math.code v3

You can incorporate coding into your math classroom

2023 OCTM Conference
October 12-13 in Sandusky, OH
Thursday 4:00 – 4:50 pm Zebrawood

<u>Session Goals for Each Participant</u>

Goal 1:	Do one more coding activity with your students than you did last year.
Goal 2:	Collect/share ideas and strategies for student coding experiences.

Materials at: https://www.github.com/teachintech90/math.code/
includes prior sessions materials

Coding activities in math classroom to explore concepts in a concrete way and problem solve.

How To Get Started If You Have Never Coded

Start with Scratch … here is one of many introductory videos to get started … https://www.youtube.com/watch?v=_pMXH9cJak4

Then try some of the other tutorials at … https://github.com/teachintech90/math.code

Alternative: Want to do text based coding instead?   Visit https://www.codebymath.com/ … and do the Let's Get Started examples on the home page then click on the Lessons to work through some lessons.

Next Steps: Work through some of the examples in this slide deck _or_ jump to the bottom of this slide deck and visit some of the listed resources.

Try, Try, Try some more … ask others or look at tutorials to start learning

My favorite question from students … When am I ever going to use this?

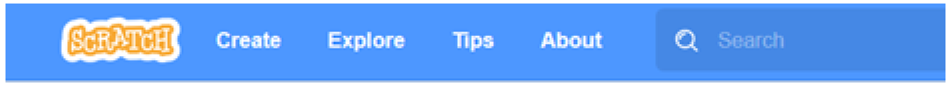
- Encourage your students to try
- Encourage your students to make mistakes while trying
- Encourage your students to learn from their mistakes
- Encourage your students to break problem down into pieces and tackle it pieces at a time (How do you eat an elephant? One bite at a time.)

Thomas Edison Quotes

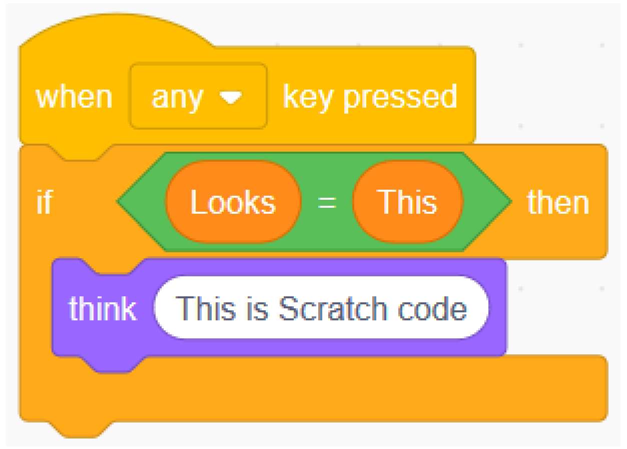
I have not failed. I've just found 10,000 ways that won't work.

Our greatest weakness lies in giving up. The most certain way to succeed is always to try just one more time.

Coding Samples



https://scratch.mit.edu



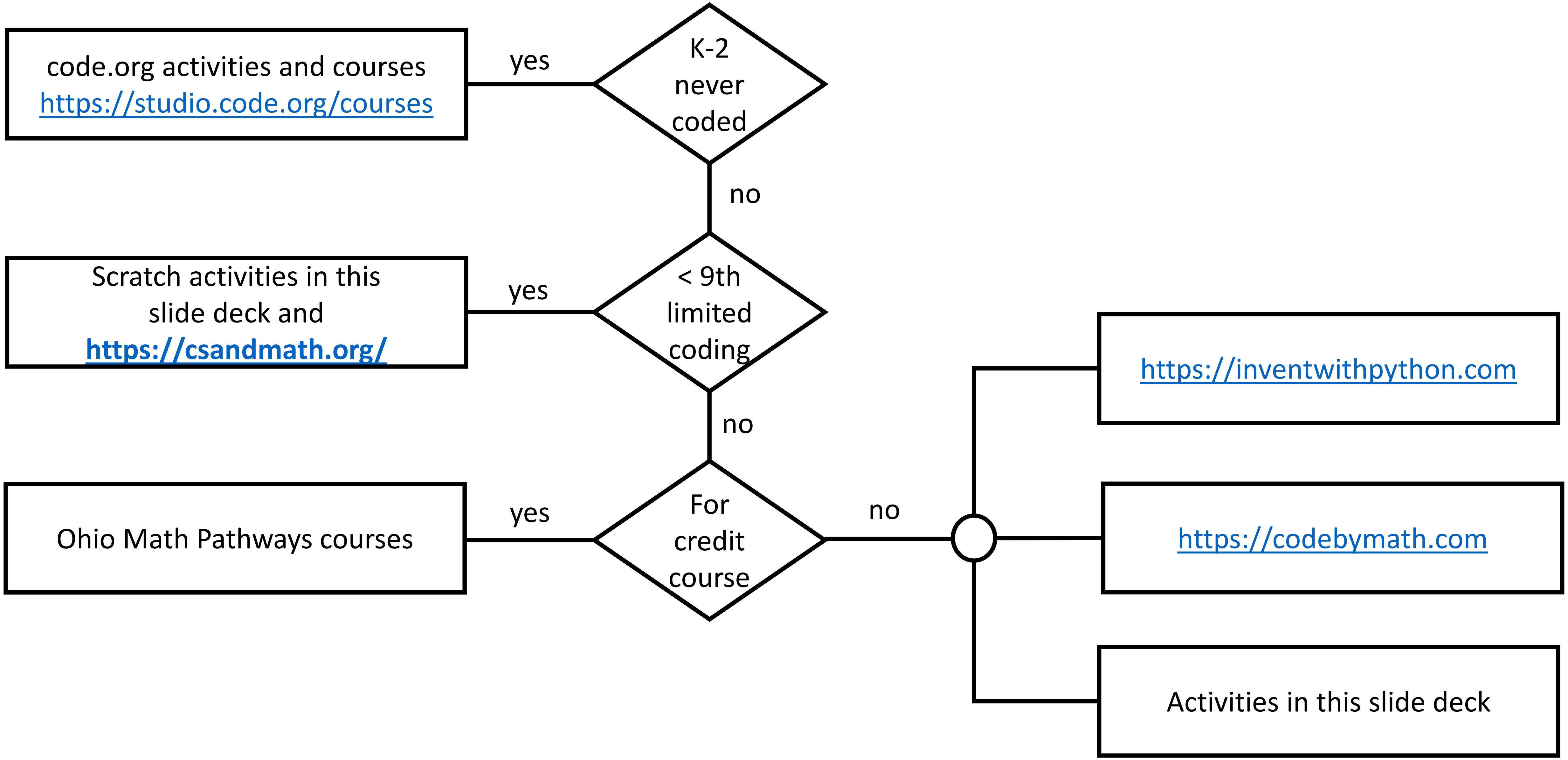
Block based examples done in Scratch 3.0. Most block based environments have similar structures.

https://www.python.org

```
if looks == likethis:
    print('Python code')
```

Text based examples done in Python 3. While the syntax of text based coding differs by coding language, the basic constructs such as variables, calculations, decision, looping, functions, etc are similar.

Some Suggested Paths (over simplified)


code.org activities and courses
https://studio.code.org/courses
yes
K-2 never coded
no
Scratch activities in this slide deck and
https://csandmath.org/
yes
< 9th limited coding
no
Ohio Math Pathways courses
yes
For credit course
no
https://inventwithpython.com
https://codebymath.com
Activities in this slide deck

Flowchart example with process, decision, and connector

… Standards / Activities …

K-5: Geometry Standards

K.G: GEOMETRY (Summary)
• Identify and describe shapes (squares, circles, triangles,
rectangles, hexagons, cubes, cones, cylinders, and spheres).
• Describe, compare, create, and compose shapes

1.G GEOMETRY (Summary)
- Reason with shapes and their attributes.

2.G GEOMETRY (Summary)
- Reason with shapes and their attributes.

3.G GEOMETRY (Summary)
- Reason with shapes and their attributes.

4.G GEOMETRY (Summary)
- Draw and identify lines and angles, and classify shapes by properties of their lines and angles.

5.G GEOMETRY (Summary)
- Graph points on the coordinate plane to solve real-world and mathematical problems.
- Classify two-dimensional figures into categories based on their properties.

K-5: Geometry Activities

Coding Activities … either code.org activities or Scratch (block based)

- Move around using move and turn blocks
- Draw using Turtle Graphics
- Draw a line, draw various letter V exploring angles
- Draw shapes starting with square and triangle
- Explore what angles and line lengths work and do not work to create shape
- Draw other polygons
- Create block to draw any regular polygon given number of sides and length
- Spin regular polygons to create unique shapes (think Spirograph)
- Make a house (or other objects) using geometric shapes
- Tessellations

Turtle Graphics - Intro

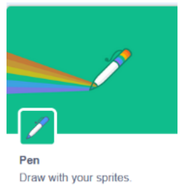
Seymour Papert and other researchers developed the Logo programming language and it controlled an actual turtle robot that would move and draw using the Logo commands such as forward, back, right, left, PenUp, and PenDown. Many of the commands had abbreviations such as fd, bk, rt, lt, pu, and pd.

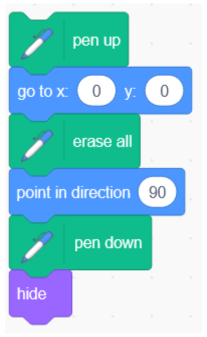
To make it more accessible, turtle graphics was the implementation of a turtle on the computer screen that would draw on the screen (rather than a robot turtle drawing on paper).

Scratch (also from MIT) has the Pen extension to implement drawing on the screen and Python has a turtle module to do this.

Turtle Graphics - Scratch

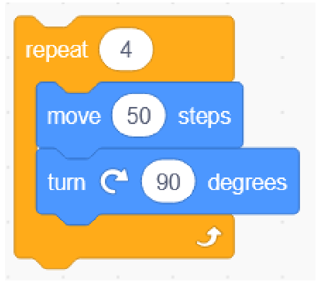
Lower left corner then select the Pen extension





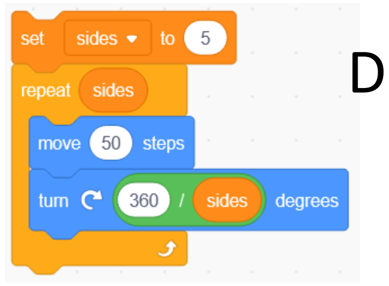
```
pen up
go to x: 0 y: 0
erase all
point in direction 90
pen down
hide
```

Move the center of the screen, facing to the right.
Hide the cat.



```
repeat 4
    move 50 steps
    turn ↻ 90 degrees
```

Draw a square



```
set sides ▾ to 5
repeat sides
    move 50 steps
    turn ↻ 360 / sides degrees
```

Draw a regular polygon

Explorations/Discussions

- Draw a line
- Draw the letter V, using right then again using left turn
- Discuss the angle in a V and different shaped Vs
- Straight line is 180 degrees, right angle is 90 degrees
- Draw a square
- What is a rectangle and how same/different than a square
- Draw a rectangle
- How many lines in a triangle.
- Draw a triangle with same size lines. Different sized lines
- Complementary angles (add to 180). Triangle has 60 degree angles but you turn 120 degrees (180 = 120 + 60). Discuss/explore.
- How many degrees in a circle? How many sides in a circle?
- Why does a 36 sided polygon look like a circle?

Do grade appropriate exploration, start with move/turn, add others as appropriate

Math Behind Regular Polygon

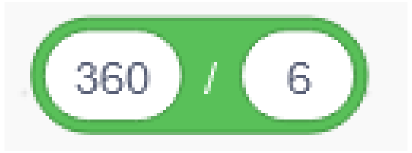
Supp Angle – inner angle

$$180 - \frac{(180(n-2))}{n}$$

$$180 - \frac{(180n - 360)}{n}$$

$$180 - 180 + \frac{360}{n}$$

$$\frac{360}{n}$$



Square – 4 sides

180(4 -2)

180(2)

360

360 / 4 = 90

Triangle – 3 sides

180(3 – 2)

180(1)

180

180 / 3 = 60

Supplementary angles … two angles that add up to a 180 (a straight line)

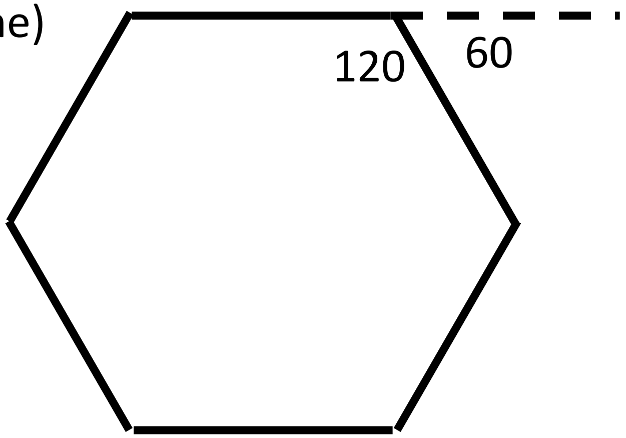
Hexagon – 6 sides

180(6 -2)

180(4)

720

720 / 6 = 120

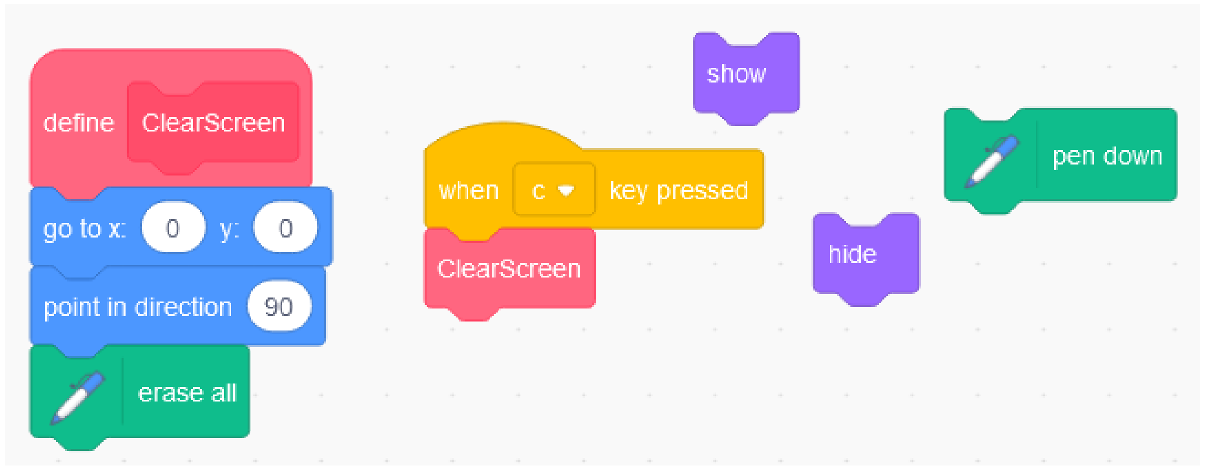


Adjust the math and the math concepts to what your students currently know or can comprehend

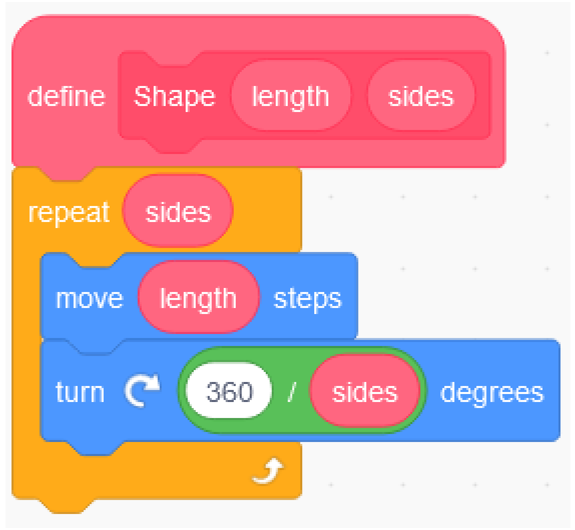
Scratch – Create a Regular Polygon

Teacher Script

- Add Extension: Pen
- Helper blocks: Erase all, pen up, pen down, goto x:0 y:0, point in direction 90, hide, show
- Main blocks: move, turn
- History tie-in: Turtle graphics/Logo with Seymour Papert at MIT
- Draw a square using move and turn
- What pattern/repetition do you see?
- Refactor using one each repeat, move, turn
- What if I wanted a bigger square?
- Draw an equilateral triangle
- Compare / contrast the square and triangle
- Math behind calculating the angle of an x sided figure
- Code drawing a regular polygon
- Test with different number of sides and validate
- What are our “unknowns” or variables?
- Create a block named regpoly or shape or ???. Discuss importance of naming.
- Run/test the block.



```
define ClearScreen
go to x: 0 y: 0
point in direction 90
erase all

when c key pressed
ClearScreen

show

hide

pen down
```



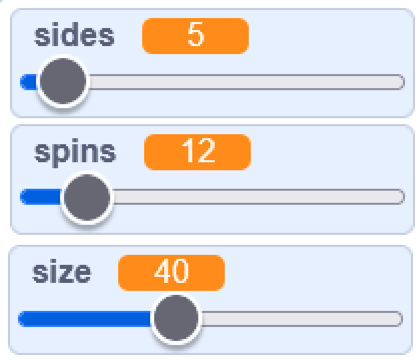
```
define Shape length sides
repeat sides
    move length steps
    turn ↻ 360 / sides degrees
```

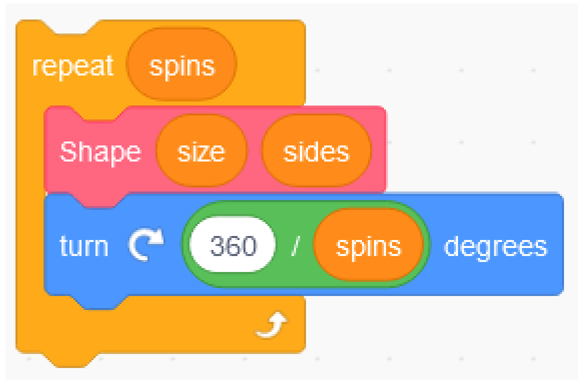
57 seconds then 3:13 – LOGO at MIT, Seymour Papert -- https://www.youtube.com/watch?v=xMzojQFyMo0

