

Ali Haider
14259
BS_SE 5
Final Term Exam
Course: Data Science
Instructor: Sir Ayub
Date: June 25, 2020



Note:

At the top of the answer sheet there must be the ID, Name and semester of the concerned Student.

Students must have to provide the output of their respective programs. Students have same answers or programs will be considered fail. Programs in Python and codes should be explained clearly.

Q1. a. Why Functions are used discuss in detail?

ANSWER#1a:

Functions:

Functions are "self-contained" modules of code that accomplish a specific task. Functions usually "take in" data, process it, and "return" a result. Once a function is written, it can be used over and over and over again. Functions can be "called" from the inside of other functions

There are three types of functions in Python:

- **Built-in functions**, such as help() to ask for help, min() to get the minimum value, print() to print an object to the terminal,... You can find an overview with more of these functions [here](#).
- **User-Defined Functions (UDFs)**, which are functions that users create to help them out; And
- **Anonymous functions**, which are also called lambda functions because they are not declared with the standard def keyword

Why do we Write Functions?

- They allow us to conceive of our program as a bunch of sub-steps. (Each sub-step can be its own function. When any program seems too hard, just break the overall program into sub-steps!)
- They allow us to reuse code instead of rewriting it.
- Functions allow us to keep our variable namespace clean (local variables only "live" as long as the function does). In other words, function_1 can use a variable called i, and function_2 can also use a variable called i and there is no confusion. Each variable i only exists when the computer is executing the given function.
- Functions allow us to test small parts of our program in isolation from the rest. This is especially true in interpreted languages, such as Matlab, but can be useful in C, Java, ActionScript, etc

b. How arguments are used in function, write a simple program in Python?

ANSWER#1b:

Arguments

- Information can be passed into functions as arguments.
- Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma

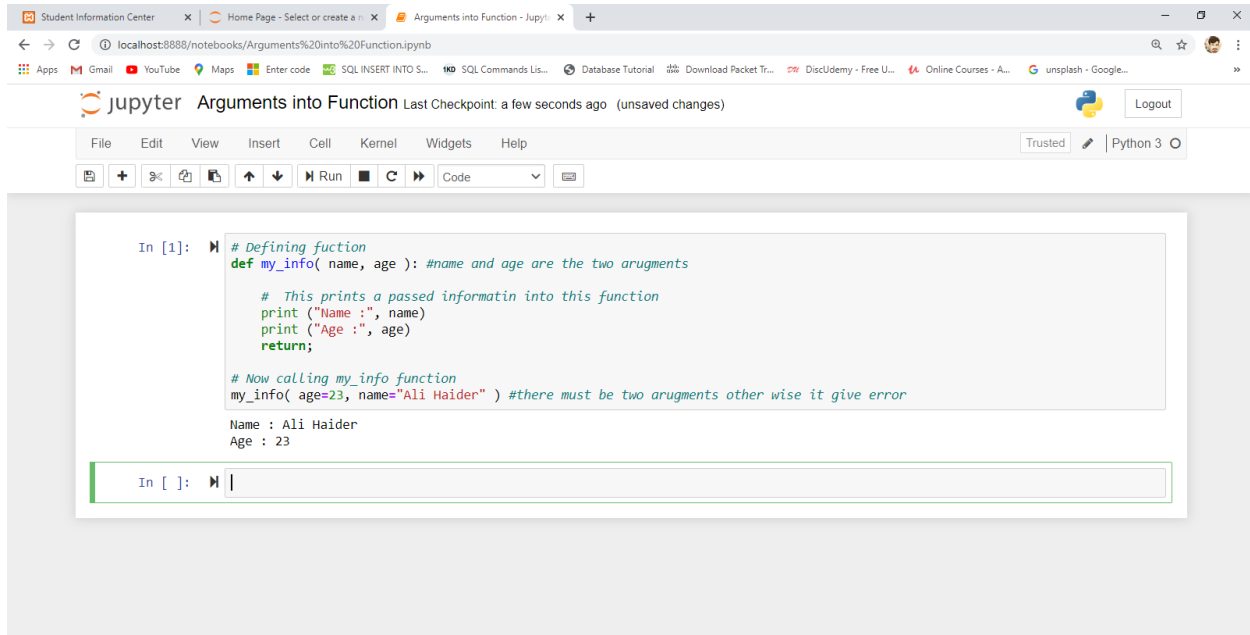
Parameters or Arguments?

The terms parameter and argument can be used for the same thing: information that are passed into a function.

