

课后练习1：自制动态二维码

- 题目解析：
 - 注意代码中参数的名称不能改变，picture参数和save_name两个参数可以写为绝对路径。
- 参考代码：

```
1 # 先导入模块
2 from MyQR import myqr
3 myqr.run(
4     words='http://weixin.qq.com/r/kzlje9TEE4lsrzAY92yB',
5     # 扫描二维码后，显示的内容，或是跳转的链接
6     version=5,    # 设置容错率
7     level='H',    # 控制纠错水平，范围是L、M、Q、H，从左到右依次升高
8     picture='she-3.gif',  # 图片所在目录，可以是动图
9     colorized=True,   # 黑白(False)还是彩色(True)
10    contrast=1.0,    # 用以调节图片的对比度，1.0 表示原始图片。默认为1.0。
11    brightness=1.0,   # 用来调节图片的亮度，用法同上。
12    save_name='Python.gif',  # 控制输出文件名，格式可以是 .jpg, .png , .bmp , .gif
13 )
14 # save_dir= r'C:\Users\user\Desktop\test'# '指定二维码生成的地址，填写路径就行'
```

课后练习2：绝对值的多种求法

- 题目解析：
 - 方法一：使用条件判断语句，负数时候将变量改为它的相反数，正数时候不变。
 - 方法二：使用内置的abs()函数计算。
 - 方法三：使用math库的fabs()函数计算。
- 参考代码：

```
1 import math
2 # 方法1：条件判断
3 def abs_value1():
4     a = float(input('1.请输入一个数字: '))
5     if a >= 0:
6         a = a
7     else:
8         a = -a
9     print('绝对值为: %f' % a)
10
11 # 方法2：内置函数 abs()
```

```

12 def abs_value2():
13     a = float(input('2.请输入一个数字: '))
14     a = abs(a)
15     print('绝对值为: %f' % a)
16
17 # 方法3: 内置模块 math
18 def abs_value3():
19     a = float(input('3.请输入一个数字: '))
20     a = math.fabs(a)
21     print('绝对值为: %f' % a)
22
23 # 运行函数, 查验一下。
24 abs_value1()
25 abs_value2()
26 abs_value3()

```

第18关 编程思维：产品思维

课后练习1：拯救选择困难症

- 题目解析:
 - ‘不知道吃什么有两种原因’，一是完全不知道要吃什么，二是在几家店之间犹豫。
 - 可先使用if...elif...else语句，判断出是属于上述的那种情况，并根据用户选择的情况，进行相关的食物推送。
 - 定义菜名选择函数，无论用户选择那种情况，都需要对推送的菜名进行选择，所以此函数可以多次复用。
- 参考代码：

```

1 import random
2
3 # 将需要用到的表格和变量放在开头
4 list_food = ['KFC', '蒸菜馆', '楼下快餐店', '桂林米粉', '东北饺子', '金牌猪脚饭', '三及第汤饭']
# 备选菜单，可自定义。
5 list_choice = []
6
7 # 菜名选择
8 def choose(list):
9     while True:
10         food = random.choice(list)
11         judgement = input('去吃【%s】好不好啊？同意的话输入y，不想吃直接回车即可。' % (food))

```

```

12         if judgement == 'y':
13             print('去吃【%s】！就这么愉快地决定啦!' % (food))
14             break
15
16 # 判断环节
17 reason = int(input('你不知道吃的原因是：1.完全不知道吃什么；2.在几家店之间徘徊（请输入1或
18 2）：'))
19 if reason == 1:
20     choose(list_food)
21 elif reason == 2:
22     add = True
23     while add:
24         choice = input('请输入让你犹豫的店名（注：一家一家输，完成后输入y）：')
25         if choice != 'y': # 这个判断语句，是为了不将 y 也添加到菜单里。
26             list_choice.append(choice)
27         if choice == 'y':
28             add = False
29     choose(list_choice)
30 else:
31     print('抱歉，目前还不支持第三种情况—不过，你可以加代码哦。')

```

课后练习2：模拟广告牌

- 题目解析：

- 定义一个变量，用于存放关于广告牌的显示内容。
- os.system()函数，对屏幕内容清空，再使用字符串拼接，将广告词的第一个元素与剩下的所有元素进行拼接，成为新的广告词。（例如广告词：‘你好啊， python!‘，拼接后尾：‘!你好啊， python’）。
- 根据规律，每一次的清屏，广告词都会进行变化，只要设计清屏的时间够快，躲过人类的反应时间，那么就可以形成滚动的效果。
- 最后添加while循环来不停的滚动广告词。

- 参考代码：

```

1 import os, time
2
3 def main():
4     content = ' 风变编程，陪你一起学Python ' # 广告词可自定义。
5     while True:
6         os.system('clear') # windows系统将'clear'改为'cls'。
7         print(content)

```

```
8         content = content[1:] + content[0]
9         time.sleep(2)
10
11 if __name__ == '__main__': # 类里面学到的检测方法，在函数中其实也可以用。
12     main()
```

