简单记一下python中List的sort方法（或者sorted内建函数）的用法。   
  
关键字：   
python列表排序 python字典排序 sorted   
  
  
List的元素可以是各种东西，字符串，字典，自己定义的类等。   
  
sorted函数用法如下：

**Python代码  [收藏代码](javascript:void())**

1. sorted(data, cmp=None, key=None, reverse=False)

其中，data是待排序数据，可以使List或者iterator, cmp和key都是函数，这两个函数作用与data的元素上产生一个结果，sorted方法根据这个结果来排序。   
cmp(e1, e2) 是带两个参数的比较函数, 返回值: 负数: e1 < e2, 0: e1 == e2, 正数: e1 > e2. 默认为 None, 即用内建的比较函数.   
key 是带一个参数的函数, 用来为每个元素提取比较值. 默认为 None, 即直接比较每个元素.   
通常, key 和 reverse 比 cmp 快很多, 因为对每个元素它们只处理一次; 而 cmp 会处理多次.   
  
通过例子来说明sorted的用法：   
  
1. 对由tuple组成的List排序

**Python代码  [收藏代码](javascript:void())**

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

用key函数排序(lambda的用法见 注释1)

**Python代码  [收藏代码](javascript:void())**

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  [收藏代码](javascript:void())**

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, 上面排序等价于:(itemgetter的用法见 注释2)

**Python代码  [收藏代码](javascript:void())**

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  [收藏代码](javascript:void())**

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

**Python代码  [收藏代码](javascript:void())**

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)]

注释1   
参考：http://jasonwu.me/2011/10/29/introduce-to-python-lambda.html   
  
注释2   
参考:http://ar.newsmth.net/thread-90745710c90cf1.html   
class itemgetter(\_\_builtin\_\_.object)   
|  itemgetter(item, ...) --> itemgetter object   
|   
|  Return a callable object that fetches the given item(s) from its operand.   
|  After, f=itemgetter(2), the call f(r) returns r[2].   
|  After, g=itemgetter(2,5,3), the call g(r) returns (r[2], r[5], r[3])   
  
相当于

**Python代码  [收藏代码](javascript:void())**

1. **def** itemgetter(i,\*a):
2. **def** func(obj):
3. r = obj[i]
4. **if** a:
5. r = (r,) + tuple(obj[i] **for** i **in** a)
6. **return** r
7. **return** func
9. >>> a = [1,2,3]
10. >>> b=operator.itemgetter(1)
11. >>> b(a)
12. 2
13. >>> b=operator.itemgetter(1,0)
14. >>> b(a)
15. (2, 1)
16. >>> b=itemgetter(1)
17. >>> b(a)
18. 2
19. >>> b=itemgetter(1,0)
20. >>> b(a)
21. (2, 1)