武汉纺织大学

Java应用开发课程设计

**基于MyBatis的图书馆系统**

**学 院： 数学与计算机学院**

**班 级： 软件工程11802**

**姓 名： 刘颖**

**学 号： 1801230203**

**指导老师： 聂刚**

**成 绩：**

**完成日期： 2020年6月20日**

目 录

[1 需求分析 1](#_Toc40880660)

[1.1用户登陆 1](#_Toc40880661)

[1.2主菜单 1](#_Toc40880662)

[1.3各项功能 1](#_Toc40880663)

[1.3.1 查询图书 1](#_Toc40880664)

[1.3.2 借阅图书 2](#_Toc40880665)

[1.3.3 归还图书 2](#_Toc40880666)

[1.3.4 查询借阅记录 2](#_Toc40880667)

[1.3.5 馆内管理 3](#_Toc40880668)

[1.3.6 修改密码 5](#_Toc40880669)

[1.3.7 退出 6](#_Toc40880670)

[2 系统设计 6](#_Toc40880671)

[2.1用户用例图 6](#_Toc40880672)

[2.2 ER图 6](#_Toc40880673)

[2.3 UML活动图（Activity Diagram） 7](#_Toc40880674)

[2.3.1 登陆 7](#_Toc40880675)

[2.3.2 查询图书 8](#_Toc40880676)

[2.3.3 借阅图书 8](#_Toc40880677)

[2.3.4 归还图书 9](#_Toc40880678)

[2.3.5 查询借阅记录 10](#_Toc40880679)

[2.3.6 馆内管理 10](#_Toc40880680)

[2.3.6.1图书增加 11](#_Toc40880681)

[2.3.6.2 图书修改 12](#_Toc40880682)

[2.3.6.3 图书删除 13](#_Toc40880683)

[2.3.6.4 读者增加 14](#_Toc40880684)

[2.3.6.5 读者修改 15](#_Toc40880685)

[2.3.6.6 读者删除 16](#_Toc40880686)

[2.3.6.7 借阅查询 17](#_Toc40880687)

[2.3.6.8 借阅统计 17](#_Toc40880688)

[2.3.7 修改密码 18](#_Toc40880689)

[3 系统实现 19](#_Toc40880690)

[3.1 项目结构 19](#_Toc40880691)

[3.2 配置文件 19](#_Toc40880692)

[3.2.1 log4j.properties文件 19](#_Toc40880693)

[3.2.2 mybatis-config.xml文件 19](#_Toc40880694)

[3.3 VO类 20](#_Toc40880695)

[3.3.1 User.java 20](#_Toc40880696)

[3.3.2 Book.java 22](#_Toc40880697)

[3.3.3 Details.java 23](#_Toc40880698)

[3.4 DAO接口类 25](#_Toc40880699)

[3.4.1 IUserDAO.java 25](#_Toc40880700)

[3.4.2 IBookDAO.java 25](#_Toc40880701)

[3.4.2 IDetailsDAO.java 26](#_Toc40880702)

[3.5 接口映射文件 26](#_Toc40880703)

[3.5.1 UserMapper.xml 26](#_Toc40880704)

[3.5.2 BookMapper.xml 28](#_Toc40880705)

[3.5.3 DetailsMapper.xml 30](#_Toc40880706)

[3.8 用户界面Driver.java 32](#_Toc40880707)

[4 系统测试 32](#_Toc40880708)

[5 系统总结 32](#_Toc40880709)

# 1 需求分析

设计一个图书馆系统，实现“登陆”、“查询图书”、“借阅图书”、“归还图书”、“查询借阅记录”、“图书增加”、“图书修改”、“图书删除”、“读者增加”、“读者修改”、“读者删除”、“借阅查询”、“借阅统计”等功能，具体要求如下：

## 1.1用户登陆

当程序运行时，显示“欢迎使用\*\*\*\*图书馆系统，请登陆”，提示用户依次输入用户名及其密码，当用户名及密码不正确时，显示提示信息“用户名或密码不正确，请重新输入”，**用户最多可以尝试3次，若输入3次均不正确，则给出提示信息“最多只能尝试3次”**，退出系统；

## 1.2主菜单

当用户名及密码正确时，进入系统主界面，如下：

===\*\*\*\*图书馆系统===

1. 查询图书
2. 借阅图书
3. 归还图书
4. 查询借阅记录
5. 馆内管理
6. 修改密码
7. 退出

当前读者：\*\*\*\*\*（显示当前用户名对应的中文名）

请选择（1-7）：

如果输入1-7以外的选择，则显示提示信息“输入无效，只能输入1-6”后，返回主界面。

## 1.3各项功能

### 1.3.1 查询图书

当选择2时，显示“请输入图书编号（6位数字字符）:”，如果编号格式不正确，则显示“你输入的编号格式不正确，请重新输入”，输入正确后，查询统计当前图书所有的基本信息，输出显示格式如下：

XXXXXX号图书信息如下

编号 图书名称 作者 出版社 单价 总数量 在馆数量

===== ======== ==== ====== ==== ===== ======

\*\*\*\*\*\* \*\*\*\*\* \*\* \*\*\*\*\* \*\* \*\* \*\*

请按任意键返回

### 1.3.2 借阅图书

当选择2时，显示“请输入图书编号（6位数字字符）：”，用户输入编号后，如果输入不是6位数字字符，则提示“编号输入格式不正确，请重新输入”，直到输入格式正确，如果输入的编号在图书表中存在，则显示“借阅成功”，否则，提示“您输入的图书编号不存在，请确认后重新输入”，

编号输入完毕后在借阅明细表中增加一条记录（各字段信息如下：记录编号根据规则生成，图书编号为用户输入，图书名称是根据输入的编号查找图书表获取的，借书人为当前登陆用户名，借书时间为系统当前时间hh:mm），增加成功后显示“成功增加一笔借书数据”，返回主菜单。

### 1.3.3 归还图书

当选择3时，显示当前用户所有在借图书，输出格式如下：

XXX 在借图书信息如下

编号 图书名称 数量 借书时间 应还时间

===== ======== ==== === ==== ===== ========

\*\*\*\*\*\* \*\*\* \*\* YYYY-MM-DD YYYY-MM-DD

请输入所要还书的图书编号：

用户输入编号后如果输入不是6位数字字符，则提示“编号输入格式不正确，请重新输入”，直到输入格式正确，如果输入的编号在借书表中存在，则显示“还书成功”，否则，提示“您尚未借阅编号为\*\*\*\*\*\*的图书，请确认后重新输入”，

编号输入完毕后在借阅明细表中增加一条记录（各字段信息如下：记录编号根据规则生成，图书编号为用户输入，图书名称是根据输入的编号查找图书表获取的，借书人为当前登陆用户名，还书时间为系统当前时间yyyy-mm-dd hh:mm:ss），增加成功后显示“成功增加一笔还书数据”，返回主菜单。