爬虫精进课程

第0关 认识爬虫

课后练习1：文章下载

- 题目解析：
 - 调用requests库，使用requests.get()获取文件，返回Response对象。
 - 由于该链接是一个文本，需要把Response对象转化为字符串输出，所以使用text属性返回数据。
- 参考代码：

```
1 import requests
2
3 destnation_url = 'https://localprod.pandateacher.com/python-manuscript/crawler-
4 html/exercise/HTTP%E5%93%8D%E5%BA%94%E7%8A%B6%E6%80%81%E7%A0%81.md'
5
6 res = requests.get (destnation_url)
7 print(res.status_code) # 查看响应码
8 article=res.text      # 把Response对象的内容以字符串的形式返回
9 print(article)
```

课后练习2：图像下载

- 题目解析：
 - 调用requests库，使用requests.get('URL')获取文件，返回Response对象。
 - 由于该链接是一个图片，需要把Response对象转化为二进制输出，所以使用content属性返回数据。
 - 返回的二进制数据通过打开文件>写入文件>关闭文件，将图片文件储存到本地。
- 参考代码：

```
1 import requests
2 res = requests.get('https://res.pandateacher.com/2019-01-12-15-29-33.png')
```

```
3 # 发出请求，并把返回的结果放在变量res中
4 pic=res.content      # 把Reponse对象的内容以二进制数据的形式返回
5
6 photo = open('spider.jpg','wb') # 新建了一个文件ppt.jpg，可以添加文件路径存放到指定位置
7 photo.write(pic)    # 写入pic的二进制内容
8 photo.close()       # 关闭文件
```

课后练习3：音频下载

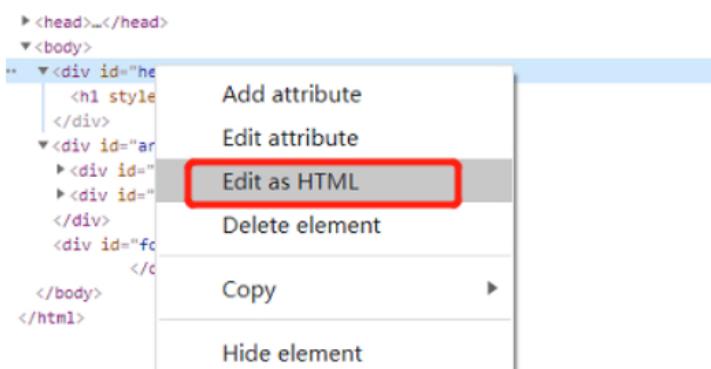
- 题目解析：
 - 音频的下载原理，同图片。
 - 参考代码：

```
1 import requests  
2  
3 res=requests.get('https://static.pandateacher.com/Over%20The%20Rainbow.mp3')  
4 # 发出请求，并把返回的结果放在变量res中  
5 mp3=res.content # 把Reponse对象的内容以二进制数据的形式返回  
6  
7 music = open('rainbow.mp3', 'wb') # 新建了一个文件ppt.jpg，可以添加文件路径存放到指定位置  
8 music.write(mp3) # 获取pic的二进制内容  
9 music.close() # 关闭文件
```

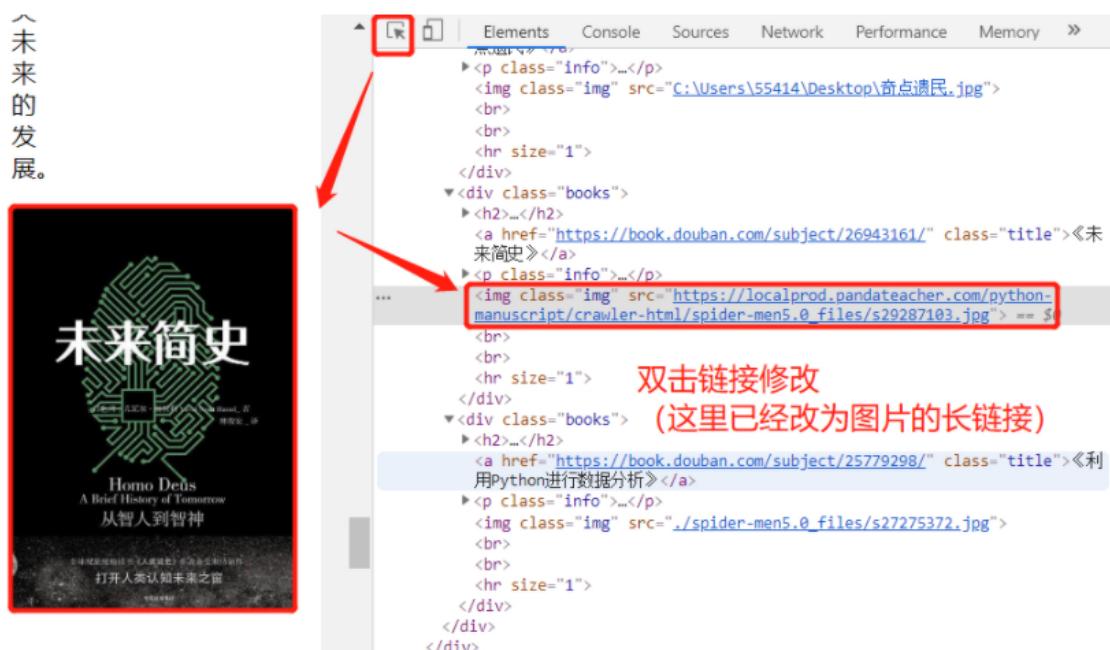
第1关 HTML基础

课后练习：我的书苑我做主

- 题目解析：
 - 右键element代码>Edit as HTML，会出现一个编辑框，可以尽情修改HTML代码。



- 注意：直接下载网页代码，保存到.html文件中，直接用浏览器打开时，会出现图片显示不了。此时修改一下图片的链接即可，可以改为图片的网页链接，也可以执行下载，使用本地路径的绝对路径。



