pro·posições
*e*-ISSN 1980-6248

http://dx.doi.org/10.1590/1980-6248-2018-0034

ARTIGOS

# Computational thinking as a heuristic endeavour: students' solutions of coding problems [1] [2] [3] [4]

# *Pensamento Computacional como uma iniciativa heurística: soluções de estudantes sobre problemas de programação*

Ricardo Scucuglia Rodrigues da Silva [(i)]

George Gadanidis [(ii)]

Janette Hughes [(iii)]

Immaculate Kizito Namukasa [(iv)]

[(i)] Universidade Estadual Paulista Júlio de Mesquita Filho – Unesp, São José do Rio Preto, SP, Brasil. http://orcid.org/0000-0002-5810-2259, ricardo.scucuglia@unesp.br

[(ii)] Western University – UWO, London, ON, Canada. https://orcid.org/0000-0002-5982-6056, ggadanid@uwo.ca

[(iii)] University of Ontario Institute of Technology – UOIT, Oshawa, ON, Canada. https://orcid.org/0000-0002-3463-8382, janette.hughes@uoit.ca

[(iv)] Western University – UWO, London, ON, Canada. https://orcid.org/0000-0001-7231-0671, inamukas@uwo.ca

[1] Apoio: Social Sciences and Humanities Research Council of Canada (SSHRC)
Ontario Research Fund: Research Excellence (ORF-RE)
National Council for Scientific and Technological Development (CNPq) (428323/2018/-9)

[2] References correction and bibliographic normalization services: Andressa Picosque (Tikinet) – revisao@tikinet.com.br

[3] Copy Editor: José Pereira de Queiroz – ze.pereira.queiroz@gmail.com

[4] Responsible Editor: Prof. Dr. Carlos Miguel da Silva Ribeiro, https://orcid.org/0000-0003-3505-4431, cmribas78@gmail.com

http://dx.doi.org/10.1590/1980-6248-2018-0034

**Abstract:** In this paper we investigate students' computational thinking in mathematics education. Specifically, through the analysis of teaching experiments conducted as qualitative case studies, we explore aspects of constructionism and problem solving. In different learning scenarios, pairs of elementary school and undergraduate students explored coding puzzles in order to complete a posed computational-mathematical task. From a constructionist point of view, the results indicate that the learning experience involved a problem solving spiral of description, execution, reflection and debugging. In the case of the experience of the undergraduate students, we also identified specific characteristics of computational thinking related to heuristic processes such as exploration, planning, analysis, and verification.
**Keywords:** mathematics education, constructionism, problem solving

***Resumo:*** *Neste artigo, investigamos o pensamento computacional de alunos em educação matemática. Especificamente, pela análise de experimentos de ensino conduzidos como estudos de caso qualitativos, exploramos aspectos do construcionismo e da resolução de problemas. Em diferentes cenários de aprendizagem, duplas de alunos do ensino fundamental e de graduação exploraram problemas de codificação para concluir uma tarefa matemático-computacional. Do ponto de vista construcionista, os resultados indicam que a experiência de aprendizagem envolveu uma espiral de descrição, execução, reflexão e depuração de problemas. No caso da experiência dos estudantes de graduação, também identificamos características específicas do pensamento computacional relacionadas a processos heurísticos, como exploração, planejamento, análise e verificação.*
***Palavras-chave:*** *educação matemática, construcionismo, resolução de problemas*

## Introduction

The genesis of the very notion of computational thinking is directly related to problem solving, which is integral to mathematics doing and learning. For Denning (2017), the history of computational thinking begins with Polya. Moreover, the emergence of new types of digital media and its relation to K-12 curricula in the mid-2000's has offered alternative ways to explore tasks based on computational affordances for (mathematical) teaching and learning (Lu & Fletcher, 2009). In the scope of this study, computational thinking is related to other

http://dx.doi.org/10.1590/1980-6248-2018-0034

types of thinking and usually involves aspects such as automation, abstraction, decomposition, coding, problem solving, heuristics, and so on. According to Wing (2008),

> Computational thinking is a kind of analytical thinking. It shares with mathematical thinking in the general ways in which we might approach solving a problem. It shares with engineering thinking in the general ways in which we might approach designing and evaluating a large, complex system that operates within the constraints of the real world. It shares with scientific thinking in the general ways in which we might approach understanding computability, intelligence, the mind and human behaviour. (p. 3717)

Computational thinking is evident in three different contexts: (i) *on-screen* programming, such as writing code in a specific language like Python to dynamically simulate the graph of a family of mathematical functions (such as y=mx+b), (ii) *off-screen* or *unplugged* programming-based solutions, such as using pseudocode to plan solutions or using commands to control objects in the physical world (for example instructing a peer how to walk a square), and (iii) using programming to control *digital tangibles,* such as a digital circuits or robots.

In this paper we focus on on-screen programming, often referred to as coding, and we are especially interested in the role coding plays in problem solving when the same task is explored at different school levels, that is, how an open-ended mathematics/coding task offers ways to engage elementary school and undergraduate students in heuristics processes of learning mathematics. Specifically, we investigate aspects of learning experiences of Grade-6 students and undergraduate mathematics majors using a framework based on two main theoretical approaches: (i) *constructionism* (Papert, 1993) as learning spiral (Valente, 2003); and (ii) *problem solving* components for heuristic learning (Polya, 1957; Schoenfeld, 1992). Our research question is: "What are the main aspects of students' learning experience in terms of computational thinking and heuristics, in different school levels, when students explore a mathematical-computational task based on coding?"

To do this, we conducted teaching experiments with two different groups of students using the computational-mathematical learning environment named *Repeating Patterns* (Gadanidis & Yiu, 2017). The experiments were each structured in four-hour sessions, with (i) three pairs of Grade-6 students; and (ii) two pairs of undergraduate majors in mathematics. Our goal is not to prove that students learned more or less (especially given the small sample in our study), but to investigate in depth and to identify and analyze the learning processes involved.

http://dx.doi.org/10.1590/1980-6248-2018-0034

Our findings indicate that, at different school levels, the exploration of computational-mathematical tasks may offer ways to conceive the learning experience as a spiral process, whose cognitive and attitudinal components reveal problem solving components. On the one hand, description, execution, reflection, and debugging are processes that form a learning spiral. On the other hand, exploration, planning, analysis, and verification are fundamental components of heuristics related to computational thinking. Although these components are also in part mathematical doing and learning, there is a difference, and an enhancement, made possible by the ability to model mathematical processes and relationships with code. These models through code make dynamic modelling possible, and enhance the ability to explore and to verify, for example.

According to Borba and Villarreal (2005), experimentation-with-technology in mathematics education offers ways for reorganization of mathematical thinking, regarding processes such as "the possibility of testing a conjecture using a great number of examples and the chance of repeating experiments, due to quick feedback given by computers" (p. 75). This study explores the reorganization of students' mathematical thinking due to the enhanced ability offered by computer programming to build and control dynamic models of mathematical relationships—that is, the enhanced ability to engage in constructionist practices.

## Theoretical framework

As an initial strategy to built our theoretical framework, we explore the very notion of *networking of theories* (Artigue & Mariotti, 2014; Bikner-Ahsbahs & Prediger, 2010), because we seek to elaborate connections between computational thinking, constructionism, problem solving, and heuristics. Specifically, we first relate constructionism to the idea of learning spiral, and problem solving to heuristics, then we make connections between learning spiral and heuristics (see Figure 1). We also regard that "theories within this semiosphere can be described as triplets (P, M, Q) that establish languages and allow the building of relationships between them" (Bikner-Ahsbahs, 2010, p. 9). In this context, P refers to principles, M to methodologies, and Q represents questions related to P and M. According to Bikner-Ahsbahs

http://dx.doi.org/10.1590/1980-6248-2018-0034

(2010), "a connection between two theories establishes a specific relation that depends on the theories' structures and the goal of this connection" (p. 9).

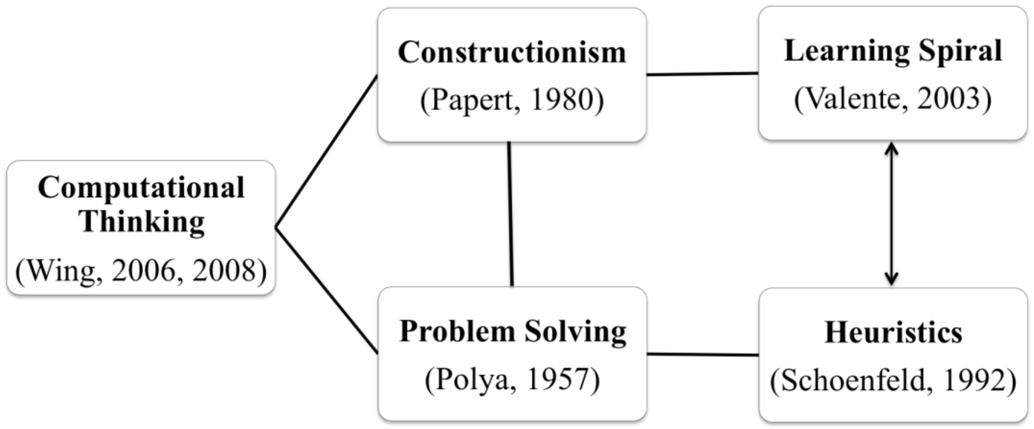


**Figure 1 – Networking theories of the study**

In the following subsections, we discuss aspects related to the principles (Ps) of the theories used in this study. First, we explore constructionism in terms of computational thinking, and thus we discuss the notion of learning spiral in constructionism. Then, we discuss computational thinking, problem solving, and heuristics seeking to establish connections between the learning spiral and heuristics components. Our goal is to show that the synergy between these perspectives, that is, they are not in epistemological conflict and they are complementary.

## Computational thinking, constructionism, and learning spiral

From Wing's (2006) work, we identify a connection between computational thinking and computer programing, although "thinking like a computer scientist means more than being able to program a computer. It requires thinking at multiple levels of abstraction" (p. 35). According to Papert (1993), computer programing has an epistemological potential for children's learning, offering ways to 'think about thinking'. Thus, the very notion of *constructionism* emphasizes students' effective actions for learning. This perspective

> is built on the assumption that children will do best by finding … for themselves the specific knowledge they need; organized or informal education can help most by making sure they are supported morally, psychologically, materially, and intellectually in their efforts. (p. 139)

Construction of knowledge is achieved "when it is supported by construction of a more public sort 'in the world'[, that is,] the product can be shown, discussed, examined, probed, and admired. It is out there" (Papert, 1993, p. 142). Using programs such as LOGO, children may program the computer and, through hands-on activities, explore science, mathematics, and other subjects (Papert, 1980). Papert (1980) suggested that exploration and a failure-positive learning environment promotes problem solving:

> many children are held back in their learning because they have a model of learning in which you have either "got it" or "got it wrong". But when you learn to program you almost never get it right the first time. Learning to be a master programmer is learning to become highly skilled at isolating and correcting "bugs", the parts that keep the program from working. The question to ask about the program is not whether it is right or wrong, but if it is fixable. If this way of looking at intellectual products were generalized to how the larger culture thinks about knowledge and its acquisition, we all might be less intimidated by our fears of "being wrong". (p. 23)

Building on the constructionist approach, Valente (2003) developed a theoretical view referred to as *learning actions or spiral* that conceive user-computer interaction as *description-execution-reflection-debugging-description-…* actions. After a first stage of debugging, a new second type of description emerges, and so on (see Figure 2).

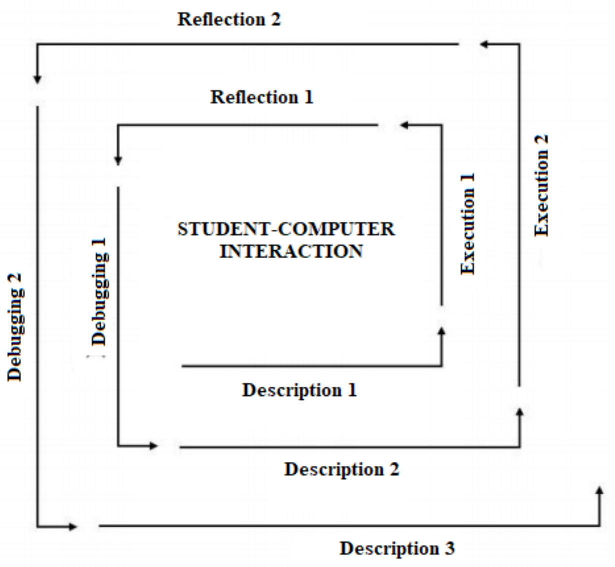


**Figure 2 – Constructionist learning spiral (Valente, 2003)**
Source: adapted by the authors from Idem (2017)

http://dx.doi.org/10.1590/1980-6248-2018-0034

According to Valente (2003), these actions can be described as:

- ***Description:*** initial proposed solution of the learner to describe steps of the problem solution in terms of the software language;
- ***Execution:*** procedures executed by the computer;
- ***Reflection:*** based on results given by the computer, the learner compares what was achieved with the initial intended ideas;
- ***Debugging:*** the learner explores new information and alternate solutions into another action of description.

Maltempi (2005) describes these constructionist processes as the following:

> The *description* action corresponds to the explication of the ideas, concepts and strategies that the learner uses to elaborate his/her (project) program, and offers the opportunity to "see" the student's reasoning process and understand what is being done, conceptual problems etc. To the student, the opportunity to "test" ideas and conceptions is amplified when the computer *executes* the program, presenting the result of the same. The answer given by the computer is faithful and immediate, offering the student the possibility of confronting their original ideas with the result. This has led to a process of *reflection* and awareness he knows it or not. When the result provided by the computer does not match the expected, the learner needs to *debug* the program, that is, review the process of representing the solution of the problem (his/her ideas). (p. 6)

Although Valente (2003) proposes a learning spiral as a sequence of actions within constructionism, computer-based learning processes are considerably dynamic, that is, actions usually happen simultaneously, revealing the complexity of cognitive, linguistic, and technological processes of learning. In our study, we identified the description and execution actions as students' inputs of commands and computer's executions, respectively. However, reflections and debugging actions appeared to happen simultaneously, through students' dialogues and interactions. Indeed, these processes seem to be similar and directly related to the main components of problem solving. We do find this type of dynamicity/complexity in computational thinking related to problem solving (Denning, 2017). In particular, problem solving is related to heuristics. Schoenfeld (1985) states:

Heuristic strategies are rules of thumb for successful problem solving, general suggestions that help an individual to understand a problem better or to make progress toward its solution. Such strategies include exploiting analogies, introducing auxiliary elements in a problem or working auxiliary problem, arguing by contradiction, working forward from the data, decomposing and recombining, exploiting related problems, drawing figures, generalizing and using 'inventor's paradox' specializing, using reductio ad absurdum and indirect proof working backward. There is some consensus among mathematicians that these strategies are useful. (p. 23)

Thus, constructionism, computational thinking, and problem solving are closely related. Constructionism, in general, is a learning theory involving problem solving that usually offers ways to explore computer programming in mathematics learning.

## Computational thinking, problem solving, and heuristics

According to Wing (2006),

Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science. Computational thinking includes a range of mental tools that reflect the breadth of the field of computer science.… Computational thinking is using heuristic reasoning to discover a solution. It is planning, learning, and scheduling in the presence of uncertainty. (p. 33-34)

The heuristics involved in computational activities have a similar nature in problem solving. Schoenfeld (1992) presents a framework on metacognition exploring heuristic aspects of students' problem solving in different scenarios. Specifically, Schoenfeld (1992) built graphs to display students' activities over time to identify the variety of thinking processes involved (or not) in solving a problem, and the depth/diversity of heuristic endeavours. We explore these processes as heuristic actions in problem solving. In some types of exploration, students only read the mathematical problem or task and briefly explored it. That is, students may not meaningfully engage with the problem solving and heuristic processes when they explore mathematics. In contrast, in rich heuristic situations, students developed dynamically the following processes, identified by Schoenfeld (1992): analyze; explore; plan; implement; and verify. Thus, regarding Schoenfeld's (1992) heuristic processes, and fundamental ideas supported by Polya (1957), we suggest for this research the following heuristic components as essential processes for problem solving in computational environments:

- ***Exploring:*** In the exploratory stage of problem solving, students read, reread, and restate the problem in order to understand it. That is, students "identify the information given and the information that needs to be determined" (Ontario Ministry of Education, 2005, p. 15). It is important for students' learning to talk about the problem in order to better understand it. Beyond understanding the problem, students "should also desire its solution" (Polya, 1957, p. 6). In constructionist terms, exploration in problem solving may be conceptualized as a first attempt of debugging.
- ***Planning:*** Students must relate the problem with problems previously explored in order to elaborate, select, and combine different potential strategies for solutions (Ontario Ministry of Education, 2005). In constructionist terms, the processes of planning in problem solving consolidates the description component of the spiral.
- ***Verifying:*** Students execute their strategies, exploring possible calculations, computations, and constructions one must perform in order to obtain the unknown (Polya, 1957). They may draw pictures, use manipulatives, verbal and written words and symbols to represent and share possible results or solutions (Ontario Ministry of Education, 2005). Regarding the constructionist spiral components, verification refers to computer execution.
- ***Analysing:*** Students check their results or answers, reviewing their methods or strategies. "By looking back at the completed solution, by considering and re-examining the result and the path that led to it, they could consolidate their knowledge and develop their ability to solve problems" (Polya, 1957, p. 14-15). In constructionist terms, analysis refers to the reflection component of the spiral.

In this way we consider two conceptualizations of related processes of learning (with technology), one regarding actions within a computer environment and another emphasizing heuristic or discovering processes in problem solving (see Figure 3). We must also acknowledge the complexity of these components and their relations, as they may be related in a variety of ways and their characteristics are quite similar depending on the nature of the investigated learning phenomena.

http://dx.doi.org/10.1590/1980-6248-2018-0034

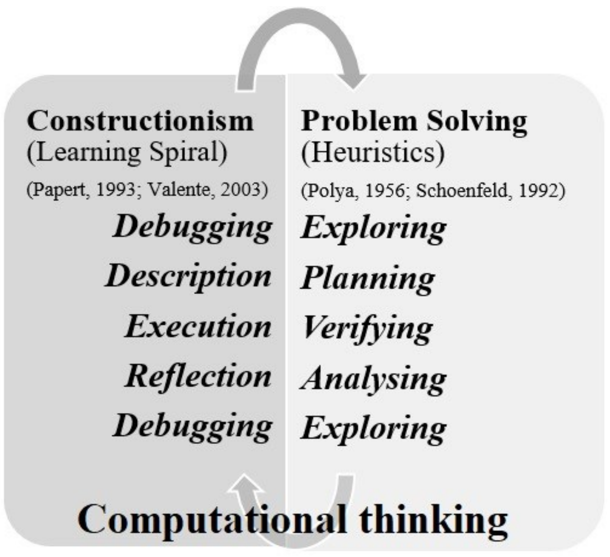


**Figure 3 – Constructionist and Problem Solving components**

## Objectives and methodology

The goal of this study is to investigate aspects of students' learning when they are engaged in the exploration of computational-mathematical tasks. Specifically, we explore connections between constructionism (Valente, 2003) and problem solving (Polya, 1957; Schoenfeld, 1992) to discuss relations between computational thinking and heuristics in school learning. To develop this investigation, we conducted teaching experiments (Steffe & Thompson, 2000), conceived in this report as qualitative case studies (Stake, 2000). As Ms within a networking of theory, teaching experiments, video analysis, and case studies are coherent methodologies in the scope of this study related to computational thinking, constructionism, and problem solving.

The teaching experiments were structured in two main sessions of four hours each with each group of participants. That is, we conducted one set of "1-hour×4 sessions" with each group as follows: (i) three pairs of Grade-6 students; and (ii) two pairs of undergraduate majors in mathematics. With the first group, we recorded field notes, saved/printed screens of their work and solutions, and interviewed all the students after each 1-hour session. With the group of undergraduates, instead of conducting interviews, we screen recorded all their actions during the experiment using the software *Flashback Pro 5* recorder.

http://dx.doi.org/10.1590/1980-6248-2018-0034

We proposed to develop the same task with different groups of students, from two different school levels, in order to explore aspects towards the very notion of "low floor, high ceiling". That is, "students engage with coding with minimal prerequisite knowledge, have opportunities to explore more complex coding concepts and problems, and can pursue many different interests and for a wide audience" (Gadanidis, 2015, p. 308). With this contrast, we can investigate similarities and differences in terms of computational/heuristic processes and aspects of elementary school students' and pre-service mathematics teachers' ways of thinking. By investigating two different groups we explore specific kinds of reasoning deployments from low floor to high ceiling.

In Chart 1 we present a description of what both groups of students explored about the task in each 1-hour session. The sample selected for discussion in this report refers to the final exploration conducted in the last session.

**Chart 1 – Description of students' explorations sessions**

| Content | Description |
|---|---|
| Session 1 – (1 hour)<br><br>Explored an introductory activity using the app available at http://researchideas.ca/wmt/c2a0.html Explored the "play with patterns" activity of the task (pages 1-3 of researchideas.ca/patterns/repeating-patterns-tutorial.pdf) | Students explored an app involving coding and arts (colors and sounds). That was their first experience and familiarization with a "simplified" tool of coding and a visual-artistic-aural way of exploring patterns. Based on the experimental actions with the app, the task was introduced to the students, in which they explored sequences of colors related to multiples of 3 (pages 1-2). They were also encouraged to create their own sequences of colors and variations of their sequences, creating different types of patterns of colors (pages 2-3). |
| Session 2 – (1 hour)<br><br>Explored "play with code", "predict, edit, and test" and "solve puzzles #1" activities of the task (pages 4-6 of researchideas.ca/patterns/repeating-patterns-tutorial.pdf) | That was students' first experience with the Repeating Patterns computational-mathematical environment. They developed an initial familiarization with commands and functions of the parameters of the application. They were engaged in conjecturing connections between the computer code and the respective execution of the computer, that is, connections between representations (Borba & Villarreal, 2005). For example, they discovered that commands such as <set x to #> and <set y to #> refer to the initial position of the sequence of shapes, that is, the visual computational display regarded a Cartesian plane. Other commands explored by the students were <set angle to #>, <set step size to #>, <set stamp rate to #>, <add color shape to pattern> and <repeat # times <do #>>. In this activity, students created and saved their first repeating pattern algorithm/image and solved three problems in puzzle #1. |

| | |
|---|---|
| Session 3 – (1 hour)<br><br>Explored the "create a grid pattern" and "solve puzzles #2" activities of the task (pages 7-9 of researchideas.ca/patterns/repeating-patterns-tutorial.pdf) | The main objective of this activity of the task was to engage students in the exploration of nested loops. Through this ability/resource, students were able to construct grid patterns, displaying a repeating pattern sequence of shapes 2-dimentionally (in a grid arrangement). Puzzle #2 is formed by 6 problems. Both groups of students had difficulty in solving problems #1 and #2. Actually, Grade-6 students were unable to present a satisfactory final solution. Undergraduate students presented solutions with some incoherence, mistakes and/or imperfections. |
| Session 4 – (1 hour)<br><br>Explored the "combine patterns" and "solve puzzles #3" activities of the task (pages 10-13 of researchideas.ca/patterns/repeating-patterns-tutorial.pdf) | In the last part of the task, the episode we selected as the one we discuss in this paper (Marshall, 1996). Students explored combined patterns using nested loops. That is, they elaborated and ran two algorithms simultaneously. A first algorithm (pattern 1) was given to them and they elaborated pattern 2 based on the first. One of the reasons we selected this episode was because it is a natural extension of using nested loops to create more elaborate repeating patterns. Another reason was because the nature of the final solutions between the two groups was qualitatively different and allowed us to connect to the goal of this paper—to understand how students at different levels engage with the same task. Within the last 1-hour session, the selected episode lasted about 20-30 minutes. |

Data analysis was conducted based on the Powell, Francisco, and Maher (2003) analytic model of video analysis. This model is composed by the following non-linear procedures: observation, description, identification of turning points, transcription, coding, and composition of the narrative. Using Marshall's (1996) notion of judgement and theoretical samples in qualitative research, we report actions of pairs of students to build our case studies. Regarding the Grade-6 case, we selected from our data the interactions of one pair of students consisting of two 12-year-old male students, who are identified in this report as Student A and Student B. Regarding the mathematics majors' case, we also selected one pair of participants consisting of one 21-year-old female student and another 22-year-old male student. We refer to the undergraduate participants as Student C and Student D. In Chart 2, we describe the participants of the research.

