

# Python程序设计报告

设计题目：**哔哩哔哩视频信息获取**

学生姓名： 袁焕发

专 业： 物联网工程

班 级： 19-2

学 号： 201921776

指导教师： 马学森

完成日期： 2020.01.01

### （一）需求和规格说明

**问题描述：**

抓取哔哩哔哩每日推荐榜单的视频链接，以及视频播放量、点赞、投币等基本信息，并储存在表格文件中

## （二）设计

## 1．设计思想

整体框架为爬虫，首先查看网页源码，找出数据存在部分的关键字，为了减少对网页代码解析的工作量，利用BeautifulSoup模块帮助解析html网页，获取title标签，作为接下来爬取的目标，接下来用request，get方法获取网页text类型的信息，依次获取name以及播放量，点赞量，投币数量，视频链接地址等信息，最后按视频类别爬取，对应好链接地址以及之前获得的各种视频信息，然后依次写入到bili.xlsx文件。

## 2. 设计表示

|  |  |  |
| --- | --- | --- |
| **类别** | **名称** | **描述** |
| 函数 | bs\_parse\_video() | 解析网页 |
| get\_video() | 获取视频类别 |
| bs\_parse\_data() | 获取视频信息 |
| get\_data() | 获取视频链接 |
| delete\_word() | 删除结点 |
| soup.find\_all（） | 查找全部标签 |
| requests.get（） | 发送请求获取链接 |
| output.write（） | 写出文件 |
| 数据 | Fenqu[] | 分区列表 |
| name | 视频类名 |
| link | 视频链接 |
| data\_list[] | 存储视频信息 |
| coin | 硬币数 |
| Like | 点赞数 |
|  |  |

## 核心算法

利用BeautifulSoup可以简化网页解析步骤，方便的获取网页基础信息。

soup=BeautifulSoup(html,'html.parser')指定一种方法解析，

soup.find\_all可以获取所有符合条件的标签，

requests.get发送请求获取信息，r.text 返回网页unicode 型的数据；

get\_text()获得标签对象的文本数据部分，返回字符串；

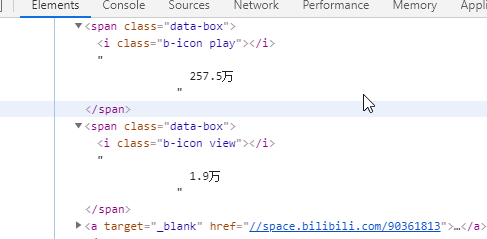
将获取的信息按照视频title分类别存放，当网页获得的视频类别与设定好的类别相同时，开始检索该视频下的文本内容，获取视频播放量点赞等信息，存储在list中，利用output.open方法打开文件，并用write方法按照格式写数据到文件。

