and students in non-technological areas.

This work discussed the differences in perception between students from different areas - Biological Sciences and Exact Sciences - and how much Biological Sciences students felt engaged to continue learning programming in their careers. In this context, some situations emerged that were open to discussion. The composition of the sample, with students from biological and computer areas, where it was realized that the use of digital games and animations, developed in Scratch, aimed at learning initial content of programming logic is a motivational factor for students who are starting their practice in this area, as well as belonging to areas such as biological sciences.

Therefore, it can be said that computational thinking has shown signs as an engaging pedagogical practice, which can be used in higher education, especially for students in biological areas. In the same sense, Santana (2023), when using Scratch in the teaching of biophysics with Biology undergraduates, observed that the block language favored the learning of complex concepts related to biological membranes, promoting the development of algorithmic thinking and contributing to an interdisciplinary understanding involving biology, physics, and chemistry. This reinforces the potential of computational thinking as a significant tool for active learning in contexts other than science education.

Despite the promising results, this research has limitations. The main one refers to the small number of participants, which restricts the generalization of the data. In addition, the time allocated to the application of the Action at School Project was short, which may have limited the deepening of programming content and the more autonomous use of the Scratch tool by students. As a proposal for future work, it is recommended to carry out investigations with larger samples, apply them in different training contexts and develop continuous training trails that integrate computational thinking into the curriculum in an interdisciplinary way. It would also be relevant to explore other visual languages and programming environments with an educational focus, in order to expand the possibilities of teaching-learning in the field of Biological Sciences and in undergraduate courses in general.

REFERENCES

1. Alves, N. da C., Wangenheim, C. G. von, Rodrigues, P. E., Hauck, J. C. R., & Borgatto, A. F. (2016). Ensino de computação de forma multidisciplinar em disciplinas de história no ensino fundamental: Um estudo de caso. Revista Brasileira de Informática na Educação, 24(3), 31–41. https://doi.org/10.5753/rbie.2016.24.3.31

2. Bardin, L. (1977). Análise de conteúdo. Livraria Martins Fontes.

3. Bishop, Ö. T., Adebiyi, E. F., Alzohairy, A. M., Everett, D., Ghedira, K., Ghouila, A., … H3Africa Consortium. (2015). Bioinformatics education: Perspectives and challenges out of Africa. Briefings in Bioinformatics, 16(2), 355–364. https://doi.org/10.1093/bib/bbu022

4. Castilho, M. I., & Ricci, T. dos S. F. (2006). O uso de animações como elemento motivador de aprendizagem. In Atas do 1º Encontro Estadual de Ensino de Física (pp. 1–10). Instituto de Física, Universidade Federal do Rio Grande do Sul.

5. Cybis, W. de A., Betiol, A. H., & Faust, R. (2015). Ergonomia e usabilidade: Conhecimentos, métodos e aplicações (3rd ed.). Novatec.

6. Czerkawski, B. C., & Lyman, E. W. (2015). Exploring issues about computational thinking in higher education. TechTrends, 59(2), 57–65. https://doi.org/10.1007/s11528-015-0840-3

7. Denning, P. J. (2018). Computational thinking in science. In M. Pitici (Ed.), The best writing on mathematics 2018 (pp. 67–77). Princeton University Press.

8. Gomes, T. C. S., Tedesco, P. C. de A. R., & Melo, J. C. B. de. (2016). Jogos no design de experiências de aprendizagem de programação engajadoras. Jornada de Atualização em Informática na Educação, 5(1), 39–77. https://doi.org/10.5753/jaie.2016.5.1.39

9. Herpich, F., & Tarouco, L. M. R. (2016). Engajamento de usuários em mundos virtuais: Uma análise teórica-prática. Renote, 14(1), Article 67387. https://doi.org/10.22456/1679-1916.67387

10. Keller, J. M. (2009). Motivational design for learning and performance: The ARCS model approach. Springer Science & Business Media.

11. Magana, A. J., Taleyarkhan, M., Alvarado, D. R., Kane, M., Springer, J., & Clase, K. (2014). A survey of scholarly literature describing the field of bioinformatics education and bioinformatics educational research. CBE Life Sciences Education, 13(4), 607–623. https://doi.org/10.1187/cbe.13-10-0193

12. Marques, D. L., Costa, L. F. S., Silva, M. A. de A., & Rebouças, A. D. D. S. (2011). Atraindo alunos do ensino médio para a computação: Uma experiência prática de introdução a programação utilizando jogos e Python. Anais do Workshop de Informática na Escola, 1(1), 1138–1147. https://doi.org/10.5753/cbie.wie.2011.1138-1147

13. Preece, J., Rogers, Y., & Sharp, H. (2013). Design de interação: Além da interação homem-computador (3rd ed.). Bookman.

