math.code v2

You can incorporate coding into your math classroom

2019 OCTM Conference
October 23-25 in Sandusky, OH
Session 38 Thursday 10:00-10:50 am Crown Palm

Session Goals for Each Participant

Goal 1: Do one more coding activity with your students than you did last year.

Goal 2: Collect some ideas and formulate a strategy on how to incorporate coding in your classroom.

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.)

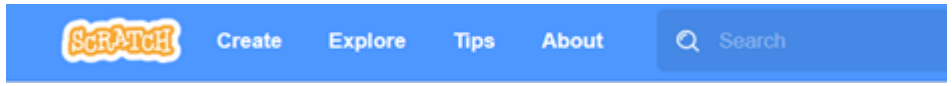
Thomas Edison Quotes

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

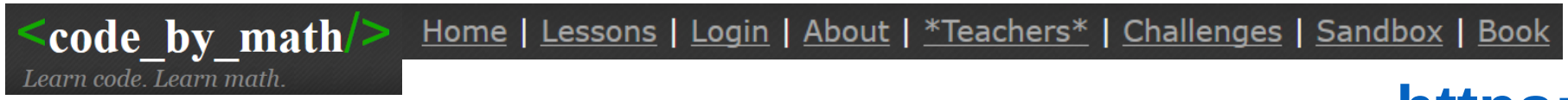
Our greatest weakness lies in giving up. The most certain way to succeed is always to try just one more time.

A Few Coding Resources/Options

https://studio.code.org



https://scratch.mit.edu



https://codebymath.com/

https://p5js.org

https://www.python.org

… and many others

<code_by_math />
Learn code. Learn math.

https://codebymath.com/

Variables in programming

Introduction
Calculate a tip
Distance formula with variables
Rescaling a result
Drawing circles with the mouse
Mathematical expressions with variables
Calculate a tip (with keyboard input)
Changing a variable in one line
Drawing lines with the mouse
The slope of a line

The for-loop: counting over a range of numbers

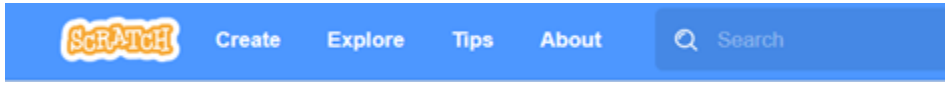
Using for-loops to count
Make math practice problems
Study the absolute value
Adding numbers to find Pi
Iterating a function
Iterate for a square-root
An even more clever for-loop for Pi
Diverging and converging series
More for-loop counting
Making math-facts tables
Adding numbers like Carl Gauss did
Finding square roots
Interating the logistic function
A clever for-loop for Pi
What kinds of numbers?
Numbers ending in 1

The for-loops: making graphs

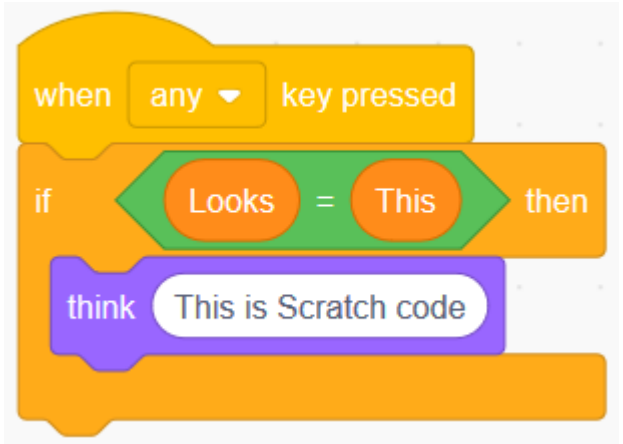
Make it snow
Graph the absolute value
Graph a function
Graph a line in slope-intercept form
Graph a parabola
The Lorenz Attractor

1. A series to find π
2. Show $\sin^2(x) + \cos^2(x) = 1$
3. Find the square root of a number.
4. Euler polynomial and primes
5. Sum square different
6. Add the digits of a number
7. Archimedean spiral
8. Recurrence relation
9. Fractions with square root
10. n integer to perfect square
11. Integer terms
12. n divisible by 3
13. Which is larger?
14. Sum equals product
15. x squared

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')
```

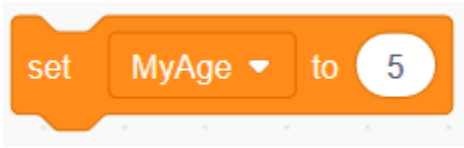
Text based examples done in Python 3.7 (or 3.6). While the syntax of text based coding differs by coding language, the basic constructs such as variables, calculations, decision, looping, functions, etc are similar.

Variables

Variable – a box that can hold only one value at any time.

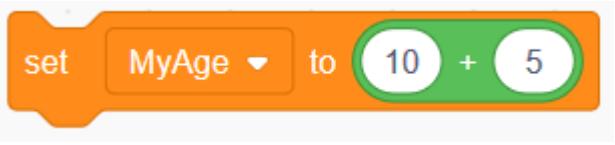
A new value placed into a variable replaces the old value.

= (equal sign) used to assign a value to a variable

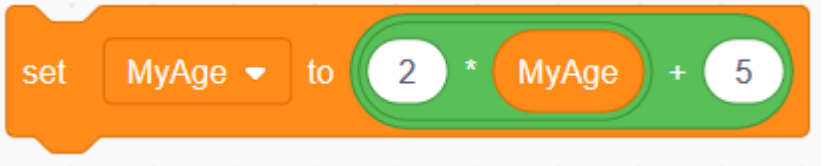



```
myage = 5
print(myage)
```

Visualize/Teach: Draw a box, calculate and put the result in the box. Do a second calculation, replace the contents of the box with the new result. Continue as needed.




```
myage = 10 + 5
```




```
myage = 2 * myage + 5
```

Variables are very important and a key component of coding.

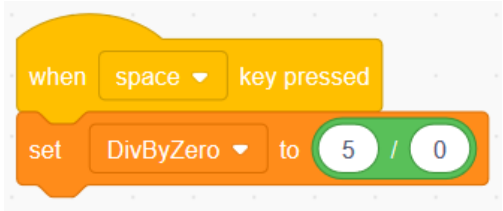
Calculations

From the early computers (like the ENIAC) to current computers, one of the primary tasks was/is to do calculations.

Research: What are the names of the some of the early computers and what type of calculations did some of the first computers do?

What are some of the more complex calculations computers do today?

What happens when you divide by zero? Varies by language (try it)

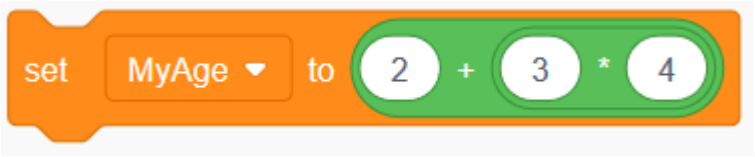



