一、实验选题、实验内容及功能说明

①实验选题

爬虫检索系统

②实验内容

从豆瓣Top250电影榜单上爬取所有电影的相关信息（包括影片海报、影片海报链接、影片中外文名、影片评分、影片评价人数、影片概况、影片缩略信息，例如：导演，部分演员，上映时间，电影类型等等），将海报以.jpg形式、将其他信息以excel文件的形式存在本地，数据量足够支撑现场演示的检索需求；在爬取信息的同时产生了一个字典，以排名为键，中文名为值，将其存在本地，方便爬虫完毕后的查找。

以豆瓣榜单中的排名为键，以待查电影存在本地的图片路径和excel路径组成的元组为值，建立B+树，为了扩充查找功能，建立一个hash表，键为豆瓣排名，值为中文名和评分人数，以及评分组成的列表。当要查找排名时，通过键的比对可以迅速定位磁盘路径，减少对磁盘的读取次数，提高查找效率，查找其他信息时通过hash表映射，将在磁盘中的搜索转化为在内存中的搜索，大大提高检索效率。

在GUI界面提供了豆瓣榜单排名准确查找，电影名准确查找，评分区间模糊查找，评分人数区间模糊查找（即查找符合要求区间的所有电影，由于数量过多此时不呈现电影海报）。

为查找的过程提供了异常处理，提高了该查找软件的用户友好度。

二、设计方案与设计思路

本爬虫检索系统分为六个部分：爬取信息，数据结构建立，建立B+树，GUI设计，检索，异常处理。

1. 爬取信息（spider.py）

对豆瓣Top250电影榜单上所有电影的相关信息进行了爬取，并将相关信息进行数据结构建模后存储到本地，以便之后的搜索。

在爬取信息的过程中使用了urllib库，进行了request\_headers的伪装。使用了第三方库fake\_uesragent，频繁更换UA，同时采用了IP代理（IP资源来源于网络），build\_opener加入到headers中，每次发送请求时，从UA和IP池中随机选择进行headers的伪装（使用random库）；在爬取图片时，由于豆瓣电影榜有防盗链，所以还需要在headers中加入Referer进行深层伪装，除此之外，每爬取一定数量的图片，还要暂停一定的时间（time库），进一步模拟用户行为。

由于信息最初请求到的信息是字节流，所以需要以utf-8编码的形式获取相应体内容，返还html形式的文件，然后进行html解析。在解析过程中，选择Beautifulsoup进行第三方库进行处理，生成字符串，最后用re库中的正则表达式匹配提取目的信息。

1. 数据结构建模（spider.py）

在爬取的过程中，以list作为容器储存所有文字信息，并使用第三方库xlwt，对list中的信息进行写入并以excel的形式保存在本地，图片采用jpg形式保存在本地；同时对扩展检索功能的字典进行维护，生成键值对，爬取完毕后，将字典存在本地，当需要进行中文名、评分、评价人数的查找时，先读取hash表，再配合排名B+树为多种查找提供支持。

1. 建立B+树(BPlus.py)

建立了叶子节点类和非叶子节点类，叶子结点中存储的数据为KeyValue型，键为排名，值为包含图片和excel路径的元组），非叶子节点中存储豆瓣排名作为Key。

1. GUI设计(UI.py,UIdesign.py)

GUI设计使用了PyQt5库，先用QTdesigner进行前端界面的设置，产生.ui的文件。然后将.ui文件转化为.py文件。此外GUI设计采用界面交互逻辑和界面设计分离的形式，界面py文件保存在UI.py中，界面逻辑保存在UIdesign.py文件中，方便之后的维护。

当用户点击相应查询框的search按钮之后，触发对应函数，查询结束之后，将结果在label中呈现。

同时为了使用的舒适度与流畅度，在细节上也进行了处理。比如单个电影信息呈现完之后，下一个电影的信息呈现之前要清空所有搜索框和信息框；异常发生时也要进行相应的清除，提高使用舒适度。

1. 检索(BPlus.py)

在进行豆瓣排名的搜索时，通过B+树的键值比对得到目标图片，excel文件的路径，读取图片并呈现在label中采用第三方库QtGUI，excel信息的读取采用第三方库xlrd。

在进行中文名检索时，通过之前建立的字典（hash表），将中文名映射到排名，再通过排名查找对应文件的地址。

在进行给定区间的评分，和评价人数的模糊查找时，根据区间，通过检索hash表，实现快速将所有符合要求的电影信息进行呈现，由于数量过多，不进行图片展示。

1. 异常处理(UI.py)

由于用户可能在排名中输入250以后的数字或者小数或者负数，在评价人数中输入负数，小数，或者在评分中输入大于10的数（10分制），甚至没有输入搜索内容就开始搜索，可能导致程序崩溃，为了提高友好性，增强了程序的异常处理，对可能情况进行异常分析与维护。