### 1.3.4 查询借阅记录

当选择4时，显示当前用户所有图书借阅记录，输出格式如下：

用户：XXX 在借图书信息如下

编号 图书名称 总数量 借书时间 应还时间 还书时间

===== ======== ==== === ==== ===== ======== ========

\*\*\*\*\*\* \*\*\* \*\* YYYY-MM-DD YYYY-MM-DD YYYY-MM-DD

\*\*\*\*\*\* \*\*\* \*\* YYYY-MM-DD YYYY-MM-DD YYYY-MM-DD

### 1.3.5 馆内管理

当选择5时，显示如下子菜单：

===\*\*\*\*图书馆系统===

1. 图书增加
2. 图书修改
3. 图书删除
4. 读者增加
5. 读者修改
6. 读者删除
7. 借阅查询
8. 借阅统计
9. 返回主菜单

当前图书馆人员：\*\*\*\*\*（显示当前用户名对应的中文名）

请选择（1-9）：

* 当选择1时，显示“请按照 图书编码，图书名称，作者，出版社，单价，图书数量 的格式输入图书信息”从键盘依次录入图书信息，格式为“图书编码，图书名称，作者，出版社，单价，图书数量”，如：“100008，Java开发实战经典，李兴华，清华大学出版社，79.80元，5”，当输入的编码在数据库中已经存在时，显示“图书编码不能重复，请重新输入”，输入完毕后将该信息增加到数据库中，并显示“增加成功”，返回子菜单。输入时要对输入的各项做格式的校验，如果输入项的格式不正确，则给出提示信息“你输入的数据格式不正确，请重新输入”。
* 当选择2时，显示“请输入所要修改的图书的编码（6位数字字符）：”用户输入编号后如果输入不是6位数字字符，则提示“编号输入格式不正确，请重新输入”，直到输入格式正确，如果输入的编号在图书表中存在，则显示该编号图书的所有信息，输出格式如下：

XXXXXX号图书信息如下

编号 图书名称 作者 出版社 单价 总数量 在馆数量

===== ======== ==== ====== ==== ===== ======

\*\*\*\*\*\* \*\*\*\*\* \*\* \*\*\*\*\* \*\* \*\* \*\*

日期：YYYY年MM月DD日

请按照格式输入新的图书信息，否则输入n返回上一级：

格式为：

图书编码，图书名称，作者，出版社，单价

如：100008，《Java开发实战经典》，李兴华，清华大学出版社，79.80元

输入完毕后将该信息增加到数据库中，并显示“增加成功”，返回子菜单。输入时要对输入的各项做格式的校验，如果输入项的格式不正确，则给出提示信息“你输入的数据格式不正确，请重新输入”。

否则，提示“馆内没有编号为\*\*\*\*\*\*的图书，请确认后重新输入”直到输入编码格式正确。

* 当选择3时，显示“请输入所要删除的图书的编码（6位数字字符）：”用户输入编号后如果输入不是6位数字字符，则提示“编号输入格式不正确，请重新输入”，直到输入格式正确，如果输入的编号在图书表中存在，则显示该编号图书的所有信息，输出格式如下：

XXXXXX号图书信息如下

编号 图书名称 作者 出版社 单价 总数量 在馆数量

===== ======== ==== ====== ==== ===== ======

\*\*\*\*\*\* \*\*\*\*\* \*\* \*\*\*\*\* \*\* \*\* \*\*

请输入y删除图书，或输入n返回子菜单：

若输入y则将数据库中的信息删除，并显示“删除成功”，返回子菜单。若输入n则返回子菜单。

* 当选择4时，从键盘依次录入读者信息，格式为“读者名称，性别，学号”，如：“张三，男，10000005”，当输入的读者在数据库中已经存在时，显示“该读者已经注册，请重新输入”，输入完毕后将该信息增加到数据库中，分配读者号并显示“增加成功”，返回子菜单。输入时要对输入的各项做格式的校验，如果输入项的格式不正确，则给出提示信息“你输入的数据格式不正确，请重新输入”。
* 当选择5时，显示“请输入所要修改的读者的读者号（8位数字字符）：”用户输入读者号后如果输入不是8位数字字符，则提示“读者号输入格式不正确，请重新输入”，直到输入格式正确，如果输入的读者号在读者表中存在，则显示该读者的所有信息，输出格式如下：

XXXXXX号读者信息如下

读者号 读者名称 学号

===== ======== ======

\*\*\*\*\*\* \*\*\*\*\* \*\*\*\*\*

用户：XXX 在借图书信息如下

编号 图书名称 总数量 借书时间 应还时间 还书时间

===== ======== ==== === ==== ===== ======== ========

\*\*\*\*\*\* \*\*\* \*\* YYYY-MM-DD YYYY-MM-DD YYYY-MM-DD

\*\*\*\*\*\* \*\*\* \*\* YYYY-MM-DD YYYY-MM-DD YYYY-MM-DD

请按照格式输入新的读者信息，否则输入n返回上一级：

格式为：

读者姓名，性别，学号

如：张三，男，100000005

输入完毕后修改数据库中的用户表，并显示“修改成功”，返回子菜单。输入时要对输入的各项做格式的校验，如果输入项的格式不正确，则给出提示信息“你输入的数据格式不正确，请重新输入”。

否则，提示“馆内没有读者号为\*\*\*\*\*\*\*\*的读者，请确认后重新输入”直到输入读者号格式正确。

* 当选择6时，显示“请输入所要删除的读者的读者号（8位数字字符）：”用户输入读者号后如果输入不是8位数字字符，则提示“读者号输入格式不正确，请重新输入”，直到输入格式正确，如果输入的编号在用户表表中存在，则显示该编号图书的所有信息，输出格式如下：

XXXXXX号读者信息如下

读者号 姓名 学号

===== ======== ======

\*\*\*\*\*\* \*\*\*\*\* \*\*\*\*\*

请输入y删除图书，或输入n返回子菜单：

若输入y则讲数据库中的信息删除，并显示“删除成功”，返回子菜单。若输入n则返回子菜单。

* 当选择7时，显示馆内所有图书借还记录，格式如下：

馆内图书借还信息如下：

序号 图书编码 图书名称 借阅人 借阅时间 还书时间

==== ======== ======== ===== ======== =======

1 \*\*\*\*\* \*\*\*\* \*\*\*\*\* YYYY-MM-DD YYYY-MM-DD

2 \*\*\*\*\* \*\*\*\* \*\*\*\*\* YYYY-MM-DD YYYY-MM-DD

按任意键返回

* 当选择8时，显示馆内所有图书的部分信息和借阅次数，按借阅次数降序排列。输出格式如下：

馆内图书借阅次数总共有\*次，信息如下：

序号 图书编码 图书名称 作者 出版社 借阅次数

==== = ======== ========= ==== ====== =========

1 \*\*\*\*\* \*\*\*\* \*\*\* \*\*\*\*\* \*\*\*

2 \*\*\*\*\* \*\*\*\* \*\*\* \*\*\*\*\* \*\*\*

按任意键返回

### 1.3.6 修改密码

当选择13时，显示“请输入当前用户的原密码：”，若原密码不正确，则提示“原密码输入不正确，请重新输入”，输入正确后，显示“请设置新的密码：”，输入的密码必须满足必要的复杂性，如果不符合要求，则提示“您的密码不符合复杂性要求（密码长度不少于6个字符，至少有一个小写字母，至少有一个大写字母，至少一个数字），请重新输入：”，输入成功后，显示“请输入确认密码：”，若两次输入不一样，则提示“两次输入的密码必须一致，请重新输入确认密码：”，输入成功后，修改数据库中的用户表，并提示“您已成功修改密码，请谨记”，返回主界面。**用户在数据库中存放时不要明文的方式进行存放，可以采用某种加密算法（如MD5加密）存放加密后的密文**。

