**面向对象程序设计（C++）**

**课程设计报告**

题目：图书管理系统的设计与实现

专业：电子信息类

设计日期：2022年4月23日 至2022年5月23日

小组成员人数：1 人

小组成员名单：王磊

组长 班级： 电2103 学号：20216234 姓名：王磊 成绩：

**报告**撰写的内容与要求

1. 项目简介：介绍本次课程设计选题的目的、意义、任务概况、本人在项目中的分工等内容。
2. 项目内容：系统的设计与实现的全面描述，介绍系统整体结构、系统框架图、UML类图、程序流程图、重点难点分析及解决方案、调试难点及解决方法、系统交互界面及结果展示等。本部分内容应以记叙或[白描手法](http://www.so.com/s?q=%E7%99%BD%E6%8F%8F%E6%89%8B%E6%B3%95&ie=utf-8&src=wenda_link" \t "_blank)为基调，在完整叙述的基础上，对自己认为有重要意义或需要研究解决的问题进行重点叙述，其它内容则可简述。
3. 总结或体会：对课设效果进行综合评价，着重介绍自身的收获与体会，内容较多时可列出[小标题](http://www.so.com/s?q=%E5%B0%8F%E6%A0%87%E9%A2%98&ie=utf-8&src=wenda_link" \t "_blank)，逐一列举。总结或体会的最后部分，应针对实习中发现的自身不足，简要地提出今后学习，努力的方向。
4. 报告正文一律采用计算机排版、A4纸**双面**打印，正文字体为**小四号**宋体，**1.35倍**行距，正文页数不少于10页、不多于20页(其中代码不超过3页，**中文字数**不少于5000字)。要求语句通顺、论述严谨、规范、正确。
5. 请注意封面页、扉页、评语页等的打印及装订顺序。分别为①封面页（其背面为扉页“实习报告的内容与要求”）、②目录、③正文、④其他附件（如有）。

目录

[一． 项目简介 4](#_Toc33138184)

[1.1项目内容 4](#_Toc33138185)

[1.2文件与执行环境 4](#_Toc33138186)

[1.2.1电子文档打包文件名及文件列表 4](#_Toc33138187)

[1.2.2编译执行环境与运行步骤 5](#_Toc33138188)

[1.3任务分工 5](#_Toc33138189)

[二．项目内容 6](#_Toc33138190)

[2.1系统的整体架构 6](#_Toc33138191)

[2.2 UML类图 7](#_Toc33138192)

[2.3系统的整体设计 9](#_Toc33138193)

[2.4功能模块设计 9](#_Toc33138194)

[2.5重点难点分析及解决方案 1](#_Toc33138195)0

[2.5.1重难点分析 1](#_Toc33138196)0

[2.5.2解决办法 1](#_Toc33138197)0

[2.6调试难点及解决方法 1](#_Toc33138198)1

[2.6.1 调试难点 1](#_Toc33138199)1

[2.6.2解决方案 1](#_Toc33138200)1

[2.7结果展示 1](#_Toc33138201)1

[三．总结与体会 1](#_Toc33138202)5

# 项目简介

## 1.1项目内容

**项目简介：**

在现代，知识是不可或缺的，而阅读这种便捷的方式受到了人们的青睐，相伴而生的图书的管理也十分重要，这将影响图书是否能有效的利用和循环使用。

本系统利用C++的知识实现图书管理系统。

大致有以下功能：

1.图书的录入，删除，多种排序，查找等功能。

2.读者的注册（添加），注销（删除），排序，查找等功能。

3.借书，还书，显示借书情况，显示多种借还记录等功能。

目的是：通过类的设计来深入体会面向对象这一思想，理解C++万物皆对象的含义，体会封装的好处；加强文件的关联操作，熟悉掌握类与结构体的定义应该放在头文件，以及类成员函数的类内声明，类外实现；熟练运用基础知识。

**主要功能：**

1. 管理员登录（管理员的操作，对图书、读者、日志和借还系统进行整体和局部操作）
2. 用户登录（对自身进行局部操作，对图书进行部分查找等操作）
3. 用户注册
4. 退出系统

