Putting the Fun in Functions with Python: Functions and Rotations

In this project, you will create rotational pieces of artwork. First, you will use functions, loops and exterior angles, to create regular polygons. Next, you will use translations to move your polygon to various locations on the coordinate plane. Lastly, you will use rotations, to create rotational pieces of artwork.

Objectives:

Programming Objectives:

- Define and use functions
- Use function notation to modularize code
- Use loops to repeat lines of code

Math Objectives:

- Use functions in a problem-solving situation

- Represent transformations in the plane
- Given a regular polygon, describe the rotations and reflections that carry it onto itself

Math Course Connections: Geometry- Translations and Rotations

For this project, you will write a program that uses functions, translations, and rotations to create works of art. You will write a program that lets you draw regular polygons anywhere on the screen. You will then use loops and rotations around a point to create symmetric art.
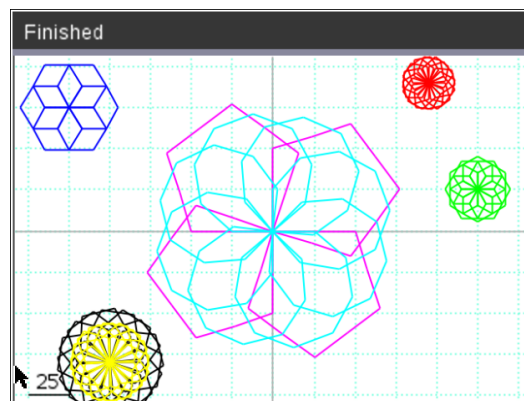
1. In math class you've used many functions. You may recall, for each input, a function has one and only one output.

   For example, the function to calculate and find the area of a square is:
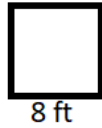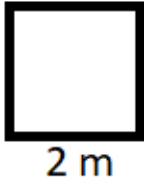 $$area(side) = side^2 \qquad \text{or short hand} \quad a(s) = s^2$$

   The input side = 5 has exactly one output, 25.
   The input side = 7 has exactly one output, 49.

2. Use the function from step 1 to find the area for the following squares:
(not drawn to scale)

2 m

13.5 in

8 ft

3. You might have used the equation f(c) = $\frac{9}{5}c + 32$ to convert degrees Celsius to degrees Fahrenheit.

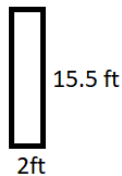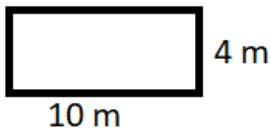Use the function to complete the table below:

Celsius	Fahrenheit
0	
10	
20	
-5	

4. Some functions take more than one parameter.

For example, the area of a rectangle is:
area(base, height) = base x height   or   a(b, h) = b x h

Use the function to find the area of the rectangles below:
(Not drawn to scale)

4 m

10 m

15.5 ft

2ft

5. Computer programmers write functions to carry out repeated actions just like the math functions above. For this first activity, you will write a function to draw regular polygons with various dimensions.

The picture on the right, has three different colored regular polygons, drawn using the same polygon function three different times.
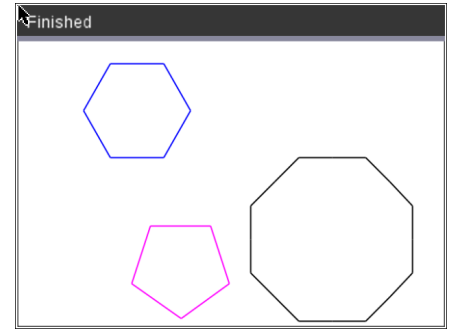
6. There are a few key pieces of information you need to draw regular polygon. What do you think this information is?

   1.)

   2.)

   3.)

   4.)

Teacher Tip: Possible answers might include: number of sides, length of sides, (x,y) location to start drawing, color. Student might also mention inter or exterior angle measurements.

7. How would you tell someone to draw a specific regular polygon?

   If your pencil is at the origin (0,0), what would be the steps to draw the given pentagon? (The first step has been completed for you.)