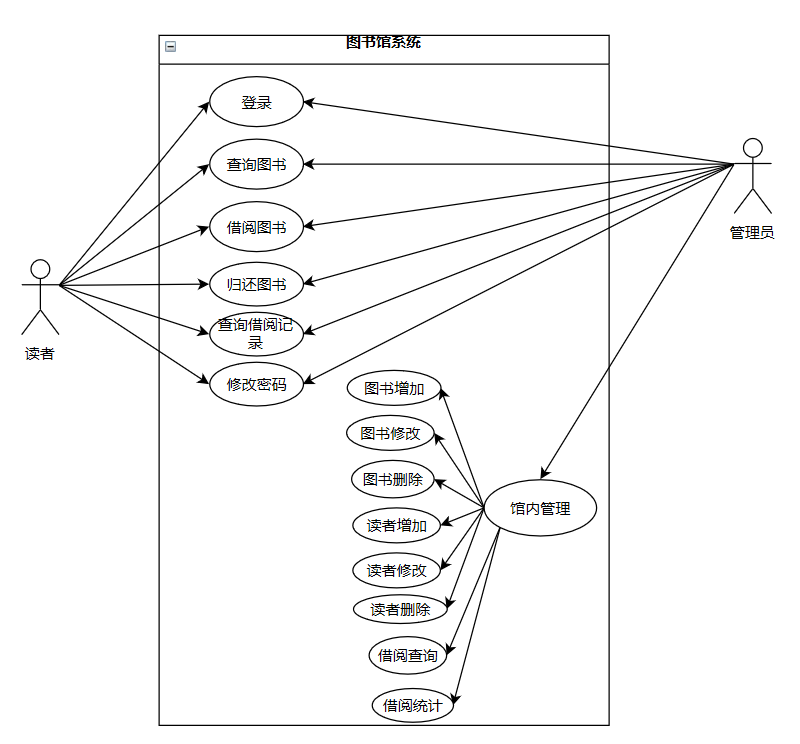
### 1.3.7 退出

选择7时，显示“您确认退出系统吗（y/n）”,输入y，显示“欢迎下次继续使用”，退出程序，输入“n”，返回主菜单。

# 2 系统设计

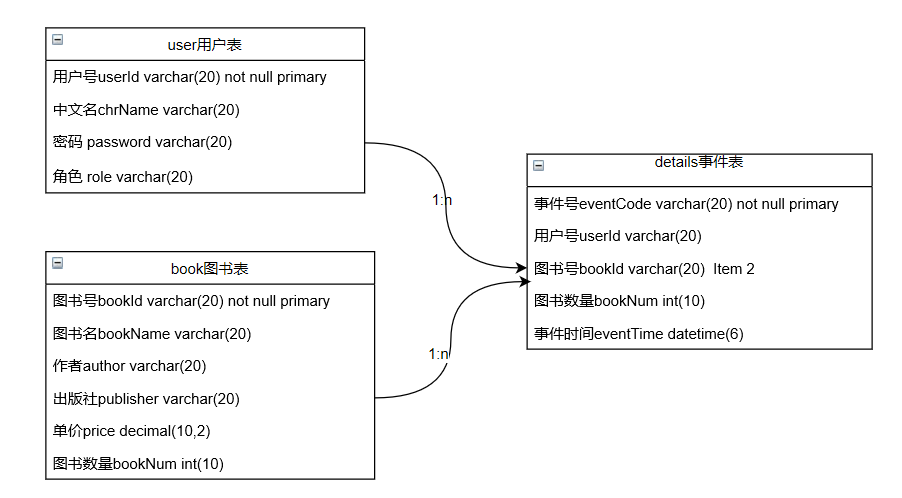
## 2.1用户用例图

系统分为读者，管理员两个角色，各角色的用户用例如下图2-1所示：



## 2.2 ER图

数据库设计3张表，用户表，图书表，事件表，对应的ER图如图3-1所示。



## 2.3 UML活动图（Activity Diagram）

### 2.3.1 登陆

登录操作需要验证用户名，密码，只有当用户名和密码同时存在即用户存在的情况下才能成功登录系统，否则会提示错误信息，具体流程如图2-6所示。

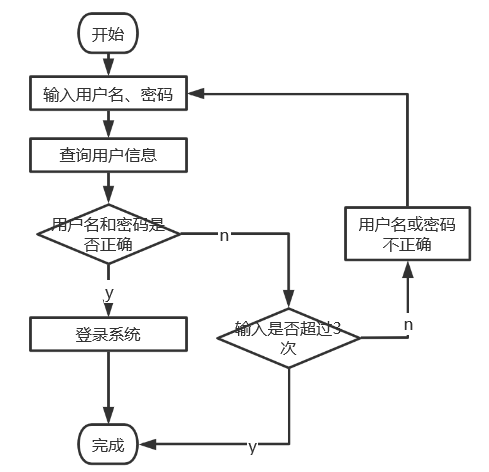
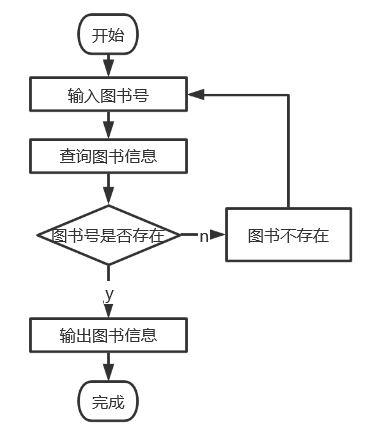


