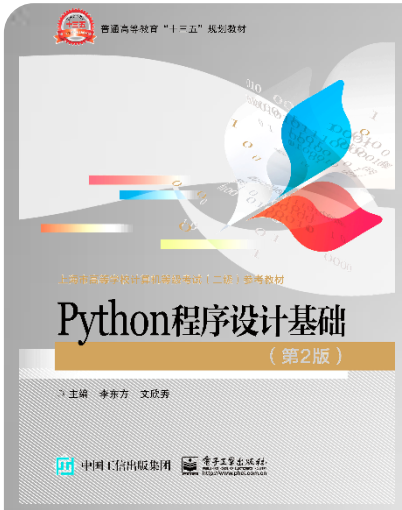
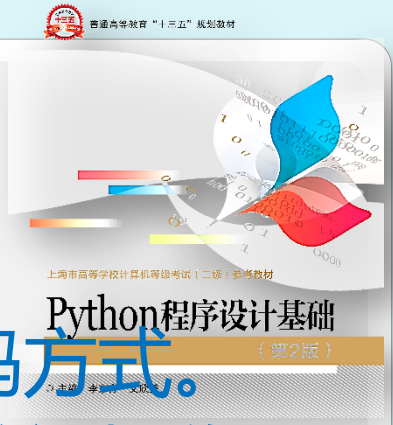


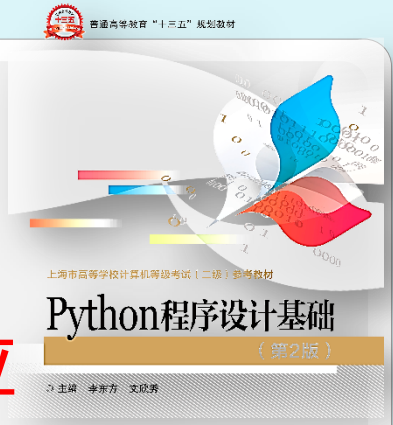
第5章 文件与基于文件的数据分析



本章教学目标：

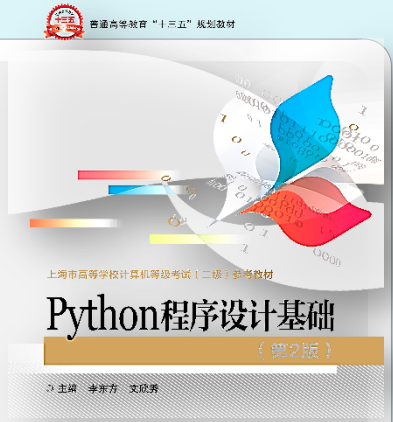
- 初步理解文件与目录的基本概念和编码方式。
- 理解文件的打开、定位、随机存取和关闭操作。
- 掌握文件的读取、写入和追加操作。
- 初步掌握基于文件的数据分析，学会利用第三方库进行中文词频分析。
- 了解利用第三方库wordcloud进行词语可视化。

5.1 基本概念



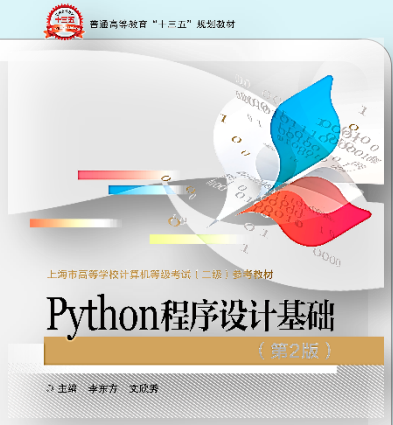
操作系统对数据进行管理是以文件为单位

- 访问磁盘等外存上的数据
  - 按文件名找到指定的文件
  - 再从该文件中读取数据
- 向外部介质上存储数据
  - 建立一个文件
  - 向其输出数据



5.1 基本概念

- 文件和目录
  - 文件是存储在外部介质上的数据集合，通常可以长久保存，也称为磁盘文件
  - 文件是通过目录来组织和管理的，目录提供了指向对应磁盘空间的路径地址
  - 目录一般采用树状结构，在这种结构中，每个磁盘有一个根目录，它包含若干文件和子目录。