From a function's perspective:

- A parameter is the variable listed inside the parentheses in the function definition.
- An argument is the value that is sent to the function when it is called

Example:



```
In [1]: # Defining function
def my_info( name, age ): #name and age are the two arugments

    # This prints a passed informatin into this function
    print ("Name :", name)
    print ("Age :", age)
    return;

# Now calling my_info function
my_info( age=23, name="Ali Haider" ) #there must be two arugments other wise it give error

Name : Ali Haider
Age : 23

In [ ]: |
```

Q2. a. Why .upper(),.lower(),capitalize() and .swapcase() function are used ?

ANSWER#2a:

Upper():

The string upper() method converts all lowercase characters in a string into uppercase characters and returns it.

The syntax for the upper() is **str_name.upper()** and it does not take any parameters.

Lower():

The string lower() method converts all uppercase characters in a string into lowercase characters and returns it.

The syntax for the upper() is **str_name.lower()** and it does not take any parameters.

capitalize():

In Python, the capitalize() method converts first character of a string to uppercase letter and lowercases all other characters, if any.

The syntax for the upper() is **str_name.capitalize()** and it does not take any parameters.

Swapcase():

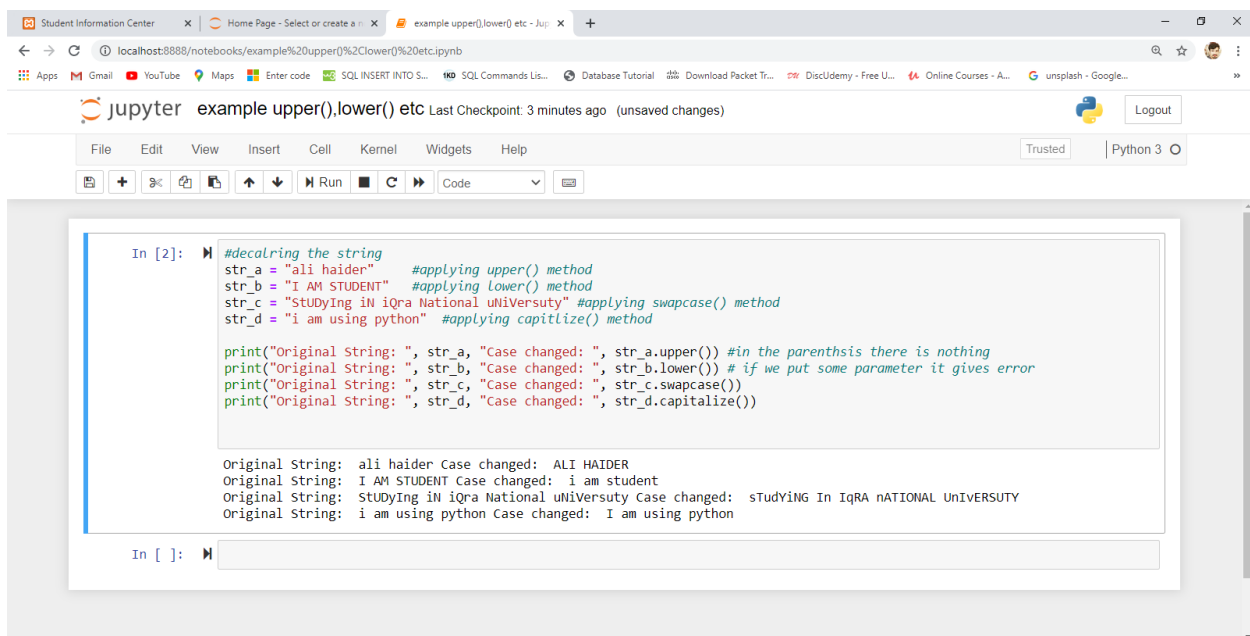
The string swapcase() method converts all uppercase characters to lowercase and all lowercase characters to uppercase characters of the given string, and returns it

The syntax for the upper() is **str_name.swapcase()** and it does not take any parameters.

b. Write a program in which the discussed functions are used.

Note : Q2 part a functions.

ANSWER#2b:



```
In [2]: #decalring the string
str_a = "ali haider" #applying upper() method
str_b = "I AM STUDENT" #applying lower() method
str_c = "StUDyIng iN iQra National uNIVERSuty" #applying swapcase() method
str_d = "I am using python" #applying capitlize() method

print("Original String: ", str_a, "Case changed: ", str_a.upper()) #in the parenthesis there is nothing
print("Original String: ", str_b, "Case changed: ", str_b.lower()) # if we put some parameter it gives error
print("Original String: ", str_c, "Case changed: ", str_c.swapcase())
print("Original String: ", str_d, "Case changed: ", str_d.capitalize())

Original String: ali haider Case changed: ALI HAIDER
Original String: I AM STUDENT Case changed: i am student
Original String: StUDyIng iN iQra National uNIVERSuty Case changed: sTUDyING In IqRA nATIONAL UNIVERSUTY
Original String: i am using python Case changed: I am using python
```

Q3. a. What are the rules for defining the function?