   Steps:
      go forward 50 units
      ?
      ?
      ?
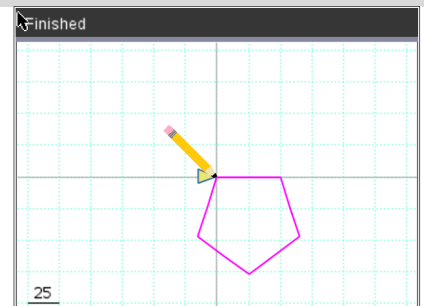
Teacher Tip: rotate 72 degrees, go forward 50 units, rotate 72 degrees, go forward 50 units, rotate 72 degrees, go forward 50 units, rotate 72 degrees, go forward 50 units
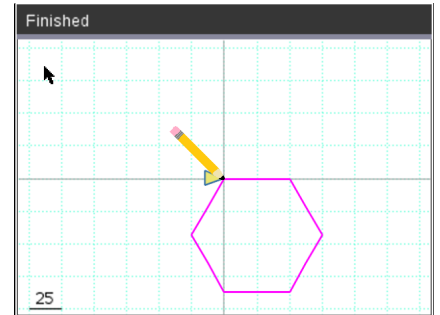
8. How would your steps listed in the previous pentagon example change if the graph was the octagon to the right?
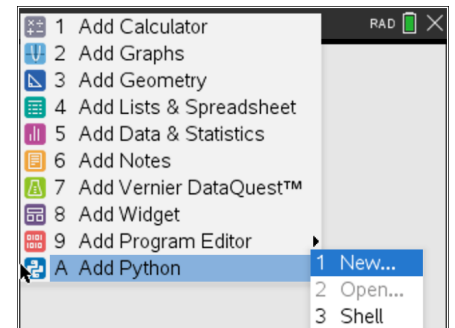
9. After completing the hexagon in part 8 and the pentagon in part 7, revisit your generic list in step 6.
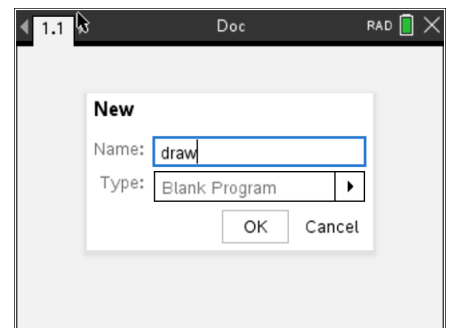
   Add any additional items you deem necessary.

10. Now, let's write your program.

    Start a new program.

    Add a New Python Page

    Name your program "draw".
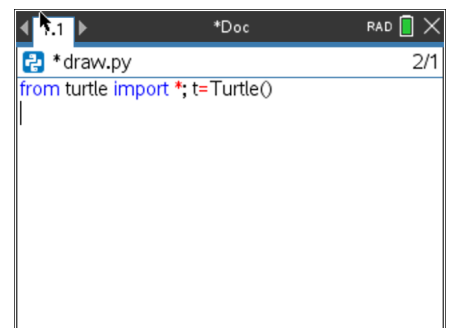   Select the type "Blank Program".

11. To draw in Python, you need the turtle library.

    Menu > More Modules > Turtle Graphics > from turtle import *

    Note: If you don't have Turtle Graphics library as an option, see your teacher or follow the directions at the link below to install it.
    https://education.ti.com/en/product-resources/turtle-module

12. In math class, functions usually start with f(x), g(x), area(s), area(b,h)….
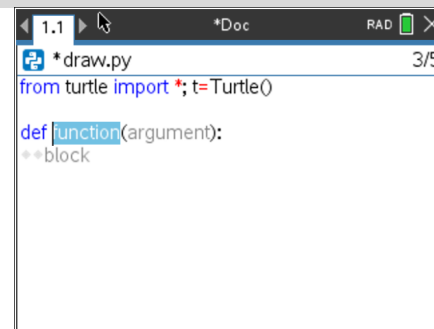You start with a name of a function and the argument (variables) it takes.

