**江苏大学京江学院**

**课程设计报告**

**设计题目: 简易银行管理系统**

**院 系: 电气信息工程学院**

**专业班级:**

**学 号:**

**学生姓名:**

**指导教师: 段学庆**

**2024年 1 月 1 日**

目 录

[一、问题描述 3](#_Toc27950046)

[二、需求分析 3](#_Toc27950047)

[三、总体设计 4](#_Toc27950048)

[四、详细设计 4](#_Toc27950049)

[五、上机操作 10](#_Toc27950050)

[六、总结及参考文献 11](#_Toc27950051)

一、问题描述

设计一个简易的银行管理系统，通过命令行交互实现以下基本功能：

**1.用户账户管理**

用户可以注册账户，包括用户名、密码等信息。

已注册用户可以登录系统。

**2.账户操作**

登录后，用户可以进行存款、取款和查询余额的操作。

每个用户的账户信息应该包括账户号、余额等。

**3.转账功能**

用户可以进行账户之间的转账操作，包括指定目标账户和转账金额。

**4.交易记录**

系统应该记录每笔交易的相关信息，包括交易时间、金额、类型等。

**5.安全性考虑**

考虑密码安全，可以选择使用某种安全手段保存用户密码。

**6.数据持久化**

用户的账户信息和交易记录应该保存在文件中，以便下次程序运行时可以重新加载数据。

**7.界面友好**

通过命令行界面，使用户能够方便地使用系统的各项功能。

**8.错误处理**

考虑用户输入错误的情况，给予相应的提示和处理。

**9.退出系统**

用户可以选择安全退出系统。

二、需求分析

**1.用户账户管理**

建立customer.txt 文件用于存储用户的id,用户的密码;每次建立一个账户,会建立其对应的交易记录文件,例如当我创建1的交易文件时,就会创建 1用户交易记录.txt

