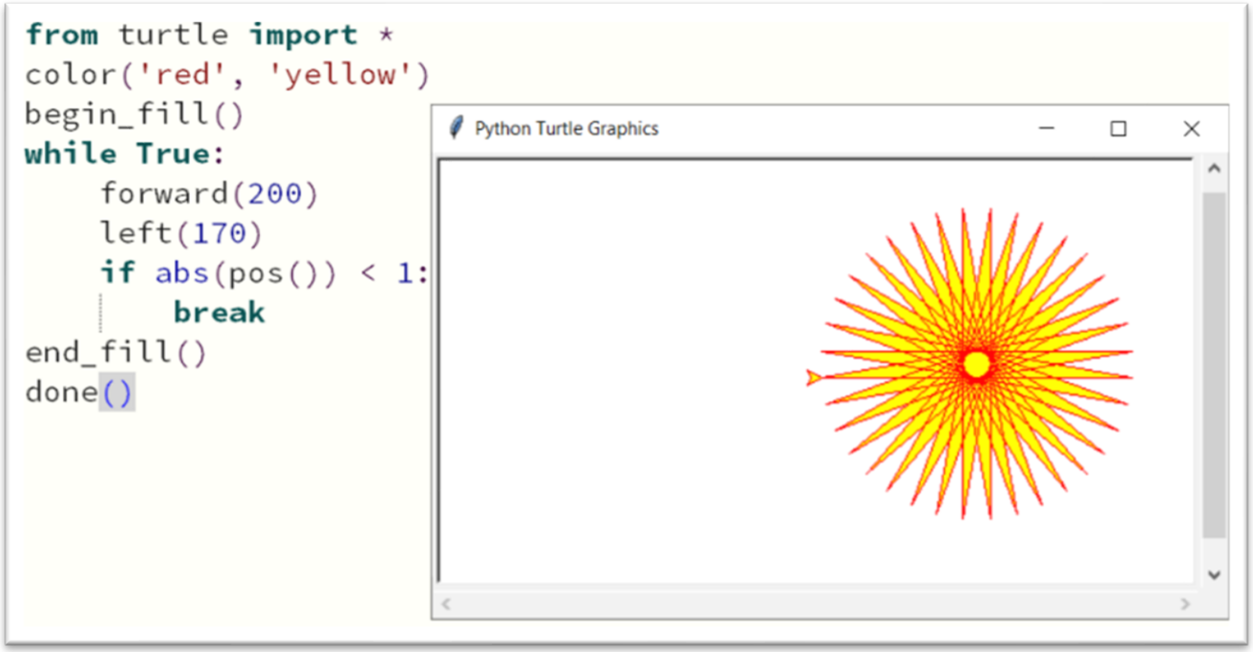
Drawing with Code



```
from turtle import *
color('red', 'yellow')
begin_fill()
while True:
    forward(200)
    left(170)
    if abs(pos()) < 1:
        break
end_fill()
done()
```

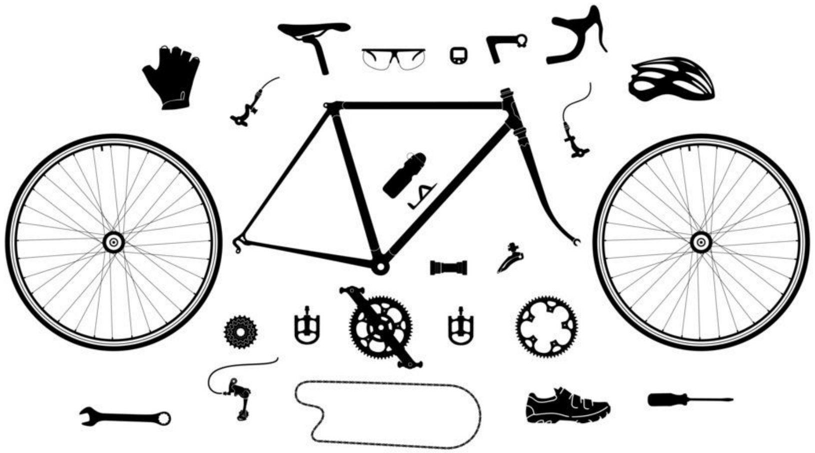
Powered by AFRL NM STEM Academy

Drawing with Code

- We will use the Python Turtle module to make drawings with code.
- A module is a collection of code that can be used in your program.
- Programming with modules makes coding faster and easier because most of the code is already written.
- We use pieces of modules that we need to make something new.