- 参考代码：

- 参考修改链接：<https://localprod.pandateacher.com/python-manuscript/crawler-html/exercise/01-01-test.html>

⚠ 注意，修改图片是直接将自己要替换的图片的链接在检查里面双击替换。

第2关 BeautifulSoup

课后练习1：博客爬虫

- 题目解析：
 - 第1步：获取数据，使用`requests.get()`
 - 第2步：解析数据，使用`BeautifulSoup(网页源代码的字符串格式,'html.parser')`
 - 第3步：提取数据，使用`find_all()`，提取之后返回一个Tag集合，需要遍历提取每一个值，然后使`text`属性提取字符串。
 - 寻找数据所在标签，可以借用“小箭头”定位；
 - 定位到数据之后，查看数据标签能否被精准定位到，可以调出搜索框搜索，一般有2种判断方法，一种是标签唯一，一种是数据的数量和标签的数量刚好对应；
 - 这里的评论数量可能只有一条也可以有多条，可以根据数据数量和标签数量判断：像以下这一条评论，最小的一个标签是`<p>`但是搜索的时候发现，源代码中有很多`<p>`标签，不能准确定位，然后就往上一级去找，找到"comment-content"属性的时候，发现一共有7个，刚好和左边的评论个数（7个）一致！（评

论会实时更新，以最新的数量为准)

1、定位内容

2、找标签

- 找到标签数和数据数量一致时，可以尝试爬虫，该作业可以准确爬取到。

- 参考代码：

```
1 import requests
2 from bs4 import BeautifulSoup
3 # 获取数据
4 destination_url = 'https://wordpress-edu-3autumn.localprod.oc.forchange.cn/all-about-
the-future_04/'
5 destination = requests.get(destination_url)           # 返回一个response对象，赋值给
destination
6 # 解析数据
7 soup = BeautifulSoup(destination.text, 'html.parser')      # 把网页解析为BeautifulSoup
对象
8 # 提取数据
9 comments = soup.find_all('div', class_= 'comment-content') # 通过匹配属性提取出我们想要的元
素
10 for comment in comments: # 遍历列表，取出列表中的每一个值
11     print(comment.text) # 打印评论的文本
```

课后练习2：书店寻宝

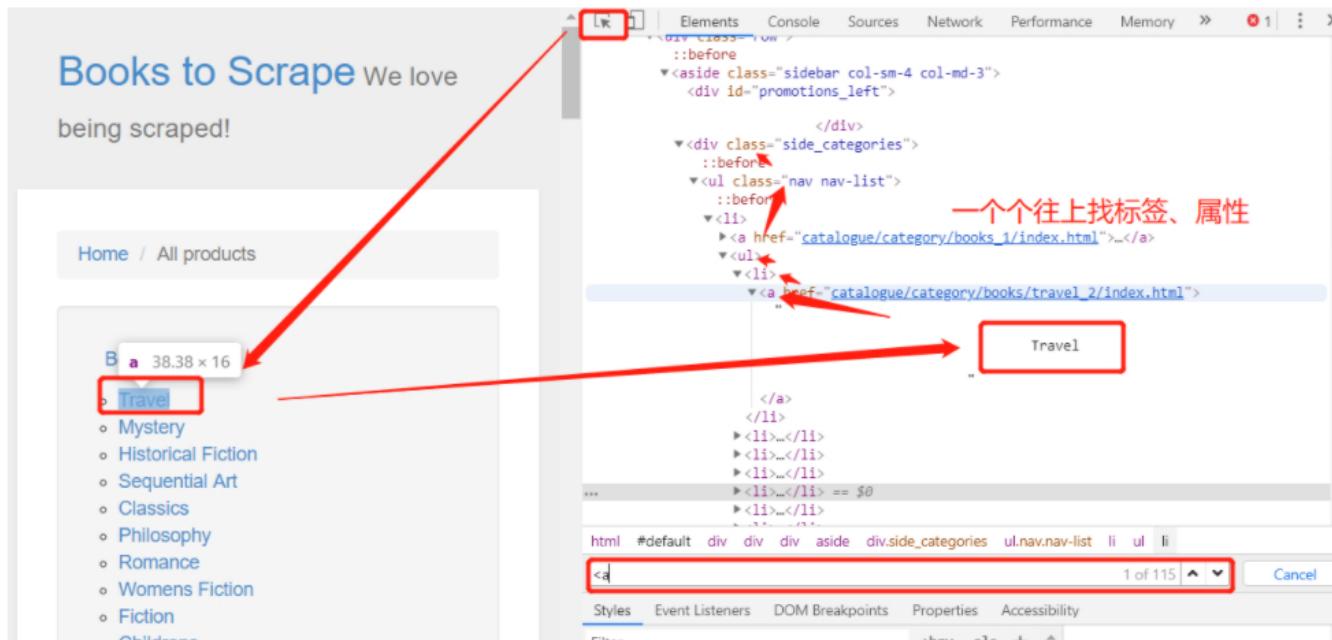
第一个小练习

- 题目解析：

- 第1步：获取数据，使用`requests.get()`。
- 第2步：解析数据，使用`BeautifulSoup(网页源代码的字符串格式, 'html.parser')`。
- 第3步：提取数据，使用`find_all()`和`find()`交替使用，提取数据Tag对象，最终根据数据需要呈现的内容（字符

