Lambda : Map Filter Reduce (Anonymous Functions in python)

> Compiled by:
>
> Md Farmanul Haque
>
> Technical Trainer,
>
> GLA University,
>
> Mathura

In Python, an anonymous function means that a function is without a name. As we already know that the def keyword is used to define a normal function in Python. Similarly, the lambda keyword is used to define an anonymous function in Python. It has the following syntax:

Syntax: lambda arguments: expression This function can have any number of arguments but only one expression, which is evaluated and returned. One is free to use lambda functions wherever function objects are required. You need to keep in your knowledge that lambda functions are syntactically restricted to a single expression. It has various uses in particular fields of programming besides other types of expressions in functions.

Lambda functions:

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

```
#Simple multiplication function
def multiply(x, y):
    return x * y
```

```
multiply(2,3)
```

```
6
```

```
#lamda function
mul = lambda x, y: x * y
```

```
mul(2,3)   # call the lambda function
```

```
6
```

```
def myFilter(func, list):
    newlist = []
    for i in list:
        if func(i):
            newlist.append(i)
    return newlist
```

```
my_list = [1, 5, 4, 6, 8, 11, 3, 12]
```

```
my_list * 2
```

```
[1, 5, 4, 6, 8, 11, 3, 12, 1, 5, 4, 6, 8, 11, 3, 12]
```

```
def is_div_by_2(x):
    return x % 2 == 0
```

```
print(myFilter(is_div_by_2,my_list))
```

```
[4, 6, 8, 12]
```

```
print(myFilter(lambda x: x % 2 == 0, my_list))
```

```
[4, 6, 8, 12]
```

```
# Program to filter out only the even items from a list
new_list = filter(lambda x: x % 2 == 0 , my_list)

print(list(new_list))
```

```
[4, 6, 8, 12]
```

```
# Program to double each item in a list using map()
new_list = map(lambda x: x * 2 , my_list)

print(list(new_list))
```

```
[2, 10, 8, 12, 16, 22, 6, 24]
```

```
import functools
lambda_reduce = functools.reduce(lambda x,y: x+y, my_list)
print(lambda_reduce)
```

```
50
```

Using lambda() Function with map() The map() function in Python takes in a function and a list as an argument. The function is called with a lambda function and a list and a new list is

returned which contains all the lambda modified items returned by that function for each item. Example:

```
# Python code to illustrate
# map() with lambda()
# to get double of a list.
li = [5, 7, 22, 97, 54, 62, 77, 23, 73, 61]

final_list = list(map(lambda x: x*2, li))
print(final_list)
```

```
[10, 14, 44, 194, 108, 124, 154, 46, 146, 122]
```

Using lambda() Function with filter() The filter() function in Python takes in a function and a list as arguments. This offers an elegant way to filter out all the elements of a sequence “sequence”, for which the function returns True. Here is a small program that returns the odd numbers from an input list:

```
# Python code to illustrate
# filter() with lambda()
li = [5, 7, 22, 97, 54, 62, 77, 23, 73, 61]

final_list = list(filter(lambda x: (x%2 != 0) , li))
print(final_list)
```

```
[5, 7, 97, 77, 23, 73, 61]
```

Using lambda() Function with reduce() The reduce() function in Python takes in a function and a list as an argument. The function is called with a lambda function and an iterable and a new reduced result is returned. This performs a repetitive operation over the pairs of the iterable. The reduce() function belongs to the functools module.

```
# Python code to illustrate
# reduce() with lambda()
# to get sum of a list

from functools import reduce
li = [5, 8, 10, 20, 50, 100]
sum = reduce((lambda x, y: x + y), li)
print (sum)
```

```
193
```

```
# python code to demonstrate working of reduce()
# with a lambda function
```

```
# with a lambda function

# importing functools for reduce()
import functools

# initializing list
lis = [ 1 , 3, 5, 6, 2, ]

# using reduce to compute maximum element from list
print ("The maximum element of the list is : ",end="")
print (functools.reduce(lambda a,b : a if a > b else b,lis))
```

```
The maximum element of the list is : 6
```

```
# map is also used in competitive coding for taking and converting inputs in one line
#Example:
arr=list(map(int,input().split()))
print(arr)
```

```
12 13 14 15 17 76
[12, 13, 14, 15, 17, 76]
```

✓ 10s completed at 8:06 PM