**Chart 2 – Description of the participants**

| School Level | Participants in the research | Selected participants for the current report |
|---|---|---|
| Grade-6 | 3 pairs of students (11-12 year old) | 1 pair (two 12 year old males) |
| Undergraduate | 2 pairs of students (21-24 year old) | 1 pair (21 year old female / 22 year old male) |

To organize, analyze, and discuss the research data through a constructionist learning spiral perspective (Valente, 2003), we created charts with four columns. In the first column, we indicated the attempt of the current students' solution (e.g., first attempt of solution, second attempt of solution, …, final solution). In the second column—named *reflection and debugging*—we presented a transcription of samples of students' and teacher's dialogues during their exploration, to give the reader a better sense of students' learning experiences. In the third column—named *description*—we displayed students' typed commands on the computer screen (from screen-captured images of the commands). In the last column—*execution*—we displayed the screen-captured images of the output generated by the computer due to the respective inputs.

Regarding students' computational-mathematical exploration, we also analyzed the data in terms of problem solving. Specifically, based on the study developed by Schoenfeld (1992), we identified the constructionist components as heuristics processes and we constructed graphs to represent these processes along time. This type of methodological approach was also conducted by Wilkerson (2016) in her study on the role of technology in engaging pre-service teachers with the iterative nature of model-based inquiry. The methodological enterprise of the study towards graph constructions focused on content analysis of group discussions. As described in Figure 2, when comparing constructionism and problem solving, we built a direct relation between description and planning, execution and verifying, reflection and analysing, and debugging and exploring. For example, when reflection and debugging were explored together, we were able to find and represent graphically different nuances between analysis and exploration based on students' speeches/actions. Thus, examples of these distinctions were identified in data analysis as follows (Chart 3):

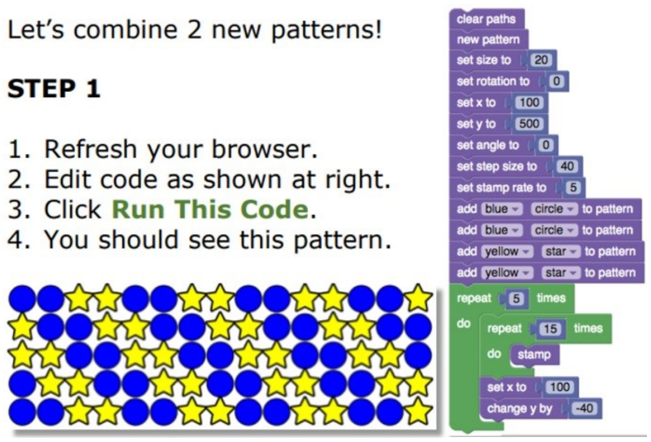
http://dx.doi.org/10.1590/1980-6248-2018-0034

**Chart 3 – Example of the categorization of data (transcription) in terms of problem solving components (analysing and exploring)**

| **Reflection and Debugging**<br><br>(actual/chronological transcription) | **Problem solving component** | **Heuristic endeavour**<br><br>(transcription related to analysis and exploration) |
|---|---|---|
| **Teacher:** Something is missing!<br>**Student C:** The directions of the diagonals are wrong too.<br>**Student D:** But we did not change anything in relation to the first pattern.<br>**Teacher:** The position is correct, right?<br>**Student C:** Only the first line of pattern 2 is right.<br>**Teacher:** What if you start pattern 2 from another position?<br>**Student C:** Let us change y to 100, change y by 40 and…<br>**Student D:** … and put the sequence as blue, blue, yellow, yellow. | **Analyzing:** students' explicit descriptions; actual configuration of parameters | **Student C:** Let us change y to 100, change y by 40 and…<br>**Student D:** … and put the sequence as blue, blue, yellow, yellow. |
| | **Exploring:** students' *debugging* actions related to the information (given/requested) and/or computer executed image. | **Teacher:** Something is missing!<br>**Student C:** The directions of the diagonals are wrong too.<br>**Student D:** But we did not change anything in relation to the first pattern.<br>**Teacher:** The position is correct, right?<br>**Student C:** Only the first line of pattern 2 is right.<br>**Teacher:** What if you start pattern 2 from another position? |

## Posing the puzzle

The mathematics-coding problem posed to students in the research is part of the sequence of puzzles organized as a task available at www.researchideas.ca/patterns (Gadanidis & Yiu, 2017). The coding environment was developed using Google's *Blockly* (developers.google.com/blockly/), which is a tool for creating applications with custom code blocks. Students explored the whole task at the teaching experiment sessions, but in this article, we only discuss the last puzzle of the task. Step 1 of this puzzle is shown in Figure 4.



Let's combine 2 new patterns!

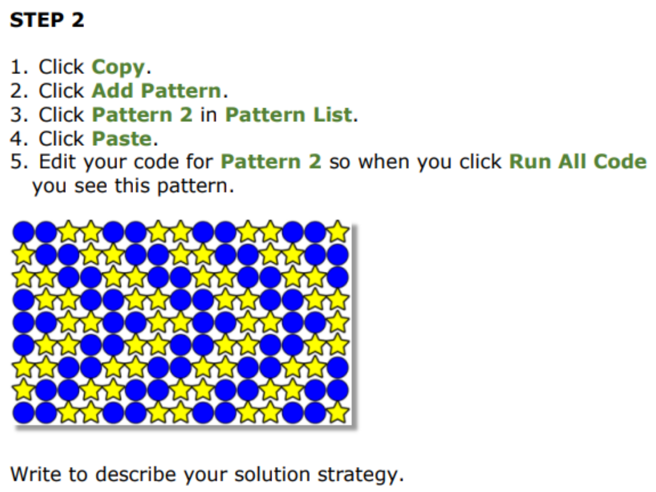
**STEP 1**

1. Refresh your browser.
2. Edit code as shown at right.
3. Click **Run This Code**.
4. You should see this pattern.

```
clear paths
new pattern
set size to 20
set rotation to 0
set x to 100
set y to 500
set angle to 0
set step size to 40
set stamp rate to 5
add blue circle to pattern
add blue circle to pattern
add yellow star to pattern
add yellow star to pattern
repeat 5 times
do  repeat 15 times
    do  stamp
    set x to 100
    change y by -40
```

**Figure 4 – The selected puzzle discussed in this paper (part 1)**

Then, in Step 2, students are asked to edit the code shown in Figure 4 so that it generates the new pattern depicted in Figure 5. One of the main pedagogic characteristics of the problem is its open-ended nature, because there is not only one solution to solve it. As we investigated, in this study, Grade-6 and undergraduate students developed different strategies of solutions, and solutions.


**STEP 2**

1. Click **Copy**.
2. Click **Add Pattern**.
3. Click **Pattern 2** in **Pattern List**.
4. Click **Paste**.
5. Edit your code for **Pattern 2** so when you click **Run All Code** you see this pattern.

Write to describe your solution strategy.

**Figure 5 – The selected puzzle discussed in this paper (part 2)**

From a didactic point of view, the objective of this task is to offer students ways to explore concepts in Geometry such as angles, plotting coordinates, and symmetry. To solve the problem, students have to think computationally, coordinating representations (visual and numerical) regarding the functionally of the parameters within an algorithm, which can be modified. The nature of the virtual object in construction assumes also an artistic dimension, since colors, shapes, order/pattern, and symmetry are fundamental elements of the construction.

We also highlight two significant (semiotic) aspects potentially prompted by this puzzle in terms of mathematical and computational thinking:

(i) **Aesthetics/artistic thinking.** The puzzle explores direct connections between representations regarding the description-execution actions. For instance, when one configures/describes different values for (x, y), that is, <set x to> and <set y to>, then, the execution changes the initial position of the object, which also can be configured differently. Thus, there is also an artistic/aesthetics component, because the object can be described in different forms, with different colours, and it produces different sounds when executed. These types of direct connections between description and execution were explored by the students from the beginning of the investigation of the first puzzle of the task.

(ii) **Thinking in two dimensions.** The puzzle introduces the need to use nested loops to develop a coherent solution for the puzzle, which enhances the complexity of computational thinking within the heuristic endeavour and opens windows into mathematics. With a "singular" loop we have a linear/unidimensional repetition of a configured core of objectives, forming a sequence of multiple cored objects. The combination of loops, in this case, the elaboration of nested loops, implies a qualitative change of the mathematical-computational construction from unidimensional to two-dimensional representations. Moreover, symmetry plays a significant role towards the connections between mathematical-aesthetic thinking and computational thinking when we consider the visual complementarity of the patterns.

## Results and discussion

Regarding the notion of networking of theories as a strategy to build connections between problem solving/heuristics and constructionism/computational thinking in this study, we are able now to represent connections between principles and methods (see Figure 6) and, thus, pose specific questions related to these principles and methods. We present some research questions of this study in the following subsections.

