Chapter 3: Working With Functions

Passing Parameters:

In Python if a *function header* has three parameters named in its header then the function call should also *pass three values*. Other than this, Python also provides some other ways of sending and matching *arguments and parameters*.

Python supports three types of *formal arguments/parameters*

1. **Positional arguments (Required arguments):**

   Positional arguments are arguments that can be called by their position in the *function definition.*

   Or

   When the *function call* statement must match the number and order of arguments as defined in the function definition, this is called the *positional argument matching.*

Example:

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

Then the possible function call for this function can be:

multiply(num1,num2) # Two values (all variables) passed

In this function call:

- *x* will get value of *num1*
- *y* will get value of *num2*

multiply(120, num1) #Two values (literal + variable) passed

In this function call:

- *x* will get value of literal *i.e. 120*
- *y* will get value of *num1*

multiply(8,90) # Two values(Both literals) passed

In this function call:

- *x* will get value of literal *i.e. 8*
- *y* will get value of literal *i.e. 90*

Thus through such function calls,

- The arguments must be provided for all *parameters (Required)*
- The values of arguments are matched with *parameters, position (order) wise (Positional)*

2. **Default arguments:**

   Python allows us to assign *default value(s)* to a *function's parameter(s)* which is useful in case a *matching is not passed in the function call statement*. The *default values* are specified in *the function header of function definition*.

   e.g

```
def interest(principal,  time, rate=0.08):
        :
        :
```

In the *function* given above ***rate=0.08*** is *default* value for parameter ***rate.*** If in function call, the value for rate is not provided Python will fill the missing value ***(for rate only)*** with this value.

Note:
*In a **function header**, any parameter cannot have a **default value** unless all parameters appearing **on its right have their default value**.*

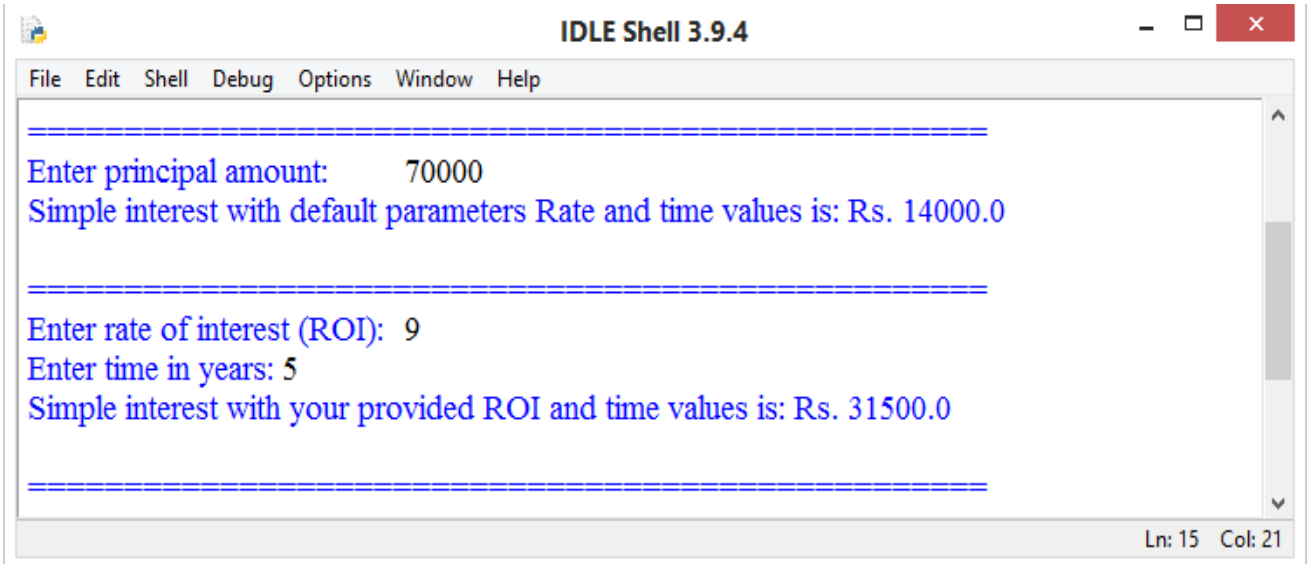
Example:

Program to calculate simple interest using a *function interest()* that can receive *principal amount, time and rate* and *returns calculated simple interest.* Do specify *default values* for *rate* and *time* as *10%* and *2 years respectively.*

