In [2]:
```
#functions in Python

#Positional parameters
def add(a,b):
    #Body of a function
    print(a)
    print(b)
    print("Addition")
    return a+b

result = add(30,40)
print(result)
```

```
30
40
Addition
70
```

In [4]:
```
#Defining the fucntion
def display():
    print("hello") #Inside the function body print statement is written
display() #Calling the function
```

```
hello
```

In [5]:
```
#Keyword parameters
def name(fname, lname):
    print("Hello",fname,lname)

#Calling function
name(lname="Sundaramoorthy",fname="Rekha")
```

```
Hello Rekha Sundaramoorthy
```

In [17]:
```
def greetings(msg,name):
    print(msg)
    print(name)
    print(name,msg)

greetings("Good Morning ","Dinesh")
print("**********************************")
greetings("Dinesh","Good Morning ")
print("**********************************")
greetings(name="Dinesh",msg="Good Morning ") #Calling the function using key word parameters
```

```
print("**********************************")
greetings(msg="Python makes me sleepy",name="Devendra")
print("**********************************")
greetings("Python makes me sleepy",name="Devendra")
```

```
Good Morning
Dinesh
Dinesh Good Morning
**********************************
Dinesh
Good Morning
Good Morning  Dinesh
**********************************
Good Morning
Dinesh
Dinesh Good Morning
**********************************
Python makes me sleepy
Devendra
Devendra Python makes me sleepy
**********************************
Python makes me sleepy
Devendra
Devendra Python makes me sleepy
```

In [20]:
```
#default parameters

def def_par (a,b,c,d=0,e=0):
    print(a+b+c+d+e)
    print(d)
    print(e)
def_par(10,20,30)
```

```
60
0
0
```

In [24]:
```
def personDetails(name,age=18):
    print(name," ",age)

personDetails("Rakesh")
personDetails("Rakesh",25)
personDetails("Chintu",15)
```

```
Rakesh  18
Rakesh  25
Chintu  15
```

In [25]:
```
#Variable length arguments
def displayNumbers(*nums):
    print(nums)
displayNumbers(1,2,3,4,5,6,7,8,9)

#def disNum(n1,n2,n3,n4,n5,n6,n7,n8,n9,n10)
```

```
(1, 2, 3, 4, 5, 6, 7, 8, 9)
```

In [26]:
```
#Variable length arguments

def var_len(*argv):
    for arg in argv:
        print(arg)

var_len("Hello","We have","come to the","end of the session", "Happy Learning")
```

```
Hello
We have
come to the
end of the session
Happy Learning
```

In [29]:
```
#Variable length arguments

def var_len(str1,str2,*argv):
    print(str1)
    print(str2)
    print("******************")
    for arg in argv:
        print(arg)

var_len("Hello","We have","come to the","end of the session", "Happy Learning")
```

```
Hello
We have
******************
come to the
end of the session
Happy Learning
```

In [33]:
```
#Anonymous function
sum=lambda x,y:x+y
res=sum(15,10) #Calling the lambda
print(res)
#Anonymous function without any parameters
msg = lambda:print("Hello All")
msg() #Calling the lambda

#Another example
#lambda accepts one argument
greet_user=lambda name:print("Hey there, ",name," don't sleep")
greet_user('John')
```

```
25
Hello All
Hey there,  John  don't sleep
```

In [41]:
```
'''
    A function which takes another function as an argument is known as higher order
    function
'''
#Function 1
def display(n):
    print(n*2)

#Function 2 takes 2 argument- 1st n is just for a number,2nd argument anotherfunction
def myFunc(n,anotherFunc):
    anotherFunc(n)

myFunc(10,display)

print("************Using lambda**************************")
#lambda for function 1
#lambda n:print(n*2)
def myFunct2(n):
    return lambda n:print(n*2)

res = myFunct2(15)

res(15)
```

```
20
************Using lambda**************************
30
```

In [54]:
```
listOfNumbers=[1,3,2,6,5,1,7,8,9,10,12,14,16,18,19]
newlist = filter(lambda x:x%2==0,listOfNumbers)
print(list(newlist))
#Using lambda
oddNumbers = filter(lambda x:x%2!=0,listOfNumbers)
print(list(oddNumbers))

#Using classic function definition
def oddNumbers(n):
    return n%2!=0
oddNums=filter(oddNumbers,listOfNumbers)
print(list(oddNums))
```

```
[2, 6, 8, 10, 12, 14, 16, 18]
[1, 3, 5, 1, 7, 9, 19]
[1, 3, 5, 1, 7, 9, 19]
```

In [55]:
```
#map function
numbers =[1,2,3,4,5]
squareOfNumbers=list(map(lambda x:x*x,numbers))
print(squareOfNumbers)
```

```
[1, 4, 9, 16, 25]
```

In [61]:
```
#reduce function
from functools import reduce

numbers =[1,2,3,4,5]
sumOfItems=reduce(lambda x,y:x+y,numbers)
print(sumOfItems)
maximumValue= reduce(lambda x,y: x if(x>y) else y,numbers)
print(maximumValue)
```

```
15
5
```

In [ ]: