爬虫山腰课后练习参考答案

第6关

练习–储存电影信息–参考

第一步：分析问题，明确结果
问题需求就是把豆瓣TOP250里面的 序号/电影名/评分/推荐语/链接 都爬取下来，结果是存储在csv和Excel中

【讲解】
问题需求就是把豆瓣TOP250里面的 序号/电影名/评分/推荐语/链接 都爬取下来，结果是存储在csv和Excel中

第二步：书写爬虫代码
回顾下第三关的爬虫代码

【解答】

```
import requests, random, bs4

for x in range(10):
    url = 'https://movie.douban.com/top250?start=' + str(x*25) +
'&filter='
    res = requests.get(url)
    bs = bs4.BeautifulSoup(res.text, 'html.parser')
    bs = bs.find('ol', class_="grid_view")
    for titles in bs.find_all('li'):
        num = titles.find('em',class_="").text
        title = titles.find('span', class_="title").text
        comment = titles.find('span',class_="rating_num").text
        url_movie = titles.find('a')['href']

        if titles.find('span',class_="inq") != None:
            tes = titles.find('span',class_="inq").text
            print(num + '.' + title + '——' + comment + '\n' + '推荐语：' +
tes +'\n' + url_movie)
        else:
            print(num + '.' + title + '——' + comment + '\n' +'\n' +
url_movie)
```

第三步： 完善代码，用Excel存储信息

要存储在Excel中呢，需要先创建工作表，重命名，再设置表头，把爬取的信息写成列表，然后用append函数多行写入Excel，最后命名保存这个Excel 文件。
请改写下方的爬虫代码，实现使用Excel存储信息。

【解答】

```
import requests, random, bs4, openpyxl

wb=openpyxl.Workbook()
#创建工作薄
sheet=wb.active
#获取工作薄的活动表
sheet.title='movies'
#工作表重命名
sheet['A1'] ='序号'        #加表头，给A1单元格赋值
sheet['B1'] ='电影名'      #加表头，给B1单元格赋值
sheet['C1'] ='评分'       #加表头，给C1单元格赋值
sheet['D1'] ='推荐语'      #加表头，给D1单元格赋值
sheet['E1'] ='链接'       #加表头，给E1单元格赋值

for x in range(10):
    url = 'https://movie.douban.com/top250?start=' + str(x*25) +
'&filter='
    res = requests.get(url)
    bs = bs4.BeautifulSoup(res.text, 'html.parser')
    bs = bs.find('ol', class_="grid_view")
    for titles in bs.find_all('li'):
        num = titles.find('em',class_="").text
        title = titles.find('span', class_="title").text
        comment = titles.find('span',class_="rating_num").text
        url_movie = titles.find('a')['href']

        if titles.find('span',class_="inq") != None:
            tes = titles.find('span',class_="inq").text
            sheet.append([num, title, comment, tes, url_movie])
            # 把num, title, comment, tes和url_movie写成列表，用append函数多行
写入Excel
            print(num + '.' + title + '——' + comment + '\n' + '推荐语：' +
tes +'\n' + url_movie)
        else:
            sheet.append([num, title, comment, None,url_movie])
            print(num + '.' + title + '——' + comment + '\n' +'\n' +
url_movie)


wb.save('movieTop250.xlsx')
#最后保存并命名这个Excel文件
```

第四步：另辟蹊径，用csv格式存储信息

思考下如何用csv创建写入存储呢?

请修改下方代码，实现使用csv存储信息。

【提示】

```
import requests,  random, bs4, csv
#引用csv模块。
csv_file=open('movieTop250.csv', 'w', newline='')
#调用open()函数打开csv文件，传入参数：文件名“movieTop250.csv”、写入模式“w”、newline=''
```

【解答】

```
import requests,  random, bs4, csv
#引用csv模块。
csv_file=open('movieTop250.csv', 'w', newline='')
#调用open()函数打开csv文件，传入参数：文件名“movieTop250.csv”、写入模式“w”、newline=''。
writer = csv.writer(csv_file)
# 用csv.writer()函数创建一个writer对象。
writer.writerow(['序号', '电影名', '评分', '推荐语', '链接'])
#调用writer对象的writerow()方法，可以在csv文件里写入title:'序号', '电影名', '评分', '推荐语', '链接'

for x in range(10):
    url = 'https://movie.douban.com/top250?start=' + str(x*25) + '&filter='
    res = requests.get(url)
    bs = bs4.BeautifulSoup(res.text, 'html.parser')
    bs = bs.find('ol', class_="grid_view")
    for titles in bs.find_all('li'):
        num = titles.find('em',class_="").text
        title = titles.find('span', class_="title").text
        comment = titles.find('span',class_="rating_num").text
        url_movie = titles.find('a')['href']

        if titles.find('span',class_="inq") != None:
            tes = titles.find('span',class_="inq").text
            # 把num, title, comment, tes和url_movie写成列表，用append函数多行写入Excel
            writer.writerow([num + '.' + title + '——' + comment + '\n' + '推荐语：' + tes +'\n' + url_movie])
        else:
            writer.writerow([num + '.' + title + '——' + comment + '\n' +'\n' + url_movie])

csv_file.close()
```

第7关

练习–做个测单词的小工具–参考

第一步：分析需求，明确目标

扇贝网：https://www.shanbay.com/已经有一个测单词量的功能，我们要做的就是把这个功能复制下来，并且做点改良，搞一个网页版没有的功能 ———— 自动生成错词本。
在这一步，请阅读文档的同时打开浏览器的扇贝网，跟着我一步步来。

【讲解】

这里是扇贝网的测单词量的界面：想要复制功能，就先做分析，这个网页是怎样的工作流程。
所以，先体验全程
先看源代码里是否有我们的单词。倘若有，就用find()/find_all()定位提取需要的数据；
没有的话，就要调用【检查】–【Network】 – 【XHR】 – 找数据。
在Headers里看网址，在Preview里看内容。

category/这一个XHR，用的是Get请求方式，访问了网址https://www.shanbay.com/api/v1/vocabtest/category/下载了一个字典。其中“data”里面，藏了十个元素。这十个元素，里面对应的内容，就是我们最开始要选择的“词汇范围”。十个元素，每个里面都有两个内容。0是什么暂时还不知道，先放着。1是我们词汇范围没错。比如我们选择高考，那么在第2个元素里就有一个0是“NCEE”，有一个1是“高考”。

我们接着看下一个XHR：
在此，“NCEE”出现了两次：这个XHR的名字里面有“NCEE”，它访问的网址里面也有“NCEE”。
这就揭示了一种对应关系：当我们选择“高考”词库，那么下一个XHR，访问的网址就会是用“NCEE”来结尾。可以多试几个词库验证下我们的猜测，的确里面的对应关系是一致的。考研和NGEE一组，四级和CET 4一组，六级和CET 6一组。

第1个XHR，所访问的网址规律就
是：’https://www.shanbay.com/api/v1/vocabtest/vocabularies/?category=’+’你选择的词库，对应的代码’。
它下载到的是一个字典。字典里，包含了用来测试词汇量的50个单词。
第0，它先给出单词。
第1，它给出四个不同的翻译，每个翻译都有一个对应的pk值和rank值。
第2，它再给出一组pk值和rank值。它们，和正确翻译里面的pk值与rank值一致。
到这里，我们就完成了至关重要的“需求分析”这个步骤。

第二步：分步讲解，书写代码 (｡▰‿‿▰｡) ♡

【讲解】

下面，我将带你一步步完成代码。
（0）. 选择题库。
写这个程序，要用到requests模块。
先用requests下载链接，再用res.json()解析下载内容。
让用户选择想测的词库，输入数字编号，获取题库的代码。

提示：记得给input前面加一个int()来转换数据类型

【解答】

```
import requests



lin=requests.get('https://www.shanbay.com/api/v1/vocabtest/category/')
#先用requests下载链接。
js_link = lin.json()
#解析下载得到的内容。
bianhao = int(input('''请输入你选择的词库编号，按Enter确认
1，GMAT  2，考研  3，高考  4，四级  5，六级
6，英专  7，托福  8，GRE  9，雅思  10，任意
>'''))
#让用户选择自己想测的词库，输入数字编号。int()来转换数据类型
ciku = js_link['data'][bianhao-1][0]
print(ciku)

```

（1）. 根据选择的题库，获取50个单词。
第0步我们已经拿到链接，这步直接用requests去下载，re.json()解析即可。

【解答】

```
test =
requests.get('https://www.shanbay.com/api/v1/vocabtest/vocabularies/?
category='+ciku)
#下载用于测试的50个单词。
words = test.json()
#对测试的单词进行解析。
print(words)

```

（2）. 让用户选择认识的单词：此处，要分别记录下用户认识哪些，不认识哪些。
已经有了单词数据，提取出来让用户识别，并记录用户认识哪些不认识哪些，至少2个list来记录。50个单词，记得要用循环。用户手动输入自己的选择，用input() 。我们要识别用户的输入，并基于此决定把这个单词放进哪个list，需要用if语句。
提示：当一个元素特别长的时候，给代码多加一个list。
提示：加个换行，优化用户视角。
新增一个list，用于统计用户认识的单词
创建一个空的列表，用于记录用户认识的单词。
创建一个空的列表，用于记录用户不认识的单词。
启动一个循环，循环的次数等于单词的数量。
记得加一个\n，用于
让用户输入自己是否认识。
如果用户认识：
就把这个单词，追加进列表words_knows。
否则
就把这个单词，追加进列表not_knows。
打印一个统计数据：这么多单词，认识几个，认识的有哪些?

