Lambda is an anonymous function, it has no name, therefore can't be defined

Parameters & code (expression) of the function, in one line

In [ ]:
```
# lambda parameter:expression
```

In [11]:
```
g = lambda x: x*x*x
```

In [12]:
```
g(10)
```

Out[12]:
```
1000
```

Lambda with filter, map and reduced

Filter

The filter() method filters the given sequence with the help of a function that tests each element in the sequence to be true or not.

In [ ]:
```
# Syntax
#filter (function, sequence)
#function: function that tests if each element of a sequence true or not.
#sequence: sequence which needs to be filtered, it can be sets, lists, tuples, or containers of any iterators.
#Returns:returns an iterator that is already filtered.
```

In [24]:
```
#Eg - Function that filters vowels
def fun(variable):
    letters = ['a','e','i','o','u']
    if (variable in letters):
        return True
    else:
        return False
s1 = ['g', 'e', 'e', 'j', 'k', 's', 'p', 'r']
f1 = filter(fun,s1)
print('The filtered letters are:')
for j in f1:
    print(j)
```

```
The filtered letters are:
e
e
```

In [25]:
```
# Lambda with Filter
```

In [32]:
```
#Filter even numbers
l1 = [12,23,34,45,56,67,78,89,90,11]
Filtered_List = list (filter(lambda x: (x%2==0),l1))
Filtered_List
```

Out[32]:
```
[12, 34, 56, 78, 90]
```

In [ ]:
```
#Break down above code
#Filtered_List = list (filter(lambda x: (x%2==0),l1))
#Step 1: filter(function,sequence)
#Step 2: filter(lambda,l1)
#Step 3: filter(lambda(parameter:expression),l1)
#Step 4: filter(lambda x: (x%2==0),l1)
# now put it in new list
#Step 5: list(filter(lambda x: (x%2==0),l1))
# now put new list into new variable
#Step 6: Filtered_List = list(filter(lambda x: (x%2==0),l1))
```

Map

map() function returns a map object(which is an iterator) of the results after applying the given function to each item of a given iterable (list, tuple etc.)

In [33]:
```
# Syntax
# map(function, iterable)
# function : It is a function to which map passes each element of given iterable.
# iterable : It is a iterable which is to be mapped.
```

Note : You can pass one or more iterable to the map() function.

In [ ]:
```
# Return :Returns a list of the results after applying the given function to each item of a given iterable (lis
```

Note : The returned value from map() (map object) then can be passed to functions like list() (to create a list), set() (to create a set) .

In [34]:
```
# Return Double of given number (n)
def addition (n):
    return n + n
numbers = (1,2,3,4)
Result = list (map(addition, numbers))
print (Result)
```

```
[2, 4, 6, 8]
```

In [5]:
```
# Multiply individual elements with 2
l1 = [1,2,3,4,5,6,7,8]
def mult (n):
    return n*2
Result = list(map(mult,l1))
print (Result)
```

```
[2, 4, 6, 8, 10, 12, 14, 16]
```

Reduce

Used when we need a single final result over a sequence.

In [6]:
```
from functools import reduce
```

In [9]:
```
sum = reduce(lambda x,y:x+y,l1)
sum
```

Out[9]:
```
36
```