**实验3 文件操作与数据存取**

**姓名:朱俊哲 学号:2212190220 班级:计科2202 成绩:**

**1. 实验目标**

本次实验旨在了解Python的文件数据操作方法，理解和掌握数据存取的各种方式。需掌握的相关知识点主要包括：

1) 文件数据读写：掌握文件的打开(open)以及打开模式（’r’,’w’和’a’）、读（read、readline、readlines）、写（write、writelines）方式，并学会with …. as句法，以及数据类型的转化、格式化处理和保存。

2) 文件管理：学会应用python的os、os.path、shutil等模块进行文件批量管理。

3) 数据永久存储：pickle模块的dump和load方法。注意此时文件对象的打开模式应为二进制形式(‘rb’或‘wb’).

请完成以下实验练习题，报告书写要求同第1次实验作业。注意，请在本报告中将自己的姓名、学号、班级书写正确。最后，请在规定时间内完成本次实验，并将报告的word文档(非pdf版本)以附件形式提交至学习通平台。

**2. 实验练习题**

**答题要求**：将正确运行的**源代码书写在题目下方**，并紧跟着贴上一份**源代码的截图**和运行结果的**截图**。答题格式同第1次实验作业。另外注意：如题目要求保存结果数据到文件，则该文件中的数据也算作运行结果，需截图粘贴。但提交报告时无需提供数据文件。

1. 文本文件“info\_stocks.txt”中每一行的name对应股票名称，shares对应多少股，price对应该只股票的单价。

(a) 请计算每只股票的总价，并按总价从大到小进行排序(程序中要求含lambda函数) ，将所有股票的名称、单价、总价依次打印输出。

(b) 要求使用Python内置函数filter过滤出单价大于80的股票有哪些? 并将符合条件的股票名称、单价、总价打印输出。

源代码：

data\_list **=** []

**with** open('info\_stocks.txt'**,** 'r') **as** file**:**

**for** line **in** file**:**

        parts **=** line**.**strip()**.**split(', ')

        name **=** parts[0]**.**split(': ')[1]**.**strip("'") *#split根据冒号和一个空格将part[0]分割为两部分，strip去除两边的单引号*

        shares **=** int(parts[1]**.**split(': ')[1])

        price **=** float(parts[2]**.**split(': ')[1])

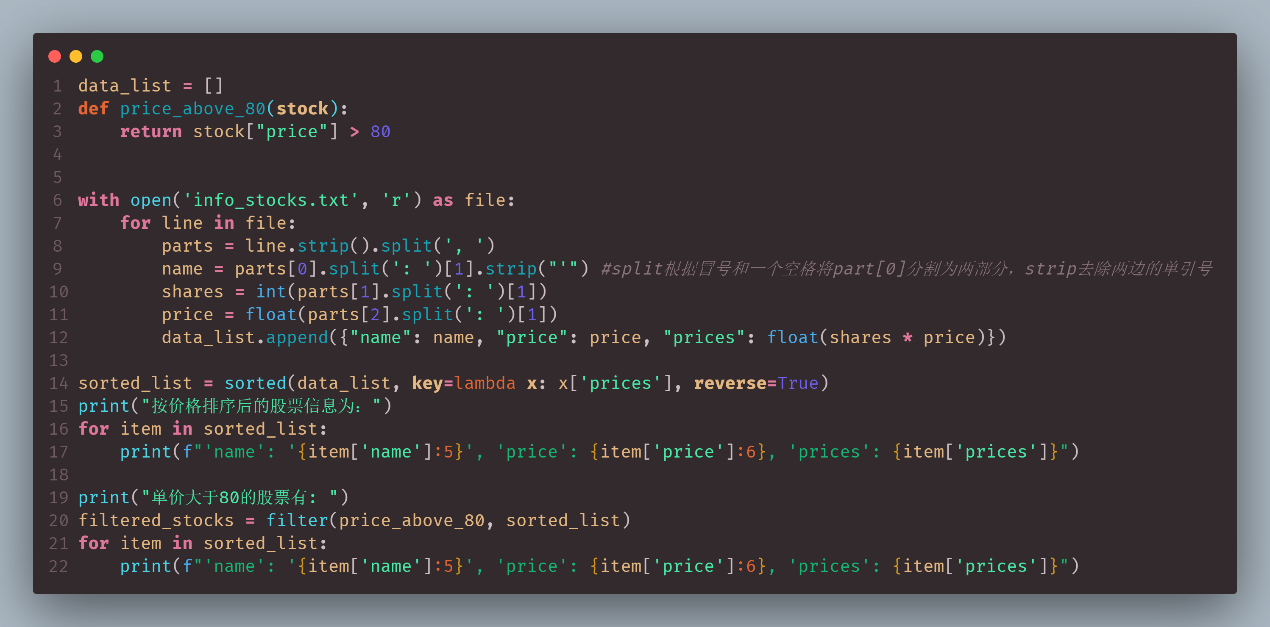
        data\_list**.**append({"name"**:** name**,** "price"**:** price**,** "prices"**:** float(shares **\*** price)})

