

## Computational Literacy & Mathematics Education

### George Gadanidis

Western University  
London — Ontario, Canada

✉ [ggadanid@uwo.ca](mailto:ggadanid@uwo.ca)

 0000-0002-5982-6056

### Ricardo Scucuglia Rodrigues da Silva

Universidade Estadual Paulista  
São Jose do Rio Preto — SP, Brazil

✉ [ricardo.scucuglia@unesp.br](mailto:ricardo.scucuglia@unesp.br)

 0000-0002-5810-2259

### Janette M. Hughes

Ontario Tech University  
Oshawa — Ontario, Canada

✉ [janette.hughes@ontariotechu.ca](mailto:janette.hughes@ontariotechu.ca)

 0000-0002-3463-8382

### Immaculate Namukasa

Western University  
London — Ontario, Canada

✉ [inamukas@uwo.ca](mailto:inamukas@uwo.ca)

 0000-0001-7231-0671


### Steven Floyd


Western University  
London — Ontario, Canada

✉ [sfloyd@uwo.ca](mailto:sfloyd@uwo.ca)

0000-0001-8793-3142 



2238-0345 

10.37001/ripem.v12i4.3144 

Recebido • 13/06/2022

Aprovado • 09/11/2022

Publicado • 19/11/2022

Editor • Gilberto Januario 

**Abstract:** Computer programming has permeated many fields as a tool to model phenomena and processes and to make new discoveries. Curricula in many jurisdictions have been revised to use computer languages across K-12, and in some cases in mathematics education. The literature suggests that while digital media in mathematics education can be used as tools that serve our purposes, they also serve to reorganize knowledge. This paper investigates the interactions among computer programming and mathematics teaching and learning. Our data sources are a) Ontario curriculum documents, b) resources developed by our team in Ontario and in Brazil, and c) our research in Ontario and Brazil. diSessa's idea of computational literacy serves as a theoretical framework and as an analytical lens for conceptualizing how the integration of computer programming may alter the structure and experience of school mathematics.

**Keywords:** Mathematics Education. Mathematics. Computational Literacy. Computer Programming.

## Alfabetización Computacional y Educación Matemática

**Resumen:** La programación computacional ha permeado muchos campos como herramienta para modelar fenómenos y procesos y para hacer nuevos descubrimientos. Los planes de estudio en muchas jurisdicciones se han revisado para usar lenguajes informáticos en K-12 y, en algunos casos, en educación matemática. La literatura sugiere que si bien los medios digitales en la educación matemática pueden usarse como herramientas que sirven a nuestros propósitos, también sirven para reorganizar el conocimiento. Este artículo investiga las interacciones entre la programación computacional y la enseñanza y el aprendizaje de las matemáticas. Nuestras fuentes de datos son a) documentos curriculares de Ontario, b) recursos desarrollados por nuestro equipo en Ontario y Brasil, y c) nuestra investigación en Ontario y Brasil. La idea de diSessa de alfabetización computacional sirve como marco teórico y lente analítico para conceptualizar cómo la integración de la programación computacional puede alterar la estructura y la experiencia de las matemáticas escolares.

**Palabras clave:** Educación Matemática. Matemáticas. Alfabetización Computacional. Programación de Computadoras.

## Literacia Computacional e Educação Matemática

**Resumo:** A programação de computadores permeou muitos campos como uma ferramenta para modelar fenômenos e processos e para fazer novas descobertas. Os currículos em muitas jurisdições foram revisados para usar linguagens de computador no K-12 e, em alguns casos, na educação matemática. A literatura sugere que, embora as mídias digitais na educação matemática possam ser usadas como ferramentas que atendem aos nossos propósitos, elas também servem para reorganizar o conhecimento. Este artigo investiga as interações entre programação de computadores e ensino e aprendizagem de matemática. Nossas fontes de dados são a) documentos curriculares de Ontário, b) recursos desenvolvidos por nossa equipe em Ontário e no Brasil, e c) nossa pesquisa em Ontário e no Brasil. A ideia de literacia computacional de diSessa serve como uma estrutura teórica e como uma lente analítica para conceituar como a integração da programação de computadores pode alterar a estrutura e a experiência da matemática escolar.

**Palavras-chave:** Educação Matemática. Matemática. Literacia Computacional. Programação.

### 1 Introduction

Computational literacy (diSessa, 2000, 2018) has become a tool in many fields of study (for example, computational biology, finance, and medicine) through use of computer programming languages to model phenomena and processes. Meanwhile, curricula in many jurisdictions have been revised to use computer programming across K-12, and in some cases in mathematics education (such as, France, Sweden, Finland and Canada).

Levy (1997) notes that a cognitive ecology develops when we use digital tools, creating a space of interactions, dialogues and relationships among actors that both exercise agency and are transformed due to the agency of others. Borba & Villarreal (2005) suggest that digital media in mathematics education are not simply tools that serve our purposes, as they also reorganize knowledge as we use them. Noss and Hoyles (1992) note that “there are tasks that children can do with a computer that would be impossible without one” (p. 455).

This paper investigates the interactions between computer programming and mathematics teaching and learning. We are especially interested in conceptualizing the ways and the extent to which computer programming in mathematics education may play a role in altering the structure and experience of school mathematics. Doyle (1992) identifies three levels of curriculum: institutional, programmatic, and classroom. The institutional curriculum is broad, encompassing belief systems and located at the intersection of schooling, culture and society. The classroom curriculum involves the learning experiences and activities within the schools. Between this institutional and classroom curriculum is the programmatic curriculum, such as policy and syllabi that organize the subject content. In this paper we focus on two of those levels — classroom (computer programming) and programmatic (curriculum expectations) — and their role in the development and impact of a computational literacy. Our data sources include a) curriculum documents, b) computer programming resources developed by our team in Ontario and in Brazil, and c) our research in Ontario and Brazil classrooms.

We chose to focus on the recent grades 1-9 Ontario, Canada mathematics curriculum for three reasons. First, our classroom resources were initially created with this curriculum in mind. Second, this curriculum explicitly mandates the use of computer programming as a tool with which to represent and model mathematical processes and relationships, thus allowing educators to legitimately view computer programming as a language and a literacy within mathematics education. This also builds on the historical direction of computer programming

in education of integrating computer programming and mathematics, such as Papert (1980), Hoyles & Noss (1987) and Noss & Hoyles (1988, 1992). Third, the new directions of this curriculum parallel the long-standing directions of our own research, with its focus on integrating computer programming and mathematics, highlighting beauty of mathematics, and the aesthetic elements of mathematics education. Some of the authors of this paper have played roles in conducting the background research for the recent Ontario mathematics and computer science curriculum reform and in advising the curriculum writing teams.

While there are different “framings” of computational literacies (Guzdial, 2019; Kafai, Proctor & Lui, 2019; Vakil, 2018), we chose to use diSessa’s (2000, 2018) idea of computational literacy (and his five literacy principles of massive social/intellectual accomplishment, remediation, reformulation, reorganization and revitalization) as a theoretical framework and as an analytical lens for conceptualizing the ways and the extent to which computer programming in mathematics education, along with other resources, may alter the structure and experience of school mathematics. We made this choice as we are interested in exploring what might be if computer programming becomes a literacy within mathematics education, as it already is in many fields in the real world.

The paper is organized in three parts. First, we discuss literacy and offer an analysis of the meaning of diSessa’s five computational literacy principles through references to the Ontario curriculum (a massive social/intellectual accomplishment) and a mathematics task (remediation, reformulation, reorganization and revitalization) focusing on Sumerian (a.k.a. Pythagorean) triples. We chose this task as we have already used it extensively in professional development workshops for educators in Canada. Our analysis is conceptual in nature and uses the approach developed by the Canadian philosopher and poet Jan Zwicky (2003) of understanding through seeing-as, by juxtaposing components of theory (diSessa’s principles of computational literacy) with examples of the theory in practice (Sumerian triples task).

Second, we offer a pilot study where the Sumerian Triples task is shared and discussed with teachers, and their responses are analyzed through the lens of diSessa’s principles of computational literacy. This use of seeing-as, of mirroring, between diSessa’s principles and what they may look like in mathematics education, offers us opportunities to better understand theory and practice, to recognize one in the other.

Lastly, we offer a discussion and concluding remarks on what we have learned through these processes about computer programming in mathematics education.

## 2 Literacy

Wing (2006) defined computational thinking as “solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science” (p. 33). Barba (2016) adds that the power of the computer is taken away when it is seen as a formal tool to first understand and then potentially apply to a problem later:

The operational aspect of making problems computable is essential, but not aspirational. Most people don’t want to be a computer scientist, but everyone can use computers as an extension of our minds, to experience the world and create things that matter to us. (Barba, 2016, p. 23)

diSessa (2018) observes that for something to have as broad aspirations as Wing’s conception of computational thinking, it cannot belong to the single discipline of computer science.

