math.code Presentation Notes

Contents

Pre-Coding

How To Incorporate Coding?

Integrated -> Choose coding activities to use as part of your curriculum that help with the topic currently being covered. For instance, if doing work with the coordinate system, pick an activity where the students get to practice using the coordinate system. If doing flipped or blended, you can assign video clips or tutorials on how to do specific coding tasks along with the mathematical concept being covered then in class the students can work on the activity. Mix individual and group work depending on the activity or a hybrid by having students work on different parts then combine those parts into a completed project.

Hour of Code -> While the schedule event is usually the first full week in December each year, you can have an Hour of Code event any time during the school year. Bring in a guest or find an activity at https://hourofcode.com/ for your students to do. Try to choose an activity that ties into what you have, are, or will be covering in class.

Remediation -> Some mathematical concepts are difficult for some students to comprehend. Coding is one more option for presenting the topic in a different way that may help it click with that student having trouble grasping the concept.

Enrichment -> Some students learn and work faster than others, some retain previous learned material better, and some students are just looking for more of a challenge then their classmates. Coding is one more option for that student that has the time and/or desire to learn more to explore additional topics or delve deeper into the ones that you are currently studying. Having some activities ready for those students so they can quickly and easily transition into them or work on them outside of school will be an efficient way of doing this for both you and them.

Coding Club -> For students interested in coding and math, you can offer a "coding club". You could do this yourself or enlist the help of one or more other staff members. Determine how often to meet and when. If your school has an activity period, this may be a good option otherwise, before or after school. One day a week keeps your commitment low but provides a weekly opportunity for students. You can use students to help with promoting the club, running the meetings, coming up with topics, etc. Keep the focus on math related projects and when possible tie into topics currently or that will be discussed in one or more of the math classes. Snacks might be appropriate and appreciated (and students can rotate bringing them if they volunteer to do so).

Get Some Help -> If you need assistance, seek out help from a family member, community member, business coder, college student, a college professor, or a K-12 student with coding experience. Be very specific with the type of help you are looking for and respect the person's time. Are you looking for a guest speaker, someone to help answer questions you have, or someone to help you to construct a realistic project? If you have a coding teacher in your building, talk with them about how to utilize their expertise and/or their better students. How can what you are doing also help what they are doing in their program.

Using students is an excellent opportunity for them to gain some experience and learn by teaching. They will get a chance to experience a leadership and mentorship type of role. Be sure to remember that you are the teacher and even if they might know more about coding, you are still in charge and helping them to learn how to mentor others properly.

Science Fair -> For schools participating in science fair an excellent learning experience is to take a project that researches a mathematical topic and uses coding as part of the project.

Introductory ...

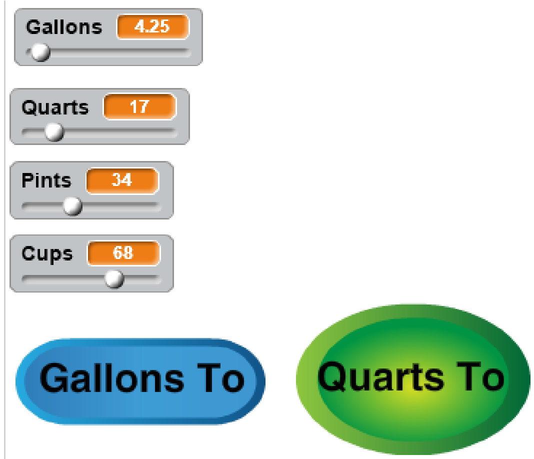
Convert Between U.S. Measurements (liquids)

This activity can also be done in any other language include Lua (CodeByMath) and Python.

Standard 5.MD.1 addressing converting between U.S. measurements. Something like this could be done with 4.MD.2b dealing with time in minutes and hours or other standards dealing with calculations and conversions.

See the Worksheet document and item titled "Convert Between U.S. Measurements (liquids) for an example of you might assign this to your students to do.

The image to the right is how this project might look (the students can probably make it look more appealing).



When going from the smaller containers to the larger ones, decimal or fractional values will not be uncommon so this is a chance to review and reinforce things not always coming out to nice whole numbers and review some of the common decimals/fractions if needed.