串或属性值) 选择text属性或["属性名"]提取。

- 定位数据所在标签 (方法同爬虫第2关课后练习1定位数据方法) 。
- 该练习比上个练习加了一个难度, 就是数据所在的标签并不能精准定位到所要爬取的数据, 因为标签的数量远大于所要爬取数据的数量, 此时需要一级一级往上找能够精准定位到的父级标签。
- 从依次通过->->到<ul class="nav nav-list">才能准确定位到。



- 注意一点: 如果属性值有多个, 会使用空格隔开, 取其中一个即可 (如: <ul class="nav nav-list">中有两个属性值nav和nav-list)

- 参考代码:

```
1 import requests
2 from bs4 import BeautifulSoup
3
4 res_bookstore = requests.get('http://books.toscrape.com/')
5 bs_bookstore = BeautifulSoup(res_bookstore.text, 'html.parser')
6 list_kind = bs_bookstore.find('ul', class_='nav').find('ul').find_all('li') # 需要提取好几层
7
8 for tag_kind in list_kind:
9     tag_name = tag_kind.find('a').text      # 提取字符
10    print(tag_name.strip())               # 去除字符串首尾空格、空行等
```

第二个小练习

- 题目解析:

- 该练习在上一个练习上也递增了难度——爬取多条数据。
- 先定位数据所在标签。
- 爬取多条数据, 需要找到每一条数据的共同父级标签; 需要一级一级往上寻找数据的最小父级标签 (一开始

不熟悉可以把各级的父级标签写下来，然后对照每一个数据的父级，找到对应的共同父级，一般10级以内可以找到）。

- 找到共同父级标签之后，可以调出element的搜索框，先在网页中查找一遍判断是否可以准确定位到，也可以在编译器敲代码进行爬取测试；如果不行，再往上找父级标签，然后再网页查找或爬取测试，直到能够准确定位到。
- 确定可以定位到的共同父级标签之后，需要再查看小标签能否精准定位，就是查看在该共同标签内能不能精准定位到我们需要的数据的位置，也可以通过网页查找或爬取测试。
- 注意一点：取星级是取class属性的属性值，需要使用['class']进行取值。

- 参考代码：

```
1 import requests
2 from bs4 import BeautifulSoup
3 res_bookstore =
4     requests.get('http://books.toscrape.com/catalogue/category/books/travel_2/index.html')
5
6 bs_bookstore = BeautifulSoup(res_bookstore.text, 'html.parser')
7 list_books = bs_bookstore.find_all(class_='product_pod')
8 for tag_books in list_books:
9     tag_name = tag_books.find('h3').find('a') # 定位到该a标签需要先提取h3
10    list_star = tag_books.find('p', class_="star-rating")
11    tag_price = tag_books.find('p', class_="price_color")
12
13    print('书名:', tag_name['title']) # 这里用到了tag['属性名']提取属性值
14    print('评分:', list_star['class'][1]) # 提取class属性的值，返回列表['star-rating',
15    'Two'], 然后取第1个索引值
16    print('价格:', tag_price.text + '\n' + '-----' + '\n') # 打印，此处换行符可以去掉，是为了
17    让数据更加清晰地分隔开
```

课后练习3：博客文章

- 题目解析：
 - 做该练习时，同学们可以参考课后练习2中的搜索过程进行探索；
 - 该练习和上一个练习也有不同之处：爬取链接。
 - 爬取链接的取法同属性值的取法，不过取出链接之后，由于链接是短链接（只有后面一段，没有域名等，相当于相对路径），所以需要在element上查看完整链接，把鼠标放在链接上即可出现完整链接，也可以点击链接调整到新页面直接复制地址栏的地址。