Jenkins (2006), considering the broader category of new media literacy, discusses the danger of reducing media literacy to technical skills, which “would be a mistake on the order of confusing penmanship with composition” (p. 20). Jenkins adds that “Because the technologies are undergoing such rapid change, it is probably impossible to codify which technologies or techniques students must know” (p. 20).

Bogost (2005) sees programming literacy as a subset of procedural literacy. He suggests that procedural literacy is an integral component of many other fields of study. For example, he refers to Diamond (1999) to argue that history is fundamentally procedural, “created out of the configurations of material conditions to produce political, social, and expressive outcomes” (Bogost, 2005, p. 35).

diSessa (2018) defines a literacy “as the adoption by a broad cultural group” — perhaps an entire civilization — of a particular infrastructural representational form for supporting intellectual activities” (p. 4). Literacies develop over time, sometimes centuries. Literacies are unique representational forms, which change what can be done in various fields of study and who can do it.

diSessa relates his experience in reading Galileo’s six proofs related to uniform motion, where he was baffled by the complexity of proportional reasoning used to show that distance = rate  $\times$  time, or  $d = rt$ . The reason Galileo’s proofs were convoluted is that algebra as we know it today was not available. Once algebra developed into a literacy adopted by various disciplines, it introduced cognitive simplicities (diSessa, 2000, 2018) that made the idea of  $d = rt$  explainable and understandable by a large portion of the population.

A literacy, such as algebra, may be seen as a field of study in itself, as well as a representational tool that may be used in other fields of study. The former perspective aligns with current directions supported by Wing (2006, 2008) who primarily identifies the advantages and benefits of learning to think like a computer scientist. The latter perspective is what for the most part constituted the previous historical focus on computer programming in education, as seen, for example, in the work of Papert (1980), Hoyles & Noss (1987) and Noss & Hoyles (1988, 1992), and diSessa (2000), who combined computer programming with mathematics. These views align with our interest in this paper.

When diSessa draws an analogy with Galileo’s work, to show the transformative effect of algebra as a literacy, he situates this effect within the physics of constant motion and within the mathematics used in this part of physics. More generally, diSessa states that “a literacy needs a literature. One needs to transcend a representational system by itself, and get to civilizations’ expanse of deep and powerful ideas” (p. 7). This suggests that computational literacy may be situated within fields of study or literatures. Luszkiwicz and Warfel (2019) similarly distinguish between computational literacy and computational specialization.

Using diSessa’s notion that a literacy needs a literature, computational literacy may be seen as applying computer science concepts and processes to investigate, to understand and to extend knowledge in another field, such as Medicine, Mathematics or Engineering. A literacy brings a representation form from one field as a thinking tool in another.

We may similarly distinguish between mathematics as a discipline and mathematics as a literacy to represent ideas and make progress in other fields. In this light, we may consider the role mathematical literacy plays within computer science, or its application in other fields, as a literacy. For example, Luszkiwicz and Warfel (2019), in their analysis of an extensive case study focusing on a computer program developed for guiding decisions about road maintenance for the Arizona Department of Transportation argue that “the key knowledge barrier between

computational literacy and computational specialization is mathematics” (p. 94). They elaborate that “What the idea of computational literacy doesn’t account for — and one of the many possible skills that differentiates computational literacy from computational specialization — is the mathematical knowledge necessary to understand code itself” (p. 96).

Computer programming as a literacy is also evident in the work of Papert (1980). He proposed that the Logo programming environment would provide young students access to a tool to think with, which would create opportunities to transform both how they learn and what they can learn. Papert viewed Logo as an immersive mathematical environment and suggested that learning mathematics with Logo was analogous to learning French by living in France. Papert had the intuition to design Logo around the movement of a turtle, thus giving Logo programming and the mathematics used to move the turtle, an embodied and tangible feel, and consequently making both computer programming and mathematics more accessible to a wider non-expert audience, including young children.

### 3 Computational Literacy

diSessa (2018) identifies 5 principles that characterize what a literacy does and affects. We define and discuss each of these principles below, in relation to the Ontario context, using the grades 8-9 mathematics task of finding Sumerian (a.k.a. Pythagorean) triples. A Sumerian triple is made of 3 integers that satisfy the property  $a^2 + b^2 = c^2$ . The clay tablet from Sumer in Figure 1 shows 15 rows of Sumerian triples. It is interesting that over 1200 years before Pythagoras, the Sumerians calculated integer triples for the sides of right triangles, such as (119, 120, 169), where  $119^2 + 120^2 = 169^2$ .

**Figure 1:** Plimpton 322 Sumerian triples clay tablet. (Photo author unknown. Public domain due to age).



**Source:** Retrieved from <https://personal.math.ubc.ca/~cass/courses/m446-03/pl322/322.jpg>

How might we go about finding Sumerian triples? The task is to find pairs of integers so that the square root of the sum of their squares is a square number  $c$ . To be efficient, we might use an organized list of pairs  $(a, b)$ . This may also then give us the idea of plotting the  $(a, b)$  pairs of possible Sumerian triples in 2 dimensions. We may consider pairs  $(a, b)$  where  $a$  and  $b$  take on integer values from 1 to 10, and we may wonder if there will be any patterns in the plot of the list of pairs  $(a, b)$  that we find. Perhaps we may consider pairs  $(a, b)$  where  $a$  and  $b$  take on integer values from -100 to 100, or even -1000 to 1000. All of this requires a lot of computation that can be done in a patterned, organized manner, which makes an investigation that uses computer programming suitable.

We will elaborate and discuss the computational task of finding Sumerian triples below as we consider diSessa’s computational literacy principles of remediation, reformulation,

reorganization and revitalization. The Scratch and Python programming samples in the analysis below are publicly available and used and adapted with permission from Gadanidis (2021).

### 3.1 A massive social/intellectual development

“A literacy is a massive social/intellectual accomplishment of a culture or civilization, where many competing forces, over decades or centuries, eventually settle on a particular representational form for wide-spread learning, use, and subsequent value” (diSessa, 2018, 7).

We noted earlier that many of today’s fields have a computer programming side, which makes computer programming a literacy — a “massive social/intellectual development” — in our society. Barba (2014) notes that scientists and professionals use computational practices for solving real-world problems and building knowledge through computational “conversation” and “interaction” with their field. This does not mean that all of society is affected in this way. As diSessa notes about algebra as a literacy:

It is not much good for many cultural functions, for example, for poetry or courting. But it is useful enough, for enough important things, that our prospering and even surviving as a civilization (what is the change in rate of change of global warming?) are implicated. (diSessa, 2018, p. 7)

A case may also be made that computer programming could be seen as a literacy in mathematics education in Ontario, due to the mandated integration of computer programming with mathematics across all grades 1-9, for over 1.7 million students and teachers (Ontario Ministry of Education, 2020, 2021). Table 1 briefly summarizes the computer programming expectations from the Ontario mathematics curriculum, grades 1-9. These expectations are listed in the algebra strand for each grade, with the intention that they will be used “in algebra and across the other strands” (Ontario Ministry of Education, 2021, p.3). Students are expected to develop skills and understanding to read, edit, write and execute code so that they may “solve problems and create computational representations of mathematical situations” (Ontario Ministry of Education, 2020, p. 4 of algebra strand).