Spinning Shapes

Teacher Script

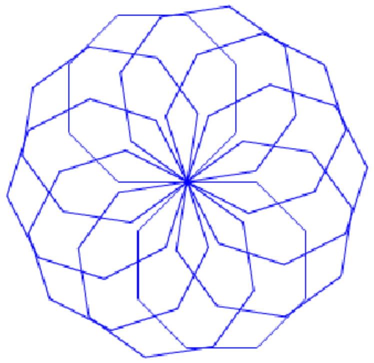
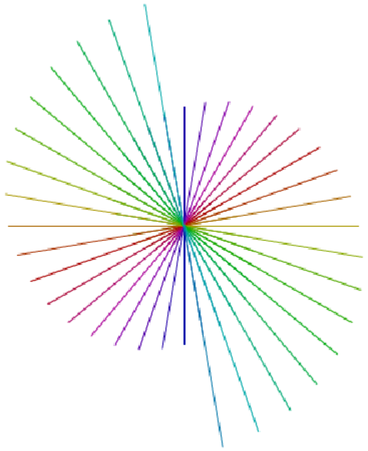
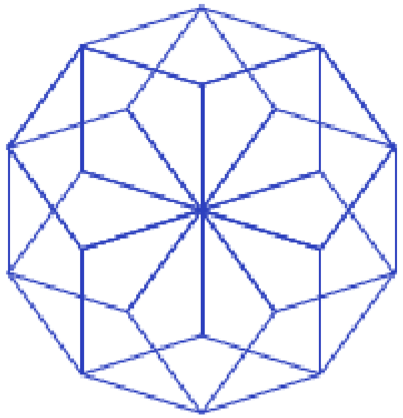
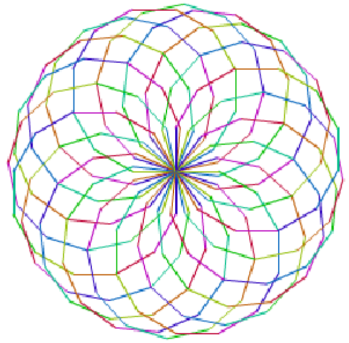
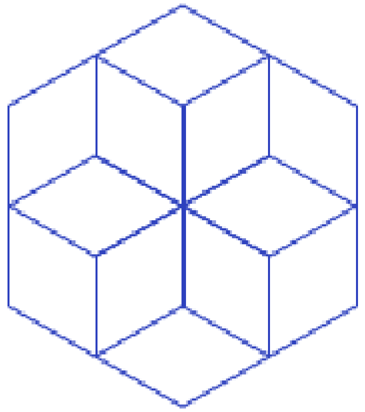
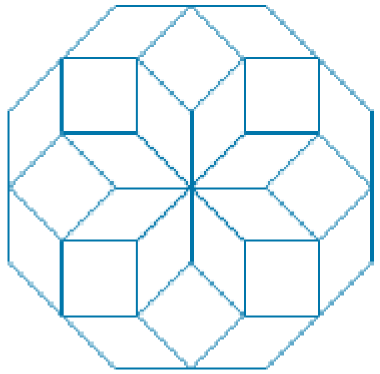
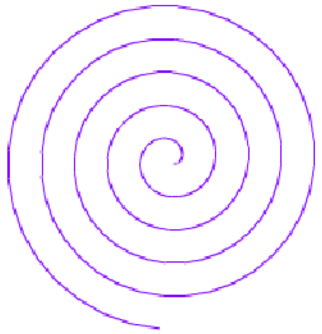
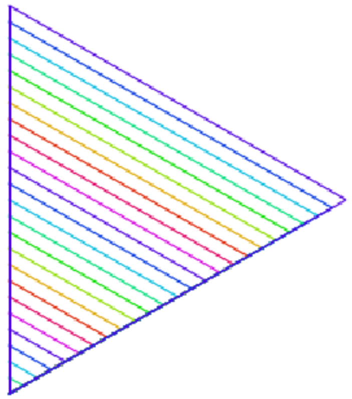
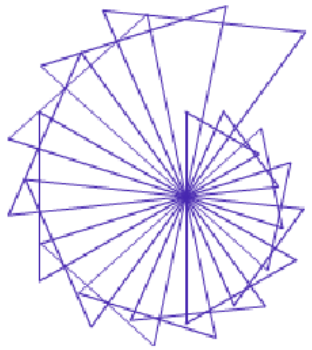
- What if we draw a shape, turn, draw a shape?
- Explore and experiment a little bit.
- What are our unknowns or variables?
- How many degrees in a circle?
- Math behind spinning x degrees, y times to complete the circle.
- Create a spin block
- Test spin block with different values
- Time to explore
- What enhancements/variations?
- Change color, different size shapes in the spin, spiral, etc.
- Have the students show you any additional creations they are excited about.


sides 5
spins 12
size 40


```
repeat (spins)
    Shape (size) (sides)
    turn ↻ ((360) / (spins)) degrees
```

When done, ask students how some of the "Can you create these?" images were done. Try those, add theirs to your own slide.

Can you create these?

Geometric patterns, most by spinning a shape. What additional ones can you create?

Turtle Graphics - Python

See link below to get started. A few different ways you will see this used.

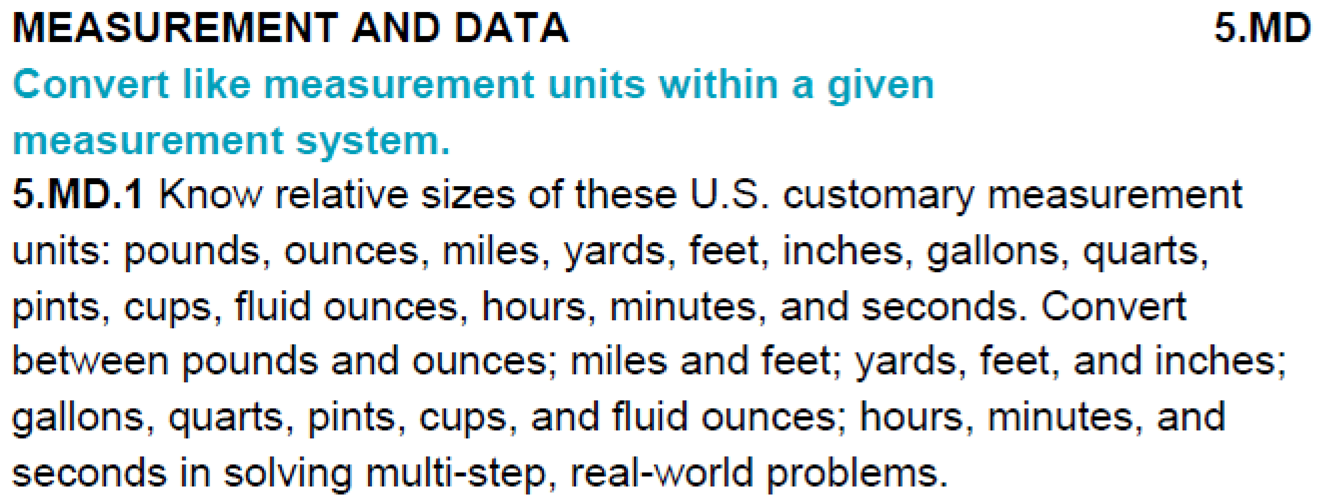
```
import turtle

x1 = 0
y1 = 0
x2 = 50
y2 = 50
turtle.color("red")
turtle.penup()
turtle.goto(x1, y1)
turtle.pendown()
turtle.goto(x2, y2)
```

```
from turtle import Turtle

x1 = 0
y1 = 0
x2 = 50
y2 = 50
t = Turtle()
t.color("red")
t.penup()
t.goto(x1, y1)
t.pendown()
t.goto(x2, y2)
```

```
from turtle import *

x1 = 0
y1 = 0
x2 = 50
y2 = 50
color("red")
penup()
goto(x1, y1)
pendown()
goto(x2, y2)
```

https://docs.python.org/3/library/turtle.html

K-5: Measurement and Data Standards



MEASUREMENT AND DATA **5.MD**

Convert like measurement units within a given measurement system.

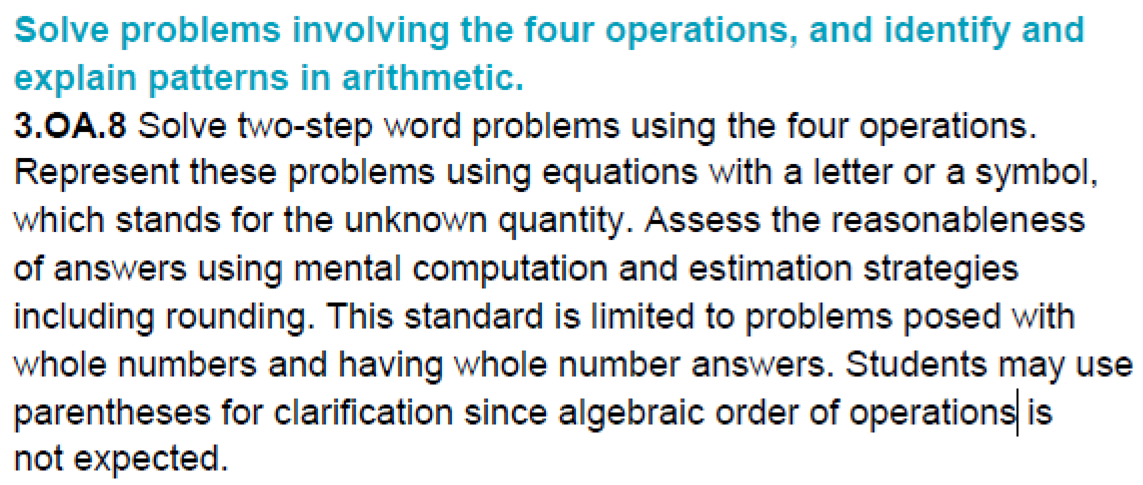
5.MD.1 Know relative sizes of these U.S. customary measurement units: pounds, ounces, miles, yards, feet, inches, gallons, quarts, pints, cups, fluid ounces, hours, minutes, and seconds. Convert between pounds and ounces; miles and feet; yards, feet, and inches; gallons, quarts, pints, cups, and fluid ounces; hours, minutes, and seconds in solving multi-step, real-world problems.

- Using the video link below, students can work through the code to convert between inches, feet, and yards.
- Discuss what is similar and different in the coding between those units.
- Assign students to groups and assign the groups different conversions to complete. Each student will write his own code but discuss as a group.
- Use rubric to evaluate work and use actual values to verify calculations

See YouTube via at https://youtu.be/stB1_cookDs and associated activity page

See page 2 of the Student Worksheet from the 2018 presentation

K-5: Operations and Algebraic Thinking Activities


Solve problems involving the four operations, and identify and explain patterns in arithmetic.

3.OA.8 Solve two-step word problems using the four operations. Represent these problems using equations with a letter or a symbol, which stands for the unknown quantity. Assess the reasonableness of answers using mental computation and estimation strategies including rounding. This standard is limited to problems posed with whole numbers and having whole number answers. Students may use parentheses for clarification since algebraic order of operations is not expected.

TIP: Use paper or electronic "cards" with the assigned problems to make this easier on you and the students. Your version of the "card" can have the values to test their code.

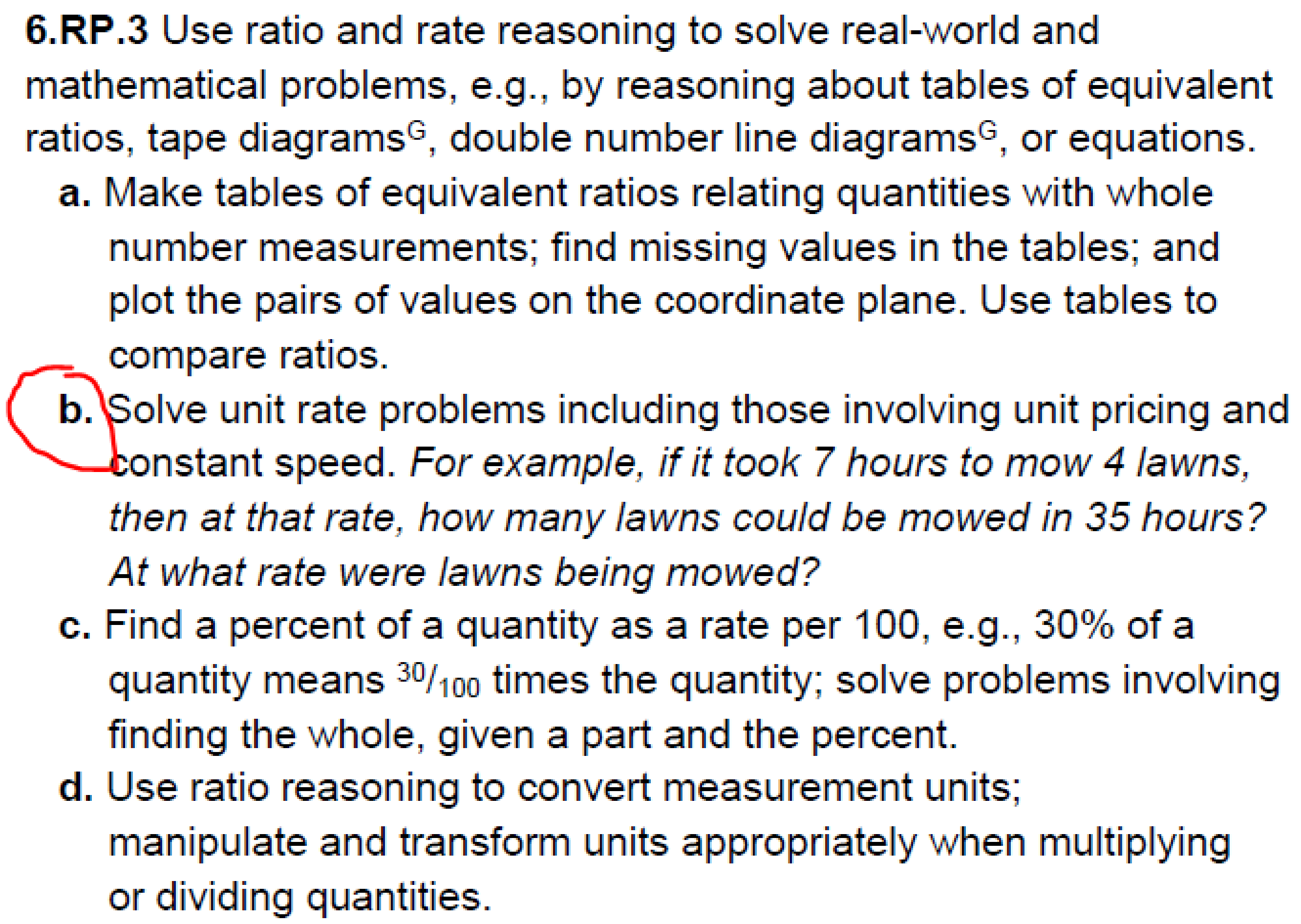
TIP2: Grade/check their work as you walk around the class. Ask them to run their code and give me the value to use. If not correct, provide immediate feedback. If correct, indicate so on your rubic/grade sheet.

- Using the video links below, students can work through the code to do calculations
- Discuss how to determine what is known and unknown then work toward a solution. Discuss the reasonableness and how to test the program.
- Assign students to groups and assign the groups different calculations to complete. Each student will write his own code but discuss as a group.
- Use rubric to evaluate work and use actual values to verify calculations

See YouTube at https://youtu.be/nVhhQRKqVsI and associated activity page

See YouTube at https://youtu.be/osZqYKOklIQ and associated activity page

Unit Rate



6.RP.3 Use ratio and rate reasoning to solve real-world and mathematical problems, e.g., by reasoning about tables of equivalent ratios, tape diagrams^{G}, double number line diagrams^{G}, or equations.

a. Make tables of equivalent ratios relating quantities with whole number measurements; find missing values in the tables; and plot the pairs of values on the coordinate plane. Use tables to compare ratios.

b. Solve unit rate problems including those involving unit pricing and constant speed. *For example, if it took 7 hours to mow 4 lawns, then at that rate, how many lawns could be mowed in 35 hours? At what rate were lawns being mowed?*

c. Find a percent of a quantity as a rate per 100, e.g., 30% of a quantity means $^{30}/_{100}$ times the quantity; solve problems involving finding the whole, given a part and the percent.

d. Use ratio reasoning to convert measurement units; manipulate and transform units appropriately when multiplying or dividing quantities.

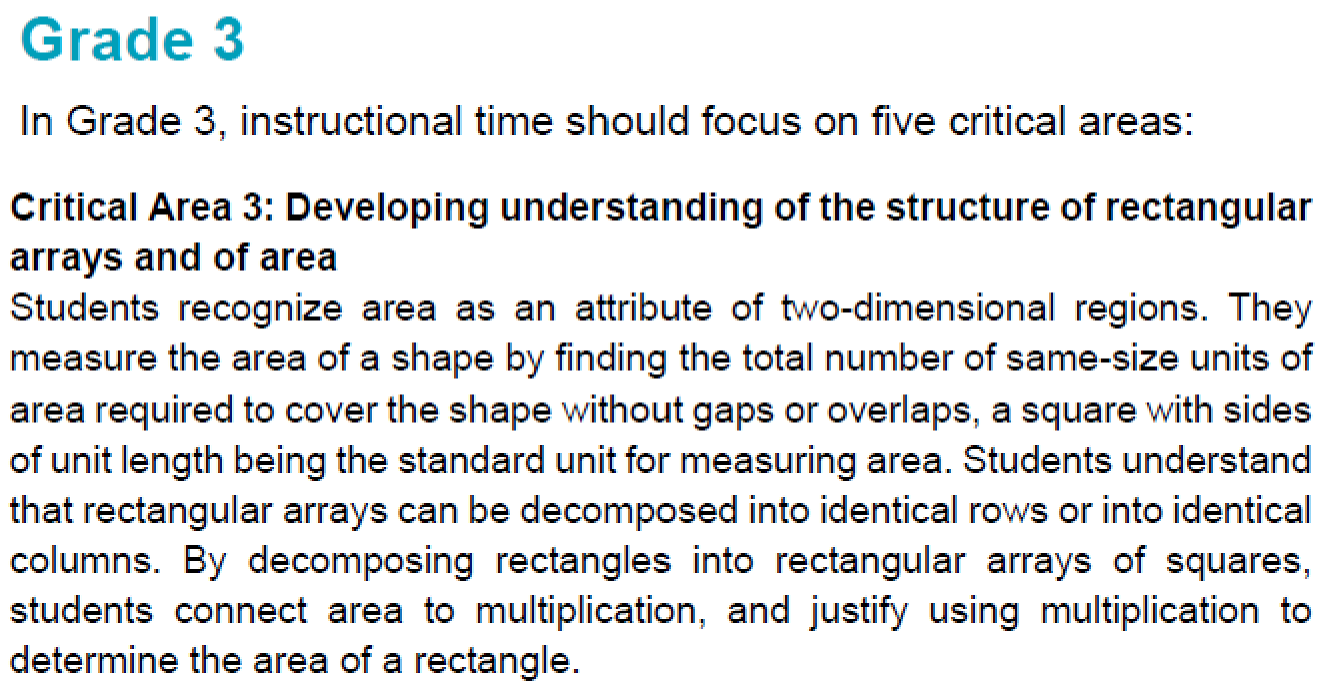
7 hours for 6 small lawns
7 hours for 4 medium lawns
16 hours for 5 large lawns

Code a table showing how many of each size lawns can be mowed in a 32 hour through 40 hour work week.

If get $10 for small yard, $20 for medium size yard, and $30 for large lawn, which type of lawn is most profitable.

Show calculations in a table.

