**Python项目说明**

1. 项目标题

本校研究生招生网的信息爬取

1. 参与人员名单：

|  |  |  |
| --- | --- | --- |
| 姓名 | 学号 | 分工 |
| 李松 | 202005002175 | 参考资料查找、代码的整合纠错、项目编写 |
| 王文辉 | 202005002169 | 代码编写、运行测试、资料整合、代码纠错 |
| 郑东旭 | 202005002175 | 代码编写、运行测试、资料整合、代码纠错 |

1. 项目介绍

##### 实现了爬取云南财经大学官网研究生招生相关文章并转换成 Word 文档的功能。其中，程序使用 requests 库获取网页内容，使用 Beautiful Soup 库对网页内容进行解析，使用 fake\_useragent 库模拟浏览器请求头，实现反反爬虫。程序通过遍历各页面和新闻的方式，根据链接的类型进行分类处理。对于网页中的文章和表格，程序将其保存至 Word 文档中；对于 pdf 文件，则直接下载到本地。

1. 详细运行说明（包括关键环节的截图）

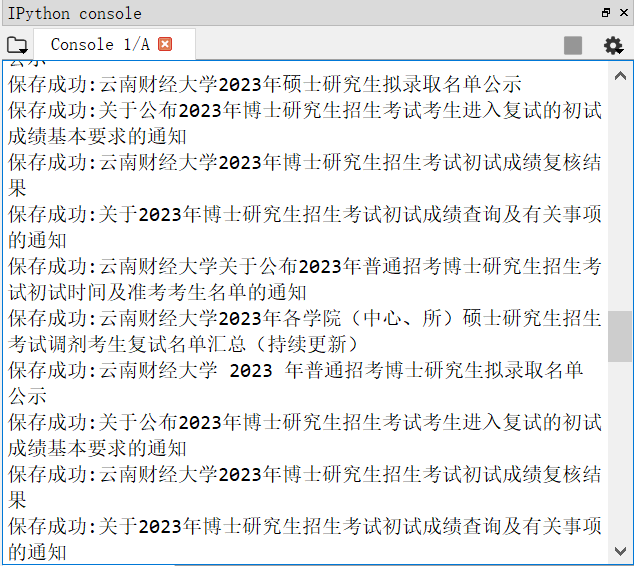
主函数 main() 在获取网页内容后，通过循环遍历不同部分的新闻（共 4 部分），然后获取每个新闻的详情链接并进行处理，最终保存为 Word 文档。

函数 save\_word() 接受一个 BeautifulSoup 对象和一个 URL，首先通过 select\_one() 方法获取新闻标题，并用 add\_heading() 方法将其添加到 Word 文档中。然后通过遍历 tag 标签的子标签，将其中的文本和链接添加到 Word 文档中。如果遇到文本，调用函数 text\_save() 进行处理；如果遇到表格，调用函数 table\_save() 处理。

函数 text\_save() 处理文本的保存。程序首先定义一个空字符串，遍历 tag 的所有子标签。如果子标签是 span 标签或 a 标签，则将其文本添加到字符串中。如果子标签是 a 标签，程序还会调用函数 a\_save() 进一步处理，将链接保存到 Word 文档中。

函数 table\_save() 处理表格的保存。程序遍历 table 标签的所有 tr 标签，然后遍历每个 tr 标签下的所有 td 标签。如果当前 td 标签中有 a 标签，则将 a 标签中的文本添加到 row 数组中，并调用函数 a\_save() 进一步处理。如果当前 td 标签中有 span 标签，则将其文本添加到 row 数组中。最后，将 row 添加到 data 数组中，再将 data 数组中的数据填入 Word 表格中。

函数 a\_save() 处理链接的保存。程序首先创建一个临时的文件夹，将文件目录指向这个文件夹。然后根据链接的类型进行处理，如果链接是以 http 开头的绝对路径，则直接访问。否则，将链接转换为相对路径，拼接前缀，再进行访问。如果链接指向的是 pdf 文件，则直接下载到本地。最后，程序返回之前的目录，继续处理其他文件。









1. 参考资料
2. ChatGTP4.0
3. Github——关于目标网站的爬取
4. Github——通过正则表达式实现获取电影的下载地址、
5. 百度网盘——如何爬取网页中动态加载的内容
6. Github——如何通过Requests模块实现Cookie登录
7. Github——如何通过爬虫实现GitHub网页的模拟登录

六、Github链接地址

[LS0310/2175 (github.com)](https://github.com/LS0310/2175)

七、代码（核心代码要求详细注释）和数据（如果涉及）

import requests

from bs4 import BeautifulSoup

from fake\_useragent import UserAgent

import os

import docx

import urllib.request

import re

def main():

# 使用fake\_useragent随机生成User-Agent伪造请求头

ua = UserAgent()

headers = {'User-Agent': ua.random}

url = 'https://www.ynufe.edu.cn/pub/yjsyxz/zs/index.htm'

res = requests.get(url, headers=headers)

# 解析HTML文件

soup = BeautifulSoup(res.text, 'html.parser', from\_encoding='utf-8')

for i in range(4):

if i == 0:

"""

解析招生动态部分

"""

news\_items = soup.select('.ui-index-item.col-md-6.col-sm-12.col-xs-12')[i].select('.news-item')

# 创建文件夹

if not os.path.exists("结果/招生动态"):

os.makedirs("结果/招生动态")

os.chdir("结果/招生动态")

elif i == 1:

"""

解析博士招生部分

"""

news\_items = soup.select('.ui-index-item.col-md-6.col-sm-12.col-xs-12')[i].select('.news-item')

# 创建文件夹

if not os.path.exists("../博士招生"):

os.makedirs("../博士招生")

os.chdir("../博士招生")

elif i == 2:

"""