5.1 基本概念

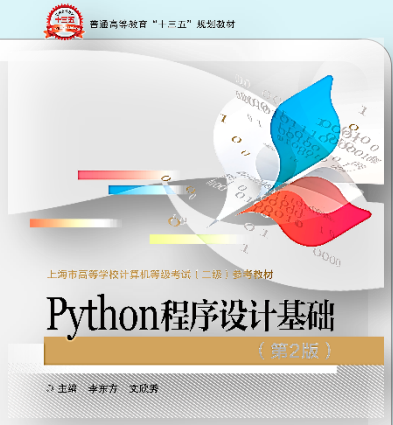
- 【例5-1】 绝对路径示例

假定文件file.txt保存在D:盘lecture5目录的ex子目录下，那么包含绝对路径的文件名是由磁盘驱动器、目录层次和文件名三部分组成的，即D:\lecture5\ex\file.txt

在Python中用字符串表示为：

"D:\\lecture5\\ex\\file.txt"　　或
"D:/lecture5/ex/file.txt"

5.1 基本概念



- 【例5-2】相对路径示例

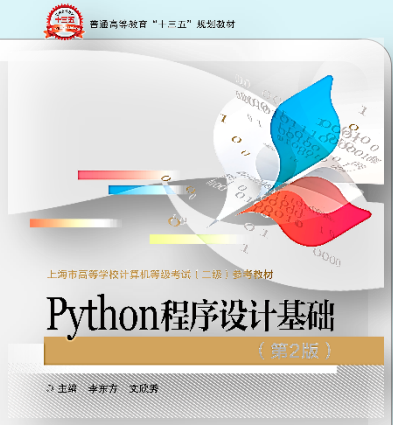
假定文件file.txt保存在D:盘的lecture5目录的ex子目录下，源程序保存在D:盘的lecture5目录下，那么包含相对路径的文件名表示为ex\file.txt

在Python中用字符串表示为：

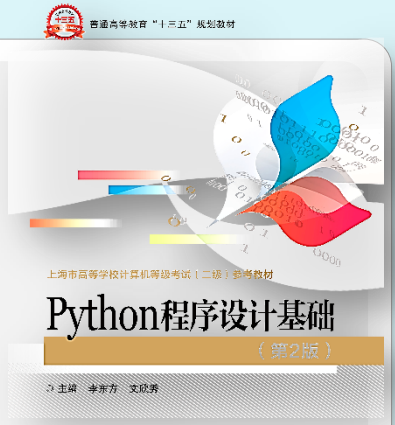
"ex\\file.txt"　　或

"ex/file.txt"

5.1 基本概念



- 文件的编码
  - 按照文件的编码方式
    - 文本文件
      基于字符编码的文件，在存取时需要编/解码
    - 二进制文件
      基于值编码的文件，存储的是二进制数据



5.1 基本概念

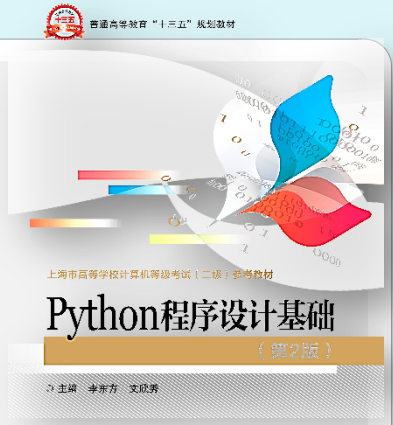
- Python语言的文件编码

Python 3.x版本中，文件的默认编码格式是UTF-8，字符串使用的是Unicode编码。所有的文本类型，都使用Unicode编码，可以直接使用str.encode()进行编码，encode()后可看到字符的UTF-8编码，再使用bytes.decode()可解码为文本

```
>>> s1="Unicode严谨编码"
>>> s1
'Unicode严谨编码'
>>> s2=s1.encode("utf-8")
>>> s2
b'Unicode\xe4\xb8\xa5\xe8\xb0\xa8\xe7\xbc\x96\xe7\xa0\x81'
>>> s2.decode("utf-8")
'Unicode严谨编码'
```

"严"的unicode是4E25（100111000100101）
"严"的UTF-8编码需要三个字节，即格式是"1110xxxx 10xxxxxx 10xxxxxx"
"严"的UTF-8编码是"11100100 10111000 10100101"

5.2 文件操作



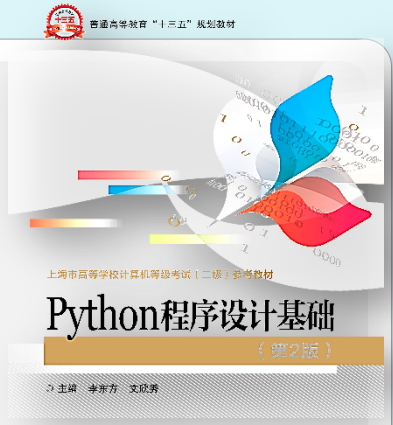
- 程序中对文件的操作
  - 打开文件
  - 读取文件
  - 对文件数据进行处理
  - 写入文件
  - 关闭文件
- 信息项