```
divbyzero = 5 / 0
print(divbyzero)
```

See the next slides and accompanying worksheets for activities your students can do to explore coding calculation solutions.

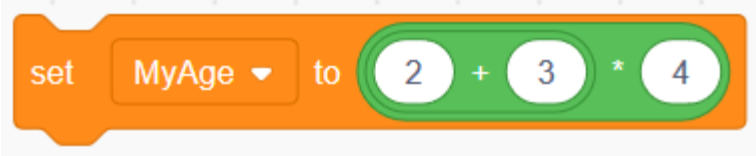
Calculations: Order of Operation

2 + 3 x 4 … is it 14 or 20?
Answer: 14
Multiply first (3 x 4) is 12
Add second (2 + 12) is 14




```
myage = 2 + 3 * 4
```

Real World: Use word problems and some scientific, physics, engineering, etc formulas to practice order of operation




```
myage = (2 + 3) * 4
```

Block based (Scratch) order determined by how the operators are stacked (top most first), text based code uses order of operation rules including parenthesis.

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.

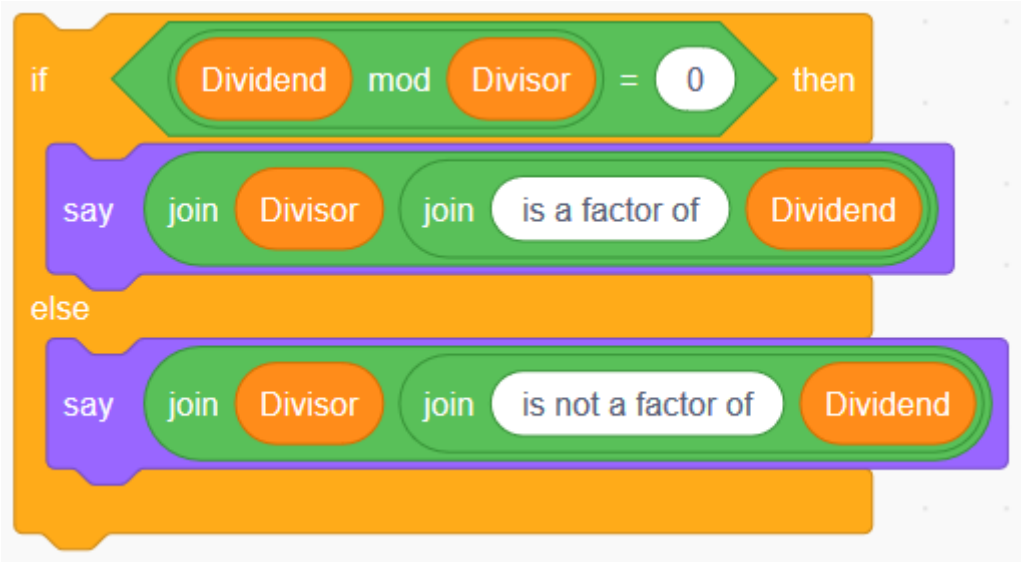
Calculations: Matrices

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))
```

Calculations: Modulo (mod)

mod is the remainder of a division (i.e. long) problem
Many interesting uses and applications of mod




```
if dividend % divisor == 0:
    print(divisor, 'is a factor of', dividend)
else:
    print(divisor, 'is not a factor of', dividend)
```

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

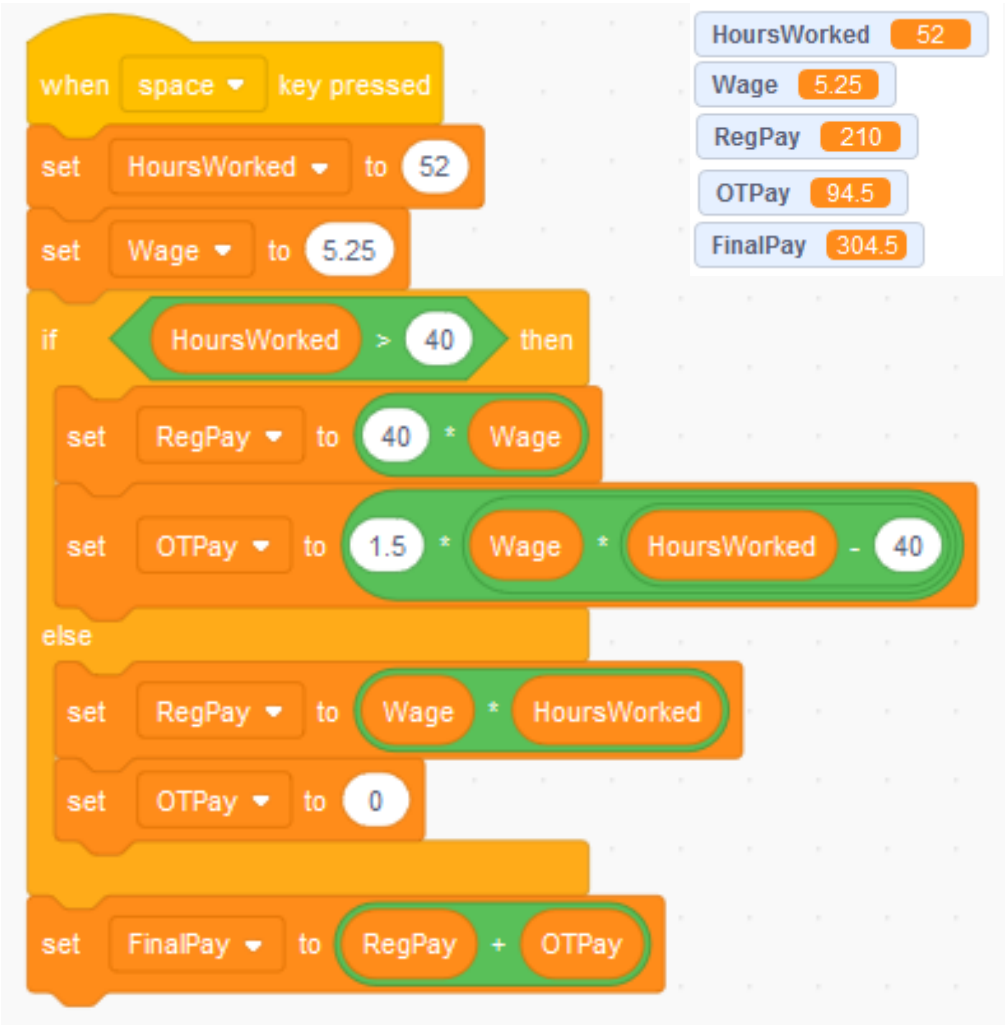
```
absent = False
score = 59
worked = 45
```

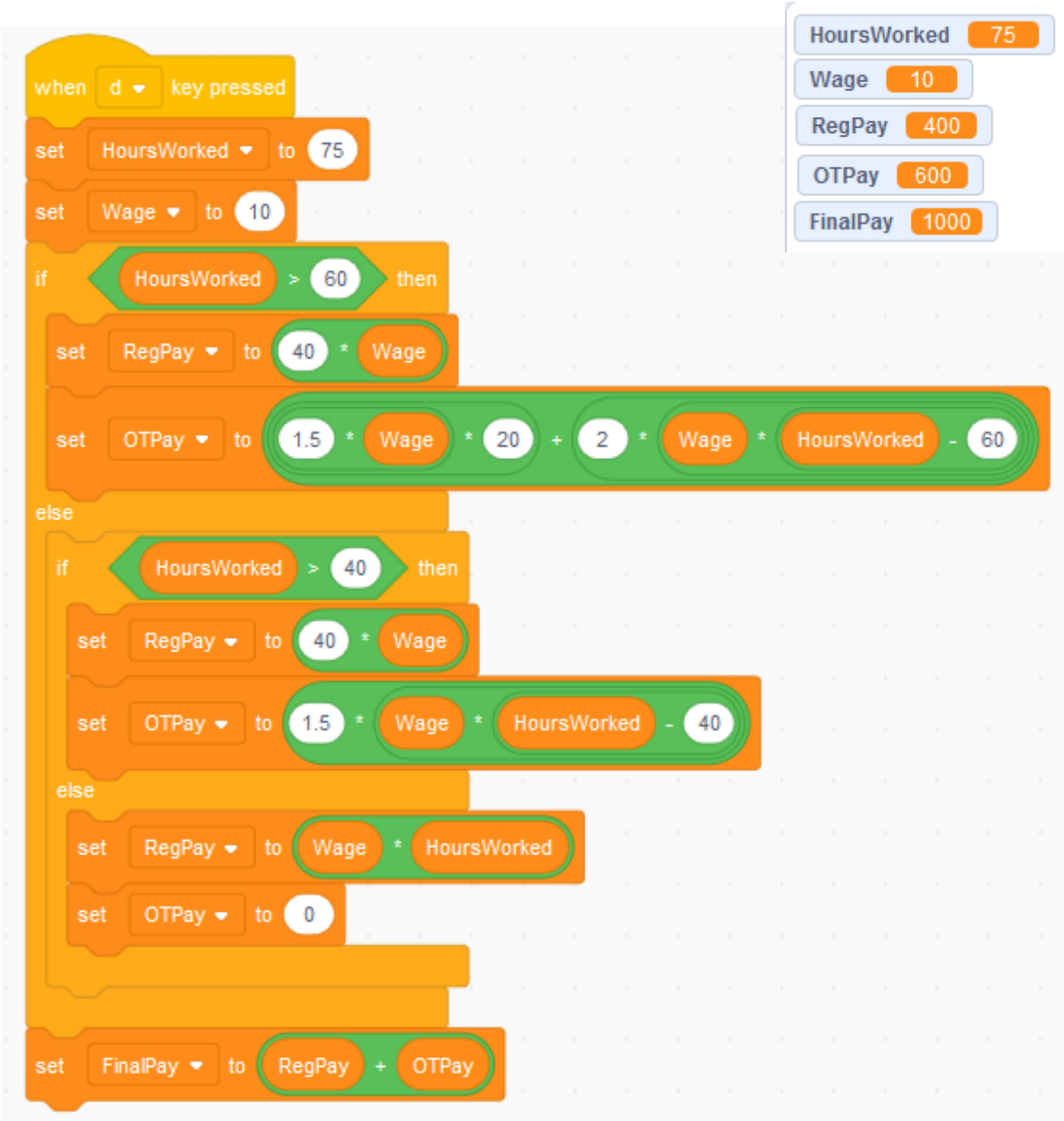
p	q	NOT p	p AND q	p OR q
T	T	F	T	T
T	F	F	F	T
F	T	T	F	T
F	F	T	F	F

```
(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.

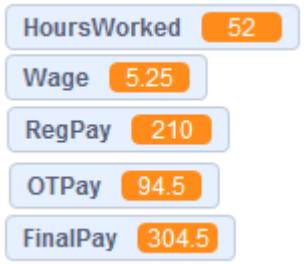
If – Scratch Example





If – Python Example

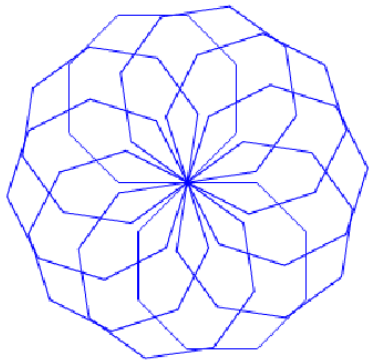
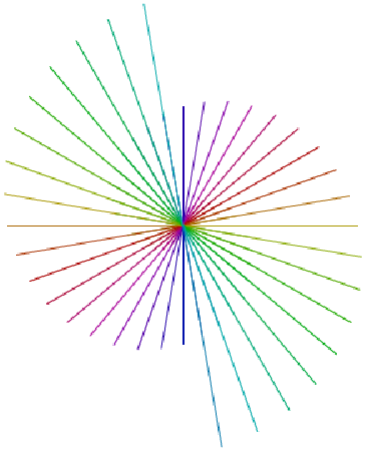
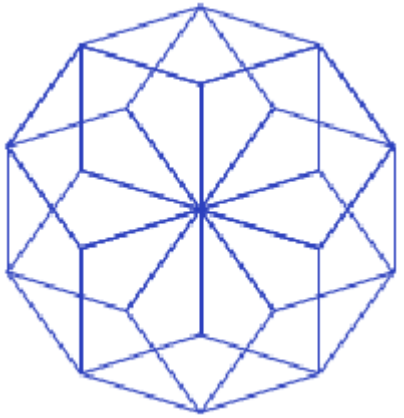
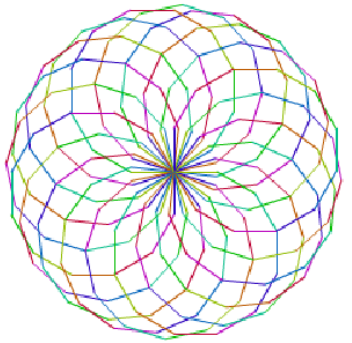
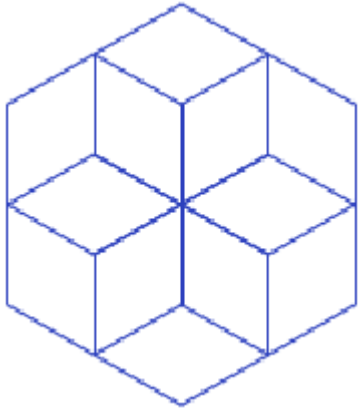
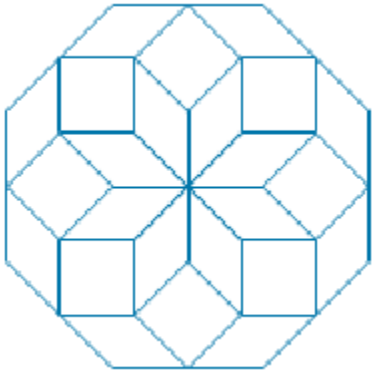
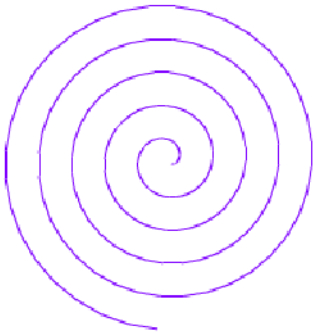
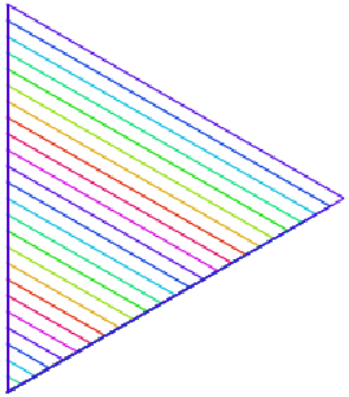
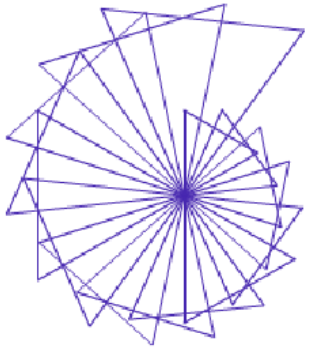
```
hours = 52
wage = 5.25