sorted\_list **=** sorted(data\_list**,** **key=**lambda **x:** x['prices']**,** **reverse=**True)

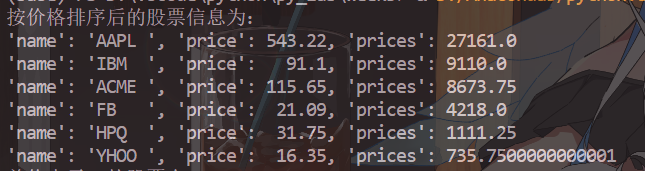
**for** item **in** sorted\_list**:**

    print(f"'name': '{item['name']:5}', 'price': {item['price']:6}, 'prices': {item['prices']}")

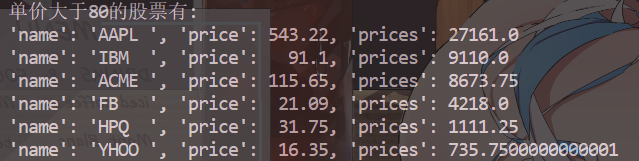
源代码截图：



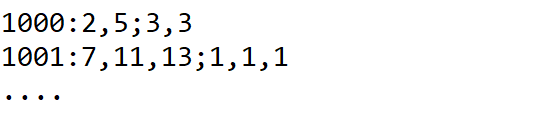
(a)运行结果截图：



(b)运行结果截图：

****

1. 将1000~2000之间的正整数质因子分解结果保存到文本文件中。保存格式：每个整数的质因子分解结果占一行，该行开始是这个整数本身，占8个字符位，然后冒号，全部质因子（逗号隔开），然后分号，最后全部质因子的幂（逗号隔开）。文件格式可参考如下：



源代码：

prime\_list **=** []

prime\_list**.**append(2)

**for** i **in** range(3**,** 2001**,** 2)**:**

**for** j **in** prime\_list**:**

**if** i **%** j **==** 0**:**

**break**

**else:**

        prime\_list**.**append(i)

file\_path **=** 'D://vscode//python//py\_lab//week3//prime.txt'

**with** open(file\_path**,** 'w') **as** file**:**

**for** i **in** range(1000**,** 2001)**:**

        factor\_list **=** []

        power\_list **=** []

        original\_i **=** i

**for** j **in** prime\_list**:**

**if** i **%** j **==** 0**:**

                factor\_list**.**append(j)

                power **=** 0

**while** i **%** j **==** 0**:**

                    i **=** i **/** j

                    power **+=** 1

                power\_list**.**append(power)