To start a function in in python you say:

def functionName(argument):

To get a blank definition,

Menu > Built-ins > Functions > def

Notice, by default, the template includes the blue keyword def,
and gray templates for the name of the function, argument(s), and block
for your code.

13. Look over your list from step 5.
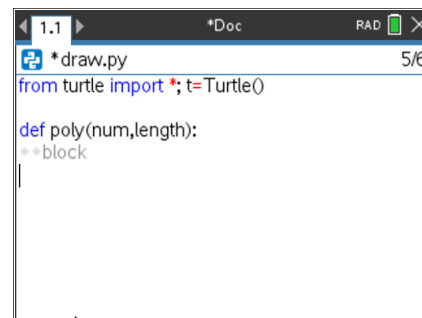
Does your list include number of sides and length of sides?

Those will be the first two arguments for your function.

Name the function poly.

Put num as the first argument and length as the second argument.

def poly(num,length):
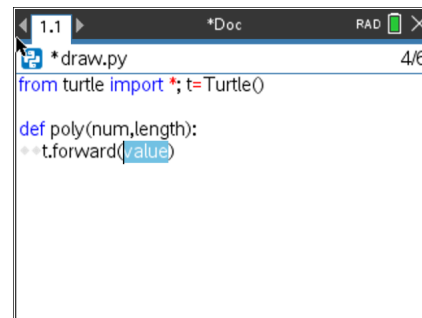
14. For all shapes, the turtle (pencil) needs to move forward. The
command is **t.forward(distance).**

Menu > More Modules > Turtle Graphics > Move > t.forward(distance)

15. Your function has an argument named length. That length will hold the distance the turtle should travel. Replace the gray template "value" with your variable length.

16. Now that your function has one command, lets use the function to draw.

    Go to the next line and remove the two diamonds. This will exit the function definition.

    Go down one more line and type poly(3,50).
    This will call (use) the poly function giving it num = 3 and length = 50.

    Execute your code **[ctrl] [r]**

    Verify your turtle moved forward 50 units.

    Press [enter] to exit the turtle page.

    This will exit to the Python Shell on page 1.2.

    Press [ctrl] [◄] to go back to your code on page 1.1.

17. The length value 50 in t.forward(50) drew a straight line. Now you need to "turn" and draw the next side.

    How far should you "turn"?

18. You are correct if you said that depends on the shape. Each regular polygon has a different exterior angle.

    Find the <u>exterior</u> angle for each shape below.
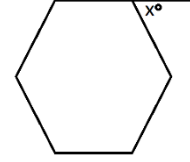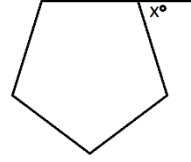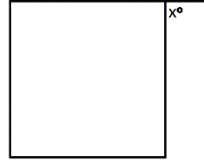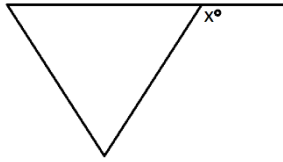
19. What is the generic formula for finding the exterior angle for a regular polygon?

    Add a right turn to your code using your formula. Use the variable num in your formula since that is the argument in your function.

    Execute your program [ctrl] [r].

    The line poly(3,50) stores 3 in the num variable and 50 in the length. Your turtle should have moved forward 50 then rotated 120°.
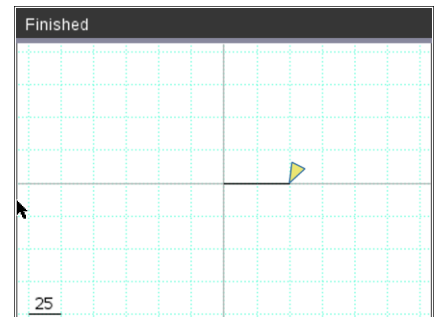
20. Now to add a loop so this process will happen **num** times.

    Add a line between the definition, def poly(num,length):, and the line t.forward(length).