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)
```




```
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

Geometric Patterns

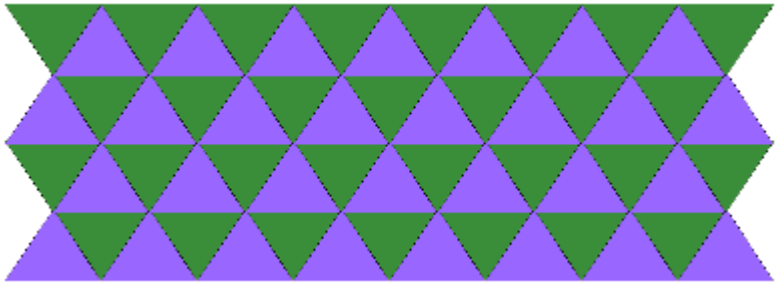
See Presentation Notes (last year) Item … Geometric Drawing, Shapes, and Patterns (Scratch)

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?

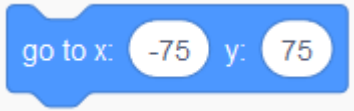


Do It:

- 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.

Graphics / Drawing – Scratch

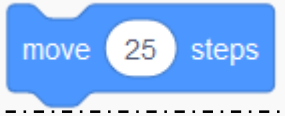
Lower left corner – click on “Add Extension” and add the Pen extension



Move to the specified location.

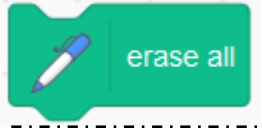
Move a specific x or specific y position

Move either the x or y position by the specified number of pixels

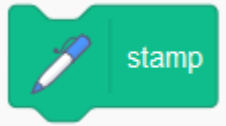


Go the specified number of pixels in the current direction.

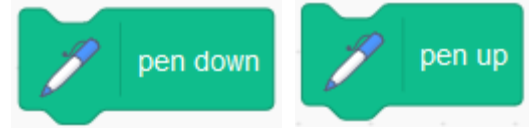
Turn right or left the specified number of degrees.



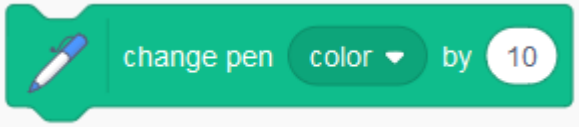
Erase all pen marks and stamps.



Stamp (leave a copy of) the current sprite at the current location.

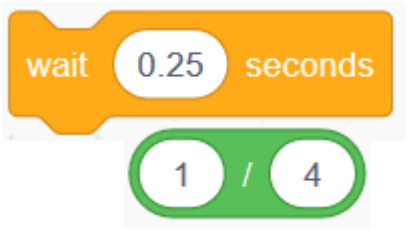


Pick up or put down the pen. Drawing or not drawing.

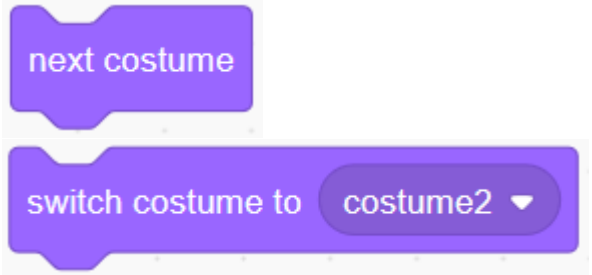


Alter the pen color

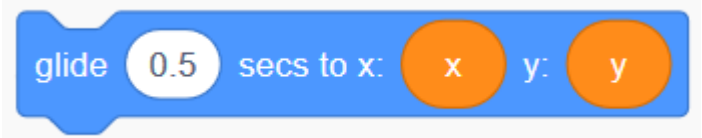
Animation – Scratch



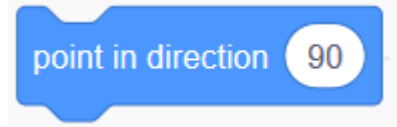
Pause between movement for effect. Can explore decimal/fractions and even use variables to progressively get faster/slower in the loop.



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.



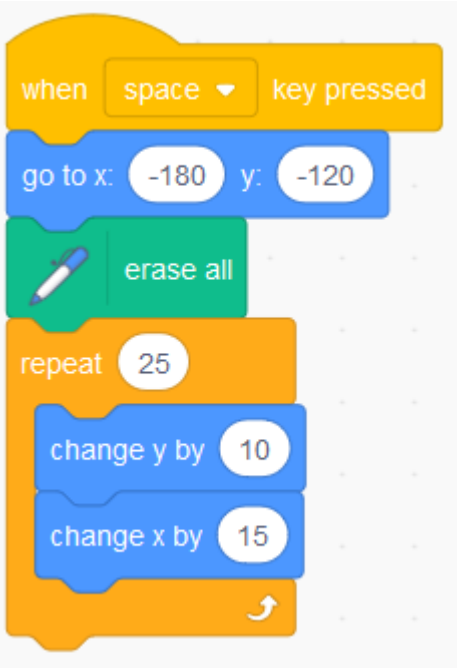
Move to the new location over the specified amount of time.



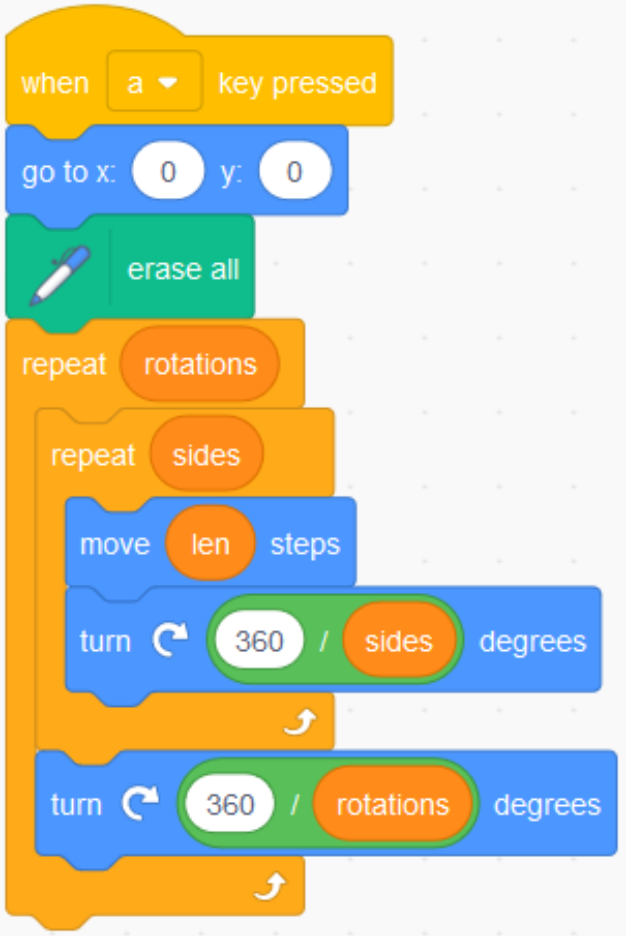
Point sprite in the desired direction. 90 is straight ahead. Click on the number for a degree dial. Positive or negative degrees.