图2-2 登录流程图

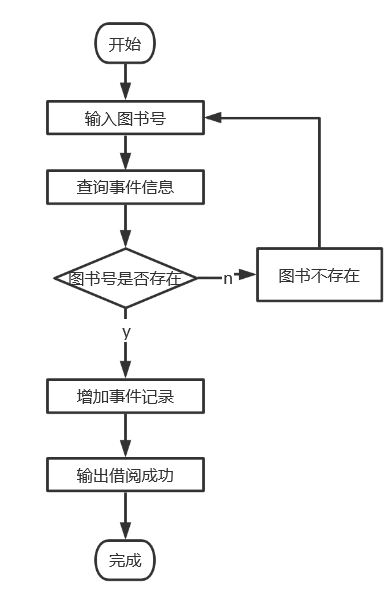
### 2.3.2 查询图书

需要输入图书号做验证信息，图书号存在的情况下才能输入图书信息。



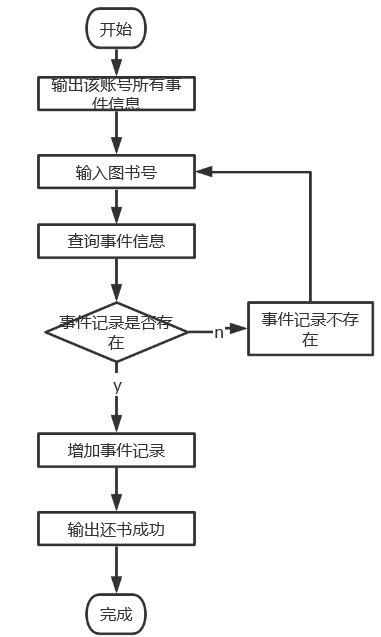
### 2.3.3 借阅图书

需要验证图书号做验证信息，图书号存在才能增加借书记录并成功借阅。



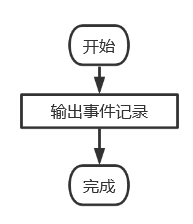
### 2.3.4 归还图书

按格式输出该账号当前所有事件信息，输入图书号，查询对应事件记录是否存在，事件存在才能增加事件还书。



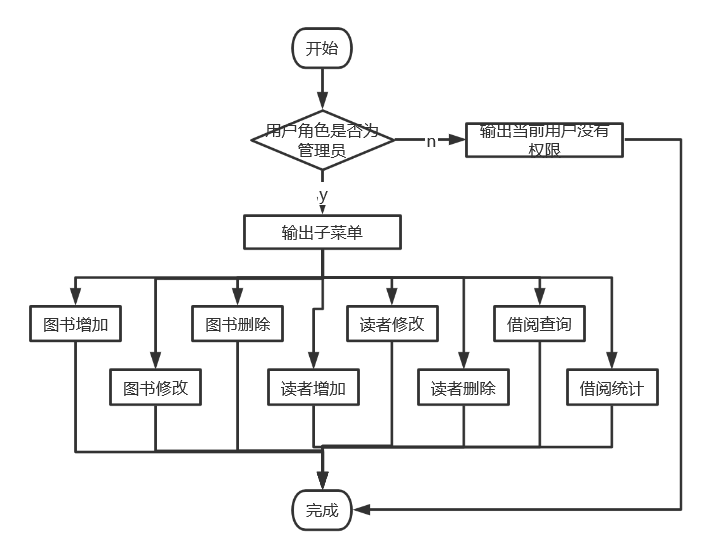
### 2.3.5 查询借阅记录

按格式输出当前账号的所有借还记录。



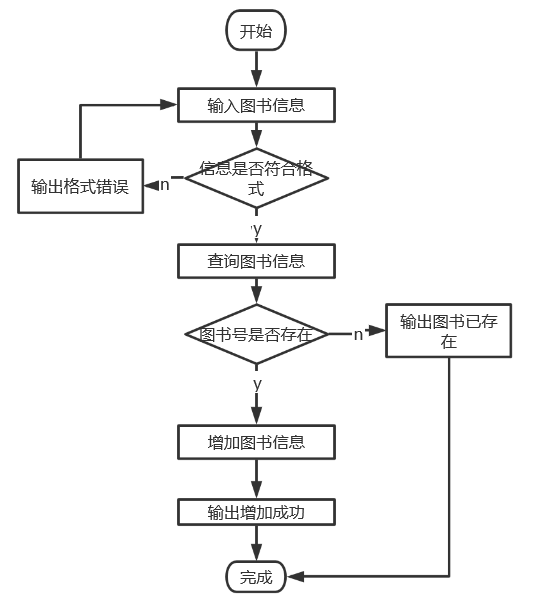
### 2.3.6 馆内管理

验证当前用户是否为管理员，若为管理员，则输出子菜单，根据用户选择进行对图书馆系统的管理。若不为管理员，则输出当前用户没有权限，返回主菜单。



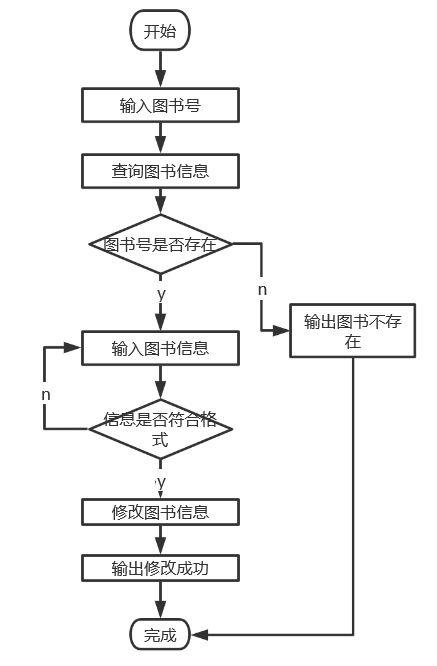
### 2.3.6.1图书增加

输入图书信息，先验证是否符合格式，若不符合格式则继续输入直到格式正确，根据输入信息验证图书是否存在，若不存在则增加图书信息，输出增加成功。若已存在则输出图书已存在。