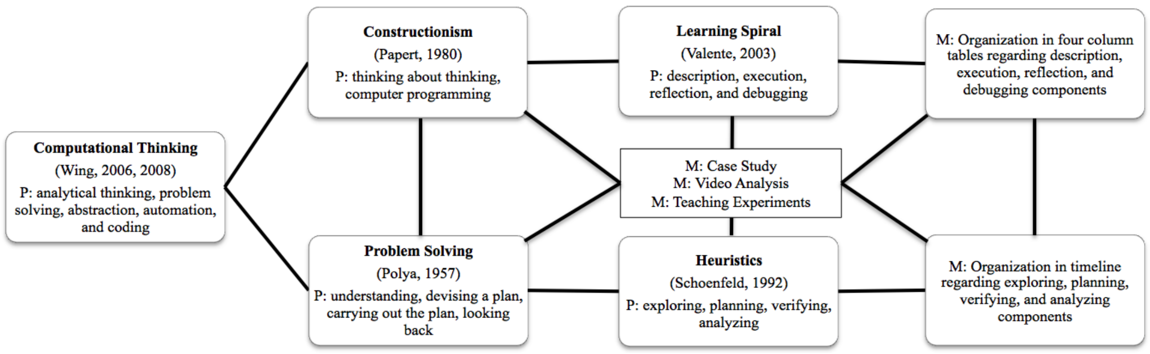

Computational Thinking
(Wing, 2006, 2008)
P: analytical thinking, problem solving, abstraction, automation, and coding
Constructionism
(Papert, 1980)
P: thinking about thinking, computer programming
Problem Solving
(Polya, 1957)
P: understanding, devising a plan, carrying out the plan, looking back
Learning Spiral
(Valente, 2003)
P: description, execution, reflection, and debugging
M: Case Study
M: Video Analysis
M: Teaching Experiments
Heuristics
(Schoenfeld, 1992)
P: exploring, planning, verifying, analyzing
M: Organization in four column tables regarding description, execution, reflection, and debugging components
M: Organization in timeline regarding exploring, planning, verifying, and analyzing components

**Figure 6 – Relations between principles and methods in this study's framework**

http://dx.doi.org/10.1590/1980-6248-2018-0034

## Constructionist actions: elementary school students

*Q: How did Grade-6 students explore the task from a constructionist point of view? What are some of the main aspects of students' learning experience?*

In this research, 3 pairs of Grade-6 students explored the proposed task in four sessions of 1 hour each. Three teachers also participated and each one of them worked with one pair of students. We took notes about the students-teacher-computer interactions, saved students' final solutions and interviewed them at the end of each 1-hour session. Following, we present the beginning of the dialogue just after one pair of students claim they had solved the last puzzle of the task.

**Teacher:** How did you get this second pattern?

**Student A:** This pattern was one of the most difficult we constructed. But, together, we understood the first part of the pattern was already done. Then, we just had to invert it. We tried many times. We tested many different parameters of the commands.

**Teacher:** So, you guys have two patterns here?

**Student B:** Yes. Here at the first pattern. We have 5 lines. The first pattern has 5 lines. The second one we constructed has 4 lines.

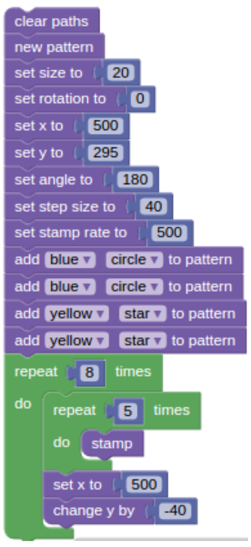
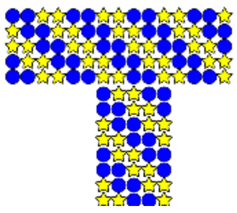
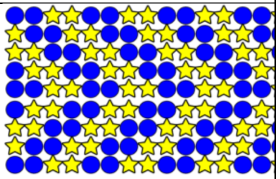
**Teacher:** What are the similarities?

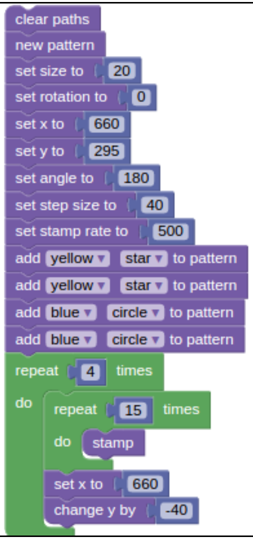
**Student A:** The order of the blocks is the same, but they are opposed. I mean, the second pattern reverses the first one. The angle changes.

We identify in the dialogue above significant aspects of mathematical thinking with computers in problem solving. Statements such as "we tried many times" and "we tested many times" reveals the emergence of exploration as a relevant heuristic activity (Schoenfeld, 1992). Authors such as Borba and Villarreal (2005) explore these aspects as experimentation-with-technology, due to the role of technology in mathematical thinking. Student B's statement, for instance, might be understood as referring to different types of problem solving components such as planning, exploration, and verification. Finally, we also identify analytic endeavours in Student A's last speech. Therefore, we found some evidence that this process involving computational-mathematical thinking offered ways for students to get heuristically engaged in solving a problem with computer programming. We also acknowledge a methodological limitation to analyze in depth students' heuristics actions in this case, because we did not video record the actual experiment with Grade-6 students. We (only) analyzed students' reports of their explorations through interviews and field notes, but these procedures were sufficient to construct graphs of their problem solving processes.

Regarding our constructionist approach (Papert, 1980), based on Valente's (2003) learning spiral metaphor and components, we present in Chart 4 each of the Grade-6 students' actions, including transcriptions of the dialogues in terms of reflection/debugging actions toward the construction of attempts of solutions in coding pattern 2 based on pattern 1.

**Chart 4 – Grade-6 students' constructionist actions**

| Solution | Reflection and Debugging | Description | Execution |
|---|---|---|---|
| **First attempt of solution** | **Teacher:** How did you solve the puzzle?<br>**Student A:** First, we were trying to repeat the pattern starting from the left to the right. Then, we decided to start from the right to the left.<br>**Teacher:** How did that work?<br>**Student B:** In our first attempt, we figured out we had to change the position, that is, x and y. So, our first try was 500 and 295. Then, we figured out that to start from right to left, we had to change the angle from 0 to 180. Also, we changed the parameters at the repeating command like this [point out the finger to the parameters of the command], because the first patterns had 5 lines and the second had 4 lines.<br>**Teacher:** Did that work?<br>**Student B:** Not exactly, you see? | <br>clear paths<br>new pattern<br>set size to 20<br>set rotation to 0<br>set x to 500<br>set y to 295<br>set angle to 180<br>set step size to 40<br>set stamp rate to 500<br>add blue circle to pattern<br>add blue circle to pattern<br>add yellow star to pattern<br>add yellow star to pattern<br>repeat 8 times<br>do<br>  repeat 5 times<br>  do stamp<br>  set x to 500<br>  change y by -40 |  |
| **Final solution** | **Teacher:** How did you fix it?<br>**Student A:** We adjusted the initial x position to 660 and we changed the sequence of blocks from blue/circle, blue/circle, yellow/star, yellow/star to yellow/star, yellow/star, blue/circle, blue/circle. Do you see? We made the sequence of blocks opposite.<br>**Teacher:** Great! Anything else?<br>**Student B:** Yes! We also changed the x position within we repeat command to 660.<br>**Teacher:** And why is the y position -40 within the repeat command?<br>**Student A:** We did not know at first, but after trying a couple of times we discovered… | |  |

| | | | |
|---|---|---|---|
| | **Student A and Student B:** …it is because the step size is 40.<br>**Teacher:** Great! You guys figured out the puzzle! | <br>clear paths<br>new pattern<br>set size to 20<br>set rotation to 0<br>set x to 660<br>set y to 295<br>set angle to 180<br>set step size to 40<br>set stamp rate to 500<br>add yellow star to pattern<br>add yellow star to pattern<br>add blue circle to pattern<br>add blue circle to pattern<br>repeat 4 times<br>do repeat 15 times<br>do stamp<br>set x to 660<br>change y by -40 | |

The Grade-6 students' solution reveals the elaboration and testing of conjectures regarding the initial position (x, y) and design/order of the sequence of blocks or shapes. In this case, students decided to create Pattern 2 starting immediately above Pattern 1. For this, they had to change the direction and the design of the sequence of blocks: from 0 to 180 degrees and the sequence of colors of shapes from blue/blue/yellow/yellow to yellow/yellow/blue/blue. Actually, they could have improved their solution by displaying <set y to 300>. If they had done that, the small space between Pattern 1 and Pattern 2 (between lines 5 and 6) would not exist. Therefore, we found one minor imprecision in elementary school students' final solution. We also highlight that Pattern 2 is (correctly) formed by 4 lines of blocks, instead of 5 lines such as in Pattern 1.