    Add a **for loop**
    Menu > Built-ins > Control > for index in range(size)

21. Replace index with the variable i

    Replace size with num

22. Remove the two blank lines.
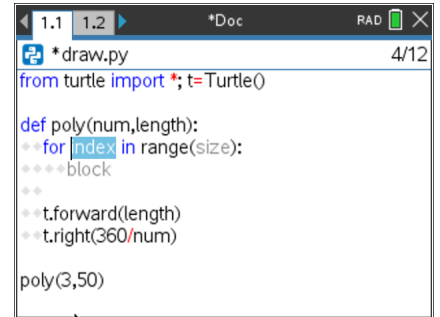
    Currently the forward and right are not part of the **for loop**.
    To be part of the loop, they need to be indented one level.

    To indent the two lines, you can either add two spaces using the space key to the right of the [z].

    Or, you can choose to put your cursor on each line and select
        Menu > Edit > Indent

23. Execute your program [ctrl] [r].

    The line poly(3,50) passes the num-3 and length-50 to the code.
    The **for loop** makes the code happen num (3) times.
        The turtle moves forward length (50)
        The turtle rotates right 360/num = 360/3 = 120°
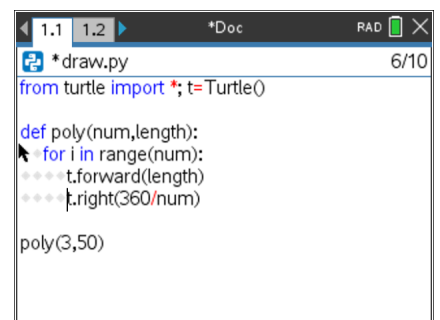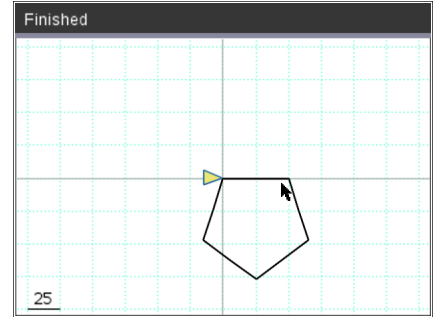
24. To draw a pentagon instead of a hexagon, change the poly(3,50) to a poly(5,50)

25. Now to add some color.

The command t.pencolor will set the pen color.

Menu > More Modules > Turtle Graphics > Pen Control > t.pencolor
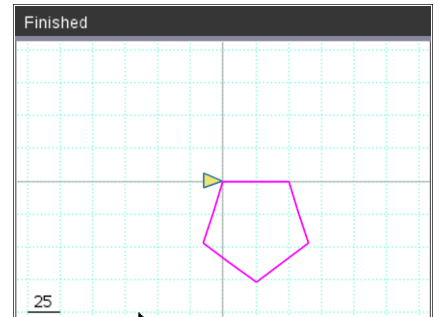
The color options are located in
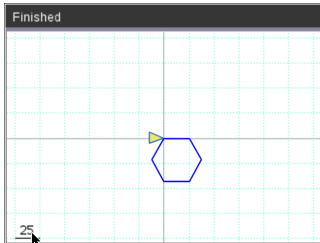Menu > More Modules > Turtle Graphics > Color > Magenta
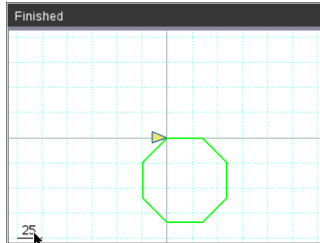
26. Can you modify your code to draw the following:

Hexagon Length 25
Color: blue

function call used:
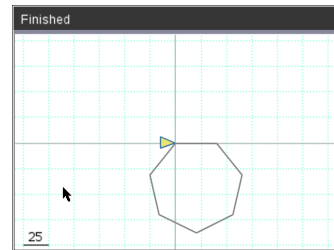
poly(   )