【解答】

```
danci = []
#新增一个list，用于统计用户认识的单词
words_knows = []
not_knows = []
print ('测试现在开始。如果你认识这个单词，请输入Y，否则直接敲Enter：')
n=0
for x in words['data']:
    n=n+1
    print ("\n第"+str(n)+'个：'+x['content'])
    #加一个\n，用于换行。
    answer = input('认识请敲Y，否则敲Enter：')
    if answer == 'Y':
        danci.append(x['content'])
        #把用户认识的单词，追加进danci这个list。
        words_knows.append(x)
    else:
        not_knows.append(x)


print ('\n在上述'+str(len(words['data']))+'个单词当中，
有'+str(len(danci))+'个是你觉得自己认识的，它们是：')
print(danci)

```

（3）. 对于用户认识的单词，给选择题让用户做：此处要记录用户做对了哪些，做错了哪些。

这一步是第0步和第2步的组合——涉及到第0步中的选择，也涉及到第2步的数据记录。

提示： 面对冗长的字典列表相互嵌套，可以创建字典。

【解答】

```
print ('现在我们来检测一下，你有没有真正掌握它们：')
wrong_words = []
right_num = 0
for y in words_knows:
    print('\n\n'+'A:'+y['definition_choices'][0]['definition'])
    #我们改用A、B、C、D，不再用rank值，下同
    print('B:'+y['definition_choices'][1]['definition'])
    print('C:'+y['definition_choices'][2]['definition'])
    print('D:'+y['definition_choices'][3]['definition'])
    xuanze = input('请选择单词\"'+y['content']+'\"的正确翻译：')
    dic = {'A':y['definition_choices'][0]
['rank'],'B':y['definition_choices'][1]
['rank'],'C':y['definition_choices'][2]
['rank'],'D':y['definition_choices'][3]['rank']}
    #我们创建一个字典，搭建起A、B、C、D和四个rank值的映射关系。
    if dic[xuanze] == y['rank']:
    #此时dic[xuanze]的内容，其实就是rank值，此时的代码含义已经和之前的版本相同了。
```

```
        right_num += 1
    else:
        wrong_words.append(y)

```

（4）. 生成报告：50个单词，不认识多少，认识多少，掌握多少，错了多少。
生成报告主要有三部分：第0，是输出统计数据；第1，是打印错题集；第2，是把错题集保存到本地。

【解答】

```
import requests

link = requests.get('https://www.shanbay.com/api/v1/vocabtest/category/')
#先用requests下载链接。
js_link = link.json()
#解析下载得到的内容。
bianhao = int(input('''请输入你选择的词库编号，按Enter确认
1，GMAT  2，考研  3，高考  4，四级  5，六级
6，英专  7，托福  8，GRE  9，雅思  10，任意
>'''))
#让用户选择自己想测的词库，输入数字编号。int()来转换数据类型
ciku = js_link['data'][bianhao-1][0]
#利用用户输入的数字编号，获取题库的代码。如果以输入“高考”的编号“3”为例，那么ciku的值就是，在字典js_link中查找data的值，data是一个list，查找它的第bianhao-1，也就是第2个元素，得到的依然是一个list，再查找该list的第0个元素。最后得到的就是我们想要的NCEE。
test = requests.get('https://www.shanbay.com/api/v1/vocabtest/vocabularies/?category='+ciku)
#下载用于测试的50个单词。
words = test.json()
#对test进行解析。
danci = []
#新增一个list，用于统计用户认识的单词
words_knows = []
#创建一个空的列表，用于记录用户认识的单词。
not_knows = []
#创建一个空的列表，用于记录用户不认识的单词。
print ('测试现在开始。如果你认识这个单词，请输入Y，否则直接敲Enter：')
n=0
for x in words['data']:
#启动一个循环，循环的次数等于单词的数量。
    n=n+1
    print ("\n第"+str(n)+'个：'+x['content'])
    #加一个\n，用于换行。
    answer = input('认识请敲Y，否则敲Enter：')
    #让用户输入自己是否认识。
    if answer == 'Y':
    #如果用户认识：
        danci.append(x['content'])
```

```
        words_knows.append(x)
        #就把这个单词，追加进列表words_knows。
    else:
    #否则
        not_knows.append(x)
        #就把这个单词，追加进列表not_knows。



print ('\n在上述'+str(len(words['data']))+'个单词当中，
有'+str(len(danci))+'个是你觉得自己认识的，它们是：')
print(danci)



print ('现在我们来检测一下，你有没有真正掌握它们：')
wrong_words = []
right_num = 0
for y in words_knows:
    print('\n\n'+'A:'+y['definition_choices'][0]['definition'])
    #我们改用A、B、C、D，不再用rank值，下同
    print('B:'+y['definition_choices'][1]['definition'])
    print('C:'+y['definition_choices'][2]['definition'])
    print('D:'+y['definition_choices'][3]['definition'])
    xuanze = input('请选择单词\"'+y['content']+'\"的正确翻译（填写数字即
可）：')
    dic = {'A':y['definition_choices'][0]
['rank'],'B':y['definition_choices'][1]
['rank'],'C':y['definition_choices'][2]
['rank'],'D':y['definition_choices'][3]['rank']}
    #我们创建一个字典，搭建起A、B、C、D和四个rank值的映射关系。
    if dic[xuanze] == y['rank']:
    #此时dic[xuanze]的内容，其实就是rank值，此时的代码含义已经和之前的版本相同了。
        right_num += 1
    else:
        wrong_words.append(y)


print ('现在，到了公布成绩的时刻:')
print ('在'+str(len(words['data']))+'个'+js_link['data'][bianhao-1][1]+'词
汇当中，你认识其中'+str(len(danci))+'个，实际掌握'+str(right_num)+'个，错
误'+str(len(wrong_words))+'个。')
#这是句蛮复杂的话，对照前面的代码和json文件你才能理解它。一个运行示例是：在50个高考
词汇当中，你认识其中30个，实际掌握25个，错误5个。


save = input ('是否打印并保存你的错词集？填入Y或N： ')
#询问用户，是否要打印并保存错题集。
if save == 'Y':
```

```
#如果用户说是：
    f = open('错题集.txt', 'a+')
    #在当前目录下，创建一个错题集.txt的文档。
    print ('你记错的单词有：')
    f.write('你记错的单词有：\n')
    #写入"你记错的单词有：\n"
    m=0
    for z in wrong_words:
    #启动一个循环，循环的次数等于，用户的错词数：
        m=m+1
        print (z['content'])
        #打印每一个错词。
        f.write(str(m+1) +'. '+ z['content']+'\n')
        #写入序号，写入错词。
    print ('你不认识的单词有：')
    f.write('你没记住的单词有：\n')
    #写入"你没记住的单词有：\n"
    s=0
    for x in not_knows:
    #启动一个循环，循环的次数等于，用户不认识的单词数。
        print (x['content'])
        #打印每一个不认识的单词。
        f.write(str(s+1) +'. '+ x['content']+'\n')
        #写入序号，写入用户不认识的词汇。
    print ('错词和没记住的词已保存至当前文件目录下，下次见！')
    #告诉用户，文件已经保存好。
    #在网页版终端运行时，文件会被写在课程的服务器上，你看不到，但它的确已经存在。
else:
#如果用户不想保存：
    print('下次见！')
    #输出“下次见！”

```

三步：参考答案

【讲解】

参考答案:

```
import requests



link = requests.get('https://www.shanbay.com/api/v1/vocabtest/category/')
#先用requests下载链接。
js_link = link.json()
#解析下载得到的内容。
bianhao = int(input('''请输入你选择的词库编号，按Enter确认
1，GMAT  2，考研  3，高考  4，四级  5，六级
6，英专  7，托福  8，GRE  9，雅思  10，任意
>'''))
#让用户选择自己想测的词库，输入数字编号。int()来转换数据类型
ciku = js_link['data'][bianhao-1][0]
```

```
#利用用户输入的数字编号，获取题库的代码。如果以输入“高考”的编号“3”为例，那么ciku的值就是，在字典js_link中查找data的值，data是一个list，查找它的第bianhao-1，也就是第2个元素，得到的依然是一个list，再查找该list的第0个元素。最后得到的就是我们想要的NCEE。
test = requests.get('https://www.shanbay.com/api/v1/vocabtest/vocabularies/?category='+ciku)
#下载用于测试的50个单词。
words = test.json()
#对test进行解析。
danci = []
#新增一个list，用于统计用户认识的单词
words_knows = []
#创建一个空的列表，用于记录用户认识的单词。
not_knows = []
#创建一个空的列表，用于记录用户不认识的单词。
print ('测试现在开始。如果你认识这个单词，请输入Y，否则直接敲Enter：')
n=0
for x in words['data']:
#启动一个循环，循环的次数等于单词的数量。
    n=n+1
    print ("\n第"+str(n)+'个：'+x['content'])
    #加一个\n，用于换行。
    answer = input('认识请敲Y，否则敲Enter：')
    #让用户输入自己是否认识。
    if answer == 'Y':
    #如果用户认识：
        danci.append(x['content'])
        words_knows.append(x)
        #就把这个单词，追加进列表words_knows。
    else:
    #否则
        not_knows.append(x)
        #就把这个单词，追加进列表not_knows。



print ('\n在上述'+str(len(words['data']))+'个单词当中，有'+str(len(danci))+'个是你觉得自己认识的，它们是：')
print(danci)



print ('现在我们来检测一下，你有没有真正掌握它们：')
wrong_words = []
right_num = 0
for y in words_knows:
    print('\n\n'+'A:'+y['definition_choices'][0]['definition'])
    #我们改用A、B、C、D，不再用rank值，下同
```

```
    print('B:'+y['definition_choices'][1]['definition'])
    print('C:'+y['definition_choices'][2]['definition'])
    print('D:'+y['definition_choices'][3]['definition'])
    xuanze = input('请选择单词\"'+y['content']+'\"的正确翻译（填写数字即可）：')
    dic = {'A':y['definition_choices'][0]['rank'],'B':y['definition_choices'][1]['rank'],'C':y['definition_choices'][2]['rank'],'D':y['definition_choices'][3]['rank']}
    #我们创建一个字典，搭建起A、B、C、D和四个rank值的映射关系。
    if dic[xuanze] == y['rank']:
    #此时dic[xuanze]的内容，其实就是rank值，此时的代码含义已经和之前的版本相同了。
        right_num += 1
    else:
        wrong_words.append(y)


print ('现在，到了公布成绩的时刻:')
print ('在'+str(len(words['data']))+'个'+js_link['data'][bianhao-1][1]+'词汇当中，你认识其中'+str(len(danci))+'个，实际掌握'+str(right_num)+'个，错误'+str(len(wrong_words))+'个。')
#这是句蛮复杂的话，对照前面的代码和json文件你才能理解它。一个运行示例是：在50个高考词汇当中，你认识其中30个，实际掌握25个，错误5个。


save = input ('是否打印并保存你的错词集？填入Y或N： ')
#询问用户，是否要打印并保存错题集。
if save == 'Y':
#如果用户说是：
    f = open('错题集.txt', 'a+')
    #在当前目录下，创建一个错题集.txt的文档。
    print ('你记错的单词有：')
    f.write('你记错的单词有：\n')
    #写入"你记错的单词有：\n"
    m=0
    for z in wrong_words:
    #启动一个循环，循环的次数等于，用户的错词数：
        m=m+1
        print (z['content'])
        #打印每一个错词。
        f.write(str(m+1) +'. '+ z['content']+'\n')
        #写入序号，写入错词。
    print ('你不认识的单词有：')
    f.write('你没记住的单词有：\n')
    #写入"你没记住的单词有：\n"
    s=0
    for x in not_knows:
    #启动一个循环，循环的次数等于，用户不认识的单词数。
        print (x['content'])
        #打印每一个不认识的单词。
        f.write(str(s+1) +'. '+ x['content']+'\n')
```

```
103             #写入序号，写入用户不认识的词汇。
104         print ('错词和没记住的词已保存至当前文件目录下，下次见！')
105         #告诉用户，文件已经保存好。
106         #在网页版终端运行时，文件会被写在课程的服务器上，你看不到，但它的确已经存在。
107 else:
108 #如果用户不想保存：
109         print('下次见！')
110         #输出“下次见！
111
```