……

## 1.2文件与执行环境

### 1.2.1电子文档打包文件名及文件列表

图书管理系统.vcxproj//工程文件

图书管理系统.vcxproj.filters//筛选器文件，指明头文件和源文件

UsersManage.h //用户类的头文件，管理用户数据

BooksManage.h //图书类的头文件，管理图书数据

Notes.h //日志类的头文件，管理日志数据

LendSystem.h /\*借书还书系统类的头文件，处理正在借书的用户与图书结点信息，并反馈回用户类与图书类\*/

mainfun.h/\*主函数头文件，内含一个抽象基类与对应的两个派生类，用于实现主界面的统一接口\*/

UsersManage.c，BooksManage.c，Notes.c，LendSystem.c，mainfun.c/\*为对应的函数实现\*/

UsersManage.txt，BooksManage,txt，Notes.txt，LendSystem.txt//为对应类的数据

main.c//是主函数main所在文件

### 1.2.2编译执行环境与运行步骤

编译环境：VS2022

运行环境：Windows环境

数据文件：以上四个.txt文件

运行命令：双击图书管理系统.vcxproj文件

运行步骤：直接运行

使用方法：打开工程文件，运行即可

## 1.3任务分工

**任务概况：**

本次任务艰巨需要在一定的时间内完成系统框架的拟定以及所有代码的编写。

1. 主菜单的编写（先进行对象实例化和初始化，读文件）
2. 管理员登录，用户登录接口函数的编写
3. 用户注册界面的编写
4. 用户类的编写（包括文件操作，以下除时间类都有文件操作）
5. 图书类的编写
6. 日志类的编写
7. 借还类（包括时间类）的编写
8. 管理员系统中完成用户，图书，日志，借还对象的交流
9. 读者系统中完成用户，图书，日志，借还对象的交流
10. 退出系统时四个对象的文件交互，链表的销毁，堆区数据释放和指针置空

**任务分工：**

项目的任务分工如表一所示。

表一 任务分工表

|  |  |
| --- | --- |
| **姓名** | **完成内容** |
| 王磊 | 完成程序总体结构的设计与功能的分配，完成所有程序，实现用户类Users，图书类Books，日志类Notes，借书还书系统管理类List，主界面统一接口类Base，Reader，Manager。List的结点中的成员类Time。并完成对象的相互作用。 |

# 二．项目内容

## 2.1系统的整体架构

图书管理系统

1.在主界面出现“管理员登录”，“读者登录”，“读者注册”，“退出系统”选择，输入对应数字即进入对应子系统，输入非法数据会报错。

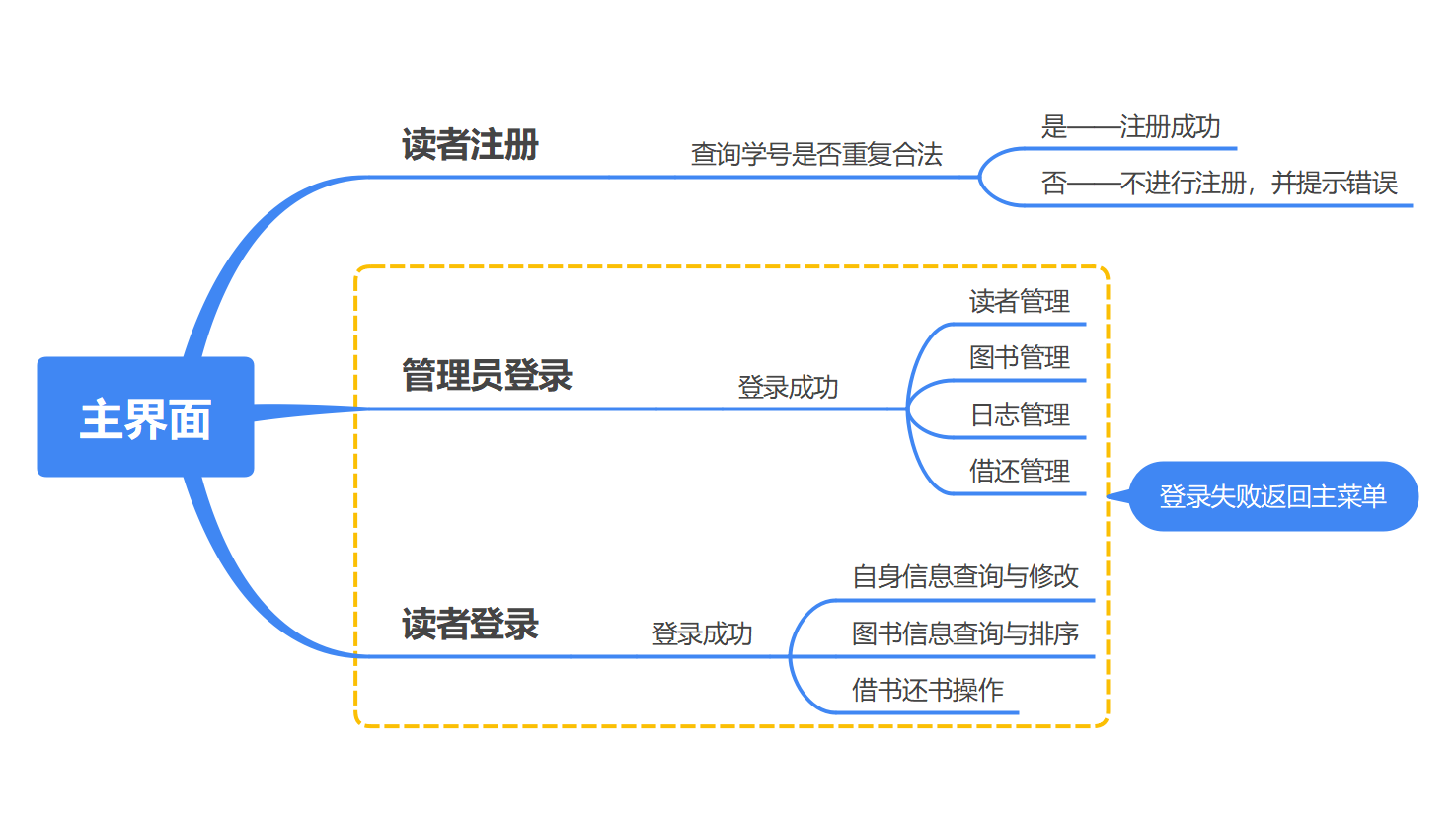
2.读者注册界面，输入学号完成注册，系统会检验学号是否已被注册，以及学号是否合法（默认8位数字），给予一定提示。

