Python Lambda

A lambda function is a small anonymous function.

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

Syntax

```
lambda arguments : expression
```

EX-Add 10 to argument a, and return the result:

```
x = lambda a: a + 10

print(x(5))
```

Multiply argument a with argument b and return the result:

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

Summarize argument a, b, and c and return the result:

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

Why Use Lambda Functions?

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

Say you have a function definition that takes one argument, and that argument will be multiplied with an unknown number:

```
def myfunc(n):
  return lambda a : a * n
```

Use that function definition to make a function that always doubles the number you send in:

Example

```
def myfunc(n):
  return lambda a : a * n

mydoubler = myfunc(2)

print(mydoubler(11))
```

use the same function definition to make a function that always *triples* the number you send in:

Example

```
def myfunc(n):
  return lambda a : a * n

mytripler = myfunc(3)

print(mytripler(11)
```

use the same function definition to make both functions, in the same program:

Example

```
def myfunc(n):
  return lambda a : a * n

mydoubler = myfunc(2)
mytripler = myfunc(3)

print(mydoubler(11))
print(mytripler(11))
```

Use lambda functions when an anonymous function is required for a short period of time.

Here we give an example of lambda function with filter() in Python. The code is given below -

```
# This code used to filter the odd numbers from the given list
list_ = [35, 12, 69, 55, 75, 14, 73]
odd_list = list(filter( lambda num: (num % 2 != 0) , list_ ))
print('The list of odd number is:',odd_list)
```

Using Lambda Function with map()

A method and a list are passed to Python's map() function.

The function is executed for all of the elements within the list, and a new list is produced with elements generated by the given function for every item.

The map() method is used to square all the entries in a list in this example.

Program Code:

Here we give an example of lambda function with map() in Python. Then code is given below -

```
#Code to calculate the square of each number of a list using the map() function
numbers_list = [2, 4, 5, 1, 3, 7, 8, 9, 10]
squared_list = list(map( lambda num: num ** 2 , numbers_list ))
print( 'Square of each number in the given list:' ,squared_list )
```

Using Lambda Function with List Comprehension

In this instance, we will apply the lambda function combined with list comprehension and the lambda keyword with a for loop. Using the Lambda Function with List Comprehension, we can print the square value from 0 to 10. For printing the square value from 0 to 10, we create a loop range from 0 to 11.

Program Code:

Here we give an example of lambda function with List Comprehension in Python. Then code is given below -

```
  #Code to calculate square of each number of lists using list comprehension
  squares = [lambda num = num: num ** 2 for num in range(0, 11)]
for square in squares:
   print('The square value of all numbers from 0 to 10:',square(), end = " ")
```

Using Lambda Function with if-else

We will use the lambda function with the if-else block. In the program code below, we check which number is greater than the given two numbers using the if-else block.

Program Code:

Here we give an example of a lambda function with an if-else block in Python. The code is given below -

```
# Code to use lambda function with if-else
Minimum = lambda x, y : x if (x < y) else y
print('The greater number is:', Minimum( 35, 74 ))
```