04/02/2022

Functions

If a group od statements is repeatedly required then it is not recommended to write these statements everytime seperately.We have to define these statements as a single unit and we can call that uniy any number of times based on our requirement without rewriting.This unit is nothing but function.

The main advantage of function is code Reusability

Note:- In other languages functions are known as methods,procedures,sunroutines etc.

Python supports 2types of functions.

1. Bulit in Functions
2. User define Functions.

1. Bulit in Functions

The functions which are coming along with python software automatically are called built in functions or pre-define functions.

ex:-

```
id()
type()
input()
dir()
eval()
help()
```

2. User define Functions.

The functions which are developed by programmer explicitly according to business requirements,are called user define functions.

Syntax to create user define functions:-

```
def function_name(parameters):
      """ doc string """
      statements
      statements

      return value
```

Note:- while creating functions we can use 2 keywords.

1. def(mandatory)
2. return(option)

```
def add(a,b):
    c = a+b
    print(c)

add(10,20)
```

OP:-

```
30
```

Parameters:-

Parameters are inputs to the functions.if a function contains parameters then at the time of calling compilsory we should provide values otherwise we will get error.

```
def add(a,b):
    c = a+b
    print(c)

add(10)
```

OP:-

```
Traceback (most recent call last):
  File
"C:\Users\chand\PycharmProjects\testing_17_18_19_python\Functions\user_def
ine_functions.py", line 52, in <module>
    add(10)
TypeError: add() missing 1 required positional argument: 'b'

Process finished with exit code 1
```

Return statements:-

Function can take input values as parameters and executes business logic and returns output to the caller with return statements.

#Q1:-WAF to take name of the students as input and print with message by name.

```
def std_name(name):
    print("Hello",name,"Good Morning")

std_name("Hitler")
```

```
Op:-
Hello Hitler Good Morning


#Q2:- WAF to take number as input and print its squre value.

def squreit(num):
    print("The squre of",num,"is:",num*num)

squreit(6)
squreit(8)

OP:-

The squre of 6 is: 36
The squre of 8 is: 64

#Q3:- WAF to check whether the given number is even or odd?

def even_odd(num):
    if num%2==0:
        print(num,"is Even number")
    else:
        print(num,"is Odd number")

even_odd(32)
even_odd(79)

Op:-.

32 is Even number
79 is Odd number

#Q4:- WAF to find factorial of given number?

def fact(num):
    result = 1
    while num>=1:
        result = result * num
        num = num -1
    return result

for i in range(1,6):
    print(i,"factorial is:",fact(i))

op:-

1. factorial is: 1
2. factorial is: 2
3. factorial is: 6
4. factorial is: 24
5. factorial is: 120
```

H/W:-

WAF for making a two digits normal(+,-,*,/) calculater?

Returning multiple values from a function:

```
def cal(a,b):
    sum = a+b
    sub = a-b
    mul = a*b
    div = a/b
    return sum,sub,mul,div

t = cal(20,10)
print(t)
```

op:-

```
(30, 10, 200, 2.0)
```

Types of arguments:-

```
def f1(a,b):
    ----
    ----
    ----

f1(10,20)
```

a,b are formal arguments where as 10,20 are actual arguments

There are 4types are actual arguments are allowed in python.

1. Positional Arguments
2. Keyword Arguments
3. Default Arguments
4. Variable length of Arguments.

1. Positional Arguments

These are the arguments passed to function in correct positinal order.

```
def sub(a,b):
    sub = a-b
    print(sub)

sub(20,10)
```

```
sub(10,20)

op:-

10
-  10

def sub(a,b):
    sub = a-b
    print(sub)

sub(20)

op:-

Traceback (most recent call last):
  File
"C:\Users\chand\PycharmProjects\testing_17_18_19_python\Functions\user_def
ine_functions.py", line 165, in <module>
    sub(20)
TypeError: sub() missing 1 required positional argument: 'b'

Note:- The number of arguments and position of arguments must be
matched.if we change the order then result may be changed.

If we change the number of arguments then we will get Error.


2.Keyword Arguments:-
--------------------
we can pass arguments values by keyword that is by parameter name.

def wish(name,msg):
    print("Hello",name,msg)

wish("Rahul","Good morning") #positinal args
wish("Good morning","Rahul") #positinal args #theroitcal error
wish(name="Rahul",msg="Good monring")#keyword args
wish(msg="Good monring",name="Rahul")#keyword args

OP:-

Hello Rahul Good morning
Hello Good morning Rahul
Hello Rahul Good monring
Hello Rahul Good monring

Note:- we can use both positional and keyword arguments simultaneously.But
fist we have to take positional arguments and then keyword
arguments,otherwise we will get syntaxerror,because keyword arguments
follows positional.


def wish(name,msg):
```

```
    print("Hello",name,msg)


wish("Rahul",msg="Good eveng")#positional+keyword
wish(name="Rahul","good eveng")#keyword+positional--syntaxerror

op:-
File
"C:\Users\chand\PycharmProjects\testing_17_18_19_python\Functions\user_def
ine_functions.py", line 176
    wish(name="Rahul","good eveng")
                                   ^
SyntaxError: positional argument follows keyword argument

Process finished with exit code 1
```

3.Default Arguments:-

Sometimes we can provide default values for our positional arguments

```
def wish(name="Guest"):
    print("Hello",name,"Good morning")

wish("Rahul")
wish("Maya")
wish()
wish()
wish()
```

Op:-

```
Hello Rahul Good morning
Hello Maya Good morning
Hello Guest Good morning
Hello Guest Good morning
Hello Guest Good morning

Process finished with exit code 0
```

Note:- If we are not passing any name then only default value will be considered.

We can not pass default arguments before the positinal or keyword arguments.