3. 管理员登录后方可进入管理员界面，进入会要求先输入时间（用于刷新读者的借书时间，这里本应该从电脑抓取数据，但是为了测试方便，采用输入的方式，这里要求必须输入正确的时间），然后执行对应功能。大致分为读者管理，图书管理，日志管理和借还管理，里面有很多小功能，供管理员使用。

4. 读者登录后进入读者界面，可完成查询书籍（按不同方式均可），借书，还书，查看借书信息，查看借书日志，图书排序，修改密码等功能。

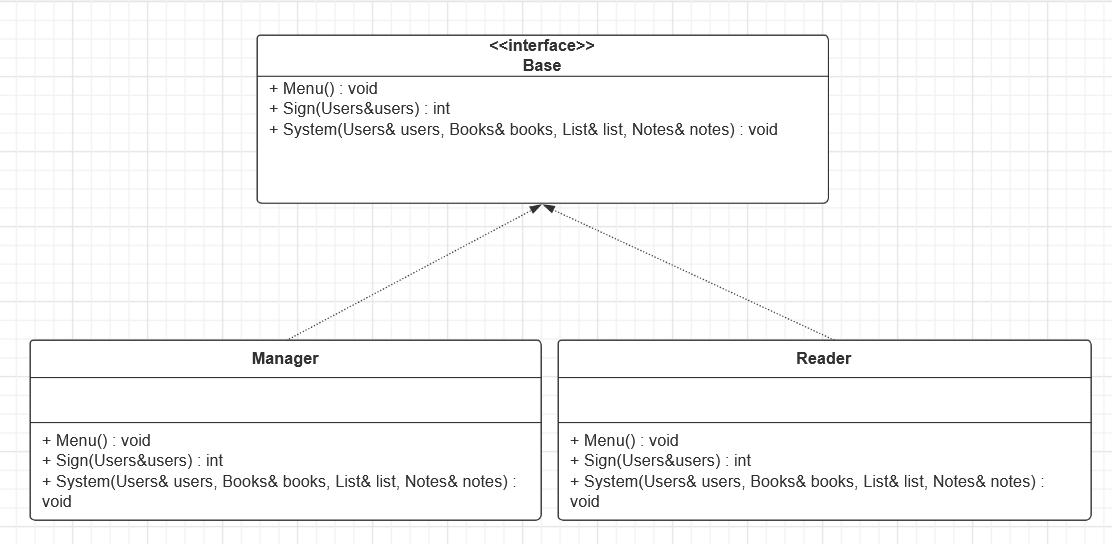
5.退出系统前完成对象的文件交互，堆区数据的释放，指针的置空，链表的销毁。

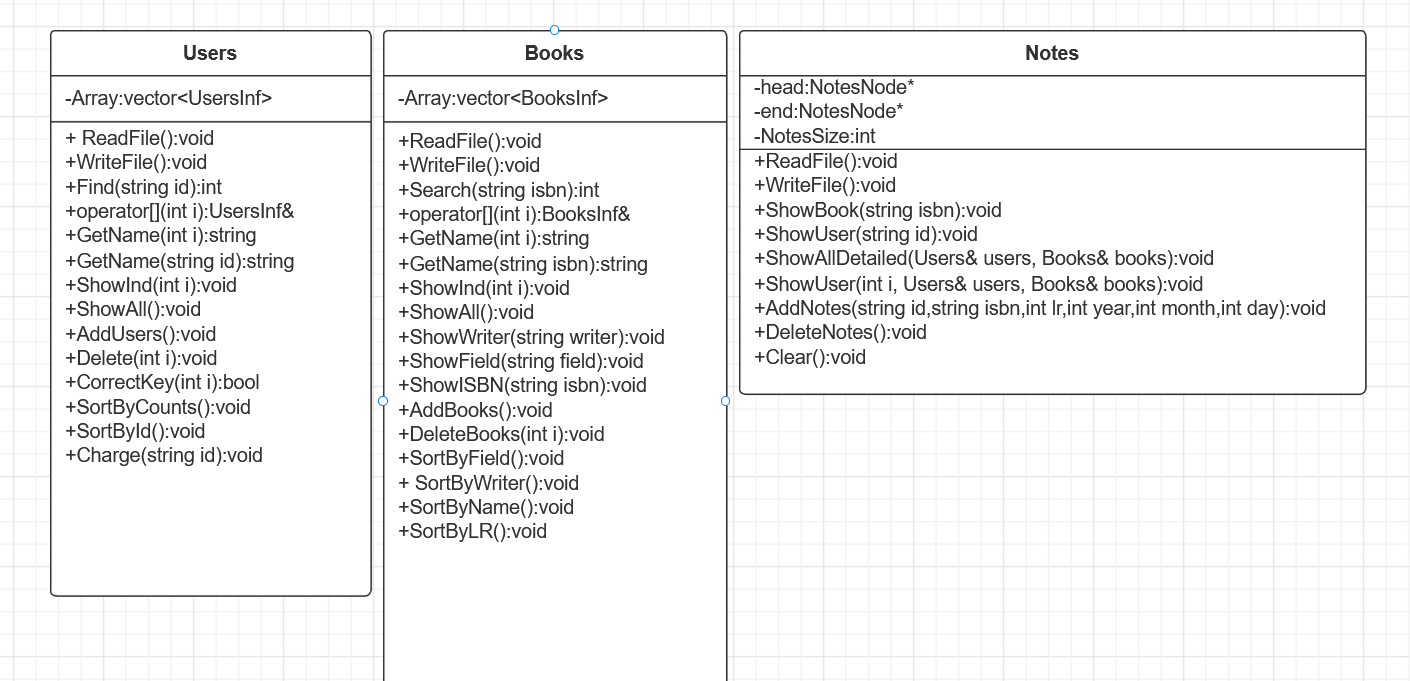
系统整体结构如图一所示（主界面还有一个退出系统未显示）：

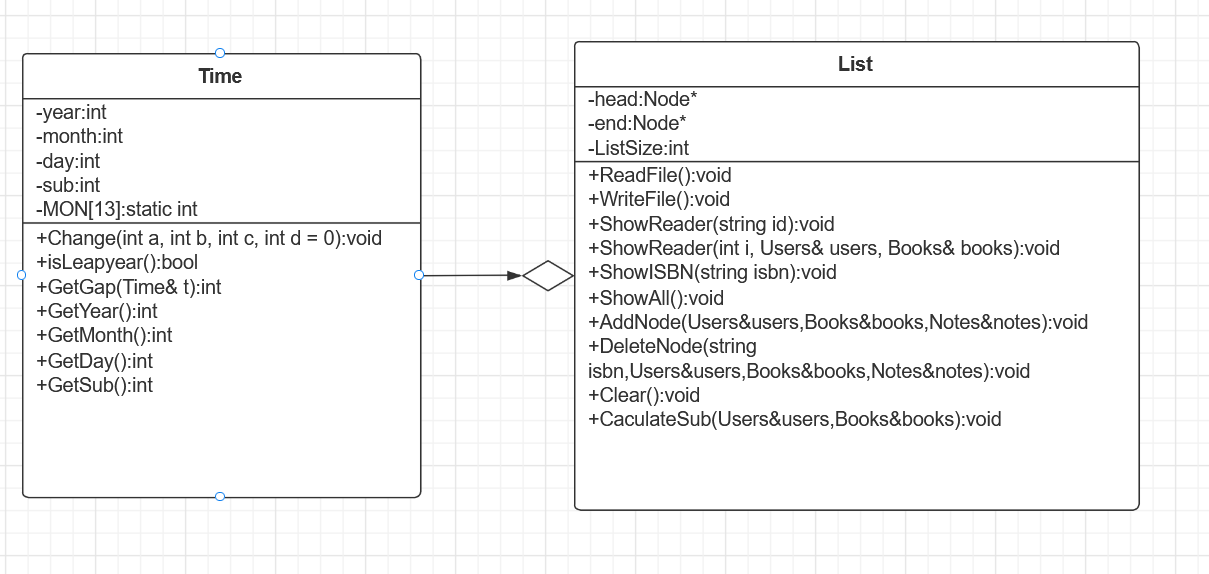


