北京邮电大学 计算机学院

《面向对象程序设计与实践》

实验报告

姓名 王睿嘉

学号2015211906

班级2015211307

目录

[银行门户 3](#_Toc502167101)

[一、 需求分析 3](#_Toc502167102)

[二、 功能说明 3](#_Toc502167103)

[三、 软件设计 5](#_Toc502167104)

[1. 高层数据结构设计 5](#_Toc502167105)

[2. 系统模块划分 7](#_Toc502167106)

[3. 高层算法设计 8](#_Toc502167107)

[四、 性能测试 9](#_Toc502167108)

[电商平台 16](#_Toc502167109)

[一、 需求分析 16](#_Toc502167110)

[二、 功能说明 16](#_Toc502167111)

[三、 软件设计 18](#_Toc502167112)

[1. 高层数据结构设计 18](#_Toc502167113)

[2. 系统模块划分 26](#_Toc502167114)

[3. 高层算法设计 27](#_Toc502167115)

[四、 性能测试 29](#_Toc502167116)

银行门户

1. 需求分析

该门户是一个单独的程序，至少支持以下功能：

1. 注册银行卡，且银行可选；
2. 修改银行卡密码；
3. 存款取款。

其中，银行卡至少要有卡号，密码，所属银行名称，持卡人身份证号，卡内金额等内容。

附加要求：

1. 把所有银行名写入文件（或数据库）。注册银行卡时，只能选择已存在的银行进行注册；
2. 做好错误场景的处理。例如，读银行文件错误、输入数据不合法等。
3. 功能说明

|  |  |  |  |
| --- | --- | --- | --- |
| 模块功能 | 输入 | 处理方式 | 输出 |
| 注册模块 | 注册银行选择  姓名  身份证号  预留手机号  密码  确认密码 | 判断银行选择是否正确；  判断身份证号是否为18位，允许字母出现；  判断手机号是否为11位数字；  判断密码长度是否为6位，且仅由数字组成；  确认密码与原密码是否相同 | 若银行选择有误，输出提示信息，并要求重新选择；若身份证号不符合规定，输出提示信息，并要求重新输入；若预留手机号不符合规定，输出提示信息，并要求重新输入；  若密码不符合规定，输出提示信息，并要求重新输入；若确认密码与原密码不符，输出提示信息，并要求重新设置密码；若信息均正确，注册成功，并输出相应提示信息 |
| 存款模块 | 卡号  密码  存款金额 | 判断账户是否处于冻结状态；  判断银行卡是否存在；  判断密码是否正确；  判断存款金额是否为正的小数或整数 | 若账户处于冻结状态，则无法进行存款操作，输出相应提示信息；若银行卡不存在，输出提示信息，并要求重新输入；若密码不正确，输出提示信息，并要求重新输入；若密码错误次数超过三次，账户被冻结，输出相应提示信息；若存款金额不符合规定，输出提示信息，并要求重新输入；若信息均正确，存款成功，并输出相应提示信息 |
| 取款模块 | 取款金额 | 判断账户是否处于冻结状态；  判断取款金额是否为正的小数或整数，且是否超过每日最大额度（5000元）和账户余额的较小值 | 若账户处于冻结状态，则无法进行取款操作，输出相应提示信息；若取款金额不符合规定，输出提示信息，并要求重新输入；若信息均正确，取款成功，并输出相应提示信息 |
| 修改密码模块 | 新密码  确认密码 | 判断账户是否处于冻结状态；  判断新密码长度是否为6位，且仅由数字组成；  判断确认密码与新密码是否相同 | 若账户处于冻结状态，则无法进行修改密码操作，输出相应提示信息；若新密码不符合规定，输出提示信息，并要求重新输入；若确认密码与新密码不符，输出提示信息，并要求重新设置密码；若信息均正确，修改密码成功，并输出相应信息 |
| 修改预留手机模块 | 新手机号 | 判断账户是否处于冻结状态；  判断新手机号是否为11位数字 | 若账户处于冻结状态，则无法进行修改预留手机操作，输出相应提示信息；若新手机号不符合规定，输出提示信息，并要求重新输入；若信息均正确，修改预留手机成功，并输出相应信息 |
| 查询交易记录模块 | 无 | 判断账户是否处于冻结状态 | 若账户处于冻结状态，则无法进行查询交易记录操作，输出相应提示信息；否则，打印每笔交易的时间及金额，存入以“+”表示，支出以“-”表示，并输出当前账户余额 |
| 挂失模块 | 无 | 使账户进入冻结状态，无法进行任何操作 | 若账户处于正常状态，则挂失成功，并输出相应提示信息 |

注：其中，存款、取款、修改密码、修改预留手机、查询交易记录、挂失模块，若为初次进入系统，需输入卡号及密码，其相应的处理及输出放至存款模块说明，其余模块不再赘述。若已进行过某一模块的操作，则无需再输入银行卡信息。另外，注册银行卡、存款或取款后，会打印当前银行卡信息，包括所属银行、持卡人姓名、持卡人身份证号、账户余额、账户状态（正常或冻结）等。