- 参考代码：

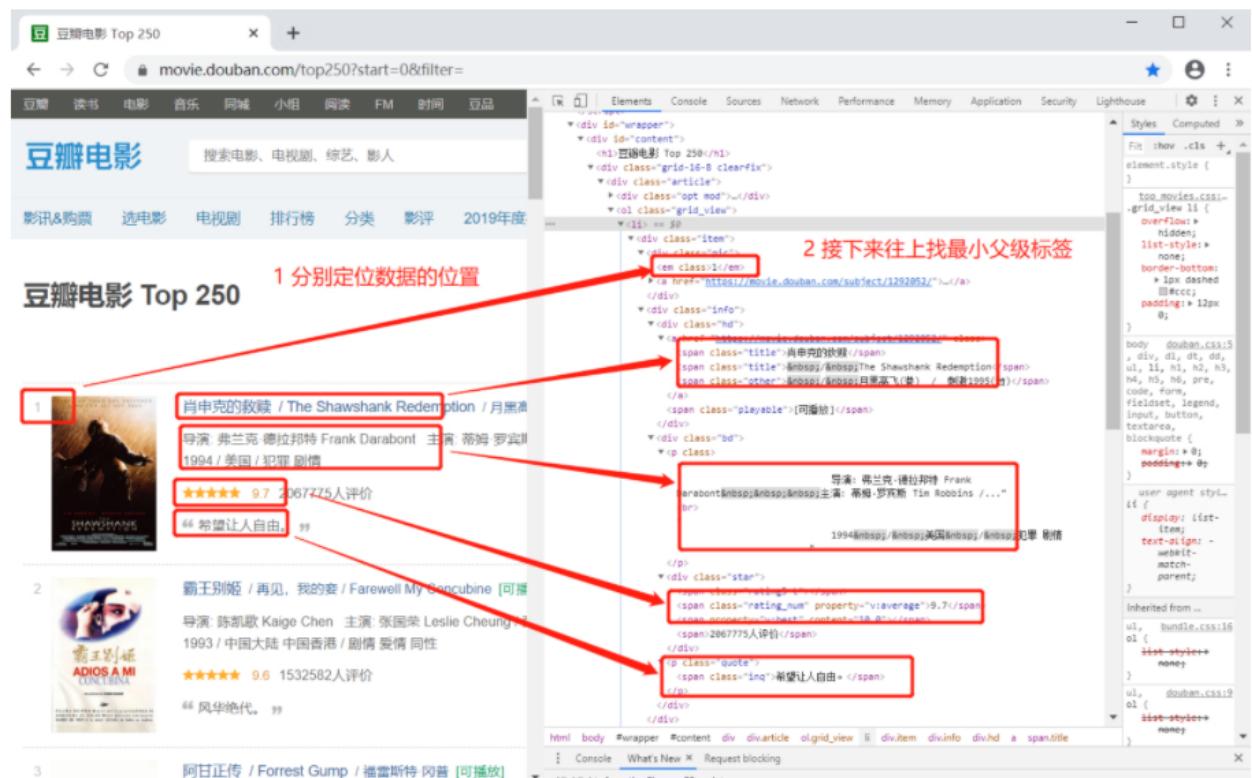
```
1 import requests
2 from bs4 import BeautifulSoup
3
4 url_destination = 'https://spidermen.cn/'
5 res_destination = requests.get(url_destination)
6 print(res_destination.status_code) # 打印响应码
7
8 bs_articles = BeautifulSoup(res_destination.text, 'html.parser')
9 list_articles = bs_articles.find_all('header', class_ = "entry-header") # 首先找到每篇文章所在的相同的元素
10 for tag_article in list_articles: # 遍历列表
11     tag_title = tag_article.find('h2', class_ = "entry-title")          # 找文章标题
12     tag_url = tag_article.find('a', rel = "bookmark")      # 找文章链接
13     tag_date = tag_article.find('time', class_ = "entry-date published") # 找文章发布时间
14     print(tag_title.text, '发布于：', tag_date.text)        # 打印文章标题与发布时间
15     print(tag_url['href'])           # 换行打印文章链接，需要使用属性名提取属性值
```

第3关 BeautifulSoup实践

课后练习：豆瓣爬虫

- 题目解析：
 - 经典三步骤：获取数据、解析数据、提取数据，前面两步同学们应该较为熟悉了，难点就是在第三步。
 - 该练习有一定的难度，不过经历过下厨房之后，应该不会觉得太难。还是使用定位数据标签>查找共同父级标签>判断标签是否可行的思路进行验证数据标签和父级标签的可行性。具体操作也可以参考爬虫第2关的课

后练习讲解，这里大概讲一下操作路径：定位到数据标签之后>往上找父级标签，最终聚焦到最小父级标签<div class="item">。



- 相比下厨房，加了一个要求：循环获取多个页面的数据。这里就需要对多个页面的网址进行查看，可以把网址放到一起，进行对比，寻找规律。经过一番对比，可以很明了的发现，除了第一页，其他的页面只有start=后面的参数不同，这些参数都是25的倍数，除以25之后，便是2,3,4,5,6.....所以可以确定第2页之后，第n页的网址中start=后面的参数是 $25*n$ ；再根据该规律，应用到第1页进行测试，结果发现可行，故十页中的网址都可以通过n的值来确定，而n的值可以使用for循环赋值，即for n in range(10)，再对网址做拼接即可。
- 注意3点：
 - 最小父级标签比较方便往下定位数据，如果在最小父级标签的上级标签中也可以定位到的话，也可以没有唯一答案，比如：使用bs.find('div', class_="article").find_all("li")和bs = bs.find('ol', class_="grid_view").find_all("li")也可以定位到所需要的数据，感兴趣可以尝试一下。
 - 如果不加headers（请求头）爬取不到数据，可以加上试试，详见以下参考代码第2行，注意在第8行处需要传入headers；
 - 由于推荐语部分数据缺失，需要加一个判断条件把没有推荐语的数据进行处理，想见以下参考代码第17~21行，当然也可以使用功能try...except...进行判断（参考以下代码第23~27行）。

