中国大学 MOOC 课程

《Python 语言程序设计》

课后练习（第 6 周）

北京理工大学

Python 语言教学团队

【说明】

本文是中国大学 MOOC 课程《Python 语言程序设计》第 6 周的课后练习，预估学习完成时间约 50 分钟。

本周课后练习内容包括 3 道编程题，帮助同学学习文件读写和字典操作。

对于尚未安装 Python 运行环境的同学，请根据第 1 周课程内容介绍的步骤安装 Python 3.5.1 或者 Python 3.5.2 版本解释器，如果操作系统兼容性有问题，可以安装 Python 3.4 版本解释器。

【课后练习】

1. 理解文本和二进制打开方式的区别

首先，用文本编辑器生成一个包含“中国是个伟大国家!”的 txt 格式文本文件，命名为 test.txt。编写程序分别用文本文件方式和二进制文件方式读入，并打印输出效果。

观察输出结果并解释。

2. 文件处理

```
fo = open(fname, "r")
for line in fo:
    # 处理一行数据
fo.close()
```

上述格式是打开文件并逐行处理的程序框架，请编写一个实例练习。

3. 哈姆雷特词频统计

Hamlet,《哈姆雷特》，是莎士比亚的一部经典悲剧作品，讲述了克劳狄斯叔叔谋害哈姆雷特父亲并篡取王位、哈姆雷特流浪在外并向叔叔复仇的故事。《哈姆雷特》也叫《王子复仇记》，代表着整个西方文艺复兴时期文学的最高成就，很多国内外电影都以这个故事为原型。获取该故事的文本文件，保存为 hamlet.txt。全文可以从 MOOC 课程下载或从网络获得。

统计 *Hamlet* 英文词频的第一步是分解并提取英文文章的单词。同

一个单词会存在大小写不同形式，但计数却不能区分大小写。假设 Hamlet 文本由变量 txt 表示，可以通过 txt.lower()函数将字母变成小写，排除原文大小写差异对词频统计的干扰。英文单词的分割可以是空格、标点符号或者特殊符号。为统一分隔方式，可以将各种特殊字符和标点符号使用 txt.replace()方法替换成空格，再提取单词。

统计词频的第二步是对每个单词进行计数。假设将单词保存在变量 word 中，使用一个字典类型 counts={}，统计单词出现的次数可采用如下代码：

```
counts[word] = counts[word] + 1
```

当遇到一个新词时，单词没有出现在字典结构中，则需要在字典中新建键值对：

```
counts[new_word] = 1
```

因此，无论词是否在字典中，加入字典 counts 中的处理逻辑可以统一表示为：

```
if word in counts:
    counts[word] = counts[word] + 1
else:
    counts[word] = 1
```

或者，这个处理逻辑可以更简洁的表示为如下代码：

```
counts[word] = counts.get(word,0) + 1
```

字典类型的 counts.get(word,0)方法表示：如果 word 在 counts 中，则返回 word 对应的值，如果 word 不在 counts 中，则返回 0。

该实例的第三步是对单词的统计值从高到低进行排序，输出前 10

个高频词语，并格式化打印输出。由于字典类型没有顺序，需要将其转换为有顺序的列表类型，再使用 sort()方法和 lambda 函数配合实现根据单词次数对元素进行排序。最后输出排序结果前 10 位的单词。

items = list(counts.items()) #将字典转换为记录列表

items.sort(key=lambda x:x[1], reverse=True) #以第 2 列排序

采用函数对获取和整理文本进行封装，下面给出该实例的完整代码。

实例代码　　CalHamlet.py

```
#CalHamlet.py
def getText():
    txt = open("hamlet.txt", "r").read()
    txt = txt.lower()
    for ch in '!"#$%&()*+,-./:;<=>?@[\\]^_‘{|}~':
        txt = txt.replace(ch, " ")   #将文本中特殊字符替换为空格
    return txt
hamletTxt = getText()
words  = hamletTxt.split()
counts = {}
for word in words:
    counts[word] = counts.get(word,0) + 1
items = list(counts.items())
items.sort(key=lambda x:x[1], reverse=True)
for i in range(10):
    word, count = items[i]
    print ("{0:<10}{1:>5}".format(word, count))
```

运行程序后，输出结果如下：

```
>>>
the       1138
```

```
and         965
to          754
of          669
you         550
a           542
i           542
my          514
hamlet      462
in          436
```

观察输出结果可以看到，高频单词大多数是冠词、代词、连接词等语法型词汇，并不能代表文章的含义。

请完成如下内容：

（1） 根据这个例子编写程序，完成上述功能和结果输出；

（2） 修改上述代码,采用集合类型构建一个排除词汇库 excludes，在输出结果中排除这个词汇库中内容，

4. 中文分词学习

对于一段英文文本，例如"China is a great country"，如果希望提取其中单词,只需要使用字符串处理的 split()方法即可,如下：

```
>>>"China is a great country".split()
['China', 'is', 'a', 'great', 'country']
```

然而，对于一段中文文本，例如"中国是一个伟大的国家"，希望获得其中的单词（不是字符）则十分困难，因为英文文本可以通过空格或者标点符号分割，而中文单词之间缺少分隔符，这是中文及类似

语言独有的“分词”问题。分词能够将"中国是一个伟大的国家"分为"中国"、"是"、"一个"、"伟大"、"的"、"国家"等一系列词语。

jieba（“结巴”）是 Python 中一个重要的第三方中文分词函数库，例子如下。

```
>>>import jieba
>>>jieba.lcut("中国是一个伟大的国家")
['中国', '是', '一个', '伟大', '的', '国家']
```

jieba 库是第三方库，不是安装包自带，因此，需要通过 pip 指令安装。pip 安装命令如下（请在命令行下执行，如果系统同时包括 Python 2.x 和 Python 3.x 系列版本，请用 pip3 命令）：

```
:\>pip install jieba  # 或者 pip3 install jieba
```

jieba 库的分词原理是利用一个中文词库，将待分词的内容与分词词库进行比对，通过图结构和动态规划方法找到最大概率的词组。除了分词，jieba 还提供增加自定义中文单词的功能。jieba.lcut()函数返回字符串分词后单词组成的列表。

文件、字典有哪些常用方法？