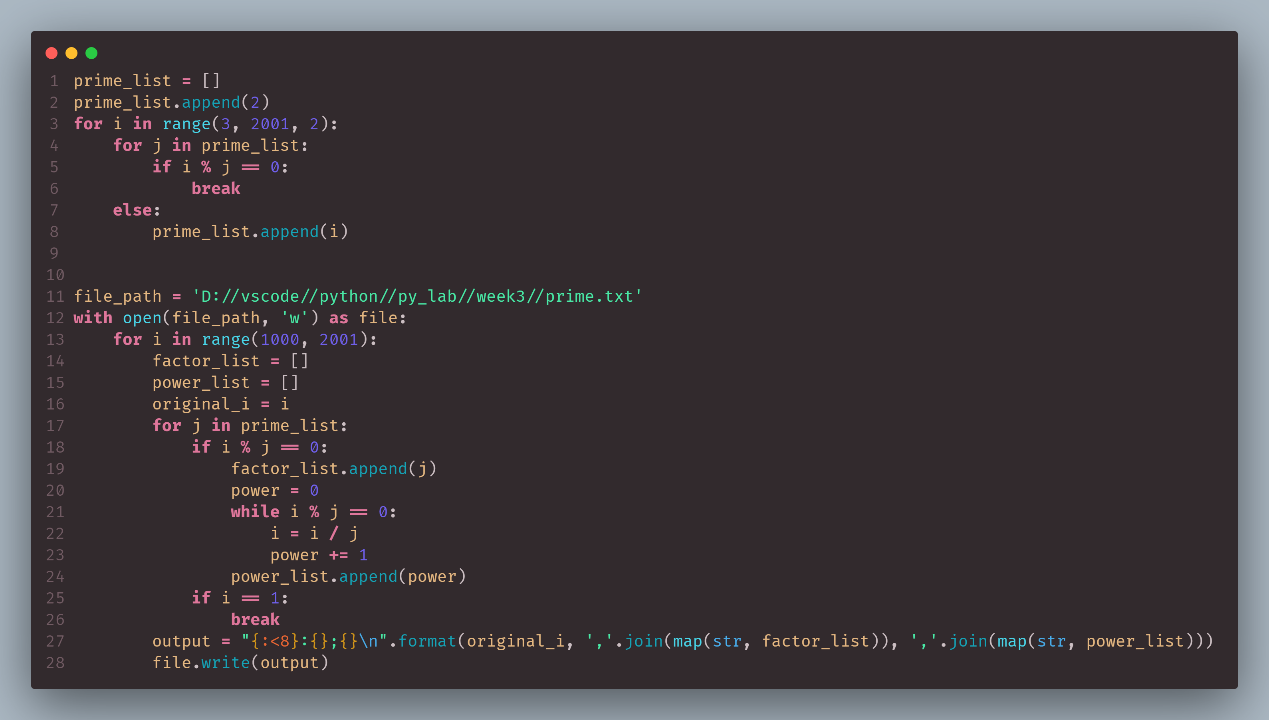
**if** i **==** 1**:**

**break**

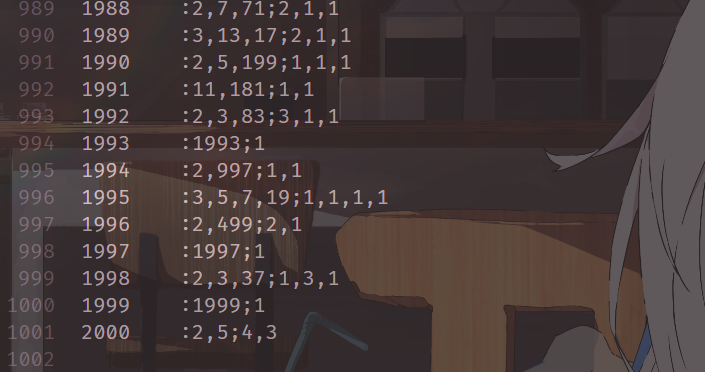
        output **=** "{:<8}:{};{}\n"**.**format(original\_i**,** ','**.**join(map(str**,** factor\_list))**,** ','**.**join(map(str**,** power\_list)))

        file**.**write(output)

