

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】

```
1  import requests, random, bs4
2
3  for x in range(10):
4      url = 'https://movie.douban.com/top250?start=' + str(x*25) +
5          '&filter='
6      res = requests.get(url)
7      bs = bs4.BeautifulSoup(res.text, 'html.parser')
8      bs = bs.find('ol', class_='grid_view')
9      for titles in bs.find_all('li'):
10         num = titles.find('em', class_='').text
11         title = titles.find('span', class_='title').text
12         comment = titles.find('span', class_='rating_num').text
13         url_movie = titles.find('a')['href']
14
15         if titles.find('span', class_='inq') != None:
16             tes = titles.find('span', class_='inq').text
17             print(num + '.' + title + '——' + comment + '\n' + '推荐语: ' +
18                 tes + '\n' + url_movie)
19         else:
20             print(num + '.' + title + '——' + comment + '\n' + '\n' +
21                 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】

```
1 import requests, random, bs4, openpyxl
2
3 wb=openpyxl.Workbook()
4 #创建工作簿
5 sheet=wb.active
6 #获取工作簿的活动表
7 sheet.title='movies'
8 #工作表重命名
9 sheet['A1'] ='序号'          #加表头，给A1单元格赋值
10 sheet['B1'] ='电影名'        #加表头，给B1单元格赋值
11 sheet['C1'] ='评分'          #加表头，给C1单元格赋值
12 sheet['D1'] ='推荐语'        #加表头，给D1单元格赋值
13 sheet['E1'] ='链接'          #加表头，给E1单元格赋值
14
15 for x in range(10):
16     url = 'https://movie.douban.com/top250?start=' + str(x*25) +
17     '&filter='
18     res = requests.get(url)
19     bs = bs4.BeautifulSoup(res.text, 'html.parser')
20     bs = bs.find('ol', class_="grid_view")
21     for titles in bs.find_all('li'):
22         num = titles.find('em',class_="").text
23         title = titles.find('span', class_="title").text
24         comment = titles.find('span',class_="rating_num").text
25         url_movie = titles.find('a')['href']
26
27         if titles.find('span',class_="inq") != None:
28             tes = titles.find('span',class_="inq").text
29             sheet.append([num, title, comment, tes, url_movie])
30             # 把num, title, comment, tes和url_movie写成列表，用append函数多行
31             写入Excel
32             print(num + '.' + title + '——' + comment + '\n' + '推荐语: ' +
33             tes + '\n' + url_movie)
34         else:
35             sheet.append([num, title, comment, None,url_movie])
36             print(num + '.' + title + '——' + comment + '\n' + '\n' +
37             url_movie)
38
39 wb.save('movieTop250.xlsx')
40 #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】

```
1 import requests, random, bs4, csv
2 #引用csv模块。
3 csv_file=open('movieTop250.csv', 'w', newline='')
4 #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】

```
1 import requests, random, bs4, csv
2 #引用csv模块。
3 csv_file=open('movieTop250.csv', 'w', newline='')
4 #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=''。
5 writer = csv.writer(csv_file)
6 # 用csv.writer()函数创建一个writer对象。
7 writer.writerow(['序号', '电影名', '评分', '推荐语', '链接'])
8 #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'
9
10 for x in range(10):
11     url = 'https://movie.douban.com/top250?start=' + str(x*25) +
        '&filter='
12     res = requests.get(url)
13     bs = bs4.BeautifulSoup(res.text, 'html.parser')
14     bs = bs.find('ol', class_="grid_view")
15     for titles in bs.find_all('li'):
16         num = titles.find('em', class_="").text
17         title = titles.find('span', class_="title").text
18         comment = titles.find('span', class_="rating_num").text
19         url_movie = titles.find('a')['href']
20
21         if titles.find('span', class_="inq") != None:
22             tes = titles.find('span', class_="inq").text
23             # 把num, title, comment, tes和url_movie写成列表，用append函数多行
              写入Excel
24             writer.writerow([num + '.' + title + '—' + comment + '\n' +
                '推荐语: ' + tes + '\n' + url_movie])
25         else:
26             writer.writerow([num + '.' + title + '—' + comment + '\n'
                + '\n' + url_movie])
27
28 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———自动生成错题本。

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】

```
1 import requests
2
3
4 lin=requests.get('https://www.shanbay.com/api/v1/vocabtest/category/')
5 #先用requests下载链接。
6 js_link = lin.json()
7 #解析下载得到的内容。
8 bianhao = int(input('请输入你选择的词库编号，按Enter确认
9 1, GMAT 2, 考研 3, 高考 4, 四级 5, 六级
10 6, 英专 7, 托福 8, GRE 9, 雅思 10, 任意
11 >'))
12 #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
13 ciku = js_link['data'][bianhao-1][0]
14 print(ciku)
15
```

(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】

```
1 test =
  requests.get('https://www.shanbay.com/api/v1/vocabtest/vocabularies/?
  category='+ciku)
2 #下载用于测试的50个单词。
3 words = test.json()
4 #对测试的单词进行解析。
5 print(words)
6
```

(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】

```
1 danci = []
2 #新增一个list, 用于统计用户认识的单词
3 words_knows = []
4 not_knows = []
5 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: ')
6 n=0
7 for x in words['data']:
8     n=n+1
9     print ("\n第"+str(n)+'个: '+x['content'])
10    #加一个\n, 用于换行。
11    answer = input('认识请敲Y, 否则敲Enter: ')
12    if answer == 'Y':
13        danci.append(x['content'])
14        #把用户认识的单词, 追加进danci这个list。
15        words_knows.append(x)
16    else:
17        not_knows.append(x)
18
19
20 print ('\n在上述'+str(len(words['data']))+'个单词当中,
21 有'+str(len(danci))+'个是你觉得自己认识的, 它们是: ')
22 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】

```
1 print ('现在我们来检测一下, 你有没有真正掌握它们: ')
2 wrong_words = []
3 right_num = 0
4 for y in words_knows:
5     print('\n\n'+ 'A: '+y['definition_choices'][0]['definition'])
6     #我们改用A、B、C、D, 不再用rank值, 下同
7     print('B: '+y['definition_choices'][1]['definition'])
8     print('C: '+y['definition_choices'][2]['definition'])
9     print('D: '+y['definition_choices'][3]['definition'])
10    xuanze = input('请选择单词\' '+y['content']+'\'的正确翻译: ')
11    dic = {'A':y['definition_choices'][0]
12           ['rank'], 'B':y['definition_choices'][1]
13           ['rank'], 'C':y['definition_choices'][2]
14           ['rank'], 'D':y['definition_choices'][3]['rank']}
15    #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。
16    if dic[xuanze] == y['rank']:
17        #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。
```