Structure of rectangular arrays



Grade 3

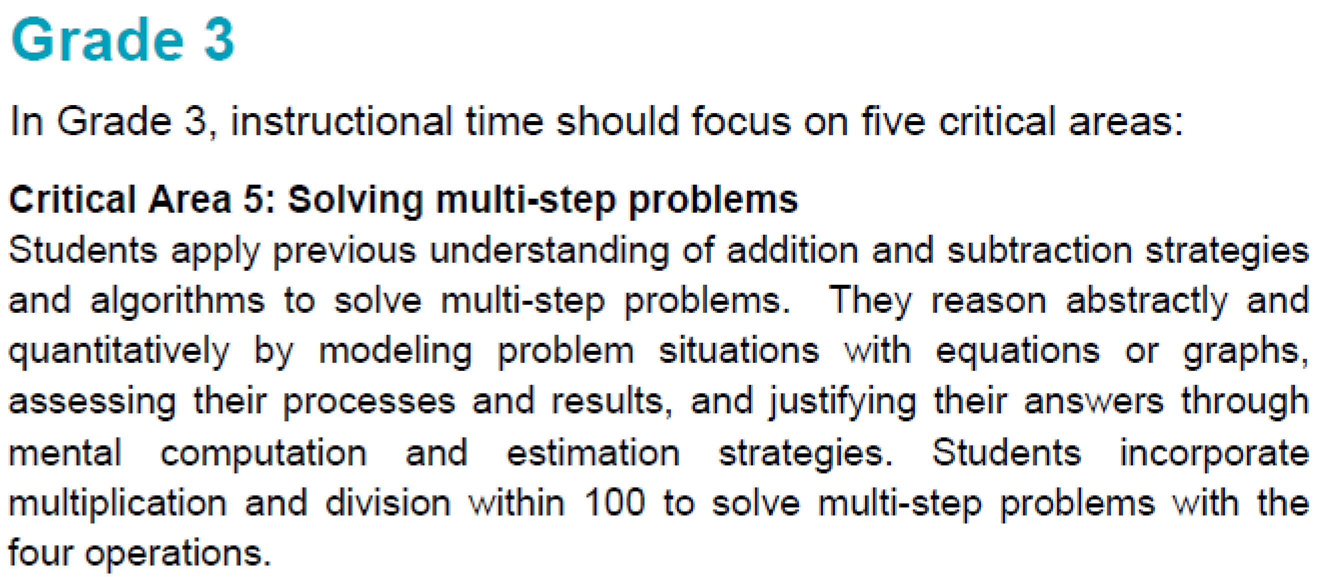
In Grade 3, instructional time should focus on five critical areas:

Critical Area 3: Developing understanding of the structure of rectangular arrays and of area

Students recognize area as an attribute of two-dimensional regions. They measure the area of a shape by finding the total number of same-size units of area required to cover the shape without gaps or overlaps, a square with sides of unit length being the standard unit for measuring area. Students understand that rectangular arrays can be decomposed into identical rows or into identical columns. By decomposing rectangles into rectangular arrays of squares, students connect area to multiplication, and justify using multiplication to determine the area of a rectangle.

Students write code to draw unit squares to represent a rectangle that is 6 units across and 7 units down. The code should include a variable that counts each unit square that is drawn. The student then adjusts the code for a 7 across 6 down rectangle. Each student then is given their own problem to solve (x by y units).

Solving multi-step problems



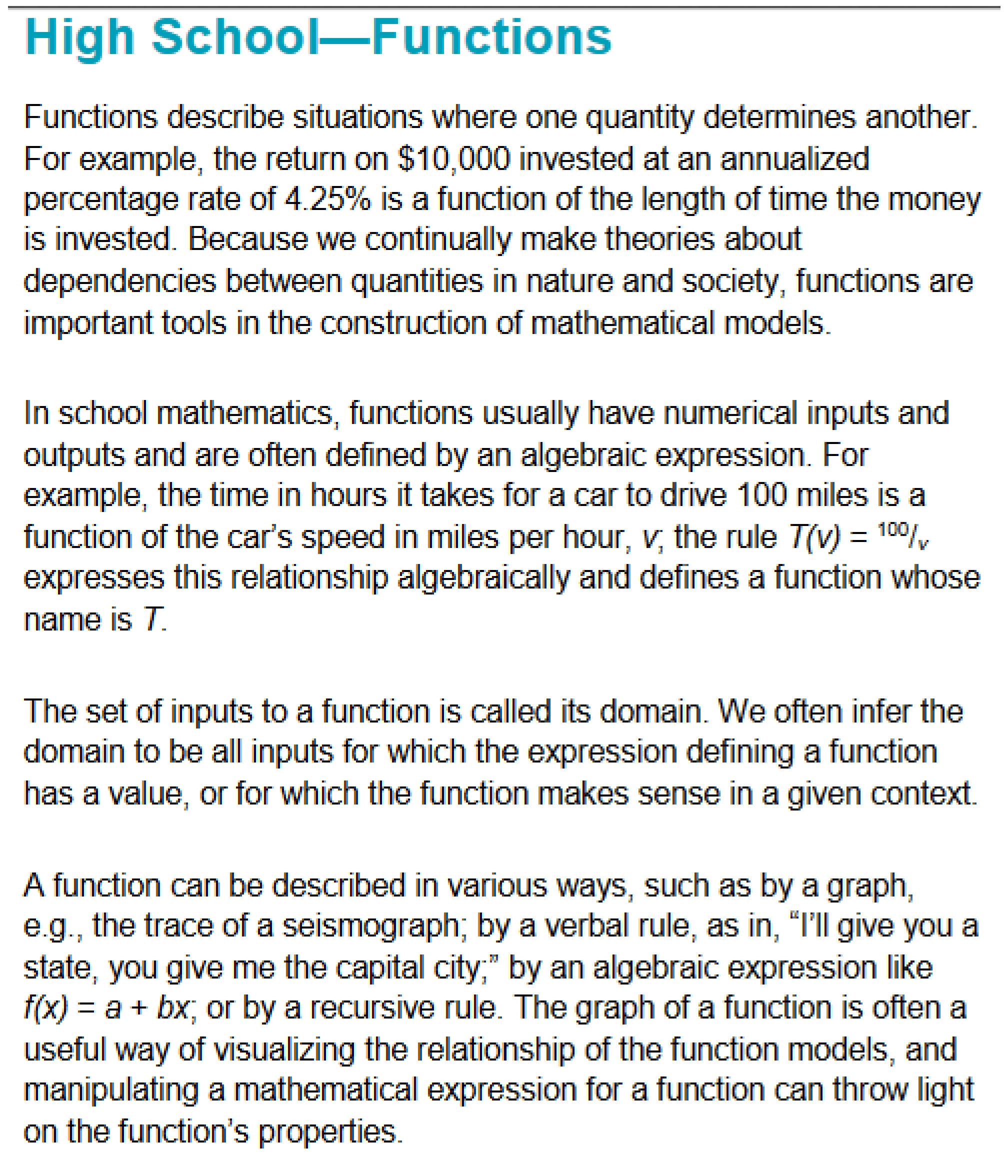
Grade 3

In Grade 3, instructional time should focus on five critical areas:

Critical Area 5: Solving multi-step problems
Students apply previous understanding of addition and subtraction strategies and algorithms to solve multi-step problems. They reason abstractly and quantitatively by modeling problem situations with equations or graphs, assessing their processes and results, and justifying their answers through mental computation and estimation strategies. Students incorporate multiplication and division within 100 to solve multi-step problems with the four operations.

Working in pairs, each pair of students is assigned a word problem recently discussed in the curriculum. The students write code to accurately solve the problem. They then are given different values to incorporate into the problem and verify that their code still produces the correct results. When applicable, the students graph the values use in and results from solving the problem.

https://education.ohio.gov/getattachment/Topics/Learning-in-Ohio/Mathematics/Ohio-s-Learning-Standards-in-Mathematics/MATH-Standards-2017.pdf.aspx pg 72 High School Functions



High School—Functions

Functions describe situations where one quantity determines another. For example, the return on \$10,000 invested at an annualized percentage rate of 4.25% is a function of the length of time the money is invested. Because we continually make theories about dependencies between quantities in nature and society, functions are important tools in the construction of mathematical models.

In school mathematics, functions usually have numerical inputs and outputs and are often defined by an algebraic expression. For example, the time in hours it takes for a car to drive 100 miles is a function of the car's speed in miles per hour, v; the rule $T(v) = {}^{100}/_{v}$ expresses this relationship algebraically and defines a function whose name is T.

The set of inputs to a function is called its domain. We often infer the domain to be all inputs for which the expression defining a function has a value, or for which the function makes sense in a given context.

A function can be described in various ways, such as by a graph, e.g., the trace of a seismograph; by a verbal rule, as in, "I'll give you a state, you give me the capital city;" by an algebraic expression like $f(x) = a + bx$; or by a recursive rule. The graph of a function is often a useful way of visualizing the relationship of the function models, and manipulating a mathematical expression for a function can throw light on the function's properties.

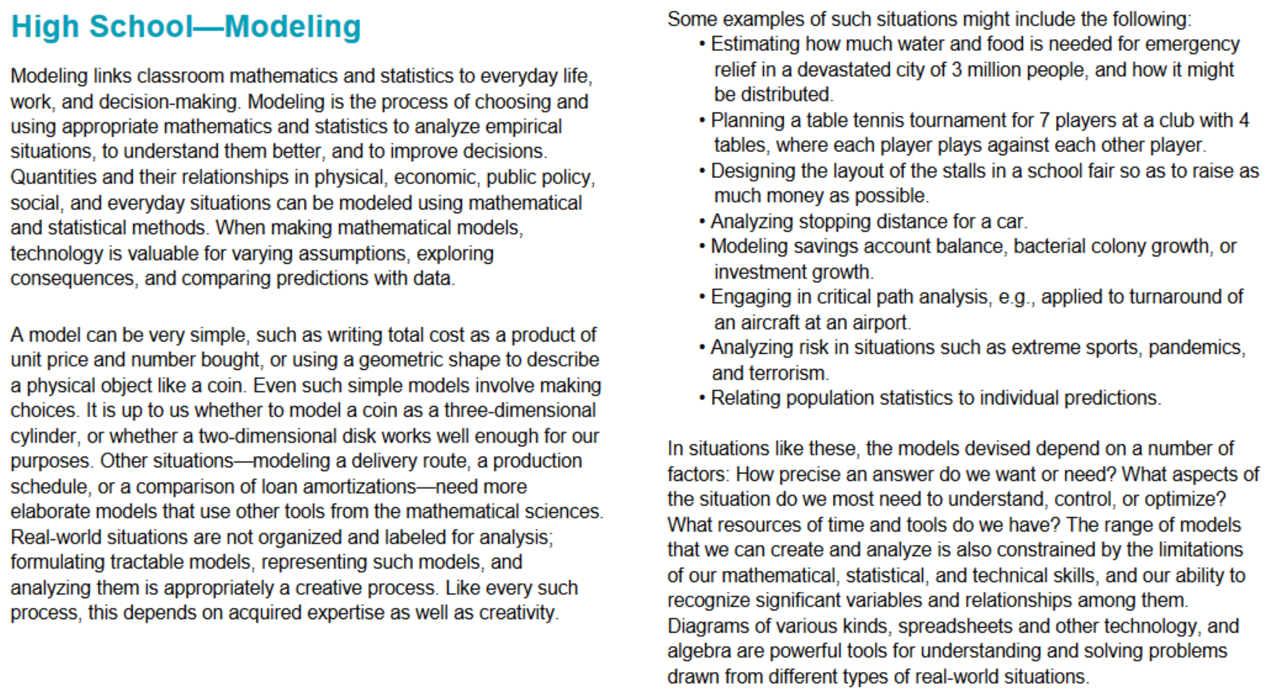
Activities …

Code a function (show in a text based and also block based language). Do the calculation. Watch for undefined.

Looping (for/while) -- Domain is your loop

Generate a table of the values

Graph the values for a visual representation https://education.ohio.gov/getattachment/Topics/Learning-in-Ohio/Mathematics/Ohio-s-Learning-Standards-in-Mathematics/MATH-Standards-2017.pdf.aspx pg 60 High School Modeling



High School—Modeling

Modeling links classroom mathematics and statistics to everyday life, work, and decision-making. Modeling is the process of choosing and using appropriate mathematics and statistics to analyze empirical situations, to understand them better, and to improve decisions. Quantities and their relationships in physical, economic, public policy, social, and everyday situations can be modeled using mathematical and statistical methods. When making mathematical models, technology is valuable for varying assumptions, exploring consequences, and comparing predictions with data.

A model can be very simple, such as writing total cost as a product of unit price and number bought, or using a geometric shape to describe a physical object like a coin. Even such simple models involve making choices. It is up to us whether to model a coin as a three-dimensional cylinder, or whether a two-dimensional disk works well enough for our purposes. Other situations—modeling a delivery route, a production schedule, or a comparison of loan amortizations—need more elaborate models that use other tools from the mathematical sciences. Real-world situations are not organized and labeled for analysis; formulating tractable models, representing such models, and analyzing them is appropriately a creative process. Like every such process, this depends on acquired expertise as well as creativity.

Some examples of such situations might include the following:

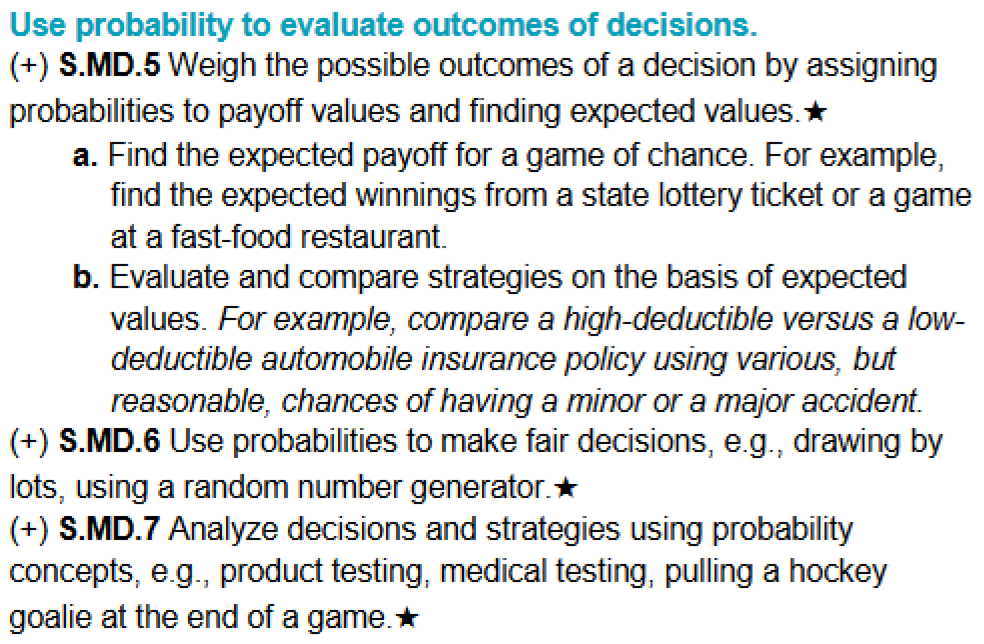
- Estimating how much water and food is needed for emergency relief in a devastated city of 3 million people, and how it might be distributed.
- Planning a table tennis tournament for 7 players at a club with 4 tables, where each player plays against each other player.
- Designing the layout of the stalls in a school fair so as to raise as much money as possible.
- Analyzing stopping distance for a car.
- Modeling savings account balance, bacterial colony growth, or investment growth.
- Engaging in critical path analysis, e.g., applied to turnaround of an aircraft at an airport.
- Analyzing risk in situations such as extreme sports, pandemics, and terrorism.
- Relating population statistics to individual predictions.

In situations like these, the models devised depend on a number of factors: How precise an answer do we want or need? What aspects of the situation do we most need to understand, control, or optimize? What resources of time and tools do we have? The range of models that we can create and analyze is also constrained by the limitations of our mathematical, statistical, and technical skills, and our ability to recognize significant variables and relationships among them. Diagrams of various kinds, spreadsheets and other technology, and algebra are powerful tools for understanding and solving problems drawn from different types of real-world situations.

Activities …

Get some raw statistics, run those through an analysis process

Stalls … given stall of x by y and walkway of z between stalls, How many stalls long/wide for an area of a by b? --- need to work this better … model will generate a table and find best result https://education.ohio.gov/getattachment/Topics/Learning-in-Ohio/Mathematics/Ohio-s-Learning-Standards-in-Mathematics/MATH-Standards-2017.pdf.aspx pg 89 High School Probability and Statistics



Use probability to evaluate outcomes of decisions.

(+) **S.MD.5** Weigh the possible outcomes of a decision by assigning probabilities to payoff values and finding expected values.★

- **a.** Find the expected payoff for a game of chance. For example, find the expected winnings from a state lottery ticket or a game at a fast-food restaurant.
- **b.** Evaluate and compare strategies on the basis of expected values. *For example, compare a high-deductible versus a low-deductible automobile insurance policy using various, but reasonable, chances of having a minor or a major accident.*

(+) **S.MD.6** Use probabilities to make fair decisions, e.g., drawing by lots, using a random number generator.★

(+) **S.MD.7** Analyze decisions and strategies using probability concepts, e.g., product testing, medical testing, pulling a hockey goalie at the end of a game.★

Activities …

… More Activities …

Animation

Animation uses mathematics and is an excellent way to teach, illustrate, and practice several mathematical concepts. Some resources …

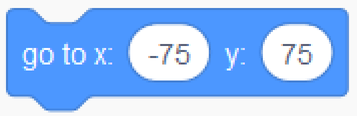
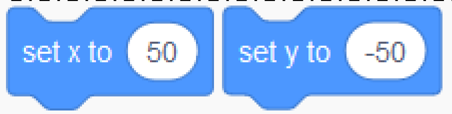
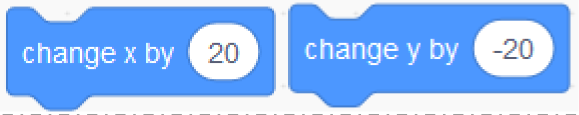
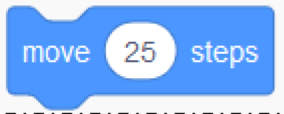
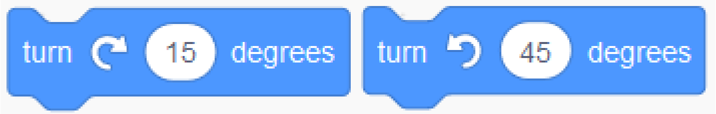
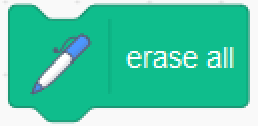
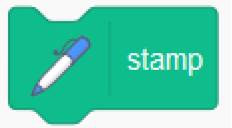
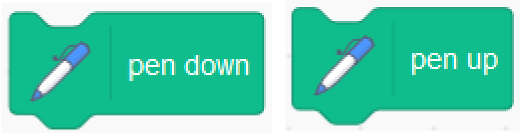
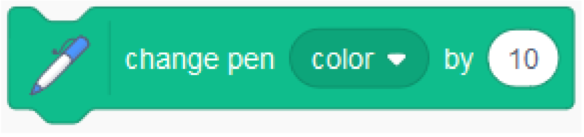
Presentation Notes – See the topic "Animation (Scratch) for some simple animation activities in Scratch to explore the coordinate system, transformations, and general math.

<code_by_math> Lessons then the "Code to make an animated-GIF" section. Go through the lessons/examples then create your own animated-GIF. Note the animated-GIFs are below the code.