第8关

练习–我家附近有啥好吃的–参考

项目目标：在本练习，我们会借助cookies的相关知识，使用Python登录饿了么网站，爬取自己家附近的餐厅列表。
网站地址：https://www.ele.me/home/

【讲解】
在本练习，我们会借助cookies的相关知识，使用Python登录饿了么网站，爬取自己家附近的餐厅。
网站地址：https://www.ele.me/home/
想要顺利地爬取饿了么上面的餐厅列表，我们需要先自己前往饿了么网站，手动找到餐厅列表所在位置。然后，再用Python代码去模拟这个过程。

首先，打开饿了么首页：(https://www.ele.me/home/)
打开【检查】工具，选择【Network】，勾选【Preserve log】（因为等会可能会有页面跳转，勾选上防止在跳转过程中请求被清空）。
然后，输入地址，如：“腾讯大厦”（推荐先把文字打好，再利用复制粘贴一次性输入，不要一个字一个字填输），页面会弹出许多个地址给你选择。此刻，会看到右侧的Network面板里多出一个XHR：pois?……
这说明，这些地址列表和XHR之间存在有对应关系。记住这个XHR，它很重要，后面会用到。
我们先点击一个地址，比如我选择的第0个，“腾讯大厦”，它会跳转至一个新地址：
(https://www.ele.me/place/ws100xkkpznf?latitude=22.54055&longitude=113.934401)

此时，页面要求我们登录。我们点击登录，会来到
(https://h5.ele.me/login/#redirect=https://www.ele.me/place/ws100xkkpznf?latitude=22.54055&longitude=113.934401)

阅读该URL，很容易能够看出这个是一个登录页，因为有login存在。同时它在?之后所携带的参数，含义是来源：我们刚刚从哪个URL跳转过来。
输入手机号码，点击收取验证码，此时浏览器会发起请求：mobile_send_code。
手机收到验证码，输入验证码，完成登录。页面会重新跳转回我们刚刚来到的地址，此时的页面，出现了我们想要的餐馆名。
login_by_mobile即是登录获取cookies的那个请求。

通过翻找Network，我们定位到，存储有餐厅列表的请求是XHR：restaurants...

该请求需要若干参数，如下：
1. extras[]：activities
2. geohash：通过搜索得之，这是一个能够代表地理位置的字符串。
3. latitude：纬度
4. limit：一次加载多少个餐馆
5. longitude：经度
6. offset：起始值
7. terminal: web

思考实现方案
这个网站，要求登录才能实现爬取餐馆列表。所以我们需要用到cookies来实现。
所以理论上，我们的代码顺序应该是：模拟登录获取cookies，再带着cookies去请求餐馆列表。
不过，必须要指出的是。请求餐馆列表，还需要一些参数才行。这些参数，和我们刚刚在首页输入“腾讯大厦”四个字的操作有关。
具体，我们在做代码实操时再展开讲解。
所以正确的过程应该是：
– 模拟登录获取cookies
– 模拟输入“腾讯大厦”获取必要的参数
– 带着参数和cookies，去请求餐馆列表
其中，前两步可以顺序调换。

【讲解】
下面，我将带你一步步完成代码。
一、使用session和cookies模拟登录
体验登录：https://h5.ele.me/login/
提示：此处需要先模拟发送验证码的请求，再模拟登录的请求。
提示：请求验证码时，会返回一个json，json里会有validate_token。它在我们输入账号验证码模拟登录的时候，会用到。
模拟登录参考示例：

```
import requests
session = requests.session()
#创建会话。
headers = {
'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110 Safari/537.36'
}
#添加请求头，避免被反爬虫。
url = ' https://wordpress-edu-3autumn.localprod.forc.work/wp-login.php'
#登录的网址。
data = {'log': input('请输入你的账号:'),
        'pwd': input('请输入你的密码:'),
        'wp-submit': '登录',
        'redirect_to': 'https://wordpress-edu-
3autumn.localprod.forc.work/wp-admin/',
        'testcookie': '1'}
#登录的参数。
```

```
session.post(url, headers=headers, data=data)
#在会话下，用post发起登录请求。
```

【解答】

```
import requests
session = requests.session()
# 创建会话
headers = {
'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110 Safari/537.36'
}
# 添加请求头，避免被反爬虫
url_1 = 'https://h5.ele.me/restapi/eus/login/mobile_send_code'
# 发送验证码的网址
tel = input('请输入手机号码：')
data_1 = {'captcha_hash':'',
        'captcha_value':'',
        'mobile':tel,
        'scf':''}
# 发送验证码的参数
token = session.post(url_1, headers=headers, data=data_1).json()
['validate_token']
# 在会话下，模拟获取验证码的请求

url_2 = 'https://h5.ele.me/restapi/eus/login/login_by_mobile'
code = input('请输入手机验证码：')
data_2 = {'mobile':tel,
        'scf':'ms',
        'validate_code':code,
        'validate_token':token}
session.post(url_2,headers=headers,data=data_2)
```

二、模拟输入地址，获取必要参数

为了请求餐馆列表，我们需要几个关键参数：

1. geohash：通过搜索得之，这是一个能够代表地理位置的字符串。
2. latitude：纬度
3. longitude：经度

这三个参数，都需要模拟输入地址来获得。请打印出这三个参数。

体验输入地址：https://www.ele.me/home/

提示：本步骤不需要模拟登录

提示：在模拟该请求时需要用到城市的geohash值，可在XHR里查看自己城市的geohash值

【解答】

```
import requests
# 导入requests模块。
address_url = 'https://www.ele.me/restapi/v2/pois?'
# 你能够在【Headers】-【General】里找到这个链接。
```

```
place = input('请输入你的收货地址：')
# 使用input输入收获地址，赋值给place。
# 因为我们的geohash使用了深圳的值，所以推荐你测试的时候使用“腾讯大厦”。
params =
{'extras[]':'count','geohash':'ws105rz9smwm','keyword':place,'limit':'20',
'type':'nearby'}
# 将要传递的参数封装成字典，键与值都要用字符串，其中keyword对于的值是place。
address_res = requests.get(address_url,params=params)
# 发起请求，将响应的结果，赋值给address_res
address_json = address_res.json()
# 将响应的结果转为列表/字典。

print('以下，是与'+place+'相关的位置信息：\n')
n=0
# 添加一个计数器，作为序号。
for address in address_json:
# 遍历我们刚爬取的地址列表。
    print(str(n)+'. '+address['name']+': '+address['short_address']+'\n')
    # 打印序号，地址名，短地址。
    n = n+1
    # 给计数器加1。
address_num = int(input('请输入您选择位置的序号：'))
# 让用户选择序号。
final_address = address_json[address_num]
# 确认地址。

print(final_address['geohash'])
print(final_address['latitude'])
print(final_address['longitude'])
```

三、带cookies和参数请求餐馆列表

最后一步，将上述两组代码组合。

拿到cookies和参数，完成请求餐馆列表。

我帮你预置了前两个代码，你可以在此基础上完成本关卡任务。

【解答】

```
import requests
session = requests.session()

headers = {
'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110 Safari/537.36'
}
url_1 = 'https://h5.ele.me/restapi/eus/login/mobile_send_code'
tel = input('请输入手机号码：')
data_1 = {'captcha_hash':'',
        'captcha_value':'',
        'mobile':tel,
        'scf':''}

```

```
token = session.post(url_1, headers=headers, data=data_1).json()
['validate_token']

url_2 = 'https://h5.ele.me/restapi/eus/login/login_by_mobile'
code = input('请输入手机验证码：')
data_2 = {'mobile':tel,
        'scf':'ms',
        'validate_code':code,
        'validate_token':token}

session.post(url_2,headers=headers,data=data_2)


address_url = 'https://www.ele.me/restapi/v2/pois?'
place = input('请输入你的收货地址：')
params =
{'extras[]':'count','geohash':'ws105rz9smwm','keyword':place,'limit':'20',
'type':'nearby'}
# 这里使用了深圳的geohash

address_res = requests.get(address_url,params=params)
address_json = address_res.json()

print('以下，是与'+place+'相关的位置信息：\n')
n=0
for address in address_json:
    print(str(n)+'. '+address['name']+': '+address['short_address']+'\n')
    n = n+1
address_num = int(input('请输入您选择位置的序号：'))
final_address = address_json[address_num]

restaurants_url = 'https://www.ele.me/restapi/shopping/restaurants?'
# 使用带有餐馆列表的那个XHR地址。
params = {'extras[]':'activities',
'geohash':final_address['geohash'],
'latitude':final_address['latitude'],
'limit':'24',
'longitude':final_address['longitude'],
'offset':'0',
'terminal':'web'
}
# 将参数封装，其中geohash和经纬度，来自前面获取到的数据。
restaurants_res = session.get(restaurants_url,params=params)
# 发起请求，将响应的结果，赋值给restaurants_res
restaurants = restaurants_res.json()
# 把response对象，转为json。
for restaurant in restaurants:
# restsurants最外层是一个列表，它可被遍历。restaurant则是字典，里面包含了单个餐厅
的所有信息。
    print(restaurant['name'])
```

