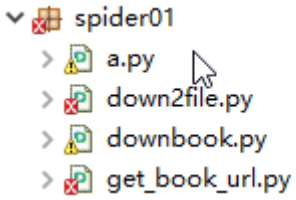
程序目的：根据笔趣看网站的排行榜：https://www.biqukan.com/paihangbang/，获取每部小说的目录页，然后逐页爬取小说，并写入文件。

程序未实现多线程，目前爬取速度缓慢。

目录：



a.py 为主程序，

down2file.py：调用downbook.py的类，逐页下载写入文件

downbook.py：据提供的网站url（用于程序中url的补全）和小说的目录页url，实现了内容的获取（函数）和文件写入（函数）

get_book_url.py:根据排行榜页获取每个小说的目录页

a.py

```
# -*- coding=utf-8 -*-
'''
程序说明：爬虫小说网站的主程序
'''
from spider01 import down2file,get_book_url
from multiprocessing import Pool


myurl='https://www.biqukan.com/paihangbang/' #这个是排行榜那个网址
myup_url='http://www.biqukan.com'#这个是网站url，用于后面程序的生成url时补全头部
get01=get_book_url #实例化
mybookurllist,mybooknamelist=get01.getlist(url=myurl,up_url=myup_url) #获取小说目录页url的列表、小说名称
book=down2file #实例化

for i in range(len(mybookurllist)):
 book.down_file(myup_url,mybookurllist[i],mybooknamelist[i]) #逐本下载
print("所有下载都已完成")

```

down2file.py

```
from spider01 import downbook
import sys ,random
from time import sleep
'''
本程序内类主要实现调用downbook的downloader类，
逐页下载写入相应文件
'''


 class down_file:
  def __init__(self,server,bookurl,bookname):
  self.server=server
  self.bookurl=bookurl
  self.bookname=bookname
  dl = downbook.downloader(server=self.server,bookurl=self.bookurl)
  dl.get_download_url()
  print('《 %s》开始下载： '%self.bookname)
  #for i in range(2): #减少每个小说的下载章数，用于测试程序的运行
  for i in range(dl.nums):
  sleep(10+random.random())#添加random时间
  dl.writer(dl.names[i], '%s.txt'%self.bookname,
dl.get_contents(dl.urls[i]))
  sys.stdout.write(" %s已下载:%.3f%%" % (self.bookname,float(i/dl.nums)) +
'\r')
  sys.stdout.flush()
  print('《%s》下载完成'%self.bookname)
```

downbook.py

```
# -*- coding:UTF-8 -*-
from bs4 import BeautifulSoup
import requests

"""
类说明:根据提供的网站url（用于程序中url的补全）和小说的目录页url，实现了内容的获
取（函数）和文件写入（函数）
Parameters:
 server：网站url（用于程序中url的补全）
 bookurl：小说的目录页url
```

```
Returns:
 无
Modify:
 2018-12-4
"""
class downloader(object):

 def __init__(self,server,bookurl):
 self.server = server
 self.target = bookurl
 self.names = [] #存放章节名
 self.urls = [] #存放章节链接
 self.nums = 0 #章节数

 """
 函数说明:获取下载链接
 Parameters:
 无
 Returns:
 无
 Modify:
 2018-12-4
 """
 def get_download_url(self):
 req = requests.get(url = self.target)
 html = req.text
 div_bf = BeautifulSoup(html,features='lxml')
 div = div_bf.find_all('div', class_ = 'listmain')
 a_bf = BeautifulSoup(str(div[0]),features='lxml')
 a = a_bf.find_all('a')
 self.nums = len(a[12:]) #剔除不必要的章节，并统计章节数
 for each in a[12:]:
 self.names.append(each.string)
 self.urls.append(self.server + each.get('href'))

 """
 函数说明:获取章节内容
 Parameters:
 target - 下载连接(string)
 Returns:
```

```
    texts - 章节内容(string)
    Modify:
    2018-12-4
    """
    def get_contents(self, target):
    req = requests.get(url = target)
    html = req.text
    bf = BeautifulSoup(html,features='lxml')
    texts = bf.find_all('div', class_ = 'showtxt')
    texts = texts[0].text.replace('\xa0'*8,'\n\n')
    return texts

    """
    函数说明:将爬取的文章内容写入文件
    Parameters:
    name - 章节名称(string)
    path - 当前路径下,小说保存名称(string)
    text - 章节内容(string)
    Returns:
    无
    Modify:
    2018-12-4
    """
    def writer(self, name, path, text):
    write_flag = True
    with open(path, 'a', encoding='utf-8') as f:
    f.write(name + '\n')
    f.writelines(text)
    f.write('\n\n')
```

get_book_url.py

```
#-*- coding:UTF-8 -*-
from bs4 import BeautifulSoup
import requests


'''
```

```
类说明:获取网页上单个小说的url（目录页）列表
'''
'''
class Getbookurl:
 def __init__(self):
 pass
'''

'''
 函数说明：获取小说的目录页
 Parameters:
 url：小说网页
 up_url：上级目录的url
 Returns:无
 Modify:2018-12-4
'''
def getlist(url,up_url):
 print("开始整理小说目录页的url列表，请稍等！")
 req=requests.get(url)
 html=req.text
 div_bf=BeautifulSoup(html,features='lxml')
 div=div_bf.find_all('div', class_='block bd')
 bookurls=[] #小说目录页列表
 booknames=[]#小说名称列表
 for i in range(len(div)):
 a_bf=BeautifulSoup(str(div[i]),features='lxml')
 a=a_bf.find_all('a')
 booknums=len(a)
 for each in a:
 bookurls.append(up_url+each.get('href'))
 booknames.append(each.string)
 print('小说目录页的url列表整理完成！',bookurls)
 print('所有需要下载的小说',booknames)
 return bookurls,booknames

```

运行a.py ,最终实现每个小说的下载。

目前程序还是单线程运行，为了规避网站“反爬虫”，设置了随机sleep时间。这样效率下降了很多。