Connecting Polygons to Coding

Allyson Klovekorn, *Talawanda High School*

Abstract: *This paper discusses a lesson implemented in a high school classroom that promotes the connection of geometry–specifically, polygons—to coding using freely available, web-based programming tools. Details of the lesson and the tools (i.e., Turtle graphics / Logo) are shared as well as a discussion of revisions based upon relevant literature and conversations with colleagues.*

Keywords: *geometry, coding*

1 Introduction

Too often, mathematics instruction at the secondary level focuses on note-taking and solving routine worksheet tasks. The National Council of Teachers of Mathematics (2010) recommends that students should have the chance to engage in rich activities that allow them to be active participants—promoting their conceptual understanding, fostering their ability to reason and communicate mathematically, and capturing their interests and curiosity—as they learn that knowledge can be a "source of power" (Papert, 1980, p.19). In this paper, I discuss the "Connecting Polygons to Coding" (CPC) project, a three-day mini-unit that I've used with my geometry students to engage them in this type of learning. Utilizing prior conceptual and procedural knowledge, students confront their misunderstandings, solidify use of key vocabulary, and learn how to use inquiry-oriented methods to solve non-routine geometry tasks as they engage in various parts of the CPC project.

A novel feature of CPC is the use of Logo, a computer programming language developed specifically for use by young children. If you are intimidated by the thought of exploring code with your high-schoolers, relax! Logo was developed to be user-friendly. I've written this paper to help you learn how to code in Logo while enjoying the many benefits that code provides for you and your students. As students use Logo, their attitudes regarding mathematics often change. Many will begin to see geometry as an active, creative area of study; the mood of your classroom improves as students experiment to create various geometric designs and shapes.