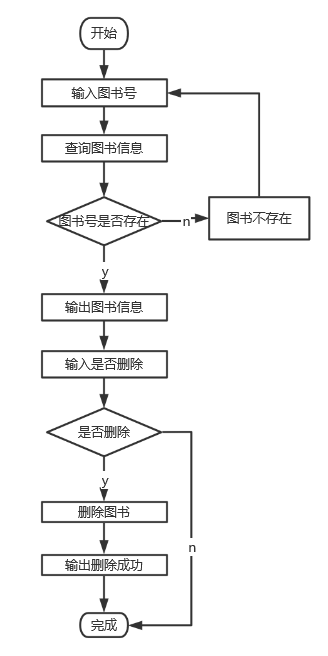
### 2.3.6.2 图书修改

验证输入的图书号是否存在，若存在则输入新的图书信息，验证是否符合格式，直到格式正确，修改图书信息，输出修改成功，若不存在则输出不存在。



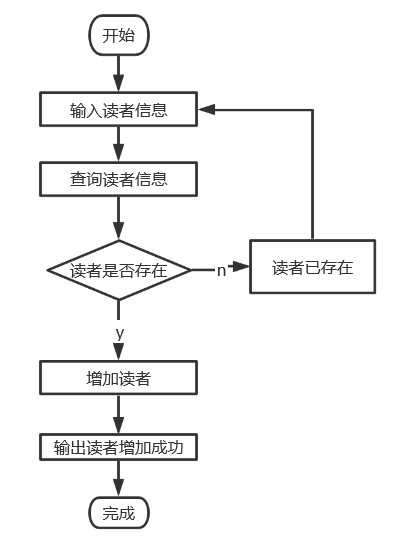
### 2.3.6.3 图书删除

验证输入的图书号是否存在，若不存在则输出图书不存在，直到图书号输入正确，则输入当前图书所有信息，输入y则删除图书，输入n则不删除图书。



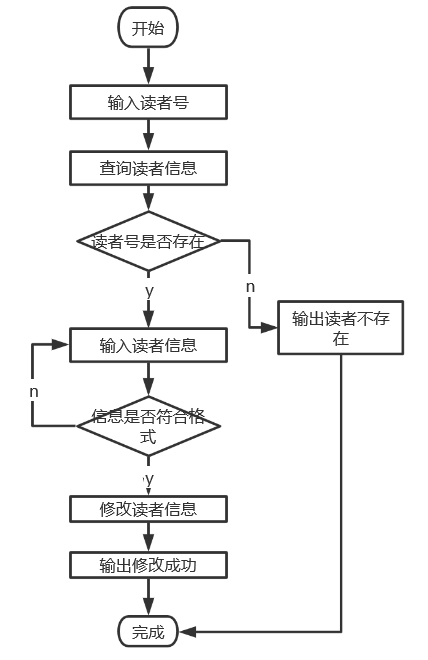
### 2.3.6.4 读者增加

验证输入的读者是否存在，若不存在，则增加读者信息并输出增加成功。



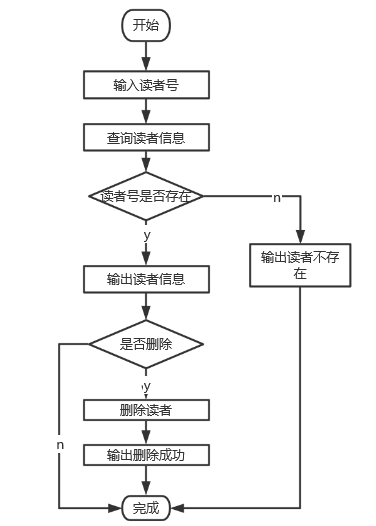
### 2.3.6.5 读者修改

验证输入的读者号是否存在，若存在则输入新的读者信息，验证是否符合格式，直到格式正确，修改读者信息，输出修改成功，若不存在则输出不存在。



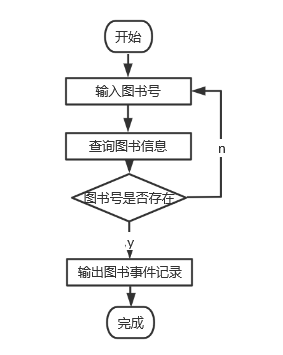
### 2.3.6.6 读者删除

验证读者是否存在，不存在则完成。存在则输入读者信息，输入y则删除读者并输出删除成功。



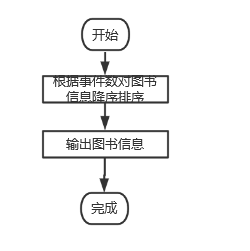
### 2.3.6.7 借阅查询

输入图书号，当图书存在时返回图书所有事件信息，否则继续输入图书号直到图书存在。



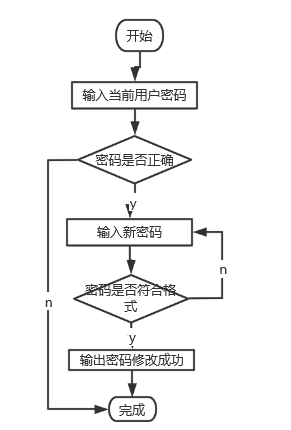
### 2.3.6.8 借阅统计

根据事件数对图书信息进行降序排序并输出。



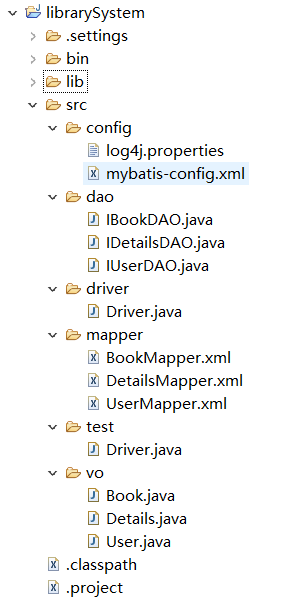
### 2.3.7 修改密码

输入当前用户密码，只有正确才可输入新密码，否则退出。若新密码不符合格式，则继续输入直到格式正确。



# 3 系统实现

## 3.1 项目结构



## 3.2 配置文件

在项目下新建Source Folder，名为resources

### 3.2.1 log4j.properties文件

描述该文件作用，并对里面的配置进行解释

log4j.rootLogger=DEBUG, stdout

log4j.appender.stdout=org.apache.log4j.ConsoleAppender

log4j.appender.stdout.layout=org.apache.log4j.PatternLayout

log4j.appender.stdout.layout.ConversionPattern=%5p [%t] - %m%n

### 3.2.2 mybatis-config.xml文件

描述该文件作用，并对里面的配置进行解释

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE configuration

PUBLIC "-//mybatis.org//DTD Config 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-config.dtd">

<!-- 根标签 -->

<configuration>

<!-- 指定 MyBatis 所用日志的具体实现，未指定时将自动查找-->

<settings>

<setting name=*"logImpl"* value=*"LOG4J"* />

</settings>

<!-- 环境，可以配置多个，default：指定采用哪个环境 -->

<environments default=*"test"*>

<!-- id：唯一标识 -->

<environment id=*"test"*>

<!-- 事务管理器，JDBC类型的事务管理器 -->

<transactionManager type=*"JDBC"* />

<!-- 数据源，池类型的数据源 -->

<dataSource type=*"POOLED"*>

<property name=*"driver"* value=*"* *com.mysql.jdbc.Driver"* />

<property name=*"url"* value= *"jdbc:mysql://127.0.0.1:3306/library?useUnicode=true&amp;characterEncoding=UTF-8"* />

<property name=*"username"* value=*"root"* />

<property name=*"password"* value=*"ABCDer123"* />

</dataSource>

</environment>

</environments>

<!-- 配置mapper映射文件 -->

<mappers>

<mapper resource="mapper/UserMapper.xml" />

<mapper resource="mapper/BookMapper.xml" />

<mapper resource="mapper/DetailsMapper.xml" />

</mappers>

</configuration>

## 3.3 VO类

### 3.3.1 User.java

对user数据表中的数据进行包装

import java.util.Date;

public class User {

private String userId; // 用户号

private String chrName; // 中文名

private String password; // 密码

private String role; // 角色

public User() {

super();

}

public User(String userId, String chrName, String password, String role) {

this.userId = userId;

this.chrName = chrName;

this.password = password;

this.role = role;

}

public void setuserId(String userId) {

this.userId = userId;

}

public String getuserId() {

return userId;

}

public void setchrName(String chrName) {

this.chrName = chrName;

}

public String getchrName() {

return chrName;

}

public void setpassword(String password) {

this.password = password;

}

public String getpassword() {

return password;

}

public void setrole(String role) {

this.role = role;

}

public String getrole() {

return role;

}

public String toString() {

return "User[userId=" + userId + ",chrName=" + chrName + ",password=" + password + ",role=" + role + "]";

}

}

### 3.3.2 Book.java

对book数据表中的数据进行包装

import java.math.BigDecimal;

public class Book {

private String bookId; // 图书号

private String bookName; // 书名

private String author; // 作者

private String publisher; // 出版社

private BigDecimal price; // 图书单价

private int bookNum; // 图书数量

public Book() {

super();

// TODO Auto-generated constructor stub

}

public String getBookId() {

return bookId;

}

public void setBookId(String bookId) {

this.bookId = bookId;

}

public String getBookName() {

return bookName;

}

public void setBookName(String bookName) {

this.bookName = bookName;

}

public String getAuthor() {

return author;

}

public void setAuthor(String author) {

this.author = author;

}

public String getPublisher() {

return publisher;

}

public void setPublisher(String publisher) {

this.publisher = publisher;

}

public BigDecimal getPrice() {

return price;

}

public void setPrice(BigDecimal price) {

this.price = price;

}

public int getBookNum() {

return bookNum;

}

public void setBookNum(int bookNum) {

this.bookNum = bookNum;

}

}

### 3.3.3 Details.java

对details数据表中的数据进行包装

import java.util.Date;

public class Details {

private String eventCode; // 事件号

private String userId; // 用户号

private String bookId; // 图书号

private int bookNum; // 图书数量

private Date eventTime; // 事件时间

public Details() {

super();

// TODO Auto-generated constructor stub

}

public Details(String eventCode, String userId, String bookId, int bookNum, Date eventTime) {

super();

this.eventCode = eventCode;

this.userId = userId;

this.bookId = bookId;

this.bookNum = bookNum;

this.eventTime = eventTime;

}

public String getEventCode() {

return eventCode;

}

public void setEventCode(String eventCode) {

this.eventCode = eventCode;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public String getBookId() {

return bookId;

}

public void setBookId(String bookId) {

this.bookId = bookId;

}

public int getBookNum() {

return bookNum;

}

public void setBookNum(int bookNum) {

this.bookNum = bookNum;

}

public Date getEventTime() {

return eventTime;

}

public void setEventTime(Date eventTime) {

this.eventTime = eventTime;