图一 系统整体结构图

## 2.2 UML类图



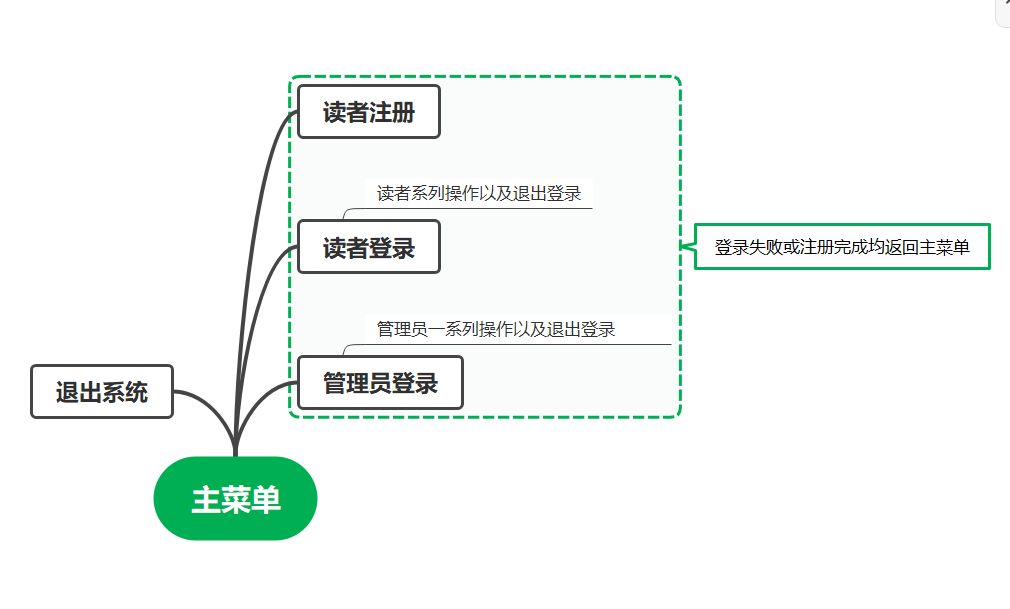




图二 UML类图

## 2.3系统的整体设计

系统的整体程序流程图如图三所示。



图三 系统整体程序流程图

## 2.4功能模块设计

系统中使用的数据表如表二所示。包括如下字段，其意义为……。

表二 数据表

|  |  |
| --- | --- |
| **字段** | **关于该字段描述** |
| Users::vector<UsersInf>Array | Users类中的vector容器，用以储存用户数据，是系统的基础，其个体数据为自定义结构体UsersInf |
| Books::vector<BooksInf>Array | Books类中的vector容器，用以储存书籍信息，和Users::vector<UsersInf>Array一样重要，个体数据为自定义数据类型BooksInf |
| Notes::head | Notes类中封装的两个指针head,end和一个整型变量NotesSize,用以对一个链表进行控制，该链表记录了用户借书还书的记录，供使用者查询 |
| Time::MON[13] | 时间类Time的静态成员数组变量，记录该年12个月的天数，并根据闰年判断来通过函数更新，全部对象共用 |
| List::head | List类中的链表控制，与Notes中的类似，但List这个要复杂的多。该链表用于建立整个借书系统，储存当前正在借书的结点信息。结点构成为：一个用户id，一个vector<SubLend>Arr数组，由于一个用户可以借多本书，为了方便，将一个用户和他借的多本书放在一个结点，书放在vector数组里面，同时每本书和一个借书时间绑定，因此将书的唯一确定方式ISBN号和时间放在一个结构体内。 |