  构成文件内容的基本单位

  Python文本文件—字符

  Python二进制文件—字节

  读指针　　写指针



5.2 文件操作

- 文件的打开和关闭

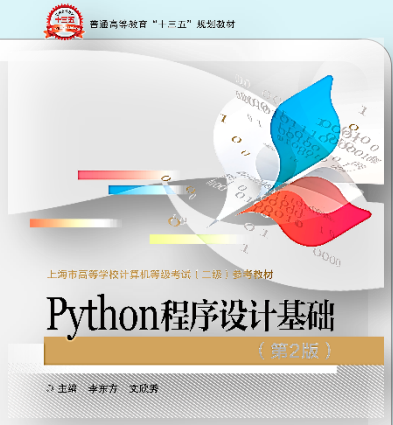
  <文件对象>= open(<文件名>[,<模式>])

  <文件对象>.close()

表 5-1 文件的打开方式

模　式	含　义
r	以只读方式打开
w	以写方式打开一个文件，若这个文件已存在，则覆盖原来的内容；若这个文件不存在，则创建这个文件
x	创建一个新文件，以写方式打开，若文件已存在，则报错 FileExistsError
a	以写方式打开，写入内容追加在文件的末尾
b	表示二进制文件，添加在其他控制字符后
t	表示文本文件，默认值
+	以修改方式打开，支持读/写

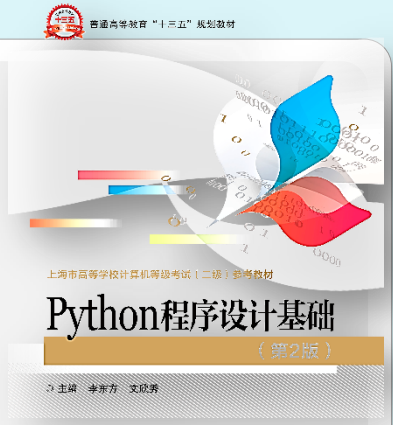
5.2 文件操作



- 【例5-4】 以读/写二进制模式打开当前目录下的文件示例。

```
>>> import os
>>> os.chdir('D:\\python')
>>> f = open('workfile.txt', 'rb+')
```

5.2 文件操作



- 定位

  文件定位语句seek()，可以帮助我们实现文件的随机读/写

  f.seek(<偏移值>[,<起始位置>])

  其中起始位置:

  为0表示自文件起始处开始(默认值，可省略)

  1表示从当前文件指针位置开始

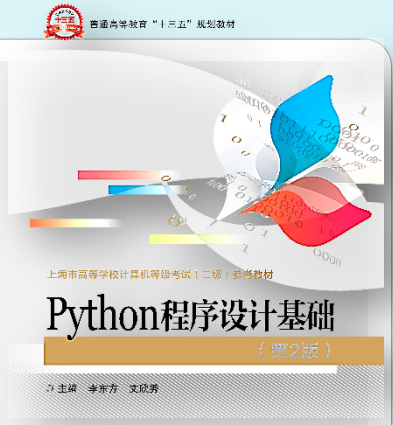
  2表示从文件末尾开始

  其中偏移值（单位为字节）：

  >0 从起始位置向右移动

  <0 从起始位置向左移动

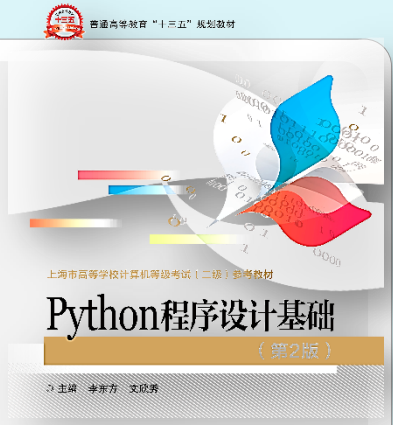
5.2 文件操作



- 【例5-5】 随机访问二进制文件示例

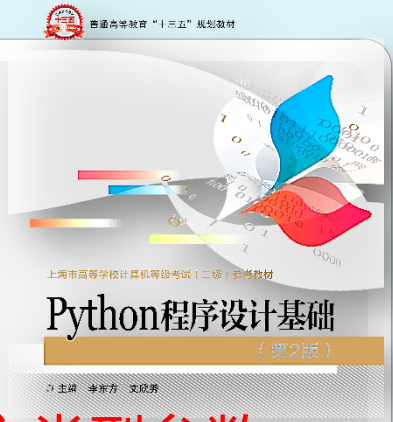
```
>>> import os
>>> os.chdir('d:\\python')
>>> f = open('workfile.txt', 'rb+')
>>> f.write(b'Be at war with your vices')
25
>>> f.seek(6)
6
>>> f.read(3)
b'war'
>>> f.seek(-5, 2)
20
>>> f.read(4)
b'vice'
>>> f.close()
```