Modules

Using modules is sort of like ordering pizza instead of making it from scratch.
It’s possible to build a bicycle by ordering needed parts and assembling the parts at your home. You may also be able to order a bicycle by specifying what parts you want to include. Thankfully, there is no need to make handlebars or seats.



Drawing with Code

Copy and paste the code below into your Python editor.

- The import turtle statement will provide access to the turtle module code.
- The next line of code creates a turtle pen and gives it a name.

```
# Drawing with turtles
import turtle
larry = turtle.Turtle()
larry.color('blue')
larry.pensize(10)
larry.shape('turtle')
#
larry.forward(50)
larry.left(45)
larry.forward(100)
```

Drawing with Code

- The color, size and shape of the pen can be selected with functions from the module.
- The forward function will move the pen forward a desired number of pixels.
- The left function will turn to the left a desired angle.

```
# Drawing with turtles
import turtle
larry = turtle.Turtle()
larry.color('blue')
larry.pensize(10)
larry.shape('turtle')
#
larry.forward(50)
larry.left(45)
larry.forward(100)
```

Drawing with Code

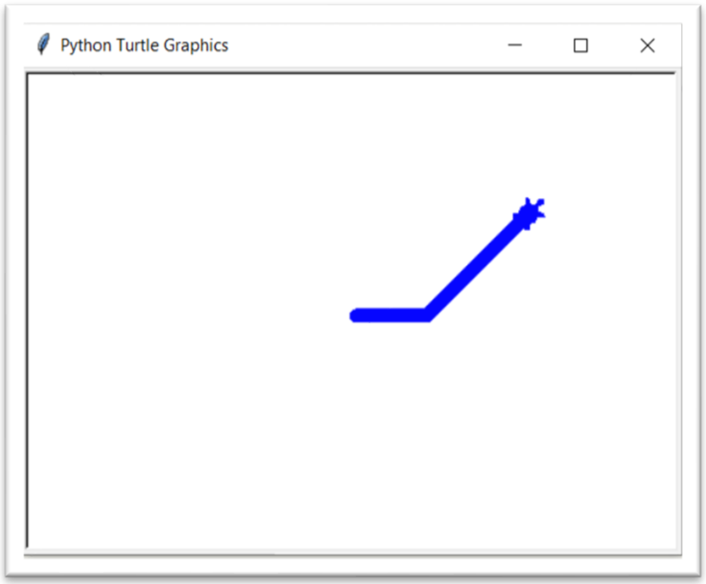
Click on **Save** and then **Run**.

```
# Drawing with turtles
import turtle
larry = turtle.Turtle()
larry.color('blue')
larry.pensize(10)
larry.shape('turtle')
#
larry.forward(50)
larry.left(45)
larry.forward(100)
```

Drawing with Code

The Turtle Graphics window will display the results of your code.

