北京邮电大学课程设计报告

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **课程设计**  **名称** | **数据结构课程设计** | | **学 院** | **计算机学院** | **指导教师** |  |
| **班 级** | **班内序号** | **学 号** | | **学生姓名** | **成绩** | |
| **2015211310** | **07** | **2015211405** | | **郑键楠** |  | |
| **2015211310** | **18** | **2015211417** | | **陈俊贤** |  | |
| **2015211310** | **24** | **2015211423** | | **袁昕钰** |  | |
| **课**  **程**  **设**  **计**  **内**  **容** | 简要介绍课程设计的主要内容，包括课程设计教学目的、基本内容、实验方法和团队分工等  该课程要求设计模拟图书馆管理系统，针对不同的身份（管理员、教室、学生等）设置不同的权限和借书数量。对于管理员而言，完成对图书的购入、编码、统计以及对图书信息的存储与管理和对读者信息的存储与管理等。对于读者而言，能够实现注册用户信息，并实现查询、借阅、归还、预约图书等功能。通过该系统最终实现图书馆日常工作的高效、有序地运行。  在该课程设计中，主要分工如下：  郑键楠：图书管理模块的设计、借还预约模块的设计、日志模块的设计以及文档的书写  陈俊贤：图形化的设计以及文档的书写  袁昕钰：读者管理模块的设计、借还预约模块的设计以及文档的书写 | | | | | |
| **学生课程设计报告**（附页） |  | | | | | |
| **课**  **程**  **设**  **计**  **成**  **绩**  **评**  **定** | 遵照实践教学大纲并根据以下四方面综合评定成绩：  1、课程设计目的任务明确，选题符合教学要求，份量及难易程度  2、团队分工是否恰当与合理  3、综合运用所学知识，提高分析问题、解决问题及实践动手能力的效果  4、是否认真、独立完成属于自己的课程设计内容，课程设计报告是否思路清晰、文字通顺、书写规范  **评语**:  **成绩**:  指导教师签名：  年 月 日 | | | | | |

数据结构

课程设计报告

设计题目：模拟图书馆管理系统的设计

# 课程设计目的

学校图书馆有很多图书，每个教师、学生有多种不同的借书权限、借书数量以及借书需求，同时图书管理员需要对图书的借入、借出、购入等实现统一的规划管理。为此，我们建立这个图书馆管理系统的目的是：方便实现图书管理员完成对图书的购入、编码、统计以及对图书信息的存储与管理和对读者信息的存储与管理。同时，方便读者完成对于图书的借出、还回、统计、查询、预约等功能，真正使得图书馆能够高效、有序地运行。

# 课程设计要求

建立图书管理管理系统，针对不同的身份（管理员、教室、学生等）设置不同的权限和借书数量。对于管理员而言，完成对图书的购入、编码、统计以及对图书信息的存储与管理和对读者信息的存储与管理等。对于读者而言，能够实现注册用户信息，并实现查询、借阅、归还、预约图书等功能。

# 具体任务

图书管理员需要完成的工作：图书的购入、编码、入库以及图书信息的统计与管理、读者信息的统计与管理。

读者需求：注册，查询（包括图书信息以及个人信息），借书、还书等。

基本功能描述：

1、完成图书信息的统计与管理，建立图书基本信息文件。具体内容包括： 每种图书的编号、书名、作者、出版社、简介、购入时间、购入数量、当前库中剩余量、借阅率等信息，同时在图书购入时根据图书信息自动生成图书编码。

2、完成读者信息的统计与管理，建立读者基本信息文件。具体内容包括：读者（分为教师、学生等）的姓名、工作证（学生证）号、所在学院、借书权限、借书数量等信息。

3、查询统计功能如下所示：

①、读者可根据图书的名称、编码、作者、出版社等信息查询图书的其他信息；也可根据图书的分类等信息查询图书的具体信息等，包括某图书库中尚存多少本。借书时填写借书单，还书时填写借书单还书项，并记录入日志文件，借书超期要罚款。

