高级程序设计实践

报 告

|  |  |
| --- | --- |
| **姓名** | 张子洋 |
| **学号** | 8208221223 |
| **班级** | 2203班 |
| **专业** | 计算机科学与技术 |
| **年级** | 22级 |
| **时间** | 2024.1.27 |

中南大学计算机学院

2023

## 实验内容

航空机票订票系统设计（java实现）：

1. 设计每条航线所涉及的信息，如起点终点站名、航班号、飞机号、飞机周日（星期几）、乘员定额、余票量、订定票的客户名单（包括姓名、订票量、舱位等级1，2或3）等；
2. 结合基本操作的单链表、队列、二叉树等数据结构以及排序算法，计实现机票订票系统的查询、订票、退票等功能；
3. 设计并实现人机交互友好的界面或菜单。

## 实验方案

**功能模块：**

Login Frame类：实现登录界面的构建

Register Frame类：实现账户注册界面的构建

Register State类：实现注册状态的判断

Information类：实现系统主要信息的读入和存储

Main Panel类：实现系统主界面的面板的构建

Main Frame类：实现承载主面板的主界面的构建

Reserve Frame类：实现订票界面的构建

Search Frame类：实现查询界面的构建

Alter Frame类：实现改签/退票界面的构建

Panel类：实现其他各功能面板的组合选择

**实现思路：**

Login Frame类：设计Register Panel并应用在登录面板上，在登录面板上放置Label，Text field和Button，并通过action Performed函数依次设计点击按钮后的响应：打开对应面板。

Register Frame类和Register State类：采用三个Text Field获取输入的账号密码，当判断账号密码符合输入条件后，通过I/O文件流操作存储在account相关的文本文档中并显示账号注册状态。

Information类：通过Buffered Reader的readline函数获取account、ticket、findticket中的文本信息，通过split函数获取文本中的账户机票信息，并将信息转化为字符串存在一个字符数组中并建立好打印函数留待后用。

Main Frame类/Main Panel类：在面板放置Button并通过action Performed设定好打开新界面的响应，并用界面承载该面板。

Reserve Frame类：将information类中获取的信息通过combo box展示出来，用户通过下拉框选好出发时间，始发地，目的地后，跳转可选择机票信息界面，打印information从Ticket文本文档中获取的机票信息，用户选取机票信息后，跳出输入用户信息输入框，根据输入框中提示输入用户身份信息，系统将输入的信息存储在字符数组中，再将字符数组中的信息存入Find Ticket文本文档中，可供后续程序读取。

Search Panel类：先跳出一个身份信息确认界面，输入乘客姓名及身份证号，将该信息存储在一个临时的字符数组中，并逐行读取Find Ticket文本文档中的信息中表示乘客姓名及身份证号的信息存储在另一个字符数组中并比较，若有该乘客信息，则进入查找界面，若无则提示该乘客无订票信息。进入查找界面后，打印information从Find Ticket文本文档中获取的机票信息。

