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

Done

1. 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(80)

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:

myPen.forward(60)

myPen.left(90)

myPen.forward(60)

myPen.left(90)

myPen.forward(60)

myPen.left(90)

myPen.forward(60)

Plus:

myPen.forward(60)

myPen.left(90)

myPen.forward(30)

myPen.left(90)

myPen.forward(60)

myPen.right(90)

myPen.forward(60)

myPen.left(90)

myPen.forward(30)

myPen.left(90)

myPen.forward(60)

myPen.right(90)

myPen.forward(60)

myPen.left(90)

myPen.forward(30)

myPen.left(90)

myPen.forward(60)

myPen.right(90)

myPen.forward(60)

myPen.left(90)

myPen.forward(30)

myPen.left(90)

myPen.forward(60)

Square with circle in it:

myPen.color("red")

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.left(90)

myPen.forward(100)

myPen.color("blue")

myPen.up()

myPen.goto(0,50)

myPen.down()

for I in range(10):

myPen.circle(50)

myPen.color("blue")

myPen.circle(50)

**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.

Goto is a way of moving your pen to a certain place without having to go left and right and turning it into a longer process.

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

-heading None Returns the current heading

-position None Returns the current position

-begin\_fill None Remember the starting point for a filled polygon

-end\_fill None Close the polygon and fill with the current fill color

-dot None Leave a dot at the current position

stamp None Leaves an impression of a turtle shape at the current location

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. import turtle
3. m = turtle.Turtle()
4. m.speed(0)
5. for i in range(100) :
6. m.begin\_fill()
7. m.fillcolor("orange")
8. m.circle(100)
9. m.end\_fill()
10. m.up()
11. m.forward(20)
12. m.left(90)
13. m.forward(25)
14. m.right(90)
15. m.forward(30)
16. m.down()
17. m.begin\_fill()
18. m.fillcolor("orange")
19. m.circle(130)
20. m.end\_fill()
21. m.begin\_fill()
22. m.fillcolor("cyan")
23. m.circle(100)
24. m.end\_fill()
25. m.up()
26. m.forward(35)
27. m.right(90)
28. m.forward(40)
29. m.left(90)
30. m.forward(45)
31. m.down()
32. m.begin\_fill()
33. m.fillcolor("cyan")
34. m.circle(130)
35. m.end\_fill()
36. m.begin\_fill()
37. m.fillcolor("orange")
38. m.circle(100)
39. m.end\_fill()
40. m.up()
41. m.forward(50)
42. m.left(90)
43. m.forward(55)
44. m.right(90)
45. m.forward(60)
46. m.down()
47. m.begin\_fill()
48. m.fillcolor("orange")
49. m.circle(130)
50. m.end\_fill()
51. m.begin\_fill()
52. m.fillcolor("cyan")
53. m.circle(100)
54. m.end\_fill()
55. m.up()
56. m.forward(65)
57. m.right(90)
58. m.forward(70)
59. m.left(90)
60. m.forward(75)
61. m.down()
62. m.begin\_fill()
63. m.fillcolor("cyan")
64. m.circle(130)
65. m.end\_fill()
66. for e in range(100) : m.begin\_fill()
67. m.fillcolor("orange")
68. m.circle(100)
69. m.end\_fill()
70. m.up()
71. m.right(20)
72. m.forward(80)
73. m.left(90)
74. m.forward(85)
75. m.right(90)
76. m.forward(90)
77. m.down()
78. m.begin\_fill()
79. m.fillcolor("orange")
80. m.circle(130)
81. m.end\_fill()
82. m.begin\_fill()
83. m.fillcolor("cyan")
84. m.circle(100)
85. m.end\_fill()
86. m.up()
87. m.forward(95)
88. m.right(90)
89. m.forward(100)
90. m.left(90)
91. m.forward(105)
92. m.down()
93. m.begin\_fill()
94. m.fillcolor("cyan")
95. m.circle(130)
96. m.end\_fill()
    1. The basic pattern must be repeated several times with a shift in starting position each time.
97. import turtle
98. m = turtle.Turtle()
99. m.speed(0)
100. for i in range(100) :
101. m.begin\_fill()
102. m.fillcolor("orange")
103. m.circle(100)
104. m.end\_fill()
105. m.up()
106. m.forward(20)
107. m.left(90)
108. m.forward(25)
109. m.right(90)
110. m.forward(30)
111. m.down()
112. m.begin\_fill()
113. m.fillcolor("orange")
114. m.circle(130)
115. m.end\_fill()
116. m.begin\_fill()
117. m.fillcolor("cyan")
118. m.circle(100)
119. m.end\_fill()
120. m.up()
121. m.forward(35)
122. m.right(90)
123. m.forward(40)
124. m.left(90)
125. m.forward(45)
126. m.down()
127. m.begin\_fill()
128. m.fillcolor("cyan")
129. m.circle(130)
130. m.end\_fill()
131. m.begin\_fill()
132. m.fillcolor("orange")
133. m.circle(100)
134. m.end\_fill()
135. m.up()
136. m.forward(50)
137. m.left(90)
138. m.forward(55)
139. m.right(90)
140. m.forward(60)
141. m.down()
142. m.begin\_fill()
143. m.fillcolor("orange")
144. m.circle(130)
145. m.end\_fill()
146. m.begin\_fill()
147. m.fillcolor("cyan")
148. m.circle(100)
149. m.end\_fill()
150. m.up()
151. m.forward(65)
152. m.right(90)
153. m.forward(70)
154. m.left(90)
155. m.forward(75)
156. m.down()
157. m.begin\_fill()
158. m.fillcolor("cyan")
159. m.circle(130)
160. m.end\_fill()
161. for e in range(100) :       m.begin\_fill()
162. m.fillcolor("orange")
163. m.circle(100)
164. m.end\_fill()
165. m.up()
166. m.right(20)
167. m.forward(80)
168. m.left(90)
169. m.forward(85)
170. m.right(90)
171. m.forward(90)
172. m.down()
173. m.begin\_fill()
174. m.fillcolor("orange")
175. m.circle(130)
176. m.end\_fill()
177. m.begin\_fill()
178. m.fillcolor("cyan")
179. m.circle(100)
180. m.end\_fill()
181. m.up()
182. m.forward(95)
183. m.right(90)
184. m.forward(100)
185. m.left(90)
186. m.forward(105)
187. m.down()
188. m.begin\_fill()
189. m.fillcolor("cyan")
190. m.circle(130)
191. m.end\_fill()
192. 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.
193. Provide a listing of your repeating pattern loop below.