Code:

```
def interest(principal, time=2, rate=0.10):
    return principal*rate*time


#____ main_________

print("\n==================================================")

prin=float(input("Enter principal amount:\t"))
SI=interest(prin)
print("Simple interest with default parameters Rate and time values is: Rs.",SI)

print("\n==================================================")
ROI=float(input("Enter rate of interest (ROI):\t"))
time=int(input("Enter time in years:\t"))
SI2=interest(prin,time,ROI/100)
print("Simple interest with your provided ROI and time values is: Rs.",SI2)
print("\n==================================================")
```

Output:



Note:
The default values for parameters are considered only if no value is provided for that parameter in the function call statement

3. Keyword (or named) arguments:

Keyword arguments are the *named arguments* with assigned values being passed in *function call statement*.

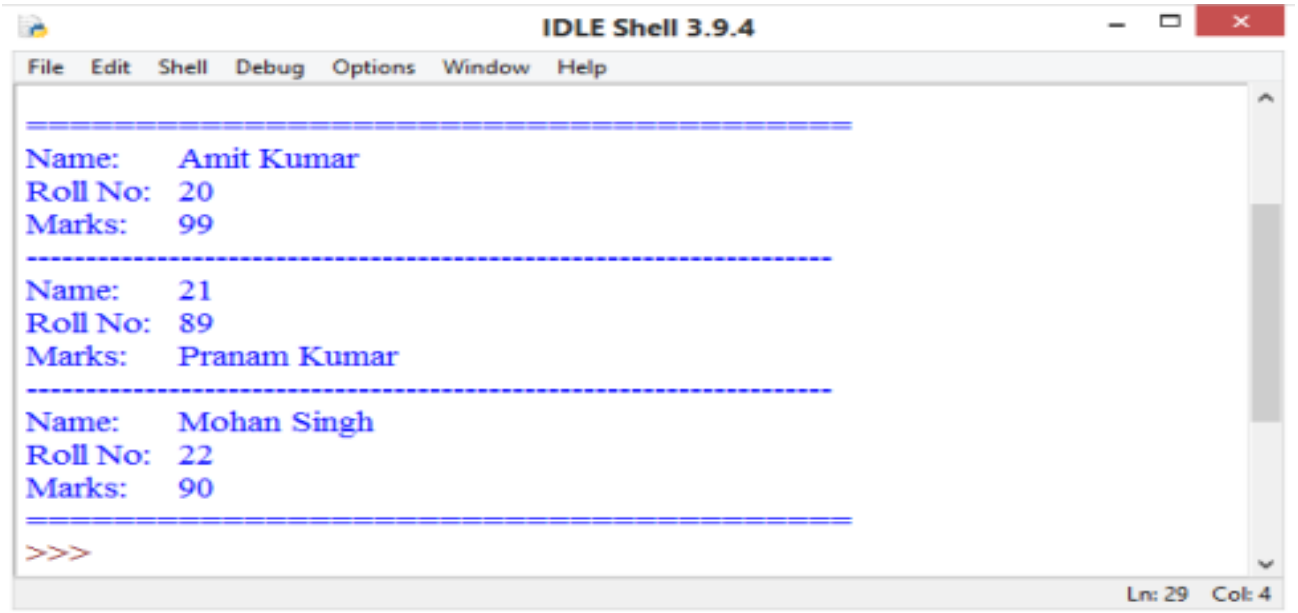
Or

The *Keyword* arguments give *complete control and flexibility* over the values sent as arguments for the *corresponding parameters*. Irrespective of the *placement* and *order of arguments*, *keyword arguments* are correctly matched. When we call function using *keyword arguments* , the *order (position) of the arguments can be changed.*

Example:

```
def keyarg(name,rollno,marks):
    print("Name:\t",name)
    print("Roll No:\t",rollno)
    print("Marks:\t",marks)
#_ _ _ _ _ _ _main_ _ _ _ _ _ _ _ _ _ _

print('\n=====================================')
keyarg ("Amit Kumar",20,99)
print("-----------------------------------------------------------------")
keyarg(21,89,"Pranam Kumar")
print("-----------------------------------------------------------------")
keyarg(marks=90,rollno=22,name="Mohan Singh")
print('=====================================')
```

Output:



Explanation:

There are three *function call statements* in the above given program:

In the first *function call* statement,

```
keyarg ("Amit Kumar",20,99)          #(Positional arguments)
```

name gets value *'Amit Kumar'*, *rollno* gets value *20* and *marks* as *99*

In the *second function* call statement,

```
keyarg(21,89,"Pranam Kumar")         #(Positional arguments)
```

name gets value *22*, *rollno* gets value *89* and *marks* as *'Pranam Kumar'*

(Which is logically wrong)

