|  |  |  |  |
| --- | --- | --- | --- |
| 东北林业大学横版组合 | | | |
|  | | | |
| **Java程序设计** | | | |
| **ATM柜员机模拟程序** | | | |
|  | | | |
|  | 姓名： | 张可心，沈阳，李诗怡，刘玉波，程铭 |  |
|  | 专业班级： | 计算机科学与技术2022 |  |

2024年6月

目录

[目录](#_Toc20485)

[1 项目介绍 1](#_Toc26449)

[1.1 项目背景 1](#_Toc13218)

[1.2 项目目标 1](#_Toc32645)

[1.3 功能模块 1](#_Toc4871)

[1.4 业务流程 1](#_Toc5614)

[2 需求分析 4](#_Toc11299)

[2.1 基本需求 4](#_Toc18225)

[2.2 功能性需求 4](#_Toc28082)

[2.3非功能性需求 4](#_Toc8480)

[3 类的设计 5](#_Toc10294)

[3.1 类图 5](#_Toc11331)

[3.2 主要类的说明 6](#_Toc7138)

[4 功能实现 7](#_Toc17060)

[4.1 类的实现 7](#_Toc29357)

[4.2 运行结果展示 8](#_Toc9742)

[5 总结 12](#_Toc28712)

[5.1 项目总结 12](#_Toc6424)

[5.2 心得体会 12](#_Toc11462)

[参考文献 13](#_Toc6577)

[附录 14](#_Toc8383)

# 项目介绍

## 1.1 项目背景

在本项目中，我们小组实现了一款模拟ATM机的系统。现代银行业务中ATM机（自动柜员机）是重要的一部分，提供了用户账户管理、存款、取款、转账等各种便捷服务。本项目旨在模拟ATM机的基本功能，系统可以创建和管理用户账户，实现存款、取款、账户查询、密码修改等功能，还能够对用户数据进行持久化存储，保证系统重启后数据仍然有效。