**Table 1:** Ontario computer programming expectations, grades 1-9

Grade	Computer programming concepts	Grade	Computer programming concepts
1	Sequential events	5	Conditional statements
2	Concurrent events	6	Efficiency of code
3	Repeating events	7	Defined count and/or subprogram
4	Nested events	8	Analysis of data
9	Apply coding skills to represent mathematical concepts and relationships dynamically, and to solve problems, in algebra and across the other strands		

**Source:** Ontario Ministry of Education (2020)

### 3.2 Remediation

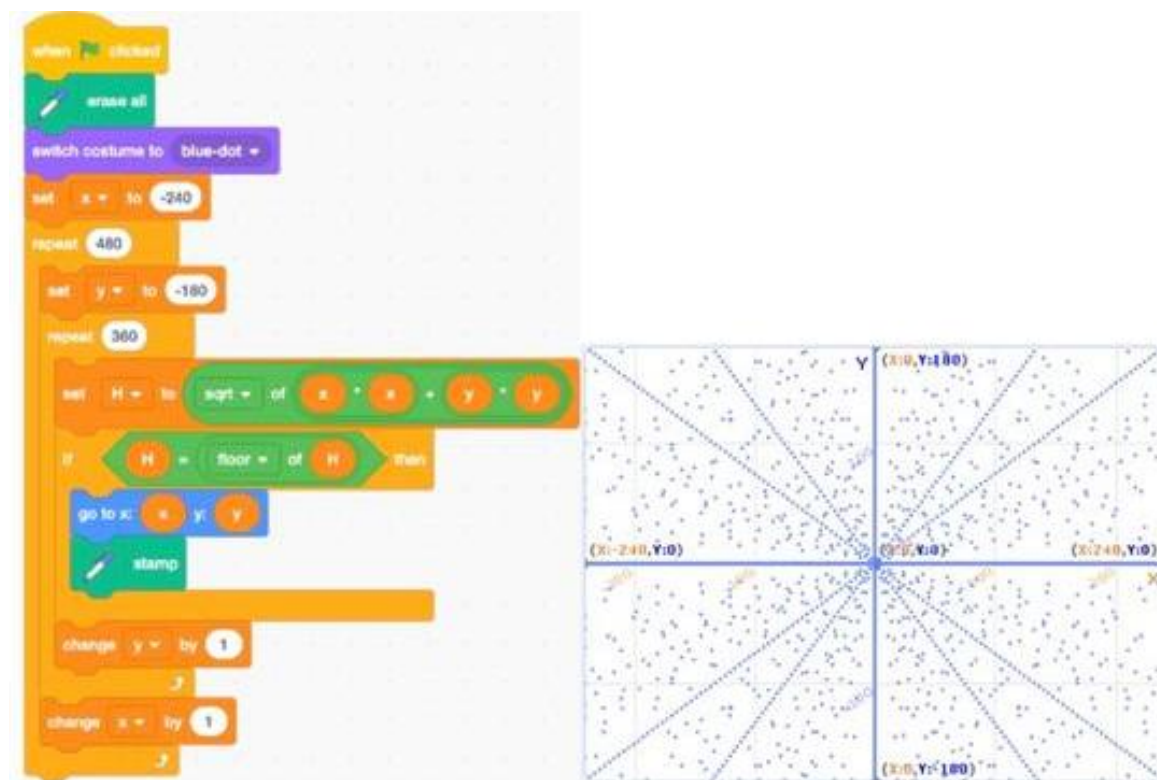
diSessa (2018) notes that a computational literacy remediates ideas of a field, through its ability to dynamically represent concepts and relationships. Remediation of mathematics concepts, problems and processes is an explicit curriculum direction in the grades 1-9 Ontario mathematics curriculum, as students in each of grades 1-9 are expected to use, edit, and create computational representations of mathematical situations by writing and executing computer code (Ontario Ministry of Education, 2020, 2021). Remediation alters the “representational

infrastructure” (diSessa, 2018, 25) available to teachers and students. This is further supported by explicitly listing specific computer programming concepts and processes to be used in each of the grades (see Table 1), which offer dynamic representations and models of mathematics concepts and relationships. Writing more broadly about possible effects in our society, diSessa (2018) notes:

Dynamic and interactive representations on computers, along with the ability to design and enact specialized representations on demand and often quickly means that intellectual changes easily on the scale of what algebra or calculus brought us are almost certainly in the offing. (p. 25)

Computational remediation is evident in the case of the Sumerian triples task discussed above. Using the Scratch code shown in Figure 2, we may compute and plot all integer pairs (a, b) that create Sumerian triples. The dimensions of the Scratch output stage are 480 by 360 pixels, so we may consider integer pairs for  $-240 \leq a \leq 240$  and  $-180 \leq b \leq 180$ . The Scratch code that models these relationships is available at <https://scratch.mit.edu/projects/588754331/editor>.

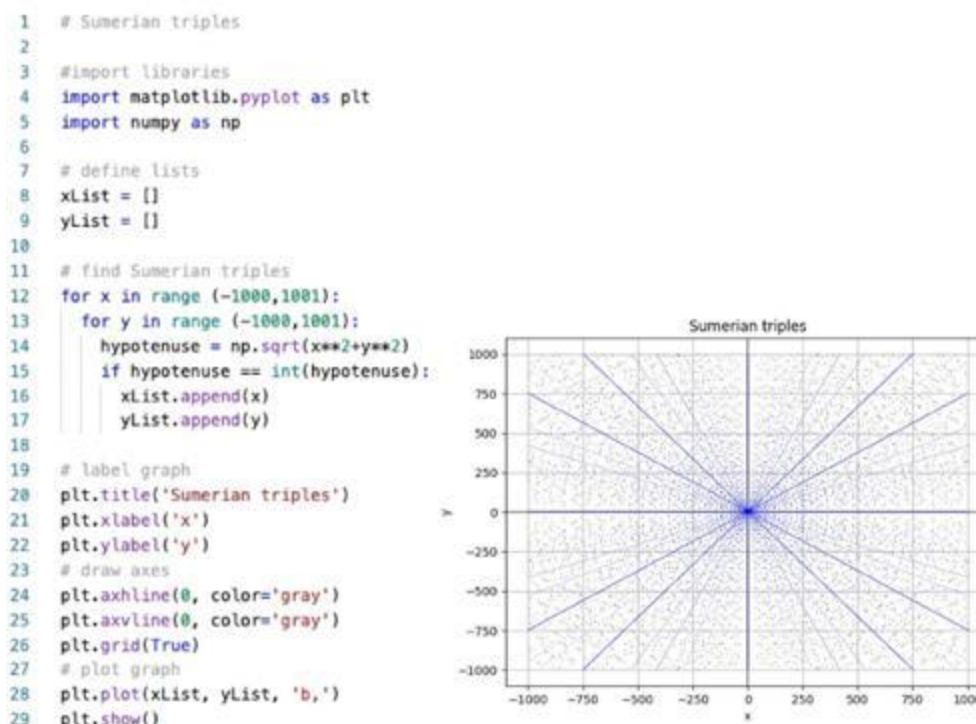
**Figure 2:** Scratch code and output: finding Sumerian triples.



**Source:** Research Data

The output is more flexible if we use Python. Figure 3 shows the Python code and its output for integer pairs (a, b) when for  $-1000 \leq a \leq 1000$  and  $-1000 \leq b \leq 1000$ . The Python code is available at <https://replit.com/@georgeuwo/MCT-sumerian-triples-v1>.

**Figure 3:** Python code and output: finding Sumerian triples.



**Source:** Research Data

Such remediation of the task of finding Sumerian triples fits within the Ontario Mathematics curriculum in grades 8 and 9. In grade 8, students are expected to “describe the Pythagorean relationship using various geometric models and apply the theorem to solve problems involving an unknown side length for a given right triangle” (Ontario Ministry of Education, 2020, p.6 of spatial sense strand). In grade 9, students are expected to “solve problems involving the side-length relationship for right triangles” (Ontario Ministry of Education, 2021, p.6), “research a geometric concept or a measurement system to tell a story about its development and use in a specific culture or community (p. 5), and “represent and analyse data involving one and two variables” (p. 3).

### 3.3 Reformulation

A computational literacy *reformulates* concepts, offering *cognitive simplicities* that lead to *cognitive shift*, where we see mathematics through new perspectives. diSessa (2018) notes that reformulation is

a domain-by-domain issue. One needs to find the productive roots of different ways of thinking about each domain. Certainly, there are some generalities, but so far as I have seen (and a lot of literature agrees), one needs to understand how learners construe particular domains, and how they may profitably construe them differently. (p. 27)

What might reformulation look like in the Ontario context?