1. 软件设计
2. 高层数据结构设计

1）类

class Account{ //账户

private:

char cardID[20]; //卡号（必为19位

string password; //密码（必为6位

int state; //银行卡状态

string name; //持卡人姓名

string ID; //持卡人身份证号（必为18位

string phoneNumber; //预留手机号（必为11位

string bank; //所属银行

double balance; //卡内余额

int transactionAmount; //交易次数

double transactionRecord[30]; //交易金额

double transactionTime[30]; //交易时间

public:

static void readAnnotation(); //跳过注释信息

static void writeAnnotation(); //向文件写入注释信息

void writeAccount(); //向文件写入账户信息

void setCardID(char\*); //设置卡号

int setPassword(string); //设置密码

void setName(string); //设置持卡人姓名

int setID(string); //设置持卡人身份证号

int setPhoneNumber(string); //设置预留手机号

void setBank(string); //设置所属银行

void setState(); //冻结账户

void generateCardID(); //随机生成卡号

int record(double); //记录交易金额

void recordTime(double); //记录交易时间

void readAccount(); //从文件读取账户信息

void printAccountInfo(); //打印账户信息

void printTransactionRecord(); //打印交易记录

void saveMoney(); //存款

void drawMoney(); //取款

char\* getcardID(); //获得卡号

string getPassword(); //获得密码

int getState(); //获得账户状态

Account();

~Account();

};

class Bank{ //银行

private:

string name; //名称

public:

void read(); //从文件读取信息

string getName(); //获得银行名称

Bank();

~Bank();

};

2）常量

#define MAX\_ACCOUNT\_AMOUNT 100 //最大账户数量

1. 全局变量

ifstream ifp; //输入文件流

ofstream ofp; //输出文件流

double systemTime; //系统时间

Account accounts[MAX\_ACCOUNT\_AMOUNT]; //存储账户信息

1. txt文件

File/ Account.txt 记录账户信息

每个账户的信息从上至下依次为：

卡号

密码

银行卡状态

持卡人姓名

持卡人身份证号

预留手机号

所属银行

卡内余额

交易次数

交易金额

交易时间

例如：

6733187098973668072

624158

0

小可爱

410303199709010023

15601282822

中国邮政储蓄银行

3000.1

2

8000.00 -4999.90

1.13 1.27

File/Bank.txt 记录银行信息

即，可注册的银行种类：

中国工商银行

中国农业银行

中国银行

中国建设银行

中国招商银行

中国邮政储蓄银行

中国银联

File/System Time.txt 记录系统时间

登陆系统后启动计时。该文件会保存时间，以便再次启动可以继续计时。

例如：

9.01

1. 源码文件

bankPortal.h 头文件：声明所有需要跨源文件调用的类、常量、函数等

bankPortal.cpp 定义main()和所有全局有效的函数

void readSystemTime();void incSystemTime();void writeSystemTime();

Bank.cpp 定义bank类的所有成员函数

Account.cpp 定义Account类的所有成员函数

1. 系统模块划分

各模块函数说明：

|  |  |  |
| --- | --- | --- |
| 模块 | 函数名称 | 功能 |
| 主模块 | int main() | 程序入口 |
| 注册模块 | void Bank.read() | 从文件读取银行种类 |
| string Bank.getName() | 获得银行名称 |
| void Account.setBank(string) | 设置所属银行 |
| void Account.setName(string) | 设置持卡人姓名 |
| int Account.setID(string) | 设置持卡人身份证号 |
| int Account.setPhoneNumber(string) | 设置预留手机号 |
| int Account.setPassword(string) | 设置密码 |
| void Account.generateCardID() | 随机生成卡号 |
| void Account.printAccountInfo() | 打印账户信息 |
| 存款模块 | char\* Account.getcardID() | 获得卡号 |
| string Account.getPassword() | 获得密码 |
| void Account.setState() | 冻结账户 |
| int Account.getState() | 获得账户状态 |
| void Account.saveMoney() | 存款 |
| int Account.record(double) | 记录交易金额 |
| void Account.recordTime(double) | 记录交易时间 |
| 取款模块 | void Account.drawMoney() | 取款 |
| int Account.record(double) | 记录交易金额 |
| void Account.recordTime(double) | 记录交易时间 |
| 修改密码模块 | int Account.setPassword(string) | 设置密码 |
| 修改预留手机模块 | int Account.setPhoneNumber(string) | 设置预留手机号 |
| 查询交易记录模块 | void Account.printTransactionRecord() | 打印交易记录 |
| 挂失模块 | void Account.setState() | 冻结账户 |