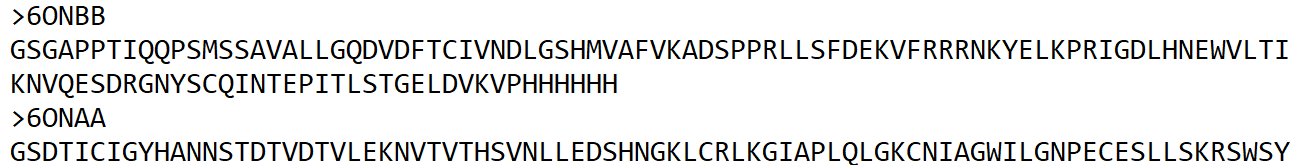
源代码截图：



文件内容截图：



1. 文件” seqs\_fasta.txt”保存有符合fasta格式的100多个蛋白质氨基酸序列，截取部分如下图所示：

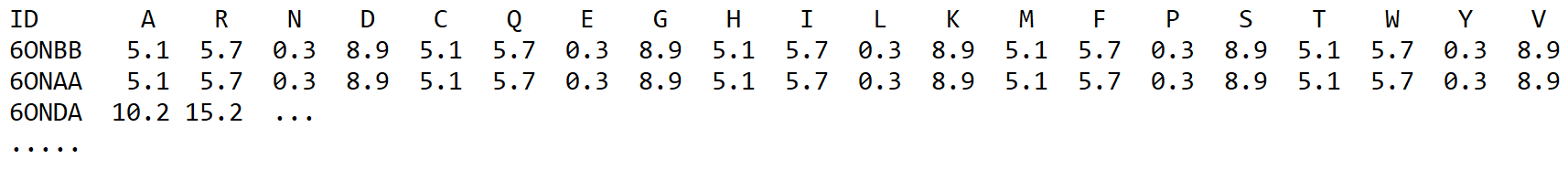


其中每条序列以“>”开头，紧接的“6ONBB”表示蛋白质序列ID，两行“>6ONBB”与“>6ONAA”之间展示的是ID为6ONBB的氨基酸序列。

自然界中，任何天然蛋白质主要由20种标准的氨基酸构成，可与字母表aa20={'A', 'R', 'N', 'D', 'C', 'Q', 'E', 'G', 'H', 'I', 'L', 'K', 'M', 'F', 'P', 'S', 'T', 'W', 'Y', 'V'}中的英文字母一一对应。同一种氨基酸在一条蛋白质序列中可能出现多次。

请计算文件“seqs\_fasta.txt”中每条蛋白质序列的氨基酸组分，即字母表aa20中的每个英文字母在该蛋白质序列中出现的频率（百分比%），并将统计结果保存至一个新的文件（创建的新文件可取名为’result6.txt’）。

注：要求氨基酸出现的频率值保留小数点后1位，且每个氨基酸所在列的频率值数据右对齐保存，保存文件的格式参考如下：



源代码：

**import** os

**from** collections **import** OrderedDict

aa20 **=** 'ARNDCQEGHILKMFPSTWYV'

seqs **=** OrderedDict()

file\_path **=** 'D://vscode//python//py\_lab//week3//result6.txt'

**with** open("seqs\_fasta.txt"**,** "r") **as** fout**:**

    seq\_id **=** ''

**for** line **in** fout**:**

**if** line[0] **==** ">"**:**

            seq\_id **=** line[1**:**]**.**strip()

            seqs[seq\_id] **=** ''

**else:**

            seqs[seq\_id] **+=** line**.**strip()

**with** open(file\_path**,** 'w') **as** fin**:**

    fin**.**write("ID"**.**ljust(9))

**for** i **in** aa20**:**

        fin**.**write(i**.**center(10))

    fin**.**write("\n")

**for** seq\_id**,** seq **in** seqs**.**items()**:**

        fin**.**write(f"{seq\_id}")

        summ **=** len(seq)