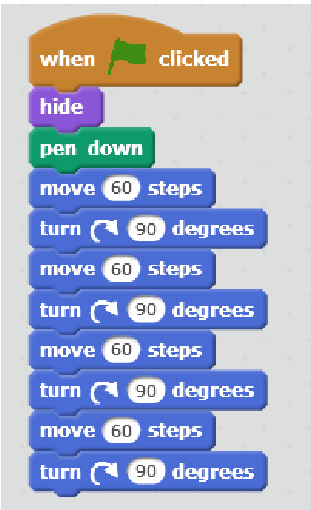
The challenge problem at the end of the student assignment requires doing division with remainders (mod or modulo). That type of calculation involves multiple steps to complete as well.

A similar type of problem to the challenge problem that is used often in computer science is making change. For instance, how can 93 cents given to a customer using the fewest number of coins.

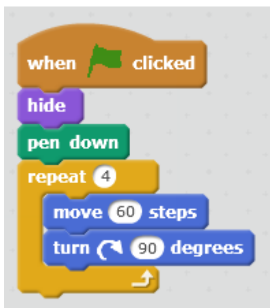
Geometric Drawing, Shapes, and Patterns (Scratch)

The following items are drawings, shapes, and patterns that can be done in Scratch (or with the Python Turtle) to explore geometric concepts. Both Scratch and Python Turtle use a LOGO style of drawing (forward, right turn, left turn, etc). Lua can also be used for drawing but uses more shapes (i.e. line, circle) to create. See the Lua Lessons for examples.

Adjust the information below based on your students' coding experience and geometric knowledge and your curriculum goals for the activity. You can do this together as a class, in smaller groups, or individually.



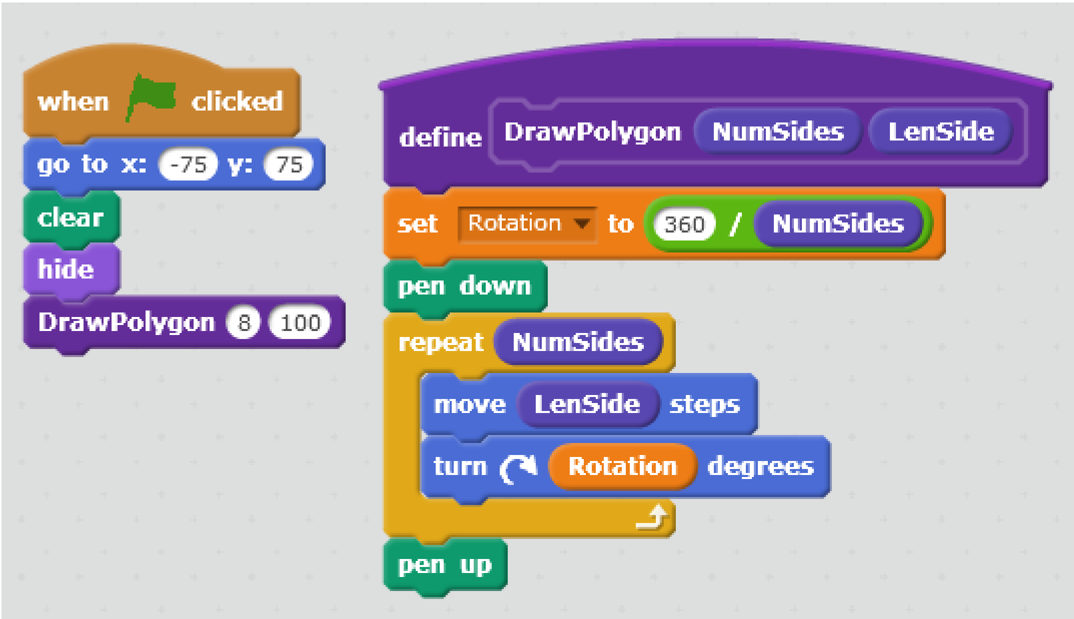
- How many degrees in a circle? In a line?
- What is a square? How do we draw a square? Angle degrees?
- Events – trigger start of program
- Hide the sprite (Looks)
- Put the pen down to draw
- Do a square with four moves (size 60) and four turns