1. 高层算法设计
   1. 注册模块

int Account.setPassword(string){

判断密码长度是否为6位，且仅由数字组成

若不是

输出提示信息，并要求重新输入

否则

将账户密码设置为输入字符串

}

int Account.setPhoneNumber(string){

判断手机号是否为11位数字

若不是

输出提示信息，并要求重新输入

否则

将预留手机号设置为输入数字串

}

* 1. 存款模块

void Account.saveMoney(){

从标准输入流读入存款金额

判断存款金额是否符合规定

若符合{

记录交易金额

记录交易时间

存款成功，输出相应提示信息

打印账户当前信息（余额等）

}

}

* 1. 取款模块

void Account.drawMoney(){

打印当前账户余额

从标准输入流读入取款金额

判断取款金额是否大于每日最大额度（5000元）或超出账户余额

若是

输出提示信息，并要求重新输入

否则{

记录交易金额

记录交易时间

取款成功，输出相应提示信息

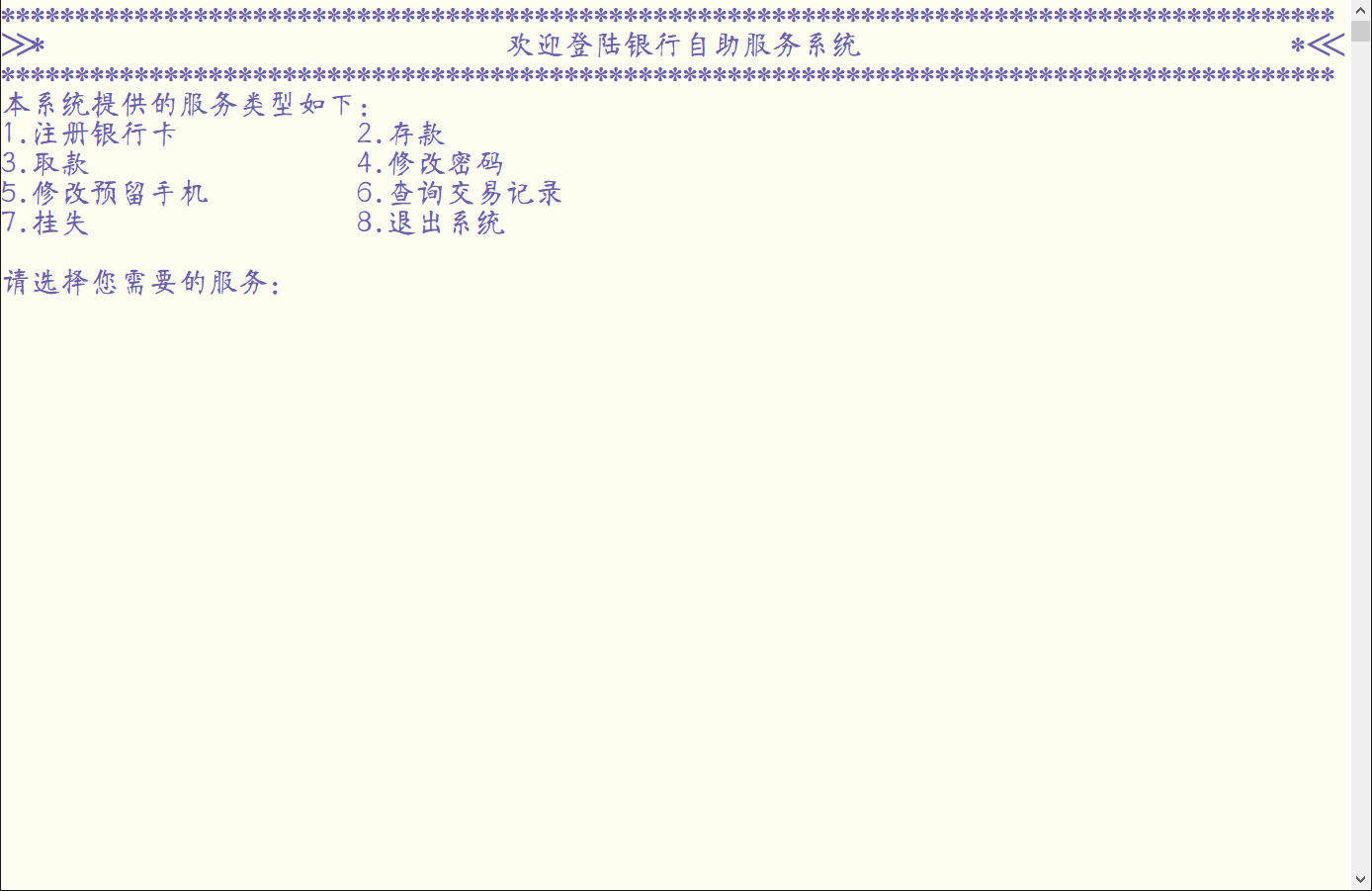
打印账户当前信息（余额等）

}

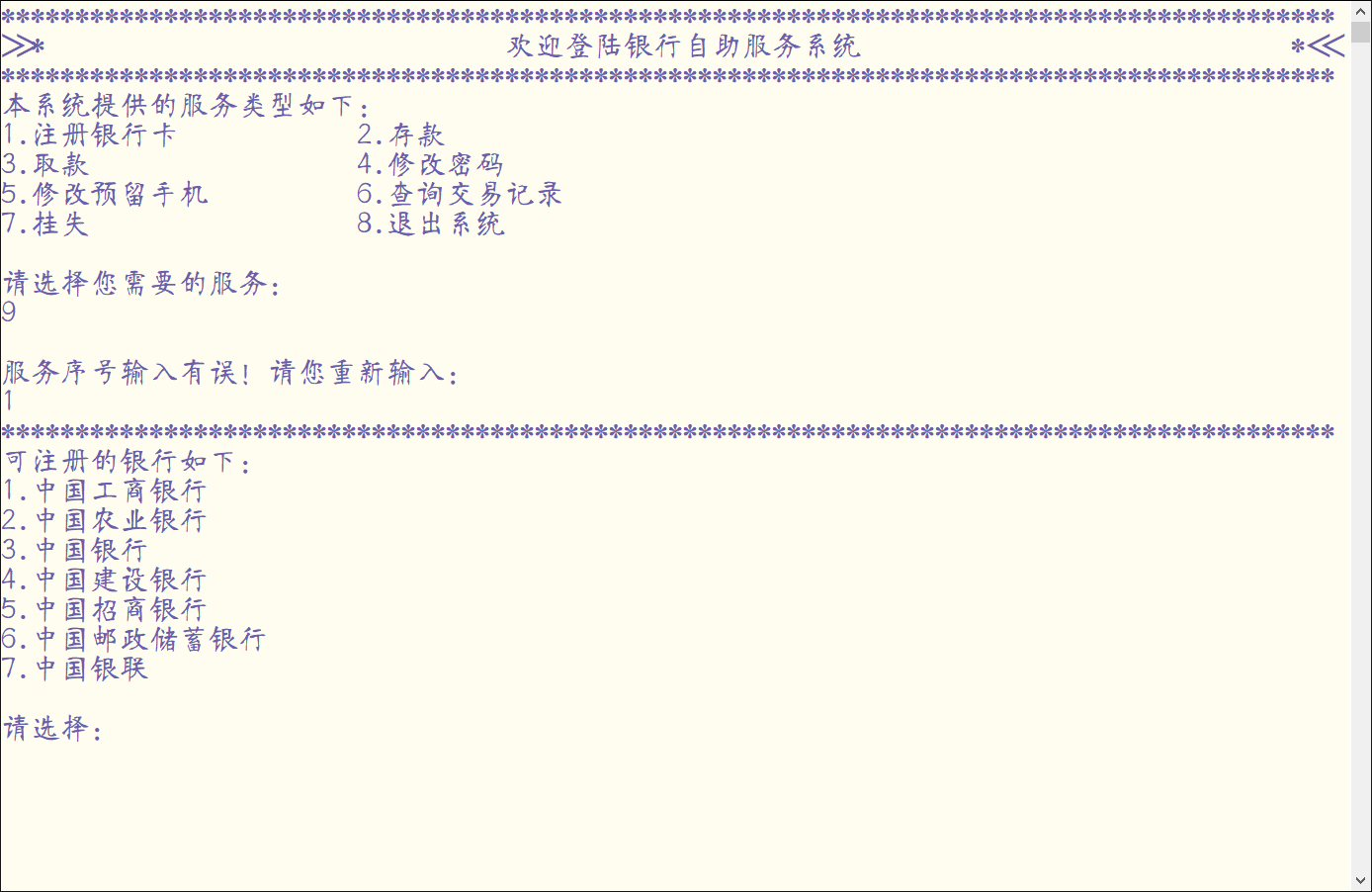
