**【Python】 sorted函数**

我们需要对List、Dict进行排序，Python提供了两个方法  
对给定的List L进行排序，  
方法1.用List的成员函数sort进行排序，在本地进行排序，不返回副本  
方法2.用built-in函数sorted进行排序（从2.4开始），返回副本，原始输入不变  
  
--------------------------------sorted---------------------------------------  
>>> help(sorted)  
Help on built-in function sorted in module \_\_builtin\_\_:  
  
sorted(...)  
    sorted(iterable, cmp=None, key=None, reverse=False) --> new sorted list  
---------------------------------sort----------------------------------------  
>>> help(list.sort)  
Help on method\_descriptor:  
  
sort(...)  
    L.sort(cmp=None, key=None, reverse=False) -- stable sort \*IN PLACE\*;  
    cmp(x, y) -> -1, 0, 1  
-----------------------------------------------------------------------------  
  
iterable：是可迭代类型;  
cmp：用于比较的函数，比较什么由key决定;  
key：用列表元素的某个属性或函数进行作为关键字，有默认值，迭代集合中的一项;  
reverse：排序规则. reverse = True  降序 或者 reverse = False 升序，有默认值。  
返回值：是一个经过排序的可迭代类型，与iterable一样。

参数说明：

(1)  cmp参数

cmp接受一个函数，拿整形举例，形式为:

def f(a,b):

     return a-b

如果排序的元素是其他类型的，如果a逻辑小于b，函数返回负数；a逻辑等于b，函数返回0；a逻辑大于b，函数返回正数就行了

(2)  key参数

 key也是接受一个函数，不同的是，这个函数只接受一个元素,形式如下

def f(a):

     return len(a)

key接受的函数返回值，表示此元素的权值，sort将按照权值大小进行排序

(3) reverse参数

接受False 或者True 表示是否逆序

例子：

(1)按照元素长度排序

L = [{1:5,3:4},{1:3,6:3},{1:1,2:4,5:6},{1:9}]

def **f**(x):

    return len(x)

sort(key=f)

print L

输出:

[{1: 9}, {1: 5, 3: 4}, {1: 3, 6: 3}, {1: 1, 2: 4, 5: 6}]

(2)按照每个字典元素里面key为1的元素的值排序

L = [{1:5,3:4},{1:3,6:3},{1:1,2:4,5:6},{1:9}]

def **f2**(a,b):

    return a[1]-b[1]

L.sort(cmp=f2)

print L

. 对**由tuple组成的List**排序

**Python代码**

1. >>> students = [('john', 'A', 15), ('jane', 'B', 12), ('dave', 'B', 10),]

用key函数排序：**返回由tuple组成的list**

**Python代码**

1. >>> sorted(students, key=lambda student : student[2])   # sort by age
2. [('dave', 'B', 10), ('jane', 'B', 12), ('john', 'A', 15)]

用cmp函数排序

**Python代码**

1. >>> sorted(students, cmp=lambda x,y : cmp(x[2], y[2])) # sort by age
2. [('dave', 'B', 10), ('jane', 'B', 12), ('john', 'A', 15)]

**用 operator 函数来加快速度,**

**Python代码**

1. >>> from operator import itemgetter, attrgetter
2. >>> sorted(students, key=itemgetter(2))

用 operator 函数进行多级排序

**Python代码**

1. >>> sorted(students, key=itemgetter(1,2))  # sort by grade then by age
2. [('john', 'A', 15), ('dave', 'B', 10), ('jane', 'B', 12)]

**2. 对由字典排序 ，返回由tuple组成的List,不再是字典。**

**Python代码**

1. >>> d = {'data1':3, 'data2':1, 'data3':2, 'data4':4}
2. >>> sorted(d.iteritems(), key=itemgetter(1), reverse=True)
3. [('data4', 4), ('data1', 3), ('data3', 2), ('data2', 1)]