* **Map函数：**

原型：map(function, sequence)，作用是将一个列表映射到另一个列表，

使用方法：

def f(x):

    return x\*\*2

l = range(1,10)

map(f,l)

*Out[3]: [1, 4, 9, 16, 25, 36, 49, 64, 81]*

*实现：*

>>> def map(func,seq):

mapped\_seq = []

for eachItem in seq:

mapped\_seq.append(func(eachItem))

return mapped\_seq

* **Reduce函数**

原型：reduce(function, sequence, startValue)，作用是将一个列表归纳为一个输出，

使用方法：

def f2(x,y):

    return x+y

reduce(f2,l)

*Out[7]: 45*

reduce(f2,l,10)

*Out[8]: 55*

*实现：*

def reduce(bin\_func, seq, initial=None):

lseq = list(seq)

if initial is None:

res = lseq.pop(0)

else:

res = initial

for eachItem in lseq:

res = bin\_func(res,eachItem)

return res

* **Filter函数**

原型：filter(function, sequence)，作用是按照所定义的函数过滤掉列表中的一些元素，

使用方法：

def f2(x):

    return x%2 != 0

filter(f2,l)

*Out[5]: [1, 3, 5, 7, 9]*

记住：这里的function必须返回布尔值。

>>> def filter(bool\_func,seq):

filtered\_seq = []

for eachItem in seq:

if bool\_func(eachItem):

filtered\_seq.append(eachItem)

return filtered\_seq

* **Lambda函数**

原型：lambda <参数>: 函数体，隐函数，定义一些简单的操作，

使用方法：

f3 = lambda x: x\*\*2

f3(2)

*Out[10]: 4*

还可以结合map、reduce、filter来使用，如：

map(f3,l)

Out[11]: [1, 4, 9, 16, 25, 36, 49, 64, 81]

* **列表推导式**

基本形式：[expression for item in sequence <if (conditions)>], 这里x表示对item的操作，

使用方法：

[i\*\*2 for i in l]

*Out[12]: [1, 4, 9, 16, 25, 36, 49, 64, 81]*