}

1. 性能测试

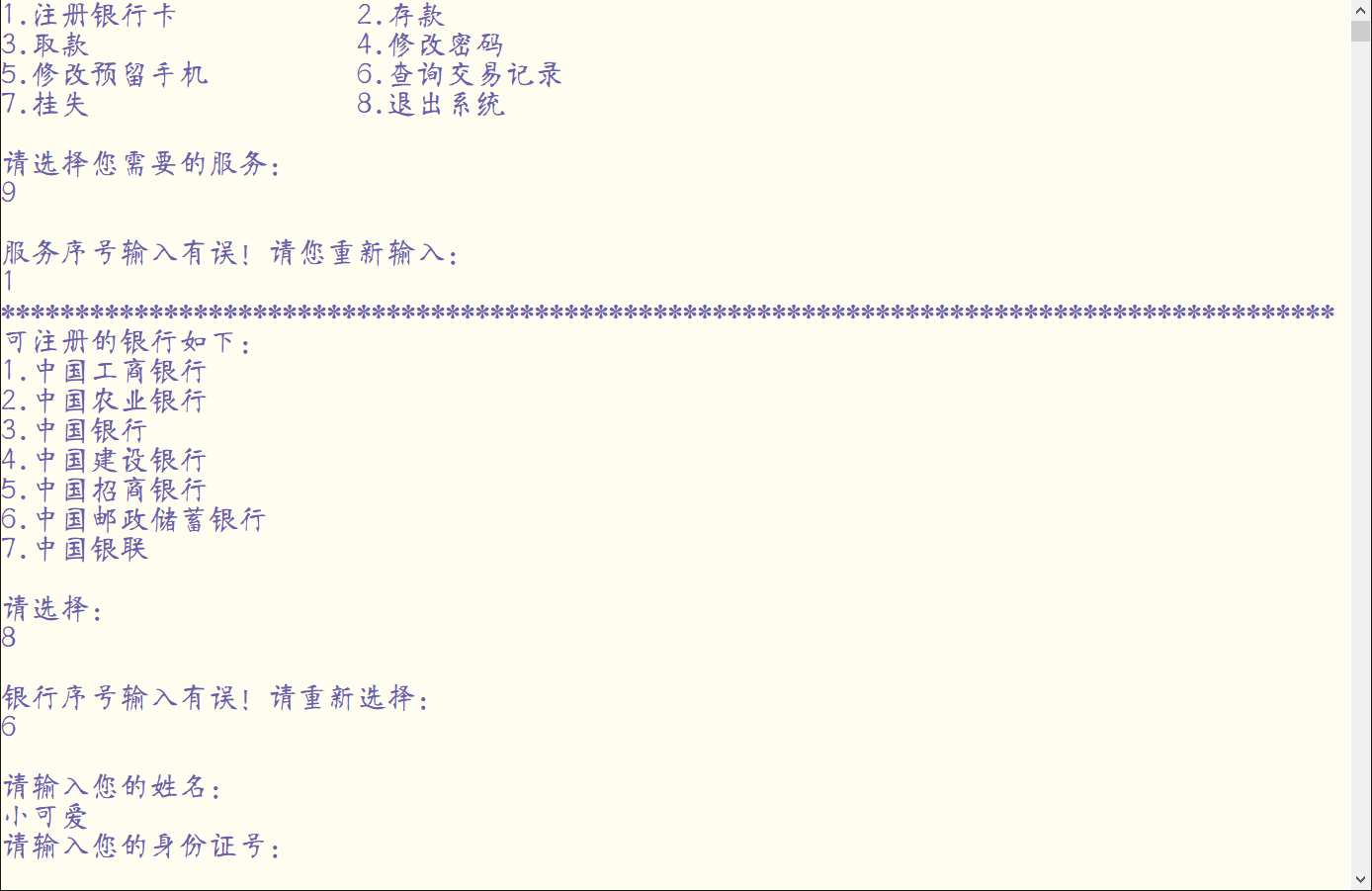
运行程序，界面如下：



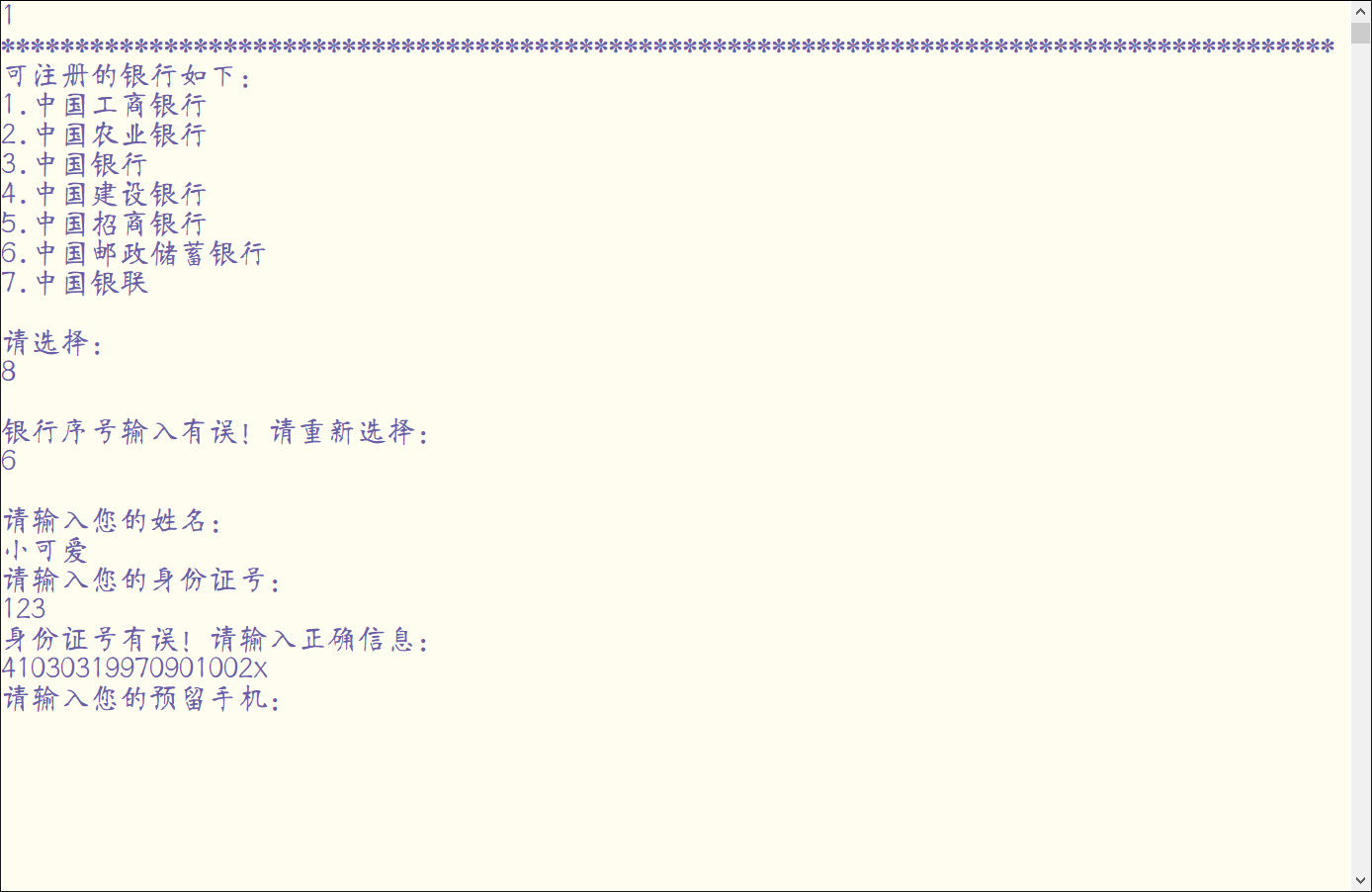
通过输入服务序号，选择相应操作。若序号输入有误，显示提示信息，并要求重新输入。接下来，逐步演示各模块功能。首先选择注册银行卡：