}

}

## 3.4 DAO接口类

### 3.4.1 IUserDAO.java

User表的接口

import java.util.Date;

import java.util.List;

import vo.User;

public interface IUserDAO{

//根据用户号查找记录

public User get(String userId) throws Exception;

//根据条件查找,查询条件封装在user对象中，若user对象的某个成员变量值为null，则表示查询时忽略该字段查询条件

public List<User> query(User user) throws Exception;

//增加一条用户信息

public boolean insert(User user) throws Exception;

//修改一条用户信息

public boolean update(User user) throws Exception;

//删除指定用户号的记录

public boolean delete(String userId) throws Exception;

}

### 3.4.2 IBookDAO.java

Book表的接口

import java.util.List;

import vo.Book;

public interface IBookDAO {

//根据主键查找记录

public Book get(String bookId);

//根据条件查找,查询条件封装在user对象中，若user对象的某个成员变量值为null，则表示查询时忽略该字段查询条件

public List<Book> query(Book book);

//增加一种图书信息

public boolean insert(Book book);

//修改一种图书信息

public boolean update(Book book);

//删除制定图书号的图书

public boolean delete(String bookId);

//根据条件删除记录

//public int deleteBatch(Book book);

}

### 3.4.2 IDetailsDAO.java

Details表的接口

import java.util.List;

import java.util.Date;

import vo.Details;

public interface IDetailsDAO {

//根据事件号查找记录

public Details get(String eventCode);

//根据图书号查找

public List<Details> query(Details details);

//增加一条记录

public boolean insert(Details details);

//修改

public boolean update(Details details);

//删除指定事件号的记录

public boolean delete(String eventCode);

}

## 3.5 接口映射文件

### 3.5.1 UserMapper.xml

对User的接口映射

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE mapper

PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="dao.IUserDAO">

<!--当数据库中字段名与实体类中属性名不一致时，需要显示定义字段映射 -->

<resultMap id="userResultMap" type="vo.User">

<id property="userId" column="userId" />

<result property="chrName" column="chrName" />

<result property="password" column="password" />

<result property="role" column="role" />

</resultMap>

<sql id="userColumns">

userId,chrName,password,role

</sql>

<select id="get" parameterType="String" resultMap="userResultMap">

select

<include refid="userColumns"></include>

from user

where userId=#{userId}

</select>

<select id="query" parameterType="vo.User" resultMap="userResultMap">

select

<include refid="userColumns"></include>

from user

<where>

<if test="userId!=null">

and userId = #{userId}

</if>

<if test="chrName!=null">

and chrName = #{chrName}

</if>

<if test="password!=null">

and password = #{password}

</if>

<if test="role!=null">

and role = #{role}

</if>

</where>

</select>

<insert id="insert" parameterType="vo.User">

insert into user(<include refid="userColumns"></include>)

values(#{userId},#{chrName},#{password},#{role})

</insert>

<update id="update" parameterType="vo.User">

update user

<set>

<if test="userId!=null">

userId = #{userId},

</if>

<if test="chrName!=null">

chrName = #{chrName},

</if>

<if test="password!=null">

password = #{password},

</if>

<if test="role!=null">

role = #{role}

</if>

</set>

where userId=#{userId}

</update>

<delete id="delete" parameterType="String">

delete from

user

where userId=#{userId}

</delete>

</mapper>

### 3.5.2 BookMapper.xml

对Book的接口映射

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE mapper

PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="dao.IBookDAO">

<!--当数据库中字段名与实体类中属性名不一致时，需要显示定义字段映射 -->

<resultMap id="bookResultMap" type="vo.Book">

<id property="bookId" column="bookId" />

<result property="bookName" column="bookName" />

<result property="author" column="author" />

<result property="publisher" column="publisher" />

<result property="price" column="price" />

<result property="bookNum" column="bookNum" />

</resultMap>

<sql id="bookColumns">bookId,bookName,author,publisher,price,bookNum,</sql>

<select id="get" parameterType="String" resultMap="bookResultMap">

select

<include refid="bookColumns"></include>

from book

where bookId=#{bookId}

</select>

<select id="query" parameterType="vo.Book" resultMap="bookResultMap">

select

<include refid="bookColumns"></include>

from book

<where>

<if test="bookId!=null">

and bookId = #{bookId}

</if>

<if test="bookName!=null">

and bookName = #{bookName}

</if>

<if test="author!=null">

and author = #{author}

</if>

<if test="publisher!=null">

and publisher = #{publisher}

</if>

<if test="price!=null">

and price = #{price}

</if>

<if test="bookNum!=null">

and bookNum = #{bookNum}

</if>

</where>

</select>

<insert id="insert" parameterType="vo.Book">

insert into book(<include refid="bookColumns"></include>)

values(#{bookId},#{bookName},#{author},#{publisher},#{price},#{bookNum})

</insert>

<update id="update" parameterType="vo.Book">

update book

<set>

<if test="bookId!=null">

and bookId = #{bookId}

</if>

<if test="bookName!=null">

and bookName = #{bookName}

</if>

<if test="author!=null">

and author = #{author}

</if>

<if test="publisher!=null">

and publisher = #{publisher}

</if>

<if test="price!=null">

and price = #{price}

</if>

<if test="bookNum!=null">

and bookNum = #{bookNum}

</if>

</set>

where bookId=#{bookId}

</update>

<delete id="delete" parameterType="String">

delete from

book

where bookId=#{bookId}

</delete>

</mapper>

### 3.5.3 DetailsMapper.xml

对Details的接口映射

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE mapper

PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="dao.IDetailsDAO">

<!--当数据库中字段名与实体类中属性名不一致时，需要显示定义字段映射 -->

<resultMap id="detailsResultMap" type="vo.Details">

<id property="eventCode" column="eventCode" />

<result property="userId" column="userId" />

<result property="bookId" column="bookId" />

<result property="bookNum" column="bookNum" />

<result property="eventTime" column="eventTime" />

</resultMap>

<sql id="detailsColumns">

eventCode,userId,bookId,bookNum,eventTime

</sql>

<select id="get" parameterType="String" resultMap="detailsResultMap">

select

<include refid="detailsColumns"></include>

from details

where eventCode=#{eventCode}

</select>

<select id="query" parameterType="vo.details" resultMap="detailsResultMap">

select

<include refid="detailsColumns"></include>

from details

<where>

<if test="eventCode!=null">

and eventCode = #{eventCode}

</if>

<if test="userId!=null">

and userId = #{userId}

</if>

<if test="bookId!=null">

and bookId = #{bookId}

</if>

<if test="bookNum!=null">

and bookNum = #{bookNum}

</if>

<if test="eventTime!=null">

and eventTime = #{eventTime}

</if>

</where>

</select>

<insert id="insert" parameterType="vo.Details">

insert into details(<include refid="detailsColumns"></include>)

values(#{eventCode},#{userId},#{bookId},#{bookNum},#{eventTime})