- 参考代码：

```
1 import requests, bs4
2 headers = {'user-agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/71.0.3578.98 Safari/537.36'}
3 # 增加请求头，放在get()前面即可
```

```

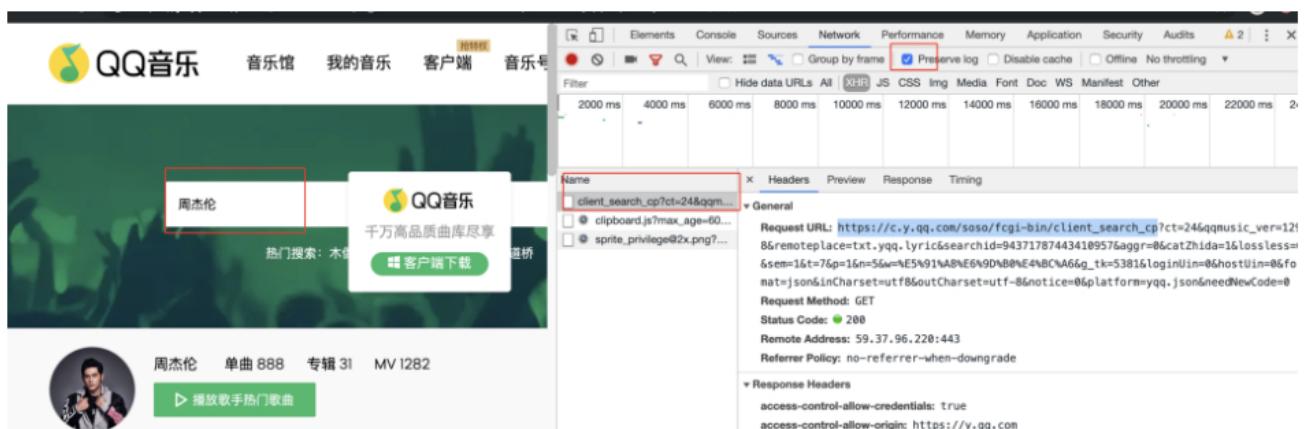
4
5     for x in range(10):
6         url = 'https://movie.douban.com/top250?start=' + str(x * 25) + '&filter='
7         # res = requests.get(url)
8         res = requests.get(url, headers=headers)  # 向get()函数添加参数headers。
9         bs = bs4.BeautifulSoup(res.text, 'html.parser')
10        bs = bs.find_all('div', class_="item")
11        for titles in bs:
12            num = titles.find('em', class_="").text
13            title = titles.find('span', class_="title").text
14            comment = titles.find('span', class_="rating_num").text
15            url_movie = titles.find('a')['href']
16
17            if titles.find('span', class_="inq") != None:
18                tes = titles.find('span', class_="inq").text
19            else:
20                tes = "没有评论语"
21            print(num + '.' + title + '—' + comment + '\n' + '推荐语: ' + tes + '\n' +
url_movie)

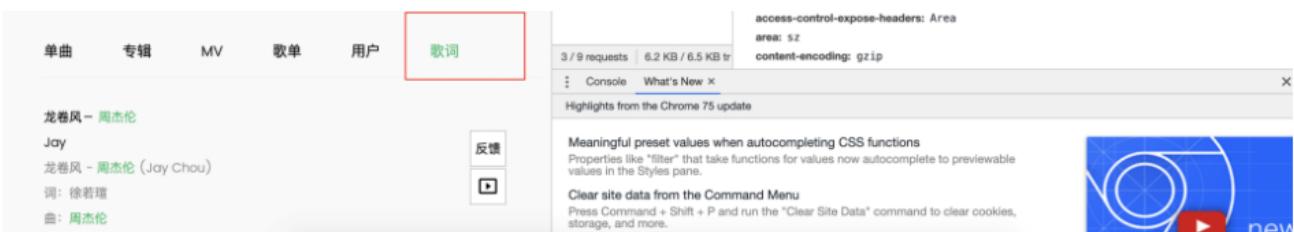
```