## 2.5重点难点分析及解决方案

### 2.5.1重难点分析

1.如何实现界面的切换

2.如何保存图书与用户数据

3.如何保存日志和当前借书信息数据

4.借书还书调用后日志、读者、图书信息如何回馈修改

### 2.5.2解决办法

1. 如何实现界面的切换：

主界面主菜单由函数调用完成，之后采用接口类，Base、Manager、Reader三个类，通过基类Base指针的不同指向调用不同的函数，进入到子系统（管理函数）。

进入到子系统后调用对象的菜单函数。返回主菜单即退出该子系统管理函数即可。

在界面的切换会清屏，保证界面的美观。

1. 如何保存图书和用户数据

用户和图书数据由于数量变化不大也相对不是很频繁，故采用数组储存。为了灵活的储存数据，采用动态数组vector容器储存，将数据开在堆区，数组元素为用户自定义结构体。读文件时将数组capacity预设为Name.Array.reserve(size+100)，此外，添加若达到最大，重新预设空间加100.这样可以避免频繁的伸缩空间。

3.如何保存日志和当前借书信息数据

日志是随当前借书信息变化的，而借书信息的变化是非常频繁的，因此两个都采用链表储存。注意到同一时间下，一个用户可以借多本书，一本书只能被一个用户借。因此将一个用户和多本图书放在一个结点。图书采用vector数组储存。由于每一本图书都和一个时间绑定，故将时间对象和书籍编号放在一个结构体里面，以此作为vector的数组元素

4.借书还书调用后日志、读者、图书信息如何回馈修改

图书信息有成员是“图书是否被借出”、“借书者ID”，而用户信息有“借书数”，日志依赖于借书还书，故借书还书时需要将信息回馈到对应地方。将信息封装为用户类（读者类）、图书类、日志类、借书管理类，在借书管理类的借书、还书成员函数传入读者。图书、日志对象作为参数，借书还书完成后直接通过对象修改对应信息。

## 2.6调试难点及解决方法

### 2.6.1 调试难点

由于对个对象相互传递信息，调试会涉及到多个对象。

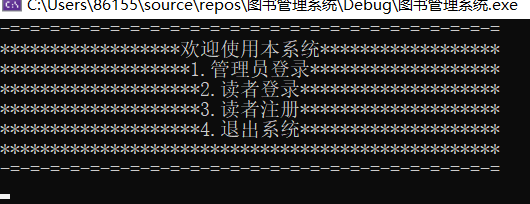
本次C++课程设计调试的难点在于测试时需要几个对象同时参与，所以需要多个类同时书写，而当有对象还没写好时，测试和调试便难以进行。

### 2.6.2解决方案

由于程序不完整导致无法进行测试和调试的问题，通过先写基本框架来解决，先定义为空实现，或者输出一段提示话语，先测试可行性。优先完成用户类和图书类，这两个是基本信息，是程序的根本，完成后再完成借书还书，同时完成一个测试调试一个功能。

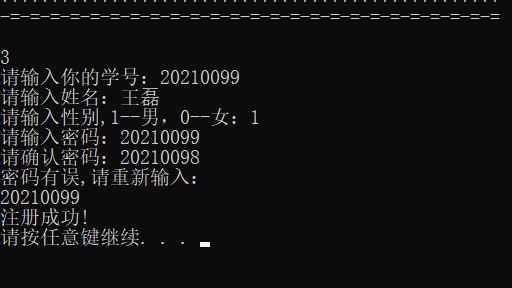
## 2.7结果展示

主菜单如图四所示。为了确保安全性以及同时只能登录一个账号，故不开放管理员注册功能，直接添加，登录之后可以改密码。



图四 系统主界面

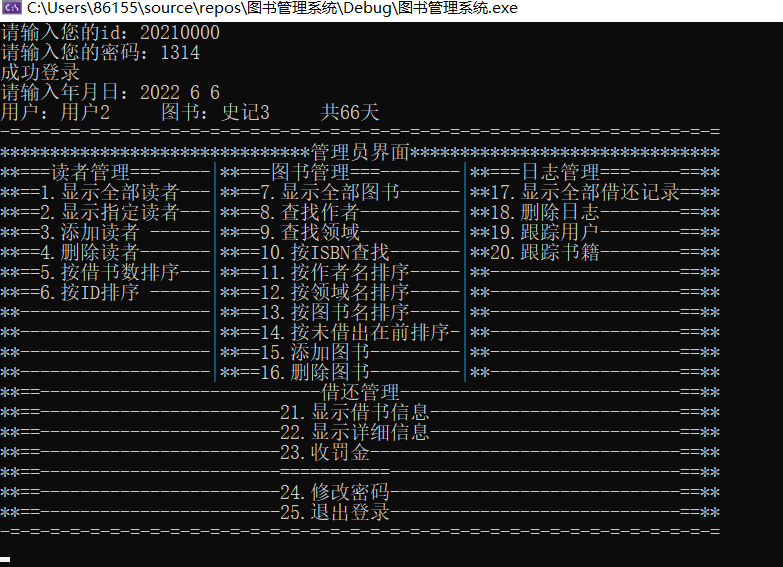
图五为读者注册，注册时会检测学号是否符合要求（默认8位数字），以及是否重复，密码确认会提示可能的错误，出现上述错误不予注册，注册完成或失败都会返回主菜单。



