

信息内容安全实验报告

实验项目名称：网络爬虫获取中国大学排名

班级：SC011701

学号：2017302240

姓名：郑博文

指导教师：杨黎斌

实验时间：2020年3月19日

目录

1. 工作背景.....	3
2. 实现描述.....	3
3. 程序结构设计	3
3.1 设计步骤	3
3.2 函数设计	3
4. 程序实现过程	4
4.1 爬虫实现的可行性	4
4.2 函数实现	4
4.2.1 getHTMLText()函数.....	4
4.2.2 fillUnivList()函数.....	5
4.2.3 printUnivList()函数.....	5
5. 实验结果.....	6
6. 问题解决.....	7

网络爬虫获取中国大学排名

1. 工作背景

对一个大学水平的评判可以通过各个方面对其进行评估分析并量化为数值，来进行排名，社会上有各种各样的大学排名标准，而本次实验通过定向网络爬虫对其中一个排名方式——上海交通大学进行的“软科”中国大学排名的数据进行抓取，创建文本并写入抓取的信息。

2. 实现描述

- (1) 输入：输入大学排名网站的 URL 链接
- (2) 输出：大学排名信息输出并创建写入文本（信息包括：排名，大学名称，地区）
- (3) 实现函数库：requests、bs4（包括 BeautifulSoup 类）、os。

3. 程序结构设计

3.1 设计步骤

- 步骤 1：从网络上获取大学排名的网页内容
- 步骤 2：分析提取网页源代码中的信息至合适的数据结构

从网页中获取的大学排名包括排名以及大学的基本信息，因此采用二维列表的数据结构形式存放从网页中获取的信息。

- 步骤 3：利用数据结构展示并输出结果以及文本

3.2 函数设计

- 函数 1：getHTMLText(url)

从网络上获取大学排名网页的内容信息。

- 函数 2：fillUnivList(ulist, html)

提取网页内容信息并分析整合至合适的数据结构——二维列表中。

- 函数 3: `printUnivList(ulist, num)`

将放入数据结构的数据信息格式化输出并创建写入文本

4. 程序实现过程

4.1 爬虫实现的可行性

Robots 协议：**robots** 协议也叫 **robots.txt**（统一小写）是一种存放于网站根目录下的 ASCII 编码的文本文件，它通常告诉网络搜索引擎的漫游器（又称网络蜘蛛），此网站中的哪些内容是不应被搜索引擎的漫游器获取的，哪些是可以被漫游器获取的。因为一些系统中的 URL 是大小写敏感的，所以 **robots.txt** 的文件名应统一为小写。**robots.txt** 应放置于网站的根目录下。如果想单独定义搜索引擎的漫游器访问子目录时的行为，那么可以将自定的设置合并到根目录下的 **robots.txt**，或者使用 robots 元数据（Metadata，又称元数据）¹。

通过访问 <http://www.zuihaodaxue.cn/robots.txt> 该网址来确认该网站是否存在 robots 协议。访问结果如下图：



图 1. www.zuihaodaxue.cn/robots.txt 的访问结果

从图中可以得知访问结果为“404 Not Found”结果及说明，即该网页不存在 robots 协议对爬虫进行限制，可以通过爬虫对该网站实现信息获取。

4.2 函数实现

4.2.1 getHTMLText()函数

`getHTMLText()`函数参数为 `url`，作用是将网络中的信息爬取下来，并将其中的 HTML 页面返回给其他函数程序。

通过 `requests.get()`函数获取网页内容，最长响应时间为 30 秒，用 `raise_for_status` 产生异常问题，并修改编码，最后将内容文本返回。若出现异常，则返回空字符。

¹ [1] 百度百科: <https://baike.baidu.com/item/robots%E5%8D%8F%E8%AE%AE/2483797?fr=aladdin&fromid=9518761&fromtitle=robots.txt>

以下是 getHTMLText()函数截图：

```
def getHTMLText(url):
    try:
        r=requests.get(url,timeout=30)
        r.raise_for_status()
        r.encoding = r.apparent_encoding
        return r.text
    except:
        return ""
    return ""
```

图 2. getHTMLText()函数

4.2.2 fillUnivList()函数

fillUnivList()函数参数为 ulist 以及 html, ulist 为待填入数据的二维列表, html 为 getHTMLText 函数获取的网页信息。该函数是提取网页中的信息并填入列表。

先通过 BeautifulSoup 对 HTML 页面进行解析,再观察网页的源代码,如图:

```
<tbody class="hidden_zhpm" style="text-align: center;">
<tr class="alt"><td>1</td><td align="left">清华大学</div></td><td>北京</td><td>94.6</td><td class="hidden-xs need-hidden
```

图 3. 包含大学信息的对应部分网页源代码

这些大学信息被封装在一个表格中,标签名为“tbody”,而每一所大学的信息又被封装在“tr”标签中,即每一个“tr”标签包含当前大学的所有信息,而每个单独的信息有存储在“td”标签中。因此整个获取信息的过程应该是如下的过程:

- ①在 HTML 文件中解析<tbody>标签,获取所有大学的信息;
- ②在<tbody>标签中解析<tr>标签,获得每个大学的信息;
- ③在<tr>标签中解析<td>标签,获得每个大学每一个属性信息的数据值写入二维列表——ulist。