第5关 带参数请求数据

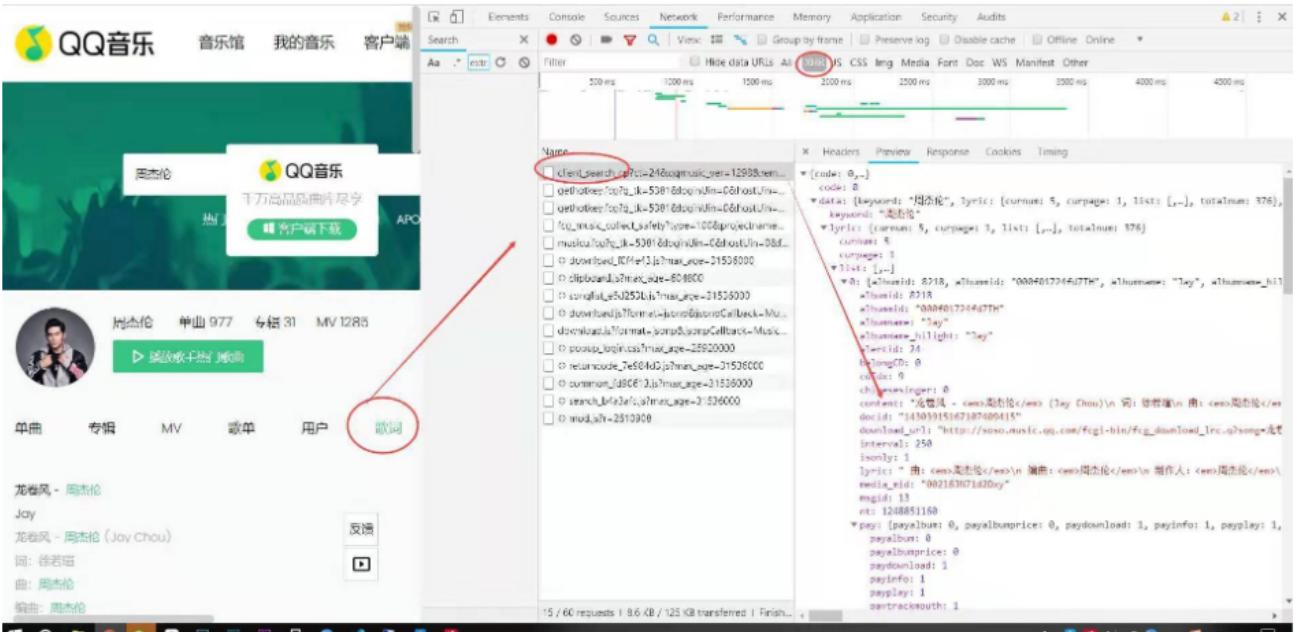
课后练习1：歌词爬取

- 题目解析：
 - 第0步-打开QQ音乐网站-输入关键字‘周杰伦’-点击歌词。 (图一)
 - 第1步-寻找在请求中的歌词信息。 (图二)
 - 第2步-改变网址中page变量的值，打开Network - XHR-client_search - Headers - Query String Parameters，观察里面参数的变化。





四



二

- 参考代码:

```
1 import requests
2 import json
3 # 引用requests,json模块
4
5 url = 'https://c.y.qq.com/soso/fcgi-bin/client_search_cp'
6 headers = {
7     'referer':'https://y.qq.com/portal/search.html',
8     'user-agent':'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/71.0.3578.98 Safari/537.36'
9     # 标记了请求从什么设备, 什么浏览器上发出
10 }
11
12 for x in range(5):
13     params = {
14         'ct':'24',
15         'qqmusic_ver': '1298',
16         'new_json':'1',
17         'remoteplace':'sizer.yqq.lyric_next',
18         'searchid':'94267071827046963',
19         'aggr':'1',
20         'cr':'1',
21         'catzhida':'1',
22         'lossless':'0',
23         'sem':'1',
24         't':'7',
25         'p':str(x+1),
26         'n':'10',
27         'w':'周杰伦',
28         'g_tk':'1714057807',
29         'loginUin':'0',
30         'hostUin':'0',
31         'format':'json',
32         'inCharset':'utf8',
33         'outCharset':'utf-8',
34         'notice':'0',
35         'platform':'yqq.json',
36         'needNewCode':'0'
37     }
38
39     res = requests.get(url, params = params)
40     #下载该网页, 赋值给res
41     jsonres = json.loads(res.text)
```

```
42     #使用json来解析res.text
43     list_lyric = jsonres['data']['lyric']['list']
44     #一层一层地取字典，获取歌词的列表
45
46     for lyric in list_lyric:
47         #lyric是一个列表，x是它里面的元素
48         print(lyric['content'])
49     #以content为键，查找歌词
```

课后练习2：头号粉丝

- 题目解析：
 - 观察xhr并爬取其他歌手的音乐信息（包括歌曲名，专辑名，播放时长，播放链接）。
 - 打开Network - XHR-client_search - Headers - Query String Parameters，通过观察，参数'w'表示的是歌手的名字，因此，我们可以改变参数'w'，来达到爬取不同歌手的音乐信息。
- 参考代码：

```
1 import requests, json
2
3 # 引用requests模块
4 url = 'https://c.y.qq.com/soso/fcgi-bin/client_search_cp'
5 singer = input('你喜欢的歌手是谁呢？')
6 for x in range(6):
7     params = {
8         'ct':'24',
9         'qqmusic_ver': '1298',
10        'new_json':'1',
11        'remoteplace':'txt.yqq.song',
12        'searchid':'70717568573156220',
13        't':'0',
14        'aggr':'1',
15        'cr':'1',
16        'catzhida':'1',
17        'lossless':'0',
18        'flag_qc':'0',
19        'p':str(x+1),
20        'n':'20',
21        'w':singer,
22        'g_tk':'714057807',
23        'loginUin':'0',
24        'hostUin':'0',
```

```

25     'format':'json',
26     'inCharset':'utf8',
27     'outCharset':'utf-8',
28     'notice':'0',
29     'platform':'yqq.json',
30     'needNewCode':'0'
31 }
32 # 将参数封装为字典
33 res_music = requests.get(url,params=params)
34 # 调用get方法，下载这个列表
35 json_music = res_music.json()
36 # 使用json()方法，将response对象，转为列表/字典
37 list_music = json_music['data']['song']['list']
38 # 一层一层地取字典，获取歌单列表
39 for music in list_music:
40     # list_music是一个列表，music是它里面的元素
41     print(music['name'])
42     # 以name为键，查找歌曲名
43     print('所属专辑: '+music['album']['name'])
44     # 查找专辑名
45     print('播放时长: '+str(music['interval'])+'秒')
46     # 查找播放时长
47     print('播放链接: https://y.qq.com/n/yqq/song/'+music['mid']+'.html\n\n')
48     # 查找播放链接