We identified problem solving heuristics processes regarding students' description of their exploration as well. The problem solving endeavour by the students developed along the constructionist spiral components and involves the main four problem solving processes: exploration, verification, analysis, and planning. However, methodologically, we did not produce data about students' actual investigation of the puzzle. That is, we only used students' descriptions about what they have done through interviews. In this sense, we are not able to represent students' problem solving processes in a graph as Schoenfeld (1992) did. In contrast, in our experience with undergraduate students in this research, we were able to analyze and discuss both the constructionist and the problem solving components of students' mathematical-computational exploration.

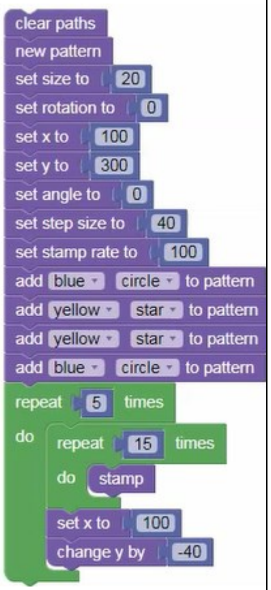
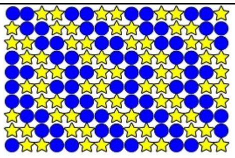
http://dx.doi.org/10.1590/1980-6248-2018-0034

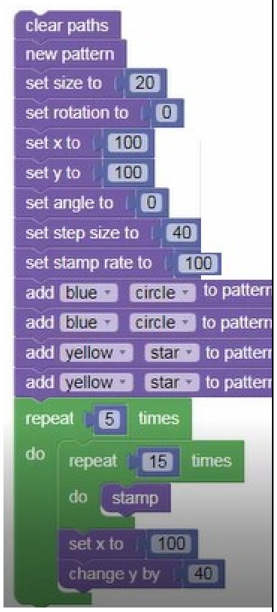
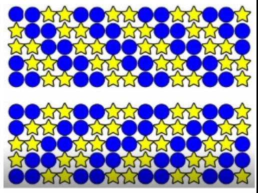
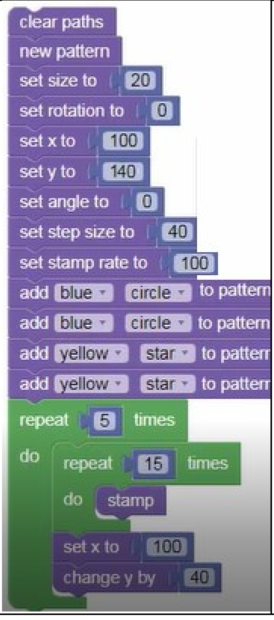
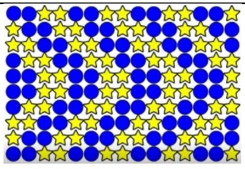
## Constructionist actions: undergraduate students

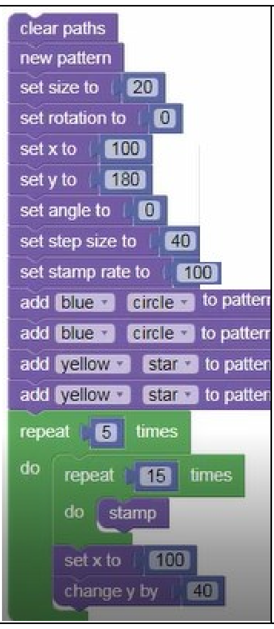
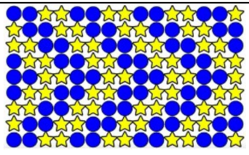
*Q: How did undergraduate students explore the task from a constructionist point of view? What are some of the main aspects of students' learning experience?*

Regarding the undergraduate students' solution, we highlight the possibility of figuring out different solutions for this puzzle on the repeating patterns activity, in comparison to Grade-6 students. As shown in Chart 5, the majors in mathematics found a solution after 3 attempts, and created pattern 2 using a different sequence of blocks (blue/yellow/yellow/blue) and (100, 180) as the initial (x, y) position. In contrast to the Grade 6 students, they did not change the angle in relation to pattern 1. Their solution involved the construction of the second pattern, which in combination with the given pattern, resulted in a solution.

**Chart 5 – Undergraduate students' constructionist actions**

| Solution | Reflection and Debugging | Description | Execution |
|---|---|---|---|
| **First attempt of solution** | **Teacher:** Now you guys must construct this pattern, ok?<br>**Student C:** Do I have to change this? [point at the *change y by -40* command].<br>**Teacher:** You already made the pattern above, which goes until the middle. What do you guys have to change over there?<br>*Students explore the x and y position commands.*<br>**Teacher:** For each line, how long is it?<br>**Student C:** What do you mean?<br>**Teacher:** From the first to the second line, how is it changing?<br>**Student C and Student D:** 40 [refers to step size].<br>**Student D:** We must consider 5 times 40, because we have 5 lines.<br>**Teacher:** So, what happens with the sequence?<br>**Student C:** It changes.<br>**Teacher:** Right. It was blue, blue, yellow, yellow. And now…?<br>**Student C:** We will try blue, yellow, yellow, blue. But look. It is not supposed to be 40 there [step size]. Each line takes 40. So, I think it is -200, because we want 5 times 40 below. | <br>clear paths<br>new pattern<br>set size to 20<br>set rotation to 0<br>set x to 100<br>set y to 300<br>set angle to 0<br>set step size to 40<br>set stamp rate to 100<br>add blue circle to pattern<br>add yellow star to pattern<br>add yellow star to pattern<br>add blue circle to pattern<br>repeat 5 times<br>do repeat 15 times<br>do stamp<br>set x to 100<br>change y by -40 |  |

| | | | |
|---|---|---|---|
| | **Teacher:** Well, are you sure you should *change y by -200* within the repeat command?<br>**Student D:** I see! We must change the original y position from 500 to 300.<br>**Student C:** Let's see what we get. | | |
| **Second attempt of solution** | **Teacher:** Something is missing!<br>**Student C:** The directions of the diagonals are wrong too.<br>**Student D:** But we did not change anything in relation to the first pattern.<br>**Teacher:** The position is correct, right?<br>**Student C:** Only the first line of pattern 2 is right.<br>**Teacher:** What if you start pattern 2 from another position?<br>**Student C:** Let us change y to 100, change y by 40 and…<br>**Student D:** … and put the sequence as blue, blue, yellow, yellow. | <br>clear paths<br>new pattern<br>set size to 20<br>set rotation to 0<br>set x to 100<br>set y to 100<br>set angle to 0<br>set step size to 40<br>set stamp rate to 100<br>add blue circle to pattern<br>add blue circle to pattern<br>add yellow star to pattern<br>add yellow star to pattern<br>repeat 5 times<br>do repeat 15 times<br>do stamp<br>set x to 100<br>change y by 40 |  |
| **Third attempt of solution** | **Student C:** Oops! There is one 40 over the top.<br>**Student D:** Let us try 140? | <br>clear paths<br>new pattern<br>set size to 20<br>set rotation to 0<br>set x to 100<br>set y to 140<br>set angle to 0<br>set step size to 40<br>set stamp rate to 100<br>add blue circle to pattern<br>add blue circle to pattern<br>add yellow star to pattern<br>add yellow star to pattern<br>repeat 5 times<br>do repeat 15 times<br>do stamp<br>set x to 100<br>change y by 40 |  |

| | | | |
|---|---|---|---|
| **Final solution** | **Student C:** it is still wrong. Look! What if we take off another 40? **Teacher:** Try it. **Student D:** Then one line will overlap another. **Student C:** Change y to 180. **Student D and Student C:** Beautiful! |  clear paths<br>new pattern<br>set size to 20<br>set rotation to 0<br>set x to 100<br>set y to 180<br>set angle to 0<br>set step size to 40<br>set stamp rate to 100<br>add blue circle to pattern<br>add blue circle to pattern<br>add yellow star to pattern<br>add yellow star to pattern<br>repeat 5 times<br>do repeat 15 times<br>do stamp<br>set x to 100<br>change y by 40 |  |

In the first attempt of solution, the focus of students' strategies was on (x, y) initial position for the beginning of the sequence of figures. They figured out that each line of the sequence of figures referred to 40 unities in position. Thus, they conjectured that they had to consider 200 units, because there were five lines (40×5=200). Moreover, since the initial position of y was 500 in pattern 1, and they would like to move the sequence bottom down in pattern 2, they described <set y to 300>, because 500–200=300. However, they did not change the configuration of the original sequence of figures. Thus, through the computer execution, they verified their first strategy, and decided to start a new attempt at a solution.

The second attempt at a solution started with students' verification that the first one was wrong. Consequently, they started off the reflect and debug aspects of their mistakes about the first attempt of solution. They identified, for instance, that the directions of the diagonals were wrong, that is, in comparison to the design of the image given in the task, the constructed image was different. This process of visualization through thinking-with-media, characterized in this case as a debugging component within a constructionist view, might be part of an exploration process in heuristics or problem solving. Then, they started to analyze/reflect about their first solution based on what was given by the computer (execution), in combination to their description/planning approach. Although the teacher had mentioned the accuracy in changing (x, y) position within the students' first attempt of solution, they decided to configure both (x, y) position in two different levels and the

http://dx.doi.org/10.1590/1980-6248-2018-0034

color/shape of the sequence that would generate the sequence of objects or shapes. Thus, their description/planning action was to configure the initial position as (x, y)=(100, 100) and the parameter <change y by 40> within the nested loop, instead of -40. The execution of these commands displayed a space between the figures generated by both patterns in combination. Quickly, they re-configured a new description as their third attempt solution or strategy, in which they change the (x, y) initial position from (100, 100) to (100, 140).

Finally, the undergraduate students recognized that the combination of images generated by the patterns was not correct yet. Visually, they identified that the current symmetry was not the same as that given/asked in the task. Thus, the final solution was interesting in the sense that students had a conjecture and conducted their plan overlapping the lines of sequences of shapes on the symmetry axis. To do that, they added 40 units for y in the initial (x, y) position, that is, they configured that to (100, 180). After describing and executing that, students arrived at their correct final solution.

## Heuristic endeavours

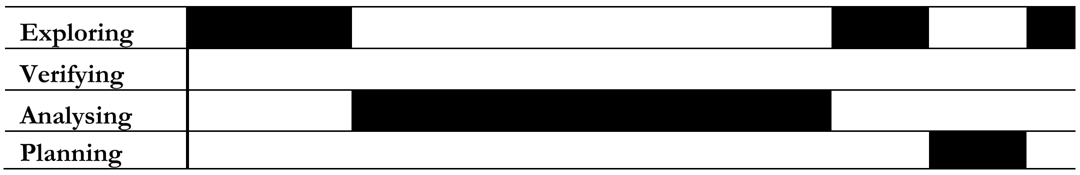
*Q: How did Grade-6 and undergraduate students explore the task from a heuristic point of view? What are some of the main aspects of students' learning experience?*

Using Schoenfeld's (1992) aspects of heuristics processes in problem solving, we identify through the learning spiral (Valente, 2003) an interesting process of thinking-with-technology. We integrated to a heuristic endeavour the instructional role of teachers on students' problem solving. We see this scenario as collectives of students-teacher-media (Borba & Villarreal, 2005) as producers of mathematical meaning through computational thinking. In Charts 6-7 and 8-11, we display a graph to indicate the heuristic processes involved on the students' learning spiral regarding an adaptation of Schoenfeld's (1992) categories: exploring, planning, verifying, and analysing. The construction of these graphs is theoretically and methodologically supported by studies such as Schoenfeld (1992) and Wilkerson (2016). Overall, the selected episode of data took 20 minutes for Grade-6 students and 22 minutes for undergraduate students.

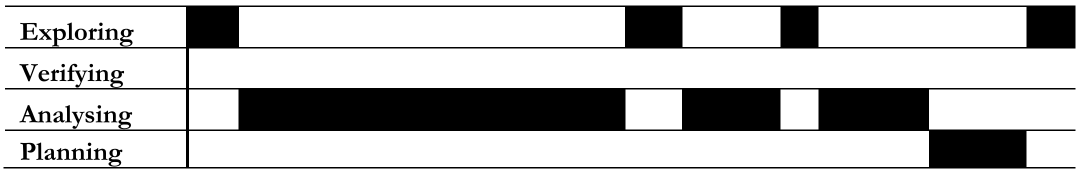
pro·posições
*e*-ISSN 1980-6248
http://dx.doi.org/10.1590/1980-6248-2018-0034

**Charts 6-7: Grade-6 students' heuristic processes/components**

**First attempt of solving – Grade-6 students (10 min)**



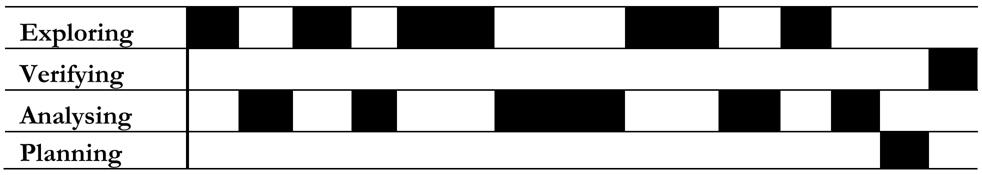
**Final solution – Grade-6 students (10 min)**



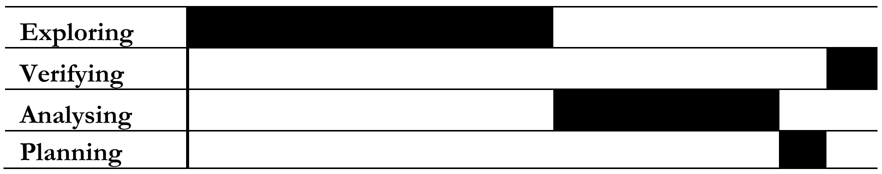
From these representations, we identified that Grade-6 solved the problem in their second attempt. In the first attempt, there was not an often variation of heuristic components, that is, we only identified one moment of exploration and a single and long moment of analysis, before exploration, planning, and exploration. In the final attempt, we identified three moments of exploration and three (longer) moments of analysis, before planning and verification. In this final situation, exploration components were shorter than analysis, which took about 7 of 10 minutes through the dynamic. As we discuss next, undergraduate students' exploration was qualitatively different not only from a constructionist point of view, but from a heuristic point of view as well.

**Charts 8-11: Majors' heuristic processes/components**

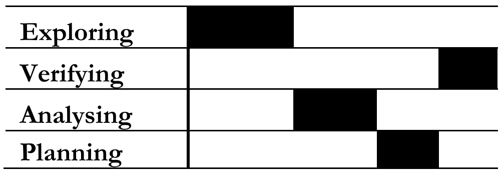
**First attempt of solving – undergraduate students (8 min)**



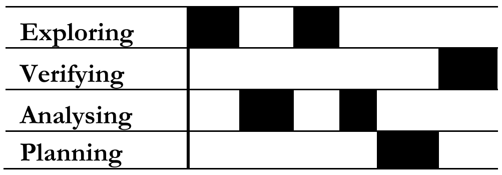
**Second attempt of solving – undergraduate students (6 min)**



**Third attempt of solving – undergraduate students (4 min)**



**Final solution – undergraduate students (4 min)**



These graphs offer some evidence toward the nature of students' heuristic processes in thinking mathematically with coding. The established connection between constructionism and problem solving revealed similarities between debugging and exploration and between reflection and analysis. Thus, the analyzed episode shows that these processes were predominant throughout students' thinking and learning experiences in solving the problem. In attempt one, we identified a significant variance between exploring and analysing along time before a key provisory ending moment of planning (description) and verification (execution). In the second attempt, there is a smaller number of variations of exploration and analysis, but the exploration took longer compared to the first attempt. In the third attempt, such as in the second one, there was only one variation between exploration and analysis. Finally, in the last strategy of solution, we identified two variations between exploration and analysis. The reciprocity between computational and mathematical thinking is conceived as a "rich" scenario for students' learning experiences, because we pointed out the existence of all the heuristic endeavours considered, regarding variations between them. According to

http://dx.doi.org/10.1590/1980-6248-2018-0034

Schoenfeld (1992), the richness in heuristic components in solving problems refers to ways mathematicians think. Thus, we argue these types of approaches, like the one developed in this research, may offer ways to open windows into (advanced) mathematical thinking (Tall, 1991).