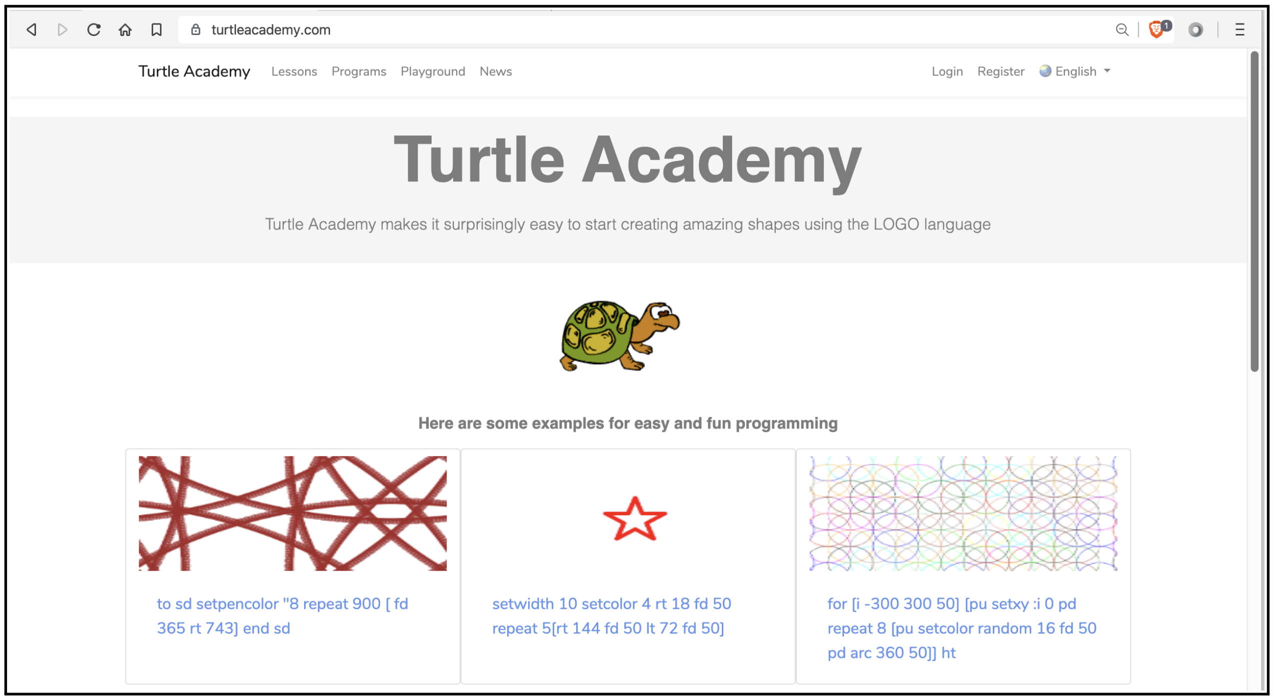
2 Statement of the Project and Context

2.1 My Students

I engaged 18 regular geometry students (10 females and 8 males) in the CPC project at the end of the school year. The students were high school sophomores and had studied polygons and their properties earlier in the year—interior and exterior angle measures, regular and irregular polygons, and classifications of triangles and quadrilaterals. The CPC project extended these topics while addressing various Standards for Mathematical Practice—including "MP.1, make sense of problems and persevere in solving them," and "MP.2, reason abstractly and quantitatively" (CCSSI, 2010). In my classroom, confusion about the properties of various shapes often arise in the quadrilateral unit (e.g., *Is this shape a rhombus? A parallelogram? Both? Neither?*). The CPC project was constructed specifically to address such misconceptions.

2.2 Turtle Academy

I first learned about Logo programming through Turtle Academy as a student in a summer Masters of Arts in Teaching (MAT) program at Miami University. Turtle Academy is a website which ”enables kids to learn programming thinking in an easy and fun way, regardless of their mother tongue or former knowledge” (Turtle Academy, 2019).



Turtle Academy

Turtle Academy makes it surprisingly easy to start creating amazing shapes using the LOGO language

Here are some examples for easy and fun programming

```
to sd setpencolor "8 repeat 900 [ fd 365 rt 743] end sd
```

```
setwidth 10 setcolor 4 rt 18 fd 50 repeat 5[rt 144 fd 50 lt 72 fd 50]
```

```
for [i -300 300 50] [pu setxy :i 0 pd repeat 8 [pu setcolor random 16 fd 50 pd arc 360 50]] ht
```

Fig. 1: *Screenshot of Turtle Academy homepage.*

My classmates and I immediately fell in love with the site and the Logo programming language in general. I could see the potential of Turtle Academy to remedy my students’ misconceptions about polygons while building upon their prior understandings in a hands-on, visual way.

2.3 Project Pacing

As stated in the introduction, the CPC project not only reinforces geometric knowledge, but also introduces students to computer programming. As I constructed my lessons, I did not assume that my students knew how to code. I expected that the project would take students at least two full days, possibly a third, which would allow them to develop a familiarity with the website’s programming language (i.e., Logo) and explore various aspects of the lesson and design project.

You can access lesson materials associated with the CPC project in my Google Drive folder at the following link: `https://tinyurl.com/turtle-resources`. Below, I discuss specifics of various parts of the project. Each part is presented to students in the following handout: `https://tinyurl.com/turtle-handout`.

In **Part 1** of the four part lesson, I ask students to brainstorm different types of polygons, both general and specific, and then share their responses as a class so that everyone has a complete list. In **Part 2** students access the student handout from their Chromebooks and follow the steps to login to Turtle Academy. After, they play around with the website and become familiar with basic Logo commands. Next, students connect properties of basic polygons to various scripts of code in a "guess which" challenge, outlined in **Part 3** of the lesson. They explore different one-line code samples and begin to recognize how various commands cause the turtle to move. Students are challenged to write their own code to create different polygons. Finally, in **Part 4**, students combine their geometric and coding knowledge to create a unique design. Students struggle from time to time with the nuances of the programming language, occasionally receiving "ERROR" messages or unexpected output. Overall, though, the ability to persevere and create a unique final project provides students with a sense of accomplishment.

Table 1: Outline and description of the lesson plan.

