**Outline**

Develop a better understanding of procedural sequencing by solving shape drawing challenges using the turtle environment.

**Objectives**

* Use correct terminology to describe programming concepts;
* Describe the types of data that computers can process and store (e.g., numbers, text);
* Explain the difference between constants and variables used in programming;
* Use variables, expressions, and assignment statements to store and manipulate numbers and text in a program

**Materials**

* Python Turtle Development Environment at: https://repl.it/
* PythonWorksheetII form the GitHub Repository
* Web links identified in the questions below

**Level 1: Drawing Basic Shapes With Python Turtle**

1. Open the document PythonWorksheetII from the class GItHub repository.   
   Read over “Part III” at the end of the PythonWorksheetII document.

I have done this.

1. Create an new Repl by selecting the “Python with Turtle” language / environment.

I have done this.

1. Begin all of your turtle programs with the following code to create a “pen”:

import turtle

myPen = turtle.Turtle()

I have done this.

1. Create a program to draw a red circle.
   1. Provide a listing of your program code below:

import turtle

myPen = turtle.Turtle()

myPen.color("red")

myPen.circle(100)

1. Create a program to draw any three of the shapes described in “Part III” of   
   the PythonWorksheetII document.
   1. Provide a listing of your program code below:

import turtle

myPen = turtle.Turtle()

myPen.width(2)

myPen.speed(5)

myPen.color("red")

myPen.circle(50)

myPen.color("Blue")

myPen.forward(50)

myPen.left(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(50)

myPen.color("Green")

myPen.circle(100)

myPen.color("red")

myPen.right(180)

myPen.circle(50)

myPen.color("Blue")

myPen.forward(50)

myPen.left(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(50)

myPen.color("Green")

myPen.circle(100)

myPen.color("Yellow")

myPen.forward(100)

myPen.left(90)

myPen.forward(200)

myPen.left(90)

myPen.forward(200)

myPen.left(90)

myPen.forward(400)

myPen.left(90)

myPen.forward(200)

myPen.left(90)

myPen.forward(200)

myPen.left(90)

myPen.forward(200)

myPen.left(180)

myPen.forward(200)

myPen.right(90)

myPen.forward(200)

myPen.color("Orange")

myPen.forward(200)

myPen.right(90)

myPen.forward(200)

myPen.right(90)

myPen.forward(200)

myPen.left(90)

myPen.forward(200)

myPen.right(90)

myPen.forward(400)

myPen.right(90)

myPen.forward(200)

myPen.left(90)

myPen.forward(200)

myPen.right(90)

myPen.forward(200)

myPen.right(90)

myPen.forward(200)

myPen.left(90)

myPen.forward(200)

myPen.right(90)

myPen.forward(400)

myPen.right(90)

myPen.forward(200)

**Level 2: Using a Loop**

1. Google the keywords “Python Turtle Methods”.
   1. Explain how the “goto” method works and how you could use it when drawing repeated shapes.

The goto method works by moving the turtle x or y positions from the beginning position. We could change the position of the shape slightly and repeat it.

* 1. List some other useful methods not listed in “Part III” at the end of the PythonWorksheetII document.

Some other useful methods are fillcolour, goto, heading, position, begin\_fill, end\_fill, dot, stamp, shape.

1. Create a repeating pattern on your screen. The pattern must meet the following requirements:
   1. The basic pattern must be made up of several individual Turtle methods (e.g. changes of colour, changes of direction, size, motion, etc.)

Code is below.

* 1. The basic pattern must be repeated several times with a shift in starting position each time.

import turtle

myPen = turtle.Turtle()

myPen.speed(10)

myPen.width(10)

for he in[1,2,3,4,5]:

myPen.color("red")

myPen.circle(50)

myPen.up()

myPen.goto(he\*50, 0)

myPen.down()

1. Use a Python Loop to create your repeating pattern
   1. The Loop may be a Counted Loop or a Conditional Loop
   2. The indented block of code for the loop should be your basic pattern.

I have done this.

1. Provide a listing of your repeating pattern loop below.

import turtle

myPen = turtle.Turtle()

myPen.speed(10)

myPen.width(10)

for he in[1,2,3,4,5]:

myPen.color("red")

myPen.circle(50)

myPen.forward(50)

myPen.left(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(50)

myPen.up()

myPen.goto(he\*100,0)

myPen.down()

**Level 3: Defining a Function**

1. Google the keywords “Python Function Syntax”.
   1. Explain what the “def” keyword does

The keyword def marks the start of the function header.

* 1. Explain any special rules regarding the function name

A function name to uniquely identify it. Function naming still follows the same rules of writing as in python.

* 1. Explain what the parameters (or arguments) do

Through parameters or arguments, we pass values to a function and they are optional.

* 1. Where should the colon “:” be placed

A colon is to be placed at the end to mark the end of function header.

* 1. Explain how to write Python statements that make up the function body

There has to be the optional documentation string(docstring) to describe what the function does. Then there has to be one or more valid python statements that make up the function body. Statements should have the same indentation level.

* 1. Explain the “return” statement

An option is return and this is to return a value from a statement.

1. Provide an example of a simple function that uses one or more parameters.
   1. Write the function definition below
   2. Write some code to call the function below

import turtle

myPen = turtle.Turtle()

def greet(name):

  print("Hey there, " + name + ". How are you?")

greet('Jasjot')

‘Jasjot’

1. Convert your basic pattern (from Level 2 above) into a function

import turtle

myPen = turtle.Turtle()

myPen.speed(10)

myPen.width(10)

def my\_pattern (x,y):

for he in[1,2,3,4,5]:

myPen.color("red")

myPen.circle(50)

myPen.up()

myPen.goto(he\*50, 0)

myPen.down()

my\_pattern(50,50)

The function name should be “my\_pattern”

The parameters should be the x and y starting position for your pattern

Your function does not need to use the “return” statement

1. Use a your basic pattern function and a Python Loop to create your repeating pattern
   1. The Loop may be a Counted Loop or a Conditional Loop
   2. Your function should be called from within the loop.
2. Provide a listing of your function definition and repeating pattern loop below.

import turtle

myPen = turtle.Turtle()

myPen.speed(10)

myPen.width(10)

def my\_pattern (x,y):

for he in[1,2,3,4,5]:

myPen.color("red")

myPen.circle(50)

myPen.up()

myPen.goto(he\*50, 0)

myPen.down()

my\_pattern(50,50)