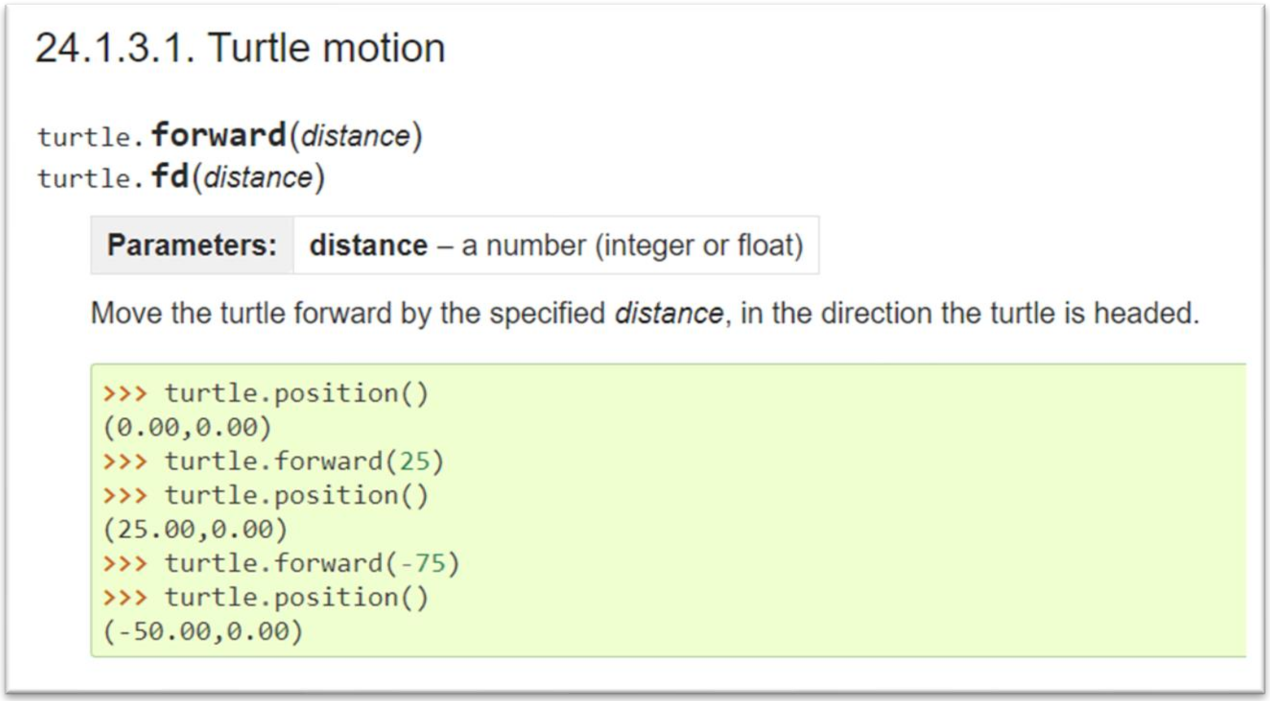
- The pen starts in the middle of the screen, facing the right of the screen.
- The pen should move quickly.
- The window will be larger or smaller depending on the editor used.



- All of the above variables(position, pen speed, window size) can be changed with Turtle functions.

Drawing with Code

Go to the Python turtle library to learn about how to use all of the options available: https://docs.python.org/3.3/library/turtle.html?highlight=turtle



24.1.3.1. Turtle motion

turtle.**forward**(*distance*)
turtle.**fd**(*distance*)

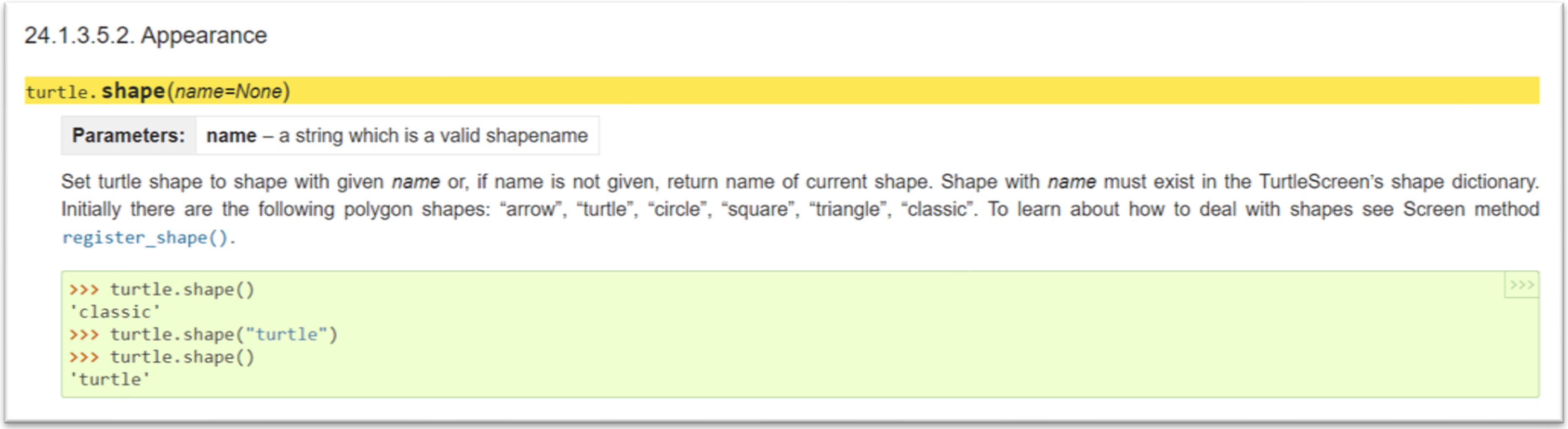
Parameters:	**distance** – a number (integer or float)