5.2 文件操作



- 【例5-6】 随机访问文本文件示例

```
>>> import os
>>> os.chdir('d:\\python')
>>> f = open('workfile.txt', 'r+')
>>> f.write('微风送来淡淡花香')
8
>>> f.seek(8)
8
>>> f.read(4)
'淡淡花香'
>>> f.seek(-8,2)
Traceback (most recent call last):
  File "<pyshell#19>", line 1, in <module>
    f.seek(-8,2)
io.UnsupportedOperation: can't do nonzero
end-relative seeks
>>> f.seek(0)
0
>>> f.seek(8)
8
>>> f.read(4)
'淡淡花香'
>>> f.read(1)
''
>>> f.seek(8)
8
>>> f.read(4)
'淡淡花香'
>>> f.close()
```

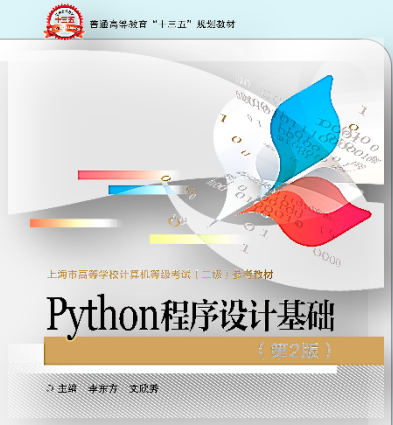


5.2 文件操作

- 文件的读取、写入、追加
  - f.read（size）方法

返回一个字符串，内容为长度为size的文本。数字类型参数size表示读取的字符数，可以省略。如果省略size参数，则表示读取文件所有内容并返回。如果已到达文件的末尾，f.read()将返回一个空字符串（''）。

```
>>> import os
>>> os.chdir('d:\\python')
>>> f = open('workfile.txt')
>>> f.read()
'宝剑锋从磨砺出\n梅花香自苦寒来\n'
>>> f.read()
''
```



5.2 文件操作

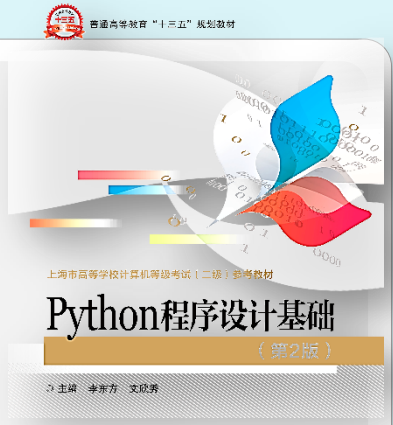
- 文件的读取、写入、追加
  - f.readline()方法

返回一个字符串，内容为文件的当前一行。换行符（\n）留在字符串的末尾。如果已到达文件的末尾，f.readline()将返回一个空字符串（''）。如果是一个空行，则返回'\n'。

```
>>> f.seek(0)
0
>>> f.readline()
'宝剑锋从磨砺出\n'
>>> f.readline()
'梅花香自苦寒来\n'
>>> f.readline()
''
```

```
>>> f.seek(0)
0
>>> for line in f:
print(line,end='')
宝剑锋从磨砺出
梅花香自苦寒来
```

从文件中读取行，更高效的方法是在文件对象上循环。这不但可以节省内存，而且代码也更简洁。

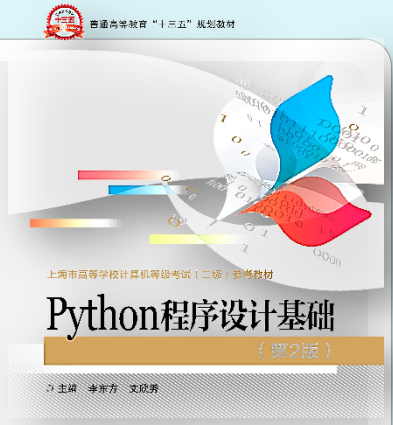


5.2 文件操作

- 文件的读取、写入、追加
  - f.readlines() 方法

返回一个列表，列表中的每个字符串类型元素对应文件的每行（包括结尾的换行符“\n”）。

```
>>> f.seek(0)
0
>>> f.readlines()
['宝剑锋从磨砺出\n', '梅花香自苦寒来\n']
>>> f.readlines()
[]
>>> f.close()
```