**for** aa **in** aa20**:**

            fre **=** seq**.**count(aa) **/** summ **\*** 100

            fin**.**write(f"{fre:.1f}"**.**rjust(10))

        fin**.**write("\n")

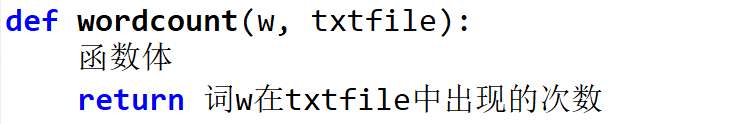
源代码截图：



文件内容截图：



1. 首先，定义一个名为“wordcount”的函数，功能为统计中文文本中某个关键字出现的次数，函数原型如下：

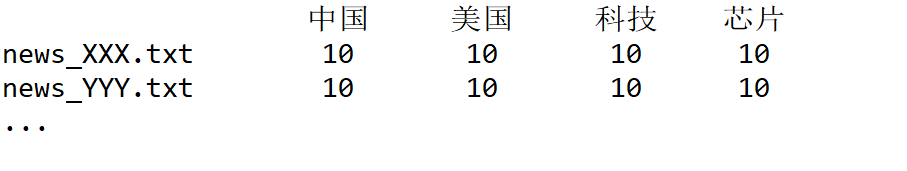


其中w和txtfile均为字符串。

其次，在存放本次实验材料的文件夹中，利用os.mkdir()创建一个新的文件夹，取名“mydir”；同时，自动识别出以“news\_”开头的所有文本文件，利用shutil模块将其移动至新建的文件目录“mydir”中(注：需编程自动实现文件的移动，不是手工拷贝或移动)。

进一步，利用pickle模块将函数wordcount以及识别出的以“news\_”开头的所有文本文件名组合成一个列表，永久保存至文件“wc.pkl”，并存储在文件夹“mydir”中。

最后，再次利用pickle模块将保存在“wc.pkl”中的列表数据载入，获得函数wordcount，并调用wordcount计算四个关键字“中国”、“美国”、“科技”和“芯片”在以“news\_”开头的所有文本文件中出现的次数，打印输出，格式参考如下：



源代码：

**def** wordcount(**w,** **txtfile**)**:**

**try:**

**with** open(txtfile**,** 'r'**,** **encoding=**'utf-8') **as** file**:**

            content **=** file**.**read()

**except** UnicodeDecodeError**:**

**with** open(txtfile**,** 'r'**,** **encoding=**'gbk') **as** file**:**

            content **=** file**.**read()

    count **=** content**.**count(w)

**return** count

**import** os

**import** shutil

**import** pickle

dir\_name **=** "mydir"

**if** **not** os**.**path**.**exists(dir\_name)**:**

    os**.**mkdir(dir\_name)

files\_moved **=** []

**for** file **in** os**.**listdir('.')**:**

**if** file**.**startswith('news\_')**:**

        shutil**.**move(file**,** os**.**path**.**join(dir\_name**,** file))

        files\_moved**.**append(file)

data\_to\_save **=** [wordcount**,** files\_moved]

**with** open(os**.**path**.**join(dir\_name**,** 'wc.pkl')**,** 'wb') **as** pkl\_file**:**

    pickle**.**dump(data\_to\_save**,** pkl\_file)

**with** open(os**.**path**.**join(dir\_name**,** 'wc.pkl')**,** 'rb') **as** pkl\_file**:**

    wordcount\_function**,** files **=** pickle**.**load(pkl\_file)

keywords **=** ["中国"**,** "美国"**,** "科技"**,** "芯片"]

results **=** {}

**for** file **in** files**:**

    file\_path **=** os**.**path**.**join(dir\_name**,** file)

    counts **=** {keyword**:** wordcount\_function(keyword**,** file\_path) **for** keyword **in** keywords}

    results[file] **=** counts

print(f"{'':<20}{'中国':>6}{'美国':>6}{'科技':>6}{'芯片':>6}")