参考答案和总结

参考答案：

```
import requests
session = requests.session()

headers = {
'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110 Safari/537.36'
}
url_1 = 'https://h5.ele.me/restapi/eus/login/mobile_send_code'
tel = input('请输入手机号码：')
data_1 = {'captcha_hash':'',
        'captcha_value':'',
        'mobile':tel,
        'scf':''}

token = session.post(url_1, headers=headers, data=data_1).json()
['validate_token']

url_2 = 'https://h5.ele.me/restapi/eus/login/login_by_mobile'
code = input('请输入手机验证码：')
data_2 = {'mobile':tel,
        'scf':'ms',
        'validate_code':code,
        'validate_token':token}

session.post(url_2,headers=headers,data=data_2)


address_url = 'https://www.ele.me/restapi/v2/pois?'
place = input('请输入你的收货地址：')
params =
{'extras[]':'count','geohash':'ws105rz9smwm','keyword':place,'limit':'20',
'type':'nearby'}
# 这里使用了深圳的geohash

address_res = requests.get(address_url,params=params)
address_json = address_res.json()

print('以下，是与'+place+'相关的位置信息：\n')
n=0
for address in address_json:
    print(str(n)+'. '+address['name']+': '+address['short_address']+'\n')
    n = n+1
address_num = int(input('请输入您选择位置的序号：'))
final_address = address_json[address_num]

restaurants_url = 'https://www.ele.me/restapi/shopping/restaurants?'
# 使用带有餐馆列表的那个XHR地址。
```

```
params = {'extras[]':'activities',
'geohash':final_address['geohash'],
'latitude':final_address['latitude'],
'limit':'24',
'longitude':final_address['longitude'],
'offset':'0',
'terminal':'web'
}
# 将参数封装，其中geohash和经纬度，来自前面获取到的数据。
restaurants_res = session.get(restaurants_url,params=params)
# 发起请求，将响应的结果，赋值给restaurants_res
restaurants = restaurants_res.json()
# 把response对象，转为json。
for restaurant in restaurants:
# restsurants最外层是一个列表，它可被遍历。restaurant则是字典，里面包含了单个餐厅的所有信息。
    print(restaurant['name'])
```

一份总结：
就是这样一个代码，它能拿到给定位置附近的餐厅名。但它的潜力并不只是如此。
如果我们尝试加载饿了么官网的首页，能够找到一个xhr叫做cities，这个xhr里包含了全国两千多个城市的经纬度。
利用Python的geohash模块，你可以将经纬度数据，转化为geohash（当然，也可以将geohash转为经纬度，我也是用这种方式，发现我的默认geohash是深圳）。
那么在理论上，其实你可以通过这种方式，拿到全国餐厅的数据……
只要稍做扩展，它还能拿到许多数据：所有的餐厅名/电话号码/评分/品牌/经纬度/介绍/均价/月销量……
此时，这个爬虫就具备了商业价值，它能胜任许多数据分析的工作：选址策略、定价策略、差异化竞争、2B营销……
或许你会质疑自己能不能做到像我描述的这样厉害，不用怕，很快你就能够做到对此心中有数。
而在后续的关卡，当你学会反爬虫的应对策略、协程、Scrapy框架……你会变得像我说的那样强大。

练习–自制翻译器–参考

第一步：分析问题，明确目标

实现功能：用户输入英文或中文，程序即可打印出来对应的译文。

【讲解】
我们在左边输入文字，那么浏览器会把输入的信息传输给服务器，再返回对应的内容。
我们希望达成的效果如下图，即用户输入英文或中文，程序即可打印出来对应的译文：

第二步：思考要用到的知识

【讲解】

实现一键翻译的功能，最简单的方案便是爬虫。在此，我们选择的网站是有道翻译。
你在左边输入文字，那么浏览器会把你输入的信息传输给服务器。再返回对应的内容。
这就是一个典型的Post操作。
我们在Headers也可以看到“Request Method: POST”哦
在前几关练习我们用的都是Get方式请求，Post是另一种常见的方式，课上已经学过其用法，在此不多赘述。
__Get是向服务器发索取数据的一种请求，而Post是向服务器提交数据的一种请求__

虽然第八关我们主要讲的是Cookies，
__Cookies用于服务器实现会话，用户登录及相关功能时进行状态管理__
但这道题并不需要用到小饼干，因为不需要登录不需要账号密码等。
主要考查的还是Post的用法。

__注意哦__ ლ(╹◡╹ლ)
有道翻译有反爬虫机制，它使用了加密技术。如果你的程序报错，你可以通过搜索、查阅资料找到解决方案：尝试把访问的网址中“/translate_o”中的“_o”删除。
服务器返回的内容，是json的格式。我们可以用处理列表、处理字典的手段来提取翻译。

第三步：写代码

你可以在浏览器的[network]-[Headers]-[General]里找到需要访问的网址，在[network]-[Headers]-[From data]里找到需要上传的数据。

__再次注意哦__ ლ(╹◡╹ლ)
有道翻译有反爬虫机制，它使用了加密技术。如果你的程序报错，你可以通过搜索、查阅资料找到解决方案：尝试把访问的网址中“/translate_o”中的“_o”删除。
服务器返回的内容，是json的格式。我们可以用处理列表、处理字典的手段来提取翻译。

【解答】

```
import requests,json
#调用了两个模块。requests负责上传和下载数据，json负责解析。



word = input('你想翻译什么呀？')
url='http://fanyi.youdao.com/translate?smartresult=dict&smartresult=rule'
#使用post需要一个链接。
data={'i': word,
      'from': 'AUTO',
      'to': 'AUTO',
      'smartresult': 'dict',
      'client': 'fanyideskweb',
      'doctype': 'json',
      'version': '2.1',
      'keyfrom': 'fanyi.web',
      'action': 'FY_BY_REALTIME',
      'typoResult': 'false'}
#将需要post的内容，以字典的形式记录在data内。
```

```
r = requests.post(url,data)
#post需要输入两个参数，一个是刚才的链接，一个是data，返回的是一个Response对象。
answer=json.loads(r.text)
#你可以自己尝试print一下r.text的内容，然后再阅读下面的代码。
print ('翻译的结果是：'+answer['translateResult'][0][0]['tgt'])


```

第四步：套层壳（小彩蛋，了解即可，感兴趣的话可以深入学习）

我们总会听到前端后端全栈，感觉神秘有高大上，你一定很好奇它们都是什么呀?

今天呢，我们就简单接触下前端～

有米有很期待呀(>﹏<`)ﾉｼ

前端，是一种GUI软件。而我们现在要用的是Python里的一个模块实现本地窗口的功能。

它就是Tkinter～

Tkinter 模块是 Python 的标准 Tk GUI 工具包的接口。

Tk 和 Tkinter 可以在大多数的 Unix 平台下使用,同样可以应用在 Windows 和 MacOS系统里。

Tk8.0 的后续版本可以实现本地窗口风格,并良好地运行在绝大多数平台中。

http://www.runoob.com/python/python-gui-tkinter.html

最后的代码大约是这个模样，注意阅读注释，

当然你可以在终端运行（复制）这些代码，观察效果～

【解答】

认真阅读注释，你也可以复制下来在你的IDE中运行下哦～

```
import requests
import json
from tkinter import Tk,Button,Entry,Label,Text,END


class YouDaoFanyi(object):
    def __init__(self):
        pass
    def crawl(self,word):
        url='http://fanyi.youdao.com/translate?
smartresult=dict&smartresult=rule'
        #使用post需要一个链接
        data={'i': word,
              'from': 'AUTO',
              'to': 'AUTO',
              'smartresult': 'dict',
              'client': 'fanyideskweb',
              'doctype': 'json',
              'version': '2.1',
              'keyfrom': 'fanyi.web',
              'action': 'FY_BY_REALTIME',
              'typoResult': 'false'}
        #将需要post的内容，以字典的形式记录在data内。
        r = requests.post(url, data)
```

```
        #post需要输入两个参数，一个是刚才的链接，一个是data，返回的是一个Response对象
        answer=json.loads(r.text)
        #你可以自己尝试print一下r.text的内容，然后再阅读下面的代码。
        result = answer['translateResult'][0][0]['tgt']
        return result



class Application(object):
    def __init__(self):
        self.window = Tk()
        self.fanyi = YouDaoFanyi()




        self.window.title(u'我的翻译')
        #设置窗口大小和位置
        self.window.geometry('310x370+500+300')
        self.window.minsize(310,370)
        self.window.maxsize(310,370)
        #创建一个文本框
        #self.entry = Entry(self.window)
        #self.entry.place(x=10,y=10,width=200,height=25)
        #self.entry.bind("<Key-Return>",self.submit1)
        self.result_text1 = Text(self.window,background = 'azure')
        # 喜欢什么背景色就在这里面找哦，但是有色差，得多试试：http://www.science.smith.edu/dftwiki/index.php/Color_Charts_for_TKinter
        self.result_text1.place(x = 10,y = 5,width = 285,height = 155)
        self.result_text1.bind("<Key-Return>",self.submit1)


        #创建一个按钮
        #为按钮添加事件
        self.submit_btn = Button(self.window,text=u'翻译',command=self.submit)
        self.submit_btn.place(x=205,y=165,width=35,height=25)
        self.submit_btn2 = Button(self.window,text=u'清空',command = self.clean)
        self.submit_btn2.place(x=250,y=165,width=35,height=25)


        #翻译结果标题
        self.title_label = Label(self.window,text=u'翻译结果:')
        self.title_label.place(x=10,y=165)
        #翻译结果


        self.result_text = Text(self.window,background = 'light cyan')
        self.result_text.place(x = 10,y = 190,width = 285,height = 165)
```

```
        #回车翻译
    def submit1(self,event):
        #从输入框获取用户输入的值
        content = self.result_text1.get(0.0,END).strip().replace("\n"," ")
        #把这个值传送给服务器进行翻译


        result = self.fanyi.crawl(content)
        #将结果显示在窗口中的文本框中


        self.result_text.delete(0.0,END)
        self.result_text.insert(END,result)

        #print(content)

    def submit(self):
        #从输入框获取用户输入的值
        content = self.result_text1.get(0.0,END).strip().replace("\n"," ")
        #把这个值传送给服务器进行翻译


        result = self.fanyi.crawl(content)
        #将结果显示在窗口中的文本框中


        self.result_text.delete(0.0,END)
        self.result_text.insert(END,result)
        print(content)
    #清空文本域中的内容
    def clean(self):
        self.result_text1.delete(0.0,END)
        self.result_text.delete(0.0,END)


    def run(self):
        self.window.mainloop()




if __name__=="__main__":
    app = Application()
    app.run()

```

练习–图灵机器人–参考

第一步：登录注册图灵机器人

注册登录，才能创建自己的图灵机器人。
根据帮助中心的“说明书”，我们可以了解如何运用这个新工具～

【讲解】
进入图灵机器人官网http://www.tuling123.com/，戳进帮助中心。
就像打开玩具先看说明书一样，我们来看看官方文档怎么说怎么用～
在功能说明中，我们知道，首先得登录注册，用免费版本就可以了（当然～土豪请随意），创建机器人在“机器人设置”中，我们用的是第一个API接入

那什么是API呢？通俗地讲：
__API就是接口__，就是通道，负责一个程序和其他软件的沟通，本质是预先定义的函数，而我们不需要了解这个函数只是调用这个接口就可达到函数的效果。
好，接下来我们看下“API V2.0接入文档”.

