Default Parameter Value

```
def my_function(country = "Norway"):
    print("I am from " + country)
```

call
```
my_function("Sweden")
```
Output
```
I am from Sweden
```
call
```
my_function("India")
```
Output
```
I am from India
```
call
```
my_function()
```
Output
```
I am from Norway
```

Passing a List as an Argument

We can send any data types of argument to a function (string, number, list, dictionary).

Eg
1)
```
def my_function(xyz):
    for x in xyz:
        print(x)
fruits = ["apple", "banana", "cherry"]
```
call
```
my_function(fruits)
```

Output
```
apple
banana
cherry
```

Return Values

Syntax : return [expression_list]

Eg
1)
```
def square_value(num):
    return num**2
```
call
```
square_value(5)
```
Output

2) def my-function (x):
    x[0] = 30

Next
lst = [9, 11, 13, 14, 15]
my-function (lst)
print (lst)

Function within Functions

```
def f1():
    s = "Priya Singh"
    def f2():
        print (s)
    f(2)
    f2()
```

Call
f1()

Priya Singh

Python lambda

In Python, anonymous function means that a function is without a name. As we already

Syntax:
lambda arguments : expression

e.g.
g = lambda x : x ** 2

call
print (g(3))

Output
27

② def power(n):
    return lambda a: a**n

Next
base = power(2)

Next
base(6)

Output
36

⊙ type(base)
function.

filter()

We can also replace list comprehension with Lambda by using a map() method, not only it is a fast but efficient too and let's also see how to use lambda in the filter().

e.g.
a = [100, 2, 4, 80, 5, 4, 3, 31, 11]

filtered = filter(lambda p: p%2==0, a)

print(filtered)

print(list(filtered))

Output
[100, 2, 80, 4]

maped()

In map either we use assignment or conditional operator, the result of the value will.
maped = map(lambda n: n%2==0, a)

print(list(maped))

[True, True, True, False, True, False, True
True False, False]

Global and Local Variables

```
s = "I love python"
def myfunction()
    print(s)
```

call
myfunction()
output
I love python..

print(s)
output
I love python.

```
def myf():
    global v
    v = "I like food"
    print(v)
```

myf()
Output
I like food
print(v)
I like food.

2)
```
def p():
    s = "me too." (local)
    print(s)
s = "I love python".
```

call
p()
me too
call print(s)
I lovepython.