Octagon Length 35
Color: green

function call used:

poly(   )
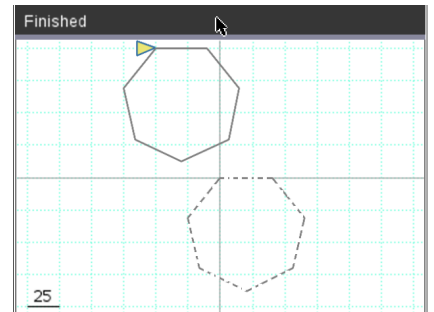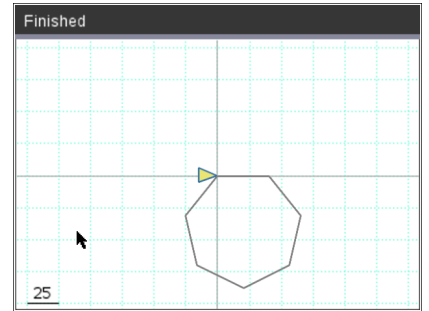
Heptagon Length 40
Color: gray

function call used:

poly(   )

27. Currently, your function draws all the regular polygons with the starting point at (0,0).
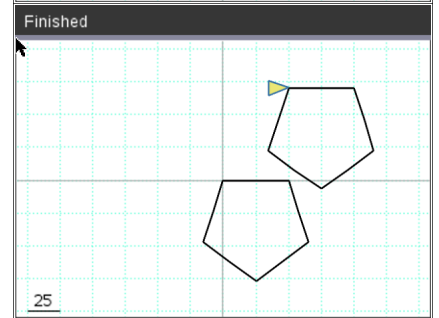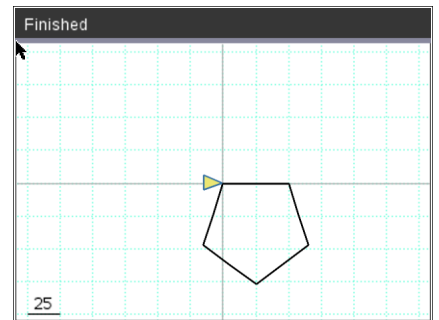
Mathematically, describe the transformation that would result in drawing the heptagon in the new location.

Answer:

28. Mathematically, describe the transformation that would result in drawing the pentagon in the new location.

Answer:

29. Adding a horizontal and vertical **translation** is simple in Python.

To start, add a horizontal and vertical argument to your definition. For ease of typing, let's use h and v.

```
from turtle import *; t=Turtle()

def poly(num,length,h,v):
    for i in range(num):
        t.forward(length)
        t.right(360/num)

t.pencolor("black")
poly(5,50)
```

Math Explorations with Python

TI-NSPIRE CX II™ TECHNOLOGY

30. You will need to pick up the pencil so it doesn't draw.
Preform a horizontal and vertical translation for the starting point.
Put the pencil down.

Menu > More Modules > Turtle Graphics > Pen Control > t.penup()

Menu > More Modules > Turtle Graphics > Move > t.goto(x,y)
     Replace the x,y templates with h and v

Menu > More Modules > Turtle Graphics > Pen Control > t.penup()

Lastly, in your function call give it a horizontal and vertical translation.
The code on the right did a 30 unit horizontal translation and a 50 unit vertical translation.


```
1.1  1.2      *Doc      RAD
draw.py                  1/14
from turtle import *; t=Turtle()

def poly(num,length,h,v):
   t.penup()
   t.goto(h,v)
   t.pendown()
   for i in range(num):
       t.forward(length)
       t.right(360/num)

t.pencolor("black")

poly(5,50,30,90)
```
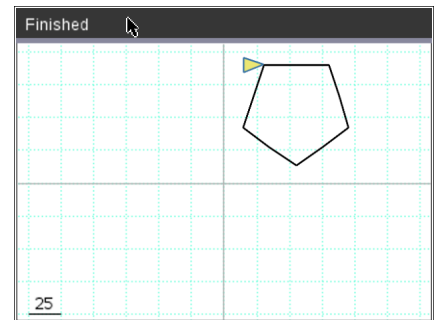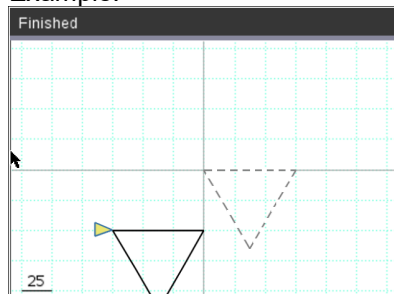