```

15         right_num += 1
16     else:
17         wrong_words.append(y)
18

```

(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】

```

1  import requests
2
3  link = requests.get('https://www.shanbay.com/api/v1/vocabtest/category/')
4  #先用requests下载链接。
5  js_link = link.json()
6  #解析下载得到的内容。
7  bianhao = int(input('''请输入你选择的词库编号，按Enter确认
8  1, GMAT  2, 考研  3, 高考  4, 四级  5, 六级
9  6, 英专  7, 托福  8, GRE  9, 雅思  10, 任意
10 >'''))
11 #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
12 ciku = js_link['data'][bianhao-1][0]
13 #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。
14 test =
    requests.get('https://www.shanbay.com/api/v1/vocabtest/vocabularies/?
        category='+ciku)
15 #下载用于测试的50个单词。
16 words = test.json()
17 #对test进行解析。
18 danci = []
19 #新增一个list，用于统计用户认识的单词
20 words_knows = []
21 #创建一个空的列表，用于记录用户认识的单词。
22 not_knows = []
23 #创建一个空的列表，用于记录用户不认识的单词。
24 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: ')
25 n=0
26 for x in words['data']:
27     #启动一个循环，循环的次数等于单词的数量。
28     n=n+1
29     print ("\n第"+str(n)+'个: '+x['content'])
30     #加一个\n，用于换行。
31     answer = input('认识请敲Y，否则敲Enter: ')
32     #让用户输入自己是否认识。
33     if answer == 'Y':
34         #如果用户认识：
35         danci.append(x['content'])

```

```

36         words_knows.append(x)
37         #就把这个单词，追加进列表words_knows。
38     else:
39         #否则
40         not_knows.append(x)
41         #就把这个单词，追加进列表not_knows。
42
43
44
45
46 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：')
47 print(danci)
48
49
50
51
52 print ('现在我们来检测一下，你有没有真正掌握它们：')
53 wrong_words = []
54 right_num = 0
55 for y in words_knows:
56     print('\n\n'+A:'+y['definition_choices'][0]['definition'])
57     #我们改用A、B、C、D，不再用rank值，下同
58     print('B:'+y['definition_choices'][1]['definition'])
59     print('C:'+y['definition_choices'][2]['definition'])
60     print('D:'+y['definition_choices'][3]['definition'])
61     xuanze = input('请选择单词\''+y['content']+'\''的正确翻译（填写数字即可）：')
62     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']}
63     #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。
64     if dic[xuanze] == y['rank']:
65         #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。
66         right_num += 1
67     else:
68         wrong_words.append(y)
69
70
71 print ('现在，到了公布成绩的时刻:')
72 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。')
73 #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。
74
75
76 save = input ('是否打印并保存你的错词集？填入Y或N：')
77 #询问用户，是否要打印并保存错题集。
78 if save == 'Y':

```



```

79 #如果用户说是:
80     f = open('错题集.txt', 'a+')
81     #在当前目录下, 创建一个错题集.txt的文档。
82     print ('你记错的单词有: ')
83     f.write('你记错的单词有: \n')
84     #写入"你记错的单词有: \n"
85     m=0
86     for z in wrong_words:
87         #启动一个循环, 循环的次数等于, 用户的错词数:
88         m=m+1
89         print (z['content'])
90         #打印每一个错词。
91         f.write(str(m+1) + '. ' + z['content']+'\n')
92         #写入序号, 写入错词。
93     print ('你不认识的单词有: ')
94     f.write('你没记住的单词有: \n')
95     #写入"你没记住的单词有: \n"
96     s=0
97     for x in not_knows:
98         #启动一个循环, 循环的次数等于, 用户不认识的单词数。
99         print (x['content'])
100        #打印每一个不认识的单词。
101        f.write(str(s+1) + '. ' + x['content']+'\n')
102        #写入序号, 写入用户不认识的词汇。
103    print ('错词和没记住的词已保存至当前文件目录下, 下次见! ')
104    #告诉用户, 文件已经保存好。
105    #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。
106 else:
107     #如果用户不想保存:
108     print('下次见! ')
109     #输出“下次见!”
110

```

三步：参考答案

【讲解】

参考答案：

```

1  import requests
2
3
4  link = requests.get('https://www.shanbay.com/api/v1/vocabtest/category/')
5  #先用requests下载链接。
6  js_link = link.json()
7  #解析下载得到的内容。
8  bianhao = int(input('请输入你选择的词库编号, 按Enter确认
9  1, GMAT  2, 考研  3, 高考  4, 四级  5, 六级
10 6, 英专  7, 托福  8, GRE  9, 雅思  10, 任意
11 >'))
12 #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
13 ciku = js_link['data'][bianhao-1][0]

```

```

14 #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。
15 test =
    requests.get('https://www.shanbay.com/api/v1/vocabtest/vocabularies/?
    category='+ciku)
16 #下载用于测试的50个单词。
17 words = test.json()
18 #对test进行解析。
19 danci = []
20 #新增一个list，用于统计用户认识的单词
21 words_knows = []
22 #创建一个空的列表，用于记录用户认识的单词。
23 not_knows = []
24 #创建一个空的列表，用于记录用户不认识的单词。
25 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: ')
26 n=0
27 for x in words['data']:
28     #启动一个循环，循环的次数等于单词的数量。
29     n=n+1
30     print ("\n第"+str(n)+'个: '+x['content'])
31     #加一个\n，用于换行。
32     answer = input('认识请敲Y，否则敲Enter: ')
33     #让用户输入自己是否认识。
34     if answer == 'Y':
35         #如果用户认识：
36         danci.append(x['content'])
37         words_knows.append(x)
38         #就把这个单词，追加进列表words_knows。
39     else:
40         #否则
41         not_knows.append(x)
42         #就把这个单词，追加进列表not_knows。
43
44
45
46
47 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: ')
48 print(danci)
49
50
51
52
53 print ('现在我们来检测一下，你有没有真正掌握它们: ')
54 wrong_words = []
55 right_num = 0
56 for y in words_knows:
57     print('\n\n'+A: '+y['definition_choices'][0]['definition'])
58     #我们改用A、B、C、D，不再用rank值，下同

```