三、程序运行效果

打开程序：



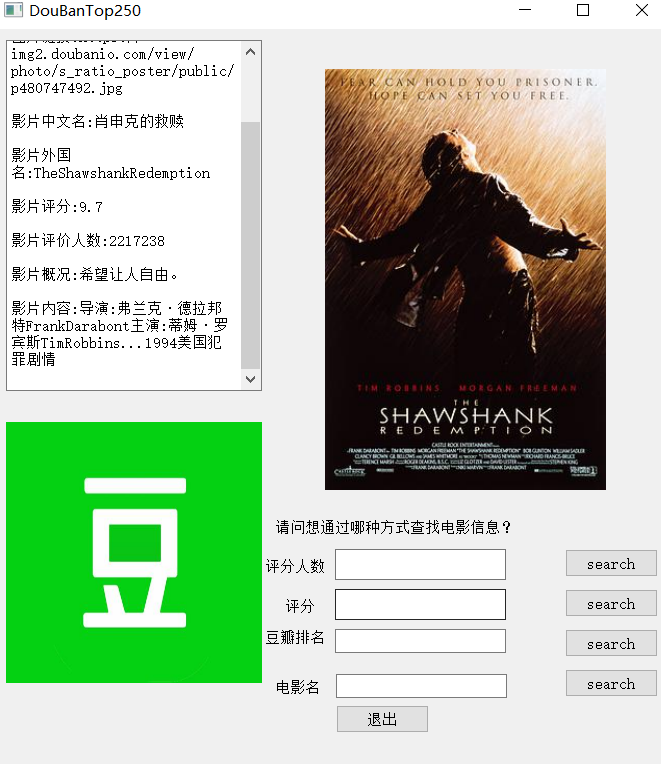
豆瓣排名搜索：





电影名搜索：





评分模糊搜索：





评价人数模糊搜索：





异常处理展示：



（所有搜索窗无输入）





（整数要求）



（排名限制）



（评分限制）





（库中无目标电影）

第三方库函数：

xlrd：excel读取

xlwt：excel写入

PyQt5（Qtcore，QtWidget）：请求html信息

urllib（request）：伪造headers

functools：UI交互逻辑

bs4：html解析

fake\_uesragent：UA更换

四、设计亮点

①爬取过程中的反反爬：由于经常被用来进行爬虫的实验，豆瓣一直在加强反爬措施。

在反反爬的处理上：采用了IP代理和随机UA以及referer进行request\_headers的伪装，在爬取过程中进行sleep，进一步模拟人类行为。

②使用了多个第三方库减小工作量：xlrd，xlwt，PyQt5，functools，bs4，fake\_useragent等等。

③进行了良好的数据结构建模，采用hash表进一步提高查找速率。

④GUI设计采取简约风格，自行探索新工具，利用QTdesigner进行前端设计，而不是直接码代码。

⑤支持多种方式的查找，支持模糊查找，精确查找。

⑥在用户进行了较为细致的UI设计，提高了用户使用舒适度

⑦在搜索过程和信息爬取的过程中提供了异常处理，提高用户友好性

⑧采用excel进行信息存取，信息有序化，方便查找与准确提取信息

⑨B+树的建树过程中用到系统自带库bisect和collections，加快建树过程、查找过程

⑩UI采取界面组件设计与界面逻辑分离的形式，在进行维护时比较简单，结构清晰

五、实验总结

（1）在实验中遇到了哪些问题？是如何解决的？

1. 不知道怎么把网页上的信息保存下来

学习多个第三方库进行html解析，再进行正则表达式匹配。

1. 不知道怎么保存信息和读出信息

学习第三方库进行xlwt，xlrd等进行操作。

1. 不知道怎么反反爬

更换IP和UA深层伪造请求头；进一步模拟用户行为

1. 不知道B+树是干嘛的，怎么写，怎么用

复习PPt，在csdn等网站搜索教程

1. 不知道怎么设计GUI

采用PyQt5，qtdesigner进行前端设计，学习相应库函数的用法

1. 使用过程中程序经常崩溃

学习异常处理

（2）请简要地总结一下自己在数据结构课程中各方面的收获。

①最重要的是，打开了算法和数据结构的大门，了解了很多从来没听过没见过的数据结构和算法。

其次对于某些比较重要，比较简单的算法有了一定的理解，能够进行一些简单的应用。

②学会了python的一些重要的语法和一些第三方库的使用。

③知道了很多以前不知道的概念，扩充了自己的见识。

④在上课和写大作业的过程中学会了通过各种途径检索资料，学会课程之外或者课程中没弄懂的知识。

（3）你对课程的教学、实验等环节有没有自己的建议？（比如课程知识点安排、实验题目难易等等）

教学：希望老师能更加通俗易懂地讲解相关理论知识（本人太菜了~~）很多时候听不太懂。而一些简单的、扩展性的东西可以适当少讲一点。

实验：难度适中，不管什么题debug之后还是能做出来的（助教也进行了积极的解答~~~）可以在课下多布置一些拓展性的作业，学习相关第三方库。