In the Sumerian triples example above, we used Scratch and Python to compute and plot integer pairs (a, b) where the square root of the sum of their squares is an integer c. Scratch and Python offer a visual representation of Sumerian triples and dynamic manipulation of parameters, such as the range of values (a, b). This remediation offers a cognitive simplicity

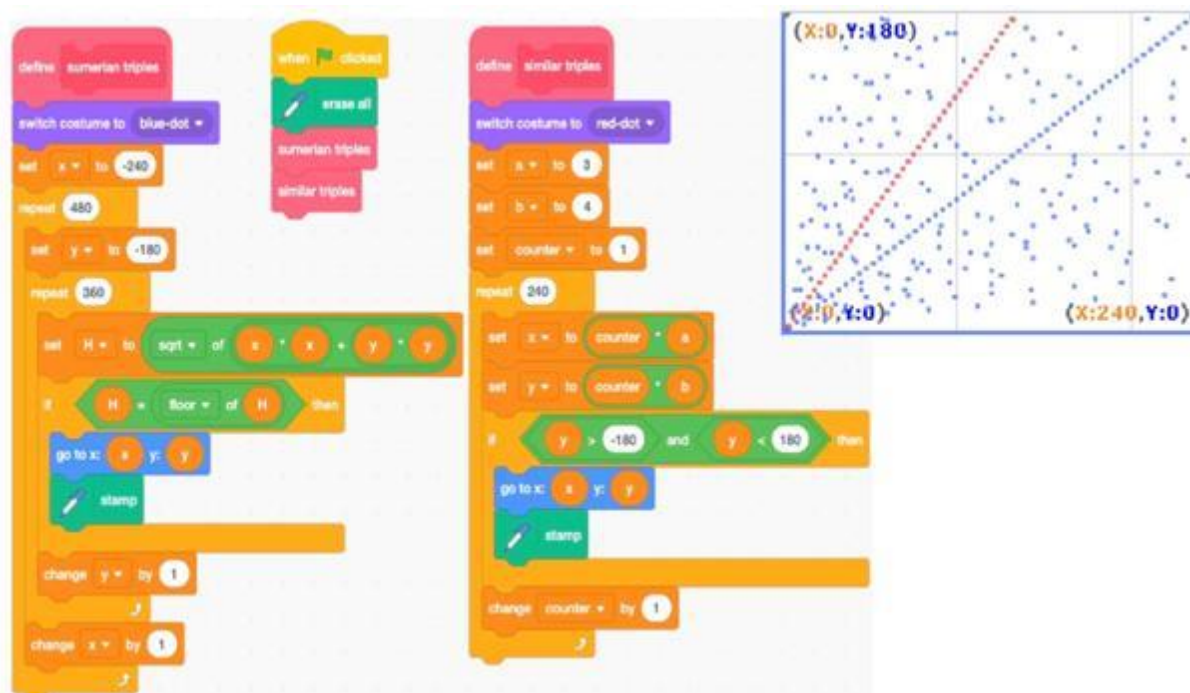


that enables us to visualize patterns in Sumerian triples.

Let's imagine doing this without computer programming. The Scratch stage, for example, is 480 pixels horizontally and 360 pixels vertically, or 172,800 integer pairs (a, b). If we could check 1 integer pair to see if c is also an integer in one minute, it would take us 2,880 hours or 120 days. If we notice the symmetries among and within the 4 quadrants, we could reduce our time to 15 days. Scratch offers the cognitive simplicity of completing this task, as well as plotting all (a, b) pairs of Sumerian triples, in about 1 minute.

The cognitive simplicity allows a cognitive shift from calculating individual Sumerian triples to noticing visual patterns when numerous (a, b) components of such triples are plotted on a grid. Seeing the problem of finding Sumerian triples through this new perspective allows us to reformulate the questions we may ask about Sumerian triples. For example, looking at the output in Figures 2 and 3, we may wonder about the meaning of the lines we see radiating from or passing through the origin. If we hypothesize that these lines represent groups of similar Sumerian triples, such as (3, 4, 5) and (6, 8, 10), we can then test our hypothesis by plotting a set of similar Sumerian triples using a different colour, as shown in Figure 4.

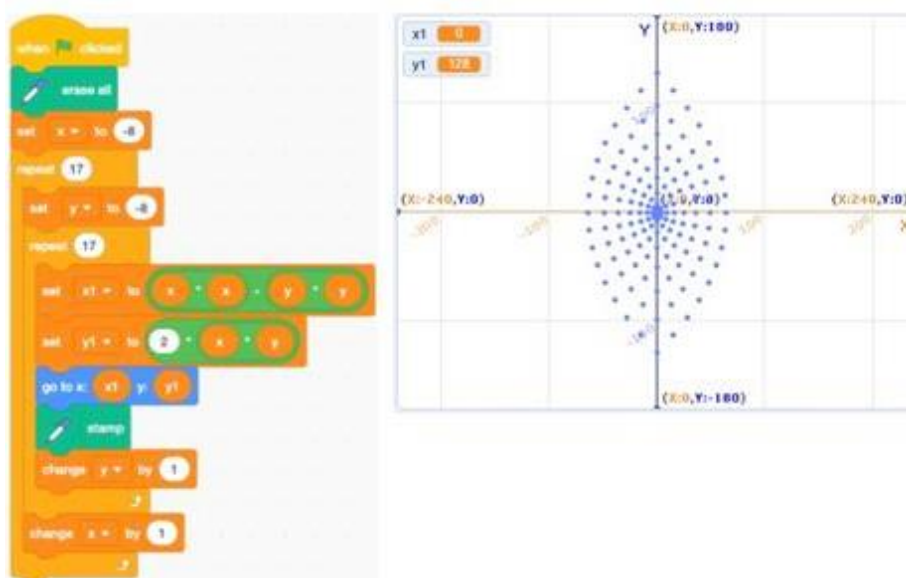
**Figure 4:** Similar Sumerian triples.



**Source:** Research Data

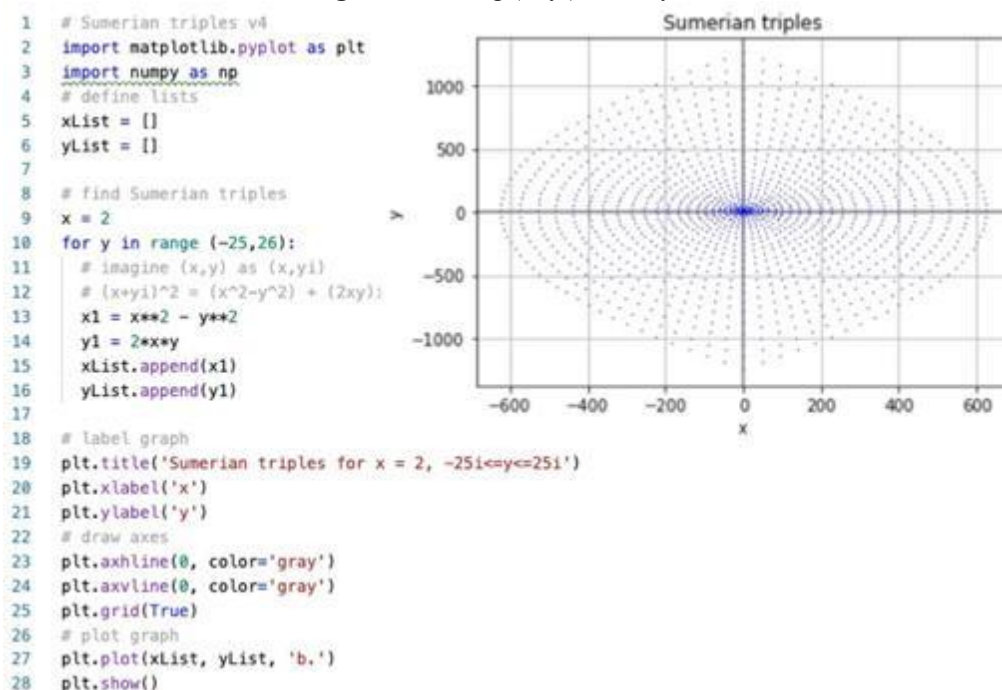
Looking at the output in Figures 2 and 3, we may also wonder about the meaning of the curves we see growing to the left and to the right of the origin. If we do some research, we may discover that such curves could be created by squaring complex number representations of integer points (a, b), that is,  $(a+bi)^2$ . (For example, see this video by Grant Sanderson: [youtu.be/QJYmyhnaek](https://youtu.be/QJYmyhnaek)). As an extension, some students may test this idea by editing their code to plot complex numbers  $(a+bi)^2$ , where a and b are integers, as shown in Figure 5 using Scratch, and in Figure 6 using Python.

Figure 5: Plotting  $(x+y)^2$  with Scratch.



Source: Research Data

Figure 6: Plotting  $(x+y)^2$  with Python



Source: Research Data

### 3.4 Reorganization

A computational literacy causes a reorganization of the conceptual framework of a field of study, changing what we can do and who can do it. (diSessa, 2018) The remediation and reformulation we discussed above, in the context of using Scratch or Python to compute and plot Sumerian triples, reorganizes the intellectual landscape of what mathematics may be done in grades 8-9 classrooms and who can do it.

It is interesting that the algebra expectations in the 2020 Ontario mathematics curriculum for grades 1-8 changed in ways that appear to complement the new computational

focus. It appears that the curriculum writing team may have had the computer programming expectations listed for each grade in mind as they decided which mathematics concepts would best fit in each grade, or vice versa. For example:

- Computer programming structures for repeating events were placed in the same grades as algebra expectations dealing with patterns that repeat.
- Mathematics expectations for understanding and using variables start in grade 2, which is much earlier than the old curriculum. Curriculum writers may have had in mind the need of variables in developing computational representation of mathematical situations.
- The topic of solving inequalities, previously a high school topic of study, was introduced starting in grade 4 and further developed in grades 5-8. Inequalities are a good fit with the computer programming concept of conditional statements, which is introduced in grade 5.