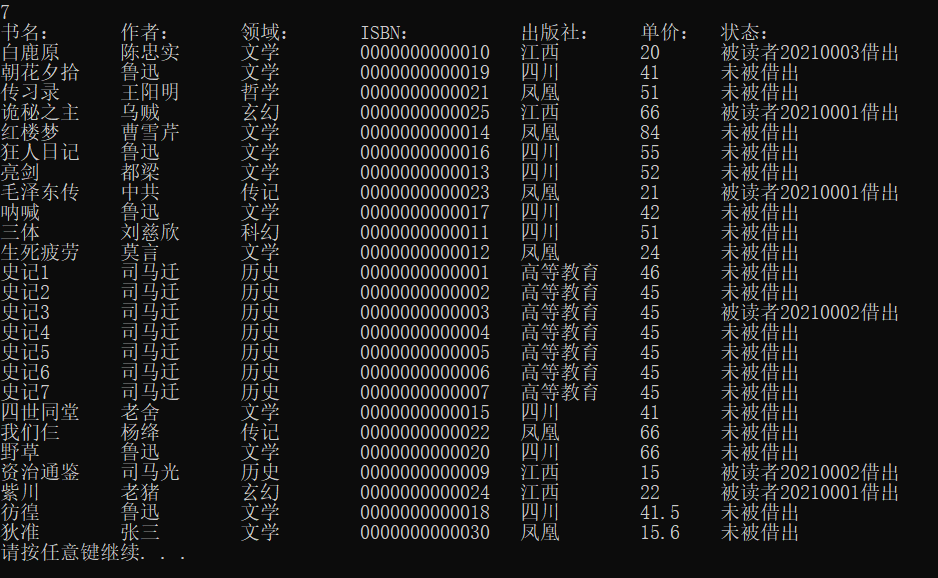
图五 读者注册

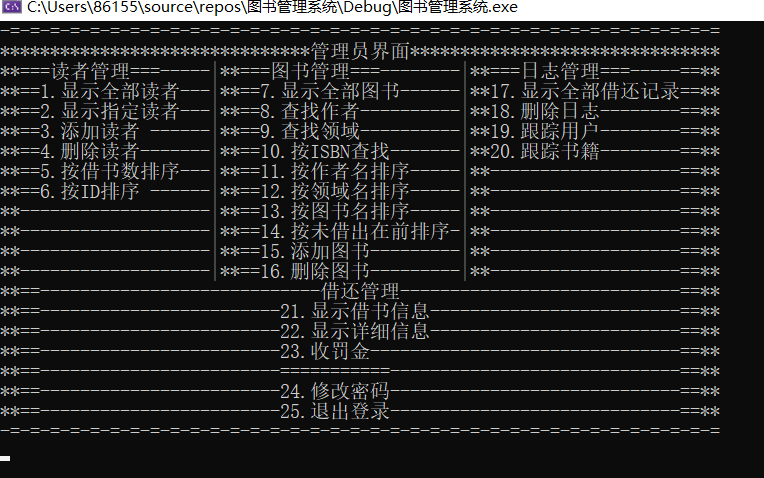
图六为管理员登录界面，管理员登录输错密码则返回主界面，提示“账号或密码错误”，保证安全性。成功则进入界面。