Part of the Lesson	Description
Part 1: Brainstorming (10 min)	Students brainstorm different types of polygons (both vague and specific) as they compile a list as a class.
Part 2: Playing (10 min)	Students go to the Turtle Academy website, following the handout instructions for logging in and accessing the "playground." This portion is printed *and* provided online, as some prefer to copy and paste commands.
Part 3: Connecting the Code (15 min)	Students work through a series of problems in which they will need to think critically in order to connect a script of code to the type of polygon it would create. Students make guesses and then justify their reasoning. They confirm their guesses by typing in code to see what polygon it actually makes.
Part 4: Designing (reminder of Day 1/Day 2)	Students create their own design with Turtle Academy. The instructions are vague, as students make many of their own decisions in the design process. They include at least one type of specific polygon that repeats in some sort of pattern, but may include additional polygons as they determine appropriate to their end goal. Their design can tell a story or simply be a picture and makes use of color.

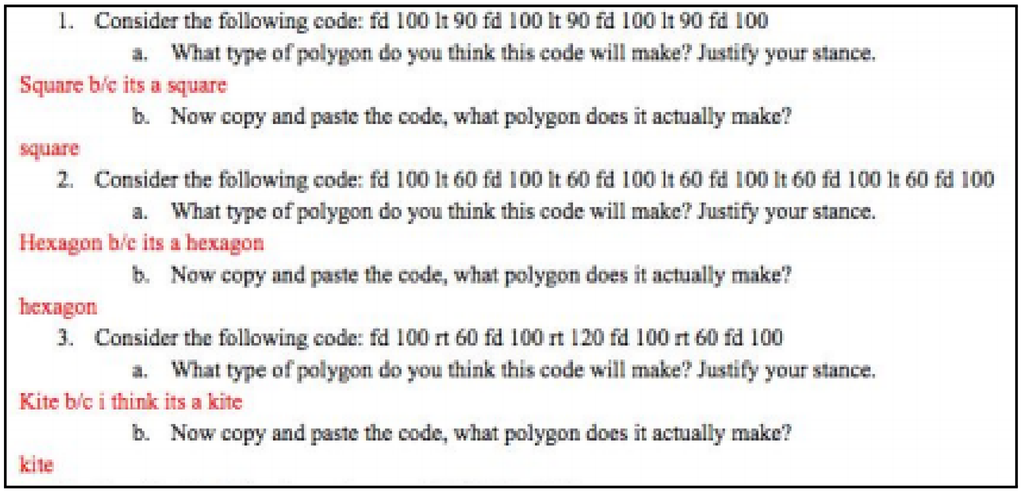
3 Analyzing Student Work

When implementing a lesson for the first time, there will always be room for improvement. My experience with the CPC project was no different. In this section of the paper, I explore student data and work samples gathered from the implementation of the project as I consider my students' learning and possible revision ideas.

Recall that in **Part 1: Brainstorming**, students worked together to recall various types of polygons and their associated properties. Next, in **Part 2: Playing**, it was time for students to actually engage with the software on the website Turtle Academy (`https://turtleacademy.com/`), where they followed the prescribed steps to login, create an account, and access the ability to create a new program (steps outlined in the student handout at `https://tinyurl.com/turtle-handout`). My students explored the commands associated with moving the turtle and tried new types of code in order to familiarize themselves with the website before delving deeper. Student data that was analyzed from these two parts was gleaned from class observations and video clips of classroom instruction. I listened carefully to student and teacher conversations to determine the impact of the lesson on student conceptual understanding and engagement.

The first aspect of the task that resulted in more formalized student work was **Part 3: Connecting the Code**. In this portion of the project, students had eight questions to investigate. Questions 1-5 involved analyzing pre-made scripts of code, making a justified "guess" for what polygon the code would create, and copying the code into the Turtle Academy website to confirm or deny their conjecture. Questions 6-8, designed to be more challenging, required students to write their own code. Again, they were able to test their prediction on the website to determine whether or not it worked—providing immediate, tangible feedback. If the code did not work the first time, students revised their code until it resulted in the desired polygon.