The above examples of curriculum changes give younger students earlier access to what were previously considered to be more advanced mathematics concepts for students in higher grades.

In addition, at the classroom level, a computer programming environment gives students access to all its features at once, offering opportunities for them to independently investigate computer programming representations of related mathematics concepts of interest.

### 3.5 Revitalization

When a field of study makes pervasive use of a computational literacy, it leads to a revitalization of the field, and how and what teachers teach and students learn (diSessa, 2018).

It is a typical occurrence in our mathematics research classrooms that teachers express surprise that (1) computer programming appears to be especially beneficial for students that for various reasons underachieve, (2) students generally take to computer programming easily, and (3) students learn incidentally rather than through explicit teaching (Gadanidis, Clements & Yiu, 2018).

The learning atmosphere also appears to be revitalized, as illustrated by the student and teacher comments below, from grade 10 classrooms where the teaching and learning of mathematics and computer programming were integrated (Gadanidis & Cummings, 2018, p. 2):

“You think: ‘I’m never going to use this.’ Then you go into coding and you actually use it. It’s more like math in action.”

“It feels like there’s more space. You don’t have to do it like everyone else.”

“In regular math class, the teacher teaches you everything. But with coding you’re more independent. It’s more of a sense of accomplishment.”

“I didn’t take this course expecting it to be more collaborative. It just happened. Naturally. I like it.”

## 4 Case study

We conducted a case study to determine teacher reactions to the Sumerian Triples activity discussed above. The study included 9 undergraduate students in Brazil, enrolled in a concurrent mathematics and mathematics education program. Two of the preservice teachers

were in year 3 of the program and seven were in year 4 of their program.

Cresswell (2002) and Yin (2014) define a case study is an in-depth exploration of a bounded system. Our case is narrowly focused on (1) a mathematics task (Sumerian Triples), (2) the role of computer programming in remediating the task, and (3) the perception of third- and fourth-year undergraduate mathematics students with a focus on mathematics education. These students are in the latter stage of their program and have developed more than sufficient mathematical knowledge to understand and extend the mathematics involved in the Sumerian triples task and sufficient in-class and practicum experiences to comment on the task from an education perspective. Merriam (1998) states that “A case study design is employed to gain an in-depth understanding of the situation and meaning for those involved” (p. 19). The narrow mathematical scope of the task, the similarity of the background of the participants, and our elaboration and illustration in-action of the computational literacy principles we used as an analytical lens (diSessa, 2000, 2018), allowed for a focused and in-depth exploration of the role and effect of computer programming on mathematics education.

#### 4.1 Method

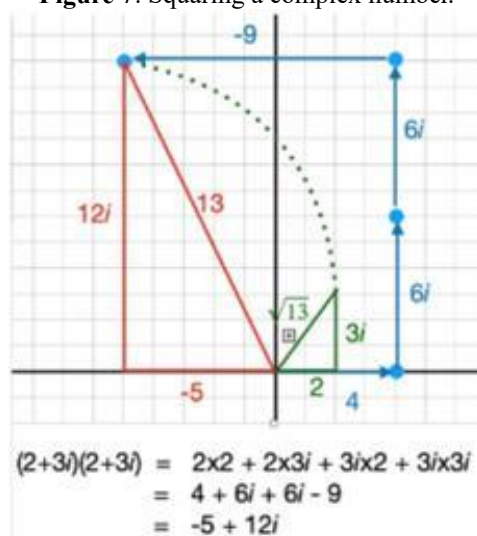
The 10 participants met for 2 hours in a recorded Google Meet session, led by the second author, with the first author observing and occasionally commenting via Chat. The activity alternated between short presentations and questions to answer, share, and discuss. Participants were given 5 or 10 minutes to record their individual responses in a Word document. Once the time was up for each set of the questions, participants were asked to copy and paste their responses in Chat. We asked participants to wait and post when we instructed so that they did not have access to one another’s responses while authoring their personal responses. For each question, a discussion followed once responses were posted. Below are the first 8 questions asked. Participants were given 5 minutes for each of questions 1-7 and 10 minutes for question 8. Question 9 (shared later in this section) was emailed to them and they had a week to reply.

1. For Question 1 (Q1), participants were shown an image of a right-angle triangle and were asked to “record what you know about the side relationships of right triangles” and to share “a story about what you know that offers mathematical insight and surprise”.
2. For Q2, they were introduced to a clay tablet of Sumerian triples (Figure 1) and were asked to “record how you might find 100 Sumerian triples”.
3. For Q3, they were shown the Python code (but not the plot of Sumerian triples) in Figure 3 and they were asked to “record how you think the code determines which pairs (a, b) to plot”.
4. For Q4, they were shown the plot of Sumerian triples in Figure 3 and were asked to record “What do you notice? What surprises you? What did you learn? What questions arise?”
5. For Q5, they were introduced to the square of a complex number  $a + bi$ , where a and b are integers, and were shown the result visually as depicted in Figure 7. Complex numbers were briefly discussed as vectors whose magnitude is the length of the hypotenuse as shown in Figure 7. In this context, they were asked to record “What do you notice about the resulting vector? How might this help us find Sumerian triples?”
6. For Q6, they were shown the Python code (but not the plot of Sumerian triples) in Figure 6 and they were asked to “record what you think the code does”.
7. For Q7, they were shown the plot of Sumerian triples in Figure 6 and were asked to

record “What do you notice? What surprises you? What do you learn? What questions arise?”

8. For Q8, they were asked to record ‘What did you learn? What surprised you? What insights did you experience? What questions arise? How might you use this in your teaching?’

**Figure 7:** Squaring a complex number.

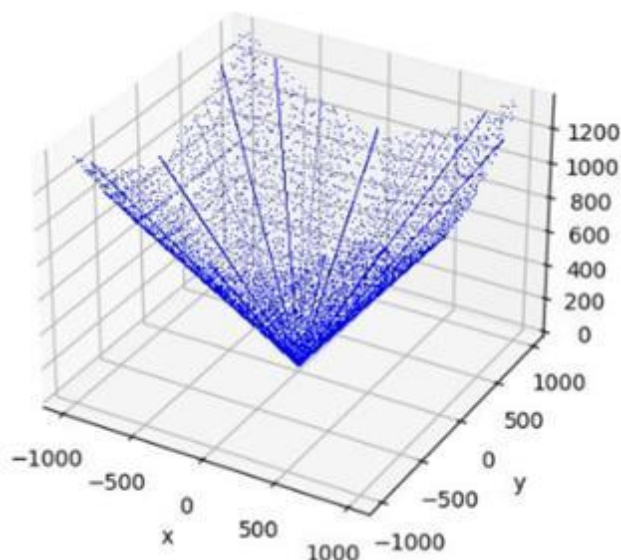


Source: Research Data

All participant responses to questions 1-8 were copied into a single document, in Portuguese, along with transcriptions of their discussion following each question. The document was also translated into English. Content analysis (Berg, 2007) was used to analyze the responses to identify themes that emerged.

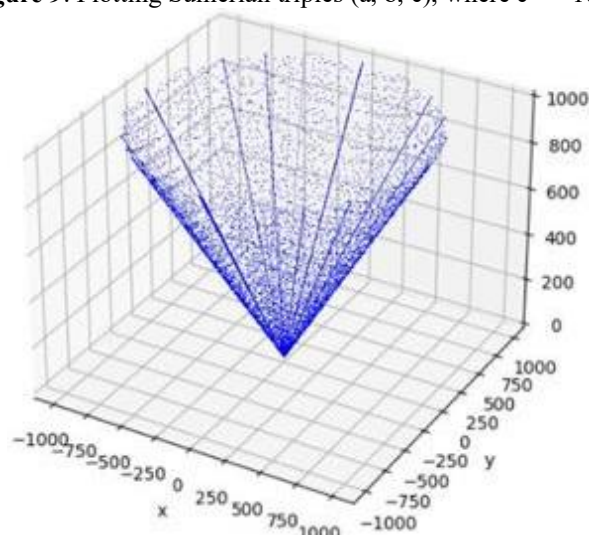
After the two-hour session, we shared with participants the following extension: Some of you wondered what the graph may look like if we also plot the hypotenuse. Plotting Sumerian triples (a, b, c) using Python would create a three-dimensional graph, as shown below (see Figures 8 and 9).