31. Use your function to draw **the solid** polygon. The dotted polygon, shows the original before the translation.
Mathematically, describe the translation and give the function call.
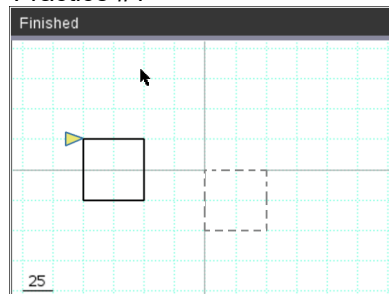
Example:

Math Description:
     Horizontal Translation -75 units
     Vertical Translation -50 units

Function call:
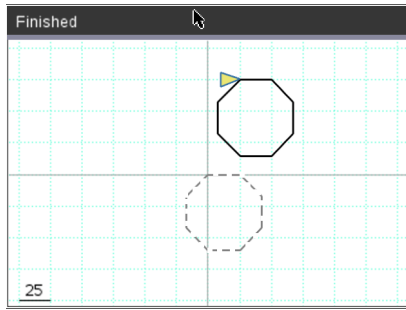   poly(3,70,-75,-50)

Practice #1
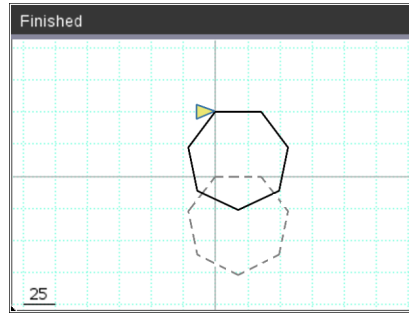

Math Description:

Function call:

Practice #2

Math Description:

Function call:

Practice #3

Math Description:

Function call:

32. Now let's add a rotation.

If you change your function call from one line, such as

poly(5,50,0,0)

to

for i in range(2):
poly(3,50,0,0)
t.right(20)

The code will draw the polygon, rotate right 20°, then repeat the code for a second polygon.

**for
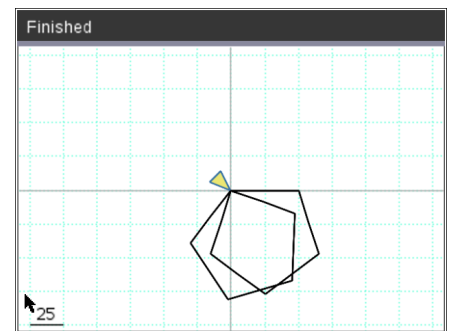Menu > Built-ins > Control > for index in range(size):

**Rotate right
Menu > More Modules > Turtle Graphics > Move >

t.right(angle)

33. Modify your code to draw 5 pentagons each rotated 20 additional degrees from the previous one.

34. Modify the right rotation degree so the 4th copy of the pentagon (5th pentagon) "maps" or traces precisely over the original pentagon.
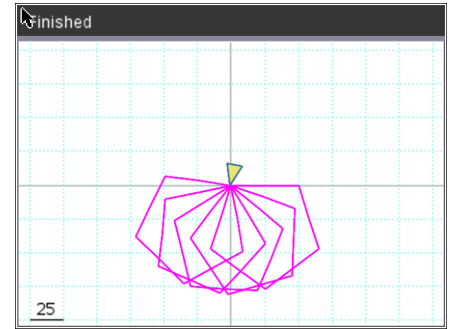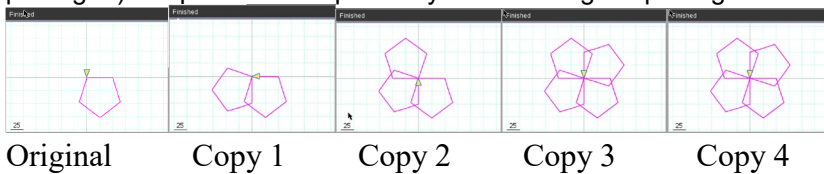
Original	Copy 1	Copy 2	Copy 3	Copy 4

???