## 1.2 项目目标

通过本项目的开发，我们希望能够：

1. 熟悉面向对象编程思想；

2. 掌握Java语言的基本语法和编程技巧；

3. 了解如何进行文件读写，实现数据持久化；

4. 增强解决实际问题的能力，提升编码水平。

## 1.3 功能模块

我们将ATM系统划分为以下主要功能模块：

1. 用户账户管理：创建新用户账户，并存储用户的基本信息，包括用户名、卡号、密码和当日取现额度。

2. 用户登录：通过卡号和密码登录系统。

3. 账户操作：登录后执行查询账户余额、存款、取款、转账、修改密码、注销账户和显示所有用户信息等操作。

4. 数据持久化：用户信息存储在文件系统中，确保系统重启后数据依然有效。

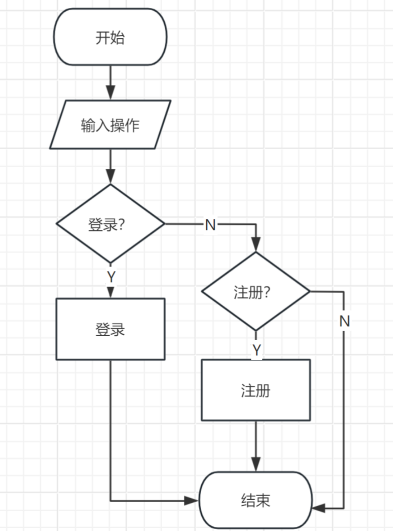
5. 安全性处理：验证密码一致性、转账验证账户有效性等。

6. 用户界面：提供清晰的文本界面引导用户操作。

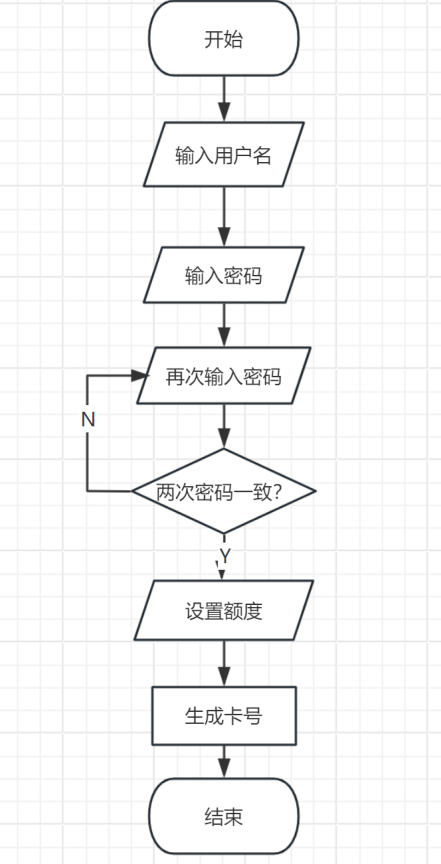
7. 错误处理：处理用户输入无效、账户余额不足等情况。

## 1.4 业务流程

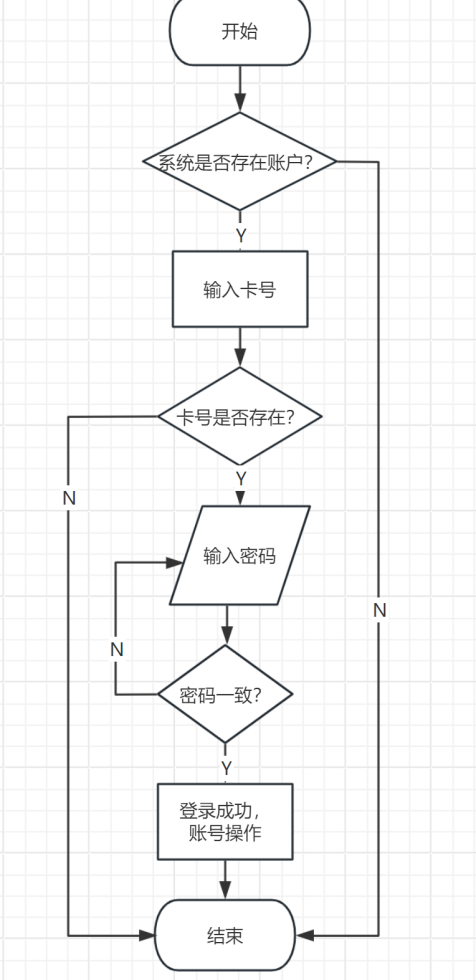
1.主菜单



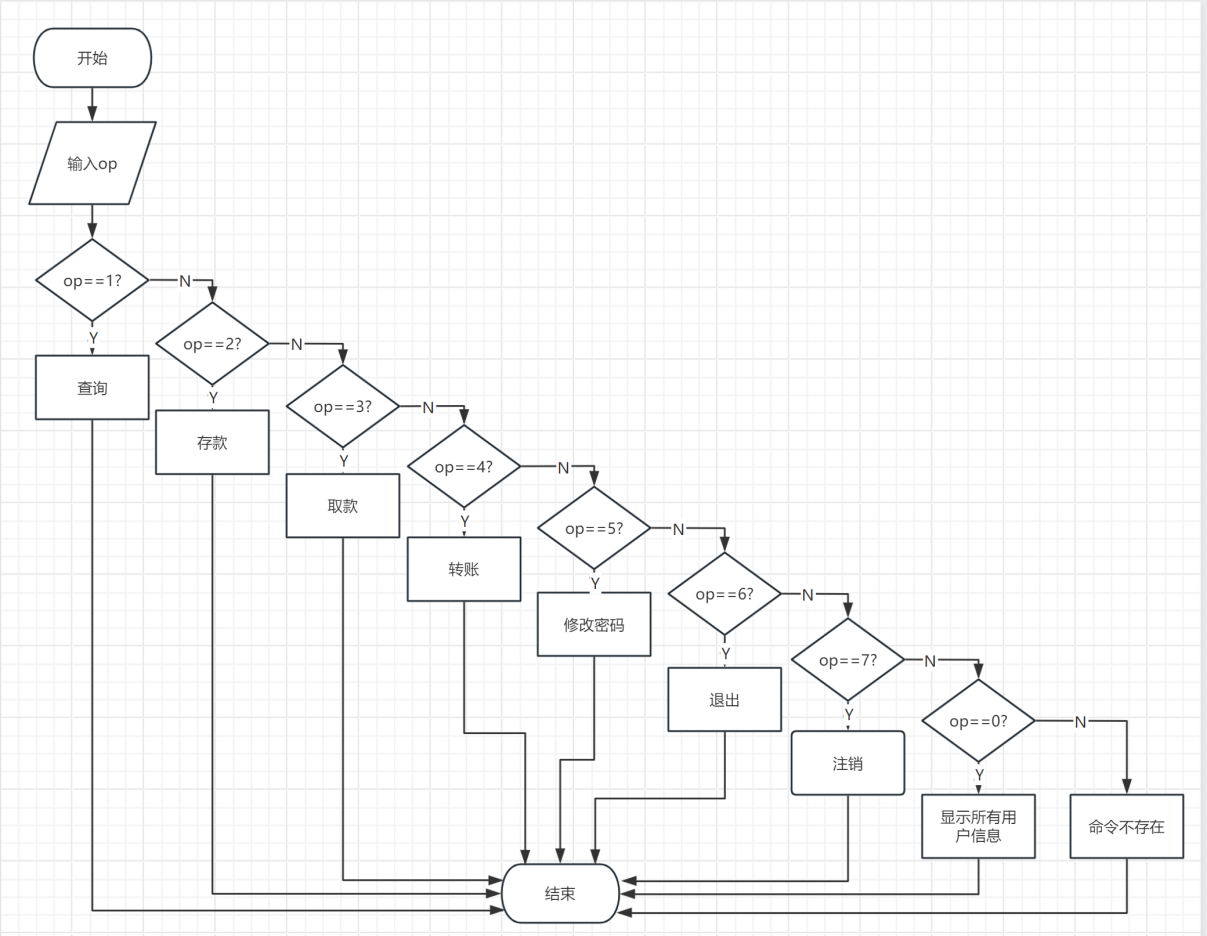
2.账户注册流程



3.用户登录流程



4. 账户操作流程



# 需求分析

## 基本需求

1.用户账户管理：系统需要能够创建新用户账户，并存储用户的基本信息，包括用户名、卡号、密码和当日取现额度，卡号需要随机生成，且保证唯一性。

2.用户登录：用户需要能够通过卡号和密码登录系统。

3.账户操作：用户登录后，应能够执行以下操作：

(1)查询账户余额和当日取现额度。

(2)存款：向账户中添加资金。

(3)取款：从账户中提取资金，需遵守当日取现额度限制。

(4)转账：在系统中的不同账户之间转移资金，需遵守当日取现额度限制。

