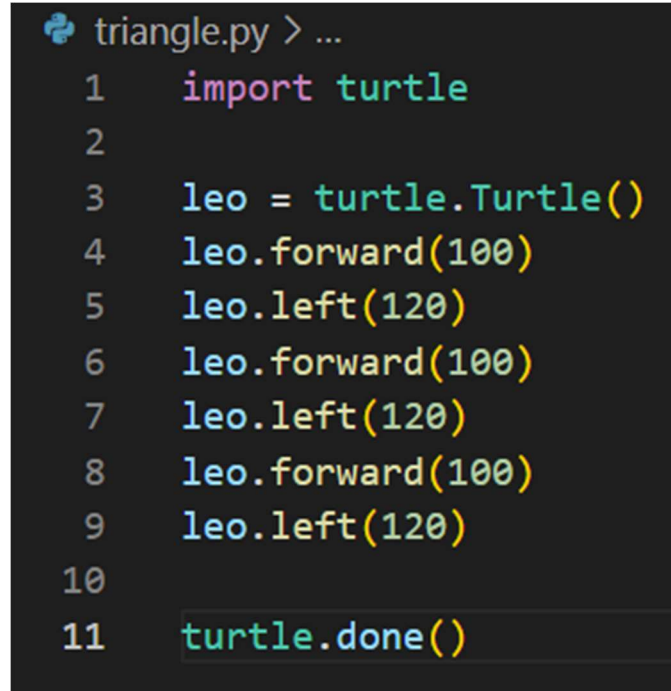


ICS 140 Computational Thinking with Programming

Lab 1

For this lab, we will explore the turtle library to write code that uses commands to create drawings. We'll introduce the various commands and use them to build some simple shapes. Let's look at the example below and examine some of the commands.

Example 1

A screenshot of a code editor window titled 'triangle.py > ...'. The code is written in Python and uses the turtle library to draw a triangle. The code consists of 11 lines: line 1 imports the turtle module; line 2 is a blank line; line 3 creates a turtle object named 'leo'; lines 4, 5, 6, 7, 8, and 9 form a loop of forward and left commands to draw the triangle's sides; line 10 is a blank line; and line 11 calls 'turtle.done()' to finish the program. The code is color-coded: 'import' is pink, 'turtle' is green, 'leo' is blue, and the function names and arguments are yellow.

```
1  import turtle
2
3  leo = turtle.Turtle()
4  leo.forward(100)
5  leo.left(120)
6  leo.forward(100)
7  leo.left(120)
8  leo.forward(100)
9  leo.left(120)
10
11 turtle.done()
```

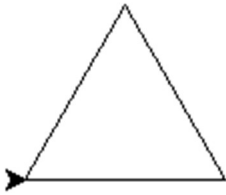
On line 1, you see the import command. This command is needed whenever you are going to use the turtle library. It tells python to load the turtle library makes the other commands available to you.

On line 3, you there is a command where a turtle object called “leo” is created. You can name your turtle whatever you want. All of your subsequent commands you use to move your turtle will reference the name you have given it.

On lines 4 -9, there is a pattern of forward() and left() commands. The forward command causes the turtle to move in the direction it is facing the number of pixels provided in the parenthesis. The left command causes the turtle to rotate the direction it's facing based on the number of degrees provided in the parenthesis.

On line 11, there is a “turtle.done()” statement. This needs to be at the end of any turtle program. It keeps the window open so that you can see the drawing.

Running the program displays a window like this:

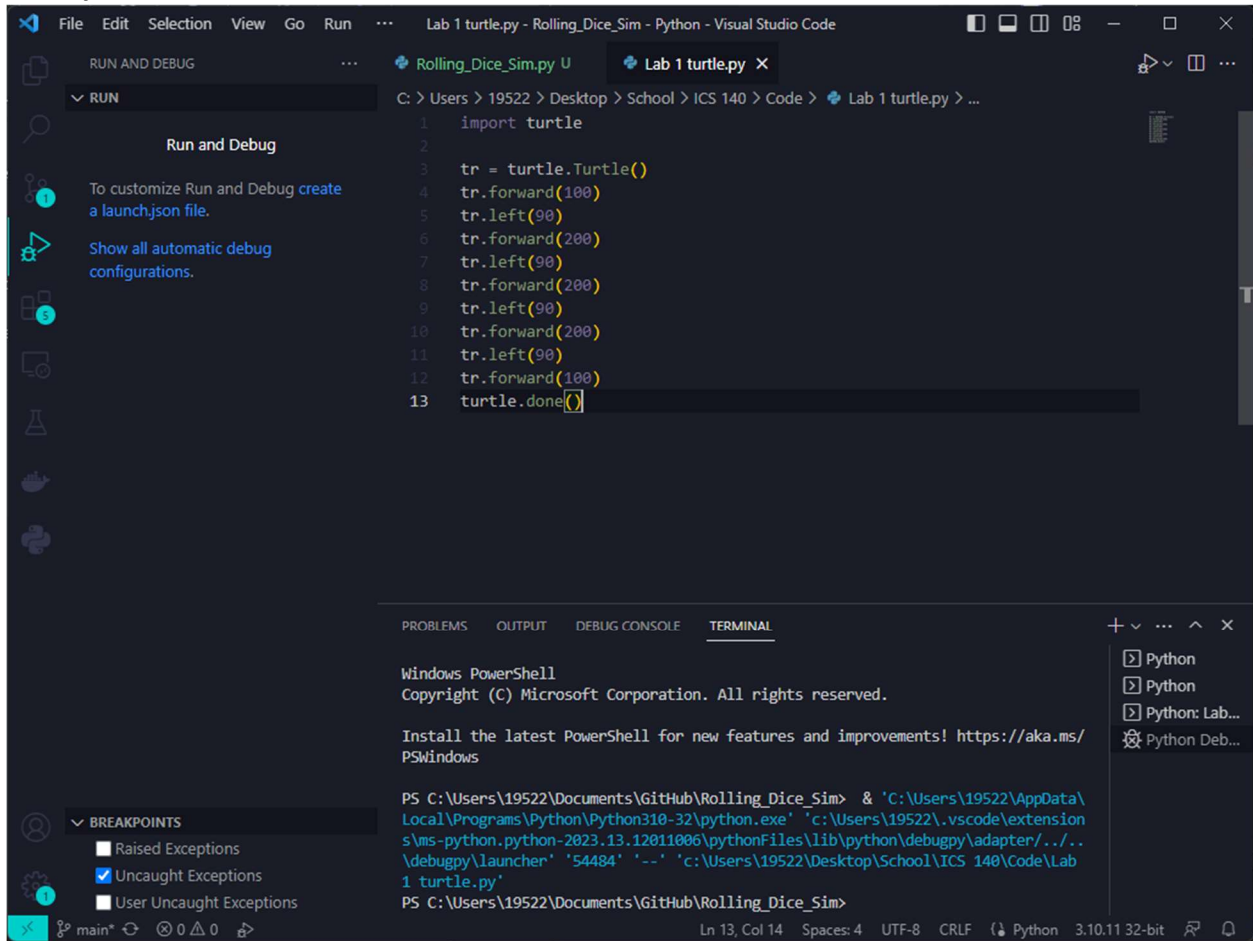


Coding Challenge 1

Using the code from above as an example, try creating code that will draw a square. Remember the following:

- You must have the import turtle command
- You must create a named turtle object (name it whatever you like)
- Use the forward() command to make your turtle move forward
- Use the left() command to make your turtle change the direction it is facing.
- After all of the commands are given to make the drawing, use “turtle.done()” to keep the image displayed

Paste your code below:



Now that we have created a drawing with straight lines, let's explore some other turtle commands.

Example 2

```
circle.py > ...  
1  import turtle  
2  
3  donny = turtle.Turtle()  
4  donny.circle(100)  
5  donny.penup()  
6  donny.setpos(0,-100)  
7  donny.pendown()  
8  donny.circle(200)  
9  donny.penup()  
10 donny.setpos(0,-200)  
11 donny.pendown()  
12 donny.circle(300)  
13  
14 turtle.done()
```

On line 1, we again have our import command.