接口说明：API接口可调用聊天对话、语料库、技能三大模块的语料。
很好，我们今天想做的聊天机器人用这个接口就刚巧合适～

同时，在使用说明中我们可以知晓：
首先创建post请求所需的json数据，然后向指定的接口发起post请求即可，
而且从参数说明中可以看到，只有参数 perception 和 userinfo 才是必须的.

对于userid这个参数官方文档说的是：长度小于32，是用户的唯一标识，这里我们只要创建 userid 是长度小于32的字符串即可，说明书已经看完啦，来，开始着手做准备工作！
那我们回到主页，注册登录

然后在机器人管理界面，创建图灵机器人，最多可以创建5个，由此得出对应的5个apikey。
(实际上一个就够啦)
apikey是针对接口访问的授权方式。
准备工作做完啦，接下来想想该如何写代码

第二步：创建自己的聊天机器人
请求过程：首先创建post请求所需的json数据，然后向指定的接口发起post请求即可，
而且从参数说明中可以看到，只有参数 perception 和 userinfo 才是必须的
想不清楚的可以看提示哦

【提示】
userid = str(1)
1 可以替换成任何长度小于32的字符串哦
apikey = str(‘A’)
这里的A，记得替换成你自己的apikey哦～

对咯，还有件小事需要注意一下，有时候可能你的代码没有错，但最后显示加密错误，那是apikey过期了，
不过没关系，不是可以最多创建5个机器人嘛，换一个apikey试试就好咯。

【解答】

```
import requests
import json
```

```
3
4
5   userid = str(1)
6   # 1 可以替换成任何长度小于32的字符串哦
7   apikey = str(''A)
8   # 这里的A，记得替换成你自己的apikey哦～
9
10  # 创建post函数
11  def robot(content):
12      # 图灵api
13      api = r'http://openapi.tuling123.com/openapi/api/v2'
14      # 创建post提交的数据
15      data = {
16          "perception": {
17              "inputText": {
18                  "text": content
19                          }
20                      },
21          "userInfo": {
22                      "apiKey": apikey,
23                      "userId": userid,
24                      }
25      }
26      # 转化为json格式
27      jsondata = json.dumps(data)
28      # 发起post请求
29      response = requests.post(api, data = jsondata)
30      # 将返回的json数据解码
31      robot_res = json.loads(response.content)
32      # 提取对话数据
33      print(robot_res["results"][0]['values']['text'])
34
35
36  for x in range(10):
37      content = input("talk:")
38      # 输入对话内容
39      robot(content)
40      if x == 10:
41          break
42          # 十次之后就结束对话，数字可以改哦，你想几次就几次
43
44
45
46  #当然咯，你也可以加一些stopwords，只要说了这些词就可以终止聊天
47
48  while True:
49      content = input("talk:")
50      # 输入对话内容
51      robot(content)
52      if content == 'bye':
53      # 设置stopwords
```

```
        break



#但是，我觉得吧，喜欢和聊天机器人玩的都是话痨，所以，可以最后加个死循环，如下：

# 创建对话死循环
while True:
    # 输入对话内容
    content = input("talk:")
    robot(content)

```

练习-nlpir人工智能-参考

第一步：明确目标
项目实现的三步是：明确目标、分析过程、代码实现。

【讲解】
我们将完成一个和语义识别相关的爬虫程序，输入任意词汇、句子、文章或段落，会返回该联想词汇。
我们爬取的网站是：http://ictclas.nlpir.org/nlpir/

第二步：分析过程
我们边看网站，边分析。
我们一步一步来分析一下这个网站，点开：http://ictclas.nlpir.org/nlpir/
“Word2vec”（联想词汇），是一个上传数据，然后服务器解析数据，返回给我们的过程。所以它不会写进网页源代码HTML里，那我们需要查看Network——XHR。
好，打开检查工具，选择Network-XHR，仍然在文本框输入“音乐剧”，然后点击一次“Word2vec”的功能，看看浏览器是如何帮我们传输数据的。
在Preview（预览）当中，我们看到一个字典，就包含了返回的联想词汇。
我们现在得去看看，这个XHR请求的网址是什么，请求的内容是什么。我们来点击Headers。
你会看到三个框，它们是我们要重点留意的东西。第0个框，告诉我们请求的网址是：http://ictclas.nlpir.org/nlpir/index6/getWord2Vec.do，以及请求的方式POST。

第1个框，是我们熟悉的User-Agent。服务器会根据这个信息判断提交请求的是人还是爬虫，所以要封装headers。

第2个框，就是POST请求所提交的数据。你看到它是一个字典的结构，包含一个键——content，和对应的值——音乐剧。
提交这些数据给服务器，服务器会返回给我们一个字典，你可以在Preview中看到。
在网页中，对应的是左侧的圆圈。

分析完数据的请求和返回，我们就有了大概的思路了： 首先发送post请求，然后在返回的数据中，取出我们需要的结果即可。

第三步：代码实现（上）

首先，要发送post请求，我们可以使用requests.post()来发送请求。
但是，里面要放三个参数，Requests URL，还有headers，还有要发送的数据data，data以字典的形式封装，然后，我们把返回的数据打印出来。

【提示】
以下不完整的代码，是一点提示。

```
import requests
url='http://ictclas.nlpir.org/nlpir/index6/getWord2Vec.do'
words=input('请输入任意文本')
data={'content':words}

```

【解答】

```
import requests
#引入requests库
url = 'http://ictclas.nlpir.org/nlpir/index6/getWord2Vec.do'
#Word2vec功能对应的请求网址。
headers = {'Origin': 'http://ictclas.nlpir.org'
'Referer': 'http://ictclas.nlpir.org/nlpir/'
'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.67 Safari/537.36'}
#封装请求头
words = input('请输入你想查询的词汇：')
#获取需要分析的文本
data = {'content':words}
#封装数据内容，我们在Headers面板中的底部看过发送的数据内容是一个字典。
res = requests.post(url,data=data,headers=headers)
#发送post请求
print(res.text)
#把返回的结果打印出来

```

第四步：代码实现（下）
好，发送post请求后，我们拿到了返回的数据结果，是一个json数据。
我们只取出主要的联想词汇和其相关系数，即网页中的词汇，和线条上的数值。对应在json中。
这时候，需要我们把这个json数据转化成字典，再根据键来取值。要用到json.loads()，具体用法如下：

```
import json
# 引入json模块
a = [1,2,3,4]
# 创建一个列表a。
b = json.dumps(a)
# 使用dumps()函数，将列表a转换为json格式的字符串，赋值给b。
print(b)
# 打印b。
print(type(b))
# 打印b的数据类型，为字符串。
```

```
c = json.loads(b)
# 使用loads()函数，将json格式的字符串b转为列表，赋值给c。
print(c)
# 打印c。
print(type(c))
# 打印c的数据类型，为列表。
```

接着，我们就把json数据转换成了字典，我们就可以根据键——w2vlist来取值。
取出来的值是个列表，我们用遍历和字符串切片的方法，把主要的联想词汇，以及其相关系数取出来。
字符串切片的方法如下：

```
a='郑云龙,阿云嘎,马佳,蔡程昱,高天鹤,余笛'
# a是一个大字符串，可以把这个字符串切开。
b=a.split(',')
# 指定分隔符是逗号，每碰到一个逗号，就切一下。
print(b)
# 打印b，结果会是一个由6个字符串组成的列表。
print(type(b))
# b是一个列表。

```

【提示】
请仔细看网页的数据结构。
以下不完整的代码，是一点小提示。

```
import requests,json
url = 'http://ictclas.nlpir.org/nlpir/index6/getWord2Vec.do'
headers = {'Origin': 'http://ictclas.nlpir.org'
'Referer': 'http://ictclas.nlpir.org/nlpir/'
'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.67 Safari/537.36'}
words = input('请输入你想查询的词汇：')
data = {'content':words}
res = requests.post(url,data=data,headers=headers)
data=res.text
# 以上，为上一步的代码


data1=json.loads(data)
# 把json数据转换为字典
for i in data1['w2vlist']:
# 遍历列表，下面的代码请你自己完成


```

【解答】

```
import requests,json
url = 'http://ictclas.nlpir.org/nlpir/index6/getWord2Vec.do'
headers = {'Origin': 'http://ictclas.nlpir.org'
```

```
'Referer': 'http://ictclas.nlpir.org/nlpir/'
'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/70.0.3538.67 Safari/537.36'}
words = input('请输入你想查询的词汇：')
data = {'content':words}
res = requests.post(url,data=data,headers=headers)
data=res.text
# 以上，为上一步的代码


data1=json.loads(data)
# 把json数据转换为字典
print ('和“'+words+'”相关的词汇，至少还有：')
# 打印文字
f=0
# 设置变量f
for i in data1['w2vlist']: # 遍历列表
    f=f+1
    word = i.split(',')    # 切割字符串
    print ('('+str(f)+')'+word[0]+'，其相关度为'+word[1]) # 打印数据


