Lab 3:

Booleans, Strings, Random Numbers, Loops, Input function

Due: Mar18 (Note that this is a 2-week lab)

This lab can be done using paired partners. Both students should turn in the lab, and both labs must have both partner's names on it.

Create a file called lab3.py. Include as comments at the very top of the file: your name, your partner's name, class time, TA's name(s), and lab due date. Then include the following comments and functions:

Note: Include the comments in your lab3.py file. This will make the lab easier to read and grade. Comments should include the number and type of the input parameters, the type of the output parameter, and a clear description of exactly what the function does. When possible, the function should include test cases. Comments should be written BEFORE the function is written.

Note 2: Test cases for functions that generate random numbers are somewhat difficult. If possible for those cases, give a sample output, i.e., an example of what you might get if a particular random number or set of random numbers are generated. Note that the rest of your comments don't change at all. All functions must have comments. In functions where I have included the comments, copy and paste them into your lab code.

Contents

Part 1: 2
1. Concatenate (2 pts) 2
2. StringMult (2 pts) 2
3.Concatenate (10) 2
a)DayOfWeek (2 pts) 2
b) Month (2 pts) 2
c) Year (2 pts) 2
d) Date (4 pts) 2
4. Calculator(4 pts) 2
Random Number Generation 2
5) BlackJack (5 pts) 3
6(a)checkvalues (5pts) 3
6(b) Lottery (4 pts) 3
Loops: 3
Part 2: Turtle: 3
7 (Circles) (3 pts) 3
8(Squares: (4 pts) 4
Spirograph/flower (11 pts) 4
9a. drawcircle: (4 pts) 5
9b Spirograph function: (7 pts) 5
10. JustLooping (5 pts) 6
Part 3:MORE LOOPS 6
11 Lottery(b)(8 pts). 6
12 MagicNumbers (9 pts): 7
13 DrawRow(6 pts): 7
b DrawGraph(7 pts): 7
Input function (Versus input parameters) 8
14 Rock/Paper/Scissors(14 pts) 8

Part 1:

1. Concatenate (2 pts) Write a function that takes as input parameters 2 strings. It outputs a string. The returned string is the two input strings concatenated together.

2. StringMult (2 pts) Write a function that takes as input 2 parameters: a string and an int. It should return 1 longer string – the string you input int times. So if your input parameters were "ha" and 4, the string returned should be "hahahaha"

3.Concatenate (10)

a)DayOfWeek (2 pts) Write a function that takes as an input parameter an integer between 1 and 7 (including 1 and 7). It should return the corresponding day of the week (e.g, if the input number is 1, the string returned should be "Sunday", if the input number is 4, the string returned should be "Wednesday", etc.

b) Month (2 pts)Write a function that takes as an input parameter an integer between 1 and 12 (including 1 and 12). It should return a string that corresponds to the month. So, for instance, if the input parameter is 2, the string returned should be "February", if the input parameter is 10, the string returned should be "October".

c) Year (2 pts)Write a function that takes as an input parameter a number between 0 and 99. It should return the STRING "20xx" where xx is the input parameter. So, for instance if the input parameter is 20, the returned string would be "2020". If the input parameter is 3, the returned string would be "2003", etc.

d) Date (4 pts) Write a function that takes 4 integer input parameters and returns the string that would correspond to the date of those 4 integers by using the functions a, b, and c. So, for instance, if the input parameters are 1, 3,2,20, the returned string should be, "Monday, March 2, 2020". If the input parameters are:4,8,4,21 the returned string should be"Wednesday, August 4, 2021"

4. Calculator(4 pts) Write a function that takes as input parameters 2 integers and a character. It then acts as a simple calculator as follows: if the character is p, it prints out that it is adding the 2 integers, and returns the resulting sum. If the character is an s, it prints out that it is subtracting the two integers and returns the resulting subtraction. If the character is an m, it prints out that it is multiplying the two integers and returns the product. If the character is a d, it prints that it is dividing the two characters and returns the result. So if your input parameters were, 5,3,m your results would look like:

Multiplying 5 and 3
15
If your input parameters were 2,7,s your results would look like:
Subtracting 7 from 2
-5