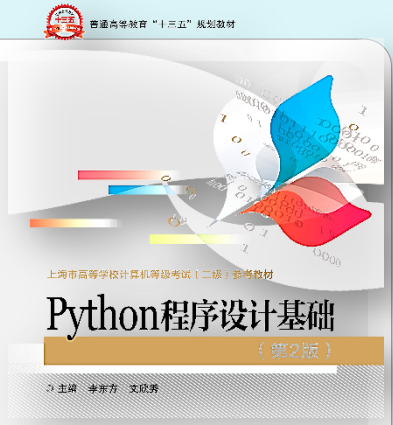
5.2 文件操作

- 文件的读取、写入、追加
  - 快速列表访问方式

<列表>=list(open(<文件名>))

```
>>> L=list(open('workfile.txt'))
>>> L
['宝剑锋从磨砺出\n', '梅花香自苦寒来\n']
```

5.2 文件操作

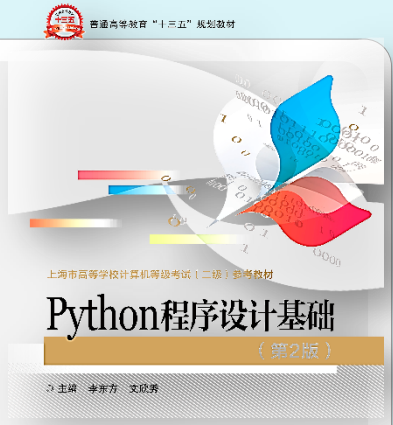


- 文件的读取、写入、追加
  - 将数据写入文件

f.write(string)方法将字符串string的内容写到f对应的文件中，并返回写入的字符数。但write语句不会自动换行，如果需要换行，则要使用换行符'\n'。

```
>>> f.close()
>>> f=open('workfile.txt','w')
>>> f.write('宝剑锋从磨砺出\n')
8
```

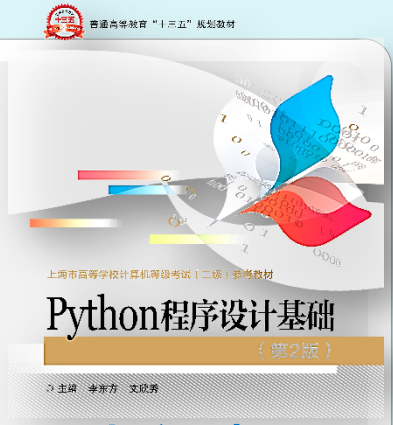
5.2 文件操作



- 文件的读取、写入、追加
  - 将数据追加到文件末尾

以a模式打开文件，指针会移到末尾处，写入的内容将追加到该文件中，但是必须关闭文件才能生效。

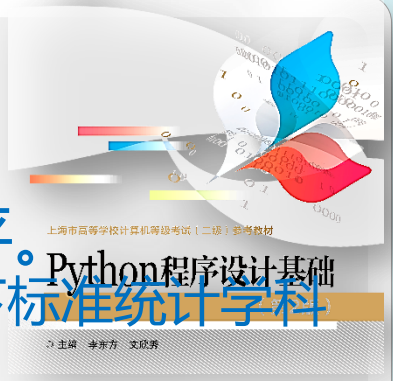
```
>>> f.close()
>>> f=open('workfile.txt','a')
>>> f.write('梅花香自苦寒来\n')
8
>>> f.close()
```