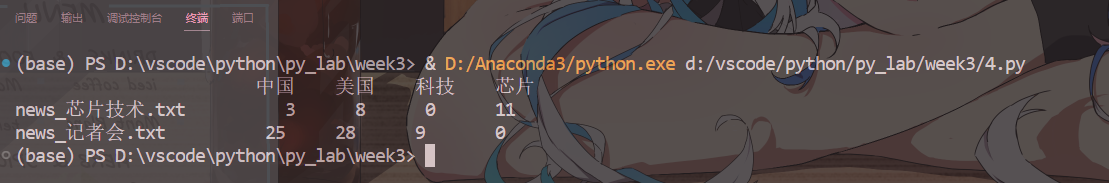
**for** file**,** counts **in** results**.**items()**:**

    print(f"{file:<17}{counts['中国']:>7}{counts['美国']:>7}{counts['科技']:>7}{counts['芯片']:>8}")

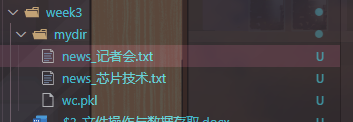
源代码截图：



运行结果截图：



文件夹mydir截图：



1. 创建文件，使用shutil模块中的copyfile()函数复制该文件，获取复制文件的创建时间。

源代码：

**from** pathlib **import** Path

**import** shutil

**import** os

**from** datetime **import** datetime

file\_path **=** Path("D://vscode//python//py\_lab//week3//original\_file.txt")

file\_path**.**touch()

copied\_file\_path **=** Path("D://vscode//python//py\_lab//week3//copied\_file.txt")

shutil**.**copyfile(file\_path**,** copied\_file\_path)

copied\_file\_creation\_time **=** os**.**path**.**getctime(copied\_file\_path)

timee **=** datetime**.**fromtimestamp(copied\_file\_creation\_time)**.**strftime('%Y-%m-%d %H:%M:%S')

print(timee)

源代码截图：



运行结果截图：



1. 从键盘输入一些字符，逐个把它们写到指定的文件out.txt中，直到输入一个@为止。

源代码：

**with** open('out.txt'**,** 'w') **as** file**:**

**while** True**:**

        char **=** input("请输入一个字符: ")

