实验一：爬虫

爬虫：爬取当当网搜索关键词网站信息

班级	SC011701
学号	2017302242
姓名	裴嘉琨
实验时间	第4周

一、实验目的

爬虫是一个模拟人类请求网站行为的程序。可以自动请求网页、并数据抓取下来，相关人员可以使用一定的规则提取有价值的数据。

本次实验的主要目的是利用Python，编写出相关代码对某特定网页的网站信息进行相关的信息爬取。我选择的目标网站为当当网，主要针对当当网主页在搜索关键词后的页面，爬取网页上相关物品的信息。

二、实验环境

Python 3.7

Pycharm

Anaconda 3

requests库

三、实验要求

- 通过python可以实现访问某个网页来进行数据的读取；
- 针对网页可以自行实现一定页数的数据自动读取，并寻找到所需要的重要信息；
- 将网页上读取的相关信息进行整理并通过excel表格进行呈现

四、设计思路及前期准备

1）爬虫

爬虫就是通过编程来全自动的从互联网上采集数据，模拟正常人类发起的网络请求，然后获取网络请求所返回的数据。通过程序来发送请求，服务器会相应的返回不同的数据格式有的会是HTML，有的是JSON,有的是二进制数据。再通过正则表达式，来进行细节的选取。

2）目标网页的选定

代码可以实现根据设定的关键字keyword获取相关商品的资源定位符(url)，然后批量爬取相关页面的商品信息，另外之所以选择当当网是因为当当网的网页商品信息不是动态加载的，因此可以直接爬取获得，例如京东、拼多多的网页就是动态加载的。相对于静态的网页而言是较难的。所以进行对静态网页的一个爬取。

3）当当网url分析

①网页规律

进入当当网进行商品查询，得到相关的网页 链接为

```
url_1='http://search.dangdang.com/?key=%BF%BC%D1%D0%D7%CA%C1%CF&act=input'
url_2='http://search.dangdang.com/?
key=%BF%BC%D1%D0%D7%CA%C1%CF&act=input&page_index=2'
#查询考研资料 后面会搭配相关的page
url_3='http://search.dangdang.com/?
key=%C9%B1%CB%C0%D2%BB%D6%BB%D6%AA%B8%FC%C4%F1&act=input&page_index=2'
#搜索英文keyword得到的
url_4='http://search.dangdang.com/?key=Python&act=input'
```

`url_1='http://search.dangdang.com/?key=%BF%BC%D1%D0%D7%CA%C1%CF&act=input'`
`url_2='http://search.dangdang.com/?key=%BF%BC%D1%D0%D7%CA%C1%CF&act=input&page_index=2'`

> 在单单输入一个关键词后会生成一个相关的链接，他会主要会显示相关的位置呈现出搜索的keyword。
> 但是再翻页后会出现相关的页码信息，出现在最后位置，呈现为：page_index=x。
>
> 后面测试也可以知道如果输入page_index=1，其实也是可以搜索到的第一页。

通过这样的规律我们可以总结得到当当网搜索界面的基础url为：

`http://search.dangdang.com/?+keyword(查询词) &act=input&page_index=`

②解析网页内容

浏览器打开任一网页，打开开发者模式可以发现，当当网的书籍商品经过调试发现，页面的商品信息都在ul标签下的中

- 标签中，书籍商品页面的ul标签的class属性是"bigmig"。所以针对书籍的数据获取只需要针对该类别中进行爬取即可。同时在找寻规律时发现，除了书籍类商品，其类别并不相同，本次实验主要针对书籍类信息进行爬虫参考。

```
971 </script>
972 </div><div class="spacer"></div><div ddt-area=5402556 ddt-expose="on"><div id='component_
973         <ul class="bigimg" id="component_59">
974                                 <li ddt-pit="1" class="line1" id="p24003310">
975                     <a title=" Python编程 从入门到实践"  ddclick="act=normalResult picture
```

