#6.补充Python函数编程内容：

* **函数式编程** 属于声明式编程中的一种，它的主要思想是 **将计算机运算看作为函数的计算**，也就是把程序问题抽象成数学问题去解决。
* 函数式编程中，我们可以充分利用数学公式来解决问题。也就是说，任何问题都可以通过函数（加减乘除）和数学定律（交换律、结合律等），一步一步计算，最终得到答案。
* Y=kx+b

优势：

#6.1map 函数将一个函数依次作用或映射到序列或迭代器对象的每个元素上，并返回一个

map 对象作为结果

案例 1:

list(map(str,range(5))) #转化为字符串

案例 2:

def add1(v):

return v+5

print list(map( ,range(10)))#将单参数函数映射到 1 个序列上

#案例 3：

def add(x,y):

return x+y

list(map(add,range(5),range(5,10)))#将双参数函数映射到两个序列上

#等同于 print list(map(lambda x,y:x+y,range(5),range(5,10)))

#等同于 print [add(x,y) for x,y in zip(range(5),range(5,10))]

#6.2reduce 函数

#标准库 functions 中的函数 reduce()可以将一个接受 2 个参数以累积的方式

# 从左到右一次作用到一个序列或迭代器对象的所有元素上

#在 Python2 中 reduce 是内置函数，不需要导入任何模块即可使用,py3 中需要导入包

from functools import reduce

print reduce(add,range(10)) #add 是上面定义的函数

seq=[1,2,3,4,5,6,7,8,9]

#将 lambda 函数应用在 seq 序列上

print reduce(lambda x,y:x+y,seq)

#6.3filter 函数将一个单参数函数作用到一个序列上，返回该序列中

# 使得该函数返回值为 True 的那些元素组成的 filter 对象

#如果指定函数为 None，则返回序列中等价于 True 的元素

#下面代码在 python3 中使用

# seq1=['foo','x41','?1','\*\*\*']

# def func(x):

return x.isalnum()

# filter(func,seq) #返回 filter 对象

# print list(filter(func,seq)) #将 filter 转化为列表

# #使用列表推到式实现相同的功能

# print [x for x in seq if x.isalnum()]

# #使用 lambda 表达式实现相同功能

# print list(filter(lambda x:x.isalnum(),seq))

# print list(filter(None,[1,2,3,0,0,4,0,5]))

#6.5zip 函数:用于将多个列表中元素重新组合为元组并返回包含这些元组的 zip 对象

#enumerate()函数返回包含若干下标和值的迭代对象

x=list(range(10))

import random

random.shuffle(x)

print x

print max(x),min(x),sum(x),len(x)

print list(zip(x,[1]\*11)) #zip 函数也可以用于一个序列或迭代对象

print list(zip(range(1,4)))

print list(zip(['a','b','c'],[1,2])) #如果两个列表不等长，以短的为主

print enumerate(x) #枚举列表对象，返回 enumerate 对象

print list(enumerate(x)) #enumreate 对象可迭代

#6.6sort()和 reverse()

print "原来的：",x

g=lambda item:len(str(item))

print g(x) #30?

print str(x) #[0, 3, 2, 8, 1, 5, 4, 7, 6, 9]

print len(str(x)) #30? len 指的是列表元素的个数

x.sort(key=lambda item:len(str(item)),reverse=True) #按指定规则排序

print "排序的：",x

x.reverse()

print "逆序的",x

x.sort(key=str) #按转化为字符串后的大小排序

print "按转化为字符串的大小排序",x

最后我们可以看到，函数式编程有如下好处：

1）代码更简单了。  
2）数据集，操作，返回值都放到了一起。  
3）你在读代码的时候，没有了循环体，于是就可以少了些临时变量，以及变量倒来倒去逻辑。  
4）你的代码变成了在描述你要干什么，而不是怎么去干。