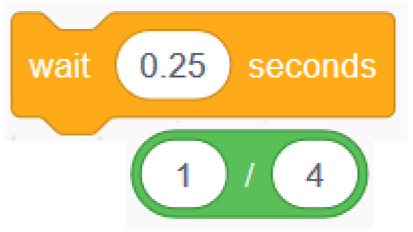
Google's CS First Curriculum --- Animate a Name --- videos on animation in Scratch
https://csfirst.withgoogle.com/c/cs-first/en/animate-a-name/overview.html

Graphics / Drawing – Scratch

Lower left corner – click on “Add Extension” and add the Pen extension

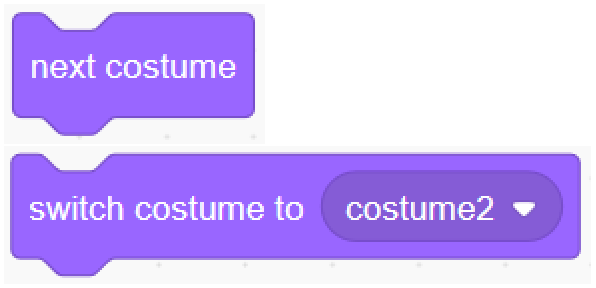
 `go to x: -75 y: 75`	Move to the specified location.
 `set x to 50` `set y to -50`	Move a specific x or specific y position
 `change x by 20` `change y by -20`	Move either the x or y position by the specified number of pixels
 `move 25 steps`	Go the specified number of pixels in the current direction.
 `turn ↻ 15 degrees` `turn ↺ 45 degrees`	Turn right or left the specified number of degrees.
 `erase all`	Erase all pen marks and stamps.
 `stamp`	Stamp (leave a copy of) the current sprite at the current location.
 `pen down` `pen up`	Pick up or put down the pen. Drawing or not drawing.
 `change pen color ▾ by 10`	Alter the pen color

Animation – Scratch



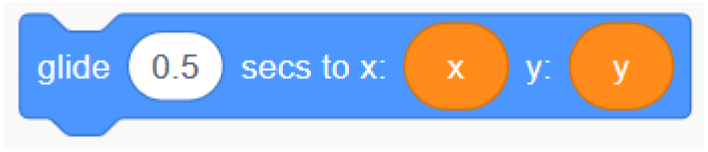
```
wait (0.25) seconds
(1) / (4)
```

Pause between movement for effect. Can explore decimal/fractions and even use variables to progressively get faster/slower in the loop.



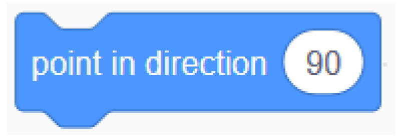
```
next costume
switch costume to (costume2 ▾)
```

A costume is one graphic. Switch between graphics to give the illusion of movement (i.e. wings flapping, legs moving). Use with if and/or wait block for different looks.



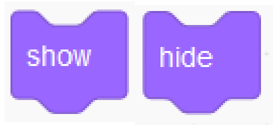
```
glide (0.5) secs to x: (x) y: (y)
```

Move to the new location over the specified amount of time.



```
point in direction (90)
```

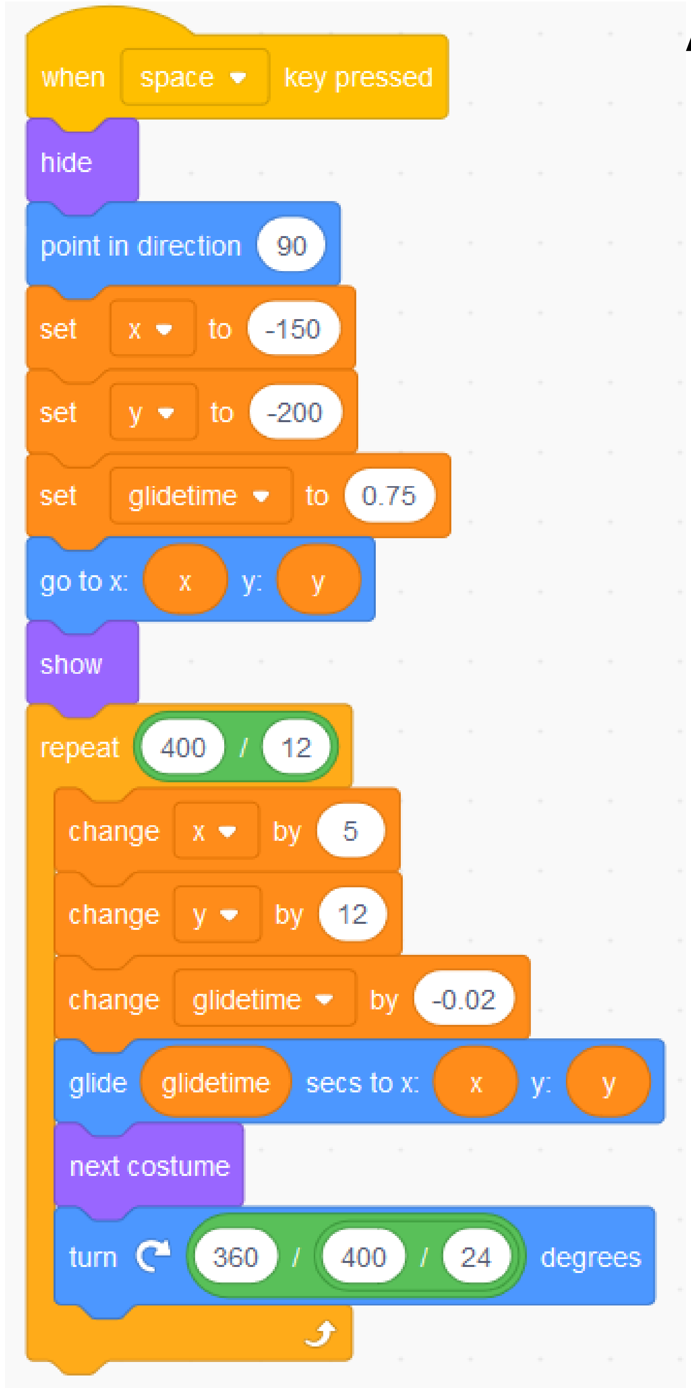
Point sprite in the desired direction. 90 is straight ahead. Click on the number for a degree dial. Positive or negative degrees.



```
show
hide
```

Show or hide the sprite. When drawing, usually hide it and show when doing an animation.

Animation Illustration



```
when [space] key pressed
hide
point in direction (90)
set [x] to (-150)
set [y] to (-200)
set [glidetime] to (0.75)
go to x: (x) y: (y)
show
repeat ((400) / (12))
    change [x] by (5)
    change [y] by (12)
    change [glidetime] by (-0.02)
    glide (glidetime) secs to x: (x) y: (y)
    next costume
    turn ↻ ((360) / ((400) / (24))) degrees
```

<u>For illustration purposes only …</u>

- Variables x and y hold the current position. Adjust to explore the coordinate system.
- Point in different directions. 0 to 360 with 90 straight ahead.
- Hide/Show lets us move without being seen.
- Want to travel from -200 to 200 on the y axis changing by 12 each time. Calculate that in the repeat statement.
- Above point … could use variables for how far to go and how many steps to move.
- x/y change is slope. Can tie in and explore slope further.
- Next costume switches to give the animation. Have students create a 5 costume animation for better effect.
- Glide time changes for illusion of going faster. Could calc.
- The turn gives the illusion of doing a front flip. Notice the calculation (discuss why and try alterations).
- Activity: Change so start upper left and go to lower right.

Patterns / Tiling / Tessellations

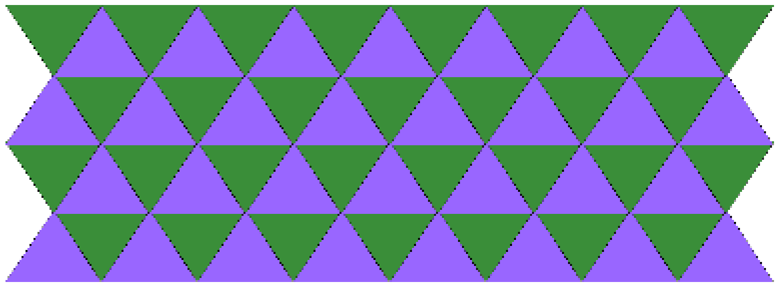
Explore patterns, shapes, and how they fit together in tiling or tessellations.

Research/explore:

- What is a tessellation?
- What are the following three (and what shapes in each): regular, semi-regular, and irregular tessellation?

Do It:

- Tile the screen with squares at least 10 by 10
- Tile the screen with equilateral triangles at least 10 wide, 10 high.
- Color the tiles above so they alternate between two colors.
- Create a semi-regular tessellation.
- Create a irregular tessellation.



Spreadsheet Automation (i.e. coding)

Spreadsheets are used as tools for various mathematical activities. Sometimes repetitive tasks are necessary to get to the analysis of the data, charting of the data, etc.   Coding can help move to the analysis quicker and while getting the benefits of designing and writing code.

Most schools likely using Excel or Google Sheets. Both Excel and Google Sheets have their own built-in scripting language.

There are also ways to use code outside of the spreadsheet to alter the spreadsheet. See Al Sweigart's Automate the Boring Stuff book (in Reference section) for examples of how to do this.

Automation could include cleaning up the data to put into the spreadsheet, auto-building the spreadsheet, adding in formulas, and creating charts.

Puzzles

- Some examples include word search, Cryptogram, Sudoku, Numbrix, Wordscapes
- General steps to solve puzzles
  - Do a few of the beginner puzzles on paper
  - Define the steps used to solve the puzzle
  - Decide how to represent the puzzle in code
  - Code a routine to load up different puzzles
  - Code a routine to display the puzzle
  - Code each of the steps to solve the puzzle, one at a time
  - Test and validate your puzzle solving solution
- Where do you find puzzle problems? Ask students what puzzles they do. Internet search. Go to a bookstore or book section of a store (including a dollar store) and look at the puzzle books. Pick one or two if financially able to.

Start with solving the basic/simple versions of the puzzles. Refine the code for harder puzzles.

Puzzles – word search

- Word search is a good puzzle project to do early on.
- Without the computer, how do you find a word?
- Describe in English how you would tell the computer to do so.
- Load the word search puzzle into an array
- Display the puzzle on the screen

```
# puzzle is 6x6 two dimensional array
rows = 6
cols = 6
puzzle = [['a','b','c','d','e','f'],
          ['g','h','i','j','k','l'],
          ['m','n','o','p','q','r'],
          ['s','t','u','v','w','x'],
          ['y','z','a','b','c','d'],
          ['e','f','g','h','i','j']]

# display the puzzle
def display_puzzle():
    # display the puzzle
    for i in range(rows):
        oneline = ""
        for j in range(cols):
            oneline = oneline + puzzle[i][j]
        print(oneline)

display_puzzle()
```

One approach, not optimal but illustrates a way to approach it.

Puzzles – word search

- Decided to create a found array that will display just the found word(s) to illustrate where they are in the puzzle
- The reset_found puts a period in all of the locations
- Could have instead displayed the location of the first letter of the found word(s)

```
found = [['a','b','c','d','e','f'],
         ['g','h','i','j','k','l'],
         ['m','n','o','p','q','r'],
         ['s','t','u','v','w','x'],
         ['y','z','a','b','c','d'],
         ['e','f','g','h','i','j']]

def reset_found():
    # display the found puzzle
    for i in range(rows):
        for j in range(cols):
            found[i][j] = "."

def display_found():
    # display the found puzzle
    for i in range(rows):
        oneline = ""
        for j in range(cols):
            oneline = oneline + found[i][j]
        print(oneline)
```

If doing this activity with your students, clearly define the program specificiations.

Puzzles – word search

- For this illustration, only doing the check for a word going from right to left (backwards), rest for your class to complete.
- Add your checks with the find_string's check_back call.
- find_string goes through each letter of the puzzle, if that letter matches the first letter of the word to find, it runs the check(s).
- check_back returns True if found and false if not. The find_string can use that result if desired.

```
def find_string(str2find):
    firstletter = str2find[0]
    for i in range(rows):
        for j in range(cols):
            if puzzle[i][j] == firstletter:
                cb = check_back(str2find,i,j)

def check_back(findit, x, y):
    result = False
    findlen = len(findit)
    if (y + 1 - findlen) >= 0:
        result = True
        for i in range(len(findit)):
            if puzzle[x][y-i] != findit[i]:
                result = False
        if (result):
            for i in range(len(findit)):
                found[x][y-i] = puzzle[x][y-i]
    return result
```

A lot of ways to break this project up into pieces for groups or individuals to work on.

Puzzles – word search

- reset_found starts the found array with all periods
- Asking for the word to find
- Run the find_string function which will place the found string in the found array
- Display the found array

```
reset_found()
findit = input("Word to find: ")
find_string(findit)
display_found()
```

Could read a list of words from a file, find each, then provide the found location for each.

Puzzles – word search - enhancements

- Build the puzzle array by reading from a text file, puzzle can be any size. Likewise the found array would be dynamically created.
- Read the words to find from a file and write out the location each word is found.
- Look at algorithms and approaches by others. How does their approach differ from yours? How is it similar?
- Build a timing mechanism that records the time the code starts, completes, and the difference between those two.
- See which group's code runs the fastest. Why is that the fastest?
- Do a small, medium, and large word find puzzle and record the time for each group to solve each puzzle. Is one the fastest of all sizes?
- Graphically display the word search and circle the found words graphically. Try either Turtle graphics or some other drawing method.

The learning process includes mistakes. What works and what does not work?

Games

- List (or one slide) of possible game projects
- General – steps to build/solve games
    - Build/load/save/display the games
    - Identify / code the rules of the game
    - Code play of the game with human players
    - Identify / figure out strategy to play the code
    - Code machine player of the game
    - Play human vs machine
- Tic-Tac-Toe … good one to start with to work through some mechanics.
- Where do you find game projects?
- How does this fit into the curriculum and big picture?
- Search/research into others algorithms and strategies.

… red zone notes here …

Games/Puzzles

Students write code to solve puzzles and play mathematically sound games. The students need to understand the puzzle/game well enough to figure out a solution/strategy/algorithm, then put that into code.

Game of 45
Pick a number between 1 and 7 which is added to the running total. The first player to 45 wins the game.

Player 1: Start with the number 3
Player 2: 6 + 3 is 9
…
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

Common Puzzles/Games
Sudoku
Tic-Tac-Toe
Word Find
Cryptoquote

Games/Puzzles

Students write code to solve puzzles and play mathematically sound games. The students need to understand the puzzle/game well enough to figure out a solution/strategy/algorithm, then put that into code.

Game of 45
Pick a number between 1 and 7 which is added to the running total. The first player to 45 wins the game.

Player 1: Start with the number 3
Player 2: 6 + 3 is 9
…
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

Common Puzzles/Games
Sudoku
Tic-Tac-Toe
Word Find
Cryptoquote

Permutations and Combinations

Given the letters rdrvie, what are all the possible ways those letters can be arranged? What 6 letter words are in those letters? 5 letter words? 4 letter words?

The local league needs some code to build the league schedule. The number of teams in the league changes. The code should be able to handle either an odd or even number of teams. It should be able to handle both play once and home-and-home scenarios.

Assuming a combination lock goes left then right then left and the dial has 39 numbers on the dial, list all of the possible ways the combination could be opened. If it takes 4.5 seconds to try each possible way, how long to try all possible ways?

Random Numbers

A few of many uses for random numbers

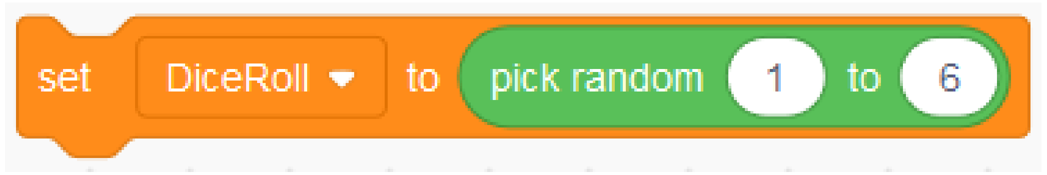
- For probability and statistics courses, can generate data sets of random numbers to study and work with. One thousand rolls of the dice, one million flips of a coin, one year of pick 5 numbers.

- Random number can be used in simulations for different results of the simulation. Can devise formulas that provide a for simulating the likelihood of specific events happening. For instance, if simulating 1000 births and the odds of having twins is 1/250, could adjust your random generation accordingly.

- Gaming using random numbers to provide for the chance in the game. Most game coding uses a lot of math and physics.

Random Numbers

generating pseudo-random numbers



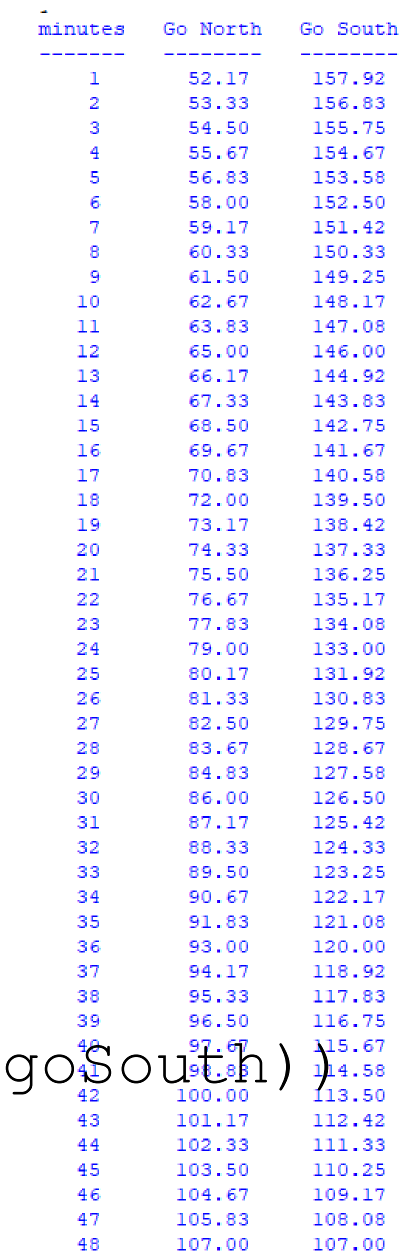
```
import random

random.seed()

print(random.random())
print(random.randint(1, 6))
```

- Use integers to get back integers.
- Use decimal numbers to get back decimal values.

- Python has many more options
- Can seed in different ways or not include a seed command.
- random.random() generates a decimal number starting at zero and going through less than one.
- random.randint generates an integer between and including the starting and ending value (i.e. 1 to 6 in this example).

Looping / Repetition - conditional

```
minutes = 0
goSouth = 159
goNorth = 51

print('minutes', '  Go North', '  Go South')
print('-------', '  --------', '  --------')

while goSouth > goNorth:
    minutes = minutes + 1
    goSouth = 159 - (minutes * (65 / 60))
    goNorth = 51 + (minutes * (70 / 60))

    print("%5d %11.2f %10.2f" % (minutes, goNorth, goSouth))
```



```
minutes   Go North   Go South
-------   --------   --------
    1      52.17     157.92
    2      53.33     156.83
    3      54.50     155.75
    4      55.67     154.67
    5      56.83     153.58
    6      58.00     152.50
    7      59.17     151.42
    8      60.33     150.33
    9      61.50     149.25
   10      62.67     148.17
   11      63.83     147.08
   12      65.00     146.00
   13      66.17     144.92
   14      67.33     143.83
   15      68.50     142.75
   16      69.67     141.67
   17      70.83     140.58
   18      72.00     139.50
   19      73.17     138.42
   20      74.33     137.33
   21      75.50     136.25
   22      76.67     135.17
   23      77.83     134.08
   24      79.00     133.00
   25      80.17     131.92
   26      81.33     130.83
   27      82.50     129.75
   28      83.67     128.67
   29      84.83     127.58
   30      86.00     126.50
   31      87.17     125.42
   32      88.33     124.33
   33      89.50     123.25
   34      90.67     122.17
   35      91.83     121.08
   36      93.00     120.00
   37      94.17     118.92
   38      95.33     117.83
   39      96.50     116.75
   40      97.67     115.67
   41      98.83     114.58
   42     100.00     113.50
   43     101.17     112.42
   44     102.33     111.33
   45     103.50     110.25
   46     104.67     109.17
   47     105.83     108.08
   48     107.00     107.00
```

Looping / Repetition - conditional

Based on true story with some alterations: My son forgot his retainers. Rather than one person make a long drive, each of us would leave at the same time and drive until we met. He started at mile marker 51 and drove north, I started at mile marker 159 and drove south. He went 70 mph, I went 65 mph.
How many minutes did we drive until we met? What mile marker did we meet at?