登录成功后先输入时间，更新借还系统的借书时间（这里为了保证安全性，应该从电脑抓取时间，但是为了测试，选择手动输入）。并显示超时未还书的结点信息。



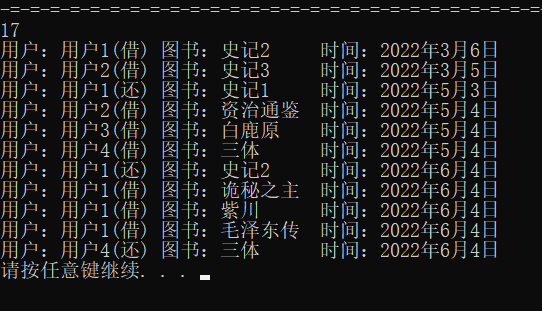
图六 管理员登录界面

如图七所示，显示全部图书后自动刷新页面，保持界面整洁

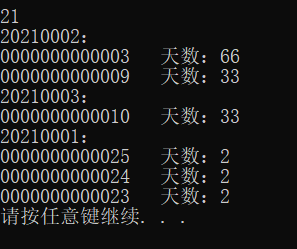


图七 显示图书与刷新

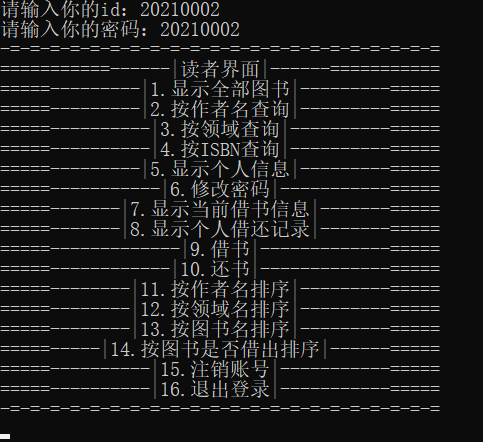
以下为其余部分界面展示，如图八。



借书信息，这里选择用户ID和图书ISBN，因为具有唯一性，而上面日志并不是非常重要，可以显示名称



读者登录，限于篇幅，功能暂不展示



图八 部分其余界面

# 三．总结与体会

#### 王磊

**收获和体会**：

本次C++课程设计我选择了图书管理系统，是因为之前C语言课设是运动场预约管理系统，两个比较相似但又有不同之处，图书管理更偏向数据的处理。而且我也想借这两个来对比看一下C++在C语言上做出的改变——面向对象。

如愿以偿，我对文件的关联理解更进一步，对顺序表的操作更加熟悉与灵活，对面向对象的思想也有了一定理解。

尤其是对链表的处理，在C语言中如果通过函数对链表进行尾结点加长，需要将end指针的地址，也就是二级指针传入函数，因为end指针的值要改变。当然还有一种方式就是传入一级指针但是要把改变的end指针值通过返回值返回赋值。在C++中便不用这么麻烦，只需创建一个类来封装head与end指针，以及对应的函数操作，当然也可以加一个size的成员。这样，链表和对应的操作就成为了一个统一的整体，只能通过对象的成员函数对成员直接进行操作，这就是封装。

C++面向对象的思想，我理解为C++一切属性、一切操作均可以是对象。但是有些事没有什么实际意义的，或者效果不好的。我认为相似属性或者可以相互作用的属性应该放在一起，抽象为类，这里抽象是很重要的，同时，为了保证封闭性，应该尽可能避免public成员，友元函数也要慎用，当然，类封闭了但是也要与外界交流以及自身的改变，所以，将成员函数也放在里面。

总结就是，面向对象是把有用的相似属性放在一起，抽象为封闭的类，同时把成员函数封装在里面。类实例化对象之后，一个个对象调用成员函数完成自身的改变，以及对象与对象之间的信息交流——也就是相互作用。

利用这种思想，我把整个系统抽象为几个相互独立类，系统即为类的对象的信息交流。不仅简化了系统与操作，而且方便后期的维护与拓展。

对于头文件的包含关系，我尽可能避免“你中有我，我中有你”的情况发生，即让头文件形成一条单向的链，当然可以有交叉。类与结构体的定义放在头文件，非静态成员变量分配空间要在对应的.cpp文件进行，避免头文件多次引用而造成重复定义

**不足之处：**

困难与挫折是不可避免的，在编写过程中还是发现了一些自身的问题。不够细心，有一次在测试类的写文件函数时，发现始终没有写进去，看了半天代码都没找到问题，结果调试的时候发现是调用时调用成了读文件函数。还有就是变量名的选取，基本上到位了，但有时候却不能很好的代表变量的真实意义。其他就是数据结构没有用到位，算法还需要优化。

**改进与建议：**

本次的c++课设我学到了要善于利用调试，调试可以帮助发现变量名的变化，栈的情况，函数是否被调用，程序执行的流程等等，以下是总结的建议：

1.继续加强类的抽象思维，将属性与操作封装，简化系统，方便维护修改。

2.尽量用引用而不是指针，虽然引用的底层依然是指针，但是引用更安全。

3.灵活使用数据结构，以及STL模板，但是有些注意开销，比如双端链表list,图map等。

4.不管学什么知识点，一定要敲一敲代码才知道自己到底会不会。