**2.账户操作**

登入的时候,会把 用户信息.txt文件的信息放进去,以达到初始化。

当用户需要存款的时候，会先对类里面进行操作，然后将此的交易记录保存到.txt文件中，

并且记录了操作时间。

当用户想要查看余额的时候，也会将此操作记录到.txt文件中

**3.转账功能**

在转账上，我为转账设置了一些安全的措施：1.不能向自己转账（无效的），2，不得向存在的用户转账，3.转账的金额小于等于余额。

转账失败会重新开始转账，通过while循环来进行控制。

1. 交易记录

我会对如下的操作进行记录：

* 存款
* 查询余额
* 转账
* 取款

在此基础上，我加了时间，用记录操作的关键点。

**5.安全性考虑**

我对用户的密码进行封装，每个用户的密码是6位数。

用户的密码保存在文件中，用户登入时会对密码逐个排查，只有密码，id正确才可以进行操作。

**6.数据持久化**

通过使用文件操作函数达到数据的持久化。

**7.界面友好**

都是通过输入简答的数字，实现界面的友好交互。会在黑窗口上出现相应的数字。

1. **错误处理**

**当用户输入错误的时候，我会用个循环条件不断的要求用户重新输入，直到输入正确。**

**9.退出系统**

**当用户退出系统的时候，我会把所有的数据存放到信息表里面。**

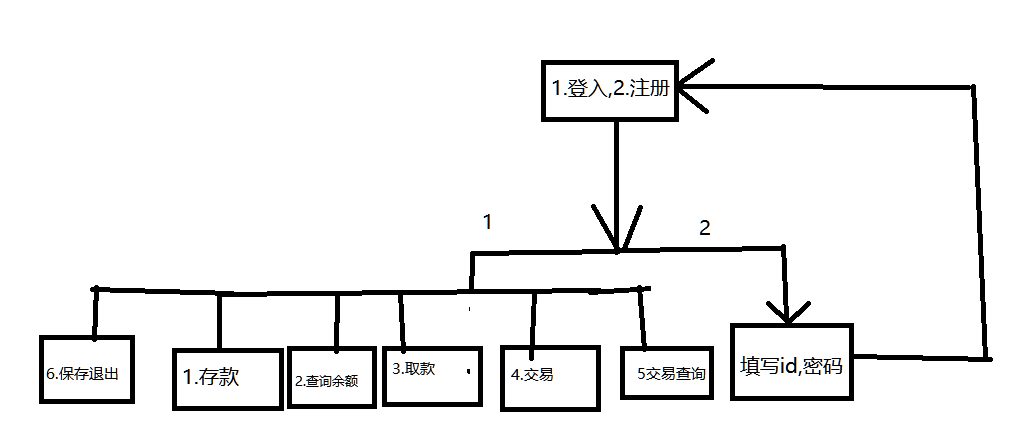
三、总体设计

1.系统总体结构设计：包括用户操作结构设计，业务员操作结构设计，系统输出结构设计。

2.功能模块设计：主要实现储户线上操作和业务员管理协助的功能。

3.系统需求分析：功能清晰，数据分为账号管理、储存管理、取款管理。

4.安全性和稳定性：充分考虑了文件的安全性、一致性、稳定性和可靠性。



四、详细设计

**总体功能代码：**

#define \_CRT\_SECURE\_NO\_WARNINGS 1

#include"Bank.h"

int get\_pos(int id, Bank\* bank)

{

for (int x = 0; x < bank->customer.size(); ++x)

{

if (bank->customer[x].id == id) return x;

}

return -1;

}

void modify(Bank\* bank,double money,string record,int pos)

{

bank->customer[pos].all\_money = money;

bank->customer[pos].trade\_record= record;

}

int Is\_same(int \_id)//判断id是否唯一

{

int id;

int passport;

FILE\* fp = fopen("C:\\面向对象\\customer.txt", "r");

while (fscanf(fp,"%d %d",&id,&passport) != EOF)

{

if (id == \_id)return 1;

}

return 0;

}

void zhuce()

{

cout << "请输入注册id:"<<endl;

int id;

cin >> id;

cout << "请输入密码(不大于6位的数字):" << endl;

int passport;

cin >> passport;

while (Is\_same(id))

{

cout << "请重新输入注册id,此id已经被使用:" << endl;

cin >> id;

}

//记录用户密码,,id

FILE\* fp = fopen("C:\\面向对象\\customer.txt", "a");

if (!fp)perror("fopen");

fprintf(fp, "%d %d\n", id, passport);

fclose(fp);

string strr = "C:\\面向对象\\用户交易\\" + to\_string(id) + "用户交易记录.txt";

const char\* str = strr.data();

FILE\* fppp = fopen(str, "a");

fclose(fppp);

fppp = nullptr;

}

Customer push(int \_id,double \_all\_money,string str)

{

Customer customer;

customer.id = \_id;

customer.all\_money += \_all\_money;

customer.trade\_record = str;

return customer;

}

void Init(Bank\* bank)//数据导入

{

string str = "C:\\面向对象\\用户信息\\用户信息.txt";

const char\* strs = str.data();

FILE\* fpp = fopen(strs, "r");

int \_id; int passport;

FILE\* fp = fopen("C:\\面向对象\\customer.txt", "r");

if (!fp)perror("fopen");

int x = 0;

while (fscanf(fp, "%d %d\n", &\_id, &passport) != EOF)

{

double money = 0;

string record;

string add;

fscanf(fpp, "%d %lf", &\_id, &money);//载入用户信息,主要是对钱的载入

string strr = "C:\\面向对象\\用户交易\\" + to\_string(\_id) + "用户交易记录.txt";

const char\* strs = strr.data();

ifstream fppp(strs);

while (getline(fppp, record)) {

add += record;

add += " ,";

}

Customer customer = push(\_id, money, add);

bank->customer.push\_back(customer);

fppp.close();

int pos = get\_pos(\_id, bank);

bank->customer[pos].all\_money = money;//导入钱

}

fclose(fpp);

}

void deposit(Bank\* bank, int id)

{

auto now = std::chrono::system\_clock::now();

std::time\_t now\_time = std::chrono::system\_clock::to\_time\_t(now);

char\* str = ctime(&now\_time);

string time = str;//时间

cout << "请输入要存多少钱:" << endl;

double money;

cin >> money;

string strrs = to\_string(money);

string record = time + "存款" + strrs+'\n';

//修改数据

int pos = get\_pos(id, bank);

bank->customer[pos].all\_money += money;

bank->customer[pos].trade\_record = record;

string strr = "C:\\面向对象\\用户交易\\" + to\_string(id) + "用户交易记录.txt";

const char\* road = strr.data();