**Figure 8:** Plotting Sumerian triples (a, b, c).



Source: Research Data

**Figure 9:** Plotting Sumerian triples (a, b, c), where  $c \leq 1000$ .



Source: Research Data

Participants also received reflection question Q9:

1. diSessa (2018) notes that when we incorporate computational tools (like Python) the following changes may occur:
  - a. Mathematics concepts, problems and processes are remediated (represented in new ways) through dynamic representation.
  - b. Seeing through a new perspective and using new tools causes mathematics to be reformulated, which creates a cognitive shift, and reveals cognitive simplicities (allowing us to do some mathematics more easily).
  - c. The mathematics landscape is reorganized, changing who gets to do what and when.
  - d. Mathematics is revitalized, changing how it is taught and learned.
2. Comment on each of the above. In what ways does the Sumerian triples activity demonstrate the potential for such changes?

We used content analysis (Berg, 2007) to identify and organize participant responses thematically. This analysis was rudimentary, given that the themes were predetermined by the questions asked and we had to deal with a relatively small amount of data. Our overarching goal was to use Zwicky's (2003) seeing-as to draw links between theory and practice, and to better understand theory and practice as a whole.

Selected participant responses summarized or quoted are presented below. Section 4.2 organizes responses based on the themes of individual questions Q1-Q7. Selected participant responses to questions Q8 and Q9, which are overarching questions, are presented in Section 4.3 and are organized based on diSessa's principles of remediation, reformulation, reorganization and revitalization. Both sections offer selected participant responses to avoid repetition of ideas.

A 9-participant, two-hour activity, with a follow-up reflection, coupled with our in-depth analysis of the task and participant responses using diSessa's principles of computational literacy, is a good starting point for understanding the role and effect on computer programming

in this particular case, and the identifying themes and directions for further study.

The two-hour activity was not a hands-on programming activity, with participants writing or altering computer programs or completing the mathematics themselves. Instead, the two-hour activity took the form of a presentation or a lecture, interspersed with opportunities to reflect and to share and discuss reflections. While perhaps not ideal, Rodd (2003) notes that "something inspiring and marvelous can happen in a lecture (...) 'awe and wonder' can be experienced in a lecture theatre" (p. 15). The goal of offering awe and wonder, which we often refer to in our classroom research as conceptual surprise and insight was a key design principle of the Sumerian triples presentation, reflection and discussion.

#### 4.2 Teacher responses to Q1-Q7

Below we identify the key ideas shared and discussed by participants during the 2-hour session, organized around questions Q1-Q7.

- *Q1 "Record what you know about the side relationships of right triangles" and share "a story about what you know that offers mathematical insight and surprise"*

Participants related traditional knowledge taught in school about the Pythagorean relationship. For example, "knowing two sides, it is possible to obtain the value of the third side, through the following relationship: 'The sum of the square of the sides is equal to the square of the hypotenuse.'" One participant also mentioned that trigonometric relationships may be defined using a right triangle. None of the participants shared any other insights or surprises related to side relationships of right triangles. One participant shared that Zeno's paradoxes are examples of mathematical surprises.

- *Q2 "Record how you might find 100 Sumerian triples"*

A few of the participants suggested starting with a known triple and then multiplying all 3 numbers by a common multiple. "Starting from an initial triple, for example (3,4,5), multiply this triple by any natural number. Thus, we can find infinite triples." Another method suggested involved using a trial-and-error process, by starting with integer values for a and b, and using the Pythagorean relationship to determine whether c is also an integer.

- *Q3 "Record how you think the code determines which pairs (a, b) to plot"*

Most participants understood that the code used integer values for a and b to calculate c. Some also explicitly stated that a and b values were only added to the list if c was also an integer. For example, one participant stated that "the program will run for all combinations of x and y and will keep in the list only those that result in an integer-valued hypotenuse".

- *Q4 "What do you notice? What surprises you? What do you learn? What questions arise?"*

Participants noticed that the graph had some lines of triples that were almost filled in, and that points were more densely plotted closer to the origin. They also noticed the symmetry of data across the four quadrants. One participant wondered how the graph may be different if one of the sides and the hypotenuse were plotted, such as (a, c) or (b, c). Another participant noted, "I learned that there are many (integer) triples that satisfy the right triangle, which go far beyond what we are used to."

- *Q5 Record "What do you notice about the resulting vector? How might this help us to find Sumerian triples?"*

Some participants had not previously seen complex numbers as forming triangles or

vectors. Some participants noticed that squaring complex numbers with integer coefficients may be a method for finding Sumerian triples. For example, one participant noted, “Therefore, to find 100 triples, we repeat the procedure 100 times, we square the result  $(-5+12i)$  (getting  $-119-120i$ ) and the hypotenuse (13) also squared (169).”

- *Q6 “Record what you think the code does”*

Looking at the code, more participants noted that squaring a complex number with integer coefficients may be a method of finding Sumerian triples. For example, one participant noted, “The rule of X is  $x^2-y^2$  and the rule of Y is  $2xy$ . And I believe that from that, the Sumerian triple will be formed.”

- *Q7 Record “What do you notice? What surprises you? What do you learn? What questions arise?”*

After seeing the second graph, more participants expressed surprise and wonder. For example:

*What surprises me is: Is there really such a pattern and if so, why does it happen? Is there any relationship with the universe of complexes or is it just an isolated case in mathematics? (Participant 2)*

*The “figure” formed is very interesting and surprising. The questions that arise are what the graph would look like with other intervals (Participant 8).*

*The pattern itself is what surprises me, because with those calculations, I didn't imagine that the result would follow any pattern (Participant 1).*

*Why aren't the x and y axes fully filled (like the last graph)? (Participant 1)*

*What would happen (...) if the hypotenuse was also plotted, what would be the patterns? (Participant 1)*

#### 4.3 Remediation, reformulation, reorganization and revitalization (Q8-9)

In this section we organize teacher responses to questions Q8 and Q9 using diSessa's (2018) principles of remediation, reformulation, reorganization and revitalization.

- *Remediation*

The school concept of Sumerian triples was remediated using a computer programming environment. Participants noted that Sumerian triples are typically experienced as an algebraic expression in mathematics classrooms:  $a^2 + b^2 = c^2$ . Using a computer programming environment (as in Figure 3) offers a way to quickly find triples with a specified range of a and b values and to plot them as ordered pairs (a, b), offering visual and a dynamic representation of patterns that may exist. One of the participants commented:

*One of the great difficulties of teaching and learning mathematics is its abstract part. Representing it through computational tools, students can then collaborate to demystify the discipline and bring new ways of learning, since each person learns in a different way (Participant 1).*

Beyond computational literacy, we may also identify *historical literacy* as an actor, which allowed Pythagorean triples (~500 BC) to be remediated across time as Sumerian triples (~1800 BC).

*Sumerian was also new to me and the relationship even with programming was surprising, I found it a very interesting method to join all these concepts (Participant 4).*

*I was surprised by the (...) history of mathematics involved in the Sumerian triples, as I only had*



knowledge of them as Pythagorean triples (Participant 1).

#### ▪ *Reformulation*

The image created in Figure 3 requires a mathematical investigation of over 4 million ordered pairs (a, b) where a and b are integers ranging from -1000 to 1000 to identify and plot only the ordered pairs where c is also an integer, and  $a^2 + b^2 = c^2$ . A computer programming environment offers the cognitive simplicity of completing this task in a couple of seconds. Just as importantly, the computer program can repeat the task in a similar fashion for other ranges of values for a and b. Additionally, we may edit the computer program to plot all 3 values (a, b, c) of possible Sumerian triples in 3 dimensions, as shown in Figures 8 and 9.

Participants noted that the two-hour activity offered novel and more complex ways of conceptualizing Sumerian triples, which created mathematical surprises and cognitive shifts.

*I learned that complex numbers can be used in the calculation of the relationships of a right triangle and it surprised me a lot because in regular education complex numbers are not usually related to the contexts of natural numbers (Participant 4).*

*The activity surprised me by the didactic used to talk about the themes and how can go deeper than we think and learn in our academic life (Participant 7).*

*I found a different way of approaching Pythagoras' theorem as we generally only work with natural numbers in schools and furthermore the questions are mostly [narrow] exercises and not open-ended questions with multiple possible answers (Participant 1).*

*I learned to relate Pythagorean triples and complex numbers and explore the result of this relationship, much more contextualized through graphic visualization (Participant 6).*

It was an “opportunity to learn and deepen the knowledge that I had acquired in high school.”

*In the case of the activity in question, we were able to reformulate what we knew about the right triangle and, in addition, learn new things (Participant 1).*

*The most surprising part is how content that is used in the classroom can be used as an object of study for more complex situations (Participant 8).*

*In general, everything surprised me with new ways of approaching and reflecting on themes that are known (Participant 7).*

*I am finishing my undergraduate degree and I learned something new. Something I never heard along the courses (Participant 2).*

Participants also found these more complex conceptualizations understandable.