Random Number Generation

Python can generate random numbers. There's a whole library dedicated to generating random numbers. It's called random. So if you want to generate random numbers, you must first import the random number library by placing at the top of your lab2.py file:

```
from random import *
```

The above line should be the top line (under your comments) in your lab2.py file. Now you can generate random numbers between x and y using randrange(x,y)

For instance, if you wanted to generate a random number between 0 and 100 (not including 100) and store it in the variable randvar, you'd write the following:

```
randvar = randrange(0,100)
```

Remember, we do the right side first and place the value calculated by the right side into the variable on the left side. So if you use the above code to generate a random number, randrange will generate a number between 0 and 99 (inclusive). Let's say the random number generated is 42. Then randvar will hold the number 42.

To see what number is generated, you can always print it:

print (randvar)

(Note: when writing test cases for functions that use random numbers, you may need to give a range of answers that would be correct. So in this case, I might say:

- Input: 3->func-> 0…2 (inclusive)
- Input: 7-> func->0…6 (inclusive)
- Input: 100->func-> 0…99 (inclusive)

5) BlackJack (5 pts)

Write a function that takes no input parameters. It generates 2 random numbers between 1 and 11, including 11. If both numbers are 11, the resulting number that should be printed out is 12 (i.e., one of the 11s should be converted to 1). Otherwise the two random numbers should be added together and the result printed out. The function should then ask the user, "Do you want another card?" It should read the user's answer (using the input function). If the user says "Yes", another random number should be generated and added to the first two. That result should be printed out. If the resulting number is greater than 21, the function should print out, "You lose!" and a -1 should be returned from the function. Otherwise the user should be asked again, "Do you want another card?". The same process should be followed again. Again, if the result is greater than 21, "You lose shouldbe printed out and -1 should be returned. This process should be repeated one more time. If at any time the user says "No" (instead of "Yes"), the function should print, "Your current value is" and the current total, and that total should be returned.

6(a)checkvalues (5pts)

Write a function that takes as input parameters 4 integers. It returns a Boolean value. It checks to see if the second two parameters are the same as the first two parameters, regardless of order. In other words:

3,7,3,7 ->checkvalues ->True
7,4,4,7->checkvalues -> True
7,8,7,4->checkvalues -> False

6(b) Lottery (4 pts)

Write a function that takes as input parameters 2 integers. The function should generate 4 random number between the two input parameters. So, for instance, if the two integers are 1 and 9, the random numbers could be 4,1,5,8 or possibly 1,2,7,5 (but never 9)**.** It then calls the function checkvalues with the 4 random numbers. It does this 3 times. If even once checkvalues returns true, the function should return the string, "You won the mini lottery!" Otherwise it should return "Sorry, you did not win."

Loops:

NOTE: For those of you who have had programming before, this lab is to ensure sure you understand while loops. You must use while loops as opposed to any other type of loop for these problems. There should be no loops other than while loops in this lab.

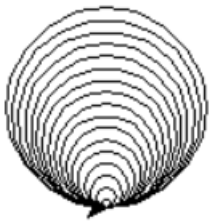
Write the following functions in Turtle using a while loop:

Part 2: Turtle:

7 (Circles) (3 pts)

Function Name: Circles
Input: 1 integer: the radius
Output: none

This function loops until the radius is 1 or less. Each time it draws a circle with the radius, and then loops with a radius 3 smaller than the previous radius. You should get something like this:



Call the function with different numbers to get different circles (remember to reposition the turtle before you call the function again so the circles don't just end up on top of each other).

8(Squares: (4 pts)

Function Name: Squares
Input: 3 integers: the length of the side, and the starting x and y coordinate
Output: none
This function loops until the length of the side is 1 or less. Each time it draws a square centered within the previous square, and then loops with a length 8 smaller than the previous side length. You should get something like this:

Note: for this you will need turtle.penup() to lift the turtle pen off the screen, then, of course, turtle.pendown() to put it down on the screen, and you'll need turtle.goto(x,y) which will move the pen to the x y coordinates.
Make sure you test this function 3 times with 3 different sets of input parameters.

Spirograph/flower (11 pts)

To do this problem, you need to know a few things about turtle's color. Color is represented by the amount of red, the amount of green, and the amount of blue (rgb values). Each value is represented by an amount between 0 and 1. To set the color, you use the command:

turtle.color(r,g,b) where r is a double between 0 and 1, g is a double between 0 and 1, and b is a double between 0 and 1.