```

第6关 CSV&Excel

课后练习：存储电影信息

- 题目解析：
 - 使用excel表格存储：
 - 爬取逻辑同第三关豆瓣练习。
 - 使用openpyxl库创建工作簿和工作表之后，在第一行A1到E1位置写上表头内容。
 - 使用工作表对象的append()方法可以一行一行地向表格中添加内容，append()函数的参数是一个列表。
 - 使用csv表格存储：
 - 使用open函数可以创建一个用于读取文件的对象。再用csv.writer()函数创建一个适用于csv表格的写入对象。
 - 使用csv表格写入对象的writerow()方法可以按行写入内容，函数的参数是列表形式。

- 参考代码:

```
1 # Excel存储
2 import requests, random, bs4, openpyxl
3 wb=openpyxl.Workbook()
4 #创建工作薄
5 sheet=wb.active
6 #获取工作薄的活动表
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 headers = {'User-Agent':
16 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_5) AppleWebKit/537.36 (KHTML, like Gecko)
17 Chrome/85.0.4183.121 Safari/537.36'}
18
19 for x in range(10):
20     url = 'https://movie.douban.com/top250?start=' + str(x*25) + '&filter='
21     res = requests.get(url,headers = headers)
22     bs = bs4.BeautifulSoup(res.text, 'html.parser')
23     bs = bs.find('ol', class_="grid_view")
24     for titles in bs.find_all('li'):
25         num = titles.find('em',class_="").text
26         title = titles.find('span', class_="title").text
27         comment = titles.find('span',class_="rating_num").text
28         url_movie = titles.find('a')['href']
29
30         if titles.find('span',class_="inq") != None:
31             tes = titles.find('span',class_="inq").text
32             sheet.append([num, title, comment, tes, url_movie])
33             # 把num, title, comment, tes和url_movie写成列表, 用append函数多行写入Excel
34             print(num + '.' + title + '—' + comment + '\n' + '推荐语: ' + tes + '\n' +
35             url_movie)
36         else:
37             sheet.append([num, title, comment, None,url_movie])
38             print(num + '.' + title + '—' + comment + '\n' + '\n' + url_movie)
39
40 wb.save('movieTop250.xlsx')
```

39 #最后保存并命名这个Excel文件

```
1  # csv方法
2  import requests, random, bs4, csv
3  #引用csv模块。
4  csv_file=open('movieTop250.csv', 'w', newline='')
5  #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=''.
6  writer = csv.writer(csv_file)
7  # 用csv.writer()函数创建一个writer对象。
8  writer.writerow(['序号', '电影名', '评分', '推荐语', '链接'])
9  #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',
10 '链接'
11
12 headers = {'User-Agent':
13     'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_5) AppleWebKit/537.36 (KHTML, like Gecko)
14     Chrome/85.0.4183.121 Safari/537.36'}
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
```

csv方法
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'

headers = {'User-Agent':
 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_5) AppleWebKit/537.36 (KHTML, like Gecko)
 Chrome/85.0.4183.121 Safari/537.36'}

for x in range(10):
 url = 'https://movie.douban.com/top250?start=' + str(x*25) + '&filter='
 res = requests.get(url, headers = headers)
 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 爬取特朗普新闻

课后练习：做个测单词的小工具

- 题目解析：
 - 第0步-选择题库。
 - 第1步-根据选择的题库，获取50个单词。
 - 第2步-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。
 - 第3步-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。
 - 第4步-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：第输出统计数据；打印错题集；把错题集保存到本地。
- 参考代码：

```
1 import requests
2 link = requests.get('https://www.shanbay.com/api/v1/vocabtest/category/')
3 #先用requests下载链接。
4
5 js_link = link.json()
6 #解析下载得到的内容。
7
8 # ----第0步-选择题库-----
9 bianhao = int(input('''请输入你选择的词库编号，按Enter确认
10 1, GMAT 2, 考研 3, 高考 4, 四级 5, 六级
11 6, 英专 7, 托福 8, GRE 9, 雅思 10, 任意
12 >'''))
13 #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
14 ciku = js_link['data'][bianhao-1][0]
15
16 # ----第1步-根据选择的题库，获取50个单词-----
17 test=requests.get('https://www.shanbay.com/api/v1/vocabtest/vocabularies/?category='+ci
ku)
18 #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。
19 words = test.json()
20 #下载并解析用于测试的50个单词。
21
22 danci = []
23 #新增一个list，用于统计用户认识的单词
24 words_knows = []
```

```
25 #创建一个空的列表，用于记录用户认识的单词。
26 not_knows = []
27 #创建一个空的列表，用于记录用户不认识的单词。
28 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: ')
29
30 # ----第2步-让用户选择认识的单词----
31 n=0
32 for x in words['data']:
33 #启动一个循环，循环的次数等于单词的数量。
34     n=n+1
35     print ("\n第"+str(n)+"个: "+x['content'])
36     #加一个\n，用于换行。
37     answer = input('认识请敲Y，否则敲Enter: ')
38     #让用户输入自己是否认识。
39     if answer == 'Y':
40         #如果用户认识:
41         danci.append(x['content'])
42         words_knows.append(x)
43         #就把这个单词，追加进列表words_knows。
44     else:
45         #否则
46         not_knows.append(x)
47         #就把这个单词，追加进列表not_knows。
48 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：')
49 print (danci)
50
51 # ----第3步-对于用户认识的单词，给选择题让用户做----
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]
63     ['rank'], 'C':y['definition_choices'][2]['rank'], 'D':y['definition_choices'][3]['rank']}
```