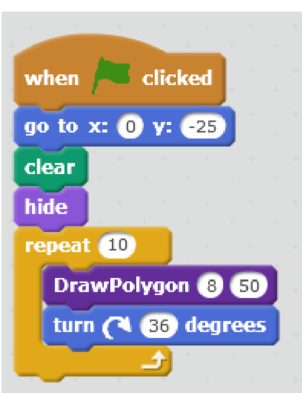
- Do you see a pattern with the code? Move/turn is repeated four times.
- Replace with the repeat loop. Show how to put in the repeat then remove the 3 pairs of unnecessary code. Change the repeat number of 4

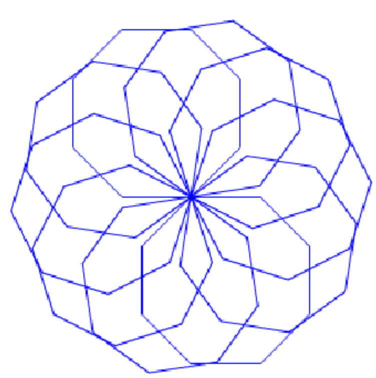
- How would we draw a triangle? How many sides? An octagon? A pentagon?
- An extremely important thing to understand and use when coding is a variable. Variables are also used in math, even more so when you get to middle school and high school.
- Variables are used to represent things that can change in your program.
- If we are drawing a triangle or a square or a pentagon, what is different about each of those? Number of sides. Angle to turn.
- Let's create a variable for the number of sides.
- Set that variable to 3 for a triangle and place that variable into the location where we repeat the number of sides.
- Run it and nothing happens. Why? If not sure, add in Clear (in Pen) or do it soon.
- Our angle is wrong. What angle should we use? There is a formula that says 180(n – 2) is the number of degrees for a polygon with n sides. For a triangle, what is n? n-2 is 1. What does 180(1) mean? Multiply 1 times 180 which is ???. So we have 180 degrees for all 3 angles, how do we find out what one angle is? Correct, 180 divided by 3 which is …
- Let's put in the 60 for the turn and run it. That did not work. Why?
- Draw out the turns and show the 180 – 60 is 120. Supplementary angles.
- Show formula (below) … write it on the board
- Put in a calculation (in Operators) for the angle 360 / Side
- Test it then change it to 4, 5, 8, 12
- Notice that we are too big to fit on the screen. What do we do?
- One option … move to a different part of the screen. The goto x/y block can be used to move to a different location.

- There are a number of ways to code this. Below is a more elaborate way illustrating creating a block to draw a polygon. You provide that block with the number of sides and length of the side. In addition to the DrawPolygon block is code to run it. Adjust your code to what you and your students are comfortable with.
- Test it out with different number of sides and lengths of those sides (and adjust the go to as needed).
- You don't want to lose your work so be sure to save it.



- How many sides in a circle? With limited number of dots on the screen, we can make a circle by picking a high enough number of sides.
- To get some neat designs, we can take shapes and move them around.
- Where is the center of the screen? Goto 0, 0
- Put a repeat 10 around the shape loop.
- How many degrees do we need to get around a circle 10 times? How do we calculate that?
- Add a turn 36 degrees at the end of the outer loop
- Run it.