②、查询教师、学生已借出的图书情况，读者的借书诚信情况（是否有过超期的情况）等。

③、图书的借阅率等。

4、对于库中已借完的图书，读者可预约，待库中有还回时能提示通知该读者。

5、模拟系统以时间为轴向前推移，每10秒左右向前推进 一天（有键盘或鼠标操作时不计时）。

6、建立日志文件，对图书借阅情况进行记录。

# 算法设计思想及算法描述

## 需求分析

见本组需求分析文档。

## 概要设计

见本组概要设计文档。

## 详细设计

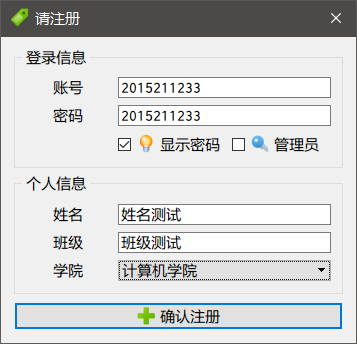
见本组详细设计文档。

# 调试分析

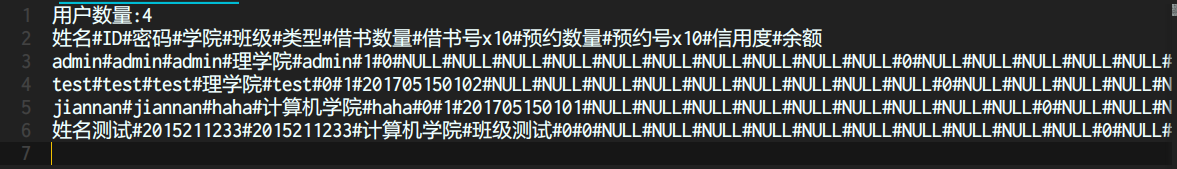
## 用户注册

* 普通用户注册：

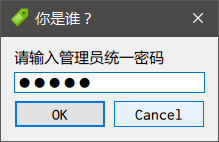
填写注册信息：



可以看见新用户已被追加到用户信息库(User-Info.txt)中：

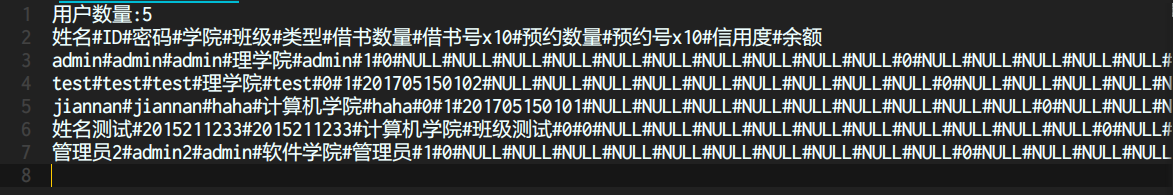


* 管理员用户注册：（勾选管理员选项并正确填写统一密码）





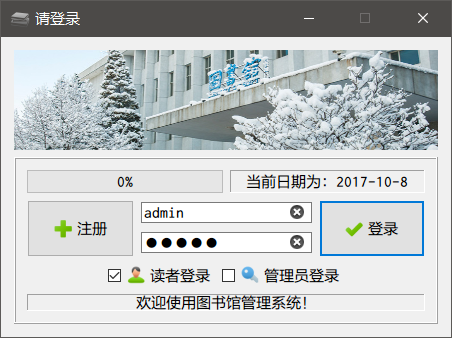
可以看见新管理员已被追加到用户信息库(User-Info.txt)中：



## 用户登录

* 管理员作为读者登录：

使用管理员账号，填写登录信息，勾选“读者登录”

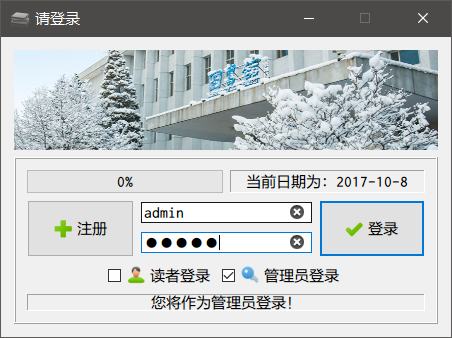


管理员admin作为读者登录成功：



* 管理员登录：

使用管理员账号，填写登录信息，勾选“管理员登录”