14. Pozzebon, L. V., Velho, A. R. T., & Barwaldt, R. (2019). Viagem espacial: Um recurso para a alfabetização. In R. Barwaldt, J. R. P. Maurell, & V. da S. Xavier (Eds.), Formação de professores em tecnologias educacionais: Construindo práticas pedagógicas (Vol. 31, pp. 387–396). Editora da FURG.

15. Qin, H. (2009). Teaching computational thinking through bioinformatics to biology students. Proceedings of the 40th ACM Technical Symposium on Computer Science Education, 188–191. https://doi.org/10.1145/1508865.1508932

16. Ramos, F., & Teixeira, L. da S. (2015). Significação da aprendizagem através do pensamento computacional no ensino médio: Uma experiência com Scratch. Anais do Workshop de Informática na Escola, 21(1), 217–226. https://doi.org/10.5753/cbie.wie.2015.217

17. Ranganathan, S. (2005). Bioinformatics education: Perspectives and challenges. PLOS Computational Biology, 1(6), Article e52. https://doi.org/10.1371/journal.pcbi.0010052

18. Salazar, V. W. (2018). Ensino de programação para biólogos: Retornando os conhecimentos de um estudo empírico em oceanografia física [Trabalho de Conclusão de Curso, Universidade Federal de Santa Catarina]. https://repositorio.ufsc.br/handle/123456789/188645

19. Santana, M. P. S. (2023). Pensamento computacional no ensino de biofísica na formação inicial de professores de biologia: Utilizando programação em blocos com o Scratch [Dissertação de mestrado, Universidade Estadual Paulista "Júlio de Mesquita Filho"]. https://hdl.handle.net/11449/253143

20. Schneider, E. M., Fujii, R. A. X., & Corazza, M. J. (2017). Pesquisas quali-quantitativas: Contribuições para a pesquisa em ensino de ciências. Revista Pesquisa Qualitativa, 5(9), 569–584. https://doi.org/10.33361/RPQ.2017.v.5.n.9.184

21. Silva, A. H., & Fossá, M. I. T. (2015). Análise de conteúdo: Exemplo de aplicação da técnica para análise de dados qualitativos. Qualitas Revista Eletrônica, 16(1), 1–14. https://doi.org/10.18391/qualitas.v16i1.2138

22. Soares, J., & Rodrigues, A. (2016). Avaliação do jogo educativo Lei de Mendel. Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação - SBIE), 27(1), 826–835. https://doi.org/10.5753/cbie.sbie.2016.826

23. Sousa, R. M., & Lencastre, J. A. (2014). Scratch: Uma opção válida para desenvolver o pensamento computacional e a competência de resolução de problemas. In Atas do 2º Encontro sobre Jogos e Mobile Learning (pp. 256–267). Universidade de Coimbra.

24. Tarouco, L. M. R., Roland, L. C., Fabre, M.-C. J. M., & Konrath, M. L. P. (2004). Jogos educacionais. Renote: Revista Novas Tecnologias na Educação, 2(1), Article 3. https://doi.org/10.22456/1679-1916.11951

25. Valente, J. A. (2002). A espiral da aprendizagem e as tecnologias da informação e comunicação: Repensando conceitos. In A tecnologia no ensino: Implicações para a aprendizagem (pp. 15–37). Casa do Psicólogo.

26. Valente, J. A. (2016). Integração do pensamento computacional no currículo da educação básica: Diferentes estratégias usadas e questões de formação de professores e avaliação do aluno. Revista e-Curriculum, 14(3), 864–897. https://doi.org/10.23925/1809-3876.2016v14i3p864-897

27. Wangenheim, C. G. von, Nunes, V. R., & Santos, G. D. dos. (2014). Ensino de computação com Scratch no ensino fundamental: Um estudo de caso. Revista Brasileira de Informática na Educação, 22(3), 115–125. https://doi.org/10.5753/rbie.2014.22.03.115

28. Welch, L. R., Schwartz, R., & Lewitter, F. (2011). Report from the ISCB Education Committee: A draft curriculum for bioinformatics [Abstract]. In Third Annual RECOMB Conference on Bioinformatics Education (RECOMB-BE), an ISMB/ECCB 2011 Satellite Meeting. 19th Annual International Conference on Intelligent Systems for Molecular Biology and 10th European Conference on Computational Biology, Vienna, Austria.

29. Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33–35. https://doi.org/10.1145/1118178.1118215

30. Zanetti, H., Borges, M., & Ricarte, I. (2016). Pensamento computacional no ensino de programação: Uma revisão sistemática da literatura brasileira. Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação - SBIE), 27(1), 21–30. https://doi.org/10.5753/cbie.sbie.2016.21