<u>What if? How Does This Change?</u>

- Son drives 90 mph, I drive 85 mph?
- 100/45 mph? Other mph, look at trend in changes.
- I start 5 minutes before him. Alter values.
- Mile 140 to 135, I drive 35 mph (construction).
- Find in quarter, tenths, 5 hundreds of a minute.

<u>Discussion/Learning Points</u>

- Estimate the answer
- Calc mph
- Map and mile markers on I-75
- Value of a model like this
- How much time do you save speeding?
- What would be other good variables to use?

Coding, Databases, Math

Databases (Relational) can be used …

- Teach specific mathematical concepts
  - Logic – explore using AND, OR, NOT in the selection of data
  - Set theory – JOINS – inner, outer
- Use with Real-World Projects
  - Provide a data store for mathematical coding projects
  - Efficiently retrieve and summarize stored data

About Relational Databases

- SQL (Structured Query Language)
- There are different dialects and subtle differences between the various databases SQL implementation

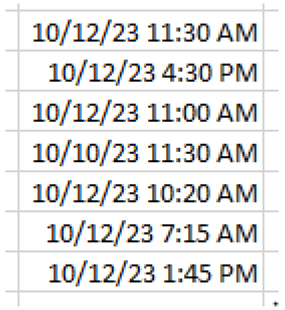
Learn initially with a small data set then work up to larger ones.

Student Research: E.F. Codd – mathematics behind relational databases

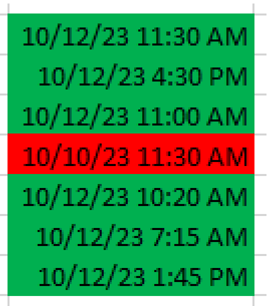
Data Visualization

Tools: Spreadsheet like Excel or Google Sheets, visualization library like matplotlib

Color matters … which is easier to spot the item older than 24 hours



10/12/23 11:30 AM
10/12/23 4:30 PM
10/12/23 11:00 AM
10/10/23 11:30 AM
10/12/23 10:20 AM
10/12/23 7:15 AM
10/12/23 1:45 PM



🟩 10/12/23 11:30 AM
🟩 10/12/23 4:30 PM
🟩 10/12/23 11:00 AM
🟥 10/10/23 11:30 AM
🟩 10/12/23 10:20 AM
🟩 10/12/23 7:15 AM
🟩 10/12/23 1:45 PM

Bar and line graphs have their place, sometimes a different visualization makes sense.

US Population Chart removed from here
Do _not_ have permission to publish.

https://matplotlib.org/stable/gallery/index.html

Big Datasets

- ❑ Acquire data
    - ❑ Capture own data
    - ❑ Local/regional data from other sources
    - ❑ National/international data sets
    - ❑ Good vs not good data
    - ❑ Do a few examples
- ❑ Quality of data
- ❑ Tools and Techniques to deal with bigger datasets

Good for students to see larger datasets and real world data

Pulling Data from External Sources (Quandl)

- Many external sources for data available via an API or REST web service. Most require registration and/or a key.
- Some data sources free while others require payment
- Can pull the data and save it in a format that students can read from … JSON, XML, CSV, pickle, etc.
- Quandl used as the data source per the code reference below. Many other data source options.

```
# May need to install these and other modules
import pandas as pd
import quandl

# quandl key in text file
api_key = open('quandlapikey.txt','r').read()

# https://www.quandl.com/data/FRED/NROUST-Natural-Rate-of-Unemployment-Short-Term
df = quandl.get('FRED/NROUST', authtoken=api_key)
print(df.head(20))

# Above pulls the data from Quandl then displays the first 20 rows
# Continued on the next slide
```

Code from https://pythonprogramming.net/ - Data Analysis with Pandas

Processing Data from External Sources

- Once you have the data, can write it to a file or do whatever processing is desired.

```
# Continued from previous slide
# print out the largest and smallest values with their date
largest = 0
smallest = 10
for index, row in df.iterrows():
    if row[0] > largest:
        largest = row[0]
        largestdate = index
    if row[0] < smallest:
        smallest = row[0]
        smallestdate = index
#    print(index, row[0])

print('largest: ', largest, ' on ', largestdate)
print('smallest: ' , smallest, ' on ', smallestdate)
```

Very simple example. Can graph or run through more rigorous math analysis methods.

Recursion

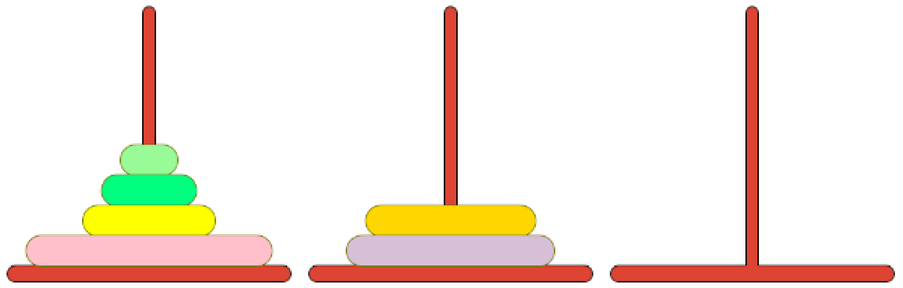
The Functions section of the Math Standards make several references to recursive and recursively defined functions. What is recursion and how does it work?

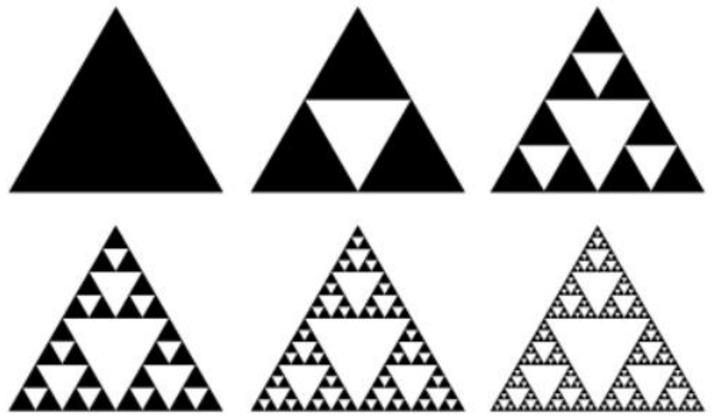
Some common uses of recursion …

0, 1, 1, 2, 3, 5, 8, 13, 21, 34 …

f(n + 1) = f(n) + f(n -1) for n >= 1

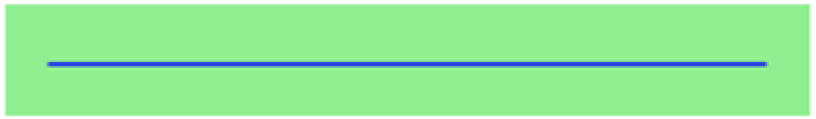
Fibonacci Sequence


Tower of Hanoi puzzle

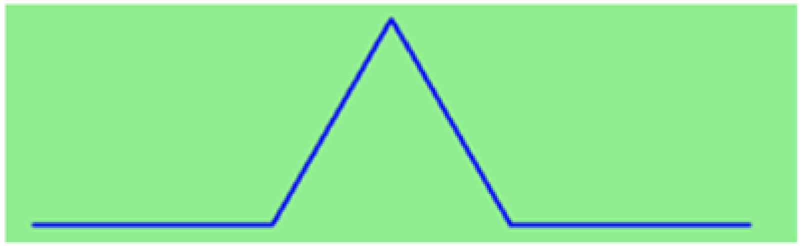

Fractals (example above is Sierpinski Triangle)

Recursion - Fractals

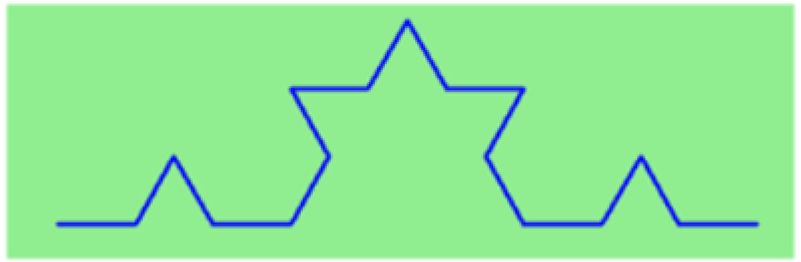
order 0 Koch fractal



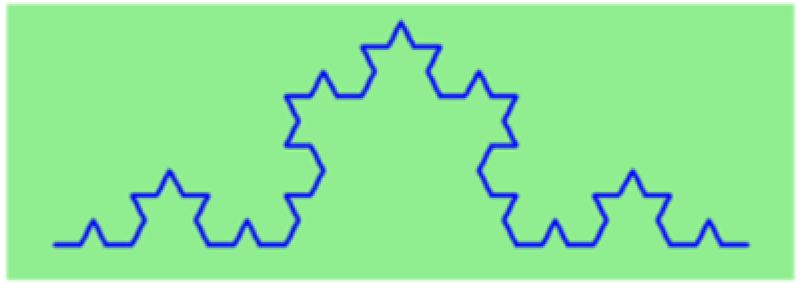
order 1 Koch fractal

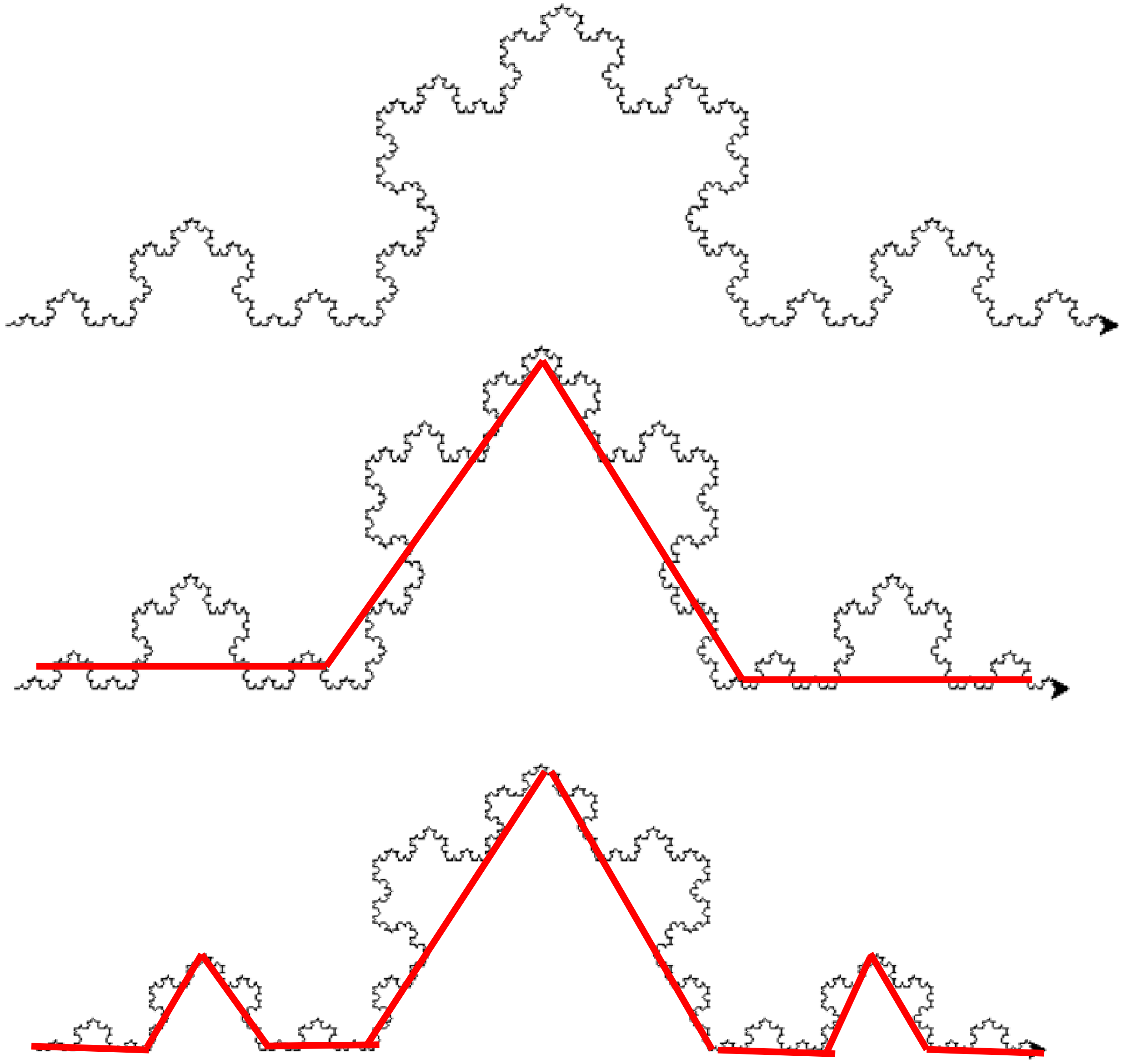


order 2 Koch fractal:



order 3 Koch fractal:

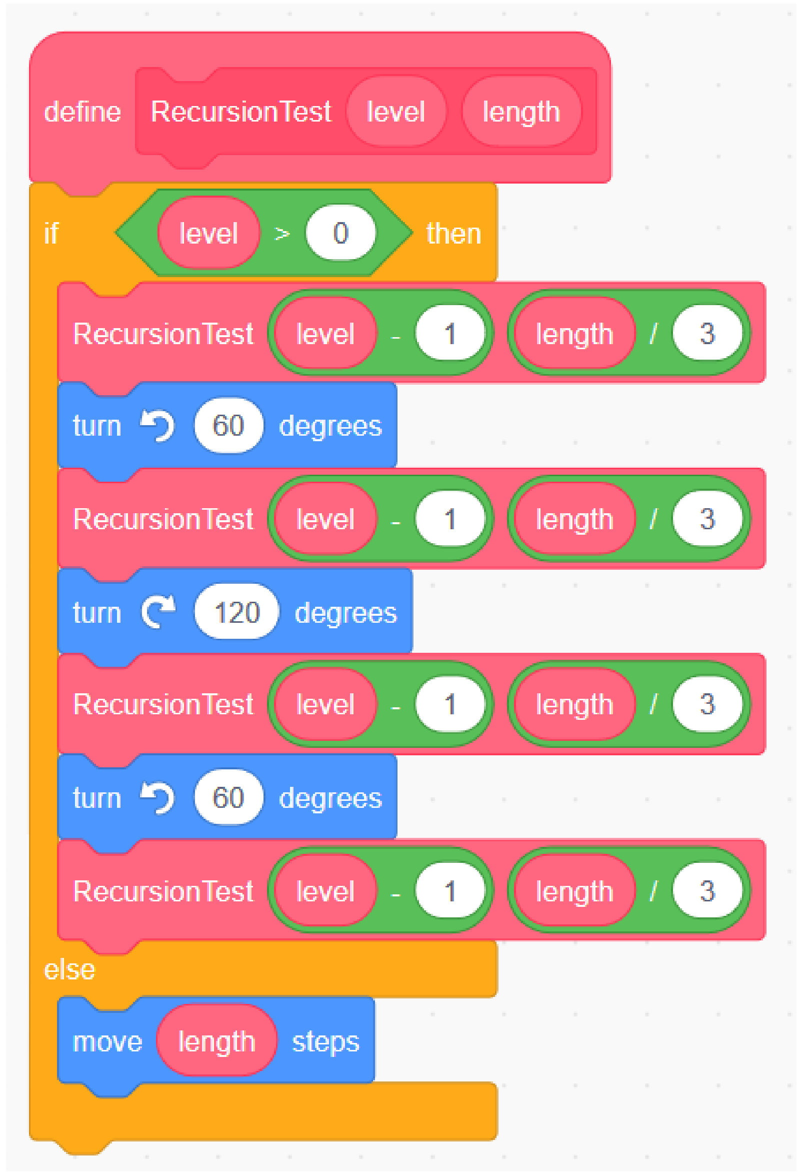




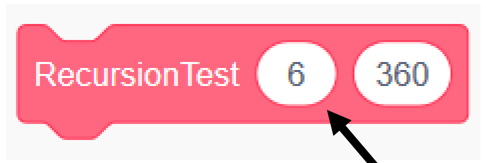
See the pattern?

http://openbookproject.net/thinkcs/python/english3e/recursion.html

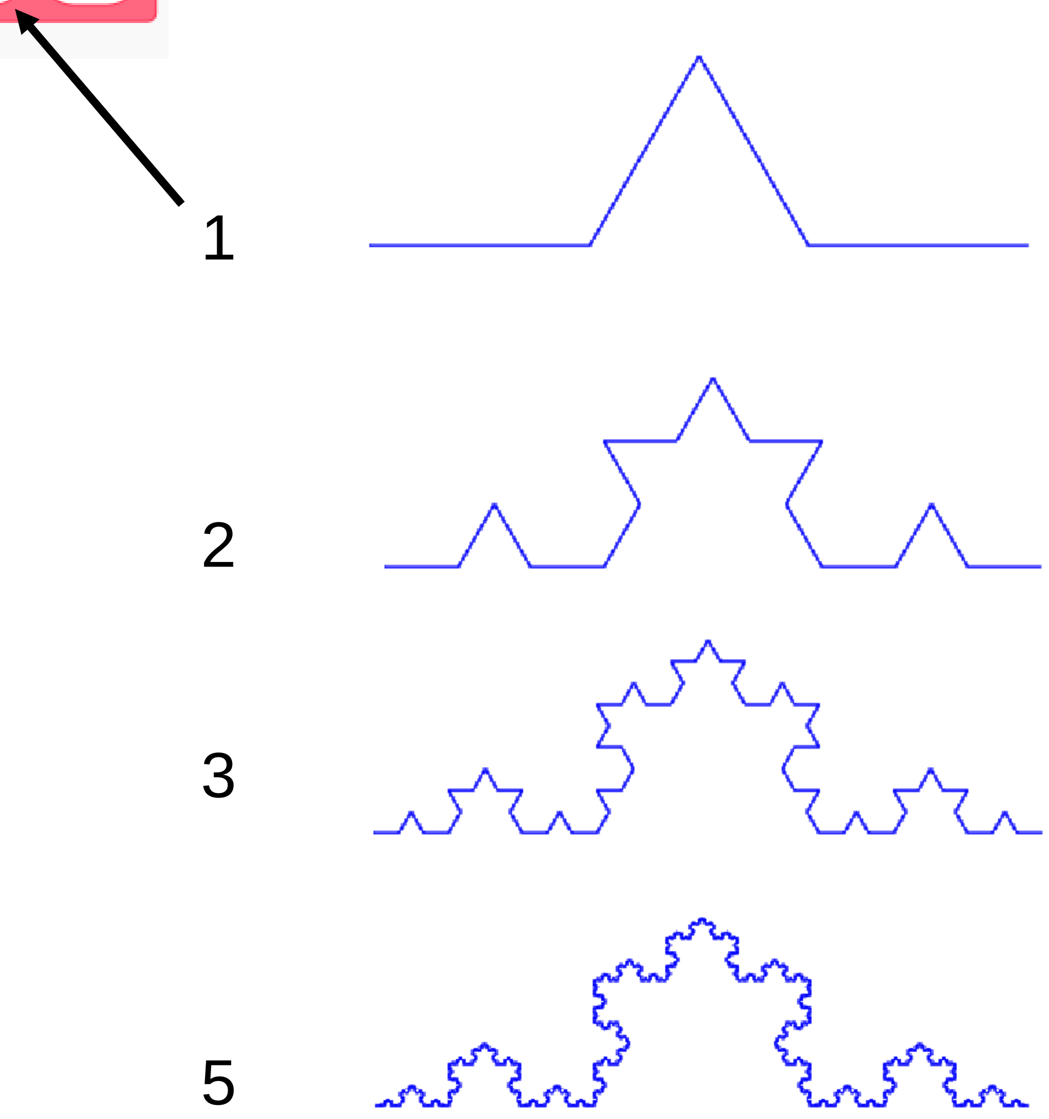
Recursion



```
define RecursionTest (level) (length)
if <(level) > (0)> then
    RecursionTest ((level) - (1)) ((length) / (3))
    turn ↺ (60) degrees
    RecursionTest ((level) - (1)) ((length) / (3))
    turn ↻ (120) degrees
    RecursionTest ((level) - (1)) ((length) / (3))
    turn ↺ (60) degrees
    RecursionTest ((level) - (1)) ((length) / (3))
else
    move (length) steps
```



```
RecursionTest (6) (360)
```

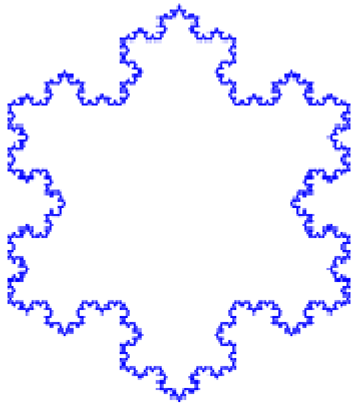


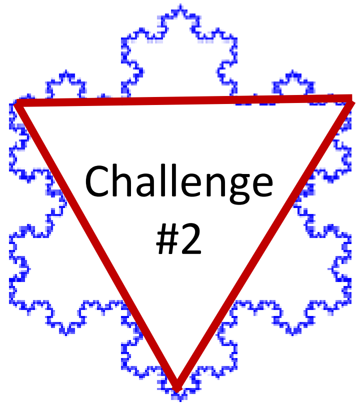
Recursion – Fractal - Challenges

Challenge 2: Koch snowflake is the proper name for the fractal. To achieve that look, there should be three sides (a triangle) rather than just one line. When and how to complete that is your challenge. Try to be as efficient and clean with your code as possible. Try, try, and try some more until you have it.