On line 3, we again have created a turtle object, this time it is named "donny"

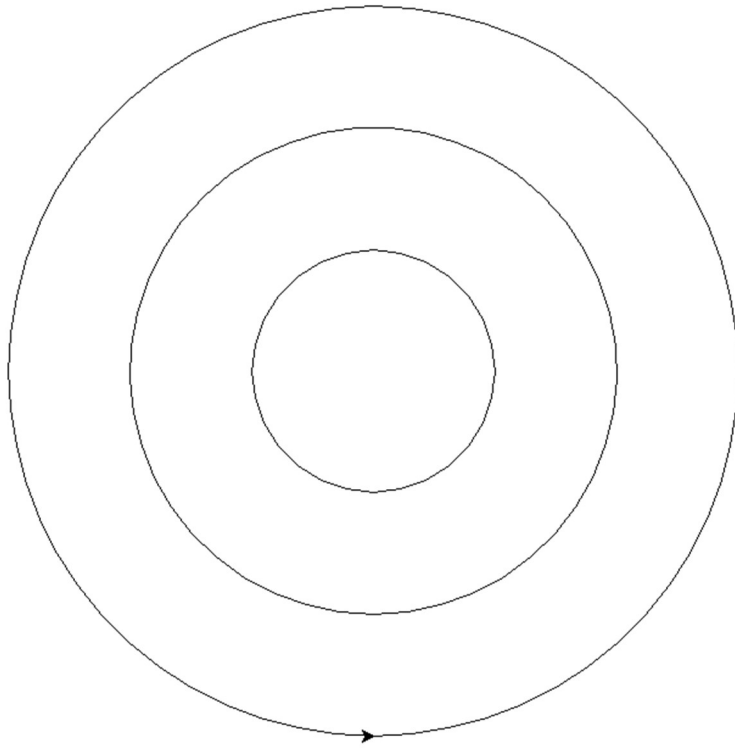
On line 4, we have an example of the circle() command. This command makes donny draw a circle with a radius based on the pixels provided in the paranthesis.

On line 5, we have an example of the penup() command. This command causes donny to stop drawing lines when moving similar to lifting your pen off of a page when writing.

On line 6, we have an example of setpos() command which moves the turtle to a set of coordinates. The turtle background uses a coordinate map using an x,y axis like in algebra with (0,0) at the center. In this case, we stay in the center of the x axis and move lower to before drawing our next circle.

On line 7, we have an example of the pendown() command. This command causes donny to start drawing again when moving.

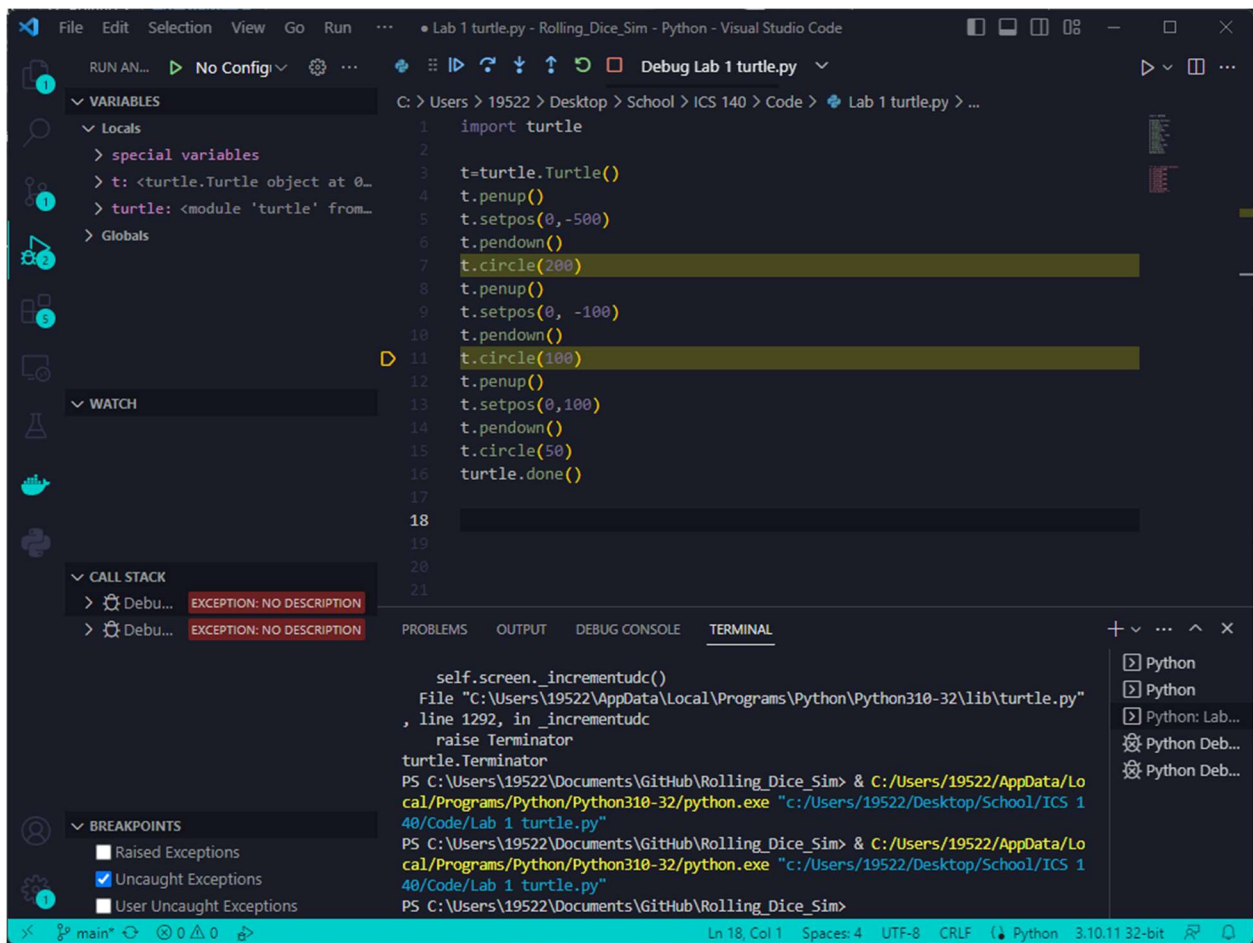
These commands are repeated in a way to create the following drawing:



Coding Challenge 2

Using similar commands to example 2, see if you can draw 3 circles stacked on top of each other like a snowman.

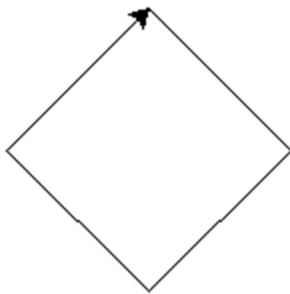
Paste the python code below:



Example 3

```
💻 diamond.py > ...  
1  import turtle  
2  
3  mike = turtle.Turtle()  
4  mike.right(45)  
5  mike.forward(100)  
6  mike.right(90)  
7  mike.forward(100)  
8  mike.right(90)  
9  mike.forward(100)  
10 mike.right(90)  
11 mike.forward(100)  
12  
13 turtle.done()
```

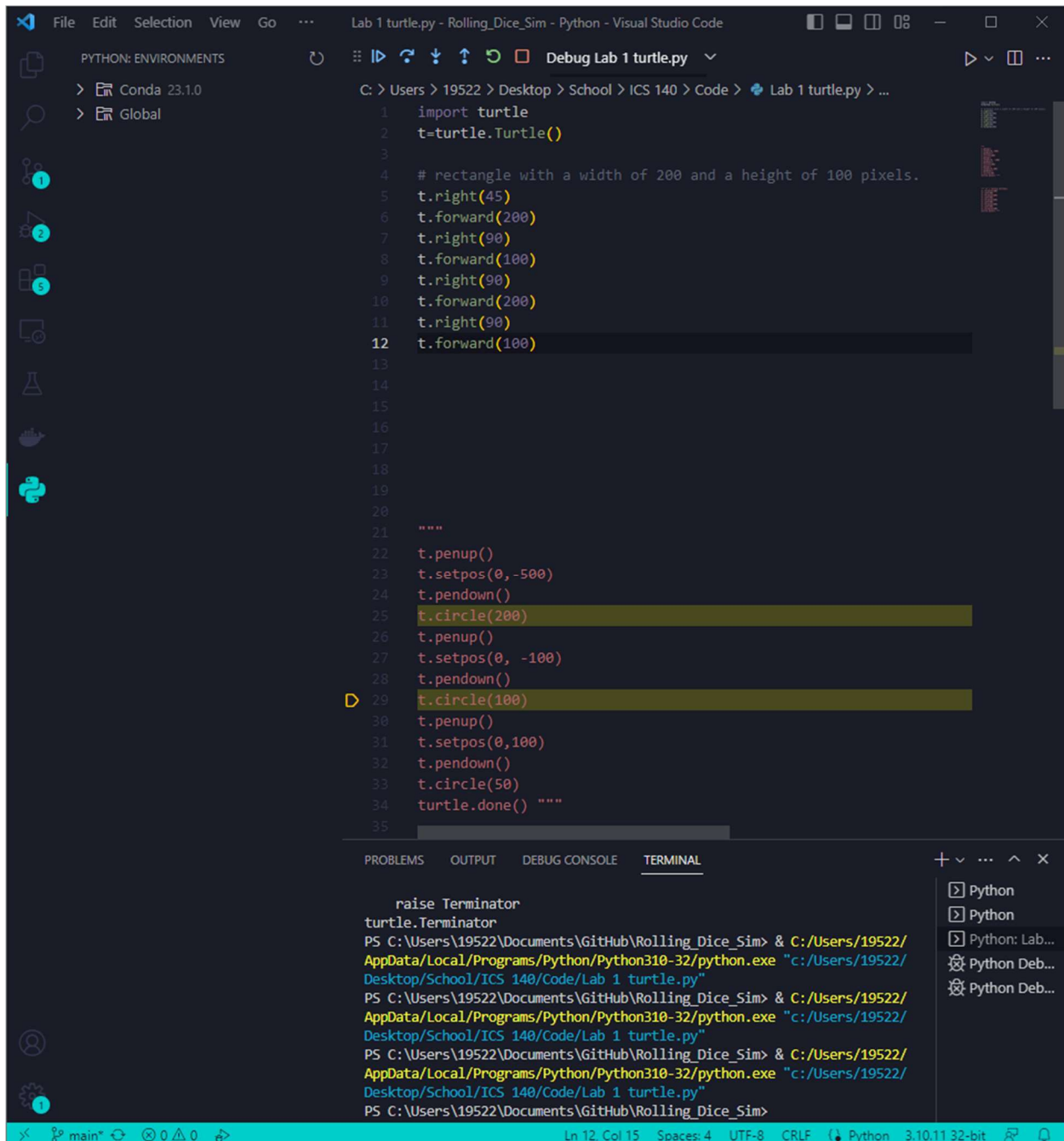
For this last example, we see the same import statement, naming statement and done statements as the other examples. Notice on line 4 that the mike turtle is rotated before drawing anything causing the shape to be drawn at an angle. The code results in a shape like this:



Coding Challenge 3

Using similar commands to the example above, draw a rectangle with a width of 200 and a height of 100 pixels.

Paste the python code below.



```
1 import turtle
2 t=turtle.Turtle()
3
4 # rectangle with a width of 200 and a height of 100 pixels.
5 t.right(45)
6 t.forward(200)