5.3 基于文件的数据分析

基于文件的数据分析，通常是利用Python对文本文件操作的便利性，读取文本文件，并转换为相应的数据列表，再利用循环结构实现统计分析。

5.3 基于文件的数据分析



- 【例5-7】 根据考试成绩，统计学科等级水平。

分析：某中学对学生的附加科目进行能力测试，并按以下标准统计学科等级水平。

（1）生物和科学两门课都达到60分，总分达到180分为及格；

（2）每门课达到85分，总分达到260分为优秀；

（3）总分不到180分或有任意一门课不到60分，为不及格。

编程要求：从score.txt文件中读取学生成绩数据，判定等级并写入level.txt文件中。

程序实现方案一：

（1）读取文件score.txt数据到列表L中

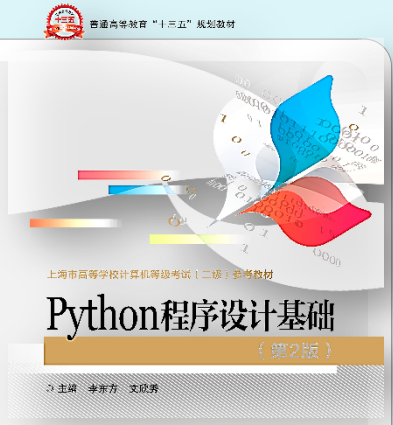
列表L中的数据项对应着文件中的每条学生记录，通过循环语句遍历L，提取需要的考号和三门课的成绩，并存放在列表x中。

（2）判定学科等级

列表x包含4个数据项，x[0]为考号，x[1]、x[2]和x[3]分别为“程序设计”、“生物”和“科学”三门课的成绩，需要转换为整数类型以便进行求和等数值运算。最后通过分支语句，将求得的等级结果存放在key变量中。

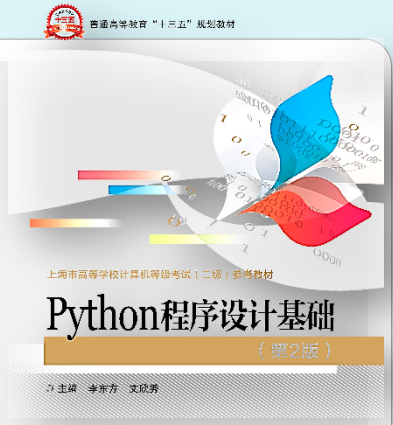
（3）将考号和等级结果按一定格式写入文件level.txt中。

5.3 基于文件的数据分析



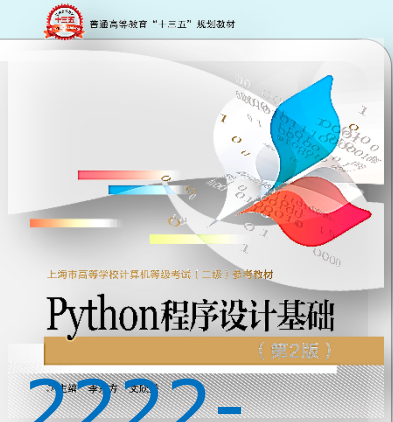

```
L=list(open('score.txt'))
f=open('level.txt','w')
del L[0]
for s in L:
    x=s.split()
    for i in range(1,len(x)):
        x[i]=int(x[i])
    sum=x[1]+x[2]+x[3]
    if x[1]>=85 and x[2]>=85 and x[3]>=85 and sum>=260:
        key='优秀'
    elif x[2]>=60 and x[3]>=60 and sum>=180:
        key='及格'
    else:
        key='不及格'
    f.write('%s\t%s\n'%(x[0],key))
f.close()
```

5.3 基于文件的数据分析



程序实现方案二：

```
s=open('score.txt')
f=open('level.txt','w')
s.readline()
while True:
    x=s.readline().split()
    if len(x)==0:
        break
    for i in range(1,len(x)):
        x[i]=int(x[i])
    sum=x[1]+x[2]+x[3]
    if x[1]>=85 and x[2]>=85 and x[3]>=85 and sum>=260:
        f.write('%s\t%s\n'%(x[0],'优秀'))
    elif x[2]>=60 and x[3]>=60 and sum>=180:
        f.write('%s\t%s\n'%(x[0],'及格'))
    else:
        f.write('%s\t%s\n'%(x[0],'不及格'))
s.close()
f.close()
```

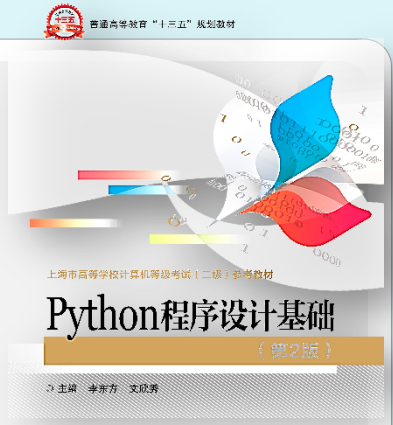


5.3 基于文件的数据分析

- 【例5-8】协助查找该日发生在北纬31.2222-31.2333，东经121.45-121.55区域内的案件，编写程序，找到并打印位于该区域内该出租车公司的车辆信息

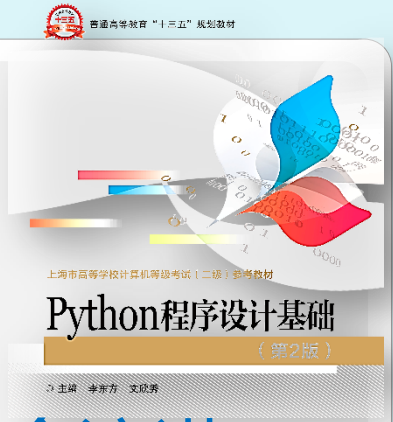
"car_data.txt"文件是以英文逗号分隔的数据文本文件（文件局部如图所示），记录了某出租汽车公司部分车辆某日0：00~23：00的车辆位置，无标题行。对应列分别是时间、车牌号、北纬、东经。

5.3 基于文件的数据分析




```
min_n, min_e =31.2222, 121.45
max_n, max_e =31.2333, 121.55
LS=list(open('car_data.txt'))
car=[]
for s in LS:
    carone=s[:-1].split(',')
    car.append(carone)
print('在该区间出现的车辆有：')
for t in range(len(car)):
if (min_n<float(car[t][2])<max_n) \
 and (min_e<float(car[t][3])<max_e):
        print('时间：%s\t车牌：%s\t北纬：%s,东经：%s'
         %(car[t][0],car[t][1],car[t][2],car[t][3]))
```

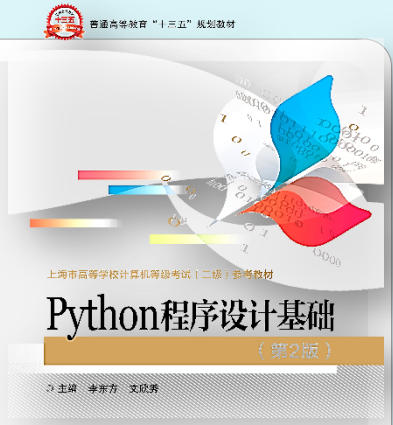
5.3.2 词频分析



- 【例5-9】统计著名黑人领袖马丁·路德金演讲“I Have a Dream”的词汇出现频次。

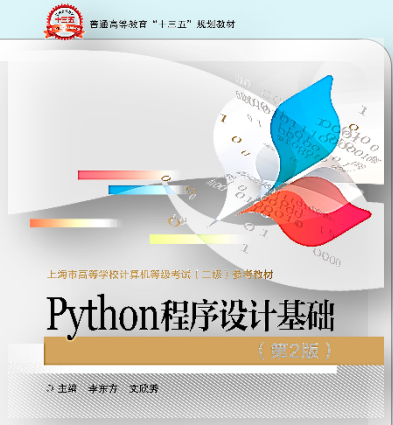
编程思想是：读取文本文件，用lower()方法将所有字符转为小写并用split()方法按空格分隔单词，将所有单词放在列表speech中。定义一个空字典dic，用循环结构遍历列表speech，将单词作为字典的键，统计每个单词出现的次数，作为字典的值

5.3.2 词频分析



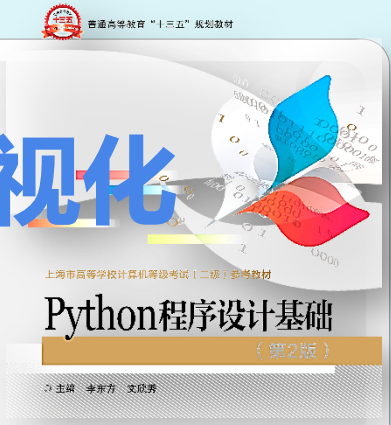

```
f=open('i_have_a_dream.txt')
speech_text=f.read()
f.close()
speech=speech_text.lower().split()
dic={}
for word in speech:
    if word not in dic:
        dic[word]=1
    else:
        dic[word]+=1
swd=sorted(list(dic.items()),key=lambda
lst:lst[1],reverse=True)
for kword,times in swd:
    print(kword,times)
```

5.3.2 词频分析




```
f=open('i_have_a_dream.txt')
speech_text=f.read()
f.close()
speech=speech_text.lower().split()
dic={}
for word in speech:
    if word not in dic:
        dic[word]=1
    else:
        dic[word]+=1
swd=sorted(list(dic.items()),key=lambda lst:lst[1],reverse=True)
f1=open('stop_word_list.txt')
stop_wds=f1.read()
f1.close()
for kword,times in swd:
    if kword not in stop_wds:
        print(kword,times)
```

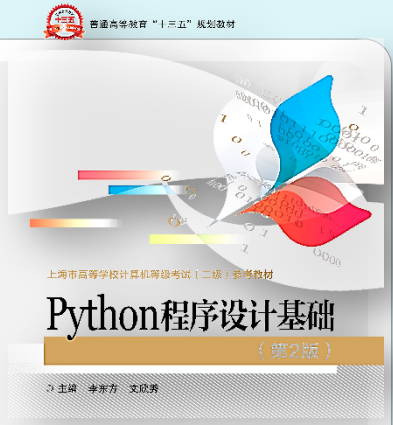
第三方库wordcloud和词语可视化



- 全自动安装
  easy_install jieba 或者 pip install jieba
- 半自动安装
  先下载https://pypi.python.org/pypi/jieba/ ，解压后运行
  python setup.py install
- 手动安装
  将 jieba 目录放置于当前目录或者 site-packages 目录
- 通过 import jieba 来引用

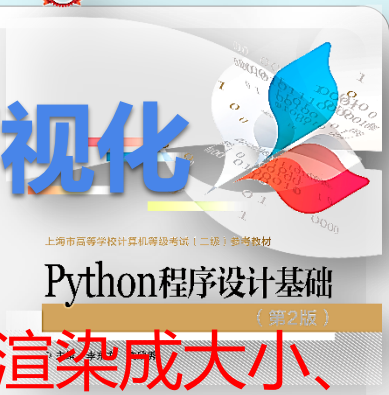
Python的第三方库jieba（“结巴”）是一个用于中文词汇分割的函数库，运用jieba.lcut()方法（0.39版本以上）可高效准确地实现将字符串中的中文词汇分割，精确返回词汇列表。

第三方库jieba和中文词频分析



【例5-10】统计朱自清散文“荷塘月色”的词汇出现频次

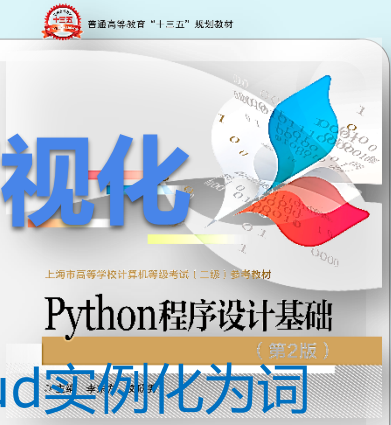
```
import jieba
f=open('荷塘月色.txt')
article_text=f.read()
f.close()
article=jieba.lcut(article_text)
dic={}
for word in article:
    if word not in dic:
        dic[word]=1
    else:
        dic[word]+=1
swd=sorted(list(dic.items()),key=lambda lst:lst[1],reverse=True)
f1=open('中文虚词列表.txt')
stop_wds=f1.read()
f1.close()
for kword,times in swd:
    if kword not in stop_wds:
        print(kword,times)
```



第三方库Wordcloud和词语可视化

- 词云库

Python的第三方库wordcloud是一种能将词语渲染成大小、颜色不一的可视化呈现形式“词云”的函数库。其效果能将枯燥呆板的文字以直观的艺术效果展示出来。



第三方库Wordcloud和词语可视化

- 创建词云
  - 先引用第三方库wordcloud，将其核心类WordCloud实例化为词云对象

    实例化对象的常用参数有：

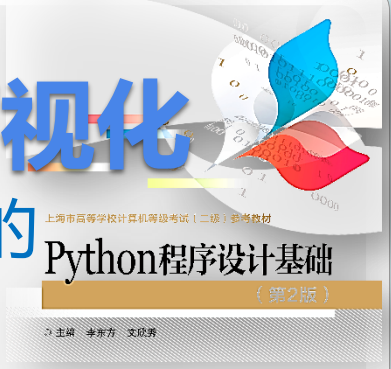
background_color	词云背景色，默认为黑
width,height	宽和高（像素）
font_path	字体文件的路径
max_font_size	最大字号
max_words	最多容纳词汇数，默认200

- 词云对象
  - genaerate()

    将文本生成词云
  - to_file()

    将词云保存为图片

第三方库Wordcloud和词语可视化



【例5-11】将上例中朱自清散文“荷塘月色”的词汇出现频次结果生成词云图

```
import wordcloud
txt='荷塘 采莲 今晚 路 叶子 想起 一条 这是 白天 树 知道 月光'
w=wordcloud.WordCloud(background_color='white',
                      width=150,
                      height=120,
                      max_font_size=48,
                      font_path='C:/Windows/Fonts/simhei.ttf')

w.generate(txt)
w.to_file('c:/test.png')
```