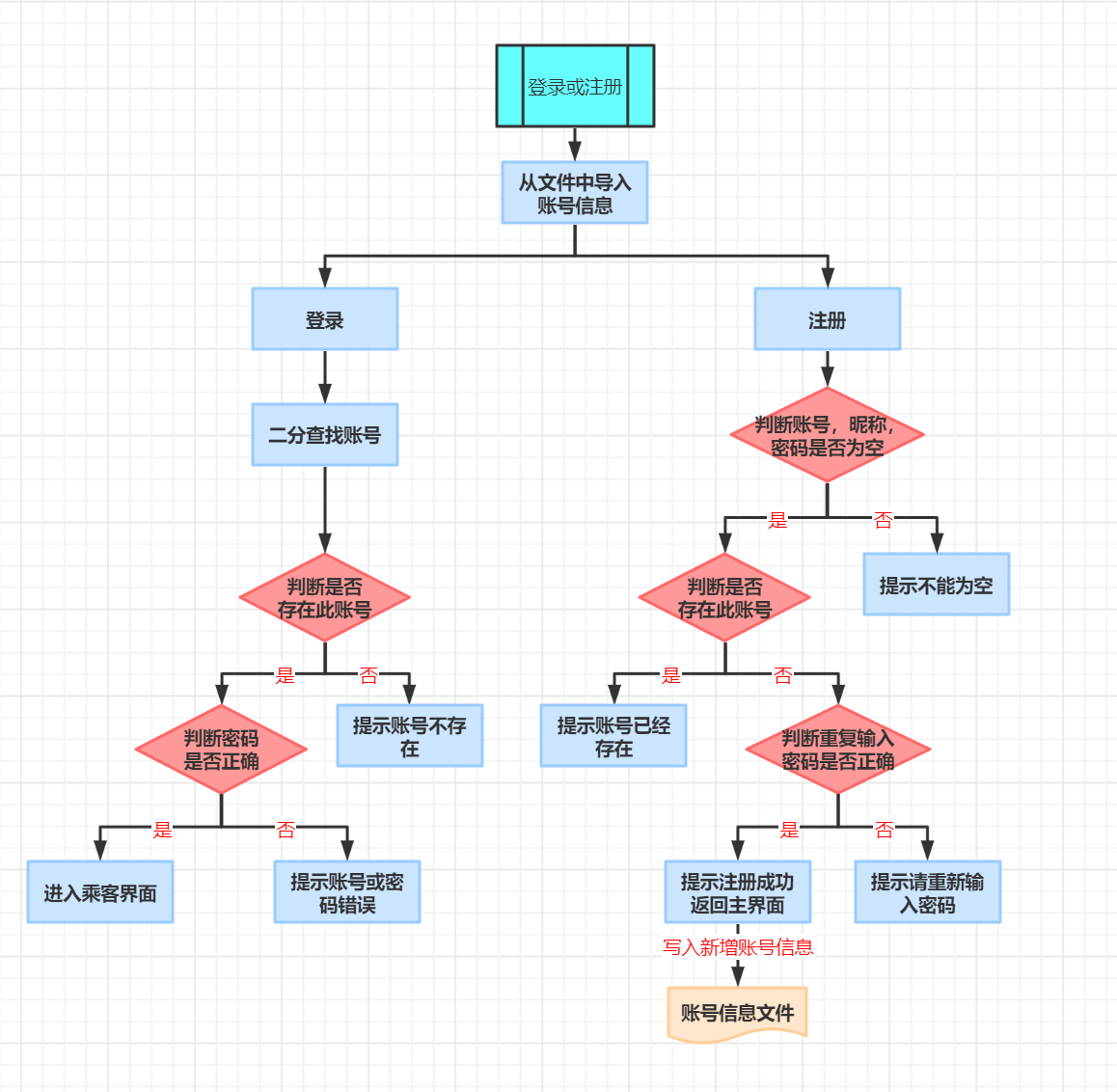
Alter Frame类：先跳出一个身份信息确认界面，输入乘客姓名及身份证号，将该信息存储在一个临时的字符数组中，读取Find Ticket文本文档中的信息中表示乘客姓名及身份证号的信息存储在另一个字符数组中并比较，若有该乘客信息，则进入改签界面，若无则提示该乘客无订票信息。进入改签界面后，打印information从Find Ticket文本文档中获取的机票信息。通过点击Radio Button选取想更改的机票信息，先跳出询问是否改签的文本框，选择是则再次打开订票界面，订票后删除原订票信息，选择否则跳出是否退票的文本框，选择是则退票，选择否则关闭文本框。