```

59     print('B:'+y['definition_choices'][1]['definition'])
60     print('C:'+y['definition_choices'][2]['definition'])
61     print('D:'+y['definition_choices'][3]['definition'])
62     xuanze = input('请选择单词\'"+y['content']+"\'的正确翻译（填写数字即可）： ')
63     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']}
64     #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。
65     if dic[xuanze] == y['rank']:
66         #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。
67         right_num += 1
68     else:
69         wrong_words.append(y)
70
71
72 print ('现在，到了公布成绩的时刻:')
73 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。')
74 #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。
75
76
77 save = input ('是否打印并保存你的错题集？填入Y或N: ')
78 #询问用户，是否要打印并保存错题集。
79 if save == 'Y':
80     #如果用户说是：
81     f = open('错题集.txt', 'a+')
82     #在当前目录下，创建一个错题集.txt的文档。
83     print ('你记错的单词有: ')
84     f.write('你记错的单词有: \n')
85     #写入"你记错的单词有: \n"
86     m=0
87     for z in wrong_words:
88         #启动一个循环，循环的次数等于，用户的错词数：
89         m=m+1
90         print (z['content'])
91         #打印每一个错词。
92         f.write(str(m+1) +'. '+ z['content']+'\n')
93         #写入序号，写入错词。
94     print ('你不认识的单词有: ')
95     f.write('你没记住的单词有: \n')
96     #写入"你没记住的单词有: \n"
97     s=0
98     for x in not_knows:
99         #启动一个循环，循环的次数等于，用户不认识的单词数。
100         print (x['content'])
101         #打印每一个不认识的单词。
102         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：[\(https://www.ele.me/home/\)](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/ws100xkmpznf?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/ws100xkmpznf?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：

```
1 import requests
2 session = requests.session()
3 #创建会话。
4 headers = {
5     'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
6     AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110 Safari/537.36'
7 }
8 #添加请求头，避免被反爬虫。
9 url = 'https://wordpress-edu-3autumn.localprod.forc.work/wp-login.php'
10 #登录的网址。
11 data = {'log': input('请输入你的账号:'),
12         'pwd': input('请输入你的密码:'),
13         'wp-submit': '登录',
14         'redirect_to': 'https://wordpress-edu-3autumn.localprod.forc.work/wp-admin/',
15         'testcookie': '1'}
16 #登录的参数。
```

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

【解答】

```
1 import requests
2 session = requests.session()
3 # 创建会话
4 headers = {
5     'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
6     AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110 Safari/537.36'
7 }
8 # 添加请求头，避免被反爬虫
9 url_1 = 'https://h5.ele.me/restapi/eus/login/mobile_send_code'
10 # 发送验证码的网址
11 tel = input('请输入手机号码: ')
12 data_1 = {'captcha_hash': '',
13           'captcha_value': '',
14           'mobile': tel,
15           'scf': ''}
16 # 发送验证码的参数
17 token = session.post(url_1, headers=headers, data=data_1).json()
18 ['validate_token']
19 # 在会话下，模拟获取验证码的请求
20 url_2 = 'https://h5.ele.me/restapi/eus/login/login_by_mobile'
21 code = input('请输入手机验证码: ')
22 data_2 = {'mobile': tel,
23           'scf': 'ms',
24           'validate_code': code,
25           'validate_token': token}
26 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】

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

```

5 place = input('请输入你的收货地址: ')
6 # 使用input输入收货地址, 赋值给place。
7 # 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”。
8 params =
    {'extras[]': 'count', 'geohash': 'ws105rz9smwm', 'keyword': place, 'limit': '20',
    'type': 'nearby'}
9 # 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。
10 address_res = requests.get(address_url, params=params)
11 # 发起请求, 将响应的结果, 赋值给address_res
12 address_json = address_res.json()
13 # 将响应的结果转为列表/字典。
14
15 print('以下, 是与'+place+'相关的位置信息: \n')
16 n=0
17 # 添加一个计数器, 作为序号。
18 for address in address_json:
19 # 遍历我们刚爬取的地址列表。
20     print(str(n)+' . '+address['name']+' : '+address['short_address']+'\n')
21     # 打印序号, 地址名, 短地址。
22     n = n+1
23     # 给计数器加1。
24 address_num = int(input('请输入您选择位置的序号: '))
25 # 让用户选择序号。
26 final_address = address_json[address_num]
27 # 确认地址。
28
29 print(final_address['geohash'])
30 print(final_address['latitude'])
31 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】

```

1 import requests
2 session = requests.session()
3
4 headers = {
5     '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'
6 }
7 url_1 = 'https://h5.ele.me/restapi/eus/login/mobile_send_code'
8 tel = input('请输入手机号码: ')
9 data_1 = {'captcha_hash': '',
10           'captcha_value': '',
11           'mobile': tel,
12           'scf': ''}
13

```

```

14 token = session.post(url_1, headers=headers, data=data_1).json()
15     ['validate_token']
16
17 url_2 = 'https://h5.ele.me/restapi/eus/login/login_by_mobile'
18 code = input('请输入手机验证码: ')
19 data_2 = {'mobile':tel,
20           'scf':'ms',
21           'validate_code':code,
22           'validate_token':token}
23
24 session.post(url_2,headers=headers,data=data_2)
25
26 address_url = 'https://www.ele.me/restapi/v2/pois?'
27 place = input('请输入你的收货地址: ')
28 params =
29     {'extras[]':'count','geohash':'ws105rz9smwm','keyword':place,'limit':'20',
30     'type':'nearby'}
31 # 这里使用了深圳的geohash
32
33 address_res = requests.get(address_url,params=params)
34 address_json = address_res.json()
35
36 print('以下, 是与'+place+'相关的位置信息: \n')
37 n=0
38 for address in address_json:
39     print(str(n)+' . '+address['name']+' : '+address['short_address']+'\n')
40     n = n+1
41
42 address_num = int(input('请输入您选择位置的序号: '))
43 final_address = address_json[address_num]
44
45 restaurants_url = 'https://www.ele.me/restapi/shopping/restaurants?'
46 # 使用带有餐馆列表的那个XHR地址。
47 params = {'extras[]':'activities',
48           'geohash':final_address['geohash'],
49           'latitude':final_address['latitude'],
50           'limit':'24',
51           'longitude':final_address['longitude'],
52           'offset':'0',
53           'terminal':'web'
54         }
55 # 将参数封装, 其中geohash和经纬度, 来自前面获取到的数据。
56 restaurants_res = session.get(restaurants_url,params=params)
57 # 发起请求, 将响应的结果, 赋值给restaurants_res
58 restaurants = restaurants_res.json()
59 # 把response对象, 转为json。
60 for restaurant in restaurants:
61     # restaurants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
62     # 的所有信息。
63     print(restaurant['name'])