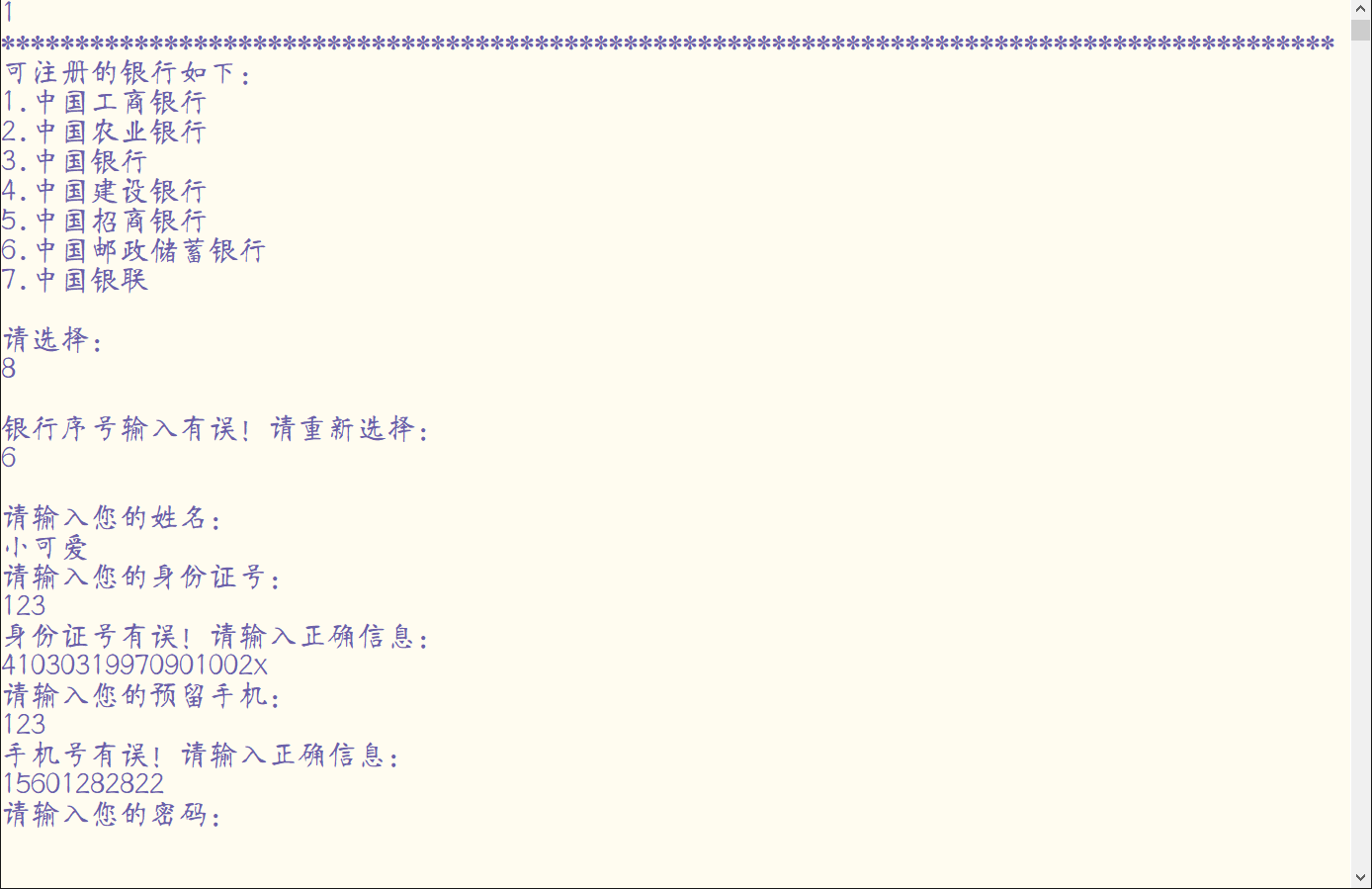
通过输入银行序号，选择所属银行。若序号输入有误，显示提示信息，并要求重新输入：



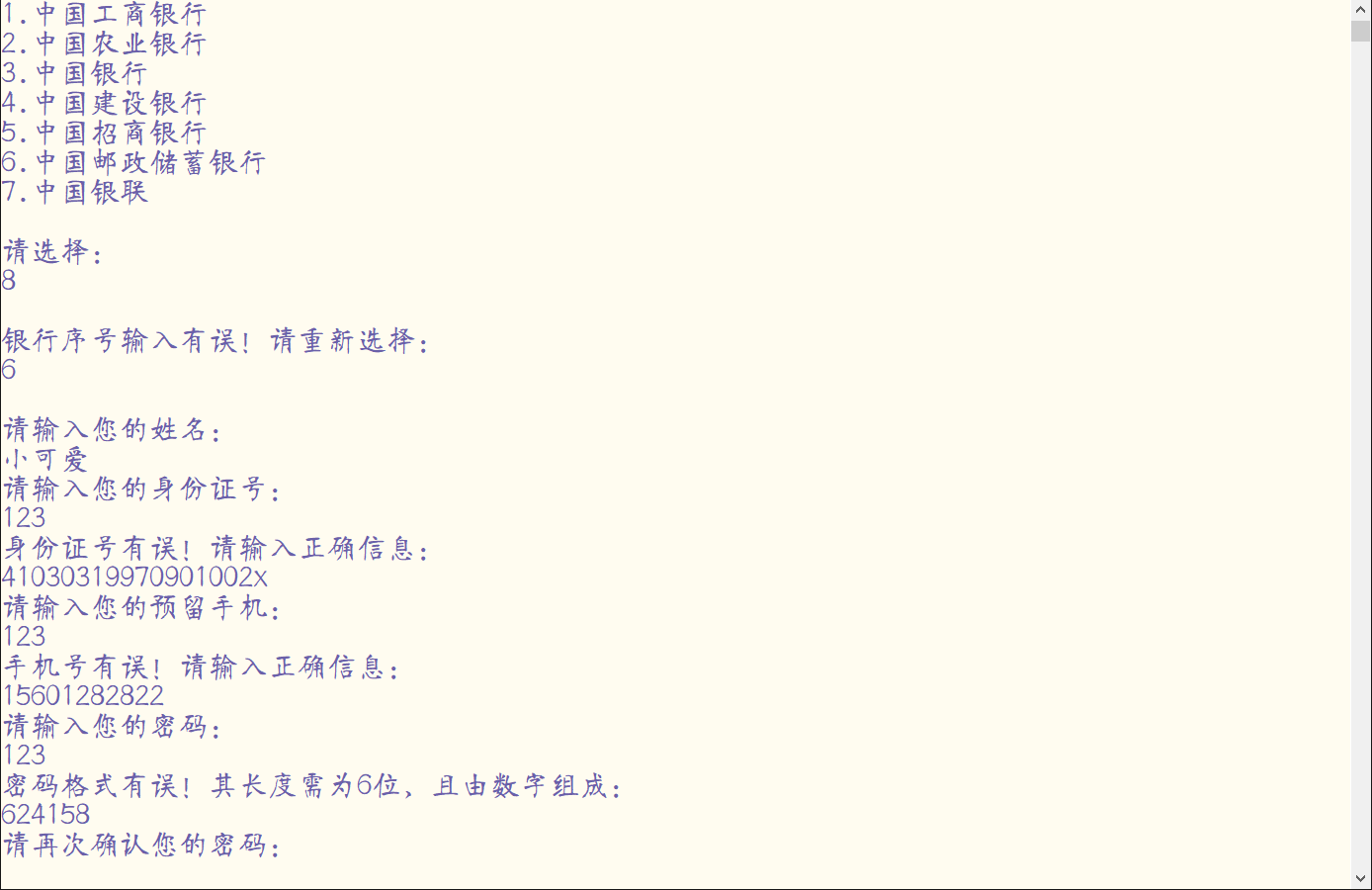