```

第9关

练习-博客达人-参考

你需要做的事情是：
首先，登录博客[人人都是蜘蛛侠](https://wordpress-edu-3autumn.localprod.forc.work/wp-login.php)。
然后，在文章《未来已来（三）——同九义何汝秀》中，发表一个评论，这个评论中必须要带有“selenium”这个词。
博客登录页面:https://wordpress-edu-3autumn.localprod.forc.work/wp-login.php
答这道题的时候，使用可视模式会对运行结果有更直观的了解，如果你想看到浏览器的操作过程，建议你在本地写好练习答案，然后再复制到这里。

【提示】
使用`selenium`操作浏览器的方法，就和人一步一步操作的步骤是一样的：
1. 获取网页
2. 输入用户名与密码，点击登录
3. 点击《未来已来（三）——同九义何汝秀》文章标题，进入文章页面
4. 找到评论区，输入评论，点击发表评论
用到的知识点都是`selenium`提取数据的方法，以及操作元素的方法。

【解答】

特别提示：

从博客首页进入文章页面时，需要用到 find_element_by_partial_link_text 通过链接的部分文本获取超链接。

发表评论之后，不会再终端返回运行结果，记得去博客文章的页面去看看自己的评论有没有成功~

由于教学系统中与你本地的浏览器设置方法不同，我给你提供了两份答案，一份可以在课程系统中运行，一份可以在你的本地运行。

```
import time


from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.chrome.webdriver import RemoteWebDriver


# 获取用户输入的评论内容
while True:
    comment_content = input('请输入你想要的评论的内容，按回车提交：')
    if comment_content == '':
        print('&' * 5, '评论内容不允许为空', '&' * 5)
    else:
        break


'''方法一：使用自己电脑上的浏览器'''
driver = webdriver.Chrome()  # 实例化浏览器对象
driver.get('https://wordpress-edu-3autumn.localprod.forc.work/wp-login.php')  # 访问页面
time.sleep(2)


'''方法二：使用教学系统的浏览器设置，只能在网页的在线编辑器上运行'''
# chrome_options = Options()  # 实例化Option对象
# chrome_options.add_argument('--headless')  # 对浏览器的设置
# driver = webdriver.Chrome("http://chromedriver.python-class-fos.svc:4444/wd/hub",
#                           chrome_options.to_capabilities())  # 声明浏览器对象
# driver.get('https://wordpress-edu-3autumn.localprod.forc.work/wp-login.php')  # 访问页面


# 定位到用户名输入框，输入用户名
login_name = driver.find_element_by_id('user_login')
login_name.send_keys('spiderman')
```

```
time.sleep(1)
# 定位到密码输入框，输入密码
password = driver.find_element_by_id('user_pass')
password.send_keys('crawler334566')
# 定位到登录按钮,并点击按钮
submit_btn = driver.find_element_by_id('wp-submit')
submit_btn.click()
time.sleep(2)


# 通过链接的部分文本定位到'《未来已来（三）——同九义何汝秀》'这篇文章
# 获取到该文章对应的a标签（超链接），并点击链接进入文章详情页
article_link = driver.find_element_by_partial_link_text('同九义何汝秀')
article_link.click()


# 进入文章详情页，定位到该页面下编写评论的文本框，输入内容
comment_area = driver.find_element_by_id('comment')
comment_area.send_keys(comment_content)
time.sleep(2)
# 定位到提交按钮，点击该按钮提交评论
comment_submit = driver.find_element_by_id('submit')
comment_submit.click()


# 评论成功10秒后关闭浏览器
time.sleep(10)
driver.close()
print('#' * 6, '评论成功，浏览器已关闭', '#' * 6)


```

练习–Python之禅–参考

题目要求
获取网站[你好，蜘蛛侠！](https://localprod.pandateacher.com/python-manuscript/hello-spiderman/)的Python之禅中文英对照文本。
需要通过两种方法获取：
- 只使用`selenium`
- `selenium`与`BeautifulSoup`配合

【讲解】
[你好，蜘蛛侠！](https://localprod.pandateacher.com/python-manuscript/hello-spiderman/)是一个动态网页，URL：https://localprod.pandateacher.com/python-manuscript/hello-spiderman/
Python之禅的内容没有存在网页源代码中，无法通过`requests.get()`与`BeautifulSoup`提取“Python之禅”的内容，不过，我们可以通过`selenium`获取到。

方法如下：

0. 使用`selenium`获取网页
1. 输入你喜欢的老师和助教，点击提交
2. 提取`Elements`中渲染完成的完整网页源代码中的，中英文对照的Python之禅
我在课堂讲解中带你做过前两步了，第三步正是你现在需要做的。

我们可以用两种方式完成它，
第一种方法：
只使用`selenium`。

第二种方法：
使用`selenium`配合`BeautifulSoup`。

第一种方法：selenium
这次我们要用`selenium`单独完成这个爬虫。获取数据、解析数据、提取数据这三个步骤全部都由`selenium`来完成，写代码吧~

【提示】
前面的步骤，在关卡课程中都已经讲过，现在要做的是，在它的基础之上，获取中英文的“Python之禅”内容。
中文和英文版的“Python之禅”都在同样的标签中。所以，需要先获取所有的`class_=”content”`标签，然后再从中分别提取标题与正文。
具体的方法都写在了代码注释中。

由于教学系统中与你本地的浏览器设置方法不同，我给你提供了两份答案，一份可以在课程系统中运行，一份可以在你的本地运行。
【解答】

```
# 下面是只能在爬虫课系统中运行的答案：
from selenium.webdriver.chrome.webdriver import RemoteWebDriver # 从selenium库中调用RemoteWebDriver模块
from selenium.webdriver.chrome.options import Options  # 从options模块中调用Options类
import time


chrome_options = Options() # 实例化Option对象
chrome_options.add_argument('--headless') # 把Chrome设置为静默模式
driver = RemoteWebDriver("http://chromedriver.python-class-fos.svc:4444/wd/hub", chrome_options.to_capabilities()) # 设置浏览器引擎为远程浏览器


chrome_options = Options() # 实例化Option对象
chrome_options.add_argument('--headless') # 把Chrome设置为静默模式
driver = RemoteWebDriver("http://chromedriver.python-class-fos.svc:4444/wd/hub", chrome_options.to_capabilities()) # 设置浏览器引擎为远程浏览器


```

```
driver.get('https://localprod.pandateacher.com/python-manuscript/hello-spiderman/') # 访问页面
time.sleep(2) # 暂停两秒，等待浏览器缓冲


teacher = driver.find_element_by_id('teacher') # 找到【请输入你喜欢的老师】下面的输入框位置
teacher.send_keys('必须是吴枫呀') # 输入文字
assistant = driver.find_element_by_name('assistant') # 找到【请输入你喜欢的助教】下面的输入框位置
assistant.send_keys('都喜欢') # 输入文字
button = driver.find_element_by_class_name('sub') # 找到【提交】按钮
button.click() # 点击【提交】按钮
time.sleep(1)


contents = driver.find_elements_by_class_name('content') # 定位到Python之禅所在的标签
for content in contents:
    title = content.find_element_by_tag_name('h1').text # 提取标题
    chan = content.find_element_by_tag_name('p').text # 提取正文
    print(title + '\n' + chan + '\n') # 打印标题与正文
driver.close()




# 下面是只能在你的本地运行的答案：
from selenium import  webdriver # 从selenium库中调用webdriver模块
import time


driver = webdriver.Chrome() # 声明浏览器对象
driver.get('https://localprod.pandateacher.com/python-manuscript/hello-spiderman/') # 访问页面
time.sleep(2) # 暂停两秒，等待浏览器缓冲


teacher = driver.find_element_by_id('teacher') # 找到【请输入你喜欢的老师】下面的输入框位置
teacher.send_keys('必须是吴枫呀') # 输入文字
assistant = driver.find_element_by_name('assistant') # 找到【请输入你喜欢的助教】下面的输入框位置
assistant.send_keys('都喜欢') # 输入文字
button = driver.find_element_by_class_name('sub') # 找到【提交】按钮
button.click() # 点击【提交】按钮
time.sleep(1)


contents = driver.find_elements_by_class_name('content') # 定位到Python之禅所在的标签
```

```
for content in contents:
    title = content.find_element_by_tag_name('h1').text # 提取标题
    chan = content.find_element_by_tag_name('p').text # 提取正文
    print(title + '\n' + chan + '\n') # 打印标题与正文
driver.close()


```

第二种方法：selenium 与 BeautifulSoup配合

先用`selenium`获取到渲染完成的`Elements`中的网页源代码，然后，`BeautifulSoup`登场解析和提取数据。

【提示】
爬取到的文字中，前面会有一些空格，可以使用`replace(' ','')`去掉文字前面的空格。
这是字符串对象的一个方法，它的意思是，把第一个参数的字符串用第二个参数的字符串替代。