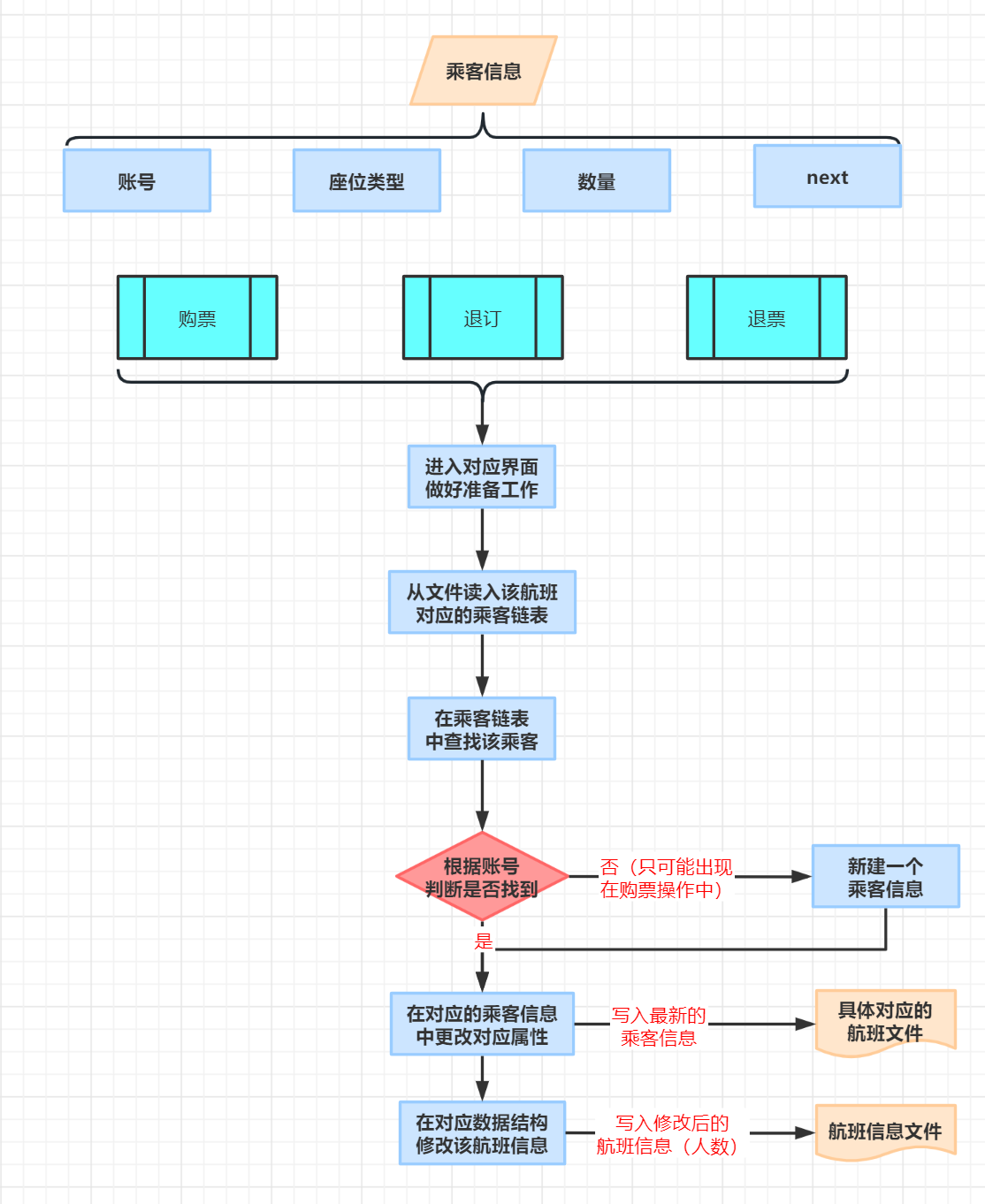
**测试方案：**

在account.txt文件中存好账户信息，或通过界面中的注册来储存客户信息。Ticket的txt文件中存储票的信息（航班，始终地点，时间，飞行时间，头等舱和经济舱价格与各自票数），并将购买成功的票信息放入findticket.txt中。用户登入系统后，可以购票，退改票以及查询结果，航班信息只能通过修改文件内容来添加或修改，事先在文件中放入样例信息，然后测试功能是否正常即可。