My hope was that students would not turn immediately to technology, but rather use paper and white boards or stand up to physically walk the commands, thus demonstrating the consideration of the mathematical properties, such as side lengths and angle measures, that differentiated each polygon. When looking at the student work submitted, I was disappointed to notice an absence of written justifications. This led me to believe that many of my students might have used technology and the classic "guess and check" method that so many of them take comfort in using. For example, one of my students simply wrote "because" or "I think," never actually giving a mathematical justification for "Part a" of questions one through five (see Figure 2). Many students submitted very similar work, leading me to consider how the set-up of the lesson or the questions themselves could be re-structured to encourage more explanation of student thought. For example, I suspect some students had valid reasons for their guesses but did not know how to articulate these thoughts in writing. It is also worth noting that the lack of justification might have stemmed from a lack of emphasis on writing throughout the year, something I plan to incorporate more in my mathematics classroom in the future.


1. Consider the following code: fd 100 lt 90 fd 100 lt 90 fd 100 lt 90 fd 100
   a. What type of polygon do you think this code will make? Justify your stance.

Square b/c its a square

   b. Now copy and paste the code, what polygon does it actually make?

square

2. Consider the following code: fd 100 lt 60 fd 100 lt 60 fd 100 lt 60 fd 100 lt 60 fd 100 lt 60 fd 100
   a. What type of polygon do you think this code will make? Justify your stance.

Hexagon b/c its a hexagon

   b. Now copy and paste the code, what polygon does it actually make?

hexagon

3. Consider the following code: fd 100 rt 60 fd 100 rt 120 fd 100 rt 60 fd 100
   a. What type of polygon do you think this code will make? Justify your stance.

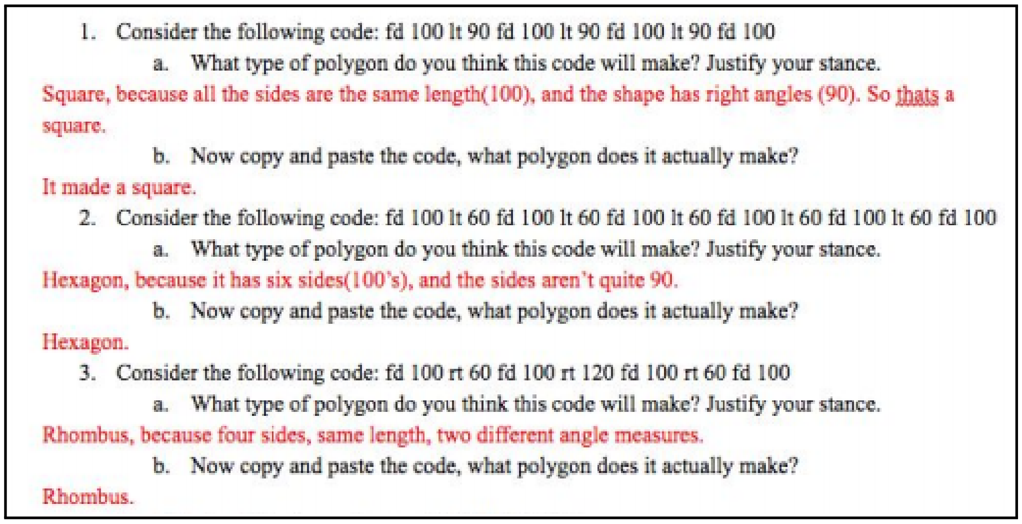
Kite b/c i think its a kite

   b. Now copy and paste the code, what polygon does it actually make?

kite

Fig. 2: *Student work lacking justifications from Part 3.*

While many students struggled with justifying their conjectures, some did meet or exceed my expectations, mentioning specific properties of polygons to support their claim. For example, when asked about the code in Question 3: `fd 100 rt 60 fd 100 rt 120 fd 100 rt 60 fd 100`, one student wrote, "Rhombus, because four sides, same length, two different angle measures" (see Figure 3). The other justifications written by this student were also well done, and it was clear that this student had carefully thought through each script of code to best determine the polygon it could represent. This is something I would like to encourage among all of my students.


1. Consider the following code: fd 100 lt 90 fd 100 lt 90 fd 100 lt 90 fd 100
   a. What type of polygon do you think this code will make? Justify your stance.