【解答】
具体的方法都写在了代码注释中。
由于教学系统中与你本地的浏览器设置方法不同，我给你提供了两份答案，一份可以在课程系统中运行，一份可以在你的本地运行。

```
# 下面是只能在爬虫课系统中运行的答案：
from selenium.webdriver.chrome.webdriver import RemoteWebDriver # 从selenium库中调用RemoteWebDriver模块
from selenium.webdriver.chrome.options import Options # 从options模块中调用Options类
from bs4 import BeautifulSoup
import time


chrome_options = Options() # 实例化Option对象
chrome_options.add_argument('--headless') # 把Chrome设置为静默模式
driver = RemoteWebDriver("http://chromedriver.python-class-fos.svc:4444/wd/hub", chrome_options.to_capabilities()) # 设置浏览器引擎为远程浏览器


driver.get('https://localprod.pandateacher.com/python-manuscript/hello-spiderman/') # 访问页面
time.sleep(2) # 暂停两秒，等待浏览器缓冲


teacher = driver.find_element_by_id('teacher') # 定位到【请输入你喜欢的老师】下面的输入框位置
teacher.send_keys('必须是吴枫呀') # 输入文字
```

```
assistant = driver.find_element_by_name('assistant') # 定位到【请输入你喜欢的助教】下面的输入框位置
assistant.send_keys('都喜欢') # 输入文字
button = driver.find_element_by_class_name('sub') # 定位到【提交】按钮
button.click() # 点击【提交】按钮
time.sleep(1) # 等待一秒


pageSource = driver.page_source # 获取页面信息
soup = BeautifulSoup(pageSource,'html.parser')  # 使用bs解析网页
contents = soup.find_all(class_="content") # 找到源代码Python之禅中文版和英文版所在的元素
for content in contents:  # 遍历列表
    title = content.find('h1').text # 提取标题
    chan = content.find('p').text.replace('  ','') # 提取Python之禅的正文，并且去掉文字前面的所有空格
    print(title + chan + '\n') # 打印Python之禅的标题与正文
driver.close()




# 下面是只能在你的本地运行的答案：
from selenium import  webdriver # 从selenium库总调用webdriver模块
import time
from bs4 import BeautifulSoup


driver = webdriver.Chrome() # 声明浏览器对象
driver.get('https://localprod.pandateacher.com/python-manuscript/hello-spiderman/') # 访问页面
time.sleep(2) # 暂停两秒，等待浏览器缓冲


teacher = driver.find_element_by_id('teacher') # 定位到【请输入你喜欢的老师】下面的输入框位置
teacher.send_keys('必须是吴枫呀') # 输入文字
assistant = driver.find_element_by_name('assistant') # 定位到【请输入你喜欢的助教】下面的输入框位置
assistant.send_keys('都喜欢') # 输入文字
button = driver.find_element_by_class_name('sub') # 定位到【提交】按钮
button.click() # 点击【提交】按钮
time.sleep(1) # 等待一秒


pageSource = driver.page_source # 获取页面信息
soup = BeautifulSoup(pageSource,'html.parser')  # 使用bs解析网页
contents = soup.find_all(class_="content") # 找到源代码Python之禅中文版和英文版所在的元素
for content in contents:  # 遍历列表
    title = content.find('h1').text # 提取标题
```

```
    chan = content.find('p').text.replace('  ','') # 提取Python之禅的正文，并且去掉文字前面的所有空格
    print(title + chan + '\n') # 打印Python之禅的标题与正文
driver.close()


```

第10关

练习-周末吃什么-参考

第一步：明确目标

先明确目标：我们曾在第3关爬取了下厨房网站中的“本周最受欢迎菜谱”，现在，我们完善这个程序，让程序在每个周五爬取数据，并把菜谱发送到我们的邮箱。

【讲解】

先明确目标：我们曾在第3关爬取了下厨房网站中的“本周最受欢迎菜谱”，现在，我们完善这个程序，让程序在每个周五爬取数据，并把菜谱发送到我们的邮箱。
该项目和第10关课堂的项目是非常相似的。

第二步：分析过程

再分析过程：这个程序一共分为三部分：爬虫、通知和定时。

【讲解】

这个程序一共分为三部分，知识我们都掌握了。
0.爬虫：爬取下厨房网站中本周最欢迎菜谱的菜名、链接、原材料。
1.通知：用smtplib、email库来发送邮件。
2.定时：用schedule和time库定时执行程序。
我们分别写出来，然后封装成函数。
先把每个程序写出来，然后拼装到一起；鉴于爬虫程序已经学过，就直接把代码提供给大家。

第三步：代码实现

接下来就是写代码啦。
0.爬虫：爬虫代码已经在课堂上学过，所以直接把爬虫代码提供给大家，并请大家先封装爬虫代码：

```
import requests
from bs4 import BeautifulSoup


res_foods = requests.get('http://www.xiachufang.com/explore/')
bs_foods = BeautifulSoup(res_foods.text,'html.parser')
list_foods = bs_foods.find_all('div',class_='info pure-u')

```

```

list_all = []


for food in list_foods:
    tag_a = food.find('a')
    name = tag_a.text[17:-13]
    URL = 'http://www.xiachufang.com'+tag_a['href']
    tag_p = food.find('p',class_='ing ellipsis')
    ingredients = tag_p.text[1:-1]
    list_all.append([name,URL,ingredients])
print(list_all)


```

1.邮件：邮件代码是第10关所学的内容，下面提供代码给大家，但最好是回忆不起来再看；写完代码后请大家封装代码。（仍然提醒同学们，学习系统会记录大家输入的内容。考虑到信息隐私的问题，大家不要在这里输入自己的邮箱密码或账号。因此，请你在本地运行邮件相关的代码）

```
import smtplib
from email.mime.text import MIMEText
from email.header import Header
#引入smtplib、MIMETex和Header

mailhost='smtp.qq.com'
#把qq邮箱的服务器地址赋值到变量mailhost上，地址应为字符串格式
qqmail = smtplib.SMTP()
#实例化一个smtplib模块里的SMTP类的对象，这样就可以调用SMTP对象的方法和属性了
qqmail.connect(mailhost,25)
#连接服务器，第一个参数是服务器地址，第二个参数是SMTP端口号。
#以上，皆为连接服务器。

account = input('请输入你的邮箱：')
#获取邮箱账号，为字符串格式
password = input('请输入你的密码：')
#获取邮箱密码，为字符串格式
qqmail.login(account,password)
#登录邮箱，第一个参数为邮箱账号，第二个参数为邮箱密码
#以上，皆为登录邮箱。

receiver=input('请输入收件人的邮箱：')
#获取收件人的邮箱。

content=input('请输入邮件正文：')
#输入你的邮件正文，为字符串格式
message = MIMEText(content, 'plain', 'utf-8')
#实例化一个MIMEText邮件对象，该对象需要写进三个参数，分别是邮件正文，文本格式和编码
subject = input('请输入你的邮件主题：')
#输入你的邮件主题，为字符串格式
message['Subject'] = Header(subject, 'utf-8')
```

```
#在等号的右边是实例化了一个Header邮件头对象，该对象需要写入两个参数，分别是邮件主题和编码，然后赋值给等号左边的变量message['Subject']。
#以上，为填写主题和正文。

try:
    qqmail.sendmail(account, receiver, message.as_string())
    print ('邮件发送成功')
except:
    print ('邮件发送失败')
qqmail.quit()
#以上为发送邮件和退出邮箱
```

2.定时：定时功能是第10关教的内容，代码如下，下面提供代码给大家，但最好回忆不起来再看；写完代码后请大家封装代码。

```
import schedule
import time
#引入schedule和time
def job():
    print("I'm working...")
#定义一个叫job的函数，函数的功能是打印'I'm working...'
schedule.every(10).minutes.do(job)        #部署每10分钟执行一次job()函数的任务
schedule.every().hour.do(job)             #部署每×小时执行一次job()函数的任务
schedule.every().day.at("10:30").do(job) #部署在每天的10:30执行job()函数的任务
schedule.every().monday.do(job)           #部署每个星期一执行job()函数的任务
schedule.every().wednesday.at("13:15").do(job)#部署每周三的13：15执行函数的任务
while True:
    schedule.run_pending()
    time.sleep(1)

```

【解答】

老师提供的参考答案是这样的：

```
import requests
import smtplib
import schedule
import time
from bs4 import BeautifulSoup
from email.mime.text import MIMEText
from email.header import Header


account = input('请输入你的邮箱：')
password = input('请输入你的密码：')
receiver = input('请输入收件人的邮箱：')

def recipe_spider():
    res_foods = requests.get('http://www.xiachufang.com/explore/')
```

```
    bs_foods = BeautifulSoup(res_foods.text,'html.parser')
    list_foods = bs_foods.find_all('div',class_='info pure-u')
    list_all = ''
    num=0
    for food in list_foods:
        num=num+1
        tag_a = food.find('a')
        name = tag_a.text.strip()
        url = 'http://www.xiachufang.com'+tag_a['href']
        tag_p = food.find('p',class_='ing ellipsis')
        ingredients = tag_p.text.strip()
        food_info = '''
        序号: %s
        菜名: %s
        链接: %s
        原料: %s
        '''%(num,name,url,ingredients)
        list_all=list_all+food_info
    return(list_all)

def send_email(list_all):
    global account,password,receiver
    mailhost='smtp.qq.com'
    qqmail = smtplib.SMTP()
    qqmail.connect(mailhost,25)
    qqmail.login(account,password)
    content= '亲爱的，本周的热门菜谱如下'+list_all
    message = MIMEText(content, 'plain', 'utf-8')
    subject = '周末吃个啥'
    message['Subject'] = Header(subject, 'utf-8')
    try:
        qqmail.sendmail(account, receiver, message.as_string())
        print ('邮件发送成功')
    except:
        print ('邮件发送失败')
    qqmail.quit()

def job():
    print('开始一次任务')
    list_all = recipe_spider()
    send_email(list_all)
    print('任务完成')

schedule.every().friday.at("18:00").do(job)#部署每周三的13：15执行函数的任务
while True:
    schedule.run_pending()
    time.sleep(1)

```

练习–看个电影吧–参考

第一步：明确目标

先明确目标：每个周五，程序在豆瓣TOP250榜单中随机选取三部电影，然后去爬取三部电影的下载链接，并把链接发送到我们的邮箱。

【讲解】

先明确目标：每个周五，程序在豆瓣TOP250榜单中随机选取三部电影，然后去爬取三部电影的下载链接，并把链接发送到我们的邮箱。该项目和第10关课堂的是非常相似的。

第二步：分析过程

我们来看看这个项目的实现思路。

【讲解】

这个程序一共分为四部分：

0.电影榜单爬虫：爬取豆瓣电影Top250的榜单，并存储文件到本地。

1.电影链接爬虫：每周五读取榜单中的三部电影，然后去爬取电影的下载链接。

2.通知功能：再把爬到的链接以邮件的形式发送给自己。

3.定时功能：用schedule和time库定时执行程序。

可以把4段代码分别写出来，然后封装成函数。鉴于爬虫程序已经做过练习，会直接把代码提供给大家。

电影榜单的爬虫是第3关课后的必做练习，电影链接的爬虫是第3关的选做练习，所以该练习的重点不会放在爬虫上。

第三步：代码实现（上）

【代码实现】又分为上中下。上、中分别把四段代码写出来，下是封装组装代码。

上：每周五晚上去豆瓣电影Top250的榜单上随机抽取3部，然后去下载这3部电影的链接，并打印出来。

先把两段爬虫的代码提供给大家，请大家先阅读，然后去写代码。（当然，你用你自己写的代码也是一样的，只要最后实现的功能和题目一致）

```
#这是爬取豆瓣电影Top250，并存为本地csv的代码
import requests,  random, csv
from bs4 import BeautifulSoup
csv_file=open('movieTop.csv', 'w', newline='',encoding='utf-8')
writer = csv.writer(csv_file)

for x in range(10):
    url = 'https://movie.douban.com/top250?start=' + str(x*25) +
'&filter='
    res = requests.get(url)
    bs = BeautifulSoup(res.text, 'html.parser')
    bs = bs.find('ol', class_="grid_view")
    for titles in bs.find_all('li'):
        title = titles.find('span', class_="title").text
        list1 = [title]
        writer.writerow(list1)
csv_file.close()

