

# Python Functions

Function is a group of related statements that perform a specific task.

Functions help break our program into smaller and modular chunks. As our program grows larger and larger, functions make it more organized and manageable.

It avoids repetition and makes code reusable.

## Syntax:

```
def function_name(parameters):  
    """  
    Doc String  
    """  
    Statement(s)
```

1. keyword "def" marks the start of function header
2. Parameters (arguments) through which we pass values to a function. These are optional
3. A colon(:) to mark the end of function header
4. Doc string describe what the function does. This is optional
5. "return" statement to return a value from the function. This is optional

## Example:

```
def print_name(name):  
    """  
    This function prints the name  
    """  
    print("Hello " + str(name))
```

## Function Call

Once we have defined a function, we can call it from anywhere

```
print_name('satish')  
Hello satish
```

## Doc String

The first string after the function header is called the docstring and is short for documentation string.

Although optional, documentation is a good programming practice, always document your code

Doc string will be written in triple quotes so that docstring can extend up to multiple lines

```
print(print_name.__doc__) # print doc string of the function  
  
This function prints the name
```

## return Statement

The return statement is used to exit a function and go back to the place from where it was called.

Syntax:

```
return [expression]
```

-> return statement can contain an expression which gets evaluated and the value is returned.

-> if there is no expression in the statement or the return statement itself is not present inside a function, then the function will return None Object

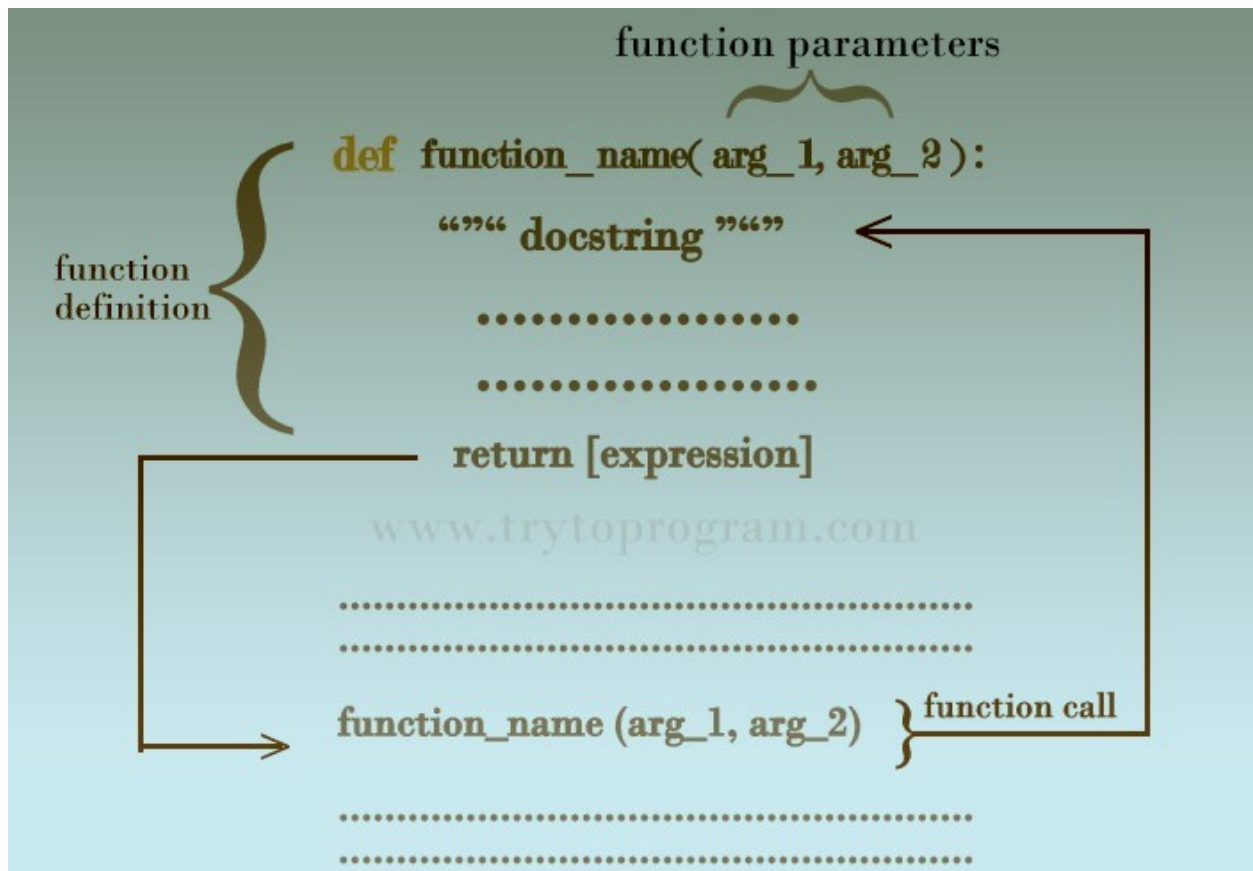
```
def get_sum(lst):  
    """  
    This function returns the sum of all the elements in a list  
    """  
    #initialize sum  
    _sum = 0  
  
    #iterating over the list  
    for num in lst:  
        _sum += num  
    return _sum  
  
s = get_sum([1, 2, 3, 4])  
print(s)
```

10

```
#print doc string  
print(get_sum.__doc__)
```

This function returns the sum of all the elements in a list

## How Function works in Python?



## Scope and Life Time of Variables

- > Scope of a variable is the portion of a program where the variable is recognized
- > variables defined inside a function is not visible from outside. Hence, they have a local scope.
- > Lifetime of a variable is the period throughout which the variable exists in the memory.
- > The lifetime of variables inside a function is as long as the function executes.

-> Variables are destroyed once we return from the function.

## Example:

```
global_var = "This is global variable"

def test_life_time():
    """
    This function test the life time of a variables
    """
    local_var = "This is local variable"
    print(local_var)          #print local variable local_var

    print(global_var)        #print global variable global_var

#calling function
test_life_time()

#print global variable global_var
print(global_var)

#print local variable local_var
print(local_var)

This is local variable
This is global variable
This is global variable
```

```
-----
-----
NameError                                Traceback (most recent call
last)
<ipython-input-12-d5226680661e> in <module>()
    19
    20 #print local variable local_var
--> 21 print(local_var)

NameError: name 'local_var' is not defined
```

## Python program to print Highest Common Factor (HCF) of two numbers

```
def computeHCF(a, b):
    """
```

*Computing HCF of two numbers*

"""

*smaller = b if a > b else a #consice way of writing if else statement*

```
hcf = 1
for i in range(1, smaller+1):
    if (a % i == 0) and (b % i == 0):
        hcf = i
return hcf
```

```
num1 = 6
num2 = 36
```

```
print("H.C.F of {0} and {1} is: {2}".format(num1, num2,
computeHCF(num1, num2)))
```

H.C.F of 6 and 36 is: 6