You can set turtle's pensize with an int, e.g.,

turtle.pensize(3)

equally, when the pen is at x,y coordinates and a circle is drawn, the x y coordinates default to the very bottom of the circle's circumference, not the center. In order to have the circle drawn with the starting location at the same x/y coordinates but at a different angle, you rotate the direction of the turtle using either turtle.left() or turtle.right(), and the degrees you want to rotate turtle's angle.

So for instance, if you wanted to draw 6 circles like a Spirograph, you'd rotate turtle.left(60) each time and then draw another circle.

Okay, now onto the drawcircles function:

9a. drawcircle: (4 pts)

This function draws one set of Spirograph circles. It takes 6 (yep, 6!) input parameters: one int representing the count, or number of times the loop needs to rotate, an int representing the size of the circle, an int representing the amount the turtle should be rotated each time, and then 3 doubles, all between 0 and 1, representing the amount of red, the amount of green, and the amount of blue in the color, respectively.

The function should return nothing using just this line at the very bottom of the function:

return

It should generate a **random number** between -.2 and .2 (random number between -20 and 20, divided by 100). And that number should be added to the amount of red. Repeat for green, and for blue. Then check to make sure that the red, green, and blue amounts are within range. If a value is < 0, it should be reset to .1. If it's > 1, it should be reset to .9.

The turtle's color should be set to the new red/green/blue values. And a circle should be drawn. Then the loop should continue.

So, if the drawcircle is called initially as:

Drawcircle(6,25,60,.5,.5,.5)

You should get something looking like this:

9b Spirograph function: (7 pts)

This function takes 3 parameters, all three are ints. The first is the count of the number of circle sets you'll create (the drawcircle function creates 1 circle set), and the second and third are the x/y coordinates of where the circles start from. (Note: these don't change throughout, but this function makes sure that we always start from those x/y coordinates in case the pen gets off by a pixel or two).

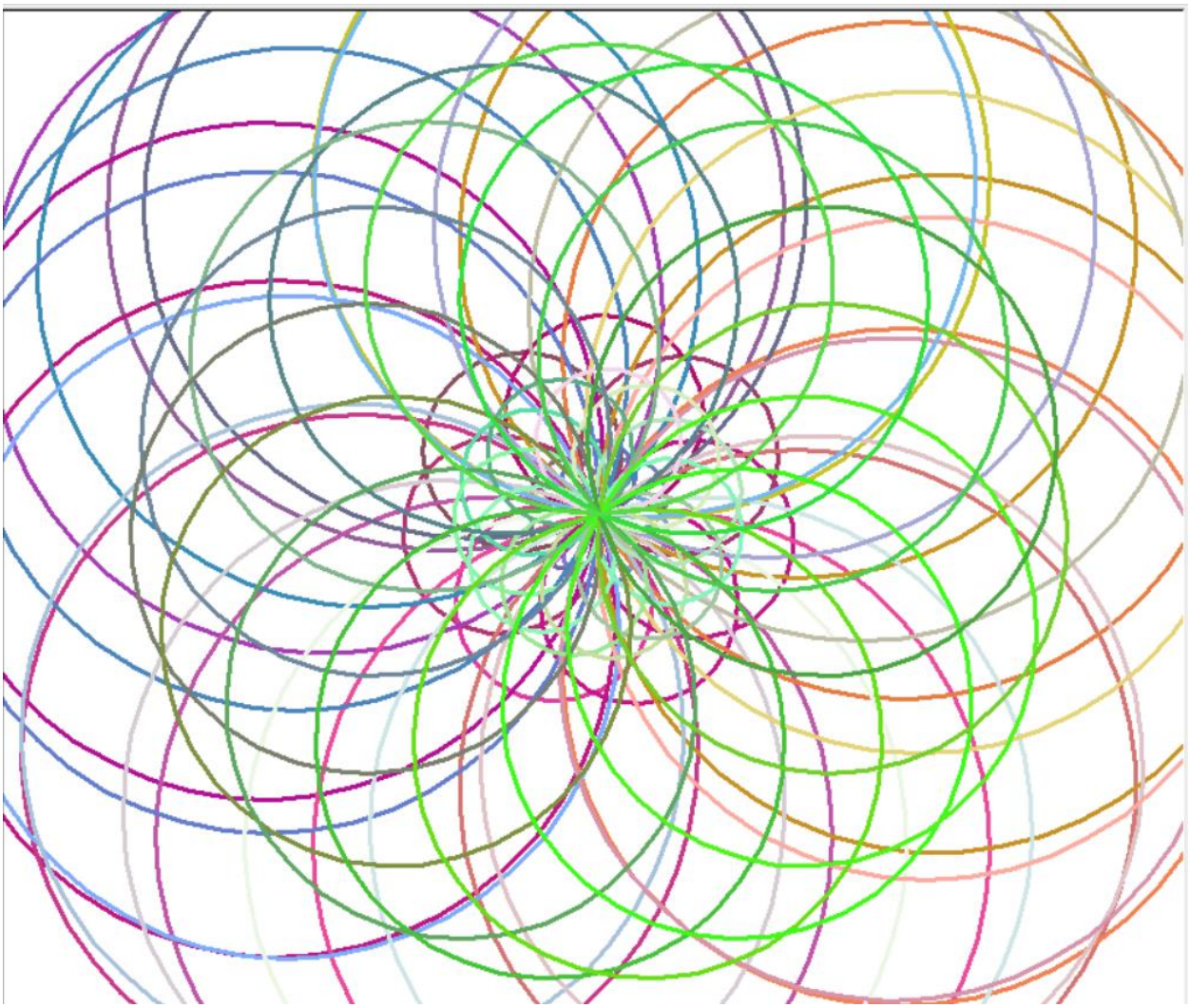
This function loops count times (the first parameter). If it does not stop, it goes to the x/y coordinate. Note: you will want to use the following command:

turtle.setheading(0)

to reset the direction of the turtle to its original direction

You will then randomly choose a number between 4 and 20 for how many circles should be generated for this circle set. Once you know the number of circles, you should calculate how much the rotation should be. Finally, you should randomly generate a number between 20 and 230, which will be the size of each circle. When you have that, you should call your drawcircle function (with initial color values of .5, .5,. .5). Then you should loop with the Spirograph function.

You might get something that looks like this:



10. JustLooping (5 pts)

In problem 4.2 under the turtle section in the last lab, you wrote a function that took as an input parameter a color, and drew an object. For this problem, you are going to use that function to draw your object all over the turtle window, in different colors. So the function you are now writing, which will loop, should have 1 parameter: the number of times the function still needs to loop. In the function, you will generate a random number between -250 and 250 for the x coordinate, and another random number between -250 and 250 for the y coordinate. You should then use turtle.penup() and turtle.goto(x,y) to have the pen go to the new x and y coordinate. Then call the function you wrote in 4.2 (you should copy and paste that function into this lab file, above this function you are writing now). Make sure you generate new random numbers for the color.

END OF TURTLE PROBLEMS:

##

Part 3:MORE LOOPS

11 Lottery(b)(8 pts). In problem 6a you wrote a function that took as input parameters 4 integers and checked to see whether the first two integers are the same as the second two integers. In this function you will be using that function to see how long it takes to generate two numbers that match. This function will generate 4 random numbers. It will call the function you wrote in 6a. As long as the function 6a returns false, it will continue

to generate 4 new random numbers and call the function. This function should have a loop count, and it should count how many times the function needs to generate random numbers before there is a match.

(Note: for the actual lottery, I think each number can be anything between 1 and 50. You can try that, but for testing purposes, your code might be running for a while. For testing purposes, you might want to limit the range to something between 1 and 10 (including or excluding 10 – your choice).

12 MagicNumbers (9 pts):

Write a function that takes as an input parameter an integer (make it something over 20). The function should print out all values that divide evenly into that number. It should also sum those numbers. If the sum of all the numbers that divide evenly into that number are equal to that number, you function should print, "Magic Number is" + the magic number. The function should return the Boolean value true. Otherwise it should print "Not a magic number: " + the number, and return false.