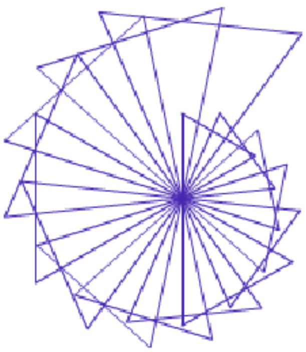
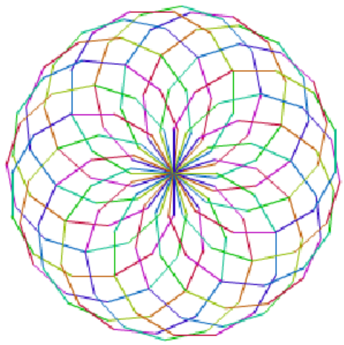
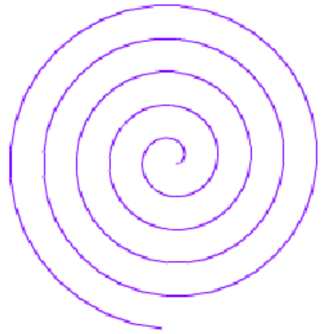
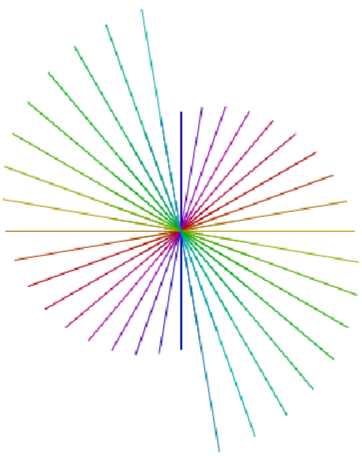
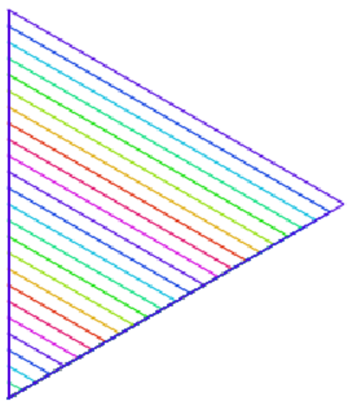
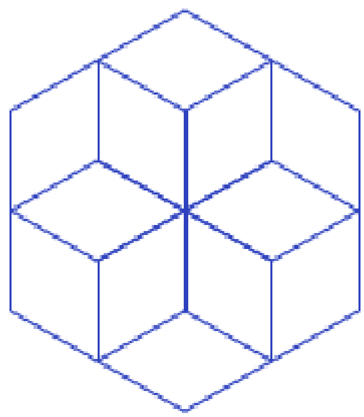


- Could we use a variable called Turn to make different shapes? How would we do that? What would the turn (degrees) calculation be using the Turn variable? What if the number is smaller than 10? Bigger than 10? Try it out.
- Let's change the color as we draw.

- Pen – change color. Can change the color when drawing a shape or change color when drawing each line in the shape. Try a smaller value (3)? To a larger value (25)? Try it and run it a few times.

Note: Use the Save As with a different name when want to keep one program and use that as a starting point for the next program.

Here are a few illustrations of things that can be done. Have the students look at the designs and determine the base shapes then the rotations to make the final design. In some cases, the shape changes size each time through the loop.

- Try!
- Have Fun!
- Learn by looking at other people's code. Then do something unique!
- Work with others!
- Ask Questions!
- Have Fun!!!

<u>Student Extension Activity</u>

Have the students search on the word tessellation or the phrase … mathematical tiling patterns. Look at some of the images associated with those search results. Choose one or more of the patterns to attempt to recreate via code. Make a copy/screenshot of the pattern to be attempted.

Animation (Scratch)

Animation is fun to do for a lot of students and uses several different mathematical concepts including the coordinate system, angles, and transformations. Scratch is built for animation so it is an excellent choice to use for an activity like this. Other environments can be used for animation as well.

This activity is simple 2D animation. More complex 3D animation uses more complex mathematics. Feel free to do some research on your own or assign some research activities to your students. Interesting topics can include Pixar's animation use of math, the Intel Drone team's events including the 2018 Olympic show, and the Purdue pre-game light show (and several other universities are doing this as well).

In Scratch, students can use the default image, choose one of the available pre-made images, or create their own. Below are some simple animations that students can try out to figure out.

Note: For each animation, have the students either create a different project or they can put multiple sprites on one project then trigger each animation differently. The show and hide blocks can be used to only view the current animation(s) running.

Start out with a quick review of the coordinate system and the layout (i.e. limits) on the Scratch screen.