```


参考答案和总结

参考答案：

```
1  import requests
2  session = requests.session()
3
4  headers = {
5      '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'
6  }
7  url_1 = 'https://h5.ele.me/restapi/eus/login/mobile_send_code'
8  tel = input('请输入手机号码: ')
9  data_1 = {'captcha_hash': '',
10           'captcha_value': '',
11           'mobile': tel,
12           'scf': ''}
13
14  token = session.post(url_1, headers=headers, data=data_1).json()
15  ['validate_token']
16
17  url_2 = 'https://h5.ele.me/restapi/eus/login/login_by_mobile'
18  code = input('请输入手机验证码: ')
19  data_2 = {'mobile': tel,
20           'scf': 'ms',
21           'validate_code': code,
22           'validate_token': token}
23
24  session.post(url_2, headers=headers, data=data_2)
25
26  address_url = 'https://www.ele.me/restapi/v2/pois?'
27  place = input('请输入你的收货地址: ')
28  params =
29      {'extras[]': 'count', 'geohash': 'ws105rz9smwm', 'keyword': place, 'limit': '20',
30       'type': 'nearby'}
31  # 这里使用了深圳的geohash
32
33  address_res = requests.get(address_url, params=params)
34  address_json = address_res.json()
35
36  print('以下，是与'+place+'相关的位置信息: \n')
37  n=0
38  for address in address_json:
39      print(str(n)+' . '+address['name']+' : '+address['short_address']+'\n')
40      n = n+1
41
42  address_num = int(input('请输入您选择位置的序号: '))
43  final_address = address_json[address_num]
44
45  restaurants_url = 'https://www.ele.me/restapi/shopping/restaurants?'
46  # 使用带有餐馆列表的那个XHR地址。
```

```

44 params = {'extras[]':'activities',
45 'geohash':final_address['geohash'],
46 'latitude':final_address['latitude'],
47 'limit':'24',
48 'longitude':final_address['longitude'],
49 'offset':'0',
50 'terminal':'web'
51 }
52 # 将参数封装，其中geohash和经纬度，来自前面获取到的数据。
53 restaurants_res = session.get(restaurants_url,params=params)
54 # 发起请求，将响应的结果，赋值给restaurants_res
55 restaurants = restaurants_res.json()
56 # 把response对象，转为json。
57 for restaurant in restaurants:
58 # restaurants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。
59     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.....你会变得像我说的这样强大。

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__ 🙏(┐┌🙏)

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__ 🙏(┐┌🙏)

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】

```
1 import requests,json
2 #调用了两个模块。requests负责上传和下载数据，json负责解析。
3
4
5 word = input('你想翻译什么呀? ')
6 url='http://fanyi.youdao.com/translate?smartresult=dict&smartresult=rule'
7 #使用post需要一个链接。
8 data={'i': word,
9       'from': 'AUTO',
10       'to': 'AUTO',
11       'smartresult': 'dict',
12       'client': 'fanyideskweb',
13       'doctype': 'json',
14       'version': '2.1',
15       'keyfrom': 'fanyi.web',
16       'action': 'FY_BY_REALTIME',
17       'typoResult': 'false'}
18 #将需要post的内容，以字典的形式记录在data内。
```

```

19 r = requests.post(url,data)
20 #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。
21 answer=json.loads(r.text)
22 #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。
23 print ('翻译的结果是: '+answer['translateResult'][0][0]['tgt'])
24
25

```

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～

```

1  import requests
2  import json
3  from tkinter import Tk,Button,Entry,Label,Text,END
4
5
6  class YouDaoFanyi(object):
7      def __init__(self):
8          pass
9      def crawl(self,word):
10         url='http://fanyi.youdao.com/translate?
smartresult=dict&smartresult=rule'
11         #使用post需要一个链接
12         data={'i': word,
13              'from': 'AUTO',
14              'to': 'AUTO',
15              'smartresult': 'dict',
16              'client': 'fanyideskweb',
17              'doctype': 'json',
18              'version': '2.1',
19              'keyfrom': 'fanyi.web',
20              'action': 'FY_BY_REALTIME',
21              'typoResult': 'false'}
22         #将需要post的内容，以字典的形式记录在data内。
23         r = requests.post(url, data)

```

```

24         #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
25         answer=json.loads(r.text)
26         #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。
27         result = answer['translateResult'][0][0]['tgt']
28         return result
29
30
31
32     class Application(object):
33         def __init__(self):
34             self.window = Tk()
35             self.fanyi = YouDaoFanyi()
36
37
38
39
40             self.window.title(u'我的翻译')
41             #设置窗口大小和位置
42             self.window.geometry('310x370+500+300')
43             self.window.minsize(310,370)
44             self.window.maxsize(310,370)
45             #创建一个文本框
46             #self.entry = Entry(self.window)
47             #self.entry.place(x=10,y=10,width=200,height=25)
48             #self.entry.bind("<Key-Return>",self.submit1)
49             self.result_text1 = Text(self.window,background = 'azure')
50             # 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
51             self.result_text1.place(x = 10,y = 5,width = 285,height = 155)
52             self.result_text1.bind("<Key-Return>",self.submit1)
53
54
55             #创建一个按钮
56             #为按钮添加事件
57             self.submit_btn = Button(self.window,text=u'翻
译',command=self.submit)
58             self.submit_btn.place(x=205,y=165,width=35,height=25)
59             self.submit_btn2 = Button(self.window,text=u'清空',command =
self.clean)
60             self.submit_btn2.place(x=250,y=165,width=35,height=25)
61
62
63             #翻译结果标题
64             self.title_label = Label(self.window,text=u'翻译结果:')
65             self.title_label.place(x=10,y=165)
66             #翻译结果
67
68
69             self.result_text = Text(self.window,background = 'light cyan')
70             self.result_text.place(x = 10,y = 190,width = 285,height = 165)

```