FILE\* fp = fopen(road, "a");

fprintf(fp, "%s", &record[0]);

fclose(fp);

}

void withdraw(Bank\* bank, int id)

{

auto now = std::chrono::system\_clock::now();

std::time\_t now\_time = std::chrono::system\_clock::to\_time\_t(now);

char\* str = ctime(&now\_time);

string time = str;//时间

int pos = get\_pos(id, bank);

double money = bank->customer[pos].all\_money;

cout << "您的余额为" << money << endl;

cout << "要取出多少钱" << endl;

double \_money;

cin >> \_money;

while (money - \_money < 0)

{

cout << "余额不足重新输入" << endl;

cin >> \_money;

}

money -= \_money;

string strs = to\_string(money);

string record = time + "取款" + strs+'\n';

//修改数据

modify(bank, money, record, pos);

string strr = "C:\\面向对象\\用户交易\\" + to\_string(id) + "用户交易记录.txt";

const char\* road = strr.data();

FILE\* fp = fopen(road, "a");

fprintf(fp, "%s", &record[0]);

fclose(fp);

}

void show\_money(Bank\* bank, int id)

{

auto now = std::chrono::system\_clock::now();

std::time\_t now\_time = std::chrono::system\_clock::to\_time\_t(now);

char\* str = ctime(&now\_time);

string time = str;//时间

int pos = get\_pos(id, bank);

double money = bank->customer[pos].all\_money;

cout << "您的余额为" << money << endl;

string strs = to\_string(money);

string record = time + "查寻余额" + strs+'\n';

string strr = "C:\\面向对象\\用户交易\\" + to\_string(id) + "用户交易记录.txt";

const char\* road = strr.data();

FILE\* fp = fopen(road, "a");

fprintf(fp, "%s", &record[0]);

fclose(fp);

}

void change\_money(Bank\* bank, int id)

{

auto now = std::chrono::system\_clock::now();

std::time\_t now\_time = std::chrono::system\_clock::to\_time\_t(now);

char\* str = ctime(&now\_time);

string time = str;//时间

cout << "请输入你要转账的id:" << endl;

int \_id;

cin >> \_id;

int pos = get\_pos(\_id, bank);

while (pos == -1&&id!=\_id)

{

cout << "该用户不存在,重新输入或者不能给自己转账,请重新输入id" << endl;

cin >> \_id;

pos = get\_pos(\_id, bank);

}

int \_pos = get\_pos(id, bank);

double money = bank->customer[\_pos].all\_money;

cout << "您的余额为" << money << endl;

cout << "要转出多少钱" << endl;

double \_money;

cin >> \_money;

while (money - \_money < 0)

{

cout << "余额不足重新输入" << endl;

cin >> \_money;

}

money -= \_money;

string num = to\_string(\_money);//转的钱

string people = to\_string(\_id);

string record = time + "转帐" + num+ "给"+people;

//修改数据

bank->customer[\_pos].all\_money -= \_money;

bank->customer[pos].all\_money += \_money;

string strr = "C:\\面向对象\\用户交易\\" + to\_string(id) + "用户交易记录.txt";

const char\* road = strr.data();

FILE\* fp = fopen(road, "a");

fprintf(fp, "%s", &record[0]);

fclose(fp);

}

void show\_trade(Bank\* bank, int id)

{

string add;

string record;

cout << "以下为用户" << id << "的交易记录" << endl;

string strr = "C:\\面向对象\\用户交易\\" + to\_string(id) + "用户交易记录.txt";

ifstream fppp(strr);

while (getline(fppp, record)) {

add += record;

add += " \n";

}

cout << add<<endl<<endl;

fppp.close();

}

void save(Bank\* bank, int id)//保存余额

{

int pos = get\_pos(id,bank);

string str = "C:\\面向对象\\用户信息\\用户信息.txt";

const char\* Road= str.data();

FILE\* fp = fopen(Road, "w");

for (int x = 0; x < bank->customer.size(); ++x)

{

fprintf(fp, "%d %lf\n", bank->customer[x].id, bank->customer[x].all\_money);