Hot-air balloon (tethered) -> Place your image in the middle bottom of the screen. Ask students … where are those coordinates? What factors are involved in getting the image in that location? A tethered hot air balloon usually goes straight up (assuming no cross wind). Ask the students if the balloon goes from the bottom middle of the screen to the top middle of the screen, what changes … x, y, or both? How does that change? With that information, we can do a little bit of coding. Below is a sample of how you might want to get started. Note: Several things have been blocked out so that they can be figured out. You can choose an event like when the sprite is click, a key (i.e. space) is pressed, or the green flag is clicked. We want to move our sprite (image) to the desired location. Then we are going to move to the starting location, repeat a certain number of times, change either the x or the y by a certain value. The wait is added for effect. The default wait time is 1 second. If you do that and move your balloon more than a few times, it will be very painful to watch. What is smaller than 1 second? A half second, quarter of a second, tenth of a second? Excellent discussion on decimals/fractions and which their size relative to each other and 1. For an additional challenge, you can have the students bring the balloon back down to its starting point.

Bat flying -> Same as the hot-air balloon above except our bat is going to fly straight across the screen starting on the left side in the middle (from top to bottom) over to the right side also in the middle. How does the code change and how is it similar? Students can draw a coordinate system or you can print one out for them to draw on if needed. Within a few animations, they will have a good grasp of the coordinate system if they don't already. For additional effect, you can include two bat images (Costumes tab) and switch between those two images (using the next costume block) … each image has a different wing position to give the illusion of the bat flying.

Jet climbing -> The animation window is going to show a jet ascending. The jet will start at the lower left corner of the screen and go to the upper right corner of the screen climbing at the same distance up as over. Who is faster, a bat or a jet? Adjust the speed of each animation accordingly.

Zip-line -> In this animation, the character will start at the upper right corner of the screen and zip line to the left side of the screen about one-half of the way from the bottom of the screen.

Cartwheel -> The character in this animation will start at the lower left corner of the screen and move to the lower right corner of the screen doing cartwheels along the way. Use the turn left/right blocks to get the character to turn. Plan out how many degrees the character needs to turn for a complete cartwheel and how many cartwheels they want to do across the screen. The turns and the moves need to work together to create a realistic (or maybe a fun non-realistic) set of cartwheels.

Bunny hop -> In bunny hop animation, a bunny (or kangaroo or other animal that likes to jump), moves across the screen bouncing up and out for each jump. To provide an additional challenge, use one or two variables with the slider on to let the person running your program determine how high and far out the bunny will jump.

Whack a wombat -> This animation is based on the popular game where a creature comes up from one of several holes briefly then goes back down while the person playing the game tries to whack the creature. This can be done using the "pick random" block in the Operators group. Which you could just pick any number, a better version is to define specific columns and specific rows where the creature can pop-up. The code then needs to use a random horizontal and random vertical number to calculate where the creature should appear. After appearing for a short amount of time, the creature should hide then pop-up somewhere else. This continues for a certain number of appearances.

Bungee jump -> Watch someone jump off of a structure with a bungee cord on and see how they reach the bottom and bounce back part of the way up, go to the bottom then bounce back up a little bit lower than before, and continue the pattern until there is no bounce in the cord. Create an animation similar to a bungee jump.

Do Your Own -> Students should clearly define an animation that they would like to do then code that animation. This requires some planning, calculating, and should result in a nice finished product.

Slope or rise/run approach … the code screenshot above used the "change x/y by" block. Another way to code this is using the "go to x y" block then using variables such as x and y. Each time through the loop, you either add or subtract (or multiple/divide) a value to the x variable and the y variable. You can tie that into slope or rise/run.

Intermediate ...

Calculate Pi (Lua) – Math By CodeChallenge #1

Pi is used often in mathematics across grade levels and throughout the standards. Pi even has its own day each year. How was 3.14... determined as the value for Pi? This activity explores that.

Go to www.CodeByMath.com click on Challenges and choose Challenge #1 ...A series to find Pi

From the site ...

Write a program that will display π on the screen, from the fact that $\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7}$. Extend the sum to the first 100 terms.

What follows is how you might work through this as a class. You can take any parts of this and assign it to the students as individuals or small groups or as the whole class. It is assumed that they have worked through some of the CodeByMath Lessons dealing with calculations and loops, that can be a homework assignment or used in prior lessons or activities. Two different approaches to solving this via code are presented below ... you or your students might come up with others, different ones, or better approaches to the ones below.