```

71         #回车翻译
72     def submit1(self,event):
73         #从输入框获取用户输入的值
74         content = self.result_text1.get(0.0,END).strip().replace("\n"," ")
75         #把这个值传送给服务器进行翻译
76
77
78         result = self.fanyi.crawl(content)
79         #将结果显示在窗口中的文本框中
80
81
82         self.result_text.delete(0.0,END)
83         self.result_text.insert(END,result)
84
85         #print(content)
86
87     def submit(self):
88         #从输入框获取用户输入的值
89         content = self.result_text1.get(0.0,END).strip().replace("\n"," ")
90         #把这个值传送给服务器进行翻译
91
92
93         result = self.fanyi.crawl(content)
94         #将结果显示在窗口中的文本框中
95
96
97         self.result_text.delete(0.0,END)
98         self.result_text.insert(END,result)
99         print(content)
100     #清空文本域中的内容
101     def clean(self):
102         self.result_text1.delete(0.0,END)
103         self.result_text.delete(0.0,END)
104
105
106     def run(self):
107         self.window.mainloop()
108
109
110
111
112 if __name__=="__main__":
113     app = Application()
114     app.run()
115

```

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好合适～

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】

```
1 import requests
2 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     jsongdata = json.dumps(data)
28     # 发起post请求
29     response = requests.post(api, data = jsongdata)
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

```



```

54         break
55
56
57
58 #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：
59
60 # 创建对话死循环
61 while True:
62     # 输入对话内容
63     content = input("talk:")
64     robot(content)
65

```

练习-nlp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。

```
1 import requests
2 url='http://ictclas.nlpir.org/nlpir/index6/getWord2Vec.do'
3 words=input('请输入任意文本')
4 data={'content':words}
5
```

【解答】

```
1 import requests
2 #引入requests库
3 url = 'http://ictclas.nlpir.org/nlpir/index6/getWord2Vec.do'
4 #Word2vec功能对应的请求网址。
5 headers = {'Origin': 'http://ictclas.nlpir.org'
6 'Referer': 'http://ictclas.nlpir.org/'
7 'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
8 (KHTML, like Gecko) Chrome/70.0.3538.67 Safari/537.36'}
9 #封装请求头
10 words = input('请输入你想查询的词汇: ')
11 #获取需要分析的文本
12 data = {'content':words}
13 #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。
14 res = requests.post(url,data=data,headers=headers)
15 #发送post请求
16 print(res.text)
17 #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：

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

```

11 c = json.loads(b)
12 # 使用loads()函数, 将json格式的字符串b转为列表, 赋值给c。
13 print(c)
14 # 打印c。
15 print(type(c))
16 # 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:

```

1 a='郑云龙,阿云嘎,马佳,蔡程昱,高天鹤,余笛'
2 # a是一个大字符串, 可以把这个字符串切开。
3 b=a.split(',')
4 # 指定分隔符是逗号, 每碰到一个逗号, 就切一下。
5 print(b)
6 # 打印b, 结果会是一个由6个字符串组成的列表。
7 print(type(b))
8 # b是一个列表。
9

```

【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。

```

1 import requests,json
2 url = 'http://ictclas.nlpir.org/nlpir/index6/getWord2Vec.do'
3 headers = {'Origin': 'http://ictclas.nlpir.org'
4 'Referer': 'http://ictclas.nlpir.org/nlpir/'
5 'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
6 (KHTML, like Gecko) Chrome/70.0.3538.67 Safari/537.36'}
7 words = input('请输入你想查询的词汇: ')
8 data = {'content':words}
9 res = requests.post(url,data=data,headers=headers)
10 data=res.text
11 # 以上, 为上一步的代码
12
13 data1=json.loads(data)
14 # 把json数据转换为字典
15 for i in data1['w2vlist']:
16 # 遍历列表, 下面的代码请你自己完成
17
18

```

【解答】

```

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

```

```

4 'Referer': 'http://ictclas.nlpir.org/nlpir/'
5 'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
  (KHTML, like Gecko) Chrome/70.0.3538.67 Safari/537.36'}
6 words = input('请输入你想查询的词汇: ')
7 data = {'content':words}
8 res = requests.post(url,data=data,headers=headers)
9 data=res.text
10 # 以上, 为上一步的代码
11
12
13 data1=json.loads(data)
14 # 把json数据转换为字典
15 print ('和“'+words+'”相关的词汇, 至少还有: ')
16 # 打印文字
17 f=0
18 # 设置变量f
19 for i in data1['w2vlist']: # 遍历列表
20     f=f+1
21     word = i.split(',') # 切割字符串
22     print ('('+str(f)+')'+word[0]+' , 其相关度为'+word[1]) # 打印数据
23
24

```

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。

```
1  import time
2
3
4  from selenium import webdriver
5  from selenium.webdriver.chrome.options import Options
6  from selenium.webdriver.chrome.webdriver import RemoteWebDriver
7
8
9  # 获取用户输入的评论内容
10 while True:
11     comment_content = input('请输入你想要的评论的内容，按回车提交：')
12     if comment_content == '':
13         print('&' * 5, '评论内容不允许为空', '&' * 5)
14     else:
15         break
16
17
18 '''方法一：使用自己电脑上的浏览器'''
19 driver = webdriver.Chrome() # 实例化浏览器对象
20 driver.get('https://wordpress-edu-3autumn.localprod.forc.work/wp-
login.php') # 访问页面
21 time.sleep(2)
22
23
24 '''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'''
25 # chrome_options = Options() # 实例化Option对象
26 # chrome_options.add_argument('--headless') # 对浏览器的设置
27 # driver = webdriver.Chrome("http://chromedriver.python-class-
fos.svc:4444/wd/hub",
28 #                           chrome_options.to_capabilities()) # 声明浏览器
对象
29 # driver.get('https://wordpress-edu-3autumn.localprod.forc.work/wp-
login.php') # 访问页面
30
31
32 # 定位到用户名输入框，输入用户名
33 login_name = driver.find_element_by_id('user_login')
34 login_name.send_keys('spiderman')
```

```

35 time.sleep(1)
36 # 定位到密码输入框, 输入密码
37 password = driver.find_element_by_id('user_pass')
38 password.send_keys('crawler334566')
39 # 定位到登录按钮, 并点击按钮
40 submit_btn = driver.find_element_by_id('wp-submit')
41 submit_btn.click()
42 time.sleep(2)
43
44
45 # 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
46 # 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
47 article_link = driver.find_element_by_partial_link_text('同九义何汝秀')
48 article_link.click()
49
50
51 # 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
52 comment_area = driver.find_element_by_id('comment')
53 comment_area.send_keys(comment_content)
54 time.sleep(2)
55 # 定位到提交按钮, 点击该按钮提交评论
56 comment_submit = driver.find_element_by_id('submit')
57 comment_submit.click()
58
59
60 # 评论成功10秒后关闭浏览器
61 time.sleep(10)
62 driver.close()
63 print('#' * 6, '评论成功, 浏览器已关闭', '#' * 6)
64
65

```

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】