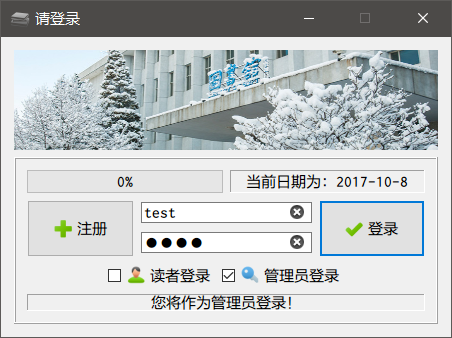


管理员admin作为管理员登录成功：

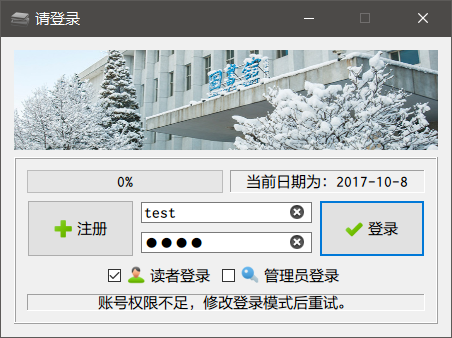


* 读者作为管理员登录：

test为读者用户。

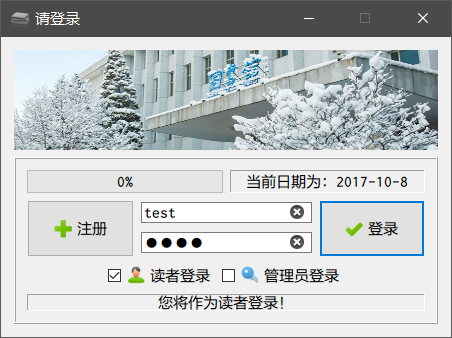


点击登录后，提示权限不足，自动修改登录模式：



* 读者作为读者登录：

账户同上

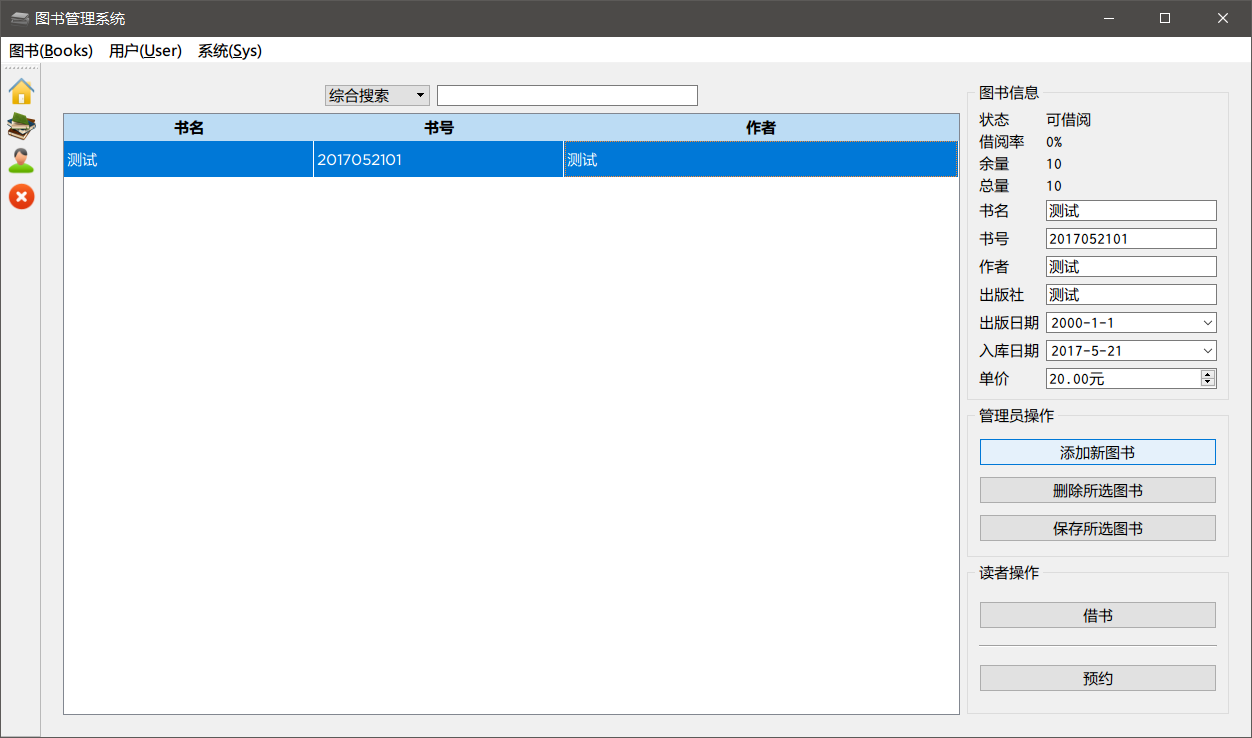


登录成功



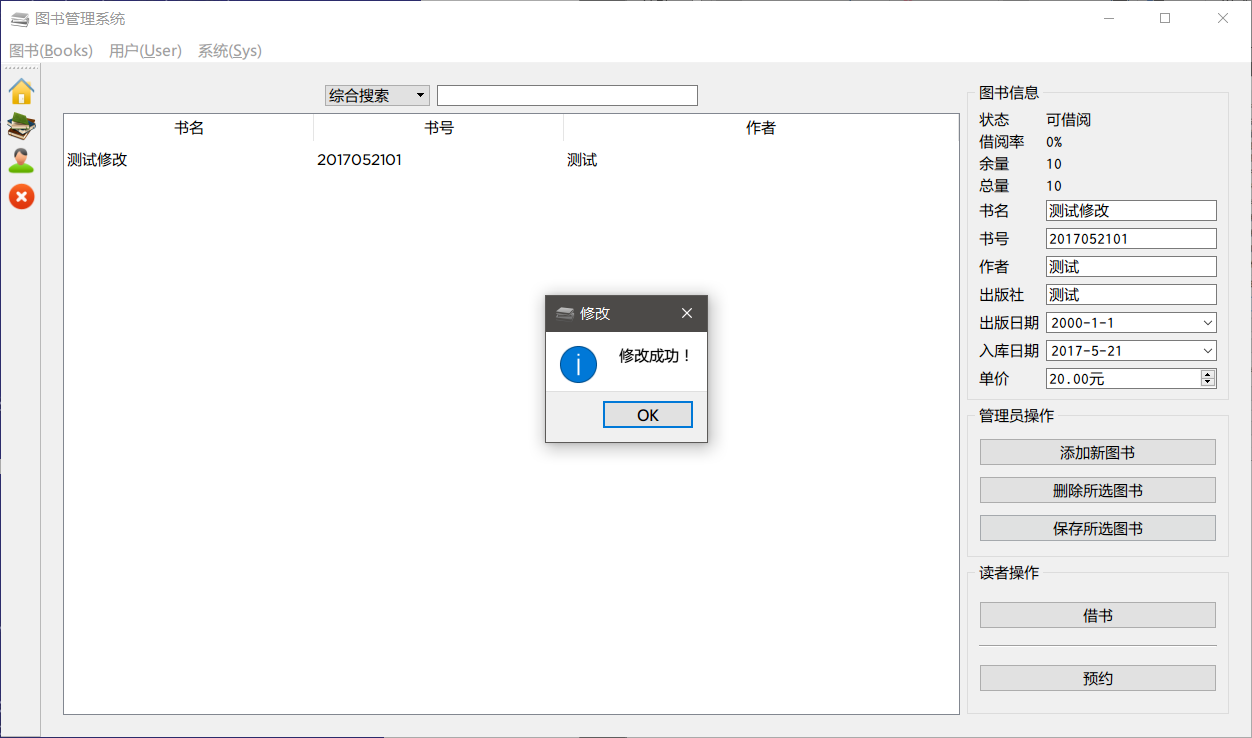
## 图书添加

作为管理员登录后，点击添加新图书，添加成功。



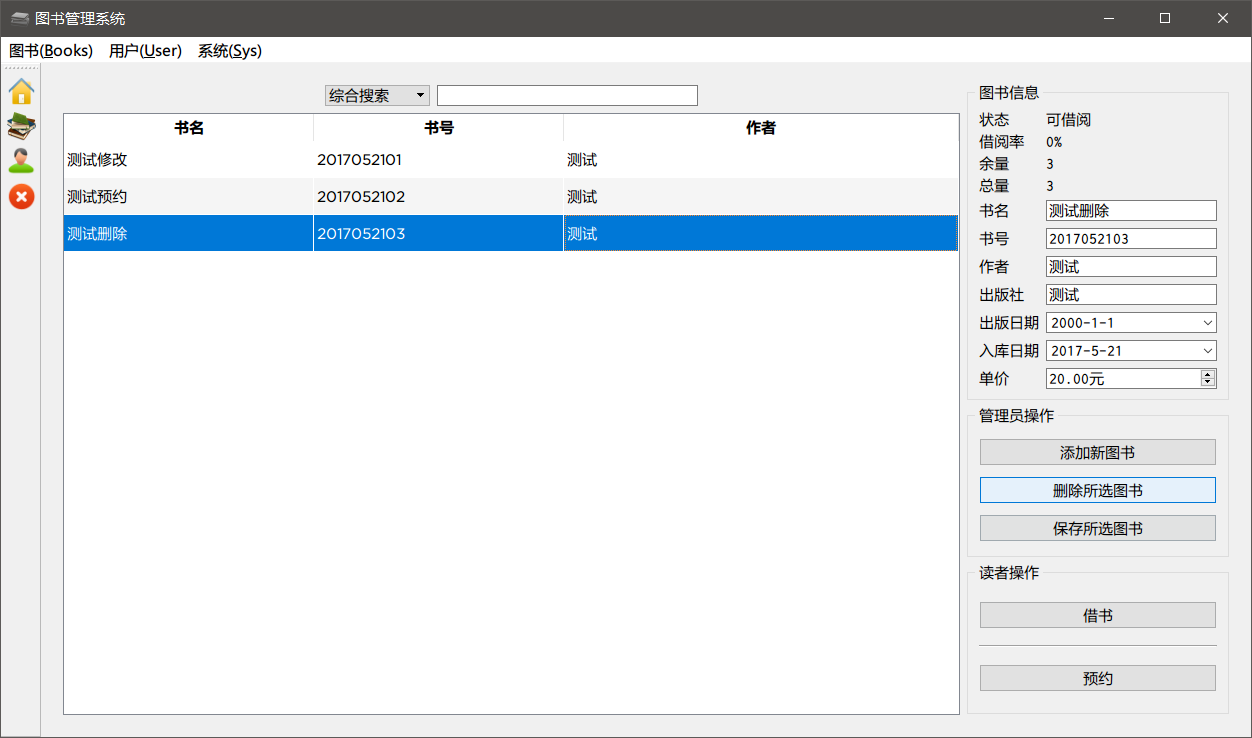
## 图书修改

点击保存所选图书，修改成功。

