Lab 2: Functions and Labs
June 21, 2022

Description: The following problems below are small examples of the things that we have learned in the last week. Whenever you finish one of the problems, raise your hand or call me to your computer so we can check the problems together. Using techniques and functions/methods we have learned in class, try to do them in as few lines of code as possible. If at any point you have questions, please raise your hand.

1. To write programs that solve more complex problems, we need code that will allow statements to be executed in more complex ways. For example, we might like to execute some statements under some conditions but not others, or we might like to execute some statements multiple times. These require our ingredients of conditionals and repetition. In this lab, we will explore programs that determine which statements should be executed based on the values of variables.

2. Create a new repl on repl.it named "Lab3". Copy and paste the code below into your repl.it file named "main.py". Enter in a few numbers and notice that the statement "That number is positive" only prints when a positive number is entered.

```
a. number = int(input("Enter a number: "))
b. if number >= 0:
c.         print("That number is positive.")
d. else:
e.         print("That number is negative.")
f. print("Nice Number!")
```

3. Now, copy and paste the following code below and see what it prints. While loops are helpful loops whenever we don't know when we want the loop to end. In this case, we could have a while loop that keeps accepting input from the user until they no longer want to input information.

```
a. count = 1
b. while count <= 5:
c.     print("Rhythm")
d.     Print("Blues")
e.     count = count + 1
f. Print("Done")
```

4. Change the condition in the while-loop above so that "Rhythm" and "Blues" is printed exactly 10 times.

5. Include an *if-statement* in your while loop so that "Rhythm" is printed when the variable **count** is an **even** number. Print "Blues" when the variable count is an **odd** number.

6. Once you get to this point, raise your hand so I can check your progress.

_____

7. Write a program called **squares()** that prints a list of the first nine positive integers and their squares. (Once again, be careful to include the colon at the end of the while statement, or Python will complain.) Your output should look like the following but does not have to be exact:

Number	*Square*
1	*1*
2	*4*
3	*9*
4	*16*
5	*25*
6	*36*
7	*49*
8	*64*
9	*81*

8. Once you get to this point, raise your hand so I can check your progress.

_____

9. Now, copy and paste the following code into your repl.it. Before you click **RUN** look and think about what the code will print out.

```
num_one=1
while num_one<10:
  num_two = 1
  while num_two<10:
    product = num_one * num_two
    print(product, end=" ")
    num_two += 1
  print()
  num_one += 1
```

   a. Why is numTwo reset to 1 at the beginning of the "outer" (top-most) loop?

   b. What would happen if this line was removed from the program?

   c. Test your prediction by "commenting" out the line. Commenting a line means that the interpreter ignores it, not trying to run it as program code. You can do this by adding a # to the beginning of the line.)

   d. Why is numTwo incremented in the body of the "inner" loop?

e. What would happen if this line was moved to the "outer" loop (e.g., by removing one of the tabs from its indentation)?

f. Hopefully, you should see that since numTwo would never change, the condition of the second while loop would always remain true, and we'd have another infinite loop

10. Write a program that plays a familiar guessing game: first your program will set some number to be the "correct answer", then your program will ask the user to guess this answer until they get it right. For each guess your program should give a hint based on whether the guess was too high or too low. An example interaction should look like this.

    Enter your guess: 46
    Your guess is too low

    Enter your guess: 97
    Your guess is too high

    Enter your guess: 62
    Your guess is too low

    Enter your guess: 88
    Your guess is too high

    Enter your guess: 82
    CORRECT! You Win!

    a. *Hint 1: Start by creating two variables:* ***correct_number*** *and* ***user_guess*** *which keep track of the correct number and the number that the user guesses on input*

    b. *Hint 2: Use a while loop. In this case, the clause is "while user_guess is not correct, keep looping"*

11. Once you get to this point, raise your hand so I can check your progress.

_____