}

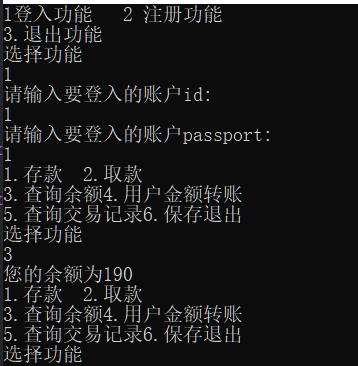
fclose(fp);

return;

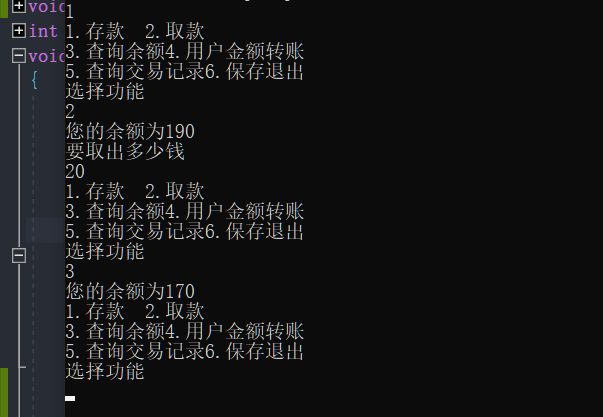
}

五、上机操作

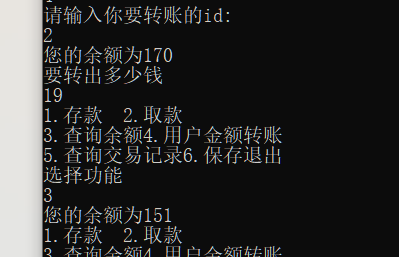
1.登入并查款：



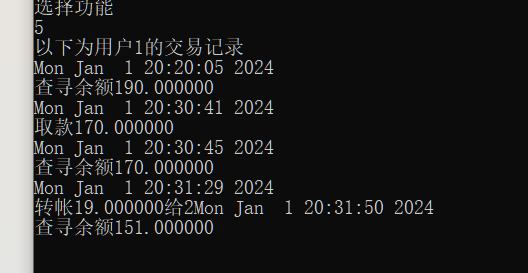
1. 取款并查询余额



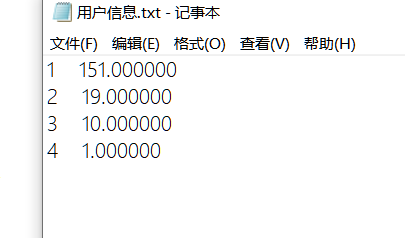
1. 转账并查询



1. 交易记录查询



1. 退出保存



六、总结及参考文献

总结：

银行管理系统是一种用于管理银行业务的计算机软件系统。它的设计需要考虑到银行业务的复杂性和多样性，以及对安全性、稳定性和可靠性的高要求。

首先，包括储户子系统、业务员子系统等部分。这种架构可以有效地处理大量的并发请求，保证系统的稳定运行。

其次，系统总体结构设计包括用户操作结构设计，业务员操作结构设计，系统输出结构设计三个部分。用户操作结构设计主要考虑到用户的使用习惯和操作便利性，业务员操作结构设计则需要考虑到业务员的工作流程和操作效率，系统输出结构设计则主要关注系统的输出格式和内容的准确性。

在功能模块设计方面，银行管理系统主要实现储户线上操作和业务员管理协助的功能，包括开户、存款、取款、查询、用户管理、储户信息等六个功能模块。这些功能模块可以满足银行业务的基本需求，提供便捷的服务给储户和业务员。

数据库设计是银行管理系统设计的重要部分，系统的数据库包含储户信息表、银行卡信息表、存款信息表、取款信息表、业务员信息表等。这些表格可以存储和管理银行业务所需的各种数据，保证数据的完整性和一致性。

系统需求分析是设计过程中的重要步骤，系统功能清晰，数据分为账号管理、储存管理、取款管理三类。通过需求分析，可以明确系统需要实现的功能，为后续的设计和开发提供指导。

最后，安全性和稳定性是银行管理系统设计的重要考虑因素。在设计过程中，需要充分考虑到数据库的安全性、一致性、稳定性和可靠性的问题，采取有效的措施来防止数据的丢失和篡改。

**参考文献：**

**陕西比特教育<<c语言从入门到起飞>>**

**谭浩强<<c++程序设计>>**