**FULL STACK ASSIGNMENT – 3**

**1.** Advantages of Using a Function

Use of functions **enhances the readability of a program**. A big code is always difficult to read. Breaking the code in smaller Functions keeps the program organized, easy to understand and makes it reusable.

**2.** Function in Python is defined by the**“def ”**statement followed by the function name and parentheses ( () )

Example:

Let us define a function by using the command ” def func1():” and call the function. The output of the function will be **“I am learning Python function”.**

**3.** Python creating functions. **A function is created with the def keyword**. The statements in the block of the function must be indented. The def keyword is followed by the function name with round brackets and a colon.

**4.** Using a function to do a particular task any point in program is called as function call.  
  
So the difference between the function and function call is,  
  
A function is procedure to achieve a particular result while function call is using this function to achive that task.  
  
so in our case function can be written as  
  
int get.square(int x)  
{  
return(x\*x);  
}  
  
The first line in above function is the function definition which is of the form Function\_Name (Argument1, Argument2 .....Argument n)  
  
The opening and closing braces defines the scope of that function while the code between these braces is the actuall code/procedure to achieve the goal.

**5.** There's only **one global Python scope per program execution**. This scope remains in existence until the program terminates and all its names are forgotten. Otherwise, the next time you were to run the program, the names would remember their values from the previous run.

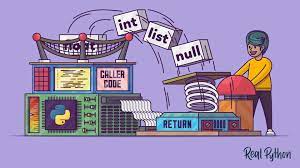
When you use an unqualified name inside a function, Python searches **three** scopes—the local (L), then the global (G), and then the built-in (B)—and stops at the first place the name is found.

**6.** A local variable **retains its value** until the next time the function is called

A local variable becomes **undefined** after the function call completes

The local variable can be used outside the function any time after the function call completes.

**7.** The Python return statement is a special statement that you can use inside a function or method to send the function's result back to the caller. A return statement consists of the return keyword followed by an optional return value. **The return value of a Python function can be any Python object**.

****

**8.** If a function doesn't specify a return value, it returns None.

In an if/then conditional statement, None evaluates to False.

**9.** If you create a variable with the same name inside a function, this variable will be local, and can only be used inside the function. The global variable with the same name will remain as it was, global and with the original value.

Example

Create a variable inside a function, with the same name as the global variable

x = "awesome"  
  
def myfunc():  
  x = "fantastic"  
  print("Python is " + x)  
  
myfunc()  
  
print("Python is " + x)

output:-

Python is fantastic

Python is awesome

**10.** The None keyword is used to define a null value, or no value at all.

None is not the same as 0, False, or an empty string. None is a data type of its own (NoneType) and only None can be None.

x = None  
  
if x:  
  print("Do you think None is True?")  
elif x is False:  
  print ("Do you think None is False?")  
else:  
  print("None is not True, or False, None is just None...")

output:-

None is not True, or False, None is just None...

**11.**  That import statement **imports a module named areallyourpetsnamederic**. (This isn't a real Python module, by the way.).

**12.** This function can be called with spam.bacon().

**13.** When it encounters an error, the control is passed to the except block, skipping the code in between. As seen in the above code, we have moved our code inside a try and except statement. **Try running the program** and it should throw an error message instead of crashing the program.

**14.** The try block lets you test a block of code for errors. The except block **lets you handle the error**.