运用 for 语句查找 HTML 文本中的<tbody>标签,对其子结点(children)进行遍历。但是在这些“tr”标签中或多或少存在“string”字符串类型,但是该网页中的信息全都是“Tag”标签类型,因此需要过滤非“标签”类型的其他信息,即只保留“标签”类型的信息,采取 isinstance()函数对文本类型进行判断、筛选,只选取“标签”类型的数据。筛选完成后将所有的“td”标签存在 tds 列表中,并在 ulist 目标列表中添加所需字段数据。具体代码如下图:

```
def fillUnivList(uList , html):
    soup =BeautifulSoup(html,"html.parser")
    for tr in soup.find ('tbody').children:
        if isinstance(tr,bs4.element.Tag):
            tds =tr('td')
            ulist.append([tds[0].string ,tds[1].string ,tds[2].string ])
```

图 4. fillUnivList()函数

4.2.3 printUnivList()函数

该函数参数为 ulist 列表和 num 所需查询的大学数目,目标是将列表格式化输出并创建 txt 文本写入数据。先定义 txt 文件存储位置,定为上级目录的 result

文件夹中，命名为“排名.txt”。最后利用 for 语句以及相关对文件的操作将列表中的所有数据写入文件中。

```
def printUnivList(ulist, num):
    path = "../result/排名.txt"
    tplt="{0:^10}\t{1:{3}^10}\t{2:^10}"
    print (tplt.format("排名", "学校名称", "地区", chr(12288)))
    with open (path, 'a') as f:
        f.write(tplt.format("排名", "学校名称", "地区", chr(12288))+'\n')
    for i in range (num):
        u=ulist[i]
        print (tplt.format(u[0], u[1], u[2], chr(12288)))
        with open (path, 'a') as f:
            f.write(tplt.format(u[0], u[1], u[2], chr(12288))+'\n')
```

图 5. printUnivList()函数

5. 实验结果

通过上述爬虫程序，我们可以轻松获取该网站上的 2019 年的全国大学的“软科”排名。最后在上层目录的 result 文件中创建 txt 文件并写入，最终结果如下图：



排名	学校名称	地区
1	清华大学	北京
2	北京大学	北京
3	浙江大学	浙江
4	上海交通大学	上海
5	复旦大学	上海
6	中国科学技术大学	安徽
7	华中科技大学	湖北
7	南京大学	江苏
9	中山大学	广东
10	哈尔滨工业大学	黑龙江
11	北京航空航天大学	北京
12	武汉大学	湖北
13	同济大学	上海
14	西安交通大学	陕西
15	四川大学	四川
16	北京理工大学	北京
17	东南大学	江苏
18	南开大学	天津
19	天津大学	天津
20	华南理工大学	广东
21	中南大学	湖南
22	北京师范大学	北京
23	山东大学	山东
23	厦门大学	福建

图 6. Txt 文档输出结果

6. 问题解决

在整个爬虫程序的编写中，遇到的比较麻烦的问题就是关于文件数据的输出格式的编写，按照以往的“f.write("排名"+" "+"学校名称"+" "+"地区"+'\n')”这种形式输出将会产生文本中文字不对齐的现象，为了将输出结果变得美观，考虑使用 format 格式化输出对其进行修改，但是究其发生格式不齐的根本原因就在于中文字符与英文字符所占空间大小不同导致，因此可以考虑使用中文空格字符 chr(12288)来进行解决。

先定义格式化模板变量“tplt='{0:^10}\t{1:{3}^10}\t{2:^10}””，再将格式运用到 format 函数上，同时增加 format 函数中的第三个变量——中文空格字符“chr(12288)”，最后便可以得到对齐后的文本输出形式。以下是两种输出方式以及对应的结果之间的对比。

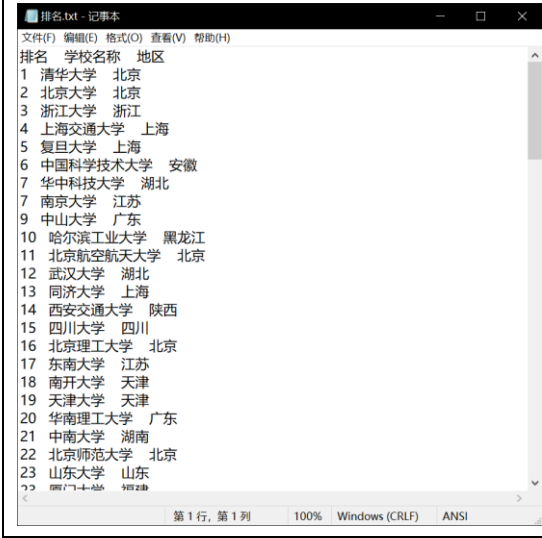

<p>普通的输出代码</p> <pre>def printUnivList(ulist, num): path = "../result/排名.txt" print ("排名"+" "+"学校名称" " "+" "+"地区") with open (path,'a') as f: f.write("排名"+" "+"学校 名称"+" "+"地区"+'\n') for i in range (num): u=ulist[i] print (u[0]+u[1]+u[2]) with open (path,'a') as f: f.write(u[0]+u[1]+u[2]+'\\n')</pre>	<p>使用 format 格式化输出的代码：</p> <pre>def printUnivList(ulist, num): path = "../result/排名.txt" tplt="{0:^10}\t{1:{3}^10}\t{2:^10} " print (tplt.format("排名","学校名称 ","地区",chr(12288))) with open (path,'a') as f: f.write(tplt.format("排名","学校 名称","地区",chr(12288))+'\n') for i in range (num): u=ulist[i] print (tplt.format(u[0],u[1],u[2],chr(12288))) with open (path,'a') as f: f.write(tplt.format(u[0],u[1],u[2],chr (12288))+'\n')</pre>
	

表 1. 两种输出方式的对比