Lab 5 – PART 1

Topics Covered:

- Using functions
- Calling functions inside other functions
- main() function

Pre-Lab Questions

Question 1: Which of the following is the correct syntax for a function header?

a) def add_numbers(a, b)
b) def add_numbers(a,b);
c) define add_numbers(a,b):
d) def add_numbers(a,b):

Question 2: Take a look at the following code segment. Here we have declared the function "my_function" to complete a task. Look at the code and determine the output.

```
def my_function(a,b):
    result = a+b
    return result

print(my_function(13,21))
```

a) 8
b) 13
c) 21
d) 34

Question 3: In the code segment in question 2, when we are returning the value, why do we write line (1) instead of line (2)?

```
(1)   print(my_function(23,43))
(2)   my_function(23,43)
```

a) They both mean the same thing, so you can write either one
b) The first statement returns the value and prints the program output, while the second one would return the value without producing any output
c) We always use print when we are calling functions, as it is syntactically correct
d) It is good programming practice to use a print statement when returning a value

Question 4: Take a look at the following function. For each call of the function below, type the output that would be printed.

```
def is_multiple(n):
    if  n%7==0:
        return n
    elif n%3==0:
        return -1
    elif n%2==1:
        return True
    else:
        return False
```

1. print(is_multiple(2)) False
2. print(is_multiple(21)) 21
3. print(is_multiple(11)) True
4. print(is_multiple(9)) -1
5. print(is_multiple(16)) False
6. print(is_multiple(0)) 0

Activity:

The factorial of a number is that number multiplied by each of its predecessors until you reach the number 1. For example:

3! (read as 3 factorial) is 3x2x1 = 6.

5! is 5x4x3x2x1 = 120

Write a function that computes the factorial of a given number.

Hint: Use a negative step in your range function to decrement. You can use this as a template when creating your code:

```
def factorial(n):
    result = n
    for i in (insert a loop range):
        (insert a statement here)
    return result
```

Lab 5 – PART 2

1. **Follow the instructions below to construct a complete program.**

Let's create a function that has 1 print statement. We want it to print "Hello World". We will start with the function header, and then make the function print.

```python
def hello_world():
    print("Hello World")
```

Now, let's say we want to create a new function that prints Hello World a specific number of times. We can use the same code as in the previous function, but add a range to make it print as many times as is entered:

```python
def hello_world_n_times(n):
    for i in range(n):
        print("Hello World")
```

Now, we have these 2 functions. One that prints "Hello World" once, and another that prints it as many times as the user enters.

```python
def hello_world():
    print("Hello World")

def hello_world_n_times(n):
    for i in range(n):
         print("Hello World")
```

Notice how in the second function, we have a line that prints Hello World. We already created a function that does this, which is the first function. Instead of printing Hello World in the second function, we can just use the function we already created to complete this task, and just call it within the function.

```python
def hello_world_n_times(n):
    for i in range(n):
        hello_world()
```

Now, when you call the second function, once it reaches the call it will go through the first function and do whatever code is in it. Think about how useful this is in complex programs. If the first function was 21 lines long, instead of rewriting those 21 lines again in the second function, we can just call the function with 1 line and use its prewritten code instead of rewriting the code. This is why functions are very useful. Let’s test our function. Testing your functions is an important step that allows you to ensure that your function works. Let's print Hello World 2 times:

```
hello_world_n_times(2)
```

Let’s add a couple more tests:

```
hello_world_n_times(2)
hello_world_n_times(1)
hello_world_n_times(3)
hello_world_n_times(2)
```

Now this is our code:

```
def hello_world():
    print("Hello World")

def hello_world_n_times(n):
    for i in range(n):
        hello_world()


hello_world_n_times(2)
hello_world_n_times(1)
hello_world_n_times(3)
hello_world_n_times(2)
```

When we run this, you can see that the code works properly and Hello World prints twice, then once, then 3 times, and finally 2 more times for a total of 8 times. However, there is one more thing we want to look at. When defining and using functions in python, it is good programming practice to place all statements into functions and to specify one function as the starting point. Right now, everything is in functions except for the 4 statements at the bottom that we used to test our functions. In python, conventionally we use a function called main to contain everything that is not already in a function. Any legal name is allowed, but main is used because it is required in other languages, so it is good programming practice. If we put our code in the main function, this is what we will have:

```python
def hello_world():
    print("Hello World")

def hello_world_n_times(n):
    for i in range(n):
        hello_world()

# Include a "main" function
def main():
    hello_world_n_times(2)
    hello_world_n_times(1)
    hello_world_n_times(3)
    hello_world_n_times(2)
```

And of course, if we run this code nothing happens. This is because we have to call the main function. This is the only time we will have code that is not placed into a function, is when we are calling the main function. Now, rather than having 4 statements outside of functions, we just have the one call to the main function.

```python
def hello_world():
    print("Hello World")

def hello_world_n_times(n):
    for i in range(n):
        hello_world()

# Include a "main" function
def main():
    hello_world_n_times(2)
    hello_world_n_times(1)
    hello_world_n_times(3)
    hello_world_n_times(2)


# Call the main function
main()
```

2. **Identify errors and fix the code.**

The following code segment is a short function that takes in a word and returns the number of vowels in the word. There are 4 errors in this code. You may have to look at the test cases and the correct output in order to determine errors in the code.

```
def countVowels(word):
  numVowels = 1
  for letter in string:
    if letter in "aeiou":
      numVowels+=1
  return letter
```

Test Case:

```
print(countVowels("AEIOu"))

What is the output?  Is it correct?
Design other test cases to test the function and then correct the code.
```