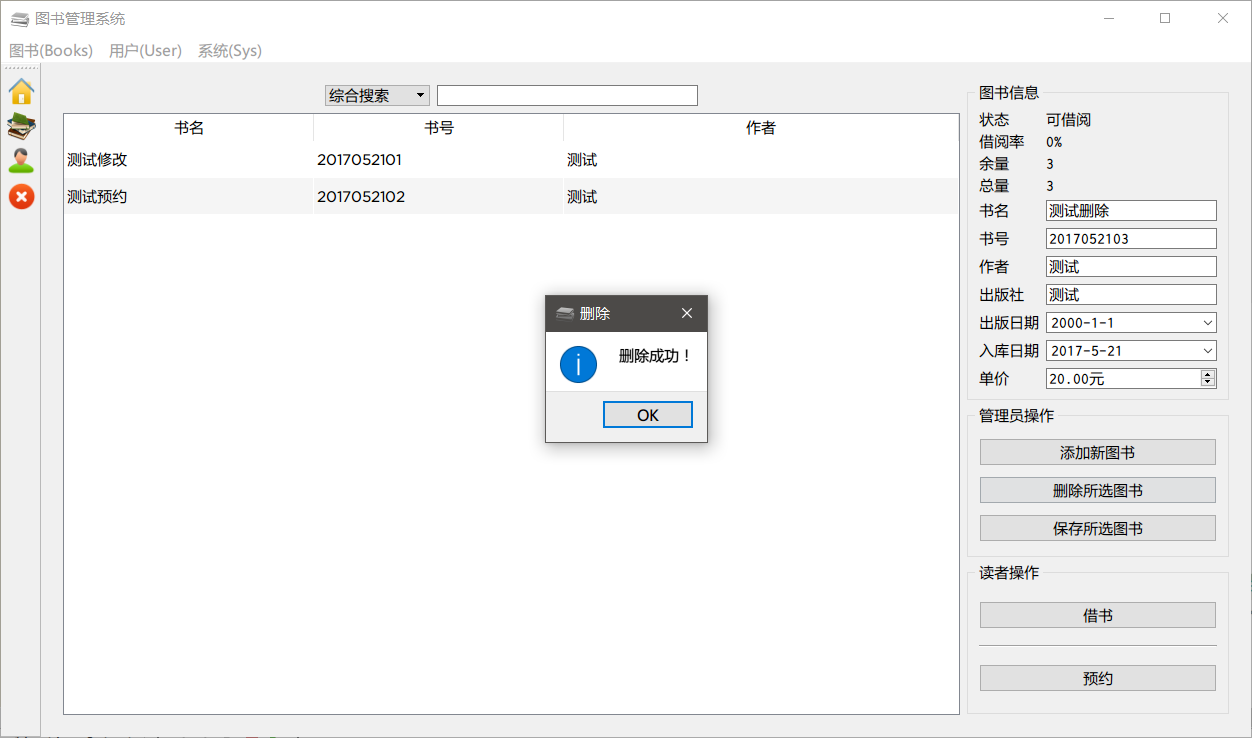


## 图书删除

选中要删除的图书：

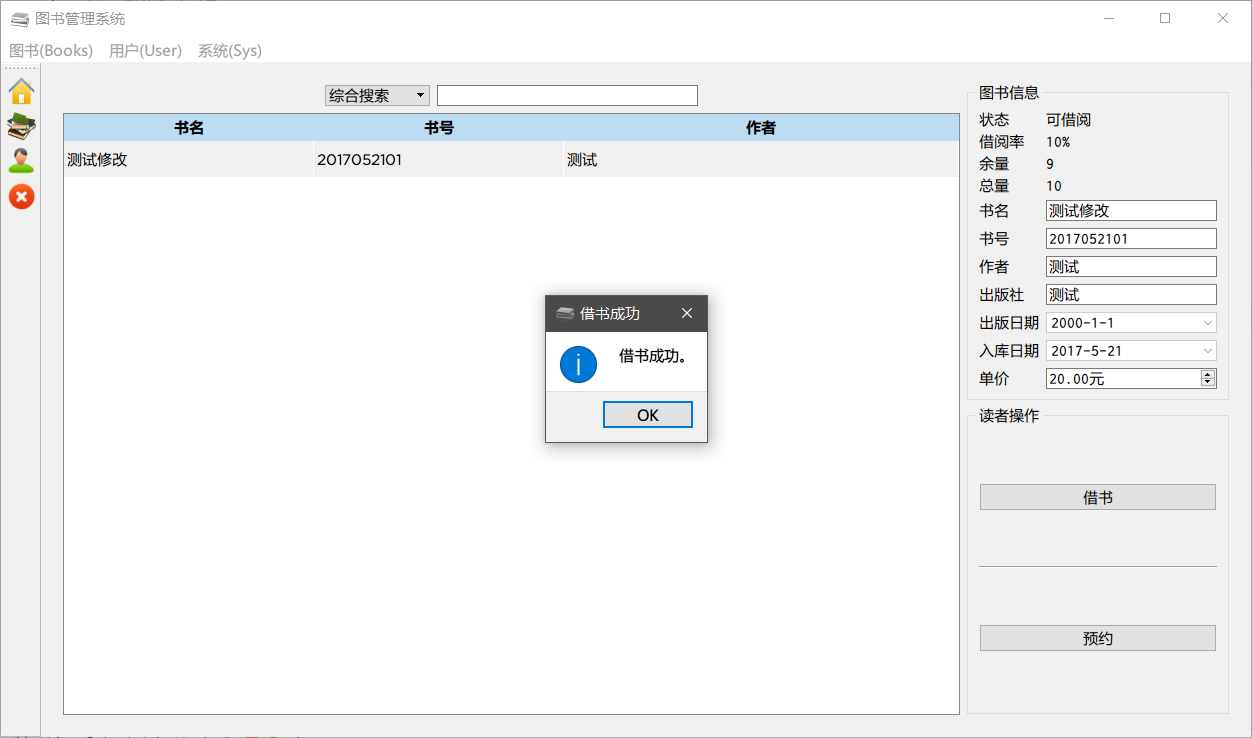


点击删除所选图书，删除成功：



## 图书借阅

任一模式登录，点击借书，借书成功。

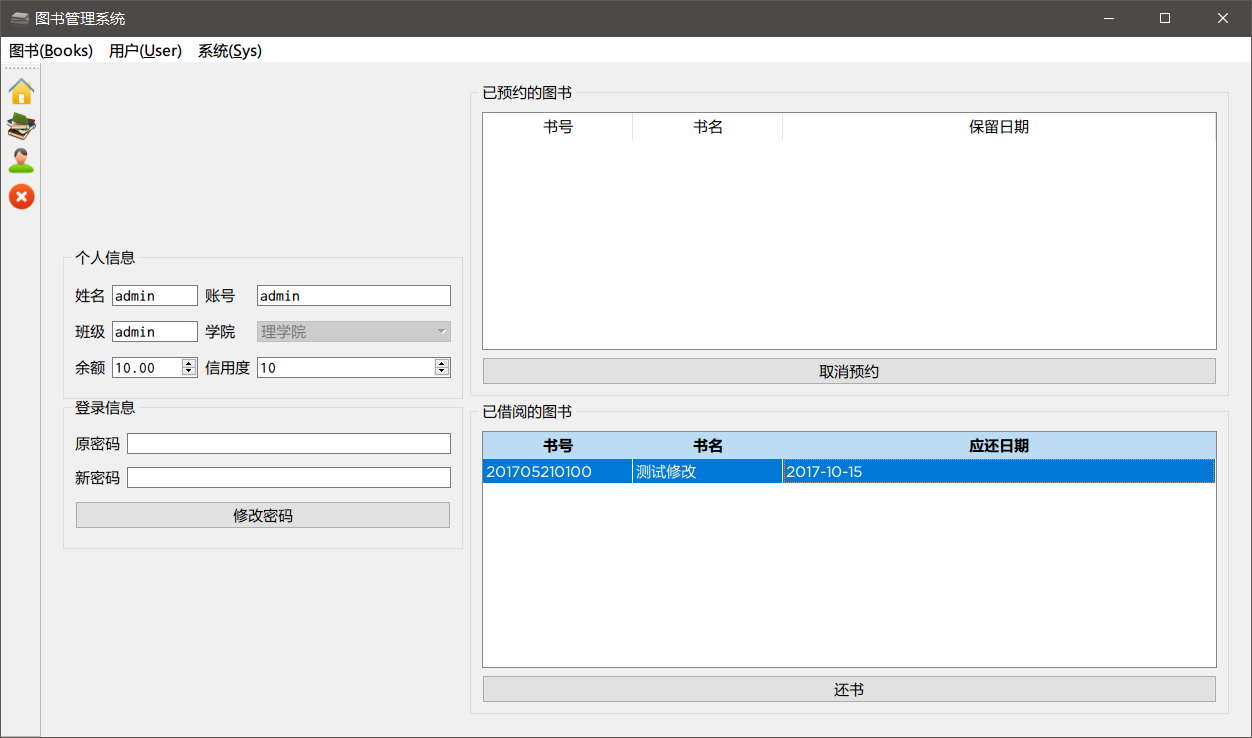


输出日志：

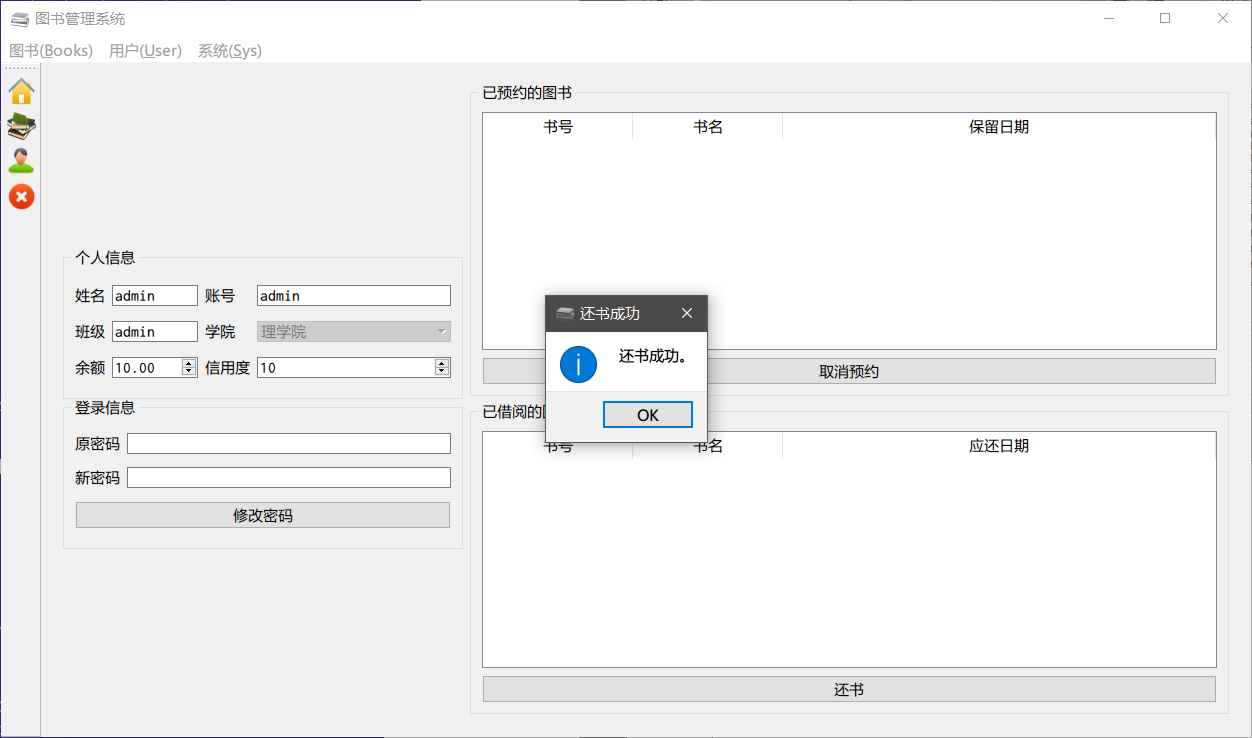


## 图书归还

选中已借阅的图书：



还书成功。

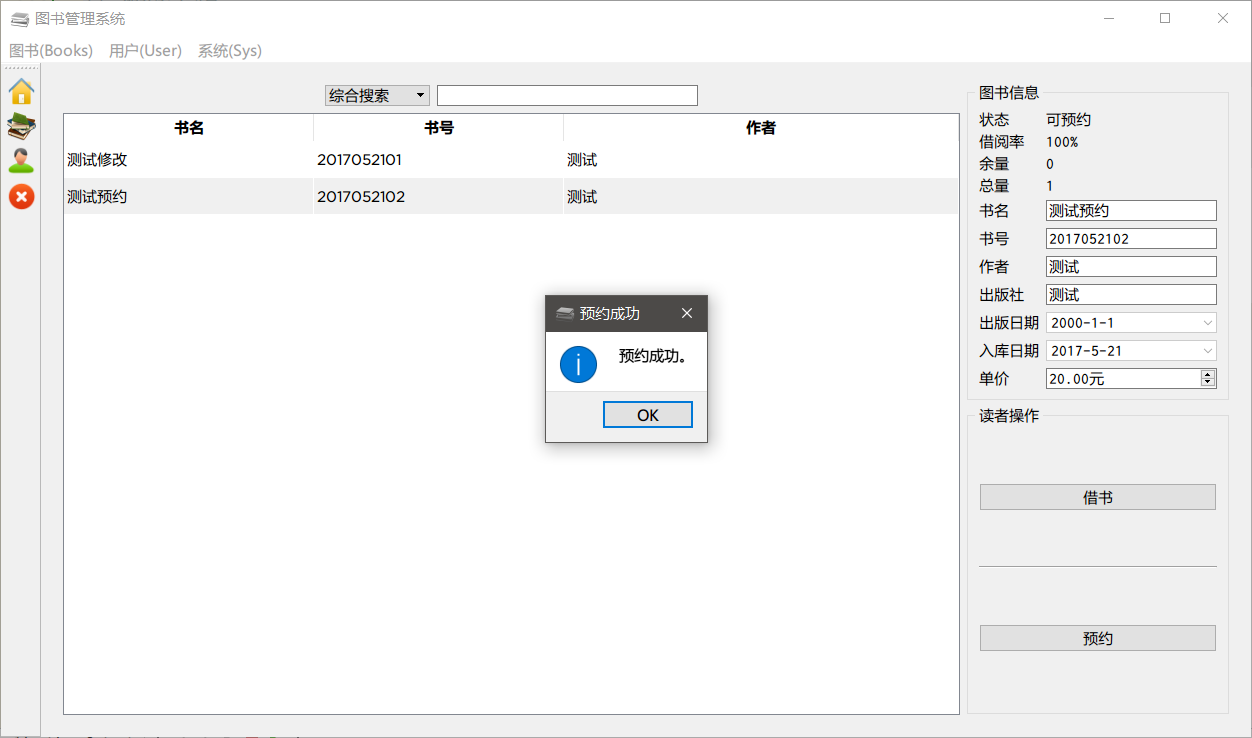


日志输出

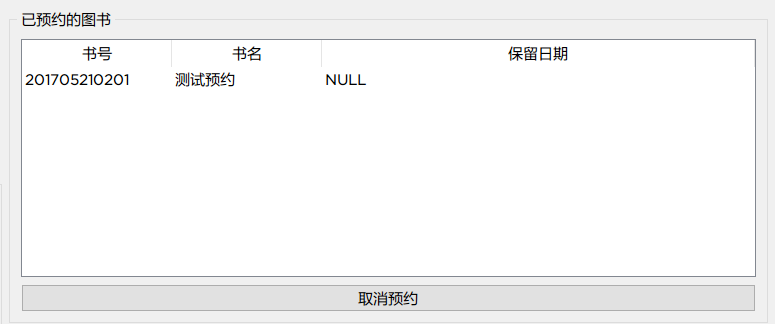


## 图书预约

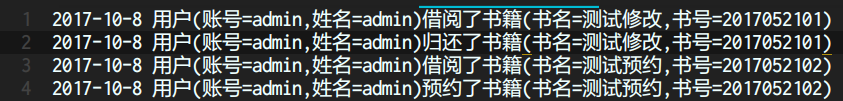