(5)修改密码：用户能够更新自己的账户密码。

(6)注销账户：用户可以选择注销自己的账户，注销后账户信息将从系统中删除并更新文件。

(7)显示所有用户信息：展示系统中所有用户的基本信息。

## 功能性需求

1.数据持久化：用户信息需要存储在文件系统中，即使系统重启后也能保留。

2.安全性：

(1)密码在注册和修改时需要验证一致性。

(2)转账时需要验证收款人账户的有效性。

3.用户界面：系统应提供清晰的文本界面，指导用户进行每一步操作，与用户友好的交互。

4.错误处理：系统需要能够处理并反馈用户操作中可能出现的错误，例如输入无效、账户余额不足等。

## 2.3非功能性需求

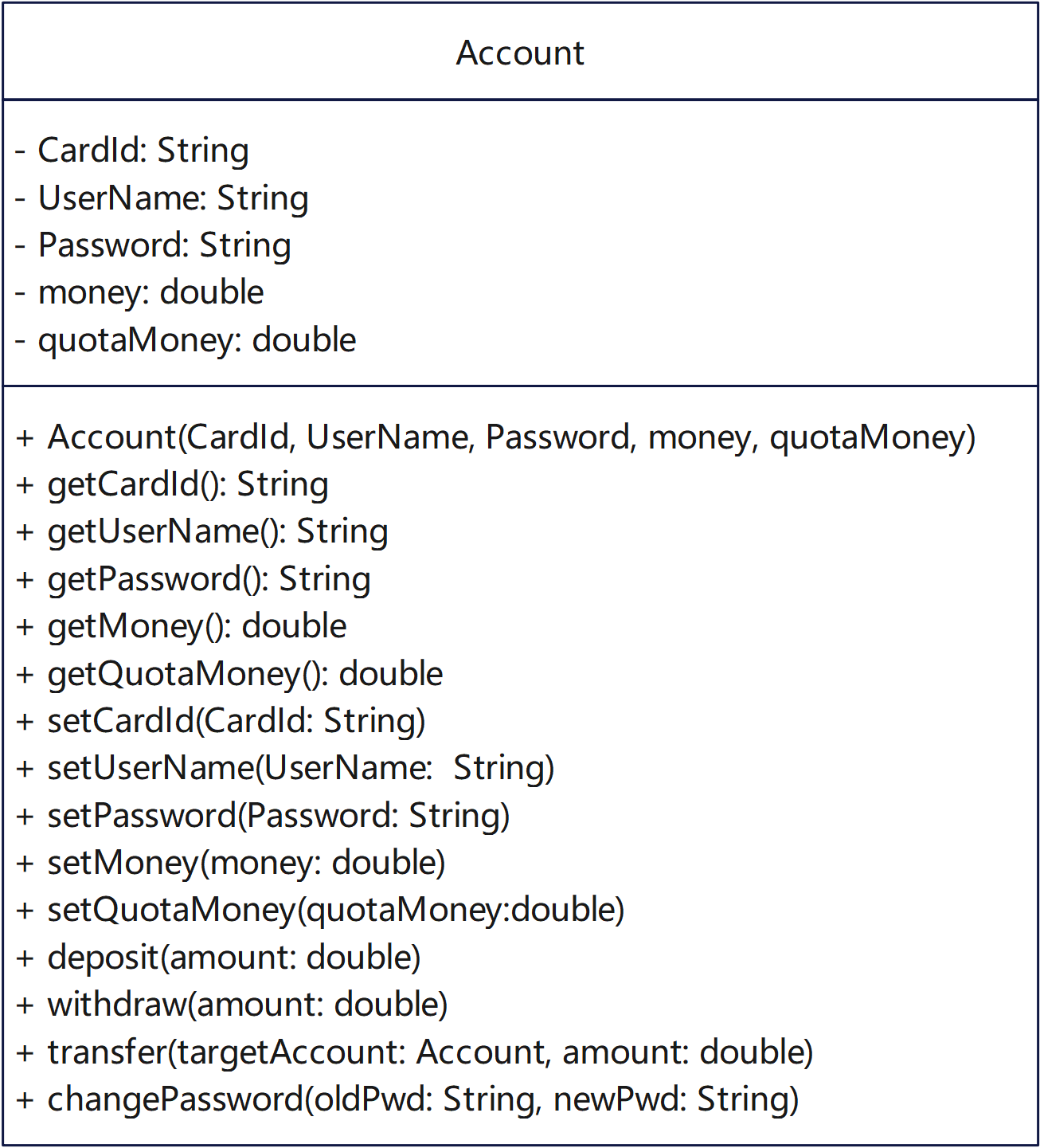
1.可用性：系统应易于使用，操作流程直观。

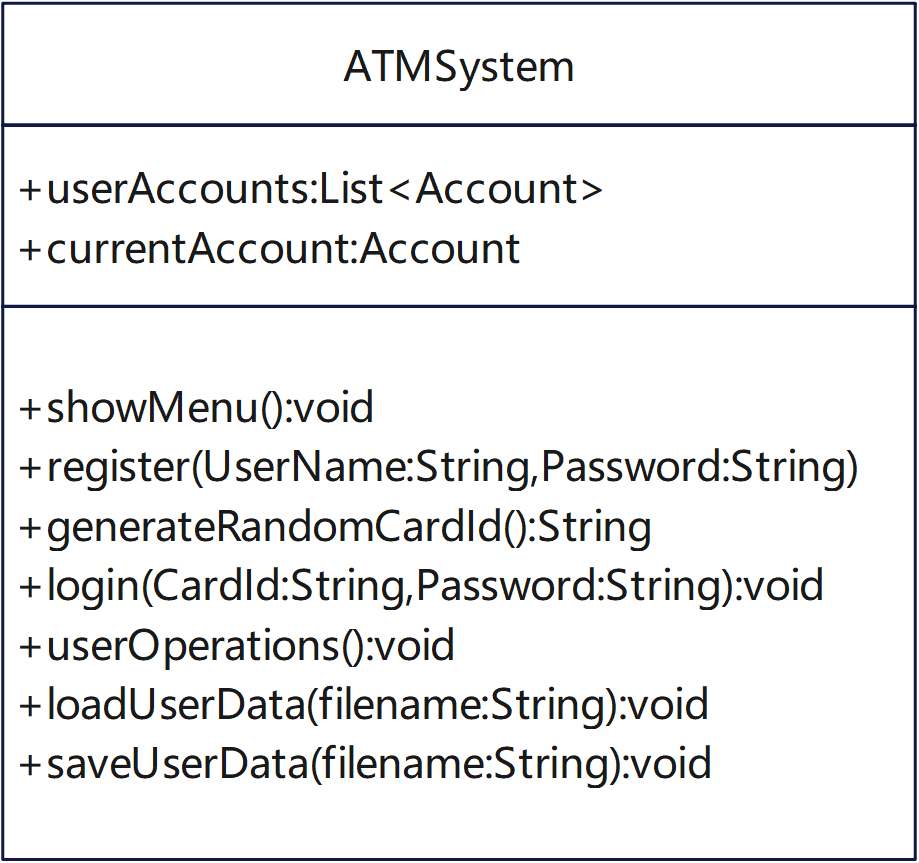
2.健壮性：系统能够处理异常情况，如用户输入错误、文件操作失败等。

3.扩展性：代码结构应便于未来添加新功能或进行修改。

# 类的设计

## 类图





## 主要类的说明

1.Account类

这个类代表了一个银行账户，包含以下属性：

- CardId：卡号，字符串类型，唯一标识一个账户。

- UserName：用户名，字符串类型。

- Password：密码，字符串类型，用户登录验证信息。

- money：账户余额，双精度浮点数类型。

- quotaMoney：当日取现额度，双精度浮点数类型。

2.Account类的主要方法

- 构造方法：初始化账户信息。

- Getter和Setter方法：获取和设置账户属性。

- deposit：向账户存入一定金额。

- withdraw：从账户中提取一定金额，检查余额和当日取现额度。

- transfer：向另一账户转账，验证目标账户有效性。

- changePassword：修改账户密码，验证原始密码和新密码一致性。

3.ATMSystem类

这个类实现了一个简单的ATM系统，主要功能如下：

- userAccounts：集合，用于存储所有用户账户信息。

- currentAccount：记录当前登录用户的信息。

- showMenu：打印主菜单，供用户选择登录或注册。

- register：注册新用户。

- generateRandomCardId：生成唯一的8位随机卡号。

- login：用户登录。

- userOperations：登录成功后，用户可选择的操作菜单。

- loadUserData：从文件中读取用户数据。

- saveUserData：保存用户数据到文件中。

# 功能实现

## 类的实现

1.以下是Account类的核心实现代码：

import java.util.ArrayList;

import java.util.List;

public class Account {

private String cardId;

private String userName;

private String password;

private double balance;

private double quotaMoney;

private List<String> transactionHistory;

public Account(String cardId, String userName, double initialBalance, double quotaMoney, String password) {

this.cardId = cardId;

this.userName = userName;

this.balance = initialBalance;

this.quotaMoney = quotaMoney;

this.password = password;

this.transactionHistory = new ArrayList<>();