Record your answer here:_____

35. What angle of rotation allows a hexagon to rotate 4 times and "map" back onto itself?

36. Do you think this same angle rotation would result in the 4th copy of an octagon "mapping" on itself? Explain your thinking.

37.

How many degrees should you rotate if you want to draw 5 hexagons before the 6th "maps" onto the original?

```
1.1  1.2          *Doc        RAD  X
*draw.py                      14/17
  t.penup()
  t.goto(h,v)
  t.pendown()
  for i in range(num):
    t.forward(length)
    t.right(360/num)

t.pencolor("magenta")
for i in range(6):
  poly(6,50,0,0)
  t.right(■■)
```
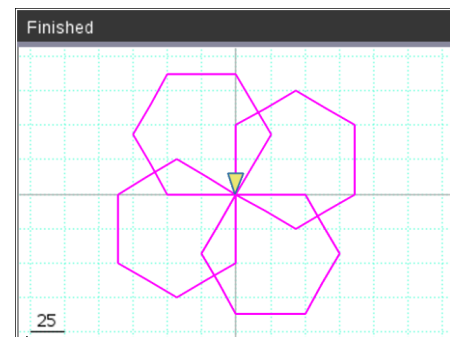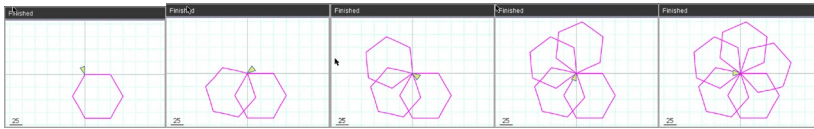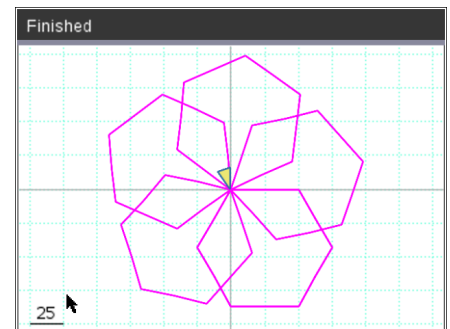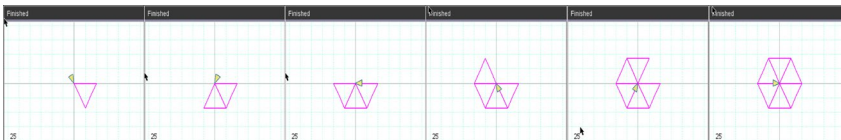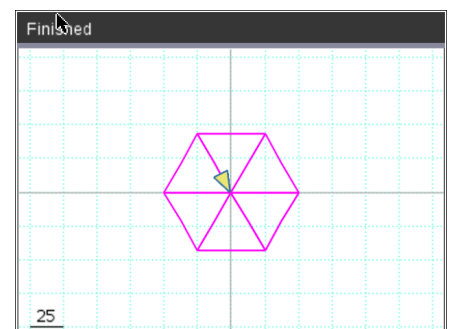

38.