</insert>

<update id="update" parameterType="vo.Details">

update details

<set>

<if test="eventCode!=null">

and eventCode = #{eventCode}

</if>

<if test="userId!=null">

and userId = #{userId}

</if>

<if test="bookId!=null">

and bookId = #{bookId}

</if>

<if test="bookNum!=null">

and bookNum = #{bookNum}

</if>

<if test="eventTime!=null">

and eventTime = #{eventTime}

</if>

</set>

where eventCode=#{eventCode}

</update>

<delete id="delete" parameterType="String">

delete from

details

where eventCode=#{eventCode}

</delete>

</mapper>

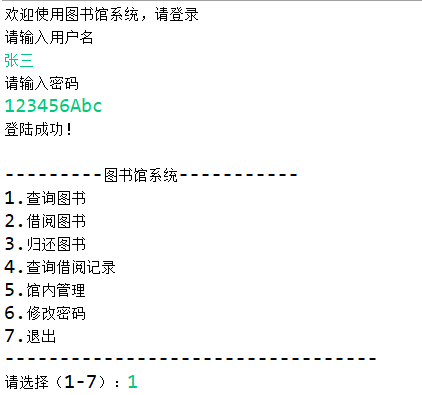
## 3.8 用户界面Driver.java

描述该文件作用，并对里面的关键代码进行解释

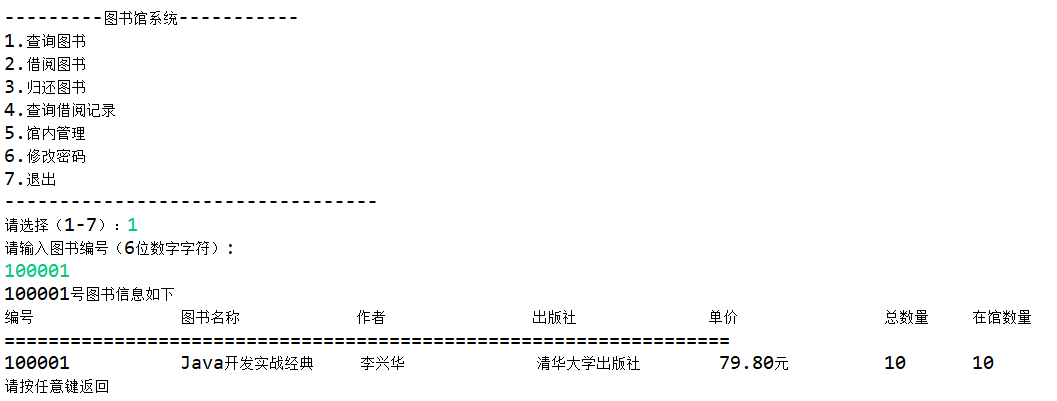
# 4 系统测试

设计测试用例，给出程序每个功能模块的运行结果截图。

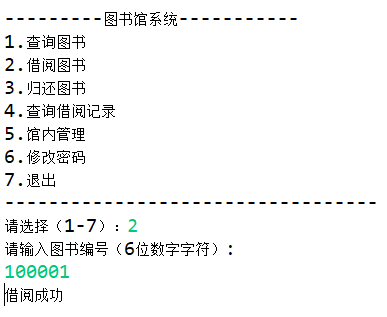
### 4.1登录



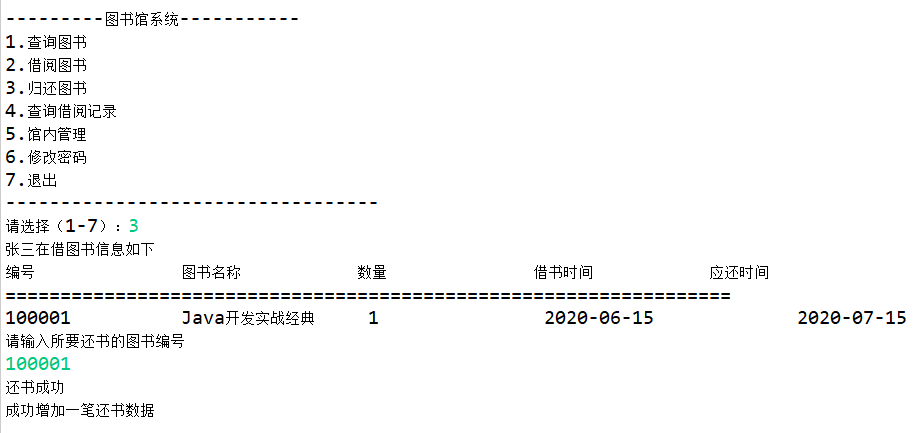
### 4.2 查询图书



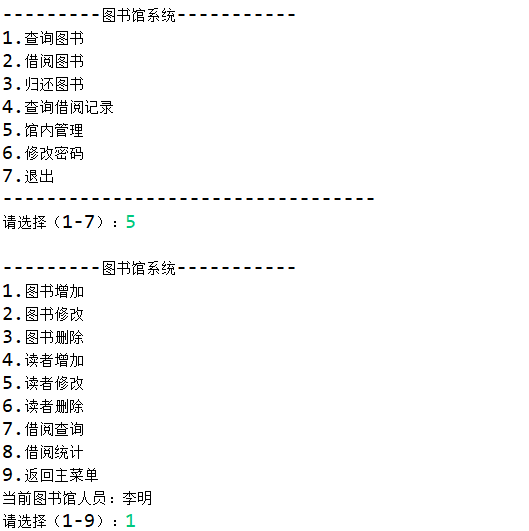