Research and Mathematics Background

There are several ways to calculate Pi and a number of resources that cover this. You can assign students research or discuss some of the other ways to calculate Pi based on how you wish to focus your time in class or for homework.

Below is one such resource with several ways to calculate Pi including the method for this challenge. The little clip below confirms the same value whether using two different ways of factoring.

https://www.wikihow.com/Calculate-Pi ... See the Gregory-Leibniz series.

```
pi = 4/1 - 4/3 + 4/5 - 4/7
print(pi)

pi2 = 4 * (1 - 1/3 + 1/5 - 1/7)
print(pi2)
```

What is the problem saying? How do we do this calculation?

To solve for Pi, we will need to multiple both sides by 4. So that will look like ...

```
pi = 4 (1 – 1/3 + 1/5 – 1/7)
```

In Lua on the Code By Math site, let's do that calculation, display it, and run it to see what we get

```
pi = 4 * (1 - 1/3 + 1/5 - 1/7)
print(pi)
```

Not quite 3.14, instead it is 2.89

What are they asking us to do? What is the pattern after 1/7? ... *wait and discuss some responses* ... + 1/9 - 1/11

Let's add those next two items into the code and see what we get ...

```
pi = 4 * (1 - 1/3 + 1/5 - 1/7 + 1/9 – 1/11)
```

```
print(pi)
```

Getting closer 2.97 which is closer than 2.89.

So if we continue this pattern, will we get closer to 3.14? Discuss what the calculation is doing, how it is working, add in whatever mathematical discussions you would like to include.

Let's Write Some Code

While we could keep extending the calculation by hand, going to 100 terms like the problem states that would be pretty tedious. We need to automate this process using some code.

* What are the obstacles to solving this problem? (Ask your students, discuss, write/draw on board) … I see the two issues of: (1) Keeping a running total (2) The alternating - and + for each term
* What are some approaches we can take to solve this problem? (ask/discuss/write/draw) …
* Implement in steps or pieces ... do not need to solve first time, first try some things and see what will work and how it will work.
* Choose a way, try it, and if doesn't work or don't like it, try a different way (save prior try if helpful, can get rid of later)

Approach #1: One way to look at this is we are subtracting then adding consecutive odd numbers. So if the first time through we subtract 1/3 and add 1/5 then the next time through we will subtract 1/7 and add 1/9 then the third time 1/11 and 1/13. To do this, I need to generate these pairs of odd numbers. The for loop lets us pick our starting point, ending point, and our step for each time through the loop. Since we are doing pairs of odd numbers, we are going to step by 4 each time. The loop calculates what is inside the parenthesis then at the end we multiply that number by 4 to get our Pi value.

```
pi = 1
for i = 3, 200, 4 do
  pi = pi - 1/i + 1/(i+2)
end
pi = 4 * pi
print(pi)
```

Doing 100 terms, we get 3.15 which is getting much closer.

Approach #2: Probably the better and more concise way to solve this problem is to figure out how to deal with the change between subtract the first fraction, add the next one, then subtract, then add. So we ask ourselves, how do I change a negative number to a positive number? How do I change a positive number to a negative number? In both cases, we multiply by -1. Since subtracting first, we are going to start with a negative then each time in the loop, we multiply by -1 to switch between negative and positive numbers. So let's try this out to see if we get the desired results.

```
addorsub = -1
for i=1, 10 do
  print(addorsub)
  addorsub = addorsub * -1
end
```

Running that code prints out a -1 then a positive 1 then a -1 then positive 1. Next we need to start with 3 then 5 then 7 as our denominator. We can do that either by the for loop with the step of 2 or we can calculate the denominator. Since we did the step of 4 in our other approach above, we will do the calculation here (you can try this both ways if you like). The code below will verify our denominator pattern with the negative/positive number change. We use the print anytime in the code to see what is going on inside the code.

```
addorsub = -1
for i=1, 100 do
  denom = i * 2 + 1
  print(addorsub * denom)
  addorsub = addorsub * -1
end
```

Now we are ready to wrap this up by putting in the rest of the code to do our Pi calculation … the 4 * (1 - …) where the … are the fractional parts we are calculating.

```
addorsub = -1
calcpi = 1

for i=1, 100 do
  denom = addorsub * (i*2 + 1)
  calcpi = calcpi + (1 / denom)
  addorsub = addorsub * -1
end

pi = 4 * calcpi
print(pi)
```

When we run the code we get 3.15 (actually 3.151493401071) which, like the previous approach is pretty close to 3.14.