```

```
# 这是爬电影的下载链接的代码
import requests
from bs4 import BeautifulSoup
from urllib.request import quote

movie=input('你想看什么电影？')
gbkmovie = movie.encode('gbk')
urlsearch = 'http://s.ygdy8.com/plus/so.php?typeid=1&keyword='+quote(gbkmovie)
res = requests.get(urlsearch)
res.encoding='gbk'
soup_movie = BeautifulSoup(res.text,'html.parser')
urlpart=soup_movie.find(class_="co_content8").find_all('table')
if urlpart:
    urlpart=urlpart[0].find('a')['href']
    urlmovie='https://www.ygdy8.com/'+urlpart
    res1=requests.get(urlmovie)
    res1.encoding='gbk'
    soup_movie1=BeautifulSoup(res1.text,'html.parser')

urldownload=soup_movie1.find('div',id="Zoom").find('span').find('table').find('a')['href']
    print(urldownload)
else:
    print('没有'+movie+'的链接')

```

好，现在，我们需要先读取存储到本地的电影榜单的csv文件，然后用random库来随机抽取三部电影，再去下载相应的电影链接，并把电影链接打印出来。
请开始写代码吧。

【提示】
如果对这里的爬虫代码有疑惑，建议先完成第3关的练习和第6关的练习。

【解答】

```
import requests,csv,random
from bs4 import BeautifulSoup
from urllib.request import quote
# 以上，为引入相应的库。


csv_file=open('movieTop.csv', 'w', newline='',encoding='utf-8')
writer = csv.writer(csv_file)
for x in range(10):
    url = 'https://movie.douban.com/top250?start=' + str(x*25) + '&filter='
    res = requests.get(url)
    bs = BeautifulSoup(res.text, 'html.parser')
    bs = bs.find('ol', class_="grid_view")
```

```
    for titles in bs.find_all('li'):
        title = titles.find('span', class_="title").text
        list1 = [title]
        writer.writerow(list1)
csv_file.close()
# 以上，为爬取豆瓣电影Top250的榜单，并存储为本地的csv文件。

movielist=[]
csv_file=open('movieTop.csv','r',newline='',encoding='utf-8')
reader=csv.reader(csv_file)
for row in reader:
    movielist.append(row[0])
# 以上，为读取豆瓣电影Top250榜单的csv文件，并写入列表movielist中。
three_movies=random.sample(movielist,3)
# 以上，是从列表movielist中，随机抽取三部电影，取出来的是一个列表。
for movie in three_movies:
    # 以上，是把电影名从列表中取出来，并把其数据类型变为字符串。下面开始，就是你熟悉的下载电影链接的代码了。
    gbkmovie = movie.encode('gbk')
    urlsearch = 'http://s.ygdy8.com/plus/so.php?typeid=1&keyword='+quote(gbkmovie)
    res = requests.get(urlsearch)
    res.encoding='gbk'
    soup_movie = BeautifulSoup(res.text,'html.parser')
    urlpart=soup_movie.find(class_="co_content8").find_all('table')
    if urlpart:
        urlpart=urlpart[0].find('a')['href']
        urlmovie='https://www.ygdy8.com/'+urlpart
        res1=requests.get(urlmovie)
        res1.encoding='gbk'
        soup_movie1=BeautifulSoup(res1.text,'html.parser')

urldownload=soup_movie1.find('div',id="Zoom").find('span').find('table').find('a')['href']
        content=movie+'\n'+urldownload
        print(content)
    else:
        content='没有'+movie+'的下载链接'
        print(content)


```

第四步：代码实现（中）

接下来，我们来完成发送邮件，以及定时功能的代码，然后在下一步我们再封装、组合四段代码。邮件代码是第10关所学的内容，下面提供代码给大家。（仍然提醒同学们，学习系统会记录大家输入的内容。考虑到信息隐私的问题，大家不要在这里输入自己的邮箱密码或账号。因此，请你在本地运行邮件相关的代码）

```
import smtplib
from email.mime.text import MIMEText
from email.header import Header
```

```
#引入smtplib、MIMETex和Header

mailhost='smtp.qq.com'
#把qq邮箱的服务器地址赋值到变量mailhost上，地址应为字符串格式
qqmail = smtplib.SMTP()
#实例化一个smtplib模块里的SMTP类的对象，这样就可以调用SMTP对象的方法和属性了
qqmail.connect(mailhost,25)
#连接服务器，第一个参数是服务器地址，第二个参数是SMTP端口号。
#以上，皆为连接服务器。

account = input('请输入你的邮箱：')
#获取邮箱账号，为字符串格式
password = input('请输入你的密码：')
#获取邮箱密码，为字符串格式
qqmail.login(account,password)
#登录邮箱，第一个参数为邮箱账号，第二个参数为邮箱密码
#以上，皆为登录邮箱。

receiver=input('请输入收件人的邮箱：')
#获取收件人的邮箱。

#content为上面的电影链接
#输入你的邮件正文，为字符串格式
message = MIMEText(content, 'plain', 'utf-8')
#实例化一个MIMEText邮件对象，该对象需要写进三个参数，分别是邮件正文，文本格式和编码
subject = '电影链接'
#输入你的邮件主题，为字符串格式
message['Subject'] = Header(subject, 'utf-8')
#在等号的右边是实例化了一个Header邮件头对象，该对象需要写入两个参数，分别是邮件主题和编码，然后赋值给等号左边的变量message['Subject']。
#以上，为填写主题和正文。

try:
    qqmail.sendmail(account, receiver, message.as_string())
    print ('邮件发送成功')
except:
    print ('邮件发送失败')
qqmail.quit()
#以上为发送邮件和退出邮箱

#定时功能是第10关教的内容，代码如下，下面提供代码给大家，但最好回忆不起来再看。
import schedule
import time
#引入schedule和time
def job():
    print("I'm working...")
#定义一个叫job的函数，函数的功能是打印'I'm working...'

schedule.every(10).minutes.do(job)     #部署每10分钟执行一次job()函数的任务
schedule.every().hour.do(job)           #部署每×小时执行一次job()函数的任务
schedule.every().day.at("10:30").do(job)#部署每天的10:30执行job()函数任务
```

```
schedule.every().monday.do(job)         #部署每个星期一执行job()函数的任务
schedule.every().wednesday.at("13:15").do(job)
#部署每周三的13：15执行函数的任务

while True:
    schedule.run_pending()
    time.sleep(1)


```

【解答】

老师提供的参考答案是这样的：

```
import requests,csv,random,smtplib,schedule,time
from bs4 import BeautifulSoup
from urllib.request import quote
from email.mime.text import MIMEText
from email.header import Header


def get_movielist():
    csv_file=open('movieTop.csv', 'w', newline='',encoding='utf-8')
    writer = csv.writer(csv_file)
    for x in range(10):
        url = 'https://movie.douban.com/top250?start=' + str(x*25) +
'&filter='
        res = requests.get(url)
        bs = BeautifulSoup(res.text, 'html.parser')
        bs = bs.find('ol', class_="grid_view")
        for titles in bs.find_all('li'):
            title = titles.find('span', class_="title").text
            list1 = [title]
            writer.writerow(list1)
    csv_file.close()

def get_randommovie():
    movielist=[]
    csv_file=open('movieTop.csv','r',newline='',encoding='utf-8')
    reader=csv.reader(csv_file)
    for row in reader:
        movielist.append(row[0])
    three_movies=random.sample(movielist,3)
    contents=''
    for movie in three_movies:
        gbkmovie = movie.encode('gbk')
        urlsearch = 'http://s.ygdy8.com/plus/so.php?
typeid=1&keyword='+quote(gbkmovie)
        res = requests.get(urlsearch)
        res.encoding='gbk'
        soup_movie = BeautifulSoup(res.text,'html.parser')
```

```
        urlpart=soup_movie.find(class_="co_content8").find_all('table')
        if urlpart:
            urlpart=urlpart[0].find('a')['href']
            urlmovie='https://www.ygdy8.com/'+urlpart
            res1=requests.get(urlmovie)
            res1.encoding='gbk'
            soup_movie1=BeautifulSoup(res1.text,'html.parser')

urldownload=soup_movie1.find('div',id="Zoom").find('span').find('table').find('a')['href']
            content=movie+'\n'+urldownload+'\n\n'
            print(content)
            contents=contents+content
        else:
            content='没有'+movie+'的下载链接'
            print(content)
    return contents

def send_movielink(contents):
    mailhost='smtp.qq.com'
    qqmail = smtplib.SMTP()
    qqmail.connect(mailhost,25)
    account = '××××××××××@qq.com' # 因为是自己发给自己，所以邮箱账号、密码都可以提前设置好，当然，也可以发给别人啦
    password = '××××××××××××××' # 因为是自己发给自己，所以邮箱账号、密码都可以提前设置好，当然，也可以发给别人啦。
    qqmail.login(account,password)
    receiver='××××××××××@qq.com'  # 因为是自己发给自己，所以邮箱账号、密码都可以提前设置好，当然，也可以发给别人啦。
    message = MIMEText(contents, 'plain', 'utf-8')
    subject = '电影链接'
    message['Subject'] = Header(subject, 'utf-8')
    try:
        qqmail.sendmail(account, receiver, message.as_string())
        print ('邮件发送成功')
    except:
        print ('邮件发送失败')
    qqmail.quit()

def job():
    get_movielist()
    contents=get_randommovie()
    send_movielink(contents)

schedule.every().friday.at("18:00").do(job)
while True:
    schedule.run_pending()
    time.sleep(1)


```