Challenge 3: Using the 7 line function as the starting point, alter the order 0 fractal from a line to something else. What does the resulting image look like? Try some different order 0 patterns and find one you like. The koch is called four different times in the non-zero section of the code (order 1 and above). Alter that number of calls as well as the base shape. Sketch out what you think it should look like then try to make it look that way.

Challenge 4: Try coding a different fractal without looking at any code for that fractal … just the resulting image and any descriptions on the base shapes and patterns used to make that fractal.





Challenge 2 can be done by adding minimal code.
It took me a little while and a lot of lines of code to figure out that code.

… Basic Coding Items …

Learn these basics and you can code a lot of problems. Became familiar with them in the language you are coding then move on to more advanced items and concepts.

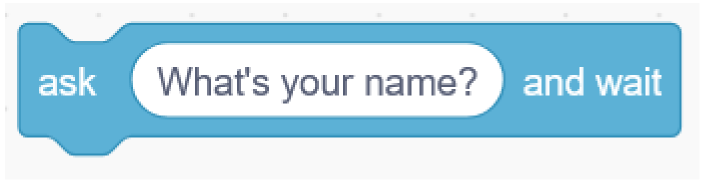
Basics: Input/Output

Types of Input

User types in information

Read information from a file

Read information from a web service



```
f = open("file.txt", "r")
print(f.readlines())
f.close()
```

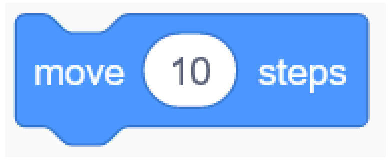
Types of Output

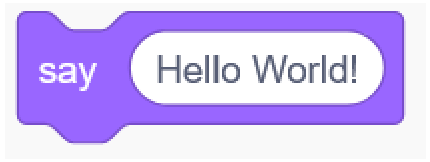
Display a message

Write information to a file

Write information to a web service

Move or draw something on the screen





```
print('Hello World!')
```

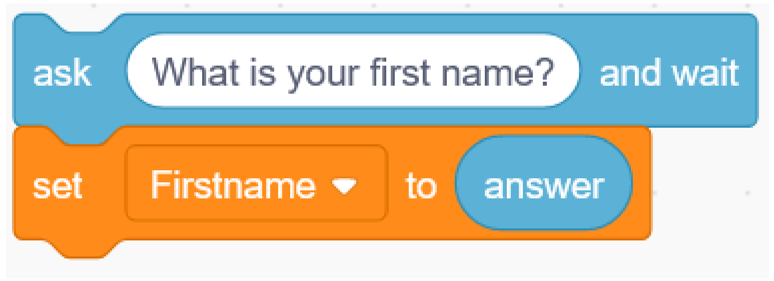
```
f = open("file.txt", "w")
f.writelines("This is a test ...")
f.close()
```

Information fed into the code (input), information the code provides (output)

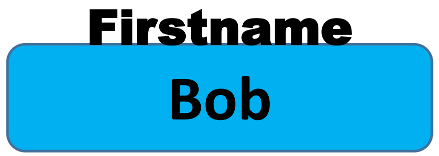
Basics: Variables

Variables can represent things we don't know but are trying to figure out. They can store information the code needs to remember or act upon.

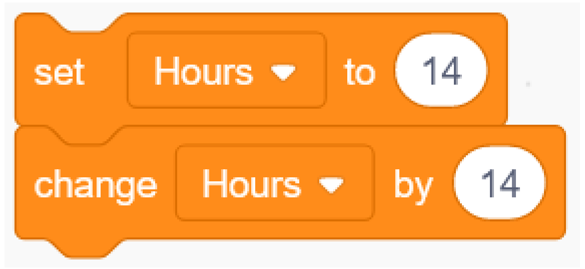
Think of a named box that can store only one value at any time. You can replace the current value with a new one but each box can only store one value.



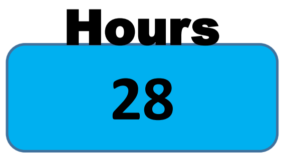
```
ask [What is your first name?] and wait
set [Firstname] to (answer)
```



```
FirstName = input('What is your first name:')
```



```
set [Hours] to (14)
change [Hours] by (14)
```

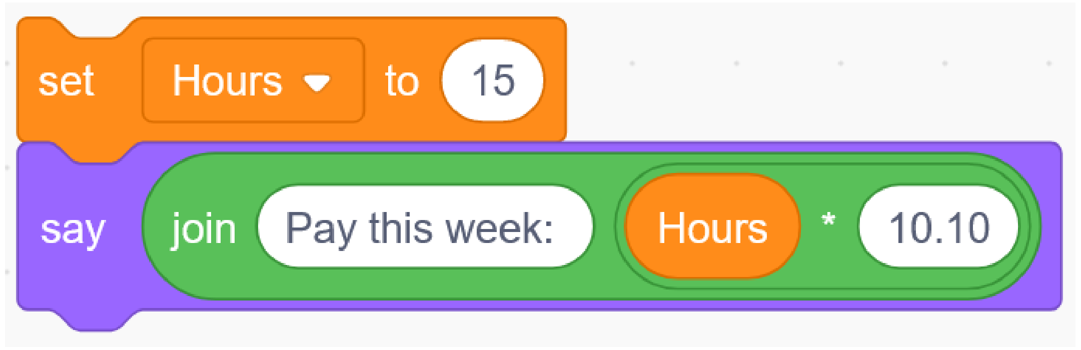


```
Hours = 14
Hours = Hours * 2
print('Double Hours:', Hours)
```

One of the most important parts of coding (and of Algebra)

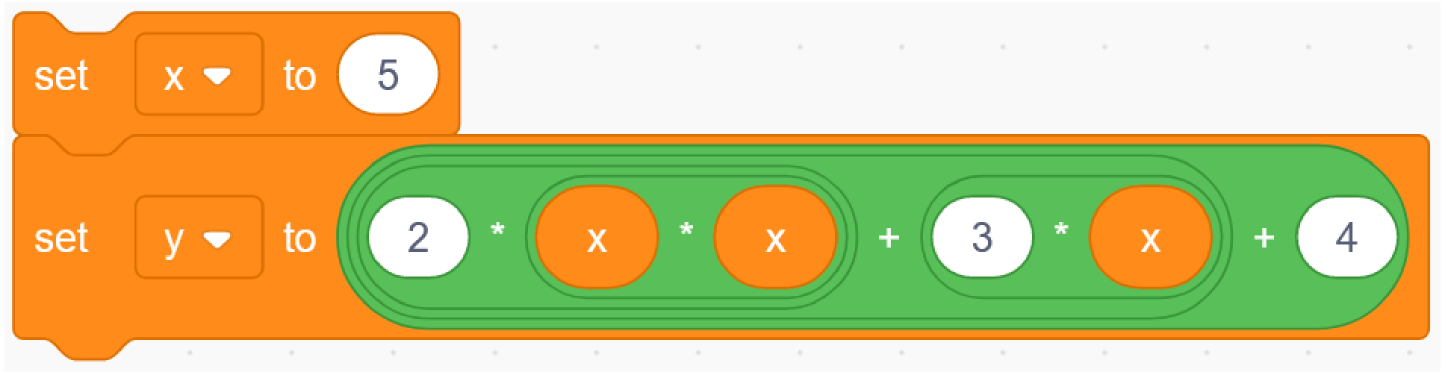
Basics: Calculations

Performing calculations was a key use of the early computers and many programs today include calculations. Computers do it well and fast.



```
set Hours to 15
say join Pay this week: Hours * 10.10
```

```
Hours = 15
Pay = Hours * 10.10
print('Pay this week:', Pay)
```



```
set x to 5
set y to 2 * x * x + 3 * x + 4
```

```
x = 5
y = (2 * (x ** 2)) + (3 * x) + 4
print(x, ',', y, sep='')
y = 2 * x ** 2 + 3 * x + 4
print(x, ',', y, sep='')
```

Find out how the language you are using does order of operation and the operators used.

Calculations: Imaginary Numbers

using to solve real problems

Python examples below with complex numbers to get you started

```
# create a complex number using j for imaginary part or the complex function
cplex1 = 1 + 2j
cplex2 = complex(3,4)

print('Real Part is', cplex2.real, 'Imaginary Part is', cplex2.imag)

# Add, subtract, multiply, divide are built in
print('Multiplication result: ', cplex1 * cplex2)

# Use cmath module for more advanced functions (import once at top of code)
import cmath
print('square root of cplex1:', cmath.sqrt(cplex1))
```

Many real world uses for imaginary/complex numbers.
Find people/resources solving those problems.

Calculations: Matrices

Python example below with matrices to get you started

```
import numpy as np

# testing with python 3.6 and numpy 1.17.3
# create two matrices, multiply them, and display the result
matrix1 = np.array([[2, 1, 4], [0, 1, 1]])
matrix2 = np.array([[6, 3, -1, 0], [1, 1, 0, 4], [-2, 5, 0, 2]])

print(matrix1)
print(matrix2)

matrix3 = matrix1.dot(matrix2)
print(matrix3)

matrix4 = matrix1 @ matrix2
print(matrix4)

print('this one is not correct')
print (np.multiply(matrix1,matrix2))
```

Basics: Conditions and Logic

Most programs make one or more decisions. The actions and flow of the program is determined by the check being true of false.

Logic

NOT	T	F
	F	T

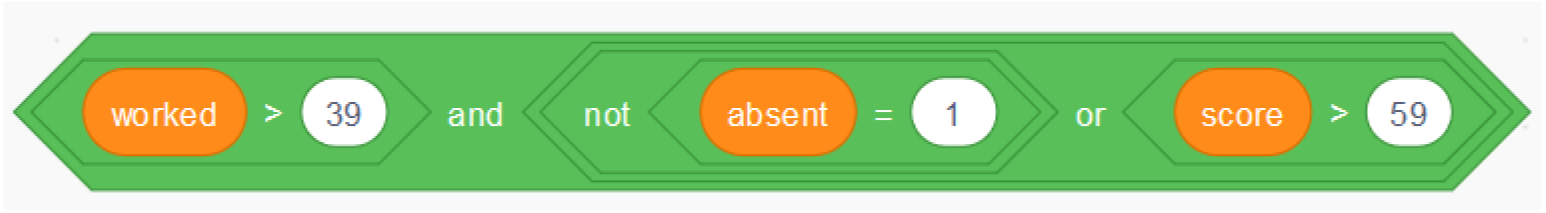
AND	T	F
T	T	F
F	F	F

OR	T	F
T	T	T
F	T	F

Truth tables helpful when learning logic. Two versions provided.
Read … NOT true is false
Read … true AND false is false
Read … true OR false is true

Some languages have logical operators besides NOT, AND, OR

```
absent = False
score = 59
worked = 45
```



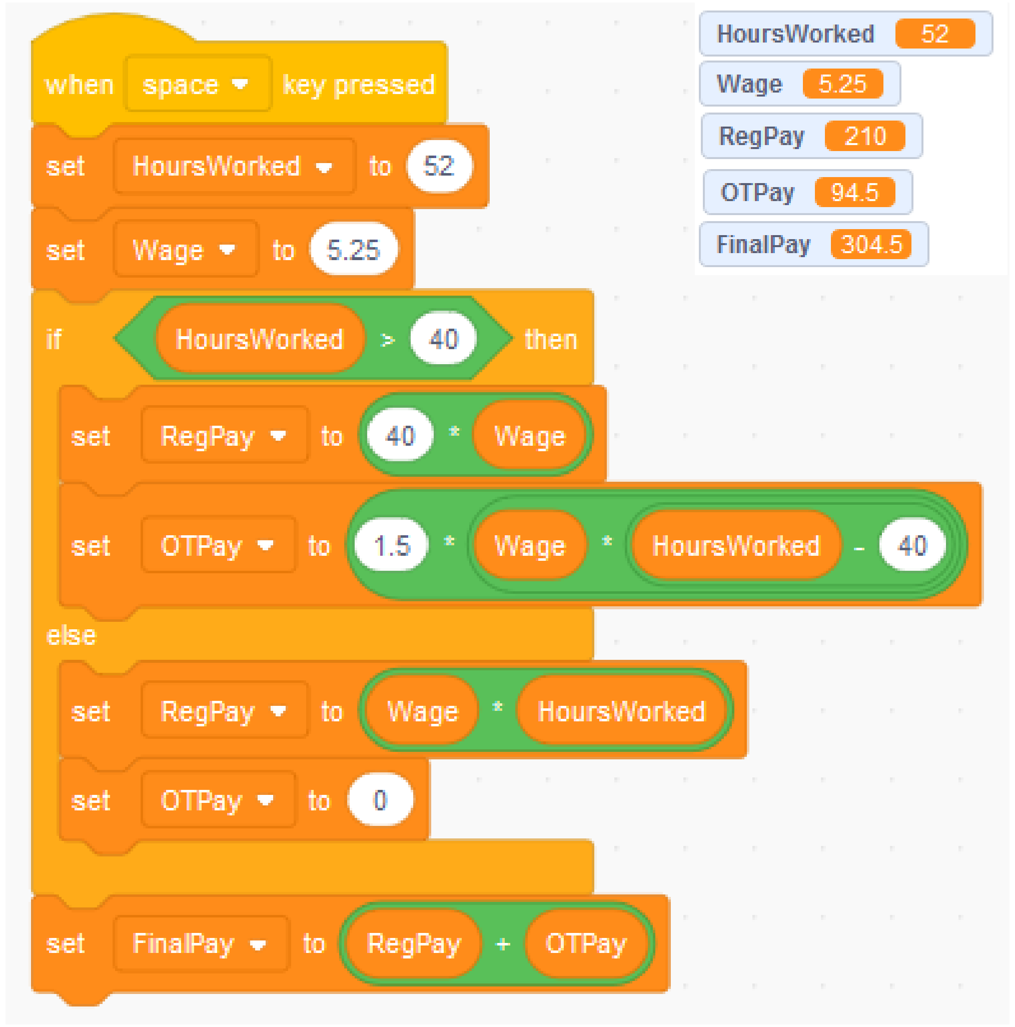
```
worked > 39 and not absent = 1 or score > 59
```

p	q	NOT p	p AND q	p OR q
T	T	F	T	T
T	F	F	F	T
F	T	T	F	T
F	F	T	F	F

```
(worked > 39) and ((not absent) or (score > 59))
(  45   > 39) and ((not F     ) or (   59 > 59))
      T       and ((not F     ) or          F   )
      T       and (     T        or          F   )
      T       and                T
               T
```

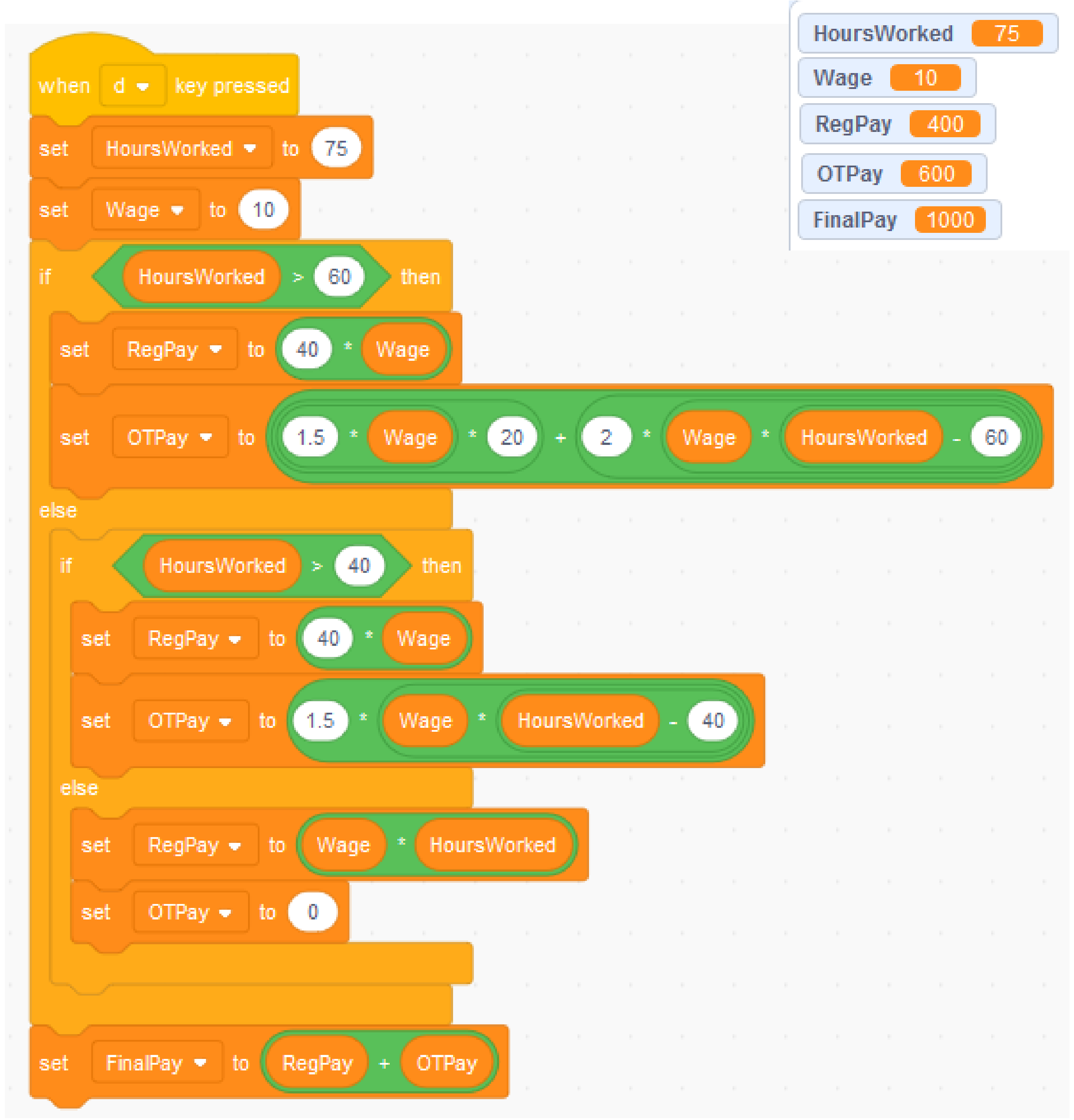
Note: Logical operators follow the coding language order of operation rules. Use parenthesis to be explicit.