import turtle

m = turtle.Turtle()

m.speed(0)

for i in range(100) :

m.begin\_fill()

m.fillcolor("orange")

m.circle(100)

m.end\_fill()

m.up()

m.forward(20)

m.left(90)

m.forward(25)

m.right(90)

m.forward(30)

m.down()

m.begin\_fill()

m.fillcolor("orange")

m.circle(130)

m.end\_fill()

**Level 3: Defining a Function**

Convince and flexibility

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

it a way of putting certain info in one place so when you have to change one certain thing all over the place in your work, you could just change it in one place instead of have to go to every single one of them and changing them

* 1. Explain any special rules regarding the function name

“Function blocks begin with the keyword def followed by the function name and parentheses ( ( ) ). Any input parameters or arguments should be placed within these parentheses. You can also define parameters inside these parentheses.”

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

“A parameter is a variable in a method definition. When a method is called, the arguments are the data you pass into the method's parameters.”

”Parameter is variable in the declaration of function. Argument is the actual value of this variable that gets passed to function.”

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

“The code block within every function starts with a colon (:) and is indented.”

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

The statement is the function body for example the if I write say\_hello as my statement it would automatically print hello

def say\_hello(myname):

print('Hello', myname)

* 1. Explain the “return” statement

The return statement is going to be what’s in the brackets

1. Provide an example of a simple function that uses one or more parameters.
   1. Write the function definition below

say\_hello("Manvir")

Hello Manvir

* 1. Write some code to call the function below

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

import turtle

m = turtle.Turtle()

m.speed(0)

def my\_pattern() :

m.begin\_fill()

m.fillcolor("orange")

m.circle(100)

m.end\_fill()

m.up()

m.forward(20)

m.left(90)

m.forward(25)

m.right(90)

m.forward(30)

m.down()

m.begin\_fill()

m.fillcolor("orange")

m.circle(130)

m.end\_fill()

#print ("for i in

for i in range(10) :

my\_pattern()

1. The function name should be “my\_pattern”

import turtle

m = turtle.Turtle()

m.speed(0)

def my\_pattern() :

m.begin\_fill()

m.fillcolor("orange")

m.circle(100)

m.end\_fill()

m.up()

m.forward(20)

m.left(90)

m.forward(25)

m.right(90)

m.forward(30)

m.down()

m.begin\_fill()

m.fillcolor("orange")

m.circle(130)

m.end\_fill()

#print ("for i in

for i in range(10) :

my\_pattern()

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

import turtle

m = turtle.Turtle()

m.speed(0)

def my\_pattern(goto 10,12) :

m.begin\_fill()

m.fillcolor("orange")

m.circle(100)

m.end\_fill()

m.up()

m.forward(20)

m.left(90)

m.forward(25)

m.right(90)

m.forward(30)

m.down()

m.begin\_fill()

m.fillcolor("orange")

m.circle(130)

m.end\_fill()

#print ("for i in

for i in range(10) :

my\_pattern()

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

import turtle

m = turtle.Turtle()

m.speed(0)

def my\_pattern() :

m.begin\_fill()

m.fillcolor("orange")

m.circle(100)

m.end\_fill()

m.up()

m.forward(20)

m.left(90)

m.forward(25)

m.right(90)

m.forward(30)

m.down()

m.begin\_fill()

m.fillcolor("orange")

m.circle(130)

m.end\_fill()

#print ("for i in

for i in range(10) :

my\_pattern()

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

import turtle

m = turtle.Turtle()

m.speed(0)

def my\_pattern() :

m.begin\_fill()

m.fillcolor("orange")

m.circle(100)

m.end\_fill()

m.up()

m.forward(20)

m.left(90)

m.forward(25)

m.right(90)

m.forward(30)

m.down()

m.begin\_fill()

m.fillcolor("orange")

m.circle(130)

m.end\_fill()

#print ("for i in

for i in range(10) :

my\_pattern()

* 1. Your function should be called from within the loop.

import turtle

m = turtle.Turtle()

m.speed(0)

def my\_pattern() :

m.begin\_fill()

m.fillcolor("orange")

m.circle(100)

m.end\_fill()

m.up()

m.forward(20)

m.left(90)

m.forward(25)

m.right(90)

m.forward(30)

m.down()

m.begin\_fill()

m.fillcolor("orange")

m.circle(130)

m.end\_fill()

#print ("for i in

for i in range(10) :

my\_pattern()

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

import turtle

m = turtle.Turtle()

m.speed(0)

def my\_pattern() :

m.begin\_fill()

m.fillcolor("orange")

m.circle(100)

m.end\_fill()

m.up()

m.forward(20)

m.left(90)

m.forward(25)

m.right(90)

m.forward(30)

m.down()

m.begin\_fill()

m.fillcolor("orange")

m.circle(130)

m.end\_fill()

#print ("for i in

for i in range(10) :

my\_pattern()