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

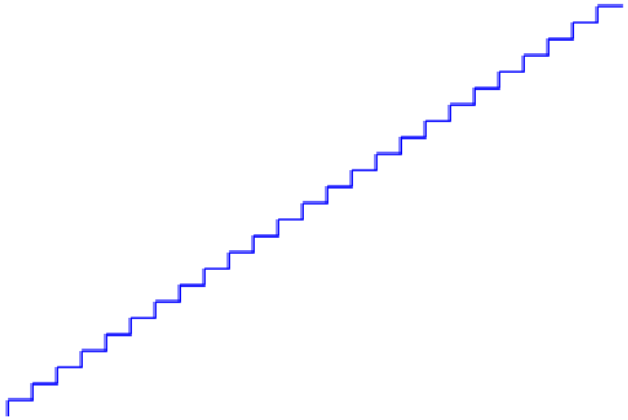
Looping / Repetition - For

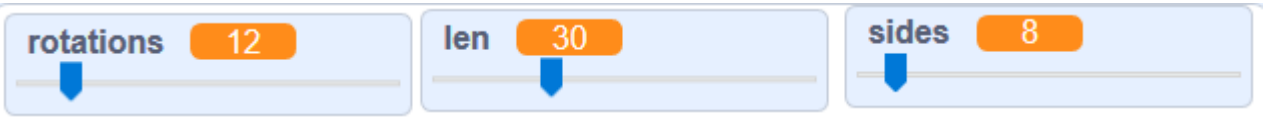


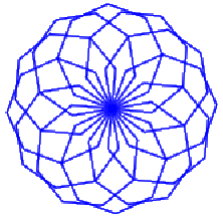
- Simple, single loop
- Hardcode numbers
- Stair step result
- Change values for y (10) and x (15)
- Be sure pen down



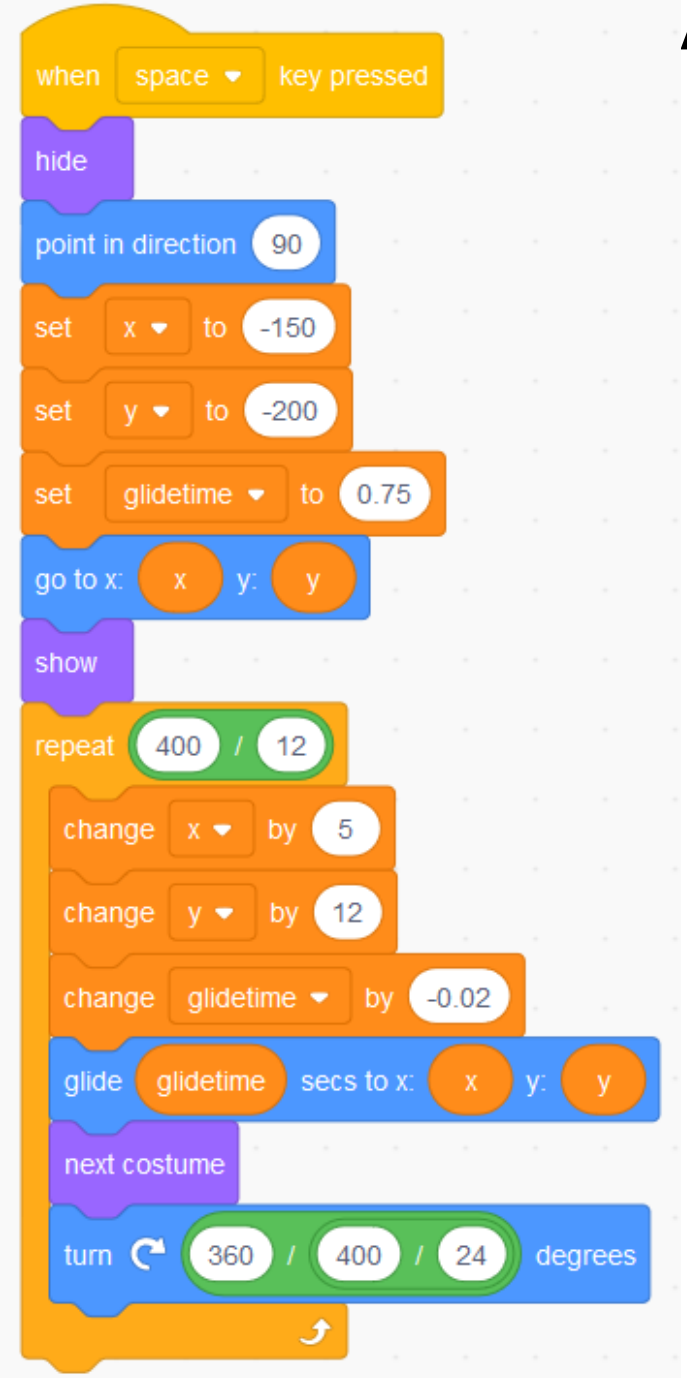
- Loop within loop
- Flexible, change variable values
- Angles calculated based on sides or rotations (variables)
- Go through math on item above
- Be sure pen down







Animation Illustration



For illustration purposes only …

- 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.

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?

