sorted() 函数对所有可迭代的对象进行排序操作。

    sort 与 sorted 区别：

    sort 是应用在 list 上的方法，sorted 可以对所有可迭代的对象进行排序操作。

    list 的 sort 方法返回的是对已经存在的列表进行操作，无返回值，而内建函数 sorted 方法返回的是一个新的 list，而不是在原来的基础上进行的操作。

语法

sorted 语法：

sorted(iterable, cmp=None, key=None, reverse=False)

参数说明：

    iterable -- 可迭代对象。

    cmp -- 比较的函数，这个具有两个参数，参数的值都是从可迭代对象中取出，此函数必须遵守的规则为，大于则返回1，小于则返回-1，等于则返回0。

    key -- 主要是用来进行比较的元素，只有一个参数，具体的函数的参数就是取自于可迭代对象中，指定可迭代对象中的一个元素来进行排序。

    reverse -- 排序规则，reverse = True 降序 ， reverse = False 升序（默认）。

返回值

返回重新排序的列表。

实例

以下实例展示了 sorted 的使用方法：

>>>a = [5,7,6,3,4,1,2]

>>> b = sorted(a)       # 保留原列表

>>> a

[5, 7, 6, 3, 4, 1, 2]

>>> b

[1, 2, 3, 4, 5, 6, 7]

>>> L=[('b',2),('a',1),('c',3),('d',4)]

>>> sorted(L, cmp=lambda x,y:cmp(x[1],y[1]))   # 利用cmp函数

[('a', 1), ('b', 2), ('c', 3), ('d', 4)]

>>> sorted(L, key=lambda x:x[1])               # 利用key

[('a', 1), ('b', 2), ('c', 3), ('d', 4)]

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

>>> sorted(students, key=lambda s: s[2])            # 按年龄排序

[('dave', 'B', 10), ('jane', 'B', 12), ('john', 'A', 15)]

>>> sorted(students, key=lambda s: s[2], reverse=True)       # 按降序

[('john', 'A', 15), ('jane', 'B', 12), ('dave', 'B', 10)]

>>>