Function

A function is a block of code that performs a particular task. There are some situations when we need to write a particular block of code for more than once in our program. This may lead to bugs and irritation for the programmer.

Python provides an approach in which you need to declare and define a group of statements once and that can be called and used whenever required. This saves both time and space.

Types of Functions

- Built-in functions
- User-defined functions

Built-in Functions

Built-in functions are those functions which are already defined in library. We have used many predefined functions in Python.

User-defined function

User-defined functions are those functions which are defined by the user at the time of writing program. Functions are made for code reusability and for saving time and space.

Defining a Function

Keyword def is used to start and declare a function. Def specifies the starting of function block. def is followed by function-

name followed by parenthesis. Parameters are passed inside the parenthesis. At the end a colon is marked. Python code requires indentation (space) of code to keep it associate to the declared block. The first statement of the function is optional. Following is the statement to be executed.

```
def  function_name(parameters):
       Statement  (Function body)
       return statement
```

Calling a Function

Defining a function only gives it a name, specifies the parameters that are to be included in the function and structures the blocks of code.

Once the basic structure of a function is finalized, you can execute it by calling it from another function or directly from the Python prompt.

Example	Output

def abc():
  str = "Hello Python"
  print str
 	
#calling abc() Function
abc()
	Hello Python

Parameters

Information can be passed to functions as parameter. Parameters are specified after the function name, inside the parentheses. You can add as many parameters as you want, just separate them with a comma.

Example	Output

def abc(str):
  print str
 	
#calling abc() Function
abc("Hello Python")
	Hello Python

Python supports following types of formal argument.

- Required argument
- Keyword argument
- Default argument

Required arguments

Required arguments are the arguments passed to a function in correct positional order. Here, the number of arguments in the function call should match exactly with the function definition.

Example	Output

def abc(str,str1):
  print str + str1
 	
#calling abc() Function
	HelloPython
	
	Traceback (most recent call last):
	  File "main.py", line 7, in <module>
	  abc("HelloPython")

abc("Hello", "Python")
 	
#This statement generate an error
abc("HelloPython")
	TypeError: abc() takes exactly 2 arguments (1 given)

Keyword Arguments

Using the Keyword Argument, the argument passed in function call is matched with function definition on the basis of the name of the parameter.

Example	Output

def abc(str,str1):
  print str + str1
 	
#calling abc() Function
abc(str="Hello", str1="Python")
abc(str1="Hello", str="Python")
	HelloPython
	PythonHello

Default Arguments

Default Argument is the argument which provides the default values to the parameters passed in the function definition, in case value is not provided in the function call default value is used.

Example	Output

def sum(a,b=20):
  c=a+b
  print c
	40
	30

#calling sum() Function
sum(a=10,b=30)
sum(a=10)	

Anonymous Function

In Python, anonymous function is a function that is defined without a name. While normal functions are defined using the def keyword, in Python anonymous functions are defined using the lambda keyword. Hence, anonymous functions are also called lambda functions.

lambda arguments : expression

Example	Output
str = lambda str1, str2: str1+str2 print(str("Hello","Python"))	HelloPython

Example	Output
sum = lambda a, b: a+b print "Sum of a and b is ,sum(10,20)	Sum of a and b is 30

return Statement

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

Example	Output
def sum(a,b): return a+b print(sum(10,20))	30