}

public String getCardId() { return cardId; }

public String getUserName() { return userName; }

public double getBalance() { return balance; }

public double getQuotaMoney() { return quotaMoney; }

public boolean checkPassword(String password) { return this.password.equals(password); }

public void deposit(double amount) { if (amount > 0) { balance += amount; transactionHistory.add("Deposited: " + amount); } }

public boolean withdraw(double amount) { if (amount > 0 && balance >= amount && quotaMoney >= amount) { balance -= amount; quotaMoney -= amount; transactionHistory.add("Withdrew: " + amount); return true; } return false; }

public boolean transfer(Account recipient, double amount) { if (amount > 0 && balance >= amount) { balance -= amount; recipient.deposit(amount); transactionHistory.add("Transferred: " + amount + " to " + recipient.userName); return true; } return false; }

public boolean changePassword(String oldPassword, String newPassword) { if (checkPassword(oldPassword) && !newPassword.equals(oldPassword)) { password = newPassword; return true; } return false; }

public void printTransactionHistory() { for (String transaction : transactionHistory) System.out.println(transaction); }

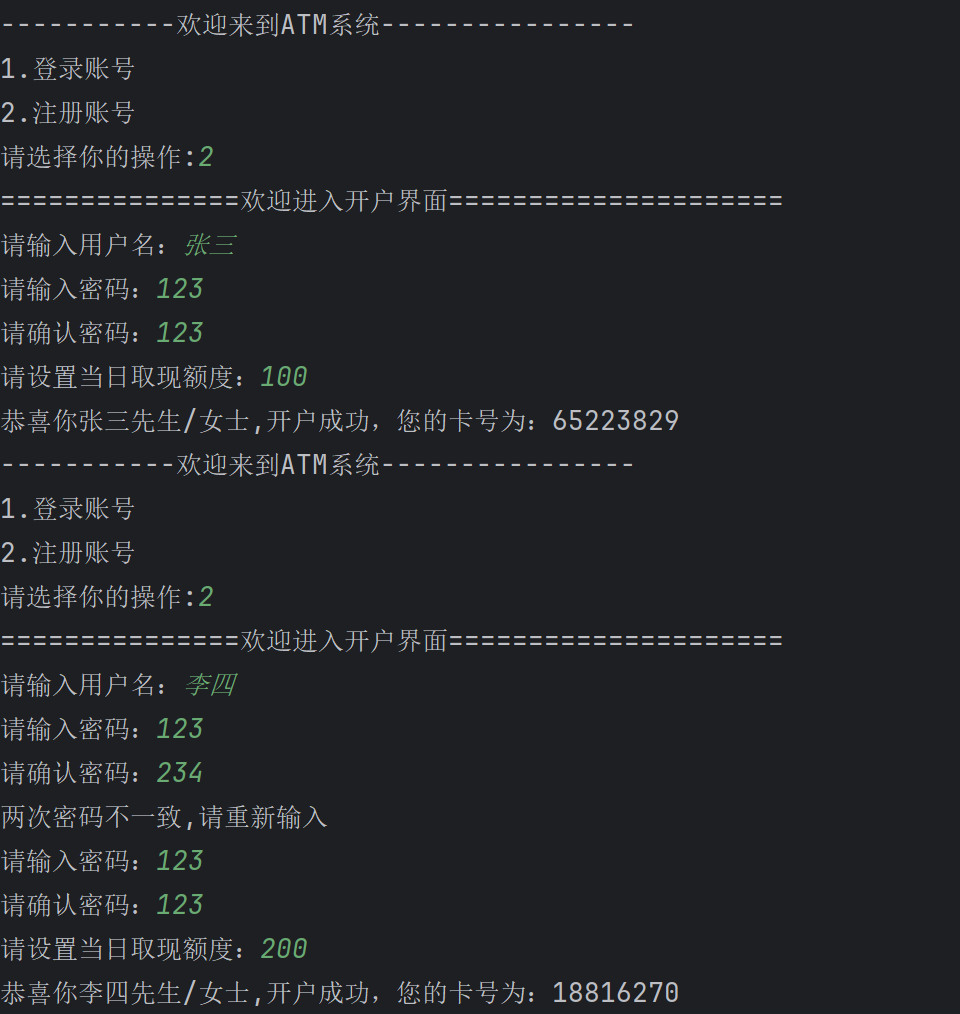
}

// ATMSystem.java 此类中主要方法实现略

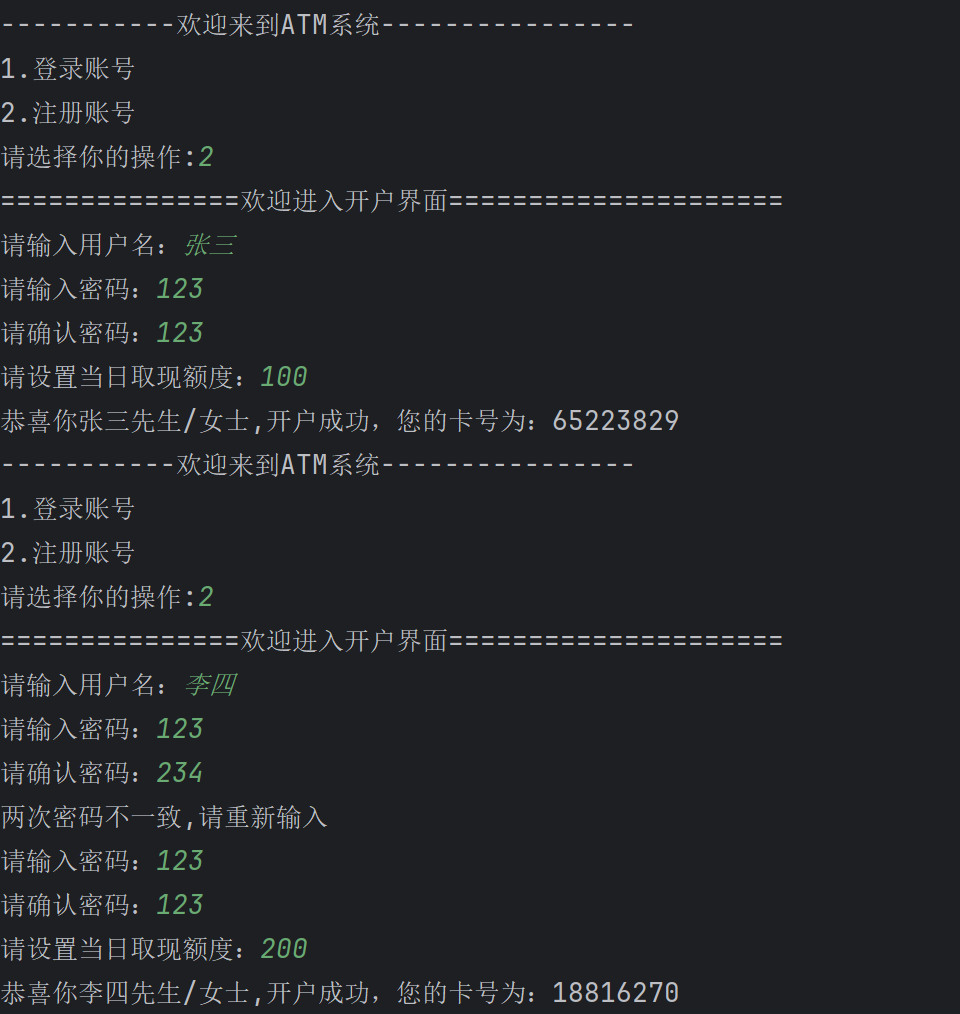
2.ATMSystem核心方法流程图

## 运行结果展示

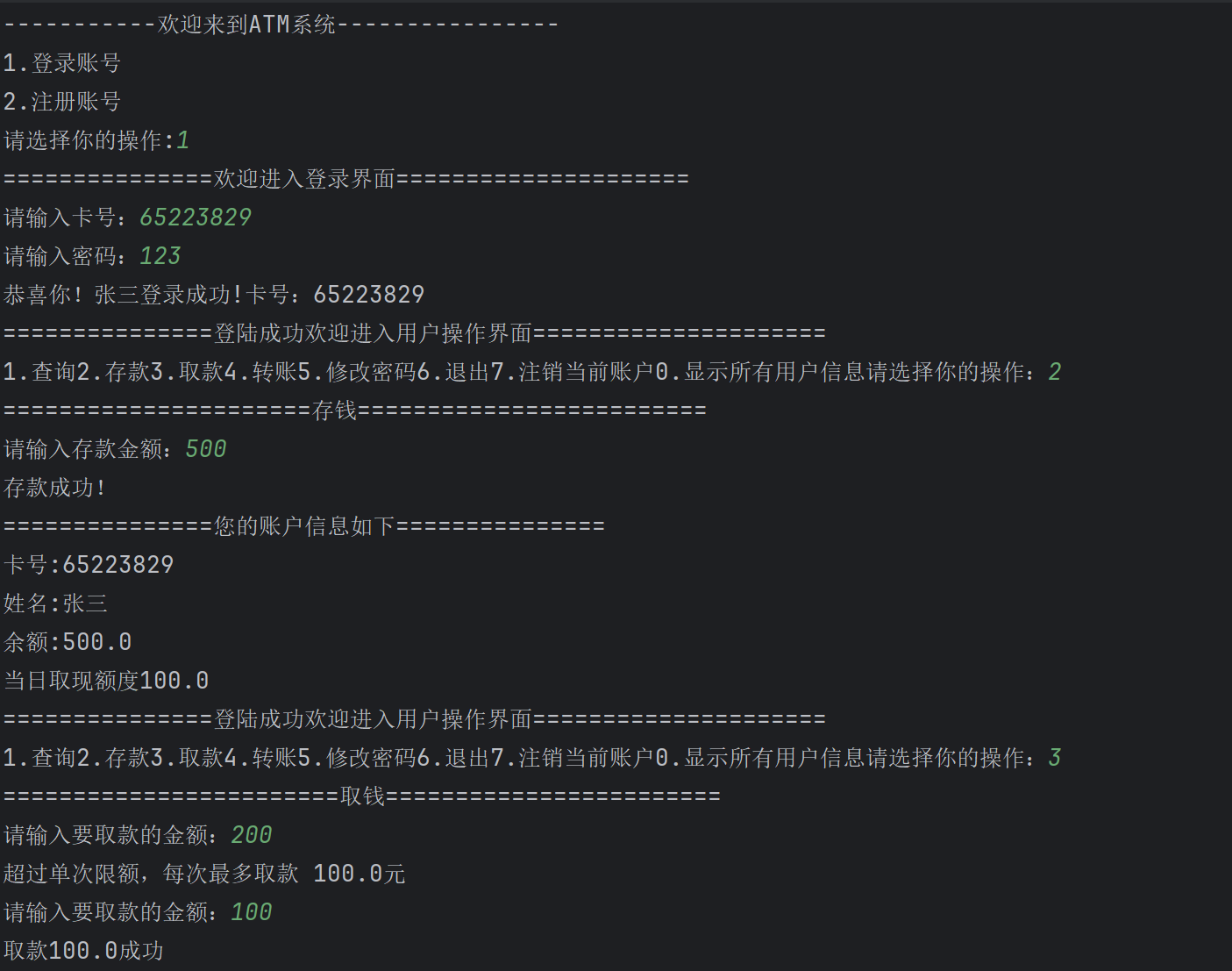
1. 主菜单展示：



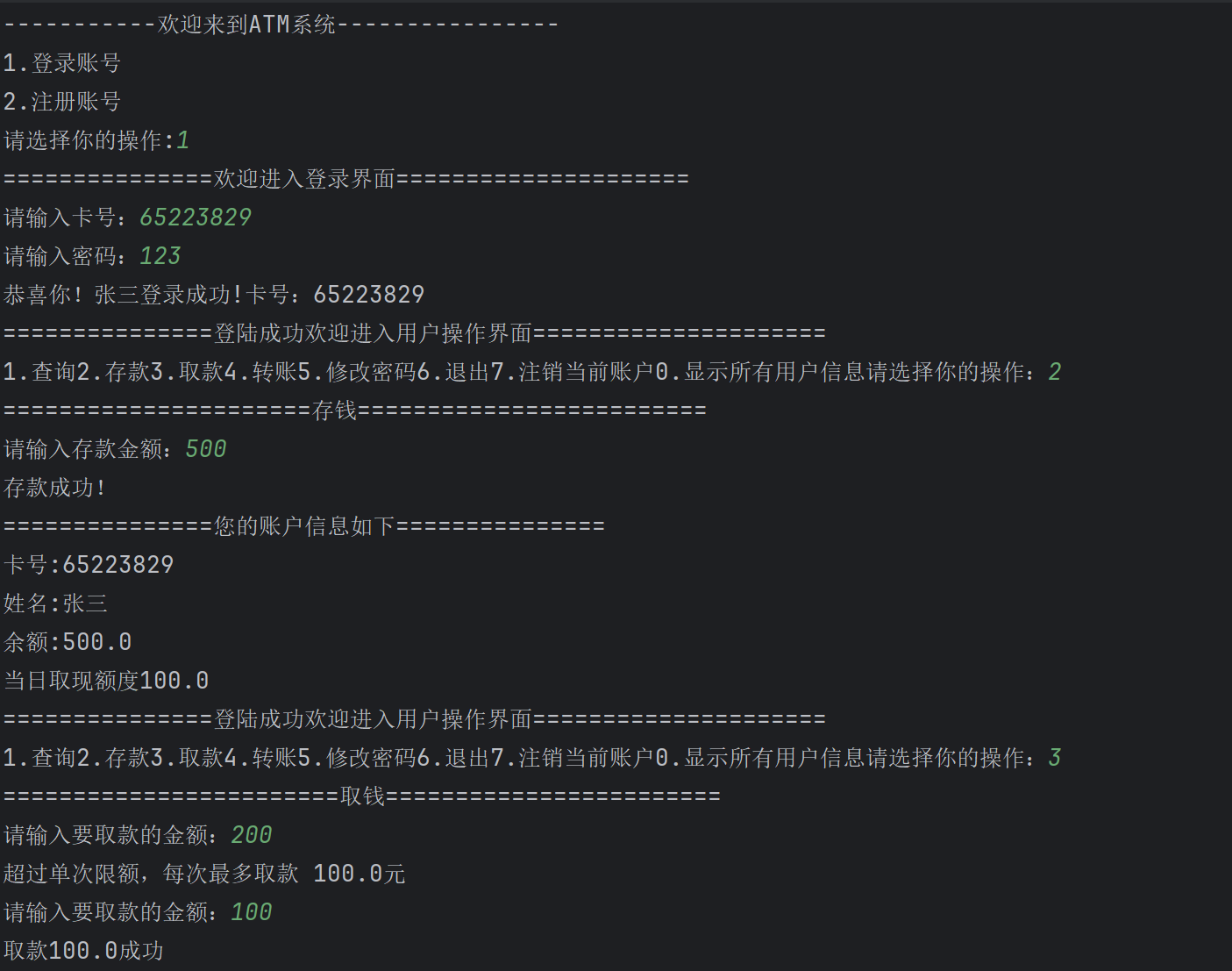
2. 注册新用户：



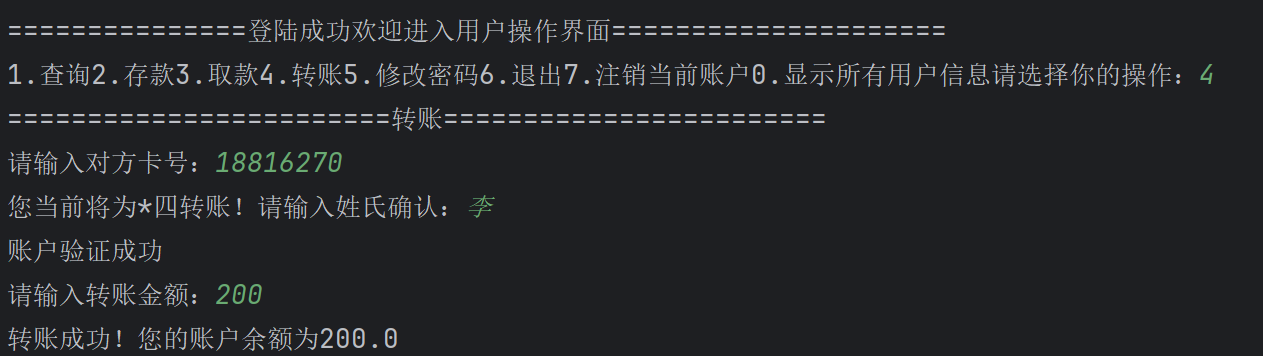
3. 用户登录及操作：



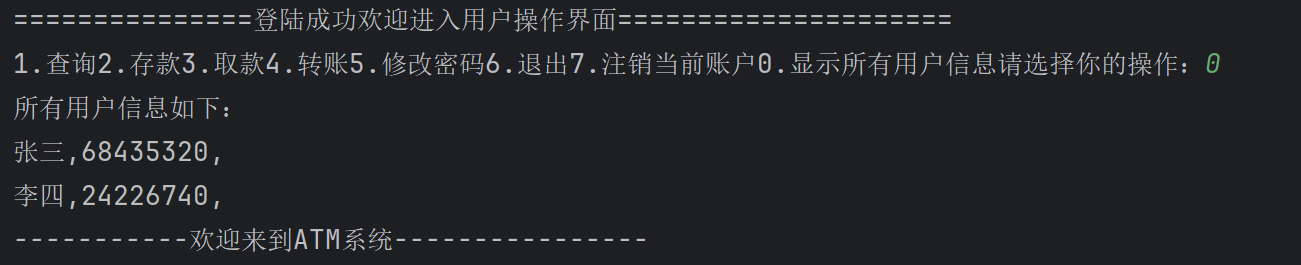
4.存款与取款



1. 转账



1. 显示用户信息



1. 注销用户



# 总结

## 项目总结

通过本项目的开发，我们成功实现了一款模拟ATM机的系统，涵盖了用户账户管理、存款、取款、转账、密码修改等核心功能。实现了数据的持久化存储，确保系统重启后数据依然有效。系统在异常处理方面处理了用户输入无效、账户余额不足等情况。未来工作中，可以进一步优化系统的用户界面，实现更多功能如多币种支持、多层次用户权限管理等。