Square, because all the sides are the same length(100), and the shape has right angles (90). So thats a square.

   b. Now copy and paste the code, what polygon does it actually make?

It made a square.

2. Consider the following code: fd 100 lt 60 fd 100 lt 60 fd 100 lt 60 fd 100 lt 60 fd 100 lt 60 fd 100
   a. What type of polygon do you think this code will make? Justify your stance.

Hexagon, because it has six sides(100's), and the sides aren't quite 90.

   b. Now copy and paste the code, what polygon does it actually make?

Hexagon.

3. Consider the following code: fd 100 rt 60 fd 100 rt 120 fd 100 rt 60 fd 100
   a. What type of polygon do you think this code will make? Justify your stance.

Rhombus, because four sides, same length, two different angle measures.

   b. Now copy and paste the code, what polygon does it actually make?

Rhombus.

Fig. 3: *Student work with better justifications from Part 3.*

In general, most students who attempted questions 1-5 were successful. Unfortunately, though "correct," many students struggled to articulate their justifications in writing, as seen in the student work sample from Figure 2. Due to time constraints (namely that this project was assigned the final week of school), I encouraged students to move on to **Part 4: Designing**, even if they had not completed all eight questions from Part 3. The results quantifying how many students attempted each problem compared to those who were correct are displayed in Figure 4. I did not assess student work on this part; rather, I wanted to explose students to various strings of code and to get ideas for their own design project. I was not surprised that students typically lost motivation as the questions got harder and will need to work through a revision that addresses this complex issue, including providing more in-class work time.

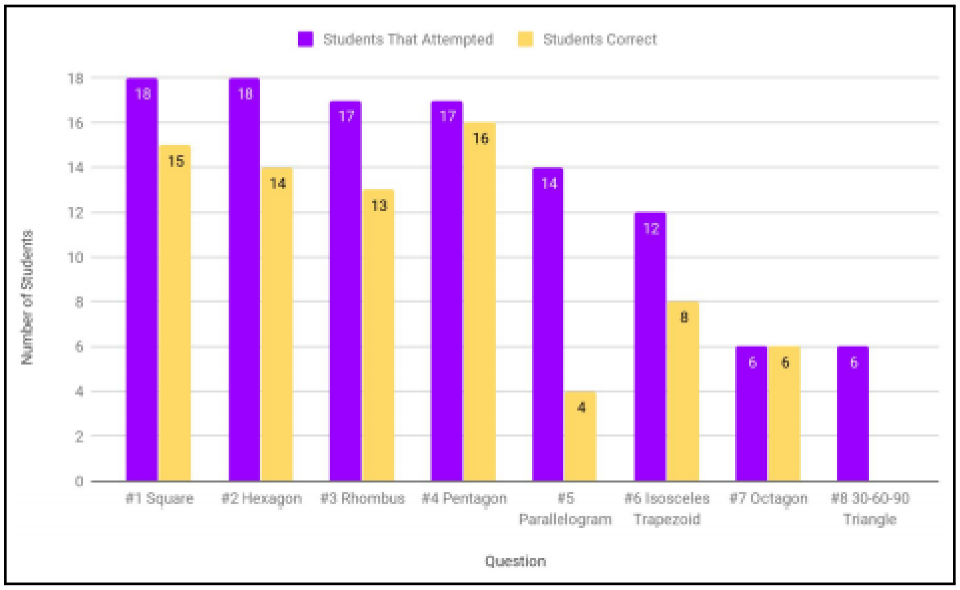
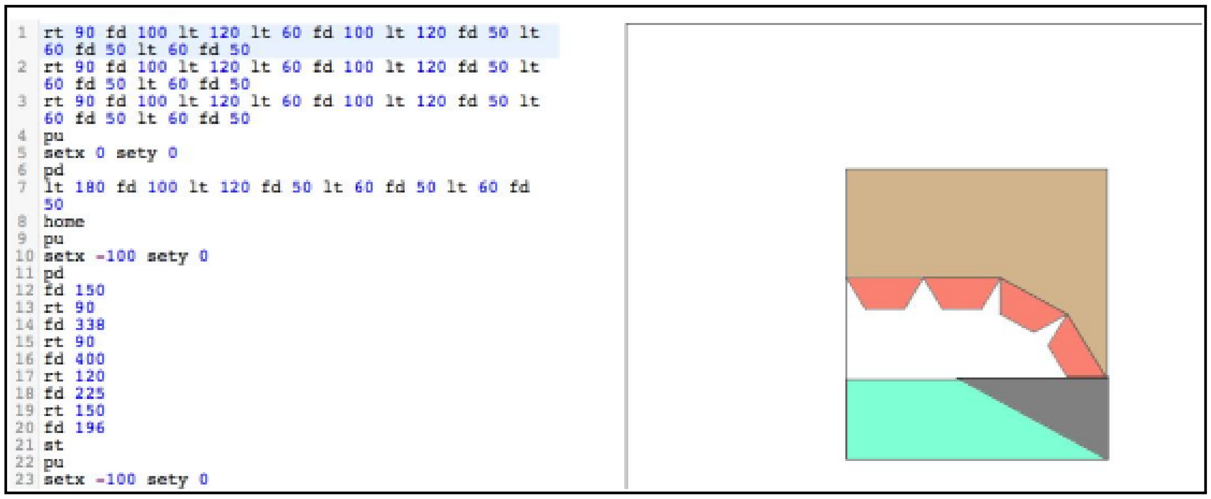