（部分相关知识来源百度、CSDN等网站）

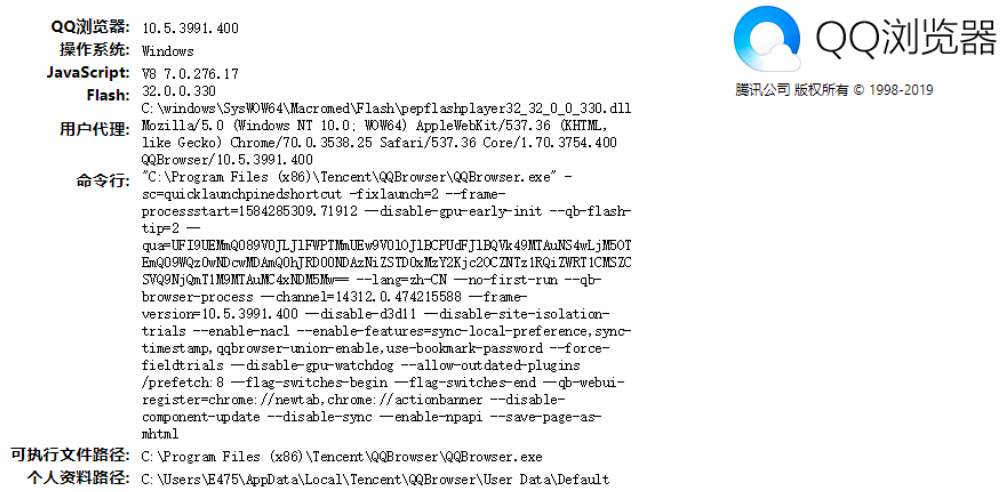
五、代码实现

1）head信息

在爬虫过程中需要运用一个user_agent来模拟一个用户。

```
user_agent:Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/70.0.3538.25 Safari/537.36 Core/1.70.3754.400
QQBrowser/10.5.3991.400
#在浏览器中可以查出自己的用户代理者
```

> 查询方式：在浏览器输入：chrome://version。得到下图：



2）数据的爬取

前面已经得到当当网首页在进行搜索关键词后会出现的基本url。我们针对其链接进行进一步读取。通过自己需要查询的相关keyword，结合整个url来爬取数据。为了可以实现多页之间的自主爬取，调用循环语句。并且在每成功爬取一页后进行文字上的反馈，确保整个过程的透明和稳定。

为了可以得到针对自己真正需要的数据，所以进行正则表达式对其再次处理，可以将主要文字进行取出保存在已经设置好的列表中。

```
    try:
        key = "数据库"  # 换成任何你想要搜索的书籍
        url = url_basic + key + "&act=input&page_index=" + str(i + 1)#设定规范的url
        print(url)
        response = requests.get(url, headers=heads)
        content = response.text
        pattern = re.compile(
            '<li.*?<a title="(.*?)".*?>.*?search_now_price">.*?(\d+\D\d+)</span>.*?search_pre_price">.*?(\d+\D\d+)</span>.*?<a href=.*?ddclick=.*?>(\d+).*?</a>.*?<a href=.*?>(.*?)</a>.*?</span>.*?</li>',
            re.S)
        results = re.findall(pattern, content)
        yucun += results
        print("获取成功")#如果可以正常输出，则可以看到情况
```

3)数据存储

在数据爬取成功后主要是要针对数据的存放，可以使用存到文本中或为了更好的观感，我选择将其放入到提前建立好的excel文件中。

```
biaotou = ["序号", "书名", "现价", "原价", "评论数", "作者"]
with open(r"D:\python.xlsx", "w") as file:
    ……
    file.save(r"D:\python.xlsx")
print("数据已保存")
```

六、实验结果及分析

实验结果为下：针对关键词的选取，我们进行中文、英文分别选取，进行相关测试。页码均取5。

首先在需要存储Excel的位置创建好Excel表格。

python.xlsx 2020/3/19 22:15 XLSX 工作表 26 KB

①keyword='信息安全'

```
for i in range(5):  #括号内写入需要爬取的具体页数
    try:
        key = "信息安全"  #输入想查询的关键词
        url = url_basic + key + "&act=input&page_index=" + st
        print(url)
```

```
http://search.dangdang.com/?key=信息安全&act=input&page_index=1
获取成功
http://search.dangdang.com/?key=信息安全&act=input&page_index=2
获取成功
http://search.dangdang.com/?key=信息安全&act=input&page_index=3
获取成功
http://search.dangdang.com/?key=信息安全&act=input&page_index=4
获取成功
http://search.dangdang.com/?key=信息安全&act=input&page_index=5
获取成功
数据已保存

Process finished with exit code 0
```

