**lambda functions**

In Python, anonymous function is a function that is defined without a name.

A lambda function can take any number of arguments, but can only have one expression.

While normal functions are defined using the def keyword, in Python anonymous functions are defined using the lambda keyword.

Syntax : lambda arguments: expression

x = lambda a : a + 10  
print(x(5))

x = lambda a, b : a \* b  
print(x(5, 6))

x = lambda a, b, c : a + b + c  
print(x(5, 6, 2))

The power of lambda is better shown when you use them as an anonymous function inside another function.

def myfunc(n):  
  return lambda a : a \* n  
  
mydoubler = myfunc(2)  
mytripler = myfunc(3)  
  
print(mydoubler(11))   
print(mytripler(11))

Lambda functions are used along with built-in functions like filter(), map() etc.

#### Example use with filter()

The function is called with all the items in the list and a new list is returned which contains items for which the function evaluates to True.

# Program to filter out only the even items from a list

my\_list = [1, 5, 4, 6, 8, 11, 3, 12]

new\_list = list(filter(lambda x: (x%2 == 0) , my\_list))

# Output: [4, 6, 8, 12]

print(new\_list)

#### Example use with map()

The map() function in Python takes in a function and a list.

The function is called with all the items in the list and a new list is returned which contains items returned by that function for each item.

# Program to double each item in a list using map()

my\_list = [1, 5, 4, 6, 8, 11, 3, 12]

new\_list = list(map(lambda x: x \* 2 , my\_list))

# Output: [2, 10, 8, 12, 16, 22, 6, 24]

print(new\_list)