*Mathematics for me sometimes, because it becomes too abstract, causes some confusion or difficulty in understanding, which did not happen in this case (Participant 2).*

*Nothing confused me, it was all learning (Participant 7).*

#### ▪ *Reorganization*

We purposely chose to represent the plots of Sumerian triples not only using Python (Figure 3) but also Scratch (Figure 2). Scratch is typically used in grade 1-9 in all schools in Ontario, while only some schools also use Python, typically in grades 7-9.

From a computer programming perspective, students need to be able to create nested repeating structures and use conditional statements, which are in grades 4 and 5, respectively, of the Ontario mathematics curriculum. Mathematically (and computationally) they also need to understand and use symbols as variables in expressions and equations, which starts in grade 4 in Ontario; plotting ordered pairs in all 4 quadrants, which starts in grade 6; and the Pythagorean relationship, which is in grades 8 and 9. Therefore, by grade 8, all students in Ontario are expected to develop the prerequisite computational and mathematical literacy to understand, edit and create the Sumerian triples computer programming representations shown in Figure 3 (plotting ordered triples) and Figure 5 (identifying similar ordered triples), and they are capable of experiencing many of the mathematical extensions, surprises and insights identified by our participants:

*What surprised me the most was the transformation with the graphics, I found it very incredible. This content, despite dealing with roughly right triangles and about Pythagoras, is related to more advanced content (Participant 8).*

*The Pythagorean Theorem can be explored beyond an algebraic expression (usually worked in the classroom) (Participant 6).*

*I learned that there are many triples that satisfy the right triangle, which go far beyond what we are used to (Participant 1).*

*I found a different way of approaching Pythagoras' theorem as we generally only work with (natural numbers) in schools and furthermore the questions are mostly (well defined) exercises and not open-ended questions with multiple possible answers (Participant 1).*

*It was very interesting to see this study that deals with subjects that we see in our daily lives, and which until then were simple, being treated in more profound ways and related to the graphic part (Participant 8).*

*Sumerian was also new to me and the relationship even with programming was surprising, I found it a very interesting method to join all these concepts (Participant 4).*

*The use of these tools optimizes human thinking and allows our conjectures to go further, for example, manually it would be unfeasible to plot the graph of the Sumerian triples, but with software and programming, we can form iterations that perform this process in a more efficient way. So we can focus on other things, like thinking about changing an axis, observing, analyzing and investigating the results (Participant 1).*

#### ▪ Revitalization

Participants in our study identified several ways in which they felt mathematics and mathematics teaching was revitalized.

*I would use it to teach, after having worked well in each of the necessary subjects. This subject causes mathematical surprise and makes students think beyond, think critically and exercise their creativity (Participant 1).*

*I feel charged with the mission of bringing these issues closer to the students' reality in some way (Participant 8).*

*This relationship with these other contents generates many different questions and can bring out the passion and taste for mathematics in students (Participant 8).*

*(...) teaching of mathematics in a differential way and with the aim of making the student interested in the themes and that there are several ways to learn that this is not a closed teaching (Participant 7).*

*I felt in a new world and it sparked my interest in researching and learning more about what was talked about (Participant 8).*

*I learned that complex numbers can be used in the calculation of the relationships of a right triangle and it surprised me a lot because in regular education complex numbers are not usually related to the contents of natural numbers (Participant 4).*

*It was a very productive experience and even made me research more on the subject in parallel with the activity (Participant 2).*

*I had never seen the teaching of the right triangle and the triples that form them in this way, so this activity changed the way of teaching and learning this subject for me, making it more interactive and less dualistic (right and wrong) as the questions were more open (Participant 1).*

*I wondered if there are applications in other areas using right triangles with complex numbers or using the Sumerian triples (Participant 1).*

*I felt in a new world and it sparked my interest in researching and learning more about what was talked about (Participant 8).*

Revitalization beyond the class activity likely happened for students who later, with the newly learned perspective on mathematics, wondered on the Sumerian triples in the context of their visual categorization as shown on the plots and why this was the case: the nature of Sumerian triples forming the rays/lines (non-primitive triples), those for the curvilinear patterns (primitive triples) of specific (e.g., scalene) shapes, and the mirroring in the different quadrants, and the relation between the overall shape stamped on the 2D axis and the overall volume projected in the 3D axis forming a cone.

## 5 Discussion

Participants' responses to the Sumerian triples activity shows evidence of the potential of this integration to align with diSessa's (2018) computational literacy principles of remediation, reformulation, reorganization and revitalization.

### 5.1 The effect on teaching and learning

The Sumerian (a.k.a. Pythagorean) theorem was remediated through a graphical representation of Sumerian triples; the programming environment offered the cognitive simplicity of dealing with millions of data and calculations in a matter of seconds, reformulating the topic from a focus on finding missing numbers in  $a^2 + b^2 = c^2$  to noticing and making sense of patterns when values  $a$  and  $b$  of Sumerian triples were plotted on a Cartesian grid and values  $a$ ,  $b$  and  $c$  were plotted in 3D space, which resulted in cognitive shifts. The remediation and reorganization of concepts, along with the cognitive simplicities that resulted, gives access to a more complex mathematical structure of the Sumerian relationship to Ontario students who are in grade 8, while offering them opportunities to extend their learning to representations that involve complex numbers. In addition, the access to a computational environment was coupled with access to important and complex concepts and relationships relating to the Sumerian theorem. All of this led to a revitalization of mathematics teaching and learning that was palpable in the comments of participating teachers.

### 5.2 The role of computer programming as a language

Swain et al. (2007) refer to Vygotsky when considering how the act of speaking and writing transforms our thoughts into artefacts we can think with: "When confronted with a complex task, we may find ourselves talking aloud or whispering to ourselves, or explaining it to someone else (all are examples of languaging)". Why? Because as Vygotsky (1987) argued, language is one of the most important mediating tools of the mind" (p. 5). This can in turn help

us develop a deeper or a new understanding (O’Connell, 1988, as cited in Swain et al, 2007).

Cowley and Kuhle (2020), discuss the development of symbolic language forms that do “not rely on first-order associations with physical events that people perceive through lived experience” (p. 8) in the context of Isaac Newton’s attempt “to deduce the law of gravity for terrestrial motion from the dynamics of celestial bodies” (p. 8), in his work *The Mathematical Principles of Natural Philosophy: The System of the World* (1846). Referencing phrases from Newton’s work, they note:

You cannot “perceive” how the parts ACB and ACD ‘press each other’, ‘attract each other’ or ‘fall towards each other’. Nor are they the named items which are visible as ‘things’ like planetary bodies. Rather, they are ‘entities’ that are represented symbolically by drawing on ‘names’ (viz. ACB and ACD). These identify parameters whereby a reader can construct reciprocal relations as material extensions of a symbolic notational space. They are detached from reality or, in Vygotsky’s (1978) terms, purely ‘hypothetical’ and ‘decontextualized’. (Cowley & Kuhle, 2020, p. 8)

Such symbolic and abstract representations are forms of languaging that add a powerful dimension in the same way that diSessa (2018) described the power of algebraic notation to communicate complex ideas in concise and abstract form, which we described above in the context diSessa’s analysis of Galileo’s attempts to describe the properties of uniform motion without algebra.

Figures 2-6 above depict excerpts from programming languages Scratch and Python. Programming languages are used to communicate instructions to computing machines, as well as for humans to communicate with one another about communicating with computing machines. Languages are tools we think with; however, computer languages appear to introduce qualitatively new dimensions of talking back and acting back. Computing machines communicate back to humans using multimodal forms of language, such as numbers, text, graphs, images, and sounds. With the increasing sophistication of AI, machines also have a form of agency, collecting and analyzing data from humans as well as from other machines, and using this to control or influence the behaviour of other actors.

### 5.3 The role of other resources

diSessa (2018) states that “a literacy needs a literature. One needs to transcend a representational system by itself, and get to civilizations’ expanse of deep and powerful ideas” (p. 7). The Sumerian triples task we used in our case study may be seen as a resource that attempts to access a part of the literature of mathematics. The resource offers a more complex mathematical structure of what is traditionally referred to as the Sumerian theorem by making connections among mathematical concepts and relationships that are not typically present in school mathematics.

Another resource that comes into play at the classroom level is the programmatic curriculum (Doyle, 1992) that teachers are expected to address. The new Ontario curriculum integrates computer programming expectations within the mathematics curriculum and creates the space for the computational literacy principles identified by diSessa to potentially come into play on a massive scale, affecting over 1.7 million students and teachers.