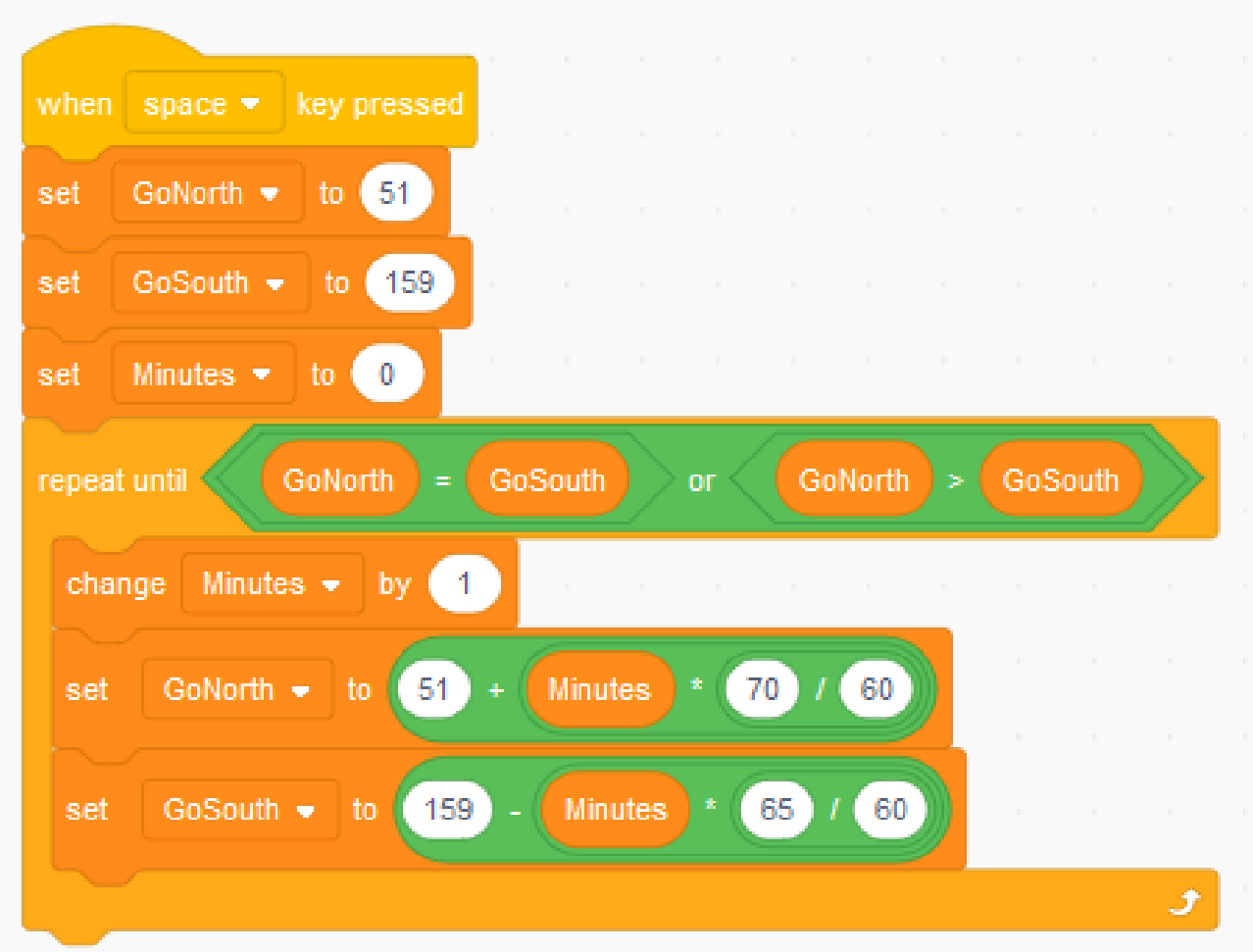
What if? How Does This Change?

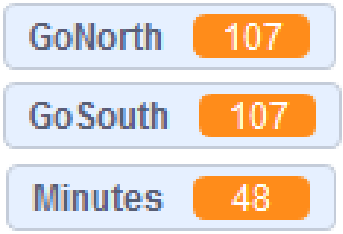
- Son drives 90 mph, I drive 85 mph?
- 100/45 mph? Other mph, look at trend in changes.
- I start 5 minutes before him. Alter values.
- Mile 140 to 135, I drive 35 mph (construction).
- Find in quarter, tenths, 5 hundreds of a minute.

Discussion/Learning Points

- 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





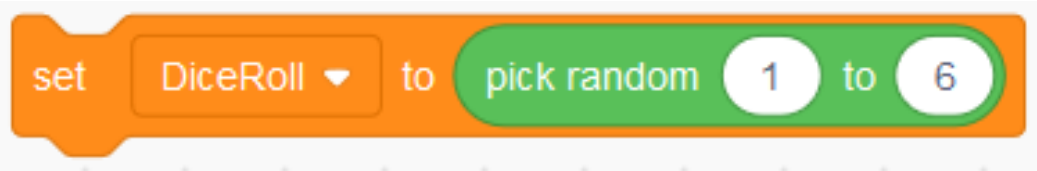
Looping / Repetition - conditional

```
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

Random Numbers

generating pseudo-random numbers



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

```
import random

random.seed()

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

- 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).

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.

Array Example in Scratch

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

- 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.

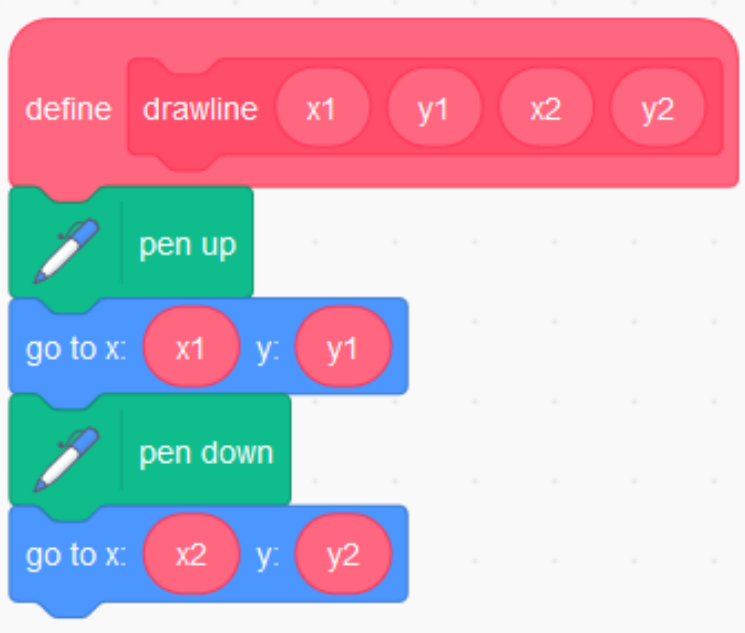
Ask how many random numbers to generate then generate that many random numbers (each random number is between 1 and the length of Var10 (which is 10). Each box stores how many of that box numbers random numbers was generated.

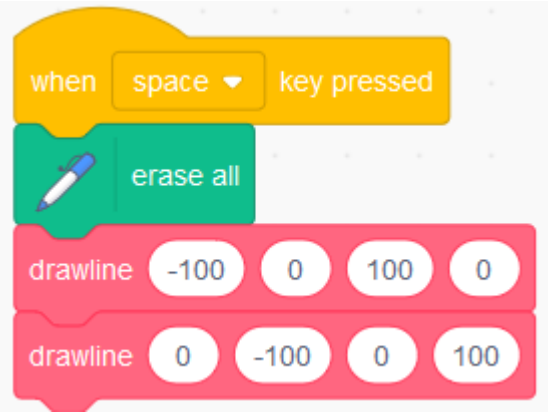
- Pick random number between 1 and the length of Var10 (is 10).
- “item randnum of Var10” pulls current value in the element at position randnum then adds 1 to it and puts it back in that element (randnum)

Array/List/Dictionary - Python

```
import random

# Create rollcount
rollcount = [0]

# Rolling the dice so need to store 1 through 6
for i in range(1,7):
  print(i)
  rollcount.append(0)
  print(rollcount[i])

print('-------------')


# Rolling the dice 100 times and will see how many of each numbers is rolled
for i in range(100):
  oneroll = random.randint(1,6)
  print(oneroll)
  rollcount[oneroll] = rollcount[oneroll] + 1

for i in range(1, 7):
  print(i, ' = ', rollcount[i])
```

Blocks and Functions





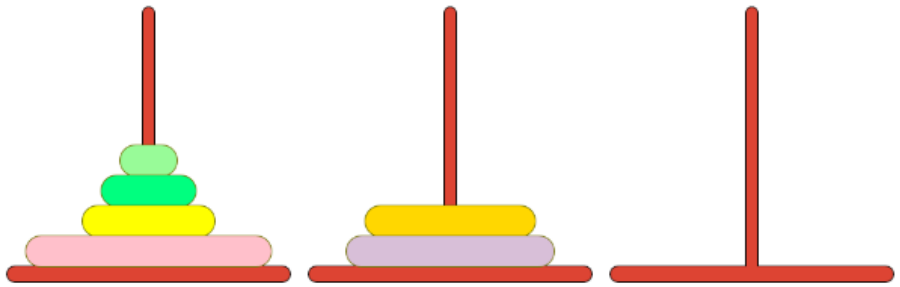

```
import turtle

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()
```

