

Chapter 3: Working With Functions

1. Definition:

A **Function** is a *subprogram* that acts on data and *often returns a value*.

Or

A **function** is a named unit of a *group of program statements*. This unit can be invoked from *other parts of the program* as and when required.

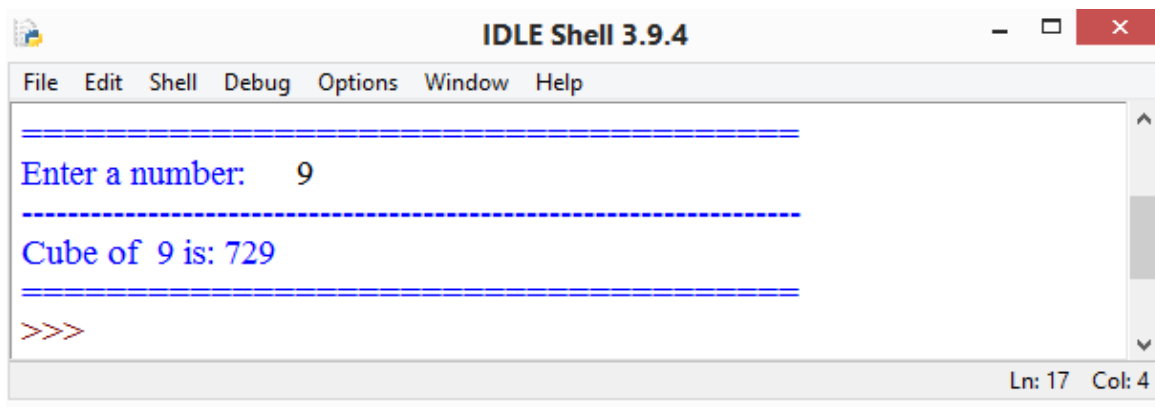
Example:

#Prg-1 Program to calculate cube of given number through function.

```
def calcube(x):           # Function Definition
    res=x**3
    return res

print('\n=====')
num=int(input("Enter a number:\t"))
print('-----')
cube=calcube(num)         # Function Call
print("Cube of ",num,"is:",cube,sep=' ')
print('=====')
```

Output:

A screenshot of the IDLE Shell 3.9.4 window. The window has a title bar with the text "IDLE Shell 3.9.4" and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with the following items: File, Edit, Shell, Debug, Options, Window, and Help. The main area of the window is a text editor with a light gray background. It contains the output of the program: a line of blue text "Enter a number:" followed by the input "9", a line of blue text "Cube of 9 is: 729", and a line of red text ">>>". The text is surrounded by blue dashed lines. At the bottom right of the window, there is a status bar showing "Ln: 17 Col: 4".

Prg-2 Program to add two numbers through function

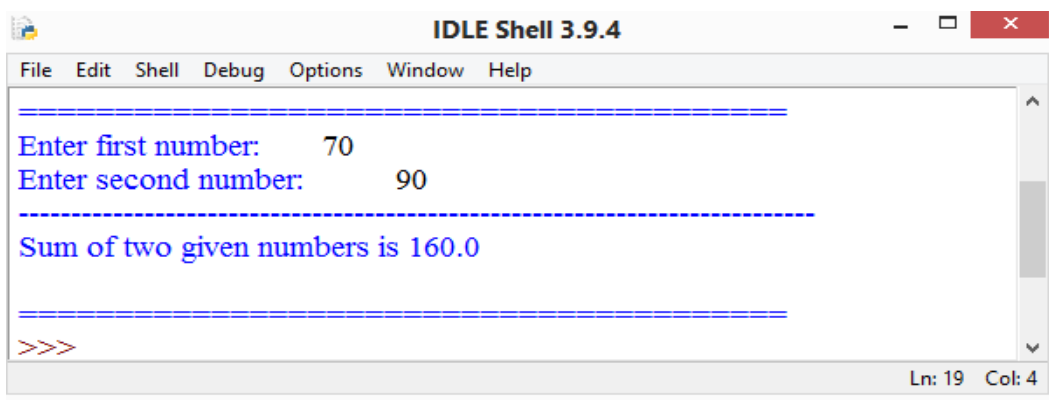
```
def calsum(x,y):                                #Function Definition
    s=x+y
    return s

print("\n=====")
num1=float(input("Enter first number:\t"))
num2=float(input("Enter second number:\t"))
print('-----')

Sum=calsum(num1,num2)                            #Function Call

print("Sum of two given numbers is",Sum,sep=' ')
print("\n=====")
```

