**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.
2. Create an new Repl by selecting the “Python with Turtle” language / environment.
3. Begin all of your turtle programs with the following code to create a “pen”:

import turtle

myPen = turtle.Turtle()

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:

Square

import turtle

myPen=turtle.Turtle()

myPen.color("red")

myPen.forward(200)

myPen.left(90)

myPen.forward(200)

myPen.left(90)

myPen.forward(200)

myPen.left(90)

myPen.forward(200)

myPen.left(90)

Cross (+)

import turtle

myPen = turtle.Turtle()

myPen.width(3)

myPen.color("red")

myPen.fill("red")

myPen.begin\_fill()

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.right(90)

myPen.forward(75)

myPen.right(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.right(90)

myPen.forward(75)

myPen.right(90)

myPen.end\_fill()

myPen.color("green")

myPen.fill("green")

myPen.begin\_fill()

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.right(90)

myPen.forward(75)

myPen.right(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.right(90)

myPen.forward(75)

myPen.right(90)

myPen.end\_fill()

Cross (x)

import turtle

myPen=turtle.Turtle()

myPen.width(5)

myPen.color("blue")

myPen.left(45)

myPen.forward(200)

myPen.up()

myPen.left(135)

myPen.forward(150)

myPen.down()

myPen.color("green")

myPen.left(135)

myPen.forward(201)

**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 moves the turtle to another spot through the use of coordinates “x” and “y”. Thi method can be used to move the shapes to different places. Here is an example of this type of code:

import turtle

myPen=turtle.Turtle()

for List in [1,2,3,4,5,6,7]:

myPen.goto(List\*100,0)

myPen.color("blue")

myPen.circle(30)

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

myPen.fill()

myPen.begin\_fill()

myPen.end\_fill()

turtle.reset()

turtle.bye()

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

import turtle

myPen = turtle.Turtle()

def iPentagon(sideSize):

myPen.width(sideSize)

myPen.color("turquoise")

myPen.forward(sideSize)

myPen.color("red")

myPen.forward(sideSize)

myPen.color("blue")

myPen.forward(sideSize)

myPen.color("yellow")

myPen.forward(sideSize)

myPen.color("green")

myPen.forward(sideSize)

myPen.color("teal")

myPen.circle(sideSize)

myPen.right(75)

iPentagon(10)

iPentagon(10)

iPentagon(10)

iPentagon(10)

iPentagon(10)

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.

import turtle

myPen=turtle.Turtle()

def drawSquare(sideSize):

myPen.forward(sideSize)

myPen.right(90)

myPen.forward(sideSize)

myPen.right(90)

myPen.forward(sideSize)

myPen.right(90)

myPen.forward(sideSize)

myPen.right(90)

drawSquare(10)

drawSquare(20)

drawSquare(30)

drawSquare(40)

drawSquare(50)

drawSquare(60)

drawSquare(70)

drawSquare(80)

drawSquare(90)

drawSquare(100)

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

for sideSize in [10,20,30,40,50,60,70,80,90,100]

**Level 3: Defining a Function**

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

“def” keyword creates a new user-defined function so that the function is performed when only the given name of the function has been inputted.

* 1. Explain any special rules regarding the function name

The first letter of the function name should start with a small letter. For example,

def iBoard

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

Parameters pass their functions a value. For example,

def iBoard (sideSize)

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

The colon”:” should be placed after the parentheses. For example,

def iBoard (sideSize):

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

The next few lines that make up the function must be indented. The body statement should be either concluded with “print” or “return”

* 1. Explain the “return” statement

The “return” has about the same function as “print”. But this statement does not use brackets while the “print” statement does. Here is an example:

def double(number):

return number \* 2

def double(number):

print (number \* 2)

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

def hello\_me (myname):

print("hello", myname)

hello\_me("Muktika")

1. Convert your basic pattern (from Level 2 above) into a function
2. The function name should be “my\_pattern”
3. The parameters should be the x and y starting position for your pattern
4. Your function does not need to use the “return” statement

import turtle

myPen = turtle.Turtle()

def inShape (my\_pattern):

myPen.goto(my\_pattern\*10,12)

myPen.right(90)

inShape(10)

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.

import turtle

myPen=turtle.Turtle()

for List in [1,2,3,4,5,6,7]:

myPen.goto(List\*100,0)

myPen.color("blue")

myPen.circle(30)

1. Provide a listing of your function definition and repeating pattern loop below.

Function Definition

def inShape (my\_pattern):

myPen.goto(my\_pattern\*10,12)

myPen.right(90)

Repeated pattern loop

for List in [1,2,3,4,5,6,7]:

myPen.goto(List\*100,0)

myPen.color("blue")

myPen.circle(30)