Move the turtle forward by the specified *distance*, in the direction the turtle is headed.

```
>>> turtle.position()
(0.00,0.00)
>>> turtle.forward(25)
>>> turtle.position()
(25.00,0.00)
>>> turtle.forward(-75)
>>> turtle.position()
(-50.00,0.00)
```

Drawing with Code

The following is a description of the shape function or method. You may see mention of methods and functions. For our purposes, methods are functions.


24.1.3.5.2. Appearance

turtle.**shape**(*name=None*)

Parameters: **name** – a string which is a valid shapename

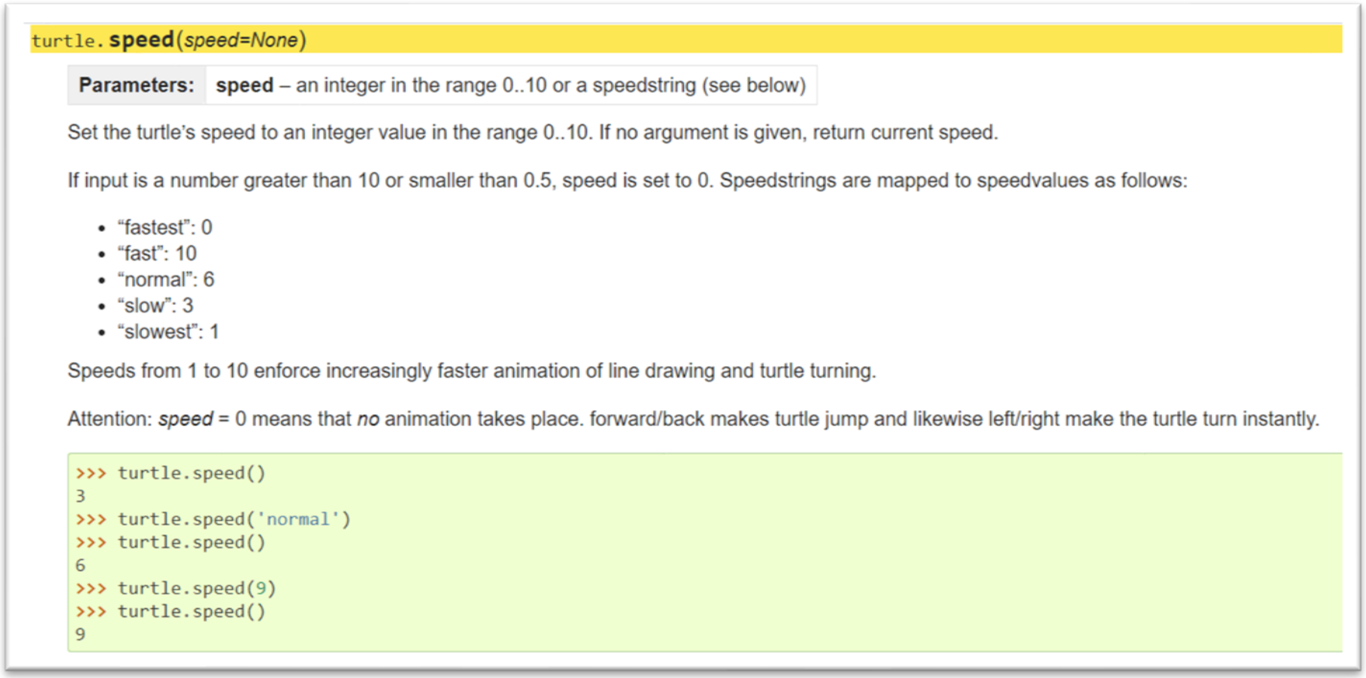
Set turtle shape to shape with given *name* or, if name is not given, return name of current shape. Shape with *name* must exist in the TurtleScreen's shape dictionary. Initially there are the following polygon shapes: "arrow", "turtle", "circle", "square", "triangle", "classic". To learn about how to deal with shapes see Screen method `register_shape()`.

```
>>> turtle.shape()
'classic'
>>> turtle.shape("turtle")
>>> turtle.shape()
'turtle'
```

Explore the library at:
https://docs.python.org/3.3/library/turtle.html?highlight=turtle

Drawing with Code

The speed function may be useful to slow down or speed up the pen.