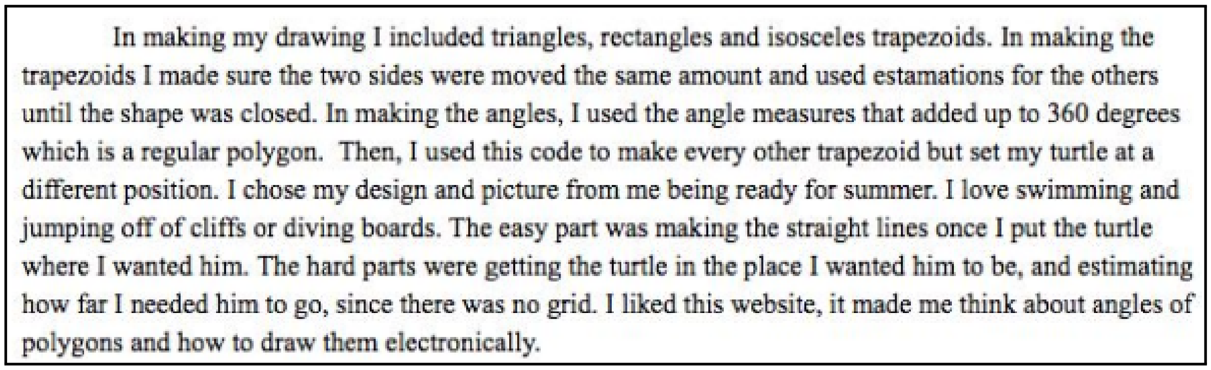
Fig. 4: *General Data results of student completion vs. correctness from Task 3.*

In the final task, which allowed for student creativity and freedom of exploration, each student coded their own design using the skills they had acquired from Parts 1-3. As students fine-tuned their designs over the two day span of the project, it was exciting to see their perseverance, growth, and productive collaboration. When students struggled to get their code to work or received the dooming message of "ERROR," they did not give up, but rather tried a new strategy to achieve the same goal. I assessed students on their final design, asking them to either submit a screenshot of their picture and code or copy the code into their Google document. I also had them engage in a short reflection, asking them to discuss the mathematics behind their design and feelings about the project overall. While many students were reluctant to write in Part 3, I was surprised by how much they shared in Part 4. One student's final code, design, and reflection paragraph are shared in Figures 5 and 6, respectively. The size of the paragraph shown by this particular student is reflective of the writing I received from the majority of students.



```
rt 90 fd 100 lt 120 lt 60 fd 100 lt 120 fd 50 lt 60 fd 50 lt 60 fd 50
rt 90 fd 100 lt 120 lt 60 fd 100 lt 120 fd 50 lt 60 fd 50 lt 60 fd 50
rt 90 fd 100 lt 120 lt 60 fd 100 lt 120 fd 50 lt 60 fd 50 lt 60 fd 50
pu
setx 0 sety 0
pd
lt 180 fd 100 lt 120 fd 50 lt 60 fd 50 lt 60 fd 50
home
pu
setx -100 sety 0
pd
fd 150
rt 90
fd 338
rt 90
fd 400
rt 120
fd 225
rt 150
fd 196
st
pu
setx -100 sety 0
```