## 流程图或类图

**登录模块流程图：**

**用户界面流程图：**

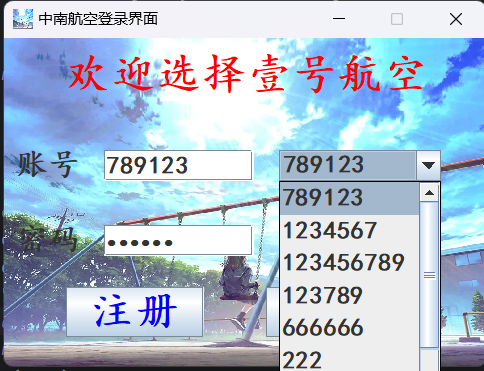


## 难点及其解决方法

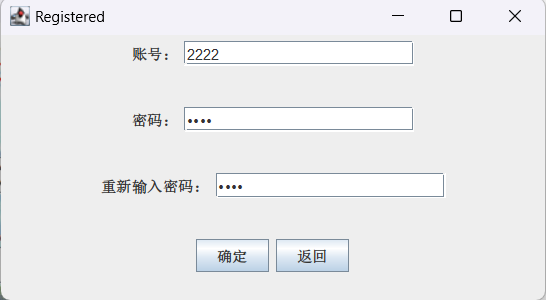
在一开始设计代码时，为了减少后续重复读取文件的次数，先写了information类，后面直接读取存储在information类中存储的信息，但是由于information在打开程序时读取信息，这导致注册等的新信息只有在程序重启之后才能读取。为了实现同一次使用程序中的同步更新数据，便把读取信息的代码插入开启界面中并在存储入新信息时关闭旧界面，开启新界面，这样每一次重新开启界面就会读取新数据并使用。综上所述，有时候节省代码运行时间而做出的操作可能会导致无法满足该功能的某些特定需求。