```
1 # 下面是只能在爬虫课系统中运行的答案：
2 from selenium.webdriver.chrome.webdriver import RemoteWebDriver # 从
   selenium库中调用RemoteWebDriver模块
3 from selenium.webdriver.chrome.options import Options # 从options模块中调
   用Options类
4 import time
5
6
7 chrome_options = Options() # 实例化Option对象
8 chrome_options.add_argument('--headless') # 把Chrome设置为静默模式
9 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
10
11
12 chrome_options = Options() # 实例化Option对象
13 chrome_options.add_argument('--headless') # 把Chrome设置为静默模式
14 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
15
16
```

```
17 driver.get('https://localprod.pandateacher.com/python-manuscript/hello-
18 spiderman/') # 访问页面
19
20
21 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
22 teacher.send_keys('必须是吴枫呀') # 输入文字
23 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
24 assistant.send_keys('都喜欢') # 输入文字
25 button = driver.find_element_by_class_name('sub') # 找到【提交】按钮
26 button.click() # 点击【提交】按钮
27 time.sleep(1)
28
29
30 contents = driver.find_elements_by_class_name('content') # 定位到Python之禅
   所在的标签
31 for content in contents:
32     title = content.find_element_by_tag_name('h1').text # 提取标题
33     chan = content.find_element_by_tag_name('p').text # 提取正文
34     print(title + '\n' + chan + '\n') # 打印标题与正文
35 driver.close()
36
37
38
39
40 # 下面是只能在你的本地运行的答案:
41 from selenium import webdriver # 从selenium库中调用webdriver模块
42 import time
43
44
45 driver = webdriver.Chrome() # 声明浏览器对象
46 driver.get('https://localprod.pandateacher.com/python-manuscript/hello-
   spiderman/') # 访问页面
47 time.sleep(2) # 暂停两秒, 等待浏览器缓冲
48
49
50 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
51 teacher.send_keys('必须是吴枫呀') # 输入文字
52 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
53 assistant.send_keys('都喜欢') # 输入文字
54 button = driver.find_element_by_class_name('sub') # 找到【提交】按钮
55 button.click() # 点击【提交】按钮
56 time.sleep(1)
57
58
59 contents = driver.find_elements_by_class_name('content') # 定位到Python之禅
   所在的标签
```



```

60 for content in contents:
61     title = content.find_element_by_tag_name('h1').text # 提取标题
62     chan = content.find_element_by_tag_name('p').text # 提取正文
63     print(title + '\n' + chan + '\n') # 打印标题与正文
64 driver.close()
65
66

```

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。

```

1 # 下面是只能在爬虫课系统中运行的答案:
2 from selenium.webdriver.chrome.webdriver import RemoteWebDriver # 从
   selenium库中调用RemoteWebDriver模块
3 from selenium.webdriver.chrome.options import Options # 从options模块中调用
   Options类
4 from bs4 import BeautifulSoup
5 import time
6
7
8 chrome_options = Options() # 实例化Option对象
9 chrome_options.add_argument('--headless') # 把Chrome设置为静默模式
10 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
11
12
13 driver.get('https://localprod.pandateacher.com/python-manuscript/hello-
   spiderman/') # 访问页面
14 time.sleep(2) # 暂停两秒，等待浏览器缓冲
15
16
17 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
18 teacher.send_keys('必须是吴枫呀') # 输入文字

```

```
19 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
20 assistant.send_keys('都喜欢') # 输入文字
21 button = driver.find_element_by_class_name('sub') # 定位到【提交】按钮
22 button.click() # 点击【提交】按钮
23 time.sleep(1) # 等待一秒
24
25
26 pageSource = driver.page_source # 获取页面信息
27 soup = BeautifulSoup(pageSource, 'html.parser') # 使用bs解析网页
28 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
29 for content in contents: # 遍历列表
30     title = content.find('h1').text # 提取标题
31     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
32     print(title + chan + '\n') # 打印Python之禅的标题与正文
33 driver.close()
34
35
36
37
38 # 下面是只能在你的本地运行的答案：
39 from selenium import webdriver # 从selenium库总调用webdriver模块
40 import time
41 from bs4 import BeautifulSoup
42
43
44 driver = webdriver.Chrome() # 声明浏览器对象
45 driver.get('https://localprod.pandateacher.com/python-manuscript/hello-
    spiderman/') # 访问页面
46 time.sleep(2) # 暂停两秒，等待浏览器缓冲
47
48
49 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
50 teacher.send_keys('必须是吴枫呀') # 输入文字
51 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
52 assistant.send_keys('都喜欢') # 输入文字
53 button = driver.find_element_by_class_name('sub') # 定位到【提交】按钮
54 button.click() # 点击【提交】按钮
55 time.sleep(1) # 等待一秒
56
57
58 pageSource = driver.page_source # 获取页面信息
59 soup = BeautifulSoup(pageSource, 'html.parser') # 使用bs解析网页
60 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
61 for content in contents: # 遍历列表
62     title = content.find('h1').text # 提取标题
```

```
63     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
64     print(title + chan + '\n') # 打印Python之禅的标题与正文
65     driver.close()
66
67
```

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：

```
1  import requests
2  from bs4 import BeautifulSoup
3
4
5  res_foods = requests.get('http://www.xiachufang.com/explore/')
6  bs_foods = BeautifulSoup(res_foods.text, 'html.parser')
7  list_foods = bs_foods.find_all('div', class_='info pure-u')
8
```

```

9
10 list_all = []
11
12
13 for food in list_foods:
14     tag_a = food.find('a')
15     name = tag_a.text[17:-13]
16     URL = 'http://www.xiachufang.com'+tag_a['href']
17     tag_p = food.find('p',class_='ing ellipsis')
18     ingredients = tag_p.text[1:-1]
19     list_all.append([name,URL,ingredients])
20 print(list_all)
21
22

```

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）