7 t.right(90)
8 t.forward(100)
9 t.right(90)
10 t.forward(200)
11 t.right(90)
12 t.forward(100)
13
14
15
16
17
18
19
20
21 """
22 t.penup()
23 t.setpos(0,-500)
24 t.pendown()
25 t.circle(200)
26 t.penup()
27 t.setpos(0, -100)
28 t.pendown()
29 t.circle(100)
30 t.penup()
31 t.setpos(0,100)
32 t.pendown()
33 t.circle(50)
34 turtle.done() """
35
```

Problems OUTPUT DEBUG CONSOLE TERMINAL

```
raise Terminator
turtle.Terminator
PS C:\Users\19522\Documents\GitHub\Rolling_Dice_Sim> & C:/Users/19522/AppData/Local/Programs/Python/Python310-32/python.exe "c:/Users/19522/Desktop/School/ICS 140/Code/Lab 1 turtle.py"
PS C:\Users\19522\Documents\GitHub\Rolling_Dice_Sim> & C:/Users/19522/AppData/Local/Programs/Python/Python310-32/python.exe "c:/Users/19522/Desktop/School/ICS 140/Code/Lab 1 turtle.py"
PS C:\Users\19522\Documents\GitHub\Rolling_Dice_Sim> & C:/Users/19522/AppData/Local/Programs/Python/Python310-32/python.exe "c:/Users/19522/Desktop/School/ICS 140/Code/Lab 1 turtle.py"
PS C:\Users\19522\Documents\GitHub\Rolling_Dice_Sim>
```

Ln 12, Col 15 Spaces: 4 UTF-8 CRLF Python 3.10.11 32-bit