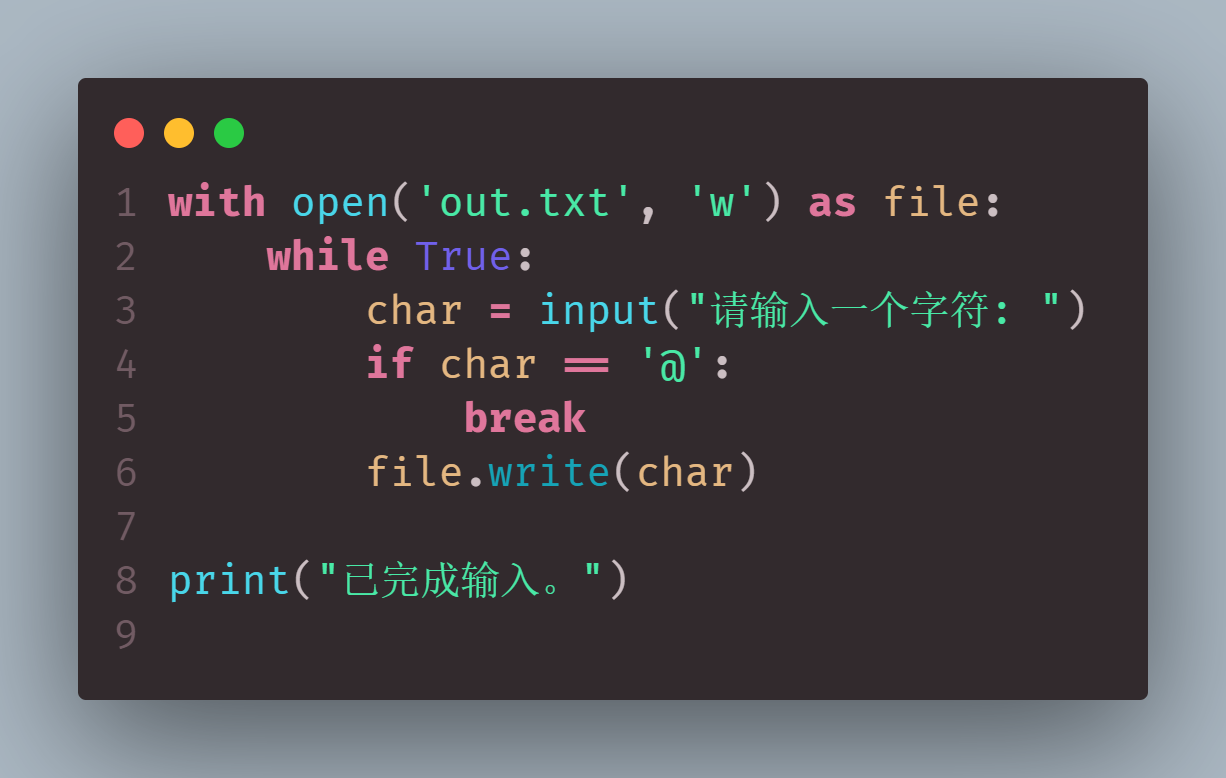
**if** char **==** '@'**:**

**break**

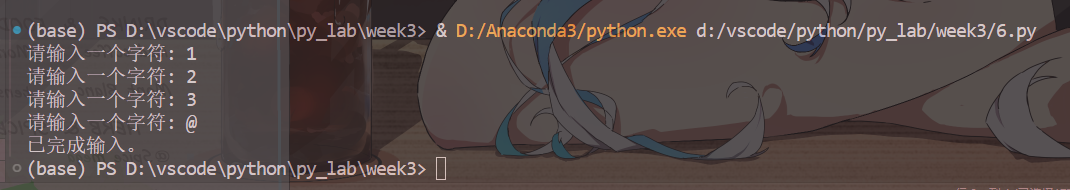
        file**.**write(char)

print("已完成输入。")

源代码截图：



运行结果截图：





1. 判断当前路径中是否有oldfilename.txt文件，如果存在则按行输出内容，否则创建该文件。  
   源代码：
2. **import** os
3. filename **=** 'oldfilename.txt'
4. **if** os**.**path**.**exists(filename)**:**
5. **with** open(filename**,** 'r') **as** file**:**
6. **for** line **in** file**:**
7. print(line**,** **end=**'')
8. **else:**
9. open(filename**,** 'w')**.**close()
10. print(f"'{filename}' 文件已创建。")

源代码截图：



运行结果截图：



1. 读取学生成绩文件stu\_grade.txt，文件第一列表示学生学号，第二列表示学生姓名，第三列表示学生成绩，按学生成绩排序后输出到新文件stu\_grade\_output.txt中。

源代码：

*# 读取 stu\_grade.txt 文件*

**with** open("stu\_grade.txt"**,** "r") **as** file**:**

    lines **=** file**.**readlines()

*# 解析文件内容，并根据成绩排序*

grades **=** []

**for** line **in** lines**:**

    student\_id**,** name**,** grade **=** line**.**strip()**.**split(',')

    grades**.**append((student\_id**,** name**,** int(grade)))

sorted\_grades **=** sorted(grades**,** **key=**lambda **x:** x[2]**,** **reverse=**True)

*# 将排序后的内容写入到新文件 stu\_grade\_output.txt 中*

output\_path **=** "stu\_grade\_output.txt"