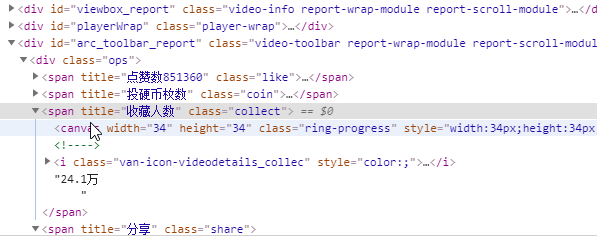
## 用户手册

创建bili.xlsx文件作为存储数据的文件，运行程序代码，等待爬取信息。

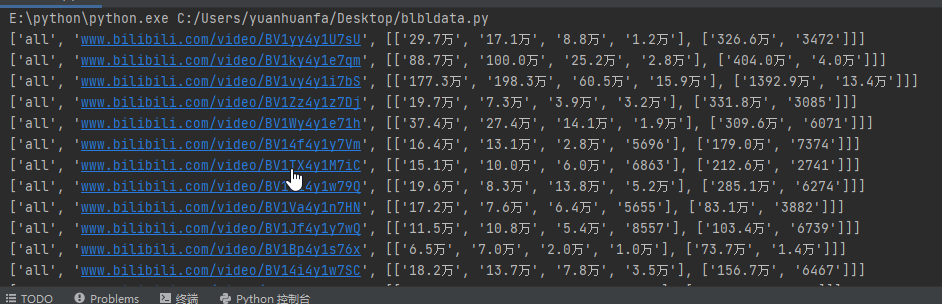
### 调试及测试

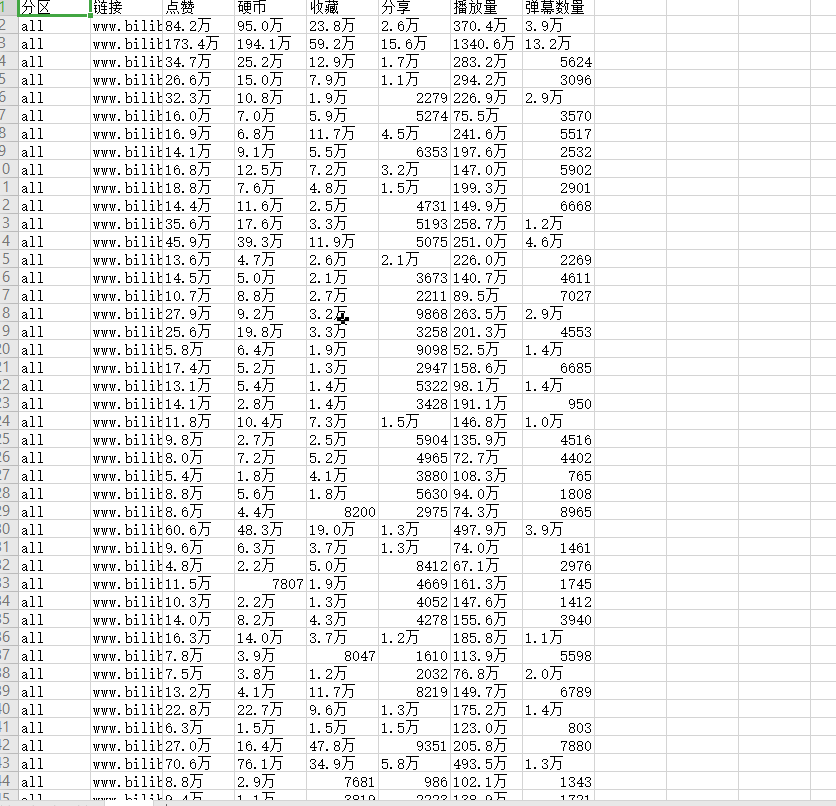
**1.待爬取网页部分信息**

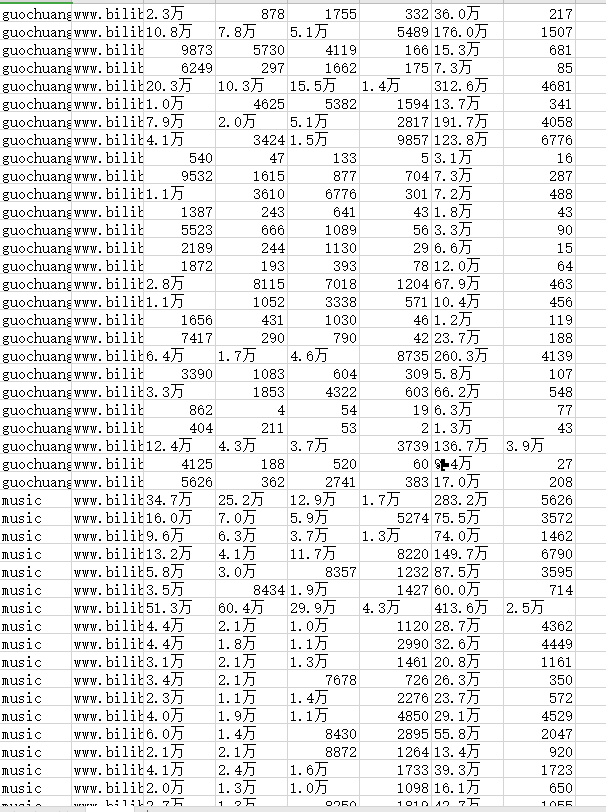




**2.爬取结果**







## (五) 感想

这次的大作业非常感谢python提供的强大的模块，让我可以避免对着网页文件源码一点点查找解析，只要在网页源码中获取一些key作为爬取时的对比标志，就能获得相应的信息，一开始爬取的只有很容易获得的视频链接部分，后来在网页信息中进一步获得了其他文本信息，但是爬取方法不能直接仿照视频链接套用，只能去求助于新的解决方法，但是我对于网页的构成又不够了解，不知道每一层的嵌套，对于网页正则表达式的匹配方法也不能完全理解，于是查找之下发现了这个傻瓜类型的模块，让我也能方便的爬取其他信息，同时众多爬虫类的模板也为我程序各个部分提供了思路。在直接使用文本编辑时很容易出现格式错误，因此选择了pycharm帮助我更好修，虽然爬取信息很方便，但是数据的处理分类却很难处理，很使用容易混乱改，这次作业让我增加了对python的理解，特别是对于模块包强大的功能感到惊讶，C++虽然符合人的思维理解方式，但是要做到这种功能完全做不到这么简洁的，这更加坚定了我用好python这个好工具的决心。

## 附录

import requests

from bs4 import BeautifulSoup

def bs\_parse\_video(html, video\_list=None) -> object:

video\_list = []

soup = BeautifulSoup(html, "html.parser")

# 查找所有class属性为title的标签

div\_list = soup.find\_all('a', class\_="title")

# 获取每个链接

for each in div\_list:

video = each.get('href')

video\_list.append(video)

return video\_list

def get\_video(name):

#网页视频分区组成

link = 'https://www.bilibili.com/v/popular/rank/' + name

r = requests.get(link)

if 200 != r.status\_code:

return None

#返回文本信息

return bs\_parse\_video(r.text)

def bs\_parse\_data(html, video\_list=None) -> object:

soup = BeautifulSoup(html, "html.parser")

# 查找所有class属性为ops的div标签

div = soup.find\_all('div', class\_="ops")

div2 = soup.find\_all('div', class\_="video-data")

# 判断视频是否已失效

div\_error = soup.find\_all('div', class\_="error-body")

if div\_error:

return []

#ops中的信息

if div:

# 获取点赞，投币，收藏与分享数

like = div[0].find('span', class\_="like").get\_text()