`turtle.`**`speed`**`(`*`speed=None`*`)`

Parameters:	**speed** – an integer in the range 0..10 or a speedstring (see below)

Set the turtle's speed to an integer value in the range 0..10. If no argument is given, return current speed.

If input is a number greater than 10 or smaller than 0.5, speed is set to 0. Speedstrings are mapped to speedvalues as follows:

- "fastest": 0
- "fast": 10
- "normal": 6
- "slow": 3
- "slowest": 1

Speeds from 1 to 10 enforce increasingly faster animation of line drawing and turtle turning.

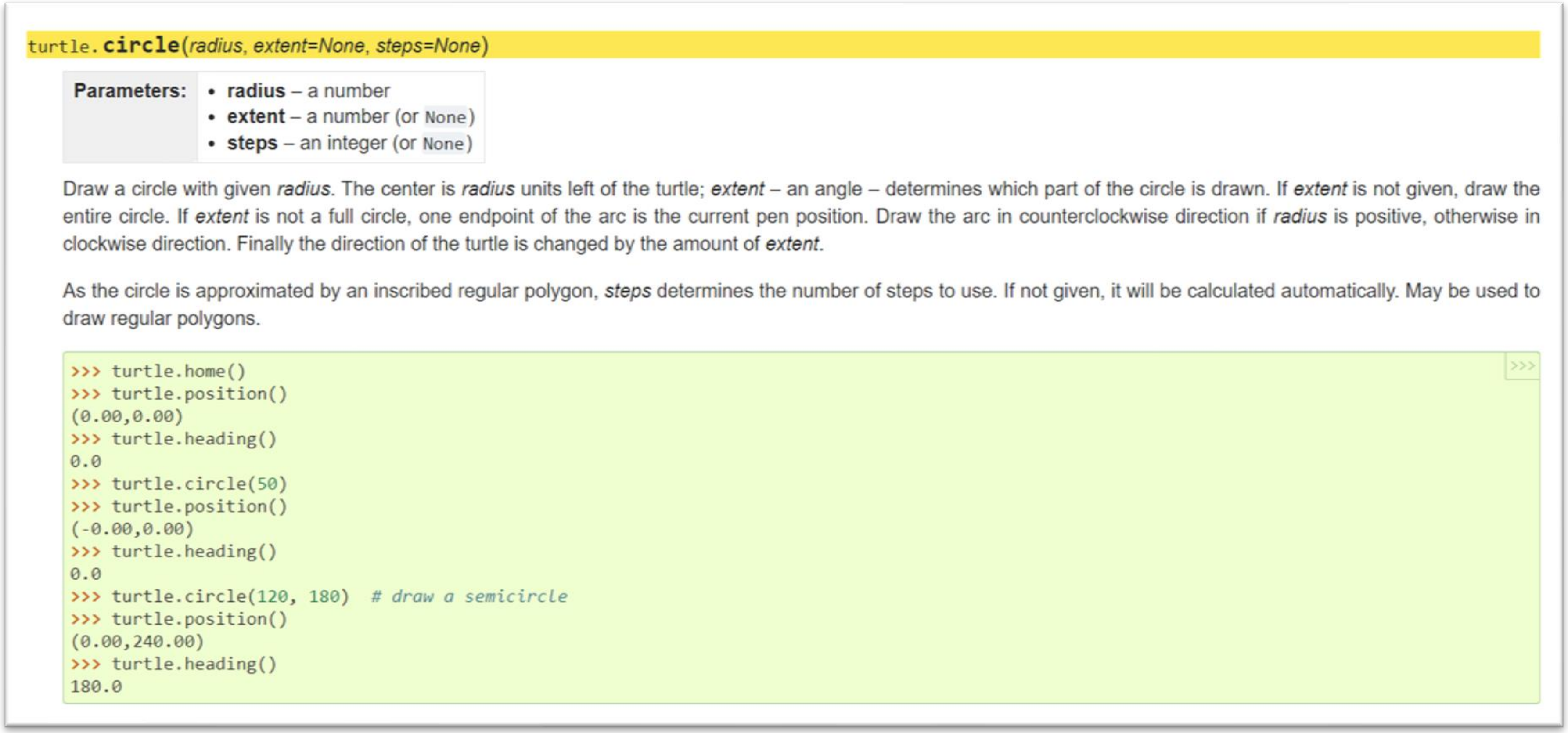
Attention: *speed* = 0 means that *no* animation takes place. forward/back makes turtle jump and likewise left/right make the turtle turn instantly.

```
>>> turtle.speed()
3
>>> turtle.speed('normal')
>>> turtle.speed()
6
>>> turtle.speed(9)
>>> turtle.speed()
9
```