最终在Excel中呈现结果（部分）为：

序号	书名	现价	原价	评论数	作者
1	信息安全	74.90	89.00	1394	斯坦普
2	网络攻防	66.50	79.00	933	郭帆
3	信息安全	107.70	128.00	749	Mark
4	企业级信	36.10	45.00	134	中国电力出
5	信息安全	49.70	59.00	119	马克·瑞恩
6	信息安全	104.90	128.00	2075	张焕国
7	信息安全	53.20	59.00	417	郭鑫
8	信息安全	68.30	99.00	53	Charles
9	信息安全	22.70	33.00	55	程庆梅
10	信息安全	53.70	68.00	6	贝森
11	信息安全	45.00	45.00	60	朱建明
12	信息安全	23.40	34.00	149	牛少彰
13	信息安全	44.20	59.00	0	周世杰
14	信息安全	21.30	27.00	103	唐成华
15	信息安全	33.10	42.00	114	汤永利
16	网络信息	15.60	21.00	30	庞淑英
17	信息安全	17.90	26.00	13	马传龙
18	信息安全	12.40	15.00	826	全国计算机
19	信息安全	81.30	118.00	75	Charles
20	信息安全	34.70	44.00	6	杨建强
21	网络信息	35.50	45.00	250	曾凡平
22	信息安全	34.30	49.80	41	罗森林
23	国土信息	16.30	18.00	24	中国科协学
24	信息安全	16.30	25.00	61	毕方明
25	信息安全	17.30	22.00	18	方祥圣
26	信息安全	17.20	25.00	61	吴晓平
27	信息安全	19.70	25.00	29	巫玲

②keyword='python'

```
try:
    key = "python"  #输入想查询的关键词
    url = url_basic + key + "&act=input&page_inc
    print(url)
    response = requests.get(url, headers=heads)
```

最终在Excel中呈现结果（部分）为：

序号	书名	现价	原价	评论数	作者
1	Python编	59.70	89.00	124695	埃里克·马
2	Python编	62.20	69.00	26199	Sweigart
3	对比Exce	54.50	59.00	10971	张俊红
4	DK编程真	116.80	118.00	2524	克雷格·斯
5	Python Q	89.30	99.00	1058	王维波
6	Python从	74.90	89.00	4357	关东升
7	Python基	89.30	99.00	17692	芒努斯·和
8	利用Pyth	87.30	119.00	15199	韦斯·麦金
9	零基础入	74.90	89.00	2802	小甲鱼
10	Python学	160.60	219.00	4142	马克·卢茨
11	Python编	67.20	79.80	5346	刘瑜
12	Python核	89.30	99.00	22676	卫斯理
13	Python神	62.20	69.00	9478	塔里克·拉
14	基于Pyth	106.40	118.00	2491	斯文
15	疯狂Pyth	109.00	118.00	6382	李刚
16	Python C	97.40	108.00	9036	比斯利
17	基于Pyth	58.80	69.80	5289	余本国
18	Python 3	89.30	99.00	11561	崔庆才
19	Python网	71.30	79.00	1566	瑞安·米切
20	教孩子学	53.20	59.00	14267	Bryson
21	Python网	73.00	79.00	502	庄培杰
22	Python真	69.50	88.00	891	刘凤飞
23	Python语	39.00	39.00	2182	嵩天
24	流畅的Py	125.30	139.00	5780	卢西亚诺·
25	Python机	82.20	89.00	1038	Chris
26	Python统	71.30	79.00	1285	托马斯·哈

> **备注**：为了保证程序的正常运行和数据的存储，我们需要每次执行完程序后将想保存的数据另存为一个新的Excel中。再关闭，执行下一次的程序。

七、总结

在本次实验中，主要实现了对特定的页面上的数据进行爬取的工作。基本工作已经实现。但是也只是对基本的爬虫功能进行了实现。调用了requests,re,xlwt三个库实现主要功能，并将数据最后存放到确定路径上的Excel中。

在本次实验中，遇到的基础问题比较多，因为自己之前接触Python相关知识较少，很多基础内容操作起来不是很顺利。比如一开始的anaconda环境的配置，一些库的安装，具体爬虫的实现还有最后对数据存储的部分，存储到一个新的文件中也是本次实验才第一次运用到自己的代码当中。在面对这些问题时，主要利用百度和csdn网站上的知识点进行操作得到最终的完整代码。

本次实验首先在爬取内容上仍是有所不足，因为当当网存储信息的分类情况，书籍类和非书籍类是不同的，所以本次实验在定关键词时需要定义一些查询结果是书籍类商品较好些，这也算是本次实验的未能实现的一个不足之处。未来针对爬虫我觉得可以逐步尝试爬虫一些动态网站或是针对不同的信息进行爬取，因为本次实验主要只是爬取书籍的信息，还有很多别的信息可以尝试的爬取、整理。在搜索过程中，也看到有些人将爬取的数据与数据库进行连接更加方便存取和查看。