In the *third function* call statement,

```
keyarg(marks=90,rollno=22,name="Mohan Singh")
                                        #(Keyword arguments)
```

name gets value *'Mohan Singh'*, *rollno* gets value *22* and *marks* as *90*

4. Using Multiple Argument Types Together :

Python allows us to combine *multiple argument types* in a *function call*. Consider the following *function call statement*:

Let the function definition be:

```
def interest(prin, time=2, rate=0.10):
    return prin*rate*time
```

Now, the call statement:

```
interest(principal, rate=11/100,time=3)
```

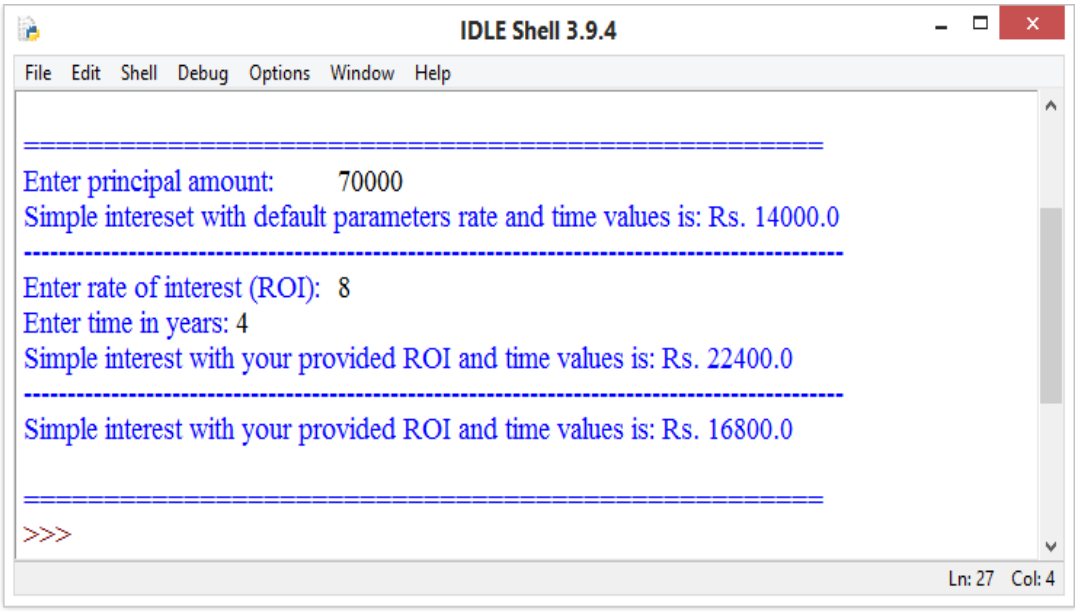
In the function call first argument value principal is representing positional argument as it will be assigned to first parameter on the basis of its position. The second and third arguments (rate=11/100 and time=3) are representing keyword arguments or named arguments, and if we skip one of the arguments either rate or time for which default values are defined in the function header, that will be called as default argument.

Example:

#Program demonstrates the functioning of passing multiple arguments types together.

```
def interest(princ, time=2, rate=0.10):
    return princ*rate*time

#____ main_________

print('\n==================================================')
principal=float(input("Enter principal amount:\t"))

SI=interest(principal)                    #Default arguments

print("Simple intereset with default parameters rate and time values is: Rs.",SI)
print('-----------------------------------------------------------------------------------------')

ROI=float(input("Enter rate of interest (ROI):\t"))
T=int(input("Enter time in years:\t"))

SI2=interest(principal,ROI/100,T)         #Positional arguments

print("Simple interest with your provided ROI and time values is: Rs.",SI2)
print('-----------------------------------------------------------------------------------------')

SI3=interest(principal,rate=8/100,time=3)
                                  # Positional, Keyword and default arguments together
print("Simple interest with your provided ROI and time values is: Rs.",SI3)
print('\n==================================================')
```

Output:


IDLE Shell 3.9.4
File Edit Shell Debug Options Window Help
==========
Enter principal amount: 70000
Simple intereset with default parameters rate and time values is: Rs. 14000.0

Enter rate of interest (ROI): 8
Enter time in years: 4
Simple interest with your provided ROI and time values is: Rs. 22400.0

Simple interest with your provided ROI and time values is: Rs. 16800.0
==========
>>>
Ln: 27 Col: 4