Extensions/Explorations: The challenge was to extend the pattern to first 100 terms. If we go to 150 terms, do we get closer to 3.1415 (or have the students find the exact value of Pi to compare with)? Go to 200 terms and see if that is even closer?

You can have the students keep trying larger and larger numbers until they get the message … "Sorry, your program is taking too long. It has been aborted." The CodeByMath site has a safeguard against a run-away program by only allowing the programs to run for a certain amount of time.

Additional Challenge 1: Write code to find out the lowest number of terms to get to 3.14. Then lowest number of terms to get to 3.141 then lowest number of terms to get to 3.1415. Make a list of the number of terms for each position of Pi (i.e. 3.14, 3.141, 3.1415, etc).

Additional Challenge 2: Try calculating Pi using one of the other methods (see link above or other resources on the Internet).

Pie is always a nice reward for calculating Pi, maybe some student or parent volunteers might bring some in to celebrate.

Function Exploration (Scratch/Lua/Python) -- The f(x)

This activity explores graphing functions via code. The https://www.desmos.com/calculator does this type of thing quite well as do graphing calculators.

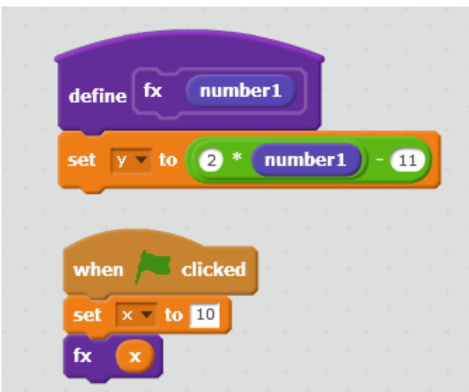
However, doing this via code establishes some basics and fundamentals that can be used and extended into solving more complex problems.

See the Student Worksheet named “f(x) – Function Exploration” for how you might assign this to your students. The intent of the worksheet is two-fold … one is to help students figure out the effect of changing different parts of the function and two is to get some practice and comfort level with writing and changing code along with trying things out.

A function in math and in computer science are different. However, using a computer science function to represent a mathematical function makes sense in this context. A computer science function is part of the program that performs a specific tasks and in some programming languages the function returns one value. In this activity, the function will take the passed in x value and use that to value to calculate the result then pass that back to the code that called the function.

In the examples below we will calculate the function of f(x) = 2x – 11 in Scratch, Lua, and Python where x = 10. Once we have that, then we can move on to looping through multiple values and graphing those values.

Scratch (2.0) has the ability to create a block which is like a function but it cannot return a value natively. In the example below y is calculated and available in the project. The block named fx is called when the green flag is clicked.



Lua’s (CodeByMath.com) function can return a value using the return command. The fx function is called and its value is placed into the yval variable.

```
function fx(x)
  result = 2 * x - 11
  return(result)
end

xval = 10
yval = fx(xval)
print(xval, yval)
```

Python’s function can return a value using the return command. The fx function is called and its value is placed into the yval variable.

```
def fx(x):
   result = 2 * x - 11
   return result

xval = 10
yval = fx(xval)
print(xval, yval)
```

Game of 45

Professor Dave Wallach ran us through the game of 45. Two players take turns picking a number between 1 and 7 then add that number to the running total. The first player to 45 wins the game. The game is easy enough to play for those that know how to add numbers whose sum is less than 50 but requires some reasoning and math to figure out what number the first player should choose to win the game each time and numbers each player chooses along the way to get the win. This also makes a nice little programming project for students.

In “expert mode”, when the computer goes first, it should win the game each time. The coded “expert mode” should also win anytime the other player does not pick the correct starting value or the correct numbers on the way to 45.

Code will not be shared here. Hint below if needed. Write it in the language of your choosing.