## 5. 程序测试

登录：



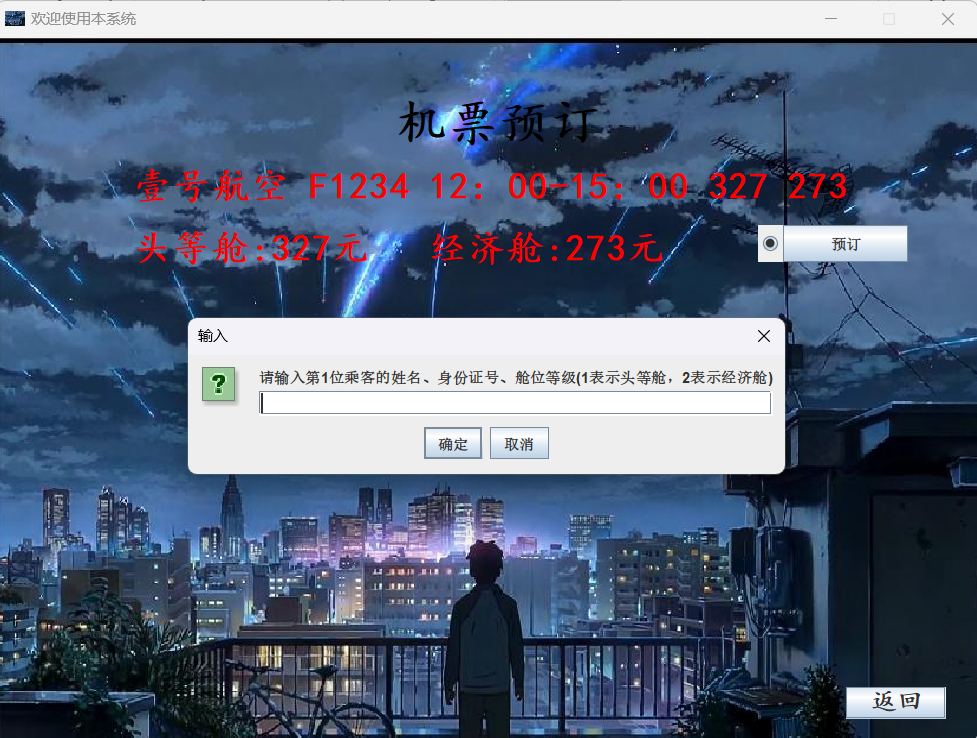
注册：





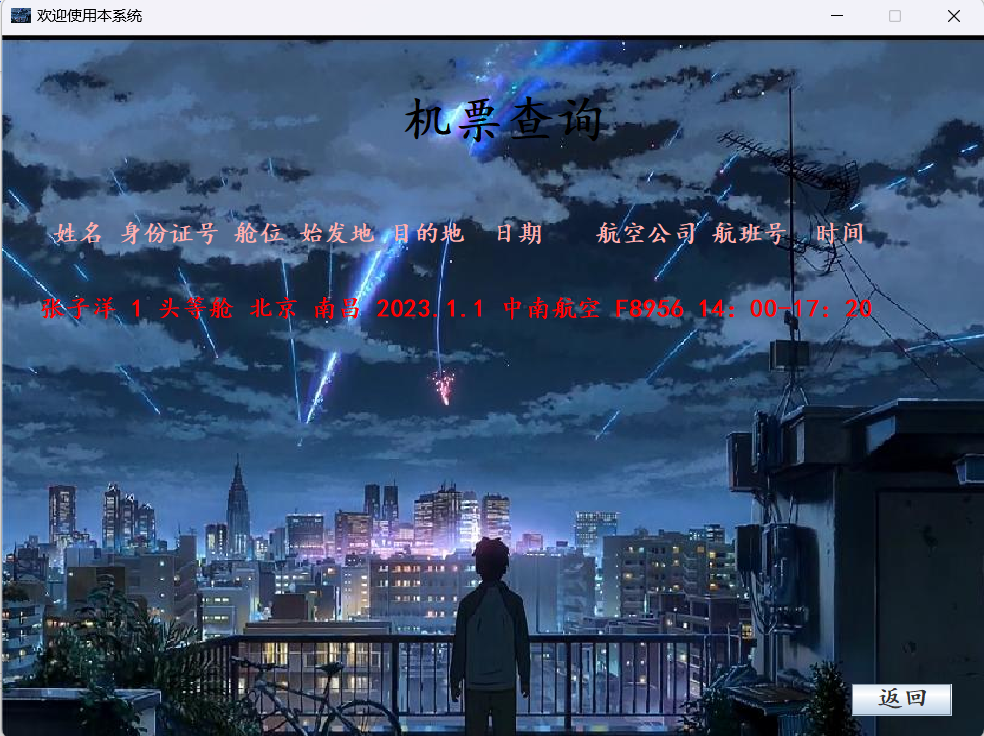
订票：



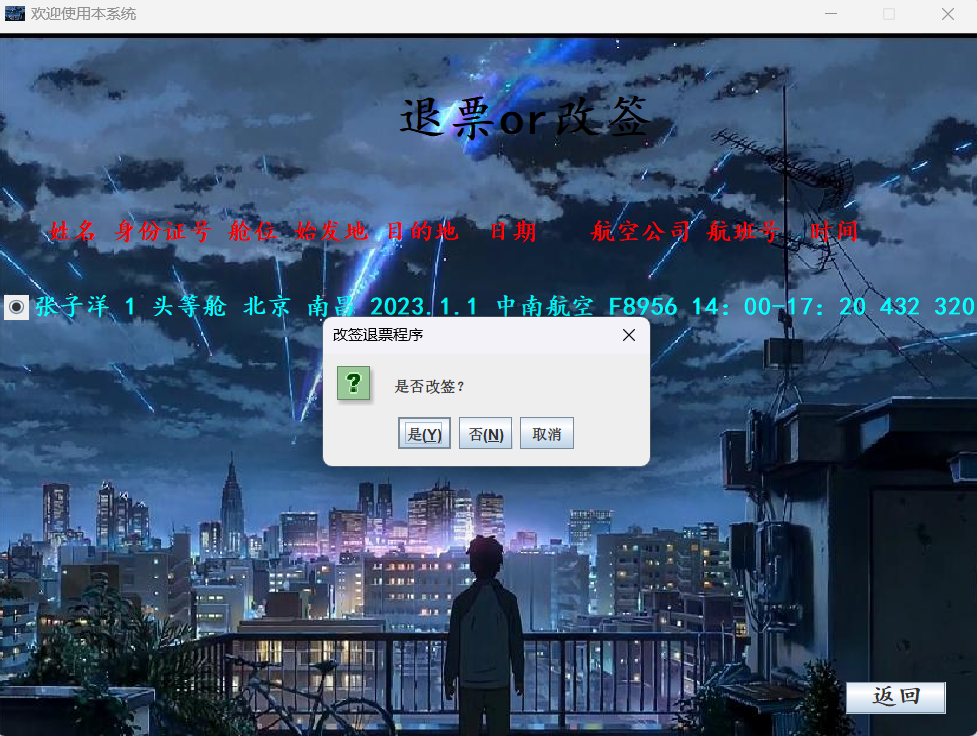


查询：

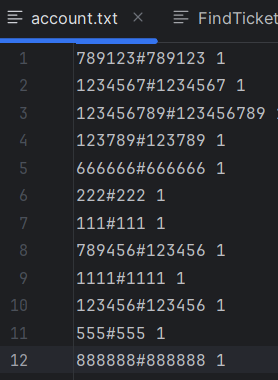




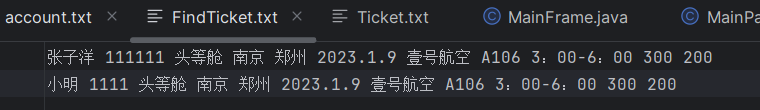
退票改签：



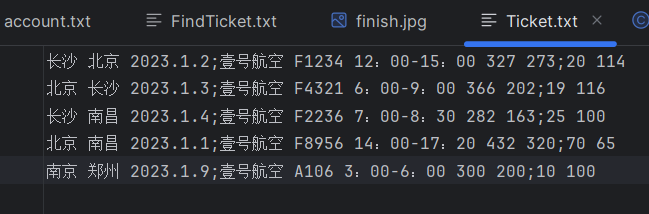
**账号：**



**已购机票：**



**所有机票信息：**



预估：总体实现没有问题，报错流程也没有问题，界面简洁友好。系统多次访问并修改文本文件的内容，对代码效率有一定影响。

## 6. 总结和展望

不足：

①在设计过程中，有一些库中的函数是有上位替代的，所以在写代码的过程中要多查阅函数文档，并对实现类似功能的函数进行横向比较。

②查询功能过于粗暴，输入身份证和姓名就可以查询，忽略了对隐私的保护，应修改为只能查询通过自己账号购买的机票。

③改签功能的实现是先退票，再让用户重新购买，与现实差别过大。

④用户信息直接储存在文本文档中，没有进行加密处理。

⑤注册时没有对帐号重复进行检验，相同的账号可以一直注册，应当有一个查重账号过程，并弹出“该账号已注册”窗口来进行报错。

⑥在购票时应当将机票信息在系统中可视化，客户只能查看ticket文本文件中信息来购票，文本文件不够直观。

总结：

代码比较复杂，要完成的功能较多，ui界面的组件之前虽然使用过，但还是比较生疏。系统的代码逻辑较为清晰，但是还是比较繁琐，尤其是报错与异常，，登录界面比较完善，购票功能基本实现，但可以更加全面。