13 DrawRow(6 pts):

Write a function that takes an integer and a character as input parameters. It then draws a row of stars the length of that integer. It returns nothing. So, for instance, if the input parameters are 7 and '*', the printed result that would be drawn out would be:

****

If the input parameters were 5 and '^' the printed result would be:

^^^^^

Note: in python, when you say print("whatever") the print command automatically includes at the end of the line a line return, so that the next time you print, it will be on the next line. You can override that automatic new line by resetting the end to be something else within the print statement. So, if I did the following:

```
print("hi there")
print("it's a great day")
```

the results would be:

```
hi there
it's a great day.
```

However, if I did the following:

```
print("hi there", end = " ")
print("it's a great day")
```

The result would be:

```
hi there it's a great day
```

b DrawGraph(7 pts):

Write a function that takes as an input parameter an integer and a character. The function then generates that many random integers between 0 and 20 and calls the function DrawRow with that random integer and the character. After it has called that function, it prints out a line break (just a simple print() statement makes a line break).

Sso if the integer was 4, it would generate 4 random integers. If those random integers were 3,6,8,2, the results would look like:

***

******

****

**

Input function (Versus input parameters)

Python has something known as an input function, which is DIFFERENT from input parameters. Input parameters are the values that are passed into a function. The input function is a function that demands that the user type something in. So, for instance,

```
def f(x,y):    #x and y are the input parameters, and we put values into x and y outside of the function
        return(x + y)
print(f(3,7)  #this is putting values into the parameters.  It's where we put 3 into x and 7 into y
```

Whereas:

```
def f():
        x = input("what value, as a number, do you want to enter?")  #Here the string shows up in python's shell
                                                                     #window, and the user must type in a value.
                                                                     #The value will go into x
        y = input("Enter a second number")
        return(x+y)
print(f())  #Notice that no values are going in here.
```

In the second function, there are no input parameters. Instead, the function interacts with the user as it runs to get actual values for x and y.

14 Rock/Paper/Scissors(14 pts)

Copy the following code into Lab 3. Make sure you've downloaded stone.gif, paper.gif, and Scissors.gif from my web page. Place them in your Python folder on your computer. Then write the following recursive function where indicated.

```
import turtle
import tkinter
from random import *
tk = tkinter.Tk()
canvas = tkinter.Canvas(tk, width = 500, height = 300, bg="white", bd="5")
canvas.pack()

def display(you,computer):
    print("You: " + str(you) + " computer: " + str(computer))
    if you == 0:
        my_img=tkinter.PhotoImage(file = "stone.gif")
    elif you == 1:
        my_img=tkinter.PhotoImage(file = "paper.gif")
    elif you == 2:
        my_img=tkinter.PhotoImage(file = "Scissors.gif")
    canvas.create_image(70,60, anchor = tkinter.NW,image = my_img)
    if computer == 0:
        c_img=tkinter.PhotoImage(file = "stone.gif")
    elif computer == 1:
        c_img=tkinter.PhotoImage(file = "paper.gif")
    elif computer == 2:
        c_img=tkinter.PhotoImage(file = "Scissors.gif")
    canvas.create_image(270,60, anchor = tkinter.NW,image = c_img)
    tk.update()
```

```
# Looping function:
# Function name: RockPaperScissors
# input: 2 integers – the number of times played and the number of times won so far
# output: a string – “You won x out of 10 times”, where x is the number of times you
won
# This function should run 10 times.  Each time it should use the input function to
ask you, “choose 0 for rock,1 for paper, or 2 for scissors: “.  It should then
generate a random number between 0 and 3 (not including 3).  It should call the
function display() with your choice and the computer’s choice. It should then
calculate whether you won and, if so, increase the won count.  The winner is
determined as follows:

      If the user chose rock:
            If the computer generated rock, it’s a tie
            If the computer generated paper, the user loses
            If the computer generated scissors, the user wins
      If the user chose paper:
            If the computer generated rock, the user wins
            If the computer generated paper, it’s a tie
            If the computer generated scissors, the user loses
      If the user chose scissors:
            If the computer generated rock, the user loses
            If the computer generated paper, the user wins
            If the computer generated scissors, it’s a tie
Again, this function should recurse 10 times

#YOUR LOOPING FUNCTION GOES HERE!

def main():
    print(RockPaperScissors(0,0))

main()
```