The type of resources available in classrooms — the classroom curriculum (Doyle (1992) — play an important role in the effect of a literacy on mathematics teaching and learning. However, these do not necessarily predestine that a deeper literature of mathematics will be accessed. For example, consider having access to a computer programming environment in a

traditional classroom, where the focus of resources is on knowing and using the relationship  $a^2 + b^2 = c^2$  to determine missing values. We might then have students write computer programs that ask the user to enter any two of the values  $a$ ,  $b$  and  $c$  and determine the missing value. This engages students with Sumerian triples at the level of skill or grammar and not with the deep and powerful ideas of the literature of mathematics referred to by diSessa. Considering diSessa's principles of a computational literacy, the computer programming of a Sumerian triples calculator developed using computer code may offer some remediation by representing  $a^2 + b^2 = c^2$  as a student-created Q & A, however it would lack reformulation, reorganization and revitalization related to mathematics concepts and relationships.

## 6 Concluding remarks

In our society, computer programming is a *literacy*. It is a representation form used widely by academics and professionals to make progress in a variety of fields. It is only natural we would use computational literacy in mathematics education (and in other school subjects).

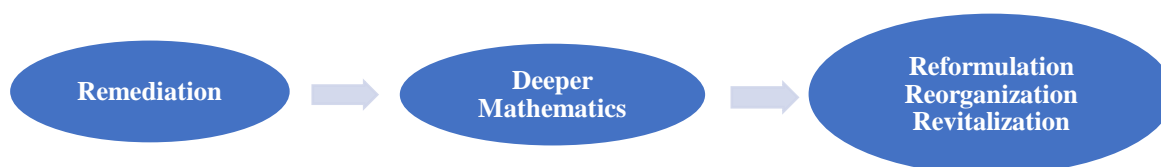
In this paper we have relied on the work of diSessa (2018) on computational literacy. We have discussed some evidence of the transformative effect of a computational literacy in the context of the Ontario mathematics curriculum. We have also considered an example of how a Sumerian triples activity may be transformed using computer programming.

Clements (2020) similarly used the lens of computational literacy to analyze the experience of approximately 1,000 first-year undergraduate students in a calculus course that integrated computer programming activities to complement the mathematics concepts studied. Clements notes:

students' conceptual understanding and affective experiences were dramatically transformed through the integration of coding and calculus. This integration revitalized their learning experiences, changed their perception of the field of mathematics, and offered unique new opportunities to dynamically interact with the theoretical ideas. (Clements, 2020, p. 98)

At the same time, we see in our research classrooms that such transformational effects are potentials rather than determinates. Computer programming can and is commonly used in classrooms to remediate shallow mathematics, where reformulation, reorganization and revitalization of mathematics does not occur. Our biggest learning in this paper is a realization of the importance of diSessa's comment that "a literacy needs a literature". As illustrated in Figure 10, remediation requires an attention to deeper mathematics for reformulation, reorganization and revitalization to take place.

**Figure 10:** A literacy needs a literature



**Source:** Elaborated by the Authors

## References

Barba, L.A. (2014). Computational thinking is computational learning. *Keynote address at SciPy (Scientific Computing with Python) Conference*, Austin, Texas. Video retrieved 5/01/17: <http://lorenabarba.com/gallery/prof-barba-gave-keynote-at-scipy-2014>.

- Barba, L. (2016). *Computational thinking: I do not think it means what you think it means*. Lorena A. Barba Group. <http://lorenabarba.com/blog/computational-thinking-i-do-not-think-it-means-what-you-think-it-means>.
- Berg, B. L. (2007). *Qualitative research methods for the social sciences*. New York, NY: Pearson Publishers.
- Bogost, I. (2005). Procedural Literacy: Problem Solving with Programming, Systems and Play. *Telemidium: The Journal of Media Literacy*, 52 (1 & 2), 32-36.
- Borba, M. C., & Villarreal, M. E. (2005) *Humans-with-media and reorganization of mathematical thinking: Information and communication technologies, modeling, experimentation and visualization*. New York, NY: Springer.
- Clements, E. (2020). *Investigating an Approach to Integrating Computational Thinking into an Undergraduate Calculus Course*. Western University Electronic Thesis Repository.
- Cowley, S.J. & Kuhle, A. (2020). The rise of languaging. *Biosystems*, 198, 1-12.
- Creswell, J. W. (2002). *Educational Research: Planning, Conducting, and Evaluating Quantitative and Qualitative Research*. London: Pearson Education.
- Diamond, J. 1999. *Guns, Germs, and Steel*. New York: W.W. Norton and Co.
- diSessa, A. A. (2000). *Changing minds: Computers, learning, and literacy*. Cambridge, MA: MIT Press.
- diSessa, A. A. (2018). Computational Literacy and “The Big Picture” Concerning Computers in Mathematics Education. *Mathematical Thinking and Learning*, 20(1), 3-31.
- Doyle, W. (1992). Curriculum and pedagogy. In P. W. Jackson (Ed.), *Handbook of research on curriculum* (pp. 486-516). Macmillan.
- Gadanidis, G. (2021). *Math + coding teams*. Whitby, ON: LearnX Publications.
- Gadanidis, G, Clements, E. & Yiu, C. (2018). Group theory, computational thinking and young mathematicians. *Mathematical Thinking and Learning* 20(1), 32-53.
- Gadanidis, G. & Cummings, J. (2018). Integrated mathematics + computer studies Grade 10 (White Paper). Toronto, ON: *Ontario Mathematics Knowledge Network*, Fields Institute for Research in Mathematical Sciences.
- Guzdial, M. (2019). Computing Education as a Foundation for 21st Century literacy. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19)* (pp. 502-503.). New York, NY: ACM.
- Hoyles, C., & Noss, R. (1987). Children working in a structured Logo environment: From doing to understanding. *Récherches En Didactiques De Mathématiques*, 8 (12), 319–352.
- Jenkins, H. (2006). *Confronting the Challenges of Participatory Culture: Media Education for the 21st Century*, Chicago: The MacArthur Foundation.
- Kafai, Y. B., Proctor, C., & Lui, D. A. (2019). Framing Computational Thinking for Computational Literacies in K-12 Education. In *Proceedings of the Weizenbaum Conference 2019 "Challenges of Digital Inequality - Digital Education, Digital Work, Digital Life"* (pp. 1-6). Berlin.
- Levy, P. (1997). *Collective intelligence: Mankind's emerging world in cyberspace*. New York: Basic Books.

- Luszkiewicz, J. & Warfel, J. (2019). The mathematical assumptions within computational literacy. In J. Jones & Hirsu, L. (Eds.) *Rhetorical Machines: Writing, Code, and Computational Ethics* (pp 93-109), University of Alabama Press.
- Merriam, S.B. (1998) *Qualitative Research and Case Study: Applications in Education*. Jossey-Bass Publishers, San Francisco.
- Noss, R., & Hoyles, C. (1988). The computer as mediating influence in the development of pupils' understanding of variable. *European Journal of Psychology of Education*, 3 (3), 271–286.
- Noss, R., & Hoyles, C. (1992). Looking back and looking forward. In C. Hoyles, & R. Noss (Eds.), *Learning mathematics and Logo* (pp. 431–468). Cambridge, MA: The MIT Press.
- O'Connell, D. C. (1988). *Critical essays on language use and psychology*. New York: Springer-Verlag.
- Ontario Ministry of Education (2020). *The Ontario Mathematics Curriculum, Grades 1 to 8*. Toronto, ON: Queen's Printer Press.
- Ontario Ministry of Education (2021). *The Ontario Mathematics Curriculum, Grade 9: Overall and specific expectations*. Toronto, ON: Queen's Printer Press.
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books.
- Rodd, M. (2003). Witness as participation: the lecture theatre as site for mathematical awe and wonder. *For the Learning of Mathematics*, 23(1), 15-21.
- Swain, M., Lapkin, S., Knouzi, I., Suzuki, W. and Brooks, M. (2007). Linguaging: university students learn the grammatical concept of voice in French. *The Modern Language Journal* 93(1), 5-29.
- Vakil, S. (2018). *Equity in computer science education*. Harvard Educational Review, 88(1), 26-53.
- Vygotsky, L.S. (1978). In: Cole, M., John-Steiner, V., Scribner, S., Souberman, E. (Eds.), *Mind in Society: the Development of Higher Psychological Processes*. Harvard University Press, Cambridge, MA.
- Vygotsky, L. S. (1987). Thought and word In R. W. Rieber & A. S. Carton (Eds.), *The collected works of L. S. Vygotsky* (Vol. 1) (pp. 243-285). New York: Plenum.
- 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*, 366 (1881), 3717–3725.
- Yin. R.K. (2014). *Case Study Research Design and Methods* (5th ed.). Thousand Oaks, CA: Sage. 282 pages.
- Zwicky, J. (2003). *Wisdom & Metaphor*. Kentville, NS: Gaspereau Press.