先把图书借完，图书余量为0，且无人预约，即可预约。



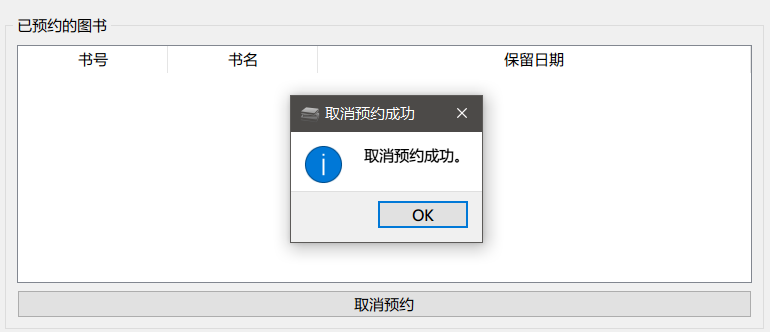
预约成功。



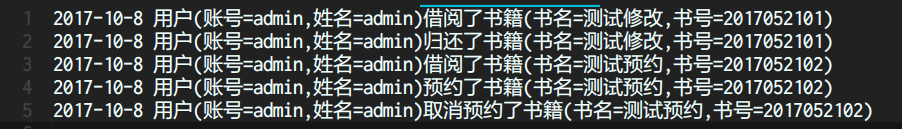
日志输出：



## 取消预约



日志输出：



# 本设计的优点

1. 各子函数功能独立。便于修改和纠错
2. 代码灵活。实现功能简单易懂
3. 把信息读入内存。查找、修改时效率高，代码设计简单
4. 代码风格较好。分段清晰，缩进体现代码层次
5. 代码健壮性强。可以较好的处理错误操作，如非法输入等

# 本设计的缺点

1. 函数设计中使用了友元函数，对类的封装性有所破坏
2. 没有实现对用户信息的加密
3. 当信息总量过大时，可能对内存造成不可预知的影响