How many degrees should your rotate if you want to draw 6 triangles before the 7th "maps" onto the original?
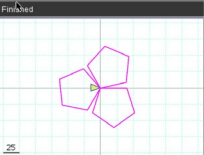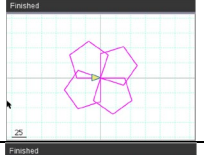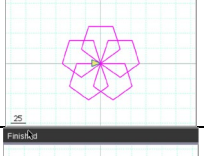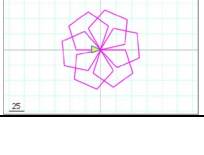
```
1.1  1.2          *Doc        RAD  X
*draw.py                      14/17
  t.penup()
  t.goto(h,v)
  t.pendown()
  for i in range(num):
    t.forward(length)
    t.right(360/num)

t.pencolor("magenta")
for i in range(7):
  poly(3,50,0,0)
  t.right(■■)
```
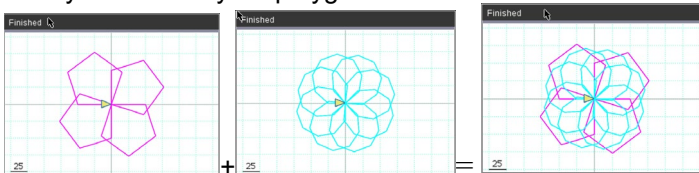
39. In general, the math to determine the number of degrees to rotate to "map" onto the original.

#of shapes drawn before a "map"		Math
3		
4		
5		
6		
N		

40. Now you can use your polygon function to make rotational art.

 + =

Rotated pentagon   Rotated nonagon   Geometric Art

For each new rotated polygon, you pick the
      color
      add a for loop
          use the poly function
          rotate

Hint:

To speed up the drawing, add t.speed(10) before you draw.

Menu > More Modules > Turtle Graphics > Setting > t.speed

To hide the turtle

Menu > More Modules > Turtle Graphics > Setting > t.hideturtle
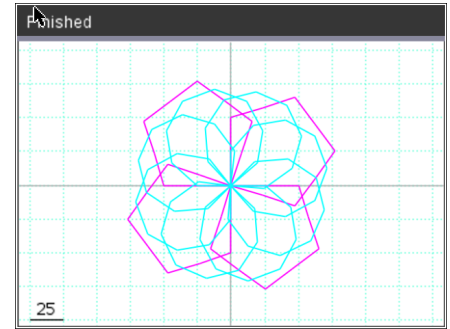
41. Add in some smaller shapes that are translated and rotated:

- What kind of artwork can you make?

- Can you make artwork that has at least three different rotational works of art centered at different points?

```
from turtle import *; t=Turtle()

def poly(num,length,h,v):
 t.penup()
 t.goto(h,v)
 t.pendown()
 for i in range(num):
   t.forward(length)
   t.right(360/num)

t.speed(15)
t.hideturtle()
t.pencolor("magenta")
for i in range(4):
 poly(5,50,0,0)
 t.right(90)

t.pencolor("cyan")
for i in range(8):
 poly(9,25,0,0)
 t.right(45)

t.pencolor("blue")
```
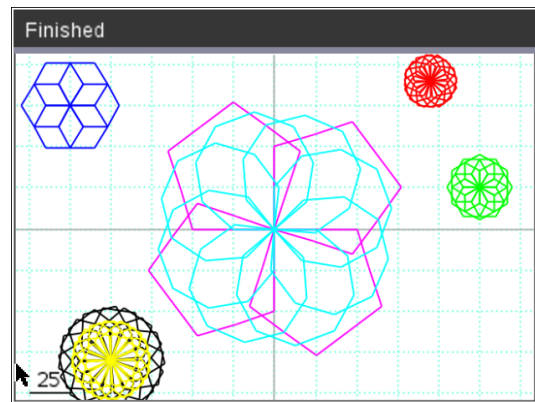
```
for i in range(6):
  poly(6,15,-125,75)
  t.right(60)
t.pencolor("green")
for i in range(12):
  poly(6,10,125,25)
  t.right(30)

t.pencolor("red")
for i in range(16):
  poly(5,10,95,90)
  t.right(360/16)

t.pencolor("black")
for i in range(16):
  poly(5,20,-100,-80)
  t.right(360/16)

t.pencolor("yellow")
for i in range(16):
  poly(5,15,-100,-80)
  t.right(360/16)
```