like = like[:-5] # 去除无效字符

coin = div[0].find('span', class\_="coin").get\_text()

coin = coin[7:-5]

collect = div[0].find('span', class\_="collect").get\_text()

collect = collect[:-5]

share = div[0].find('span', class\_="share").get\_text()

share = share[:-7]

a = [like, coin, collect, share]

else:

a = []

#videodata中的信息

if div2:

# 获取播放量与弹幕数

view = div2[0].find('span', class\_="view").get\_text()

view = view[:-5]

dm = div2[0].find('span', class\_="dm").get\_text()

dm = dm[:-2]

b = [view, dm]

else:

b = []

return [a, b]

#链接地址

def get\_data(link):

links = 'https:' + link

r = requests.get(links)

if 200 != r.status\_code:

return None

return bs\_parse\_data(r.text)

fenqu = ["all", "guochuang", "music", "douga", "dance", "game", "technology", "digital", "life", "food", "kichiku",

"fashion", "ent","cinephile", "origin", "rookie"]

data\_list = [] # 储存已得到的数据

output = open('bili1.xlsx', 'w')

output.write('分区\t链接\t点赞\t硬币\t收藏\t分享\t播放量\t弹幕数量\n')

for name in fenqu:

movies = get\_video(name)

for link in movies:

data = get\_data(link)

link = link[2:]

data\_list.append([name, link, data])

print([name, link, data])

if data:

if (data[0] != []) & (data[1] != []):

# 按格式读出到excel文件

output.write(str(name))

output.write('\t')

output.write(str(link))

output.write('\t')

output.write(str(data[0][0]))

output.write('\t')

output.write(str(data[0][1]))

output.write('\t')

output.write(str(data[0][2]))

output.write('\t')

output.write(str(data[0][3]))

output.write('\t')

output.write(str(data[1][0]))

output.write('\t')

output.write(str(data[1][1]))

output.write('\n')

output.flush()

output.close()