Explore the library at:
https://docs.python.org/3.3/library/turtle.html?highlight=turtle

Drawing with Code

The circle function below is fun to play with. There are several other drawing options available.



turtle.**circle**(*radius, extent=None, steps=None*)

Parameters:
- **radius** – a number
- **extent** – a number (or `None`)
- **steps** – an integer (or `None`)

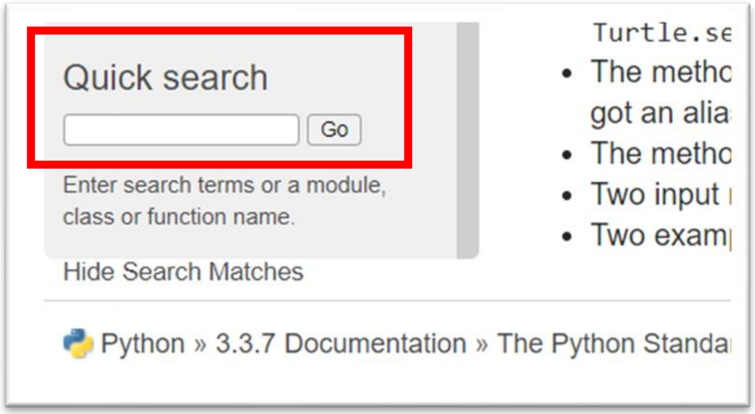
Draw a circle with given *radius*. The center is *radius* units left of the turtle; *extent* – an angle – determines which part of the circle is drawn. If *extent* is not given, draw the entire circle. If *extent* is not a full circle, one endpoint of the arc is the current pen position. Draw the arc in counterclockwise direction if *radius* is positive, otherwise in clockwise direction. Finally the direction of the turtle is changed by the amount of *extent*.

As the circle is approximated by an inscribed regular polygon, *steps* determines the number of steps to use. If not given, it will be calculated automatically. May be used to draw regular polygons.

```
>>> turtle.home()
>>> turtle.position()
(0.00,0.00)
>>> turtle.heading()
0.0
>>> turtle.circle(50)
>>> turtle.position()
(-0.00,0.00)
>>> turtle.heading()
0.0
>>> turtle.circle(120, 180)  # draw a semicircle
>>> turtle.position()
(0.00,240.00)
>>> turtle.heading()
180.0
```

Drawing with Code

There is a great search feature on the Python library site. Scroll to the bottom of the page. There is a small search tool on the left side of the page. This makes it easier to find something you may be trying to find.


Quick search

Go

Enter search terms or a module, class or function name.

Hide Search Matches

`Turtle.se`

- The metho got an alia
- The metho
- Two input
- Two examp

Python » 3.3.7 Documentation » The Python Standa

Drawing with Code

As you write code to draw shapes, you may want to lift the pen, move it to another place, and continue to draw. The below code shows how to move the pen to a different place in the window to continue drawing.

```
larry.penup()
larry.goto(-280,100)
larry.pendown()
larry.forward(200)
```

- The goto() function will move the pen to an X, Y coordinate in the drawing window.
- The middle of the screen is (0,0).

Drawing with Code

This is your code so far.

