大作业报告

小组成员：胡锦毅 林晓恩 梁念宁

一、 大作业架构概述

我们组的项目是信息抓取系统，核心是一个爬虫。这个程序中主要采用了适配器模式，工厂模式和模板方法。最主要的类有爬虫类，事物类，数据类，脚本解析类和核心控制类以及数据库类，另外数据类和事物类分别有对应的工厂类。主程序中部分代码如下：

```
1.      Parser* mainParser = new Parser;
2.      mainParser->solve_info();
3.      int len = mainParser->get_type().size();
4.      DataBase* database = new DataBase(*mainParser);
5.      Control* spider_control;
```

首先申明一个脚本解析类 Parser 的指针 mainParser，调用 Parser 类的成员函数 solve_info( )。该函数可以获取用户输入的信息来配置爬虫。输入的信息包括用户希望爬去的网站，希望获取电视剧还是电影，希望获取的信息类型以及当前文件保存的路径。之后声明一个数据库类 Database 的指针 database 用来将信息存入数据库中。之后在声明一个 Control 类指针，control 类指针根据网站的类型会来实际实现。Control 类将会对整个爬去过程的接口进行了封装，对外只暴露了一个接口就可以实现爬取过程。

二、 具体实现分析

(1)爬虫类 baseSpider

爬虫类主要负责网页 HTML 的下载和解析。下载 HTML 的函数是 downloadhtml( )，函数中主要使用了 curl 库中的函数。解析我们借鉴正则表达式的思想使用自己的 字符串匹配方法获取 HTML 中的特定信息。每一个网站的一类信息会设定一种特定的 match_symbol。这个字符串作为数据成员存在爬虫基类的子类中。匹配的函数是 Match，用来匹配特定的信息。爬虫类对外暴露的接口是 analyse_info ( )接口，函数接受 parser 类的一个引用获取用户要爬取的信息，针对一个 HTML 爬取一部电影或电视剧特定信息。最后将爬取的信息进行整合构成一个完整的事物，调用工厂类的函数 createProduct 接口创建事物类对象。对象会包含爬虫中爬取一个 HTML 文件所获取的信息。具体的匹配方法解释会附在最后。

(2)数据类 baseData

数据类是各种数据的基类，各种具体的数据类都继承这个类。这个类重要有 type 和 value 两个数据成员用来记录这个数据的类型和具体值。对外有 get_type( )和 get_value( )两个接口用来获取数据类型和数据的值。Datafactory 类用来创建数据类对象，根据数据类型创建一个特定类型的数据类对象。

(3)事物类 baseObject

事物类是一个包含信息的电影或电视剧类型。事物类中会有一个 vector<baseObject*>数据类型和 name 保存事物所含的信息和名字。对外暴露的接口有 saveobject 用来保存对象，printObject 用来打印数据数据的信息。与数据类类似，对象类也有一个工厂类用来创建对象。

(4)控制类 Control

Control 类的功能是用于控制爬虫进行下载 html，分析 html 并拓展 url 的一系列操作。它封装了爬虫提供的一系列接口，相当于一个爬虫类的适配器，使得外部操作变得简单一些。

先简单介绍一下我们的爬虫是如何拓展 html 的（限于豆瓣和 IMDb）。我们发现在很多类似的网页中都存在这样一个栏目：猜你喜欢，或是类似的推荐。这些电影的 url 是可以在 html 中提取出来的，因此在下载了一个 html 之后，可以从中获取更多的电影的 url。这让我们得以使用宽搜的方式进行 url 的拓展，进而一步步拓展我们的数据库。

Control 基类中含有的数据对象主要是一个爬虫指针，还有

Control 基类在初始化的时候接受三个参数，分别是存放有初始 url 的网页，记录上次读取位置（一个 int 类型的数），并传入存有用户输入信息的 parser 对象的引用。

在初始化 Control 基类对象的时候，完成了对其中上次的宽搜队列与已搜过网页 url 的集合的恢复。

⑸数据库类 Database

数据类使用的是 sqlite3 数据库，该类封装了接口 useDatabase( )用来打开数据库和创建数据表以及 insertData( )用来插入数据。其中数据库类可在程序源码的目录下创建出数据库。

三、 设计模式

程序中主要使用了三种设计模式。第一种适配器模式主要体现在我们设定了一个 Control 类来控制爬虫的进程，baseSpider 的进口封装在这个类中。同时创建了 Database 类封装了 sqlite3 数据库的使用接口。

第二种工厂模式主要体现在设定了数据工厂和事物工厂来管理创建电影或电视剧的创建。

第三种模板方法主要体现在设定了爬虫基类，控制基类来设定接口，子类根据基类设定的接口来实现。

四、 可拓展性分析

该程序具有良好的可拓展性。对于新增加的网页，只需将增加的网页继承于 baseSpider，将这个网页的控制类继承于 Control 类即可。Control 类的子类可以通过网页的具体特点来设定合适的爬取方法。