### 4.3 借阅图书



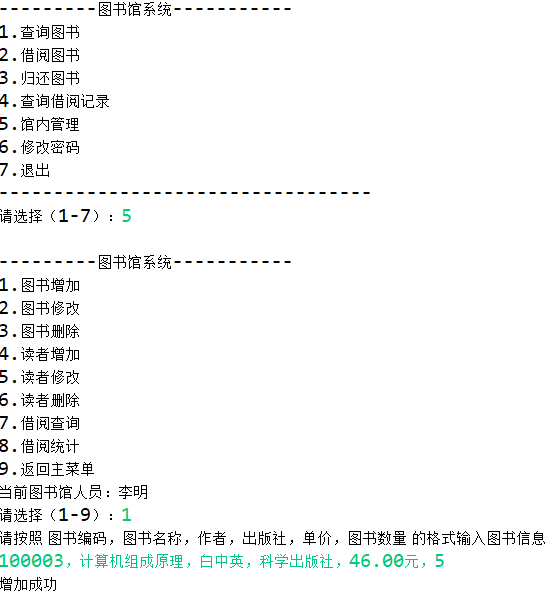
### 4.4归还图书



### 4.5馆内管理



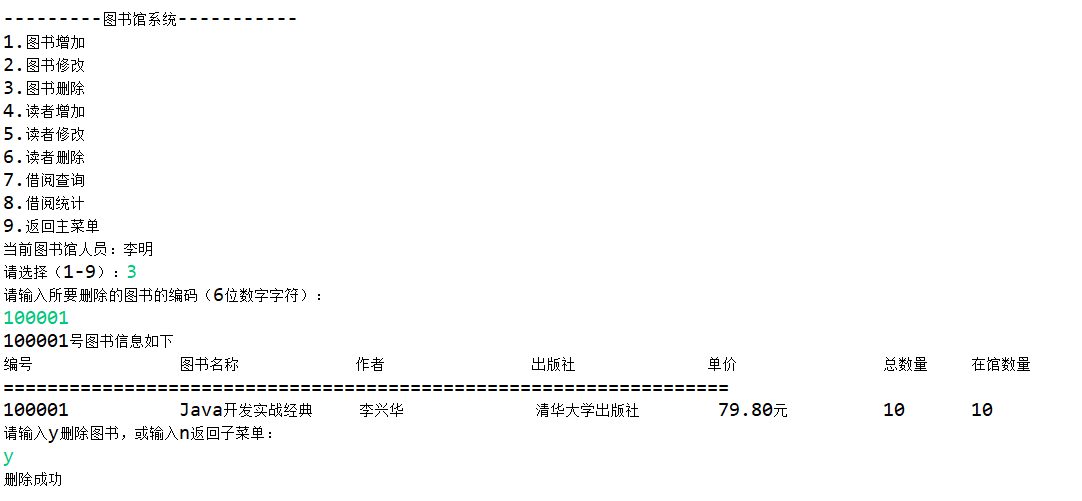
### 4.5.1图书增加



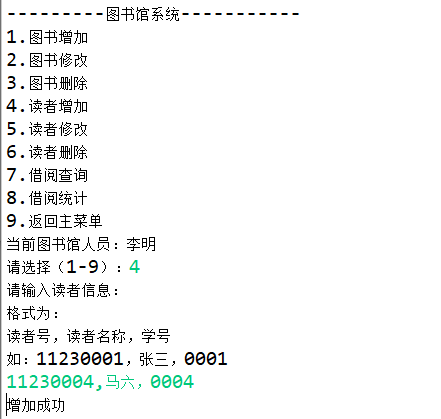
### 4.5.2图书修改



### 4.5.3图书删除



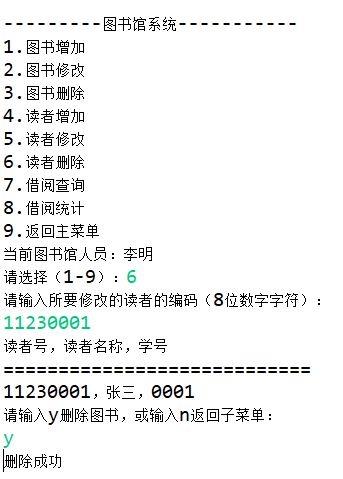
### 4.5.4读者增加



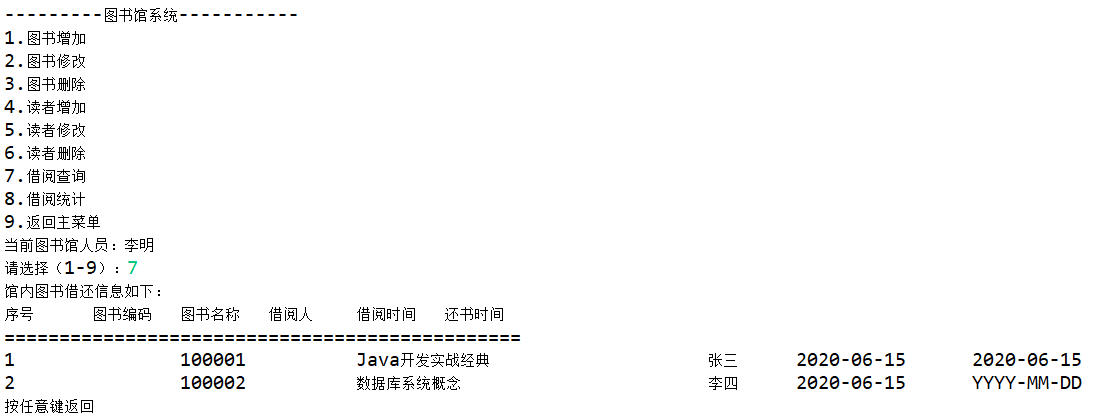
### 4.5.5读者修改



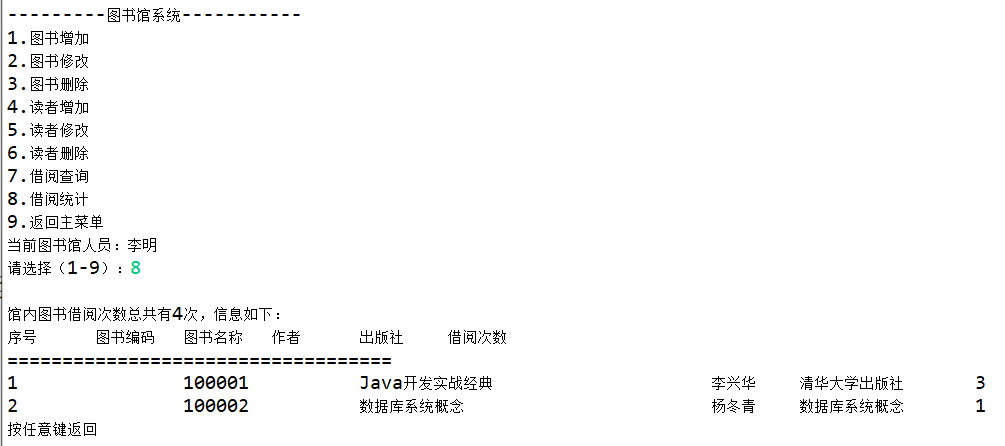
### 4.5.6读者删除



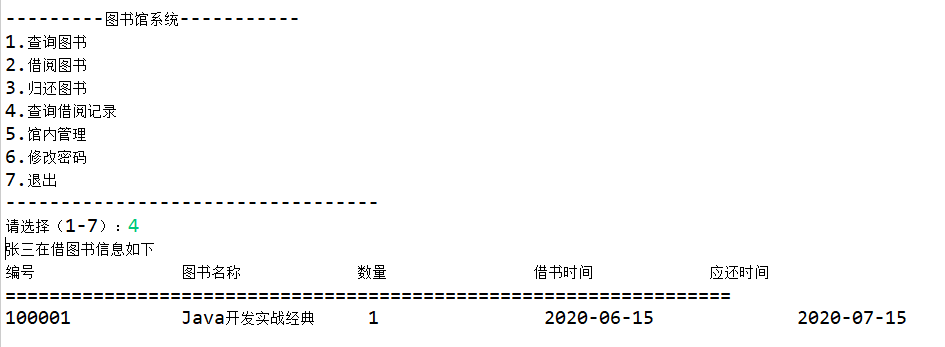
### 4.5.7借阅查询



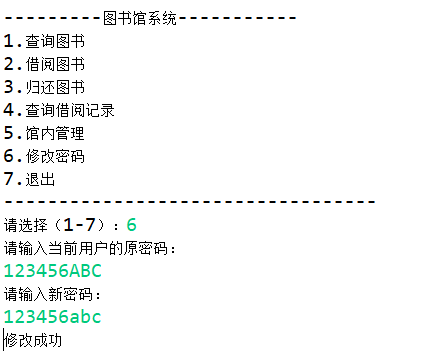
### 4.5.8借阅统计



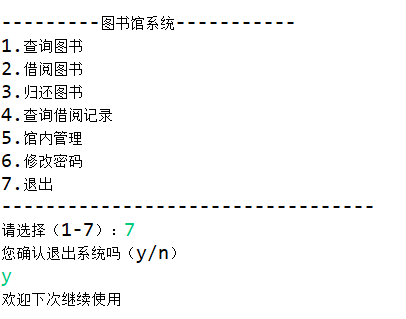
### 4.6查询借阅记录



### 4.7修改密码



### 4.8 退出



# 5 系统总结

在完成作业的过程中我又碰到了数据库连不上，文件调用报错等问题，经过重新阅读教案，请教同学和上网查找解决方案的方法解决了这些问题。这次作业也让我意识到自己能力的薄弱，系统不够灵活，而且完成相同的项目我要比其他同学花更多的时间去解决，也在解决问题当中学习。这门课程与其他课程不同，让我真实的感受到了系统开发的整个流程，不止是单知道只是点，更学会了知识如何运用到开发当中，以后会更多的使用java。我认为自己开发的能力还比较一般有更大的上升空间。