Fig. 5: *Screenshot of student's coding work and resulting design directly from Turtle Academy website.*



In making my drawing I included triangles, rectangles and isosceles trapezoids. In making the trapezoids I made sure the two sides were moved the same amount and used estamations for the others until the shape was closed. In making the angles, I used the angle measures that added up to 360 degrees which is a regular polygon. Then, I used this code to make every other trapezoid but set my turtle at a different position. I chose my design and picture from me being ready for summer. I love swimming and jumping off of cliffs or diving boards. The easy part was making the straight lines once I put the turtle where I wanted him. The hard parts were getting the turtle in the place I wanted him to be, and estimating how far I needed him to go, since there was no grid. I liked this website, it made me think about angles of polygons and how to draw them electronically.

Fig. 6: *Screenshot of student's written reflection.*

Overall, despite having some issues with student work accurately reflecting their thought processes, I was very impressed with my students' ability to create a thoughtful design and put time into writing about the mathematics behind their code as well as their final reflections about the project itself. I truly enjoyed looking at their designs and reading their reflections, gaining insight into who they are beyond the scope of a geometry student and into the realm of their likes and dislikes as people. I would be remiss if I did not mention that some even said that they had fun (what a concept!) or that they had a greater appreciation for computer programmers. If anything, students were exposed to a very basic example of coding and designing, and maybe some of them will consider pursuing a specific field or job as a result of the CPC project.

4 Revision Analysis & Review of the Literature

Throughout the process of analyzing student data, I was able to identify many aspects of the lesson that I wanted to change. In looking at research conducted by other educators, I began formulating some of these ideas. The proposed revisions for my lesson plan are the result of not only a review of relevant literature, but also the result of various compelling and inspiring discussions with colleagues during our summer MAT experience. Some of the revisions are minor adjustments, including providing graph paper to students to encourage the use of mathematical thought on paper, making minor edits to the arrangement of questions in Part 3 so that the tasks increase in difficulty, and creating a grid with painter's tape on the floor and moving desks to make space for students to physically walk out the steps of the code. While these revisions require little additional effort on the part of the instructor, I believe these changes will enhance students' experiences with this task. See my revised student worksheet here: `https://tinyurl.com/turtle-handout-revision`.

In exploring the research about the use of technology and coding to enhance students' learning, I was directed by my professor towards the work of Papert—specifically, the text *Mindstorms: Children, Computers, and Powerful Ideas* (1980). It was while reading one of Papert's chapters that I felt supported in my desire to create an environment for my students that includes active learning, something Papert suggests can be afforded by the use of computers, which is what my lesson centered around. Because successful use of Logo requires students to understand the mathematics behind the code—angles, side lengths, rotations, sequential logic, function—Turtle Academy is a valuable tool for helping my students develop a much deeper conceptual understanding of mathematics. Papert's writing reinforced this idea for me.

Rather than letting the computers do the work for us, Papert (1980) discusses the role of programming, saying that "in teaching the computer how to think, children embark on an exploration about how they themselves think" (p.19). Papert notes that "thinking about thinking turns the child into an epistemologist, an experience not even shared by most adults" (p.19). Note that Papert does not suggest the use of technology to replace paper and pencil work. Rather, coding can be used to reinforce thought processes done on paper. This observation has implications for the CPC project. For instance, one issue I noticed after analyzing student work was their lack of mathematical justification. Students had a tendency to rely heavily on trial and error with technology. This undermined their ability (or motivation) to articulate their reasoning. One suggestion to remedy this issue, as proposed by a colleague, is to adjust Part 1 of CPC to introduce the task without initial reference to a computerized portion. This will allow time for students to practice drawing polygons on graph paper and write out in sentences the "steps" one would take to create the polygon if they had to, for example, tell someone else what to draw. Then students could practice abbreviating words like "forward" and "turn" as they start creating a list of "commands." This could lead to more challenges where the teacher provides students with a picture and the students practice listing out the codes. This practice round would help students develop more tools for their metaphorical "coding toolbox" so that they do not have as many limitations when it comes to the design project in Part 4.

