1.自己实现一个排序算法，不能使用python内置的sorted和sort，具体哪种排序算法不限；

函数接口：mysort(data)

可选部分：【 对于有一定基础的同学，可以考虑扩展接口如下

mysort(data,key=somefunc,reveresed=True|False）

支持自定义比较函数，比如按照sin(x)或者abs(x)结果排序这样；

支持正序或者逆序排序；

】

2.实现测试用例：

3. 实现wordcount, 自己找一篇英文文章或者句子，统计每个单词出现次数，并使用1中的排序算法输出排序后的结果。

1.

**def** func(x, y):  
 **if** (x > y):  
 **return** 1  
 **else**:  
 **return** -1  
  
**def** sin\_func(x, y):  
 **if** (math.sin(x) > math.sin(y)):  
 **return** 1  
 **else**:  
 **return** -1  
  
**def** cos\_func(x, y):  
 **if** (math.cos(x) > math.cos(y)):  
 **return** 1  
 **else**:  
 **return** -1  
  
**def** mysort(data, key=func, reveresed=False):  
 **for** j **in** xrange(len(data), -1, -1):  
 **for** i **in** xrange(0, j - 1, 1):  
 **if** key(data[i], data[i + 1]) > 0:  
 data[i], data[i + 1] = data[i + 1], data[i]  
 **if** reveresed == True:  
 data.reverse()  
 **return** data

2.

**if** \_\_name\_\_ == **"\_\_main\_\_"**:  
 a = [1,2,6,4,543,4]  
 b = mysort(a, func, False)  
 **print** b  
 b = mysort(a, func, True)  
 **print** b  
 b = mysort(a, sin\_func, False)  
 **print** b  
 b = mysort(a, sin\_func, True)  
 **print** b  
 b = mysort(a, cos\_func, False)  
 **print** b  
 b = mysort(a, cos\_func, True)  
 **print** b

输出：

[1, 2, 4, 4, 6, 543]

[543, 6, 4, 4, 2, 1]

[4, 4, 6, 543, 1, 2]

[2, 1, 543, 6, 4, 4]

[543, 4, 4, 2, 1, 6]

[6, 1, 2, 4, 4, 543]

3.

**def** readFile(filename):  
 f = open(filename, **'r'**)  
 y = []  
 x = f.readlines()  
 *# 获取word* **for** line **in** x:  
 y.extend(line.split())  
 f.close()  
 word\_list = []  
  
 *# 过滤不需要的字符* **for** word **in** y:  
 word1 = word  
 **while** True:  
 lastchar = word1[-1:]  
 **if** lastchar **in** [**","**, **"."**, **"!"**, **"?"**, **";"**, **'"'**]:  
 word2 = word1.rstrip(lastchar)  
 word1 = word2  
 **else**:  
 word2 = word1  
 **break  
  
 while** True:  
 firstchar = word2[0]  
 **if** firstchar **in** [**","**, **"."**, **"!"**, **"?"**, **";"**, **'"'**]:  
 word3 = word2.lstrip(firstchar)  
 word2 = word3  
 **else**:  
 word3 = word2  
 **break** word\_list.append(word3)  
  
 *# 计算词频* freq\_list = []  
 word\_saved = []  
 **for** word **in** word\_list:  
 **if not** word **in** word\_saved:  
 word\_saved.append(word)  
 freq\_list.append((word, word\_list.count(word)))  
 **return** freq\_list

**if** \_\_name\_\_ == **"\_\_main\_\_"**:

freq\_list = readFile(**"wordcount.txt"**)sort\_list = mysort(freq\_list, (**lambda** x, y: 1 **if** x[1] > y[1] **else** -1), True)  
**print** sort\_list

输出:

[('is', 13), ('of', 11), ('a', 11), ('to', 9), ('and', 9), ('are', 9), ('happy', 7), ('the', 7), ('It', 4), ('as', 4), ('not', 4), ('you', 3), ('it', 3), ('Being', 3), ('people', 3), ('duty', 3), ('an', 3), ('hashable', 3), ('since', 3), ('set-like', 3), ('wider', 2), ('You', 2), ('will', 2), ('be', 2), ('others', 2), ('find', 2), ('for', 2), ('There', 2), ('being', 2), ('in', 2), ('into', 2), ('like', 2), ('Happy', 2), ('or', 2), ('set', 2), ('Then', 2), ('view', 2), ('that', 2), ('so', 2), ('all', 2), ('unique', 2), ('entries', 2), ('views', 2), ('friends', 1), ('grateful', 1), ('with', 1), ('thronged', 1), ('gardens', 1), ('unimaginable', 1), ('doors', 1), ('opens', 1), ('habit', 1), ('established', 1), ('realized', 1), ('once', 1), ('service', 1), ('yourself', 1), ('forget', 1), ('can', 1), ('mind', 1), ('peace', 1), ('secret', 1), ('possess', 1), ('reality', 1), ('becomes', 1), ('make-believe', 1), ('good', 1), ('circles', 1), ('center', 1), ('rewarding', 1), ('deeply', 1), ('how', 1), ('discover', 1), ('them', 1), ('attract', 1), ('repelling', 1), ('instead', 1), ('long', 1), ('Before', 1), ('works', 1), ('pretend', 1), ('feel', 1), ('don\xa1\xaft', 1), ('if', 1), ('ridiculous', 1), ('glance', 1), ('first', 1), ('at', 1), ('seem', 1), ('simple', 1), ('cure', 1), ('however', 1), ('embittered', 1), ('miserable', 1), ('alone', 1), ('himself', 1), ('finds', 1), ('soon', 1), ('He', 1), ('sufferer', 1), ('from', 1), ('away', 1), ('shrink', 1), ('causes', 1), ('disease', 1), ('infectious', 1), ('unhappy', 1), ('ourselves', 1), ('indeed', 1), ('strive', 1), ('selfish', 1), ('character', 1), ('soul', 1), ('triumph', 1), ('accomplishment', 1), ('staying', 1), ('But', 1), ('dividend', 1), ('unexpected', 1), ('sort', 1), ('extremely', 1), ('who', 1), ('failures', 1), ('so-called', 1), ('invalids', 1), ('beggars', 1), ('we', 1), ('well-being', 1), ('physical', 1), ('wealth', 1), ('key', 1), ('he', 1), ('T', 1), ('reasons', 1), ('sorts', 1), ('happiness', 1), ('word', 1), ('definition', 1), ('exact', 1), ('no', 1), ('said', 1), ('has', 1), ('Stevenson', 1), ('As', 1), ('ripples', 1), ('circle', 1), ('ever-widening', 1), ('motion', 1), ('pool', 1), ('dropped', 1), ('pebble', 1), ('Happiness', 1), ('Door', 1), ('The', 1), ('set):', 1), ('another', 1), ('either', 1), ('refers', 1), ('(\xa1\xb0other\xa1\xb1', 1), ('available', 1), ('operations', 1), ('these', 1), ('unique.)', 1), ('generally', 1), ('treated', 1), ('(Values', 1), ('also', 1), ('items', 1), ('then', 1), ('pairs', 1), ('value)', 1), ('(key', 1), ('values', 1), ('If', 1), ('their', 1), ('Keys', 1)]