## 心得体会

在本项目中，我们小组深刻体会到了团队协作的重要性。每个团队成员分工合作，共同完成了需求分析、设计、编码和测试等各个环节。通过本次开发，我们将课堂上学到的Java知识应用到实际项目中，加深了对知识的理解和掌握。我们不仅巩固了面向对象编程的基本概念，还学会了如何将抽象的需求转化为具体的代码实现。在错误处理和数据持久化方面的实际操作也让我们受益匪浅，锻炼了独立思考和解决问题的能力。同时，我们理解了用户体验在软件开发中的重要性，懂得了如何从用户的角度考虑设计。总的来说，本项目为我们提供了一个良好的实践平台，让我们在动手操作中不断成长。

未来我们将考虑增加更多实用的功能，如多币种支持、多层次用户权限管理、转账限额、历史交易记录查询等，以提升系统的实用性。同时我们会进一步加强安全措施，持续优化数据库查询和交易处理的性能。随着技术的发展，我们会探索将新技术如AI辅助决策融入到系统中。

参考文献

[1]胡向前.基于ITIL的银行业IT服务管理系统设计与实现[D].复旦大学,2010.

[2]韩小龙,司珍,吕晓峰,梁博,冯元,方静,李超.基于面向对象编程的Java语言程序设计方法分析[J].集成电路应用,2024,41(1):228-229

[3]柳怡.某基层银行财务管理系统的设计与实施[D].电子科技大学,2014(03).

附录

1.ppt   
2.小组讨论记录图片  
3.资料论文和其他参考代码