解析硕士招生部分

"""

news\_items = soup.select('.ui-index-item.col-md-6.col-sm-12.col-xs-12')[i].select('.news-item')

# 创建文件夹

if not os.path.exists("../硕士招生"):

os.makedirs("../硕士招生")

os.chdir("../硕士招生")

elif i == 3:

"""

解析港澳台招生部分

"""

news\_items = soup.select('.ui-index-item.col-md-6.col-sm-12.col-xs-12')[i].select('.news-item')

# 创建文件夹

if not os.path.exists("../港澳台招生"):

os.makedirs("../港澳台招生")

os.chdir("../港澳台招生")

for item in news\_items:

# 获取链接

parts = url.split("/")

del parts[-1] # 删除最后一部分index.htm

new\_url = "/".join(parts) # 组成https://www.ynufe.edu.cn/pub/yjsyxz/zs

link = new\_url+ "/" +item.select\_one('a')['href']

if link.endswith("pdf"): #对pdf结尾单独处理

item\_text = str(item).encode('ISO-8859-1').decode('utf-8')

item\_text = re.sub(r'<.\*?>', '', item\_text) # 去除所有标签

item\_text = re.sub(r'\s+', ' ', item\_text) # 替换连续的空白字符为一个空格

match = re.search(r'\s\*(\w+)\.{3}', item\_text)

if match:

result = match.group(1)

filename = result+".pdf"

else:

filename = "文件名提取失败.pdf"

urllib.request.urlretrieve(link, filename) # 下载文件

else:

# 访问链接并获取html内容

res1 = requests.get(link, headers=headers)

res1.encoding = 'utf-8' # 设置网页编码为utf-8

html\_content = res1.content.decode('utf-8') # 对获取的HTML内容进行解码

soup1 = BeautifulSoup(html\_content, 'html.parser') # 对HTML内容进行解析

parts = link.split("/")

del parts[-1] # 删除最后一部分index.htm

new\_link = "/".join(parts)

# 此时soup1为招生动态的每一个子页面内容

save\_word(soup1,new\_link)

保存方式：

遍历正文部分进行处理，并且拥有参数"当前文件夹"

内容中遇到文本即存入,遇到表格则传递给表格函数存入,遇到链接则判断是htm还是pdf,htm则递归至"当前文件夹+标题"子文件夹

"""

def save\_word(soup:BeautifulSoup,url:str):

"""

将传入的页面保存在当前目录文件夹中,通过os.chdir指定

:param soup对象

:param url目前所在文档链接

"""

title = soup.select\_one('.news-cont-info-article .title').text.strip()

# 创建word和写入标题

doc = docx.Document()

doc.add\_heading(title, 0)

# 处理class="page-news-cont"的主内容

page\_news\_cont = soup.find(class\_='page-news-cont')

# 遍历page\_news\_cont标签中的所有子标签

for tag in page\_news\_cont.descendants:

# 如果是<p>标签则调用text\_save函数

if tag.name == 'p':

text\_save(tag,title,doc,url)

# 如果是<table>标签则调用table\_save函数

elif tag.name == 'table':

table\_save(tag,title,doc,url)

else:

pass

doc.save(title+'.docx')

print("保存成功:"+title)

def text\_save(tag,title,doc,url):

# 处理<p>标签的代码,对应网页中一段话，直接保存，遇到链接做额外处理

text = ""

for child in tag.children:

if child.name == 'span':

text += child.text

elif child.name == 'a':

text += child.text

link = child.get('href')

a\_save(None,title,doc,url,link,child.text)

doc.add\_paragraph(text)

def table\_save(tag,title,doc,url):

# 处理<table>标签的代码

data = []

# 遍历table标签中的所有tr标签

for tr in tag.find\_all('tr'):

row = []

# 遍历当前tr标签中的所有td标签

for td in tr.find\_all('td'):

# 如果当前td标签中有<a>标签，则将<a>标签中的文本添加到row数组中，然后对<a>标签进行额外处理

if td.find('a') is not None:

text = td.find('a').text.strip()

row.append(text)

link = td.find('a')['href'].strip()

a\_save(tag, title, doc, url, link, text)

# 如果当前td标签中有<span>标签，则将<span>标签中的文本添加到row数组中

elif td.find('span') is not None:

row.append(td.find('span').text.strip())

# 否则，将当前td标签中的文本添加到row数组中

else:

row.append(td.text.strip())

# 将当前行数据添加到data数组中

data.append(row)

# 创建一个新的表格对象(网站中有时候有空表格，这时候跳过)

try:

table = doc.add\_table(rows=len(data), cols=max(len(row) for row in data))

except:

return

# 遍历data数组中的每一行数据，并将每个单元格的值填入表格中

for i in range(len(data)):

for j in range(len(data[i])):

table.cell(i, j).text = data[i][j]

def a\_save(tag,title,doc,from\_url,link,text):

"""

处理<a>标签的代码

:param tag:从save\_word传递来的tag标签，暂时没用上

:param title:目前所在文件的文件名

:param doc:目前打开的word文档对象

:param from\_url:目前所在文档的网页链接

:param link:目前要处理的a标签href链接,用于判断直接访问还是和from\_url连接

:param text:目前要处理的a标签的文本

"""

# 将目前位置临时改为文件名，如"结果/招生动态/xxx名单/"

if not os.path.exists(title):

os.makedirs(title)

os.chdir(title)

if link.startswith("http"):

# a标签的href以http开头，直接访问，不需要拼接前置

link=link

else:

# a标签的href是相对路径，需要拼接前置

link=from\_url+'/'+link

if link.endswith("pdf"):

urllib.request.urlretrieve(link, text+".pdf") # 下载文件

# 返回之前的目录，继续其他文件

os.chdir("..")

main()