If – Scratch Example



```
when space key pressed
set HoursWorked to 52
set Wage to 5.25
if HoursWorked > 40 then
    set RegPay to 40 * Wage
    set OTPay to 1.5 * Wage * HoursWorked - 40
else
    set RegPay to Wage * HoursWorked
    set OTPay to 0
set FinalPay to RegPay + OTPay
```

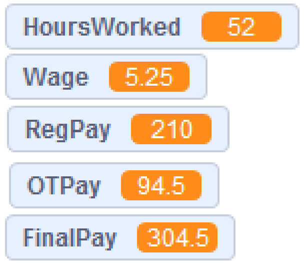
```
HoursWorked 52
Wage 5.25
RegPay 210
OTPay 94.5
FinalPay 304.5
```



```
when d key pressed
set HoursWorked to 75
set Wage to 10
if HoursWorked > 60 then
    set RegPay to 40 * Wage
    set OTPay to 1.5 * Wage * 20 + 2 * Wage * HoursWorked - 60
else
    if HoursWorked > 40 then
        set RegPay to 40 * Wage
        set OTPay to 1.5 * Wage * HoursWorked - 40
    else
        set RegPay to Wage * HoursWorked
        set OTPay to 0
set FinalPay to RegPay + OTPay
```

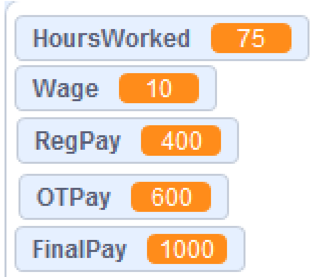
```
HoursWorked 75
Wage 10
RegPay 400
OTPay 600
FinalPay 1000
```

If – Python Example



HoursWorked	52
Wage	5.25
RegPay	210
OTPay	94.5
FinalPay	304.5

```
hours = 52
wage = 5.25

if hours > 40:
    regpay = 40 * wage
    otpay = 1.5 * wage * (hours - 40)
else:
    regpay = hours * wage
    otpay = 0

finalpay = regpay + otpay

print('Regular Pay:', regpay)
print('Overtime Pay:', otpay)
print('Final Pay:', finalpay)
```



HoursWorked	75
Wage	10
RegPay	400
OTPay	600
FinalPay	1000

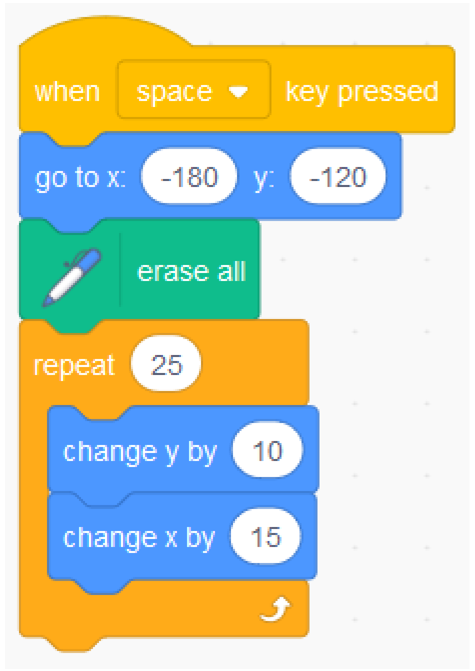
```
hours = 75
wage = 10

if hours > 60:
    regpay = 40 * wage
    otpay = (1.5 * wage * 20)
    otpay = otpay + (2 * wage * (hours - 60))
else:
    if hours > 40:
        regpay = 40 * wage
        otpay = 1.5 * wage * (hours - 40)
    else:
        regpay = hours * wage
        otpay = 0

finalpay = regpay + otpay

print('Regular Pay:', regpay)
print('Overtime Pay:', otpay)
print('Final Pay:', finalpay)
```

Note: otpay calculation done in two lines to fit easier onto this page

Basics: Repetition

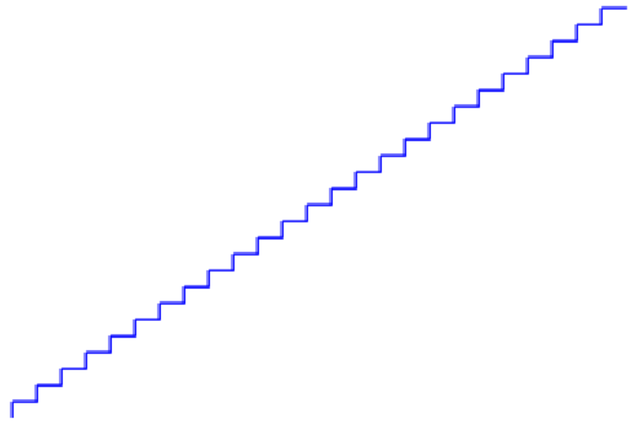
Repetition in coding is usually called loops or looping. The two primary types of loops are looping a determined number of types or looping until a condition is met.

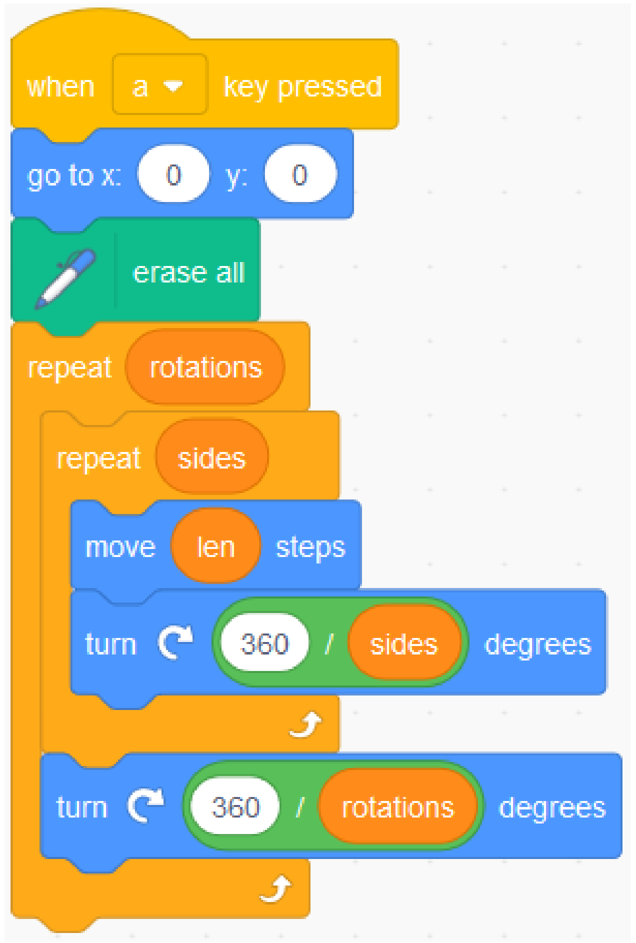
Looping / Repetition - For



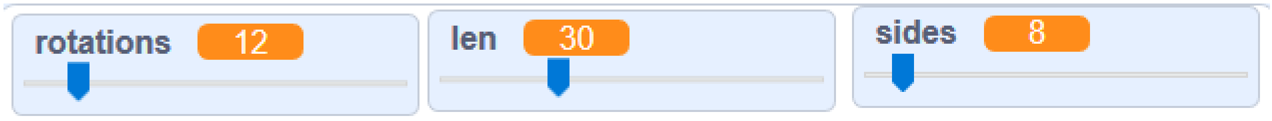
```
when [space] key pressed
go to x: -180 y: -120
erase all
repeat 25
    change y by 10
    change x by 15
```

- Simple, single loop
- Hardcode numbers
- Stair step result
- Change values for y (10) and x (15)
- Be sure pen down



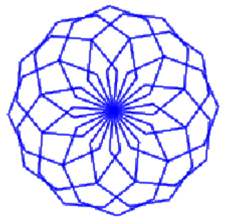


```
when [a] key pressed
go to x: 0 y: 0
erase all
repeat (rotations)
    repeat (sides)
        move (len) steps
        turn ↻ (360 / sides) degrees
    turn ↻ (360 / rotations) degrees
```

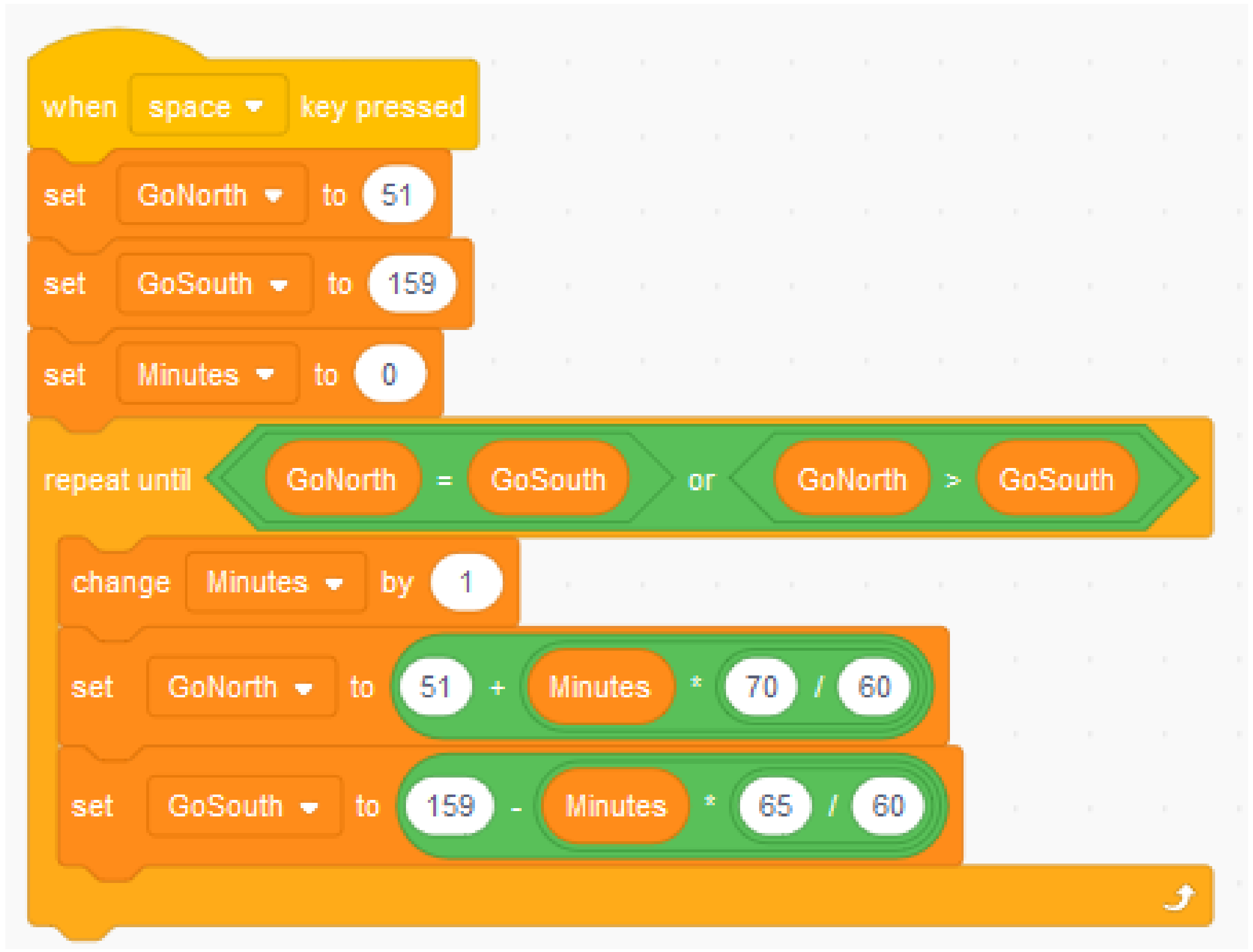


rotations 12 | len 30 | sides 8

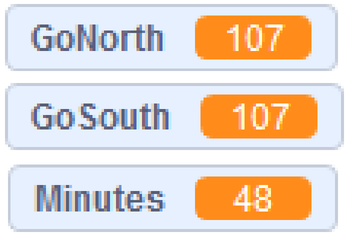
- Loop within loop
- Flexible, change variable values
- Angles calculated based on sides or rotations (variables)
- Go through math on item above
- Be sure pen down



Looping / Repetition - conditional



```
when [space] key pressed
set [GoNorth] to (51)
set [GoSouth] to (159)
set [Minutes] to (0)
repeat until <<(GoNorth) = (GoSouth)> or <(GoNorth) > (GoSouth)>>
    change [Minutes] by (1)
    set [GoNorth] to ((51) + ((Minutes) * ((70) / (60))))
    set [GoSouth] to ((159) - ((Minutes) * ((65) / (60))))
```



GoNorth	107
GoSouth	107
Minutes	48

Looping / Repetition - conditional

Based on true story with some alterations: My son forgot his retainers. Rather than one person make a long drive, each of us would leave at the same time and drive until we met. He started at mile marker 51 and drove north, I started at mile marker 159 and drove south. He went 70 mph, I went 65 mph.
How many minutes did we drive until we met? What mile marker did we meet at?

<u>What if? How Does This Change?</u>

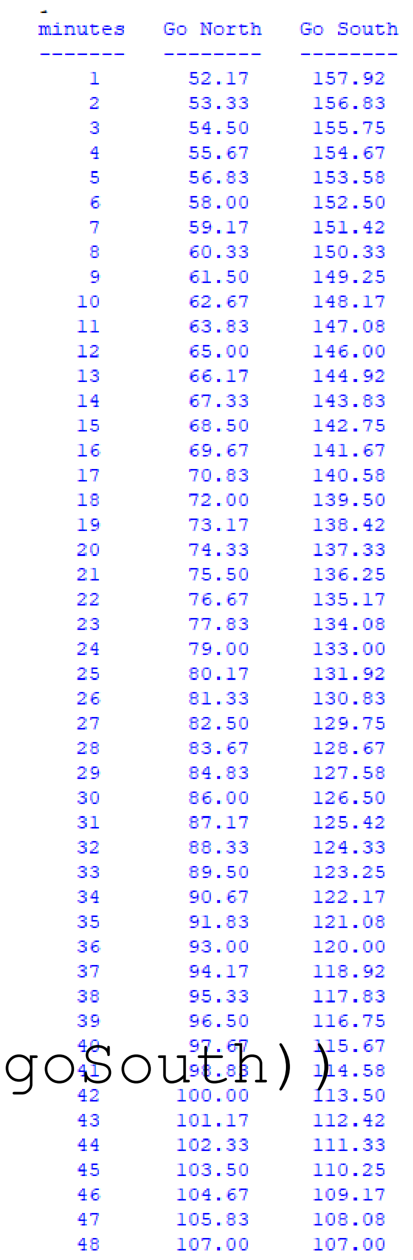
- Son drives 90 mph, I drive 85 mph?
- 100/45 mph? Other mph, look at trend in changes.
- I start 5 minutes before him. Alter values.
- Mile 140 to 135, I drive 35 mph (construction).
- Find in quarter, tenths, 5 hundreds of a minute.

<u>Discussion/Learning Points</u>

- Estimate the answer
- Calc mph
- Map and mile markers on I-75
- Value of a model like this
- How much time do you save speeding?
- What would be other good variables to use?

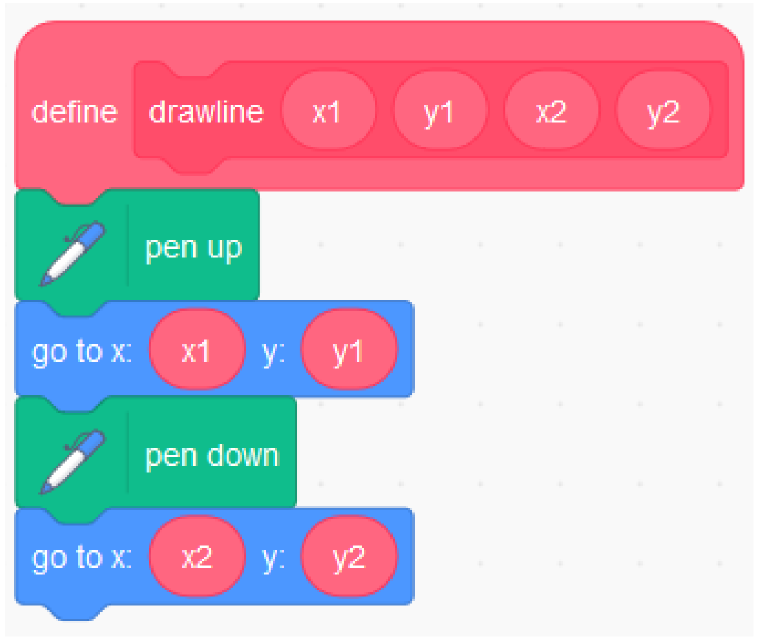
Looping / Repetition - conditional

```
minutes = 0
goSouth = 159
goNorth = 51

print('minutes', '  Go North', '  Go South')
print('-------', '  --------', '  --------')

while goSouth > goNorth:
    minutes = minutes + 1
    goSouth = 159 - (minutes * (65 / 60))
    goNorth = 51 + (minutes * (70 / 60))

    print("%5d %11.2f %10.2f" % (minutes, goNorth, goSouth))
```

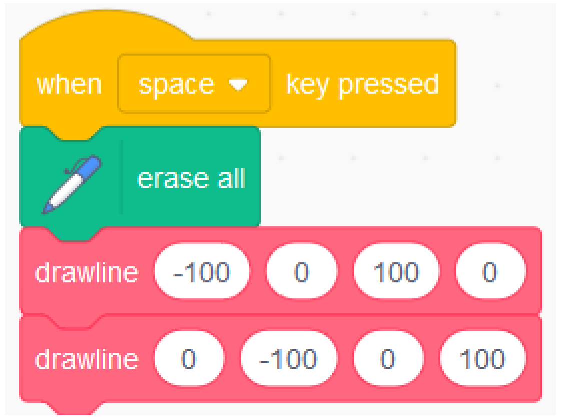


```
minutes   Go North   Go South
-------   --------   --------
    1       52.17     157.92
    2       53.33     156.83
    3       54.50     155.75
    4       55.67     154.67
    5       56.83     153.58
    6       58.00     152.50
    7       59.17     151.42
    8       60.33     150.33
    9       61.50     149.25
   10       62.67     148.17
   11       63.83     147.08
   12       65.00     146.00
   13       66.17     144.92
   14       67.33     143.83
   15       68.50     142.75
   16       69.67     141.67
   17       70.83     140.58
   18       72.00     139.50
   19       73.17     138.42
   20       74.33     137.33
   21       75.50     136.25
   22       76.67     135.17
   23       77.83     134.08
   24       79.00     133.00
   25       80.17     131.92
   26       81.33     130.83
   27       82.50     129.75
   28       83.67     128.67
   29       84.83     127.58
   30       86.00     126.50
   31       87.17     125.42
   32       88.33     124.33
   33       89.50     123.25
   34       90.67     122.17
   35       91.83     121.08
   36       93.00     120.00
   37       94.17     118.92
   38       95.33     117.83
   39       96.50     116.75
   40       97.67     115.67
   41       98.83     114.58
   42      100.00     113.50
   43      101.17     112.42
   44      102.33     111.33
   45      103.50     110.25
   46      104.67     109.17
   47      105.83     108.08
   48      107.00     107.00
```

Blocks and Functions



```
define drawline (x1) (y1) (x2) (y2)
pen up
go to x: (x1) y: (y1)
pen down
go to x: (x2) y: (y2)
```

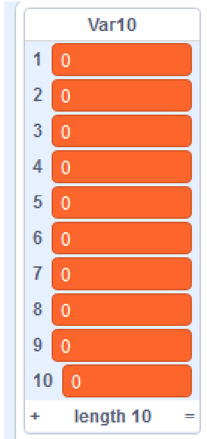


```
when [space ▾] key pressed
erase all
drawline (-100) (0) (100) (0)
drawline (0) (-100) (0) (100)
```

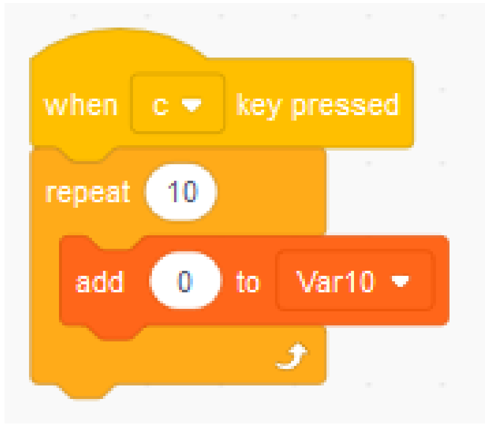
```
import turtle

def line(x1, y1, x2, y2):
    turtle.penup()
    turtle.goto(x1, y1)
    turtle.pendown()
    turtle.goto(x2, y2)

def fx(x):
    result = 1 / 2 * x + 40
    return result

turtle.color("red")
line(-100, 0, 100, 0)
line(0, -100, 0, 100)

turtle.penup()
turtle.color("black")
for x in range(-100, 100, 1):
    y = fx(x)
    turtle.goto(x, y)
    turtle.dot()
```

Array Example in Scratch

Array is a collection of variables that can be manipulated programmatically.