# 本设计的时间性能

在本设计中，在关键功能的实现上，用到的主要操作有：图书的添加、删除、修改、查询以及读者的注册、修改信息。其中，对于图书信息的保存使用了两个类的数组，大书数组用来保存书的种类信息，小书数组用来保存每一种类的书的每一本确定的书的具体信息（包括书名、状态（是否被借阅、预约等）、借阅（或预约）人信息以及图书的归还日期等）。

1. 当进行图书的添加时，需要进行的操作数是确定、有限步的，时间复杂度为O(1)；
2. 当进行图书的查询时，需要遍历大书数组，从而得到查询结果，此过程的时间复杂度为O(m) （m为当前图书馆的总的书的种类数）；
3. 当进行图书信息的修改、删除时，需要先进行查询操作，此过程时间复杂度为O(m)，找到相对应的图书，才能进行修改、删除操作，修改、删除大书目录的信息时间复杂度为O(1)，在对相对应的小书信息进行操作时，时间复杂度为O(s)（s为对应的每一种类书所拥有的数量）；
4. 在进行读者注册时、修改时所需要的操作数都是有限、确定的，因而时间复杂度为O(1)；
5. 在借阅、预约图书时，首先需要对图书进行遍历搜寻找到要借阅的图书。由于此过程采用线性搜索，所以时间复杂度为O(m)，在查询到确定的书籍时，需要再一次对保存该类书的每本书的具体信息的小树目录进行遍历，以便确定是否有图书可以借阅、或者预约，此过程的时间复杂度为O(s)，在确定可以借阅或预约之后，对图书信息的修改以及读者信息的修改的时间复杂度均为O(1)，所以，整个过程的时间复杂度为O(m\*s)；
6. 在归还图书时，归还的同时，需要判断是否有读者在预约这本书，才能够确定这本书的归还之后的状态（入馆可以借阅或者放上预约书架等候预约人来取），判定状态的时候由于要根据预约的先后顺序才能确定下一个预约人是谁，所以时间复杂度为O(s)。之后，对于图书信息的修改以及对于读者信息的修改的时间复杂度均为O(1)，至此，还书操作完成，整个过程的时间复杂度为O(s)。
7. 综上所述，可以发现，所有操作的中最大时间复杂度为O(m\*s)，因而整个的程序设计的时间性能还是比较不错的。

