**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.forward(100)

myPen.right(90)

myPen.forward(100)

myPen.right(90)

myPen.forward(100)

myPen.right(90)

myPen.forward(100)

Circle in Square

import turtle

myPen = turtle.Turtle()

myPen.color("red")

myPen.circle(100)

myPen.color("Blue")

myPen.forward(100)

myPen.left(90)

myPen.forward(200)

myPen.left(90)

myPen.forward(200)

myPen.left(90)

myPen.forward(200)

myPen.left(90)

myPen.forward(100)

Cross

import turtle

myPen = turtle.Turtle()

myPen.right

myPen.forward(100)

myPen.right(90)

myPen.forward(50)

myPen.right(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.right(90)

myPen.forward(50)

myPen.right(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.right(90)

myPen.forward(50)

myPen.right(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.right(90)

myPen.forward(50)

myPen.right(90)

myPen.forward(100)

**Level 2: Using a Loop**

**1. Google the keywords “Python Turtle Methods”.**

**a. Explain how the “goto” method works and how you could use it when drawing repeated shapes.**

“goto” moves the turtle to another location given which can make it easier to draw several shapes.

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

Other methods such as “fillcolor”,”heading”,”position”,”dot”,”stamp”,etc weren’t mentioned.

**2. Create a repeating pattern on your screen. The pattern must meet the following requirements:**

**a. The basic pattern must be made up of several individual Turtle methods (e.g. changes of colour, changes of direction, size, motion, etc.)**

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

import turtle

Mypen=turtle.Turtle()

Mypen.goto(12,3)

Mypen.width(3.14195)

for repeat in "123":

Mypen.up()

Mypen.forward(100)

Mypen.down()

Mypen.color("blue")

Mypen.circle(100)

**3. Use a Python Loop to create your repeating pattern**

**a. The Loop may be a Counted Loop or a Conditional Loop**

**b. The indented block of code for the loop should be your basic pattern.**

**4. Provide a listing of your repeating pattern loop below.**

import turtle

Mypen=turtle.Turtle()

Mypen.goto(12,3)

Mypen.width(3.14195)

def my\_pattern (x,y):

for repeat in "123":

Mypen.up()

Mypen.forward(100)

Mypen.down()

Mypen.color("blue")

Mypen.circle(100)

my\_pattern(32,23)

**Level 3: Defining a Function**

1. Google the keywords “Python Function Syntax”.

a. Explain what the “def” keyword does

Marks the start of a function header.

b. Explain any special rules regarding the function name

A colon needs to be placed at the end of a function header for it to work.

c. Explain what the parameters (or arguments) do

Parameters (arguments) through which we can pass values to any function. These are optional in Python.

d. Where should the colon “:” be placed

At the end of the function’s header.

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

You have to use “def {command of your choice}

{what you want}

One or more valid statements that have the same indentational level.

f. Explain the “return” statement

The return statement is used to exit a function and go back to the placed where it was called. For example, you may want to go to the beginning if the program is set that way.

2. Provide an example of a simple function that uses one or more parameters.

def my\_function(color=”Black”):

print(“My favourite colour is” + color)

my\_function(“Blue”)

my\_function(“red”)

my\_fuction()

my\_function(“green”)

a. Write the function definition below

The function tells the program to print certain words or letters in a line. For example, you may want to have something different printed with the print command.

b. Write some code to call the function below

def my\_function(color=”Black”)

print(“My favourite colour is”+color)

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

4The function name should be “my\_pattern”

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

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

def my\_pattern(color=” Red”)

print(color+”is a good colour”)

my\_pattern(“blue”)

my\_pattern()

my\_pattern(“green”)

my\_pattern(“black”)

7. Use a your basic pattern function and a Python Loop to create your repeating pattern

a. The Loop may be a Counted Loop or a Conditional Loop

b. Your function should be called from within the loop.

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

for hi in “123456789”:

def my\_pattern(color=” Red”)

print(color+”is a good colour”)

my\_pattern(“blue”)

my\_pattern()

my\_pattern(“green”)

my\_pattern(“black”)