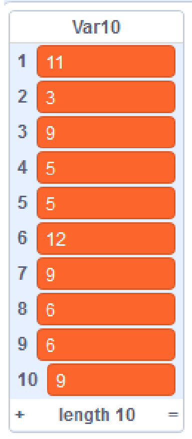
Var10	
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
+ length 10 =	



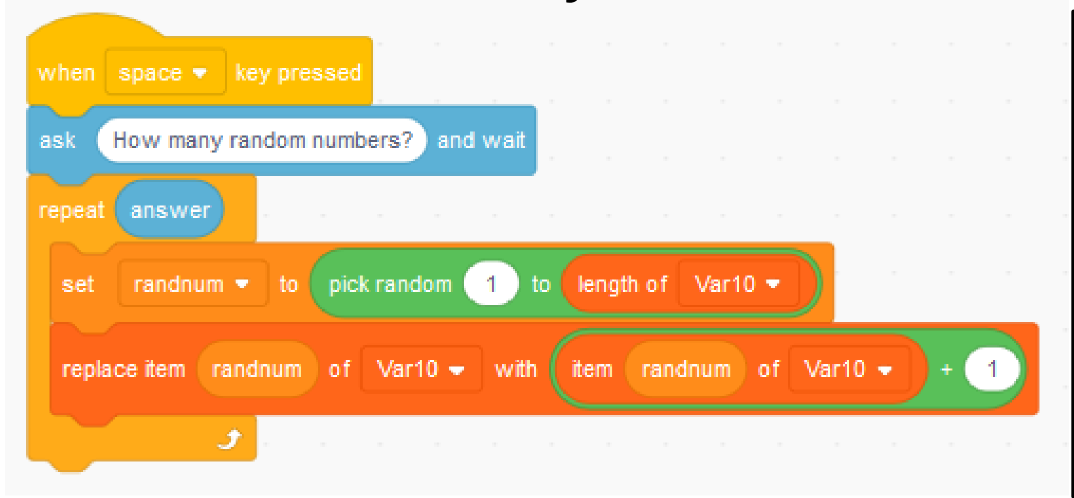
```
when c key pressed
repeat 10
    add 0 to Var10
```

- In Variables, click on "Make a List" and give it a name.
- This code adds 10 elements to the empty list, each with the value of 0. Now we have 10 variables with a 0 in each.

Ask how many random numbers to generate then generate that many random numbers (each random number is between 1 and the length of Var10 (which is 10). Each box stores how many of that box numbers random numbers was generated.



Var10	
1	11
2	3
3	9
4	5
5	5
6	12
7	9
8	6
9	6
10	9
+ length 10 =	



```
when space key pressed
ask How many random numbers? and wait
repeat answer
    set randnum to pick random 1 to length of Var10
    replace item randnum of Var10 with item randnum of Var10 + 1
```

- Pick random number between 1 and the length of Var10 (is 10).
- "item randnum of Var10" pulls current value in the element at position randnum then adds 1 to it and puts it back in that element (randnum)

… Miscellaneous …

Mathematics Pathways Pilots

- Alternatives to (or in addition to) Algebra II
- Two courses with mathematics and coding
  - Discrete Mathematics/Computer Science
  - Data Science Foundations
- Teacher as facilitator
- Solid and rigorous curriculum
- Teachers participate in training sessions and are provided support

https://education.ohio.gov/Topics/Learning-in-Ohio/Mathematics/Resources-for-Mathematics/Math-Pathways

Real World Projects

Examples

- Quality Assurance (QA) for a manufacturing company
- Research project
- Analyze survey/data
- Layout/design of new space

Invite local expert to help you and your students manage and learn the math for the project.

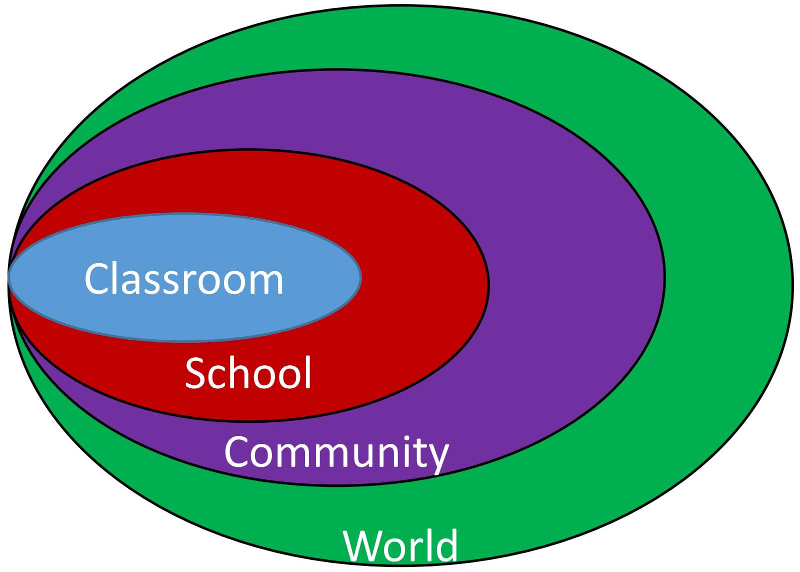
Work with …

- local college/university
- local government
- local non-profit
- local business

What math do your students know or can learn to work on the project?

Actual, real project or a simulated project.

Real World Project Focus and Scope of Impact



- Start with projects that stay in the classroom
- When students skills are ready, move into projects for those in the building or school district
- Community projects require more work and communication. Start with people or organizations that you know or who have worked with the school.
- World projects can be facilitated with the Internet. Develop a class web presence and use that as a primary communication tool.

From my presentation … Real World Projects in the K-12 Computer Classroom

Math and Coding Across Curriculum Areas

Art – digital art creation (spinning geometric shapes, patterns, trig, fractals, etc). Where does math come in and where is the artistic touch. Golden ratio in the Mona Lisa and other places. Pattern vs free form and how to blend the two.

Science – computations, data processing, sensors/data, physics/gaming,

BioTech/Science Tech – intersection of science, math, code.

Music – Math behind music (and also science)

English / foreign language … natural language processing/structures.

Partner with other curriculum areas

AI and Coding
Can’t I have the computer write it for me?

AI:coding::calculators:math

Substitute other math technology aids for calculators … such as spreadsheets, Desmos, PhotoMath, etc.

When should students do their math without a calculator and why?
When should students do their math with a calculator and why?

Likewise, there are times when students need to code without AI assistance and other times where AI is a beneficial learning experience and tool.

Students should always be able to explain their code (ask them on occasion), even when borrowing code.

Ask a foreign language teacher how they know when a student uses a translator.

… Why/How/When …

Coding is a tool many students will use again in some form

modeling / simulations

Explore a topic too time consuming to explore by hand or with other tools.

Fun!

automate the repetitive

why code?

Students learn by teaching/instructing a concept … to a computer. They must learn/understand the topic well. Must be precise when coding, get immediate feedback.

visualization of patterns, results, data, etc

let computer do what it does best, let humans do what they do best

Enrichment – finished early, need a challenge

Integrated – coding activities as part of the curriculum

Research project – partner with community, company, college needing some research

Remediation – trouble learning or visualizing a concept

Incorporate coding … how?

Hour of Code event --- https://hourofcode.com/ for more information.

coding club – before/after school, during activity period (student led?)

get some help from others

Science fair – structured approach to exploring and application of mathematics and coding

See previous session Presentation Notes page 2 for more explanation of these items

when?

"Free day"

Guest speaker/career day

PBIS or Activity Day with activity options

Catch up day – activity for those already "caught up"

Before vacation (alternative to watching a movie)

Math/coding club during building activity period

STEM/Technology time

Problem of the day/week at the start of class.

… Resources …

A small collection of free (at least some part is free) resources.

You can also use a search engine and browse the various free video sites for additional free content.

There are a number of pay resources such as books, on-line courses, curriculum materials, etc. that are beyond the scope of this section and presentation.

CS and Math

* Scratch based coding activites

* Free Lessons link organized by K-5, 6-8, 9-12

* Workshop and interactive options (cost involved)

* Activities have lesson plans that can include standards, videos, teacher guide, and student guide.

https://csandmath.org/

<code_by_math*/> Website

Lessons – math and coding lessons.

Teachers – track your student progress.

Challenges – short math coding challenges.

Lots of good math information and application of coding to solve math problems.

Coding in the Lua language.
Can also try lesson and challenges in other programming languages.

<code_by_math*/>

http://www.codebymath.com/

Al Sweigart Books – No Starch Press

* Python is the language used in most books, there is a Scratch book also.

* All books available to read for free on-line at the link below. Can also purchase print and e-book versions.

Books and Highlights …

The Recursive Book of Recursion – practical coverage of recursion and alternatives

Automate the Boring Stuff with Python – Python fundamentals, Excel and Google Sheets manipulation

Cracking Codes with Python - Cryptography and ciphers

… others …

https://inventwithpython.com/

code.org … HourOfCode.com

code.org
Learn – Coding curriculum and activities

Teach – Teacher information and materials

HourOfCode.com
Clicking on Activities, under Topics click on Math. Apply any other filters.

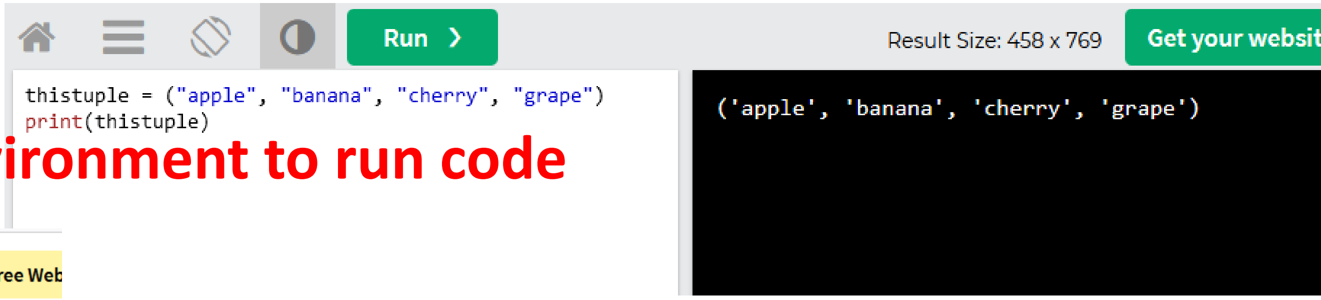
https://code.org/ … https://hourofcode.com/

w3schools.com

Free, advertising pays for the site

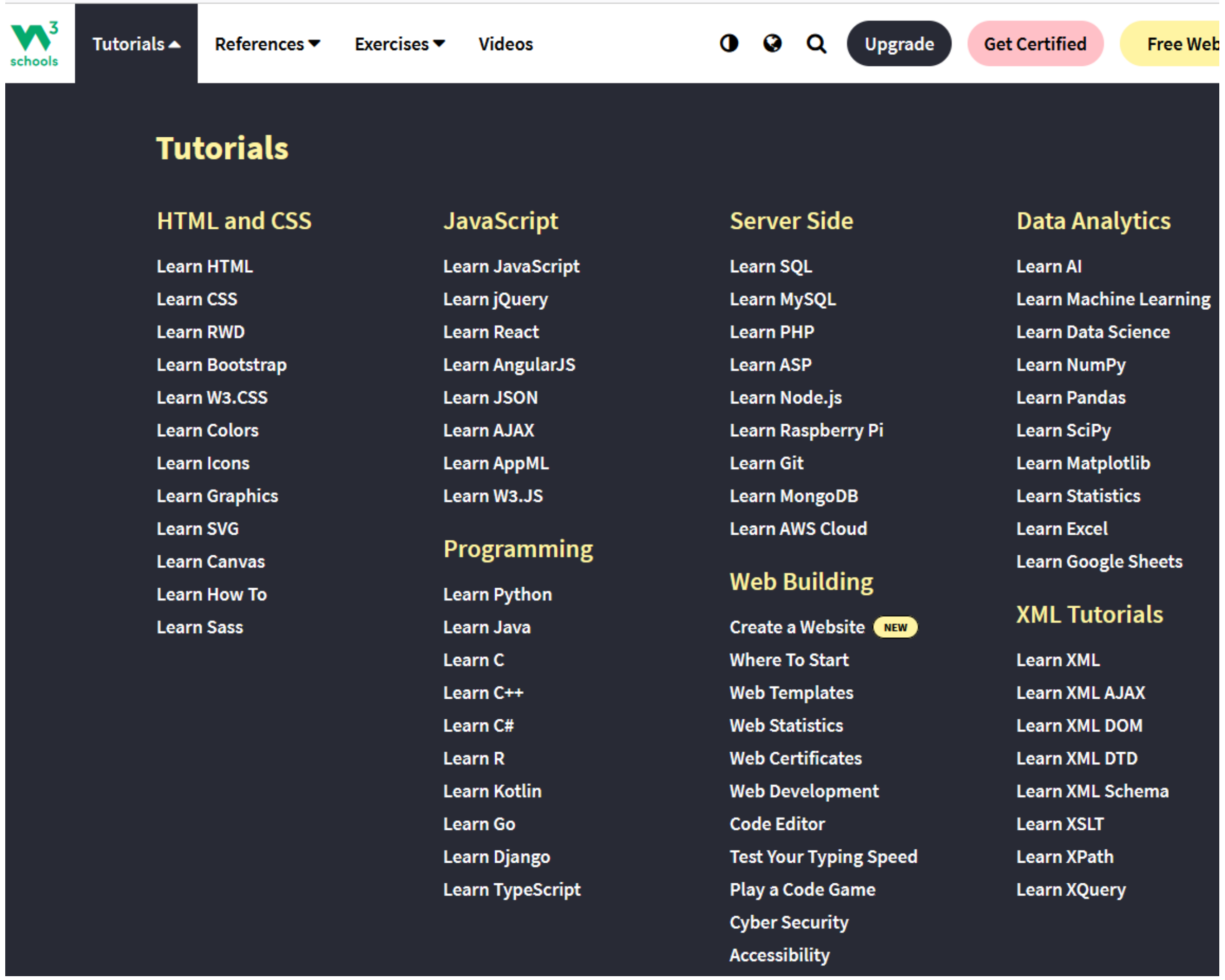
Tutorials available

On-line environment to run code



```
thistuple = ("apple", "banana", "cherry", "grape")
print(thistuple)
```

```
('apple', 'banana', 'cherry', 'grape')
```



Tutorials

HTML and CSS
- Learn HTML
- Learn CSS
- Learn RWD
- Learn Bootstrap
- Learn W3.CSS
- Learn Colors
- Learn Icons
- Learn Graphics
- Learn SVG
- Learn Canvas
- Learn How To
- Learn Sass

JavaScript
- Learn JavaScript
- Learn jQuery
- Learn React
- Learn AngularJS
- Learn JSON
- Learn AJAX
- Learn AppML
- Learn W3.JS

Programming
- Learn Python
- Learn Java
- Learn C
- Learn C++
- Learn C#
- Learn R
- Learn Kotlin
- Learn Go
- Learn Django
- Learn TypeScript

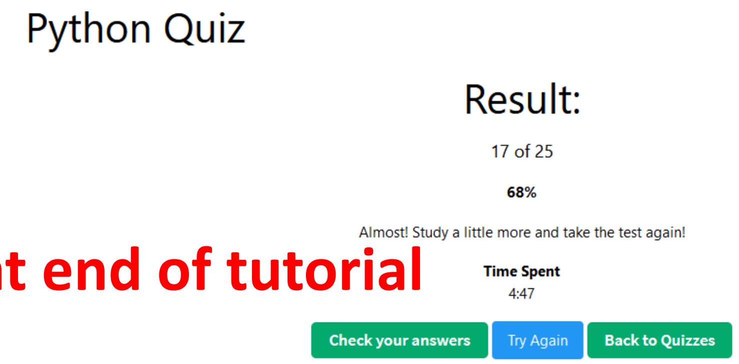
Server Side
- Learn SQL
- Learn MySQL
- Learn PHP
- Learn ASP
- Learn Node.js
- Learn Raspberry Pi
- Learn Git
- Learn MongoDB
- Learn AWS Cloud

Web Building
- Create a Website NEW
- Where To Start
- Web Templates
- Web Statistics
- Web Certificates
- Web Development
- Code Editor
- Test Your Typing Speed
- Play a Code Game
- Cyber Security
- Accessibility

Data Analytics
- Learn AI
- Learn Machine Learning
- Learn Data Science
- Learn NumPy
- Learn Pandas
- Learn SciPy
- Learn Matplotlib
- Learn Statistics
- Learn Excel
- Learn Google Sheets

XML Tutorials
- Learn XML
- Learn XML AJAX
- Learn XML DOM
- Learn XML DTD
- Learn XML Schema
- Learn XSLT
- Learn XPath
- Learn XQuery

Good reference and tutorial site for a number of coding languages. No math specific items.



Python Quiz

Result:

17 of 25

68%

Almost! Study a little more and take the test again!

Time Spent

4:47

Short quizzes at end of tutorial

Q&A/Discussion/Wrap-Up/Moving Forward

- Questions for others in the session or me?
- Suggestions for the group?
- How will today's session positively impact your students?

What resources/tips from those doing this? What would be helpful (others/myself)?