MAP, Filter and Lambda

Why MAP ?

You are having a list of elements, you want to itereate through and pass that particular element into function, and generate another list of results

Step 1 - You all are having a iterable List Step 2 - You have to send each element of the list to functions step 3 - collect all the outputs from the function and store in new list

In [1]:
```
#Demo Function
def getPowerFive(num):
    return num * num * num * num * num
```

In [ ]:
```
```

In [3]:
```
# iterable Object LIST
lst = [1,2,3,4,5,6,7,8,9,10]
```

In [4]:
```
#Approch Number 1 Trad.
newLst = []
for item in lst:
    temp = getPowerFive(item)
    newLst.append(temp)
```

In [5]:
```
newLst
```

Out[5]:
```
[1, 32, 243, 1024, 3125, 7776, 16807, 32768, 59049, 100000]
```

In [6]:
```
# Using Maps - Approch 2
newMappedList = map(getPowerFive, lst)
```

In [7]:
```
newMappedList
```

Out[7]:
```
<map at 0x10ce62790>
```

In [8]:
```
list(newMappedList)
```

Out[8]:
```
[1, 32, 243, 1024, 3125, 7776, 16807, 32768, 59049, 100000]
```

Map is used for generating new list based on old list and a function which you want to use for generating new list items,

you dont need to write whole for loop and temp code, just Use Map to get all the elements of the old list mapped with function output in the new list

Filter

Question ??

- Step 1 - you all have some iteratable object
- Step 2 - You want to make a new list which contains of all the elements from first list which returns TRUE when passed to a Function
- Step 3 - print that new list with only filtered elements

In [13]:
```
def isOdd(num):
    if num%2 == 1:
        return True
    else:
        return False
```

In [15]:
```
numList = [34,56,91,24,56,78,74,68,83,71,67,34,54,21]
```

In [23]:
```
#Approch number 1 - for filtering all the ODD Numbers from the List
subList = []
for item in numList:
    if isOdd(item) == True:
        subList.append(item)
```

In [24]:
```
subList
```

Out[24]:
```
[91, 83, 71, 67, 21]
```

In [25]:
```
# Approh 2 - Filtering all the odd numbers from the list
subFilterList = filter(isOdd,numList)
```

In [22]:
```
list(subFilterList)
```

Out[22]:
```
[91, 83, 71, 67, 21]
```

Lambda Exp.

- Lambda Expressions are annonymous functions
- they can be written in one line
- and these are the functions which can be used for on-the go job
- you dont require to change anything in the backend for using Inline function

In [26]:
```
def isEven(num):
    if num%2==0:
        return True
    else:
        return False
```

In [27]:
```
def getPowerSeven(num):
    return num ** 7
```

In [28]:
```
getPowerSeven(7)
```

Out[28]:
```
823543
```

In [30]:
```
getPowerSevenV2 = lambda num: num ** 7
```

In [31]:
```
getPowerSevenV2(7)
```

Out[31]:
```
823543
```

In [32]:
```
lst = [45,55,65,75,85,95,15,25,35]
```

In [33]:
```
newSquareList = list(map(lambda num: num*num, lst))
```

In [34]:
```
newSquareList
```

Out[34]:
```
[2025, 3025, 4225, 5625, 7225, 9025, 225, 625, 1225]
```

In [35]:
```
numList = [34,56,91,24,56,78,74,68,83,71,67,34,54,21]
```

In [39]:
```
isEven = lambda num : num%2==0
newFilterdEvenList=list(filter(isEven, numList))
```

In [40]:
```
newFilterdEvenList
```

Out[40]:
```
[34, 56, 24, 56, 78, 74, 68, 34, 54]
```

In [41]:
```
isEven(97)
```

Out[41]:
```
False
```

In [ ]:
```

```