报告3参考答案（实验5 文件读写）

注：代码有多种写法，仅供参考。

1．将整数12345分别写入文本文件test.txt和二进制文件test.dat，并比较两个文件的不同之处。

a = 12345

with open("test.txt", "wt") as f: # 文本文件

f.write(str(a) )

import pickle

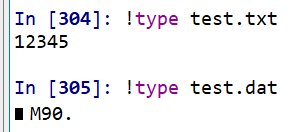
a = 12345

with open("test.dat", "wb") as f: # 写入pickle二进制文件

pickle.dump(a, f)

with open("test.dat", "rb") as f: # 读出pickle二进制文件

x = pickle.load(f)



test.txt是无格式的文本文件，test.dat是二进制文件，有特定格式。

2．以只写方式打开一个文本文件file.txt，将一个实数写入该文件，然后关闭文件。用只读方式打开同样的文件，读入一个实数，然后关闭该文件。最后输出读入的实数。

a = 12.345

with open("file.txt", "wt") as f:

f.write(str(a) )

with open("file.txt", "rt") as f:

b = eval(f.readline())

print(b)

print('a==b', a==b)

3．当前目录下有一个文本文件data.txt，文件的内容为若干件商品一月到六月的销售额，各月销售额之间用逗号分隔，查找并输出6个月总销售额最小的商品及其总销售额。data.txt数据示例如下，每行代表一个商品，每行6个数据。注：“data.txt”随书提供

*12,24,15,21,16,31*

*5,8,12,13,20,21*

*10,11,24,18,20,10*

lst = []

f = open("data.txt")

for i, line in enumerate(f):

alst = line.split(',')

alst = [int(s) for s in alst] # 转为整数

lst.append((i, sum(alst))) # 将 (行号, 该行的和) 构成的元组插入lst

f.close()

lst.sort(key=lambda x: x[1]) # 按元组第1项(即销售额之和) 排序

print("第 {}个货品有最小销售额{} ".format(lst[0][0], lst[0][1]))

4．当前目录下有一个test.docx的Word文档，读取该文件内容并统计其中每个汉字或字母出现的频次。注：该题是统计每个字，不是每个词，所以无需分词。

from docx import Document

doc = Document("test.docx")

txt = ""

for p in doc.paragraphs:

txt += p.text # 将所有段落连接为一个字符串

counted = {}

for ch in txt:

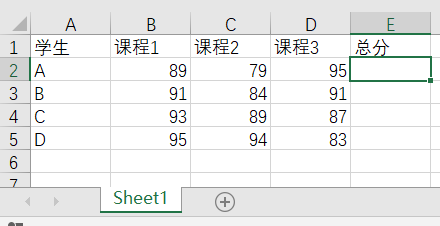
if not ch in counted.keys():

cnt = txt.count(ch) # 统计字的频次，使用了字符串的count函数来统计

counted[ch] = cnt

print(counted)

5．当前目录下有一个“成绩.xlsx”工作簿文件，第一列为学生姓名，第二列到第四列分别是学生的语文、数学和英语成绩。编写程序计算每名学生的总分，并输出成绩前三名的学生姓名。注：“成绩.xlsx”随书提供



# 方法一 只读数据，用python计算总分

from openpyxl import Workbook

from openpyxl import load\_workbook

wb = load\_workbook("成绩.xlsx")

ws = wb["Sheet1" ] # 选择Sheet1表

rows = ws.max\_row # 返回有数据的最大行数，所以不要在原表格后面随意插入新数据

i = 2 # 数据从第2行开始

std = { }

while i <= rows:

sm = ws.cell(i,2).value + ws.cell(i,3).value + ws.cell(i,4).value # 总分

std[ws.cell(i,1).value] = sm # 在字典中记录下 “姓名:总分”键值对

i += 1

lst = sorted(std.items(), key=lambda x:x[1], reverse=True) # 按总分逆序排列

print(lst[:3]) # 输出前3

上面的程序是在python中计算总分，另一种做法是将计算公式填入excel文件，希望excel来计算总分，但有几个细节要注意,如下所示：

# 方法二 在电子表格中填入公式，由excel计算结果

from openpyxl import Workbook

from openpyxl import load\_workbook

wb = load\_workbook("成绩.xlsx")

ws = wb["Sheet1" ] # 选择Sheet1表

rows = ws.max\_row # 返回有数据的最大行数

i = 2 # 数据从第2行开始

while i <= rows:

calc='=sum(B'+str(i)+':D'+str(i)+')' # 拼凑形如 "=sum(B2:D2)"的求和公式

ws['E'+str(i)] = calc # 将公式填入 E2单元格

i += 1

wb.save("成绩.xlsx") # 保存

注意：上面填入的公式并没有计算，必须用excel打开"成绩.xlsx"文件，略作改动（例如在某行后面随意输入空格）并保存，这样单元格中才能含有公式及其对应的计算结果。如果没有略作修改，后续读值时取出的是None。

后续如需读出公式计算的数值，则代码如下

wb = load\_workbook("成绩.xlsx"**, data\_only=True**) # data\_only表示读值而不是公式

ws = wb["Sheet1" ] # 选择Sheet1表

i=2

stu=[ ]

while i <= 5: # 读取第2-5行

stu.append((ws['A'+str(i)].value,ws['E'+str(i)].value)) # 读 (姓名,总分)

i=i+1

stu.sort(key=lambda x:x[1], reverse=True) # 按总分逆序排列

print(stu[:3]) # 输出前3

6．统计story.txt这篇英文小说中每个单词出现的频次，输出频次最高的10个单词。统计时剔除分割后可能存在的单个字母或标点符号。注：“story.txt”随书提供

f = open("story.txt" ) # 小说是文本格式

txt = f.read()

f.close()

txt = txt.lower() # 转小写

for c in ".,!;?:\n-())": # 将可能影响单词统计的标点符号都用空格替代

txt = txt.replace(c,' ') # 将字符c替换为空格

words = txt.split() # 分解为单个单词

words = [x for x in words if len(x)>1] # 剔除长度为1的单个字母或标点符号

counted = {}

for word in words:

counted[word] = counted.get(word, 0) + 1 # 以字典方式统计

sc = sorted(counted.items(), key=lambda x:x[1], reverse = True) # 从高到低排列

print(sc[:10]) # 频次最高的10个单词

注：如果需要找出所有的非英文字母的那些符号，可以用正则表达式。

import re # 正则模块

f = open("story.txt" )

txt = f.read()

f.close()

lst = re.findall('[^a-z]', txt, re.I) # [^a-z]表示非字母， re.I 忽略大小写

set1 = set(lst) # 集合去重

print(set1) # 所有的非英文字母的符号

7．从股票软件中导出某只股票的历史交易数据，计算收盘价的5日均价。5日均价是股票术语：5日均价 = 最近连续5个交易日收盘价之和/5。

注：stock.txt随书提供，每行仅包含一个收盘价数据。前4行数据不足5个，可以填充0.

close = [ ] # 原始收盘价

avg5 = [ ] # 5日均价

with open("stock.txt", "r") as f:

for i in f:

close.append( eval(i) ) # 原始数据

for i in range(len(close)):

if i <4:

avg5.append(0) # 前4个数据无法计算5日均价，填入0

else:

avg = round(sum(close[i-4 : i+1])/5, 2)

avg5.append(avg)

print(avg5)

计算5日均价的方法二

import pandas as pd # pandas 见课本第8章

s1 = pd.read\_csv('stock.txt', names=['close'], header=None) # 读数据文件

s2 = s1.rolling(5).mean() # 滑动窗口大小为5，计算平均值mean()

8．搜索一篇有关“电子烟”的中文新闻报道，将网页内容复制到smoker.txt文本文件中。利用jieba库对新闻报道做分词处理，并做出词云图（注意：首先要执行如下两个命令安装需要的库）。注：smoker.txt 文件见随书配套的数据文件。

pip install jieba # 安装jieba分词库

pip install wordcloud # 安装词云库

参考代码如下：

from wordcloud import WordCloud # 词云库

import jieba # 分词库

def wc(note, fname, bgcolor="white"): # 产生词云图的函数

wc = WordCloud(background\_color=bgcolor, font\_path='simfang.ttf') # 背景色，仿宋体

wc.generate(note) # 生成词云图片

wc.to\_file(fname) # 保存为文件

with open('smoker.txt') as f: # 打开保存的新闻文件smoker.txt

s = f.read()

# 这篇文章是关于“电子烟”的新闻报道，向jieba库中临时添加新词“电子烟”

# 如不添加新词，“电子烟”将被拆分为“电子”和“烟”两个词

jieba.add\_word('电子烟')

wd = jieba.lcut(s) # 分词，得到词列表

words = [x for x in wd if len(x)>1] # 剔除空格、单字

s = ' '.join(words) # 将词列表连接为空格分隔的字符串（' '中间必须有一个空格）

wc(s, 'wc.png') # 调用函数，在当前目录中生成词云图片wc.png

生成词云后，在IPython交互窗口继续探索研究得到的分词列表。

In: len(wd), len(words) # 初始分词的词汇数、剔除了空格和单字后的词汇数

Out: (228, 132)

In: from collections import Counter # 引入统计函数Counter

In: dct=Counter(words) # 对words做词频统计，返回类似字典的统计结果

In: dct.most\_common(5) # 输出词频最高的5个词

Out: [('电子烟', 7), ('美国', 5), ('病例', 3), ('这些', 2), ('四分之三', 2)]

9. 统计babynames目录下的所有50个数据文件，计算50年间使用频次最高的男性和女性的姓名分别是什么？

boy=[ ] # 存放所有男孩

girl=[ ] # 女孩

for k in range(1, 51):

# 拼凑得到数据文件名

fname='babynames/yob' + str(1950+k) + '【瑞客论坛 www.ruike1.com】.txt'

f=open(fname)

for line in f:

alst=line.split(',') # 分解每行

alst[2]=int(alst[2]) # 将第2项(频次) 转为整数

if alst[1]=='F':

girl.append((alst[0],alst[2],1950+k)) # 以(姓名,频次,年份)的形式加入对应列表

else:

boy.append((alst[0],alst[2],1950+k))

f.close()

for i, sexlst in enumerate([boy, girl]): # 完成两个列表的统计

dic={ }

for x in sexlst: # x 是形如 (姓名, 频次, 年份) 的元组

dic[x[0]]=dic.get(x[0], 0) + x[1] # 得到字典统计结果

lst = sorted(dic.items(), key=lambda x:x[1], reverse=True) # 按频次逆序排列

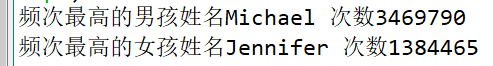
if i==0:

sex="男孩"

elif i==1:

sex="女孩"

print(f'频次最高的{sex}姓名{lst[0][0]} 次数{lst[0][1]}')



#方法二 用pandas库, 见课本第8章

import pandas as pd

columns=('name','sex','num') # 列名

df=pd.DataFrame(columns=columns) # 先定义一个空表

for i in range(1, 51):

# 注：低版本pandas读取文件名含中文的文件时将报错，需升级pandas或将文件名

# 改为纯英文和数字（例如使用第10题改名后的文件）

fname = 'babynames/yob'+str(1950+i)+'【瑞客论坛 www.ruike1.com】.txt'

x = pd.read\_csv(fname, header=None, names=columns) # 读一个文件，得到一个数据框

df = df.append(x) # 将df 和 x 两个数据框合并

# 筛选性别M, 按name分组，对每组的num求和，再排序，取最后一个

print('男性:', df[df.sex=='M'].groupby('name').num.sum().sort\_values()[-1:])

print('女性:', df[df.sex=='F'].groupby('name').num.sum().sort\_values()[-1:])

10. 当前目录下有babynames子目录，其中有很多txt格式的数据文件，请将这些txt文件名中的“【瑞客论坛 www.ruike1.com】”字样去掉，文件名的其余部分不变，将新文件另存到 当前目录的newfiles子目录下，原旧文件名不变。

提示：文件复制可使用 shutil.copy() 函数

import os,shutil

newpath='newfiles'

if not os.path.exists(newpath):

os.mkdir(newpath) # 如无此子目录，则创建

oldpath='babynames'

for fname in os.listdir(oldpath):

print(fname) # 调试用

lst = os.path.splitext(fname) # 分解文件名，得到(主文件名, 扩展名)

if lst[1].lower()=='.txt': # 如果扩展名为txt

newname=fname.replace('【瑞客论坛 www.ruike1.com】','') #将特定文字替换为空

shutil.copy(oldpath+'/'+fname, newpath+'/'+newname) # 文件复制

11. 爬取图片网站 www.99118.com 首页的风景图片，保存在pic子目录中。本例练习将图片以二进制文件保存。

注：需安装requests 和 bs4库 (anaconda均已包含)。

在该网站中图片的网址形如 'http://www.99118.com/300/201403/907\_300.jpg'

import requests , os

from bs4 import BeautifulSoup

import time

if not os.path.exists('pic'):

os.mkdir('pic') # 创建图片子目录

url='http://www.99118.com/'

resp = requests.get(url) # 访问网站

resp.encoding = 'gb2312' # 网页编码

bs = BeautifulSoup(resp.text,'lxml') # 构造网页解析对象bs

# 查看网页源码，发现图片均在img标签中，且css样式为 "cImage8"

lst\_img = bs.findAll('img', class\_="cImage8") # 找到所有符合要求的img标签

for img in lst\_img:

src = img['src'] # 取出img中的src属性，即图片的对应下载链接

filename = src.split('/')[-1] # 按/分割，最后一项是形如 907\_300.jpg的文件名

with open('pic/'+filename,'**wb**') as f: # 二进制文件

r = requests.get(src) # 访问图片网址

f.write(r.content) # r.contents是图片的字节码,将此字节码以二进制文件保存

print(filename) # 输出图片名，调试用

time.sleep(0.2) # 暂停0.2秒，减轻网站负荷

print('over')

另一图片网站也可尝试爬取： https://www.mmonly.cc/wmtp/fjtp/

12. 爬取起点网www.qidian.com上小说“万族之劫”前10章的公众内容，并保存在 novel.txt文件中。本例练习文本文件保存。

import requests

from bs4 import BeautifulSoup

import time

def getChapter(name,url):

url = "https:"+url # 拼凑为完整网址

resp = requests.get(url=url)

bs = BeautifulSoup(resp.text,"lxml")

# 查找小说主体对应的div

div = bs.find("div", class\_="read-content j\_readContent")

with open("novel.txt","a+") as file: # a+添加模式

file.write(name+"\n") # 先写入章名

for p in div.find\_all("p"): # 查找该div中所有的段落p

file.write(p.text+"\n") # 将各段文字存入文件

print("%s 成功" % name)

# 万族之劫 小说网址

url = "https://book.qidian.com/info/1018027842#Catalog"

resp = requests.get(url=url)

#resp.encoding="utf-8" # 有时需要设置编码

bs = BeautifulSoup(resp.text,"lxml") # 创建bs对象，以便解析网页

# 找到css 样式为"cf"的 ul， 其内所含的a标签即各章的链接

a\_list = bs.find("ul", class\_="cf").find\_all("a")

for item in a\_list[:10]: # 只下载前10章

chapter = item.text # 章名

url = item.get("href") # 该章对应链接

getChapter(chapter,url)

time.sleep(0.2)

注1：上面的语句也可改为 a\_list = bs.select("ul[class='cf'] a") ,表示使用css语法选择器，先选中ul[class='cf']的元素，再选中其所有的子节点a标签。

注2：bs.find()返回找到的第一个网页元素，本例要查找的ul是第一个，所以用bs.find即可。有些书的正文目录不是在第一个ul中，而是在第二个ul中，此时可将代码改为bs.find\_all(),此函数返回由所有的ul构成的一个列表，然后用[1]选取第二个ul,如下所示：

a\_list=bs.find\_all("ul", class\_="cf")[1].find\_all("a")

注3：BeautifulSoup的使用请参考其中文帮助文档：

https://www.crummy.com/software/BeautifulSoup/bs4/doc.zh/