Another way to scaffold students' technology use is to require students to create a rough draft design on graph paper and have it approved by the teacher first. There is no reason that students' designs cannot change, but this step encourages students to consider the placement of their shapes on the coordinate plane and how they can achieve their goals mathematically prior to implementing their ideas as code. For instance, Clements (2000) shares a figure where students designed a flag on graph paper, and then translated their design to the computer through the coding language. Similar to Papert, Clements (2000) urges his readers that "the point is not the drawing, it's the thinking about doing the drawing" (p.26). He writes that students had to "analyze the relationships between intrinsic turtle geometry and extrinsic geometry" in order to get their program to run correctly (p. 27). Before students can tell the computer what to do, they should engage in thinking about what they are actually trying to accomplish, which I hope to support more through the revisions of my lesson.

While the revisions above would fix many aspects of the task, another issue addressed through discussions with my colleagues is the lack of time to really delve into the code and website. One colleague proposed utilizing aspects of this lesson, but throughout the entire polygon unit as a way to better reinforce properties of polygons and enhance student learning while supporting struggling students. She shared ideas such as having small challenges as homework, bell ringers, or exit slips, where students write out the construction of various quadrilaterals in Logo then test their conjectures. Another idea for incorporating Turtle Academy throughout an entire unit could be accomplished through a "What-If Not" approach (Brown & Walter, 1993). For example, warm-up activities could include asking students to construct a quadrilateral with no right angles or a four-sided polygon with no congruent angles. Shifting the perspective of the problem in this way will require students to adjust their thought processes and develop different, deeper thinking skills. These shorter, interspersed activities could lead up to the final design project. Students could work on the final design project throughout the unit as they learn more about the properties of polygons and gain confidence with coding. Since the software will be explored over the span of multiple days or even weeks, students will have the opportunity to become familiar with its capabilities and thus learn more about what they are capable of creating along the way.

5 Conclusion

The lesson "Connecting Polygons to Coding" gave both my students and me a unique opportunity to learn new things about ourselves and mathematics. Research strongly supports computer programming as a tool to help students reason both abstractly and quantitatively as they develop a deeper understanding and level of thinking (Clements, 2000, p. 28). It is my hope that as I conduct this lesson again in the future, I will continue to learn from my students and make worthwhile adjustments to better implement the project. For those who are inspired to try this lesson with their own students, I hope that you find it as meaningful as my students and I found it to be, and I welcome and truly desire constructive feedback and suggestions for improvements. You will never know the impact coding might have on your classroom, or yourself, until you try.

References

Brown, S. I., & Walter, M. I. (1993). *Problem posing in mathematics education*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Clements, D.H. (2000). From exercises and tasks to problems and projects: Unique contributions of computer to innovative mathematics education. *Journal of Mathematical Behavior, 19*, 9-47.

National Council of Teachers of Mathematics. (2010). *Research brief: Why Is Teaching With Problem Solving Important to Student Learning?*. Available on-line at `https://www.nctm.org/uploadedFiles/Research_and_Advocacy/research_brief_and_clips/Research_brief_14_-_Problem_Solving.pdf`.

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

Turtle Academy. (2019). Retrieved from `https://turtleacademy.com/`

Allyson Klovekorn, `klovekam@miamioh.edu`, is a secondary mathematics teacher at Talawanda High School in Oxford, Ohio, and a graduate student in the Masters of Arts in Teaching (MAT) program in Department of Mathematics at Miami University in Oxford, Ohio. Her research interests include the use of rich mathematics tasks to promote student sense making.