Recursion

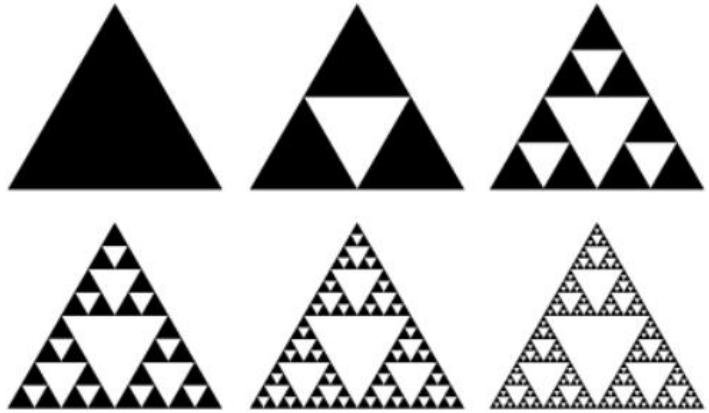
The Functions section of the Math Standards make several references to recursive and recursively defined functions. What is recursion and how does it work?

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

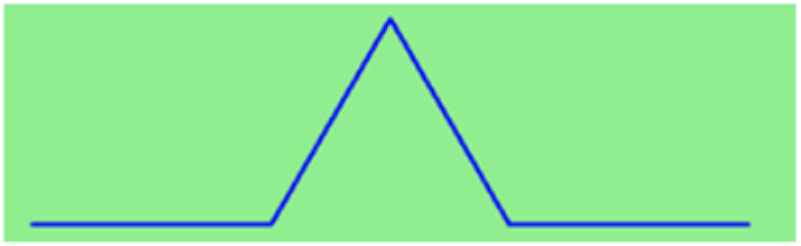


Fractals (example above is Sierpinski Triangle)

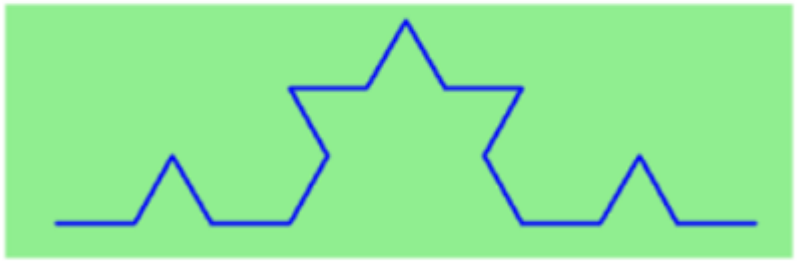
Recursion - Fractals

order 0 Koch fractal

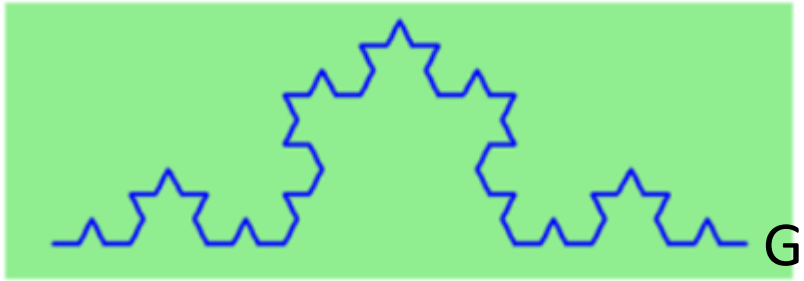
order 1 Koch fractal

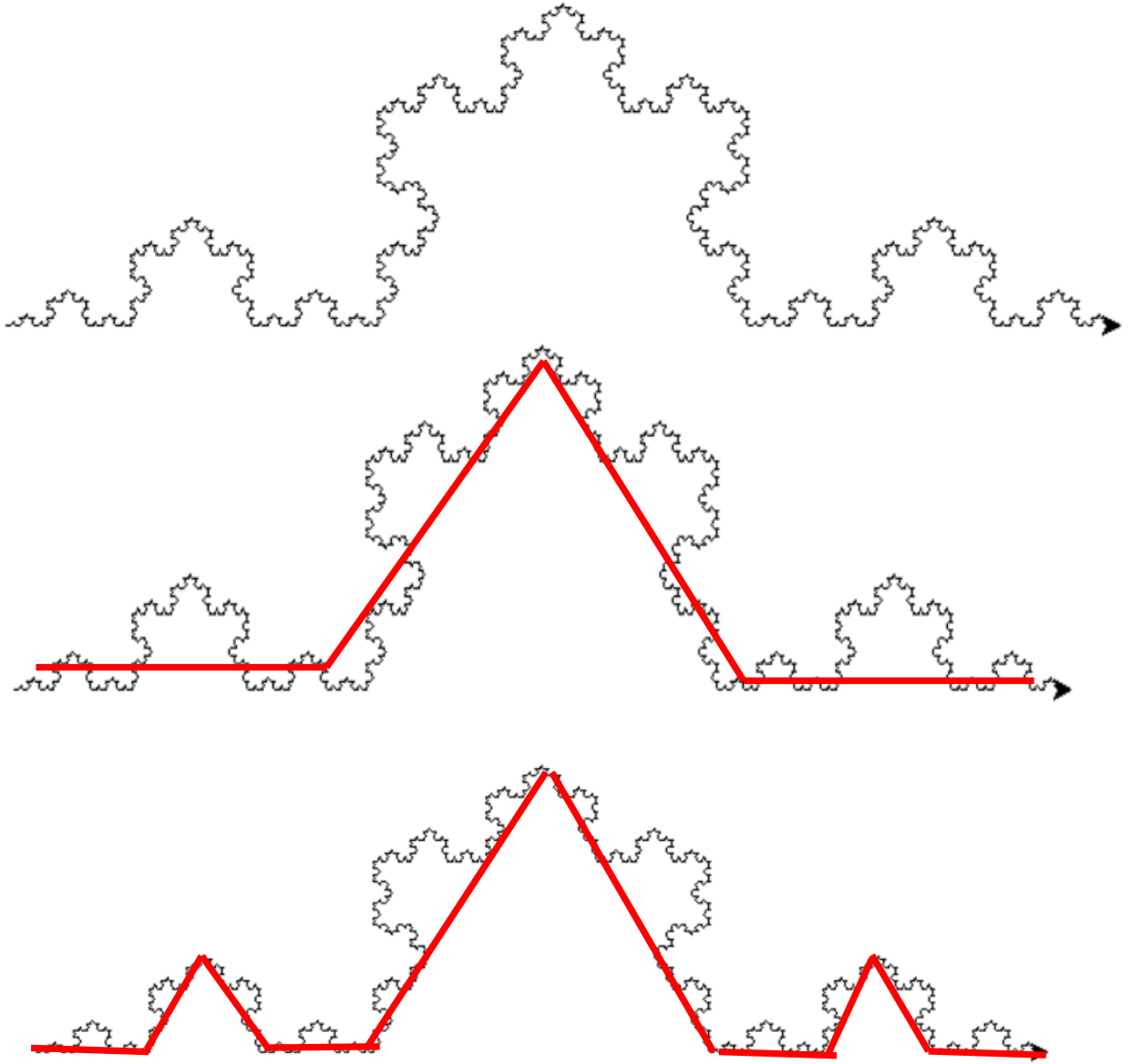


order 2 Koch fractal:



order 3 Koch fractal:



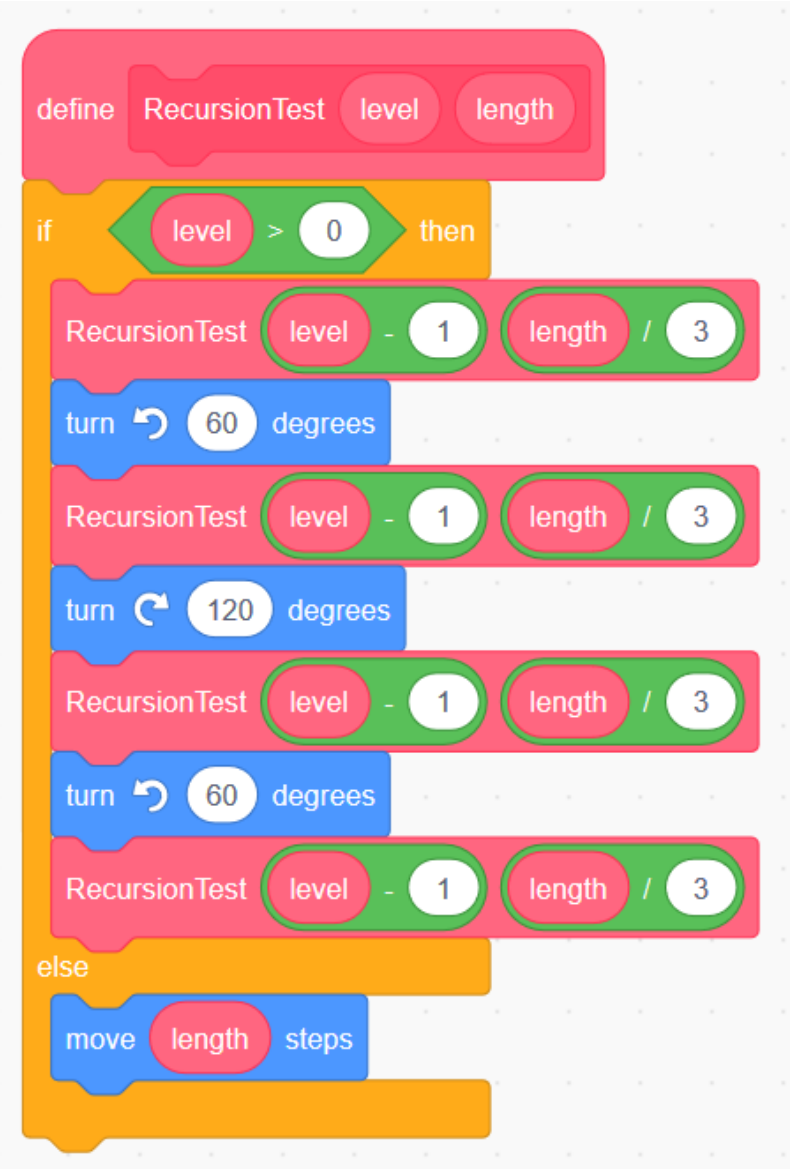


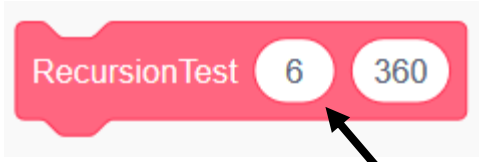
See the pattern?

Graphics on left and Python code at:

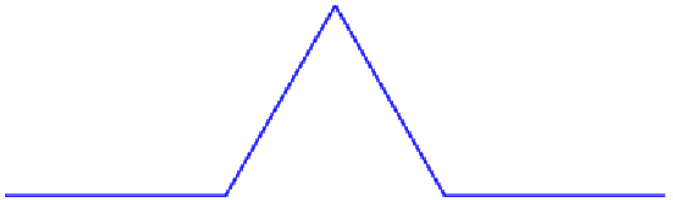
http://openbookproject.net/thinkcs/python/english3e/recursion.html

Recursion

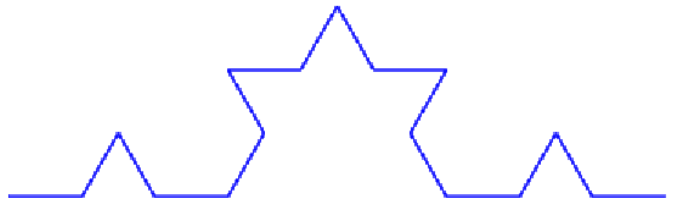




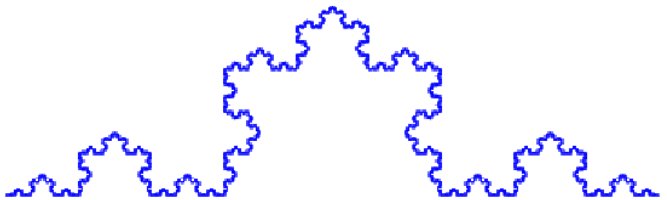
1



2



3



5

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?

A Few Thoughts …

Solving mathematical problems that would be too time consuming or complex to complete by hand. Student and teacher can focus on studying, understanding, and solving the problem rather than the manual calculating. There is a place for the manual calculations and a place for this type of activity.

Exploring mathematics in a visual and concrete way to provide for a more complete comprehension and understanding. Combining coding with the lecture, worksheets, practice exercises, and other activities gives the student more touch points and opportunities to better explore and understand.

Iterate … write one part of the code, add some additional code, keep revising. Sometimes it is best to throw out the code and start over taking a new approach.

Suggestions, Tips, and Examples

Tip: Do talk through how to approach and solve a problem. Let students throw out suggestions and be creative. Try a few different approaches (paper or code).

Tip: Do not explain a lot of the coding concepts. Give a simple explanation, do a few examples, and give students work to do. Explain more as they progress.

Ex1: Younger students can use variables. Box that stores one value at a time. Ask what things don't we know or what things do we need to store in a box.

Ex2: Negative numbers and the coordinate system, draw the x and y axis. Plot points. Plot negative numbers (bigger number further to left for x and down for y). Use white board and code, the students get immediate feed back. For animation, drop an object from the sky to the ground (straight down). What does x do (bigger, smaller, same)? What does y do? Airplane on bottom left to top right of the screen, what does x and y do? Race car goes straight across the screen from right to left, what does x and y do (bigger, smaller, same)?

Q&A … math.code v2

You can incorporate coding into your math classroom

- Questions for others in the session or me?
- Suggestions for the group?
- How will today's session positively impact your students?
- Please email me if any questions or things I can help you with. Let me know how your coding activity went.

Materials at: http://teachintech.noacsc.org/math.code
including last year's session materials (v1)