```

1  import smtplib
2  from email.mime.text import MIMEText
3  from email.header import Header
4  #引入smtplib、MIMEText和Header
5
6  mailhost='smtp.qq.com'
7  #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
8  qqmail = smtplib.SMTP()
9  #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
10 qqmail.connect(mailhost,25)
11 #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。
12 #以上，皆为连接服务器。
13
14 account = input('请输入你的邮箱: ')
15 #获取邮箱账号，为字符串格式
16 password = input('请输入你的密码: ')
17 #获取邮箱密码，为字符串格式
18 qqmail.login(account,password)
19 #登录邮箱，第一个参数为邮箱账号，第二个参数为邮箱密码
20 #以上，皆为登录邮箱。
21
22 receiver=input('请输入收件人的邮箱: ')
23 #获取收件人的邮箱。
24
25 content=input('请输入邮件正文: ')
26 #输入你的邮件正文，为字符串格式
27 message = MIMEText(content, 'plain', 'utf-8')
28 #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
29 subject = input('请输入你的邮件主题: ')
30 #输入你的邮件主题，为字符串格式
31 message['Subject'] = Header(subject, 'utf-8')

```

```

32 #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']。
33 #以上，为填写主题和正文。
34
35 try:
36     qqmail.sendmail(account, receiver, message.as_string())
37     print ('邮件发送成功')
38 except:
39     print ('邮件发送失败')
40 qqmail.quit()
41 #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。

```

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

```

【解答】

老师提供的参考答案是这样的：

```

1  import requests
2  import smtplib
3  import schedule
4  import time
5  from bs4 import BeautifulSoup
6  from email.mime.text import MIMEText
7  from email.header import Header
8
9
10 account = input('请输入你的邮箱: ')
11 password = input('请输入你的密码: ')
12 receiver = input('请输入收件人的邮箱: ')
13
14 def recipe_spider():
15     res_foods = requests.get('http://www.xiachufang.com/explore/')

```

```

16     bs_foods = BeautifulSoup(res_foods.text,'html.parser')
17     list_foods = bs_foods.find_all('div',class_='info pure-u')
18     list_all = ''
19     num=0
20     for food in list_foods:
21         num=num+1
22         tag_a = food.find('a')
23         name = tag_a.text.strip()
24         url = 'http://www.xiachufang.com'+tag_a['href']
25         tag_p = food.find('p',class_='ing ellipsis')
26         ingredients = tag_p.text.strip()
27         food_info = ''
28         序号: %s
29         菜名: %s
30         链接: %s
31         原料: %s
32         '%(num,name,url,ingredients)
33         list_all=list_all+food_info
34     return(list_all)
35
36 def send_email(list_all):
37     global account,password,receiver
38     mailhost='smtp.qq.com'
39     qqmail = smtplib.SMTP()
40     qqmail.connect(mailhost,25)
41     qqmail.login(account,password)
42     content= '亲爱的, 本周的热门菜谱如下'+list_all
43     message = MIMEText(content, 'plain', 'utf-8')
44     subject = '周末吃个啥'
45     message['Subject'] = Header(subject, 'utf-8')
46     try:
47         qqmail.sendmail(account, receiver, message.as_string())
48         print ('邮件发送成功')
49     except:
50         print ('邮件发送失败')
51     qqmail.quit()
52
53 def job():
54     print('开始一次任务')
55     list_all = recipe_spider()
56     send_email(list_all)
57     print('任务完成')
58
59 schedule.every().friday.at("18:00").do(job)#部署每周三的13: 15执行函数的任务
60 while True:
61     schedule.run_pending()
62     time.sleep(1)
63

```

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）

```
1 #这是爬取豆瓣电影Top250，并存储为本地csv的代码
2 import requests, random, csv
3 from bs4 import BeautifulSoup
4 csv_file=open('movieTop.csv', 'w', newline='',encoding='utf-8')
5 writer = csv.writer(csv_file)
6
7 for x in range(10):
8     url = 'https://movie.douban.com/top250?start=' + str(x*25) +
9         '&filter='
10    res = requests.get(url)
11    bs = BeautifulSoup(res.text, 'html.parser')
12    bs = bs.find('ol', class_="grid_view")
13    for titles in bs.find_all('li'):
14        title = titles.find('span', class_="title").text
15        list1 = [title]
16        writer.writerow(list1)
17 csv_file.close()
```

```

1 # 这是爬电影的下载链接的代码
2 import requests
3 from bs4 import BeautifulSoup
4 from urllib.request import quote
5
6 movie=input('你想看什么电影? ')
7 gbkmovie = movie.encode('gbk')
8 urlsearch = 'http://s.ygdy8.com/plus/so.php?
   typeid=1&keyword='+quote(gbkmovie)
9 res = requests.get(urlsearch)
10 res.encoding='gbk'
11 soup_movie = BeautifulSoup(res.text,'html.parser')
12 urlpart=soup_movie.find(class_="co_content8").find_all('table')
13 if urlpart:
14     urlpart=urlpart[0].find('a')['href']
15     urlmovie='https://www.ygdy8.com/'+urlpart
16     res1=requests.get(urlmovie)
17     res1.encoding='gbk'
18     soup_movie1=BeautifulSoup(res1.text,'html.parser')
19
20     urldownload=soup_movie1.find('div',id="Zoom").find('span').find('table').f
   ind('a')['href']
21     print(urldownload)
22 else:
23     print('没有'+movie+'的连接')

```

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】

```

1 import requests, csv, random
2 from bs4 import BeautifulSoup
3 from urllib.request import quote
4 # 以上，为引入相应的库。
5
6
7 csv_file=open('movieTop.csv', 'w', newline='', encoding='utf-8')
8 writer = csv.writer(csv_file)
9 for x in range(10):
10     url = 'https://movie.douban.com/top250?start=' + str(x*25) +
   '&filter='
11     res = requests.get(url)
12     bs = BeautifulSoup(res.text, 'html.parser')
13     bs = bs.find('ol', class_="grid_view")

```



```

14     for titles in bs.find_all('li'):
15         title = titles.find('span', class_="title").text
16         list1 = [title]
17         writer.writerow(list1)
18 csv_file.close()
19 # 以上，为爬取豆瓣电影Top250的榜单，并存储为本地的csv文件。
20
21 movielist=[]
22 csv_file=open('movieTop.csv','r',newline='',encoding='utf-8')
23 reader=csv.reader(csv_file)
24 for row in reader:
25     movielist.append(row[0])
26 # 以上，为读取豆瓣电影Top250榜单的csv文件，并写入列表movielist中。
27 three_movies=random.sample(movielist,3)
28 # 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。
29 for movie in three_movies:
30     # 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。
31     gbkmovie = movie.encode('gbk')
32     urlsearch = 'http://s.ygdy8.com/plus/so.php?
typeid=1&keyword='+quote(gbkmovie)
33     res = requests.get(urlsearch)
34     res.encoding='gbk'
35     soup_movie = BeautifulSoup(res.text,'html.parser')
36     urlpart=soup_movie.find(class_="co_content8").find_all('table')
37     if urlpart:
38         urlpart=urlpart[0].find('a')['href']
39         urlmovie='https://www.ygdy8.com/'+urlpart
40         res1=requests.get(urlmovie)
41         res1.encoding='gbk'
42         soup_movie1=BeautifulSoup(res1.text,'html.parser')
43
44         urldownload=soup_movie1.find('div',id="Zoom").find('span').find('table').f
ind('a')['href']
45         content=movie+'\n'+urldownload
46         print(content)
47     else:
48         content='没有'+movie+'的下载链接'
49         print(content)
50