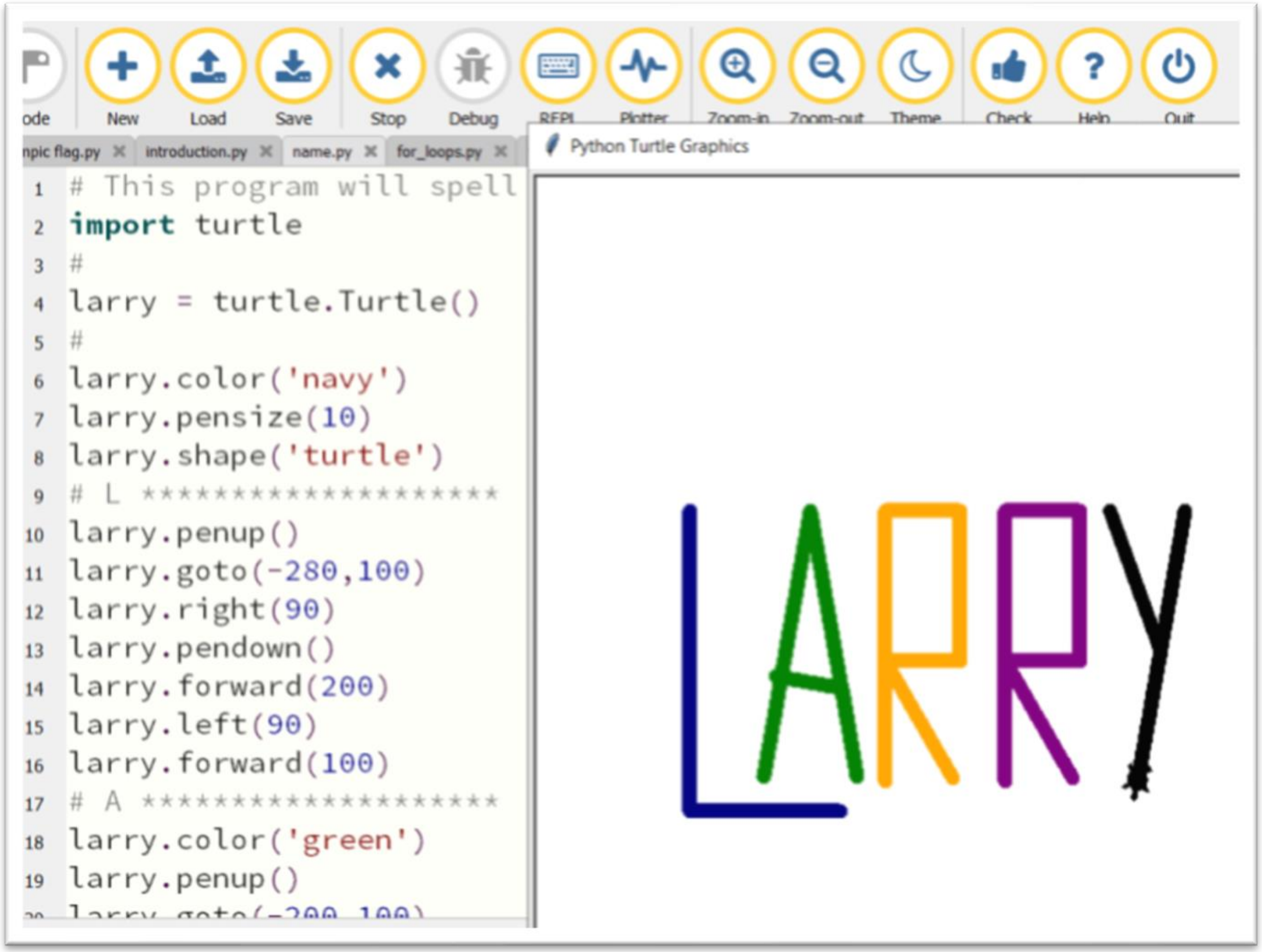
You can modify this code to draw most shapes.

Spend some time playing with it. Try to draw some simple shapes.

Copy and paste the code into your Python editor and run the program.

```
# Drawing with turtles
import turtle
larry = turtle.Turtle()
larry.color('blue')
larry.pensize(10)
larry.shape('turtle')
#
larry.forward(50)
larry.left(45)
larry.forward(100)
#
larry.penup()
larry.goto(-280,100)
larry.pendown()
larry.forward(200)
larry.left(90)
```

Your first challenge is to write code that will draw your first name.



```
# This program will spell
import turtle
#
larry = turtle.Turtle()
#
larry.color('navy')
larry.pensize(10)
larry.shape('turtle')
# L *******************
larry.penup()
larry.goto(-280,100)
larry.right(90)
larry.pendown()
larry.forward(200)
larry.left(90)
larry.forward(100)
# A *******************
larry.color('green')
larry.penup()
larry.goto(-200,100)
```