# 本设计的空间性能

在该设计中，所有的图书信息包括大书（书的种类数组）、小书（书的详细信息数组）以及所有的读者信息均在程序开始执行时以数组的形式保存在内存中，（假设数的种类总数为m，书的总数量为n），同时，在内存中还保存有int reserve[]，其中，保存着被预约待取书籍的序号。当日期发生变化时，对保存在reserve数组中的图书的待取时间和现在当前时间进行比较，从而判断是否预约超时未取，由于被预约的书籍一般来说相对较少，所以，占用的内存空间不会特别大。

当图书的数量或者读者的数量较多时，保存图书信息和读者信息占用的内存会比较大。

# 与其他小组的比较

和其他小组的比较中，还是会发现一些算法设计上的不同，例如：

1. 在对预约操作的处理上，我们采用的是优先级的做法，最先预约的人拥有最高的优先级，每次有新的预约信息时，原先所有对于同一本书的预约人的优先级都会加1，从而，每次有图书归还时可以通过优先级的比较，选出一个合适的预约人来拿取这本书。在和其他组的讨论时，发现，他们对于这个问题的解决才采用时链表的数据结构。最终，都达到了解决问题的目的。
2. 在搜索操作上，我们实现了模糊搜索，即使没有输入全部的图书信息，也能够根据当前的输入筛选出符合条件的书籍。在和其他组讨论时发现，有的组实现查询的方式是要通过精确查找才能实现。
3. 在对于图书信息以及读者信息的存储上，和其他小组也有一些差异。

# 遇到的问题及思考、解决

在程序设计的过程中，我们遇到过很多问题，最终，通过交流讨论，问题也得到了解决，现在，将主要问题罗列如下：

1. 对于图书以及读者信息的访问方式的思考。由于考虑到如果将全部的图书以及读者信息全部读取到内存中，可能回占用较大的内存空间，因此，刚开始，我们对于这种访问方式持谨慎态度，希望能够通过直接从文件中获取有用的信息进行利用，从而能够极大的降低内存的开销。但是，后来，通过讨论发现，如果直接从文件中获取信息的话，会使得一些操作难以进行或者复杂度很高，同时，从文件中获取信息的方式本身也会比从内存中获取更加低效。综上所述，最终，我们采取了在程序启动时，将信息全部读取到内存中的做法。
2. 对于图书信息中不同类型信息的定义。刚开始，我们考虑的是，建立一个大的图书类数组，在其中，保存每一本图书的信息。但是，后来发现，如果对于一种图书，如果数量不为1时，在进行查询时，会对同一种书，进行多次重复操作，从而会明显降低查询的效率。最终，考虑了种种可能，我们决定对于图书建立两个类数组，一个用来保存每一类图书的信息（包括书名、编号、出版社、出版日期、购入数量、借阅率等信息），另一个用来保存每一类图书中，每一本图书的具体信息（包括书名、状态（是否被借阅、预约等）、借阅（或预约）人信息以及图书的归还日期等）。对于不同的需求，对不同的数组进行操作，极大的提高了一些算法的效率，避免了一些不必要的重复操作。
3. 对于预约图书的处理上刚开始我们产生了一些分歧。有人认为应该按照预约的先后顺序决定当有书归还时，取书的先后顺序；有人认为，应该是，每个人可以预约其中的一本书，当且仅当他所预约的这本具体的书归还时，他才会收到可以取书的信息提示。但是，结合现实考虑以及用户需求，我们最终选择了第一种方案，在实现上，我们采用的是优先级的做法，最先预约的人拥有最高的优先级，每次有新的预约信息时，原先所有对于同一本书的预约人的优先级都会加1，从而，每次有图书归还时可以通过优先级的比较，选出一个合适的预约人来拿取这本书，从而实现了先约先得的目标，同时，如果预约人在规定的时间内没有取走预约的图书，那么会按照超时处理，取消其预约这本书的资格，从剩余预约人中选择预约优先级最高的人继续提醒其可以拿取图书。如若，没有继续预约的人，那么这本书则回归正常书架，可以进行正常的借阅。

# 源代码

## 各源代码文件所实现的功能

|  |  |  |
| --- | --- | --- |
| 序号 | 源代码文件名 | 所实现的功能 |
| 1 | mainwindow | 所有操作的图形界面，以及一部分程序与用户的交互 |
| 2 | classBook | 对图书类的定义，对图书信息的存储、读取和修改 |
| 3 | classUser | 对用户类的定义，对用户信息的存储、读取和修改 |
| 4 | publicfuns | 通过调用图书和用户源代码中定义的函数，实现借书、还书、预约书、取消预约等功能 |
| 5 | registerdialog | 实现注册过程中，程序与用户的交互 |
| 6 | logindialog | 实现登陆过程中，程序与用户的交互 |
| 7 | aboutdialog | 显示关于本程序的一些相关信息。 |

## 各源代码文件的交互关系图