Output:

A screenshot of the IDLE Shell 3.9.4 window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main text area shows the program's output in blue text: 'Enter first number: 70', 'Enter second number: 90', followed by a dashed line, 'Sum of two given numbers is 160.0', and another dashed line. At the bottom left of the text area is the prompt '>>>'. The status bar at the bottom right shows 'Ln: 19 Col: 4'.

2. Arguments and Parameters:

The values being passed through a function-call statement are called **arguments** or (**actual parameters or actual argument**).

The values received in the function definition/header are called **parameters** or (**formal parameters or formal arguments**)

We can say that Python refers to the values being passed (through function call) as **argument** and value being received (in function definition) **as parameters**.

Arguments in Python can be one of these following value types:

i) Literals

e.g.

multiply(17,9) #both literal arguments

ii) Variable

e.g.

multiply(d, 90) #One variable i.e ***d*** and one literal i.e. ***90*** argument

iii) Expressions

e.g.

multiply(d, d*10) #One variable i.e ***d*** and one expression i.e. ***d*10*** argument

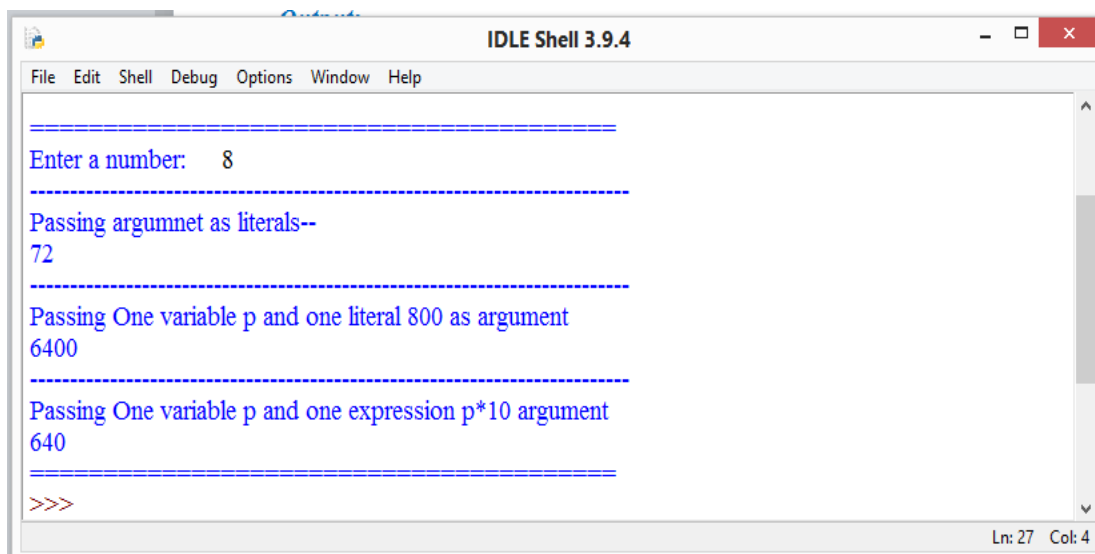
Example:

```
def multiply(x,y):
    print(x*y)

print("\n=====")
p=int(input("Enter a number:\t"))
print('-----')

print("Passing argumnet as literals--")
multiply(9,8)
print('-----')
print("Passing One variable p and one literal 800 as argument")
multiply(p,800)
print('-----')
print("Passing One variable p and one expression p*10 argument")
multiply(p,p*10)
print("=====")
```

Output:



```
=====
Enter a number: 8
-----
Passing argumnet as literals--
72
-----
Passing One variable p and one literal 800 as argument
6400
-----
Passing One variable p and one expression p*10 argument
640
=====
>>>
```

Ln: 27 Col: 4