For an additional challenge, allow the user to choose a different total than 45 and also choose a different number than 7 for the highest number that can be used each round.

Here is a sample run (using no strategy at all) of how this game flows …

Player 1: Start with the number 3
Player 2: 6 + 3 is 9
Player 1: 7 + 9 is 16
Player 2: 5 + 16 is 21
Player 1: 1 + 21 is 22
Player 2: 4 + 22 is 26
Player 1: 2 + 26 is 28
Player 2: 6 + 28 is 34
Player 1: 5 + 34 is 39
Player 2: 6 + 39 is 45
Player 2 wins

Keep reading if you would like a hint …

Hint: Work backwards starting at 45. To win, what number do you want to get to so that the other player cannot get to 45 but you will on your turn. Once you find that number, work to the previous number to get you to that number. Continue to work down to find what the starting value should be in order to win each time. The lowest number a player can choose is 1 and the highest number is 7 … you have to choose a number each turn. Once you have those winning step numbers, what is the pattern? How can you code that. How do you code when your opponent hits one of those numbers in the winning sequence?

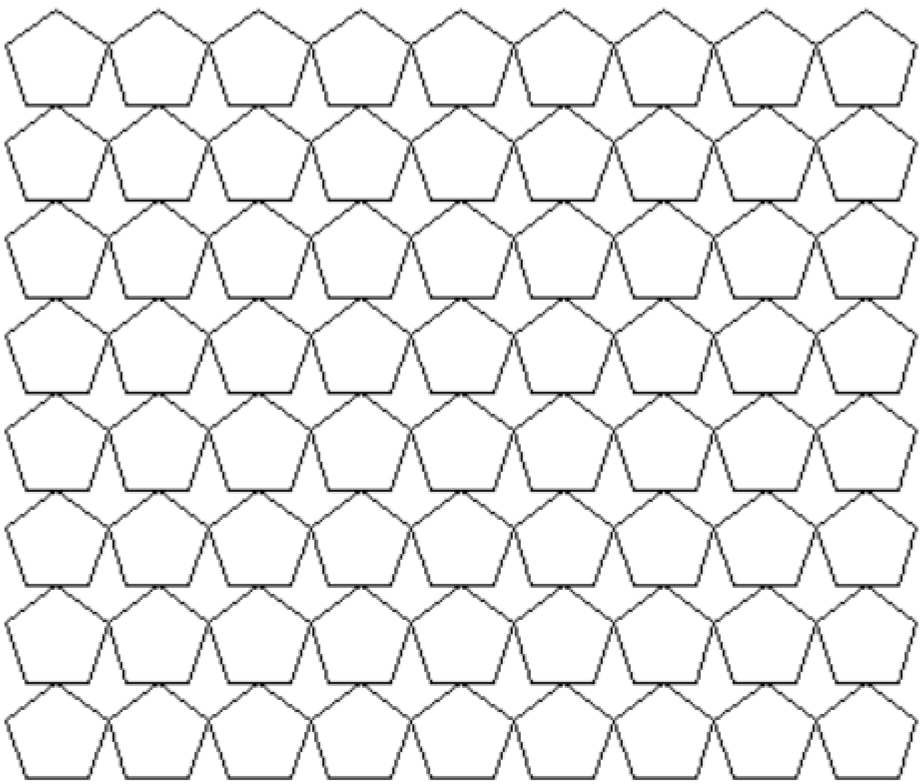
Pattern Generation

See the Student Worksheet titled Pattern Generation.

Key for the later challenges is to either calculate the size of the shapes or use the xcor/ycor to pick up the position while drawing the shape to figure out where the next one goes.

```
xpos = turtle.xcor()
ypos = turtle.ycor()
```

```
How long should the side be? 25
How many shapes across? 9
How many shapes down? 8
How many sides to your shape? 5
```



Higher End ...

Recursion (Python) - Fractals

Reference the Worksheet called "Create a Fractal (and break it up a little)". That activity walks the student through creating a fractal and hopefully starting to understand how recursion works. You can then apply it to recursive functions that you are studying. The slides have a few fractal slides for your reference as well including common problems/puzzles where recursion provides an excellent approach to solving that problem/puzzle.