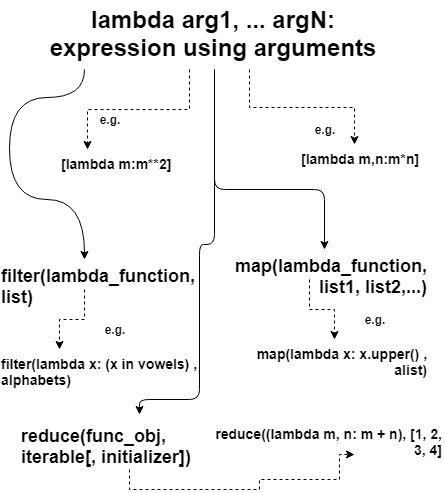
**Lambda function**

* It is unnamed function



**Syntax**

lambda arg1, arg2, ... argN: expression using arguments

* lambda function can’t include any statements. It only returns a function object which you can assign to any variable.
* can appear in places where the def is not allowed
* For example – **inside a list literal or a function call’s arguments**, etc.

**Lambda inside a list**

alist = [lambda m:m\*\*2, lambda m,n:m\*n, lambda m:m\*\*4]

print(alist[0](10), alist[1](2, 20), alist[2](3)) # Output: 100 40 81

**Lambda inside a dictionary**

key = 'm'

aDict = {'m': lambda x:2\*x, 'n': lambda x:3\*x}

print (aDict[key](9)) # Output: 18

Python provides two built-in functions like **filter(), map()** which can receive lambda functions as arguments.

Map()

map(function\_object, iterable1, iterable2,...)

**It expects variable-length arguments:** first is the **lambda function object**, and rest are the **Iterables such a list, dictionary**, etc.

**How it works:**

The map function iterates all the lists (or dictionaries etc.) and calls the lambda function for each of their element.

* **Output of map() function is list**

# Python lambda demo to use map()

alist = ['learn', 'python', 'step', 'by', 'step']

output = list(map(lambda x: x.upper() , alist))

# Output: ['LEARN', 'PYTHON', 'STEP', 'BY', 'STEP']

print(output)

# Python lambda demo to use map() for adding elements of two lists

list1 = [1, 2, 3, 4]

list2 = [100, 200, 300, 400]

output = list(map(lambda x, y: x+y , list1, list2))

# Output: [101, 202, 303, 404]

print(output)

**Filter() Function**

Select the iterable (list,dictionary,etc..) based on test function

**Syntax**:

filter(function\_object, list)

**How it works:**

The filter function iterates all the lists (or dictionaries etc.) and calls the lambda function for each of their element.

* **Output of map() function is list**

# Python lambda demo to filter out vowles from a list

alphabets = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i']

vowels = ['a', 'e', 'i', 'o', 'u']

output = list(filter(lambda x: (x in vowels) , alphabets))

# Output: ['a', 'e', 'i']

print(output)

# Python lambda demo to filter out vowles from a list

alphabets = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i']

vowels = ['a', 'e', 'i', 'o', 'u']

output = list(filter(lambda x: (x in vowels) , alphabets))

# Output: ['a', 'e', 'i']

print(output)

**Reduce() Function**

* The reduce method continuously applies a function on an iterable (such as a list) until there are no items left in the list. It produces a non-iterable result, i.e., returns a single value.
* helps in aggregating data from a list and returning the result

**Syntax:**

reduce(func\_obj, iterable[, initializer])

**Example:**

from functools import reduce

def fn(m, n) : return m + n

print(reduce((lambda m, n: m + n), [1, 2, 3, 4]))

print(reduce(fn, [1, 2, 3, 4]))

output:

10

10