**with** open(output\_path**,** "w") **as** file**:**

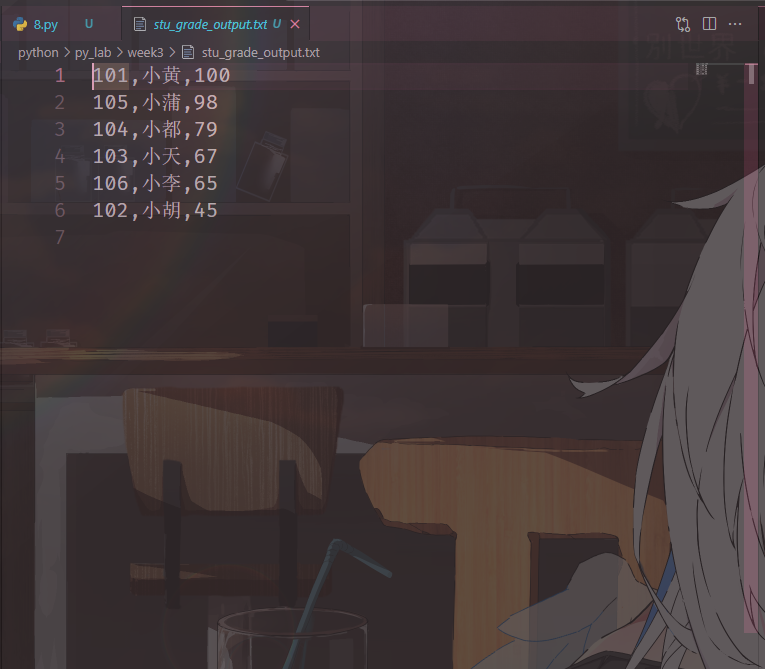
**for** student **in** sorted\_grades**:**

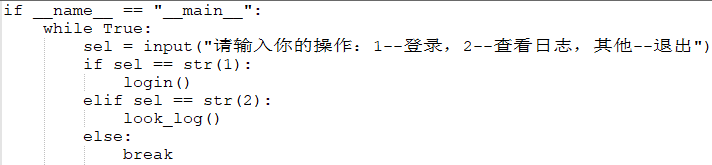
        file**.**write(f"{student[0]},{student[1]},{student[2]}\n")

源代码截图：



运行结果截图：



1. 创建一个叮咚客服管理系统的界面，每次登录时，将用户的登录日志写入文件中（包含登录时间），并且在程序中查看用户的登录日志。假定系统中只有一个用户（用户名： "张三" ，密码"123456"）可以登录成功；请记录用户是否登录成功以及登录时间。  
    下图为系统的主函数，请写出login()函数和look\_log()函数，实现上述要求。

源代码：

**import** datetime

**def** login()**:**

    valid\_username **=** "张三"

    valid\_password **=** "123456"

    username **=** input("请输入用户名: ")

    password **=** input("请输入密码: ")

    now **=** datetime**.**datetime**.**now()

    timestamp **=** now**.**strftime("%Y-%m-%d %H:%M:%S")

**if** username **==** valid\_username **and** password **==** valid\_password**:**

**with** open("login\_log.txt"**,** "a") **as** log\_file**:**

            log\_file**.**write(f"登录成功: 用户名 {username}, 登录时间 {timestamp}\n")

        print("登录成功!")

**else:**

**with** open("login\_log.txt"**,** "a") **as** log\_file**:**

            log\_file**.**write(f"登录失败: 用户名 {username}, 登录时间 {timestamp}\n")

        print("用户名或密码错误, 登录失败!")

**def** look\_log()**:**

**try:**

**with** open("login\_log.txt"**,** "r") **as** log\_file**:**

            print(log\_file**.**read())

**except** FileNotFoundError**:**

        print("登录日志文件不存在。")

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

**while** True**:**

        sel **=** input("请选择你的操作：1-登录, 2-查看日志, 其他-退出")

**if** sel **==** str(1)**:**

            login()

**elif** sel **==** str(2)**:**

            look\_log()

**else:**

**break**

源代码截图：



运行结果截图：

