Python Function

A function is a block of code which only runs when it is called.
You can pass data, known as parameters, into a function.
A function can return data as a result
A colon (:) to mark the end of the function header.
An optional return statement to return a value from the function.

Creating a Function

In Python a function is defined using the def keyword:

```
def function():
    print("Hello, Welcome to MachineLearning.org.in")

function()
Hello, Welcome to MachineLearning.org.in
function()
Hello, Welcome to MachineLearning.org.in
```

Scope of a Variable in Function

Local variable is declared inside a function whereas Global variable is declared outside the function. Local variables are created when the function has started execution and is lost when the function terminates, on the other hand, Global variable is created as execution starts and is lost when the program ends.

```
a=10
def fun():
    print("Inside:",a)

print("Outside:",a)
fun()
Outside: 10
Inside: 10
```

Note: Global variable can be used anywhere

```
a=10 #Local Variable
def fun():
    a=20 #Global Variable
    print("Inside:",a)

fun()
print("Outside:",a)
Inside: 20
Outside: 10
```

Note: Inside local function, local variable priority is first

```
a=10
def fun():
    a=a+30
    print("Inside:",a)
```

```
fun()
print("Outside:",a)
--------------------------------------------------------------------
-------
UnboundLocalError                         Traceback (most recent
call last)
<ipython-input-5-4196ca061bb8> in <module>
      4     print("Inside:",a)
      5
----> 6 fun()
      7 print("Outside:",a)

<ipython-input-5-4196ca061bb8> in fun()
      1 a=10
      2 def fun():
----> 3     a=a+30
      4     print("Inside:",a)
      5

UnboundLocalError: local variable 'a' referenced before assignment
```

Note:: We can’not change global variable value inside a local function directly

```
a=10
def fun():
    global a
    a=a+30
    print("Inside:",a)

fun()
print("Outside:",a)
Inside: 40
Outside: 40
```

Types of Functions
Function with No Argument, No Return Value
Function with No Argument, with Return Value
Function with Argument, No Return Value
Function with Argument, with Return Value

1. Function with No Argument, No Return Value

```
def add():
    a=int(input("Enter First No:"))
    b=int(input("Enter Second No:"))
    print("Sum:",a+b)

add()
Enter First No:10
Enter Second No:20
Sum: 30
```

```
2. Function with No Argument, with Return Value
def add():
    a=int(input("Enter First No:"))
    b=int(input("Enter Second No:"))
    print("Sum:",a+b)

a=add()
print(a)
Enter First No:10
Enter Second No:20
Sum: 30
None
def add():
    a=int(input("Enter First No:"))
    b=int(input("Enter Second No:"))
    return a+b

a=add()
print("Sum:",a)
Enter First No:10
Enter Second No:20
Sum: 30



3. Function with Argument, No Return Value
def add(x,y):
    print("Sum:",x+y)

a=int(input("Enter First No:"))
b=int(input("Enter Second No:"))
add(a,b)
Enter First No:10
Enter Second No:20
Sum: 30



4. Function with Argument, With Return Value
def add(x,y):
    return x+y

a=int(input("Enter First No:"))
b=int(input("Enter Second No:"))
print("Sum:",add(a,b))
Enter First No:10
Enter Second No:20
Sum: 30


Default Argument
def fun1(a="itronix",b=20):
    print(a,b)
```

```
fun1() #No Argument pass
fun1(b=30) #Single Argument Pass
fun1(b=30,a="sol") #Both Argument Pass
itronix 20
itronix 30
sol 30
```

Calculate time of execution of a function

```
import time

def square():
    start=time.time()
    for i in range(100000000):
        i=i*i
    end=time.time()
    print("Time Taken:",end-start)

square()
Time Taken: 6.7782227993011475
```

Pass Function as a Argument in other function

```
def fun(a):
    a("Itronix Solutions")

fun(print)
Itronix Solutions
```

Return Function from another Function

```
def fun():
    return print

a=fun()
a("Machine Learning")
Machine Learning
```

Nested Function

```
def fun1():
    print("Hello")
    def fun2():
        print("Machine Learning")
    fun2()

fun1()
Hello
Machine Learning
```

Calculate the time of execution of each function

```
import time
```

```
def fun1(fun):
    def fun2():
        s=time.time()
        fun()
        e=time.time()
        print("Time Taken:",e-s)
    return fun2

def square():
    for i in range(10000000):
        i=i*i

def cube():
    for i in range(10000000):
        i=i*i*i

a=fun1(square)
a()

a=fun1(cube)
a()
Time Taken: 0.6668052673339844
Time Taken: 1.196408748626709
```

Function Decorators : Time of execution of each function

```
import time
def fun1(fun):
    def fun2():
        s=time.time()
        fun()
        e=time.time()
        print("Time Taken:",e-s)
    return fun2


@fun1
def square():
    for i in range(10000000):
        i=i*i

@fun1
def cube():
    for i in range(10000000):
        i=i*i*i

square()
cube()
Time Taken: 0.6784050464630127
Time Taken: 1.183851957321167
```