```

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）

```

1 import smtplib
2 from email.mime.text import MIMEText
3 from email.header import Header

```

```

4 #引入smtpLib、MIMETex和Header
5
6 mailhost='smtp.qq.com'
7 #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
8 qqmail = smtpLib.SMTP()
9 #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
10 qqmail.connect(mailhost,25)
11 #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。
12 #以上，皆为连接服务器。
13
14 account = input('请输入你的邮箱: ')
15 #获取邮箱账号，为字符串格式
16 password = input('请输入你的密码: ')
17 #获取邮箱密码，为字符串格式
18 qqmail.login(account,password)
19 #登录邮箱，第一个参数为邮箱账号，第二个参数为邮箱密码
20 #以上，皆为登录邮箱。
21
22 receiver=input('请输入收件人的邮箱: ')
23 #获取收件人的邮箱。
24
25 #content为上面的电影链接
26 #输入你的邮件正文，为字符串格式
27 message = MIMEText(content, 'plain', 'utf-8')
28 #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
29 subject = '电影链接'
30 #输入你的邮件主题，为字符串格式
31 message['Subject'] = Header(subject, 'utf-8')
32 #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']。
33 #以上，为填写主题和正文。
34
35 try:
36     qqmail.sendmail(account, receiver, message.as_string())
37     print ('邮件发送成功')
38 except:
39     print ('邮件发送失败')
40 qqmail.quit()
41 #以上为发送邮件和退出邮箱
42
43 #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。
44 import schedule
45 import time
46 #引入schedule和time
47 def job():
48     print("I'm working...")
49 #定义一个叫job的函数，函数的功能是打印'I'm working...'
50
51 schedule.every(10).minutes.do(job)      #部署每10分钟执行一次job()函数的任务
52 schedule.every().hour.do(job)          #部署每x小时执行一次job()函数的任务
53 schedule.every().day.at("10:30").do(job)#部署每天的10:30执行job()函数任务

```

```

54 schedule.every().monday.do(job)           #部署每个星期一执行job()函数的任务
55 schedule.every().wednesday.at("13:15").do(job)
56 #部署每周三的13: 15执行函数的任务
57
58 while True:
59     schedule.run_pending()
60     time.sleep(1)
61
62

```

【解答】

老师提供的参考答案是这样的：

```

1  import requests, csv, random, smtplib, schedule, time
2  from bs4 import BeautifulSoup
3  from urllib.request import quote
4  from email.mime.text import MIMEText
5  from email.header import Header
6
7
8  def get_movielist():
9      csv_file=open('movieTop.csv', 'w', newline='', encoding='utf-8')
10     writer = csv.writer(csv_file)
11     for x in range(10):
12         url = 'https://movie.douban.com/top250?start=' + str(x*25) +
13         '&filter='
14         res = requests.get(url)
15         bs = BeautifulSoup(res.text, 'html.parser')
16         bs = bs.find('ol', class_="grid_view")
17         for titles in bs.find_all('li'):
18             title = titles.find('span', class_="title").text
19             list1 = [title]
20             writer.writerow(list1)
21         csv_file.close()
22
23 def get_randommovie():
24     movielist=[]
25     csv_file=open('movieTop.csv', 'r', newline='', encoding='utf-8')
26     reader=csv.reader(csv_file)
27     for row in reader:
28         movielist.append(row[0])
29     three_movies=random.sample(movielist,3)
30     contents=''
31     for movie in three_movies:
32         gbkmovie = movie.encode('gbk')
33         urlsearch = 'http://s.ygdy8.com/plus/so.php?
34         typeid=1&keyword='+quote(gbkmovie)
35         res = requests.get(urlsearch)
36         res.encoding='gbk'
37         soup_movie = BeautifulSoup(res.text, 'html.parser')

```

```

36         urlpart=soup_movie.find(class_="co_content8").find_all('table')
37         if urlpart:
38             urlpart=urlpart[0].find('a')['href']
39             urlmovie='https://www.ygdy8.com/'+urlpart
40             res1=requests.get(urlmovie)
41             res1.encoding='gbk'
42             soup_movie1=BeautifulSoup(res1.text,'html.parser')
43
44         urldownload=soup_movie1.find('div',id="Zoom").find('span').find('table').f
45         ind('a')['href']
46         content=movie+'\n'+urldownload+'\n\n'
47         print(content)
48         contents=contents+content
49     else:
50         content='没有'+movie+'的下载链接'
51         print(content)
52     return contents
53
54 def send_movielink(contents):
55     mailhost='smtp.qq.com'
56     qqmail = smtplib.SMTP()
57     qqmail.connect(mailhost,25)
58     account = 'xxxxxxxxx@qq.com' # 因为是自己发给自己，所以邮箱账号、密码都可以
59     提前设置好，当然，也可以发给别人啦
60     password = 'xxxxxxxxxxxxxx' # 因为是自己发给自己，所以邮箱账号、密码都可以
61     提前设置好，当然，也可以发给别人啦。
62     qqmail.login(account,password)
63     receiver='xxxxxxxxx@qq.com' # 因为是自己发给自己，所以邮箱账号、密码都可以
64     提前设置好，当然，也可以发给别人啦。
65     message = MIMEText(contents, 'plain', 'utf-8')
66     subject = '电影链接'
67     message['Subject'] = Header(subject, 'utf-8')
68     try:
69         qqmail.sendmail(account, receiver, message.as_string())
70         print ('邮件发送成功')
71     except:
72         print ('邮件发送失败')
73     qqmail.quit()
74
75 def job():
76     get_movielist()
77     contents=get_randommovie()
78     send_movielink(contents)
79
80 schedule.every().friday.at("18:00").do(job)
81 while True:
82     schedule.run_pending()
83     time.sleep(1)

```