> Using this style of approach, it is possible to get undergraduates to develop original ways of solving problems. The solutions may not be found as quickly as they might, given active teaching by the lecturer but the activities can help the students gain in confidence and desire to attack problems that they might previously have been unwilling to attempt. Such problem-solving activities can also help to stimulate reflective thinking and to develop an internal monitor within the student's mind to help keep track on the progress of the solution process and to ring warning bells when the solution may be leading up a blind alley. (Tall, 1991, p. 12)

## Conclusions

Our findings show that for both young and older students, the exploration of the patterning computational-mathematical tasks illustrated learning experience as a spiral process that also revealed problem solving components. Students' learning actions that involved computer interaction, specifically thinking about and changing code showed the description-execution-reflection-debugging-description stages. Each action took place not one at a time and not in a sequential order. Moreover, certain qualitative differences regarding the stages of exploring the problem between young children and young adults were evident. We identified the description and execution actions as students' inputs of commands and computer's executions, respectively. However, reflections and debugging actions appeared to happen simultaneously, through students' dialogues and interactions during varied attempts at solving the problem. Indeed, these processes of dynamicity/complexity in computational thinking seem to be similar and directly related to the main components of problem solving of exploration, planning, analysis, and verification that are fundamental components of heuristics related to computational thinking. Although these components are also part mathematical doing and learning, there is a difference, and an enhancement, made possible by the ability to model mathematical processes and relationships with code. These models through code enhance the ability to explore and to verify, for example, through different attempts of solving the problem.

http://dx.doi.org/10.1590/1980-6248-2018-0034

Students' learning experiences in this study involved mathematical contents/processes and computational thinking skills. In terms of mathematical content, they explored concepts of Geometry such as angles, plotting coordinates, symmetry, and sequences of colored shapes. In terms of processes, they were immersed in a constructionist environment built from the exploration of a computer task for solving a coding problem. In terms of computational thinking, the heuristics identified components revealed the analytic nature of thinking. In this case, thinking computationally involved mathematical, computational, and artistic elements. We also must highlight the connections between these types of representations conducted by the students in thinking computationally.

Computational thinking was a relevant educational trend for teaching and learning mathematics during the 1980's, primarily based on Papert's (1980, 1993) work. This trend has become significant again today due to the development of new, visual coding environments that allow even very young students to engage meaningfully with computer programming. In this study, students solved a computational-mathematical problem through a process of elaborating and testing conjectures, that is, experimentation-with-technology, and thinking-with-media (Borba & Villarreal, 2005). The goal of the study was to explore how elementary school students and undergraduate mathematics students engaged with a mathematics task using a coding environment. We noted that students at both levels engaged in constructionist practices and enhanced learning by using computer programming to build and control dynamic models of mathematical relationships. A difference between the two levels of students emerged in the final, more complex task. Regarding undergraduate students, we identified more attempts of solving the problem and a higher variation of heuristic components when comparing to Grade-6 students. That is considered a qualitative difference between the ways in thinking mathematically and acting computationally between the two groups of students. We see both of these groups' ways of thinking as complex, but the complexity involving undergraduate students seems more sophisticated in terms of heuristics.

http://dx.doi.org/10.1590/1980-6248-2018-0034

## References

Artigue, M., & Mariotti, M. A. (2014). Networking theoretical frames: the ReMath enterprise. *Educational Studies in Mathematics*, *85*(3), 329-355.

Bikner-Ahsbahs, A. (2010). Networking of theories: why and how? In V. Durand-Guerrier, S. Soury-Lavergne, & F. Arzarello (Eds.), *Proceedings of the Sixth Congress of the European Society for Research in Mathematics Education* (pp. 6-15). Institut National de Recherche Pédagogique, Paris.

Bikner-Ahsbahs, A., Prediger, S. (2010). Networking of theories: an approach for exploiting the diversity of theoretical approaches. In B. Sriraman, & L. English (Eds.), *Theories of mathematics education: seeking new frontiers* (pp. 483-506). Berlin: Springer.

Borba, M. C., & Villarreal, M. E. (2005). *Humans-with-media and the reorganization of mathematical thinking: information and communication technologies, modeling, visualization and experimentation*. New York: Springer.

Denning, P. J. (2017) Remaining trouble spots with computational thinking. *Communications of the ACM*, *60*(6), 33-39.

Gadanidis, G. (2015). Young children, mathematics, and coding: a low floor, high ceiling, wide walls environment. In D. Polly (Ed.), *Cases on technology integration in mathematics education* (pp. 308-329). Hershey: IGI Global. doi:10.4018/978-1-4666-6497-5.ch015

Gadanidis, G., & Yiu, C. (2017). Repeating patterns + code + art. *Math surprise: conceptual, emotional, engaging*. Retrieved from http://researchideas.ca/patterns/

Idem, R. C. (2017). *Construcionismo, conhecimentos docentes e GeoGebra: uma experiência envolvendo licenciandos em matemática e professores*. Masters' Thesis, Universidade Estadual Paulista, Rio Claro.

Lu, J. J., & Fletcher, G. H. L. (2009). Thinking about computational thinking. *ACM SIGCSE Bulletin*, *41*(1), 260-264.

Maltempi, M. V. (2005). Novas tecnologias e construção de conhecimento: reflexões e perspectivas. In *Anais do V Congresso Ibero-americano de Educação Matemática*. Associação de Professores de Matemática de Portugal, Porto.

http://dx.doi.org/10.1590/1980-6248-2018-0034

Marshall, M. N. (1996). Sampling for qualitative research. *Family Practice*, *13*(6), 522-526.

Ontario Ministry of Education. (2005). *The Ontario curriculum grades 1-8: mathematics*. Retrieved from www.edu.gov.on.ca/eng/curriculum/elementary/math18curr.pdf

Papert, S. (1980). *Mindstorms: children, computers, and powerful ideas*. New York: Basic Books.

Papert, S. (1993). *The children's machine:* rethinking school in the age of the computer. New York: Basic Books.

Polya, G. (1957). *How to solve it: a new aspect of mathematical method* (2a ed.). Garden City: Doubleday Anchor.

Powell, A. B., Francisco, J. M., & Maher, C. A. (2003). An analytical model for studying the development of learners' mathematical ideas and reasoning using videotape data. *Journal of Mathematical Behavior*, *22*(4), 405–435.

Schoenfeld, A. H. (1985). *Mathematical problem solving*. New York: Academic Press.

Schoenfeld, A. H. (1992). Learning to think mathematically: problem solving, metacognition, and sense-making in mathematics. In D. Grouws (Ed.), *Handbook for research on mathematics teaching and learning* (pp. 334-370). New York: MacMillan.

Stake, R. E. (2000). Case studies. In N. K. Denzin, & Y. S. Lincoln (Eds.), *Handbook of qualitative research* (2a ed., pp. 435-453). Thousand Oaks: Sage.

Steffe, L. P., & Thompson, P. W. (2000). Teaching experiment methodology: underlying principles and essential elements. In R. Lesh, & A. E. Kelly (Eds.), *Research design in mathematics and science education* (pp. 267-307). Hillsdale: Erlbaum.

Tall, D. (1991). The psychology of advanced mathematical thinking. In D. Tall (Ed.), *Advanced mathematical thinking* (pp. 3-21). Dordrecht: Springer.

Valente J. A. (2003). In service teacher development using ICT: first step in lifelong learning. In C. Dowling, & K.-W. Lai (Eds.), *Information and communication technology and the teacher of the future* (pp. 97-108). Boston: Springer.

Wilkerson, M. H., Andrews, C., Shaban, Y., Laina, V., & Gravel, B. E. (2016). What's the technology for? Teacher attention and pedagogical goals in a modeling-focused professional development workshop. *Journal of Science Teacher Education, 27*(1), 11-33.

http://dx.doi.org/10.1590/1980-6248-2018-0034

Wing, J. M. (2006) Computational thinking. *Communications of the ACM*, *49*(3), 33-35.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 366*(1881), 3717-3725.

***Submission data:*** *Submitted to evaluation on April 9th, 2018; revised on November 6th, 2018; accepted for publication on January 3rd, 2019.*

***Corresponding author:*** *Ricardo Scucuglia Rodrigues da Silva- Unesp (Education) - Rua Cristóvão Colombo, 2265, São José do Rio Preto, SP, Brazil, 15054-000*