五、 编辑环境

Mac 环境下在终端中打开到文件夹源码目录运行 make 即可。生成的数据库电影.db 可以通过数据库的浏览其查看，也可以通过 Qt 程序来查看。Qt 程序提供了查找的检索功能。由于路径问题 Qt 程序可能在不同电脑上无法运行。运行情况可以查看程序运行结果视频。

六、 分工

林晓恩：负责字符串匹配的研究和 Match 函数

梁念宁：Spider 类和 Control 类的书写

胡锦毅：程序架构的设计，事物类，数据类，数据库类等以及 Qt 的书写

Debug 工作由三人共同完成。

附：
Match 函数说明：
一.功能

这个函数的目标是实现：输入一个配置串，每当 html 中的一个子串能和配置串匹配成功，就返回一个在匹配过程中生成的结果串。

例如，假设 html 中有一个子串 :"电影 A 将在 7 月 5 日上映"，那么我们可以构造一个配置串，其含义为当匹配到"将在***上映"时，返回"***"部分作为结果串。这样，将 html 和配置串传入 Match 函数，我们就可以得到 A 电影上映时间"7 月 5 日"作为返回结果，同时该匹配串还可以通过与 B 电影的匹配得到 B 电影的上映时间等等。

二.函数由来

按照普通做法，在设置配置信息时，针对每一样信息都得写一个专门的函数来从 html 中获取，比较麻烦。但经过比较分析，大多数获取信息的过程都为 :匹配到某些特定字符(串) 之后，在其附近截取一段作为获取到的信息。针对这一特点，我们经过讨论，模仿正则表达式来设计规则，实现了一个能在大多数情况下较好地实现匹配信息的通用函数。

三.函数的优点

1.通过这个函数，当我们需要新添加一个网站时，我们添加的配置信息只需要大约 400B（甚至可以压缩至 200B）左右。
2.这个函数接口简单，只需传入 html 和配置串，直接返回匹配信息的结果串。
3.适用性广，设置配置串较灵活，可以适应多种网站、获取多类信息。

四.函数的具体设计说明：
配置串说明：
配置串由多个单位组成，每个单位有两种形式，
1、由`或~（选择这两个字符是因为其在 html 中几乎不出现）引出的几个字符按特定格式构成的特殊单位（具体用法会在下面说明）
2、其余情况由单个字符组成的普通单位
普通单位用法：完全相同的两个字符才能匹配
特殊单位的用法：
① 带有`的只有一种格式：
`A1`A2`，其中 A1 是不带`的串，A2 是只由数字组成的串（即一个数字）
这是一种简写，意思是把 A1 串重复 A2 遍，替换该特殊单位
② 带~的有多种格式，分为三类：
第一类是功能型的，就是匹配过该单位后实现一些功能
1.~%A1~%:只能在匹配串的开头使用，且 A1 是由数字组成的串，意思是最多只返回前 A1 个匹配成功生成的结果串，多余结果串直接忽略
2.~@:从该单位之后，配置串与 html 匹配的过程中，html 的内容将写入结果串
3.~!:从该单位之后，匹配串与 html 匹配的过程中，取消 html 内容的写入结果串
4.~L:将已写入结果串的最后一个字符删去
5.~~A1~~:在当前结果串的末尾写入 A1 串
第二类是单字符匹配型的，就是用来判断配置串当前单位与 html 是否匹配
1.~?:该字符可以与任意一个字符匹配

2.~N:该字符可以与任意一个数字字符匹配
3.~|A|B|C……:~后面可以加任意多（大于等于 1）个|X，意思是，该字符可以与这些字符中的任意一个匹配
第三类是多字符匹配型的，
以下 A1 均指不含第一、三类~型特殊字符的串
1.~*A1~*:此处开始可以任意匹配，直到匹配到能与 A1 串匹配停止
2.~*~!A1~*:此处开始可以任意匹配，直到匹配到能与 A1 串匹配停止，其中 A1 串不写入结果串，即在 A1 匹配开始前调用~!
3.~*A1~/A2~*:此处开始可以任意匹配，直到匹配到能与 A2 串匹配停止，其过程中匹配到的 A1 串不写入结果串
4.~*~!A1~/A2~*:2 和 3 的结合（匹配至 A2 停止，其中的 A1 不写入）
举些例子：
html:abcde
匹配串:bc，则匹配成功一次，结果串为空
html:ababc
匹配串:~@ab~!，则匹配成功两次，结果串为 ab,ab
html:abababc
匹配串:~@`ab`3`~!，则匹配串先被展开成~@ababab~!，匹配成功一次，结果串为 abababc
html:233ab333ab345
匹配串:~@2~*~!ab~/45~*，则匹配为：从 2 开始到 45 结束，ab 串和结束判断串 45 不写入，匹配了 233ab333ab345，结果串为 2333333