ANSWER#3a:

Defining a Function:

You can define functions to provide the required functionality. Here are simple rules to define a function in Python.

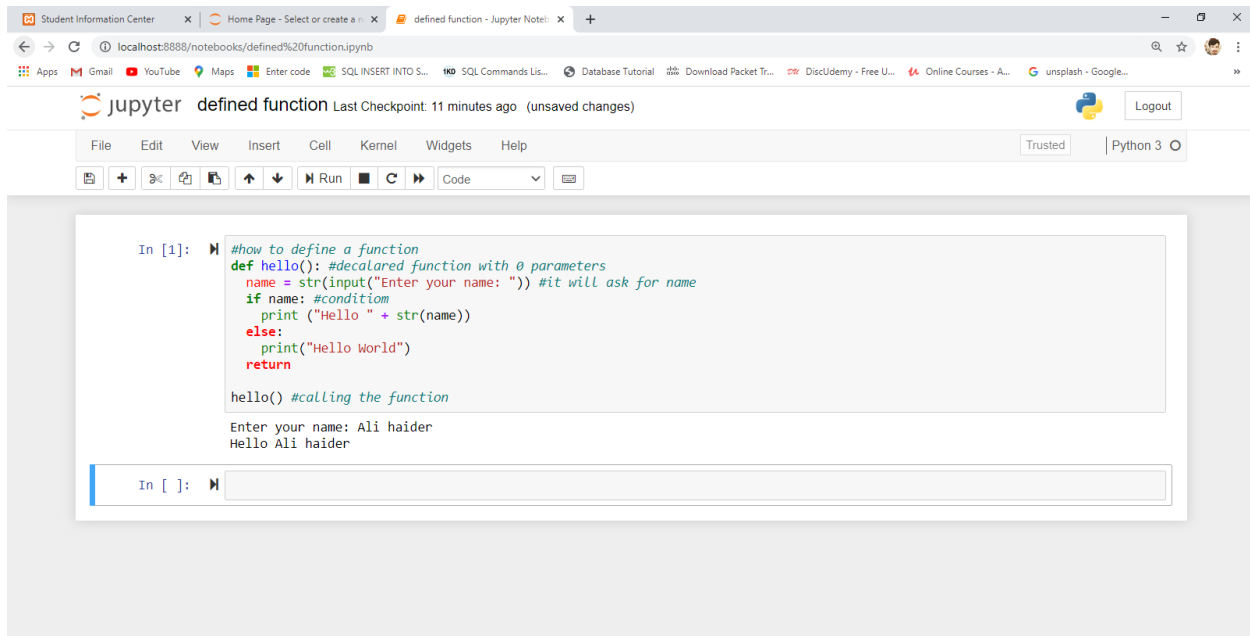
- Function blocks begin with the keyword **def** followed by the function name and parentheses (()).
- Any input parameters or arguments should be placed within these parentheses. You can also define parameters inside these parentheses.
- The first statement of a function can be an optional statement - the documentation string of the function or *docstring*.
- The code block within every function starts with a colon (:) and is indented.
- The statement `return [expression]` exits a function, optionally passing back an expression to the caller. A return statement with no arguments is the same as `return None`

Syntax:

```
def function_name( parameters ):
    "function_docstring"
    function_suite
    return [expression]
```

b. Write a suitable program of our defined function in Python?

ANSWER#3b:



```
In [1]: #how to define a function
def hello(): #decalared function with 0 parameters
    name = str(input("Enter your name: ")) #it will ask for name
    if name: #condition
        print ("Hello " + str(name))
    else:
        print("Hello World")
    return

hello() #calling the function

Enter your name: Ali haider
Hello Ali haider

In [ ]:
```

Q4. a. What are the rules for defining the function and Parameter passing to the function?

ANSWER#4a:

Parameters in a Function:

We often use the terms **parameters and arguments** interchangeably. However, there is a **slight difference between them**.

Parameters are the variables used in the function definition whereas **arguments** are the values we pass to the function parameters.

The following rules apply to parameters and arguments of C functions:

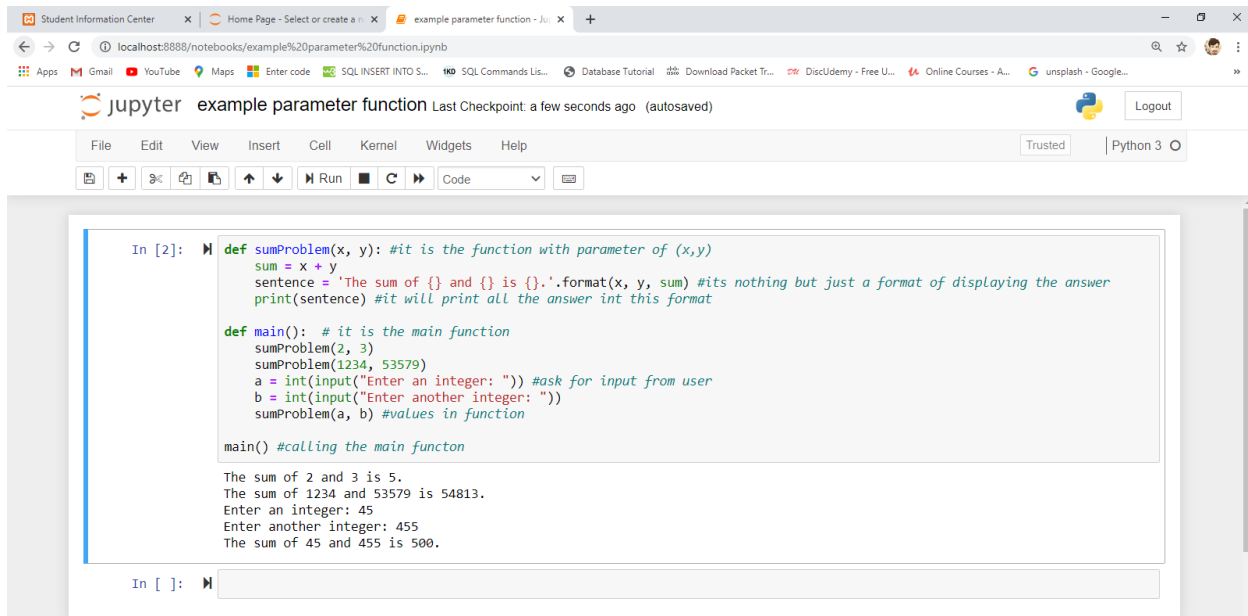
- Except for functions with variable-length argument lists, the number of arguments in a function call must be the same as the number of parameters in the function definition. This number can be zero.
- The maximum number of arguments (and corresponding parameters) is 253 for a single function.
- Arguments and parameters are separated by commas. However, the comma is not an operator in this context, and the arguments can be

evaluated by the compiler in any order. There is, however, a sequence point before the actual call.

- Arguments are passed by value; that is, when a function is called, the parameter receives a copy of the argument's value, not its address. This rule applies to all scalar values, structures, and unions passed as arguments.
- Modifying a parameter does not modify the corresponding argument passed by the function call. However, because arguments can be addresses or pointers, a function can use addresses to modify the values of variables defined in the calling function.
- In the old style, parameters that are not explicitly declared are assigned a default type of int .
- The scope of function parameters is the function itself. Therefore, parameters of the same name in different functions are unrelated.

b. Write a suitable program of our defined function by parameter passing in Python?

ANSWER#4b:



```
In [2]: def sumProblem(x, y): #it is the function with parameter of (x,y)
        sum = x + y
        sentence = 'The sum of {} and {} is {}'.format(x, y, sum) #its nothing but just a format of displaying the answer
        print(sentence) #it will print all the answer int this format

        def main(): # it is the main function
            sumProblem(2, 3)
            sumProblem(1234, 53579)
            a = int(input("Enter an integer: ")) #ask for input from user
            b = int(input("Enter another integer: "))
            sumProblem(a, b) #values in function

        main() #calling the main function

The sum of 2 and 3 is 5.
The sum of 1234 and 53579 is 54813.
Enter an integer: 45
Enter another integer: 455
The sum of 45 and 455 is 500.
```

Q5. a. What are return values to a Function discuss in detail?

ANSWER#5a:

Return Values To A Function:

A return statement is used to end the execution of the function call and “returns” the result (value of the expression following the return keyword) to the caller. The statements after the return statements are not executed. If the return statement is without any expression, then the special value *None* is returned.

Return statement cannot be used outside the function

A return value can be any one of the four variables types:

- Handle
 - Integer
 - Object
 - String
- Most functions take in arguments, perform some processing and then return a value to the caller. In Python this is achieved with the `return` statement.
 - Python also has the ability to return multiple values from a function call, something missing from many other languages.
 - An alternate syntax when dealing with multiple return values is to have Python “unwrap” the tuple into the variables directly by specifying the same number of variables on the left-hand side of the assignment as there are returned from the function

b. Write a suitable program of a Function with returning value?

ANSWER#5b:


```
In [1]: # Python program to
# demonstrate return statement
def add(a, b): #decarling function with parameters (a,b)

    # returning sum of a and b
    return a + b

def is_true(a):

    # returning boolean of a
    return bool(a)

# calling function
res = add(2, 3)
print("Result of add function is {}".format(res))

res = is_true(2<5)
print("\nResult of is_true function is {}".format(res))
```

Result of add function is 5

Result of is_true function is True