身份证号需为18位，允许字母出现。若输入有误，显示提示信息，并要求重新输入：



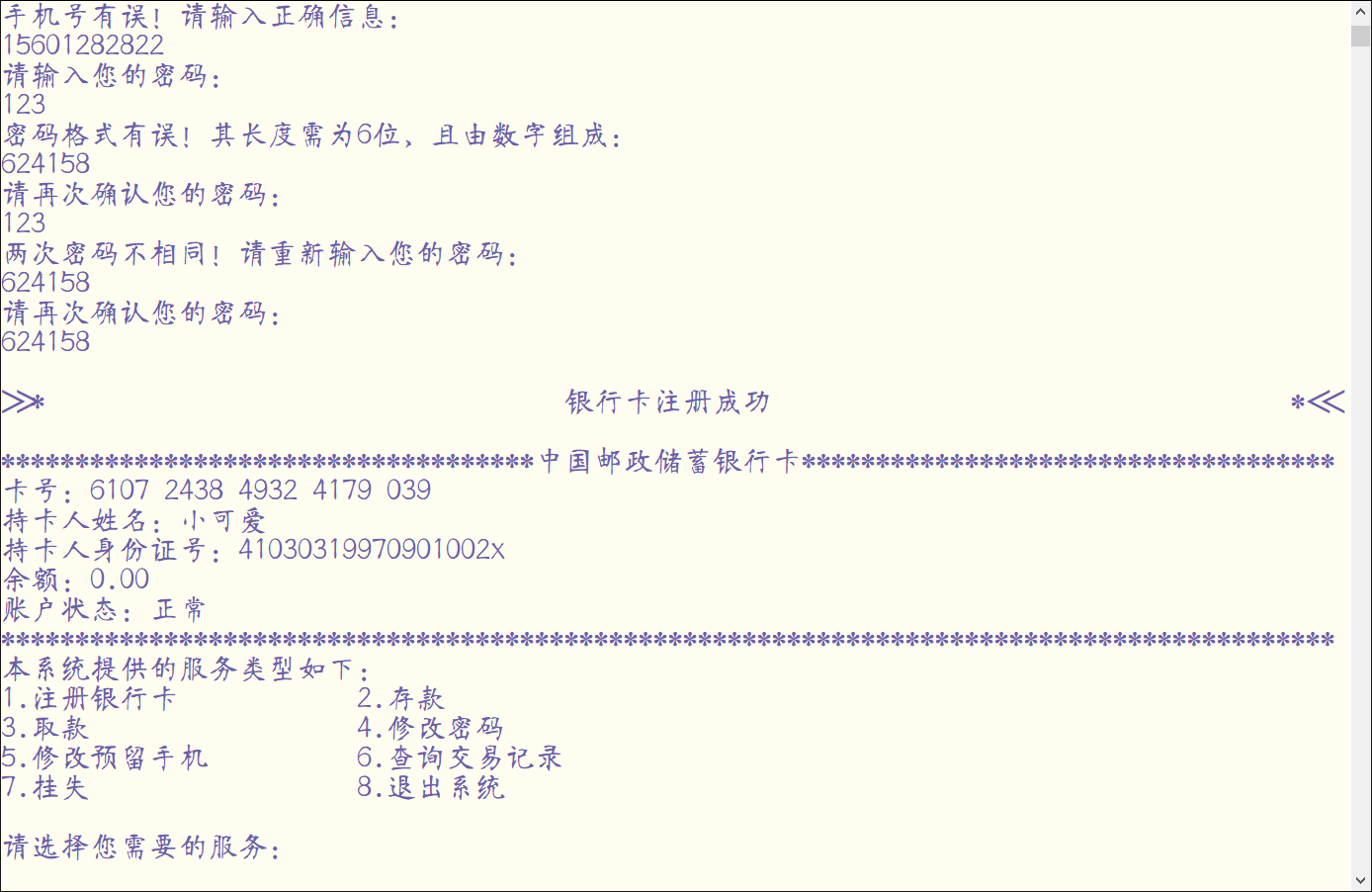
预留手机号需为11位数字。若输入有误，显示提示信息，并要求重新输入：



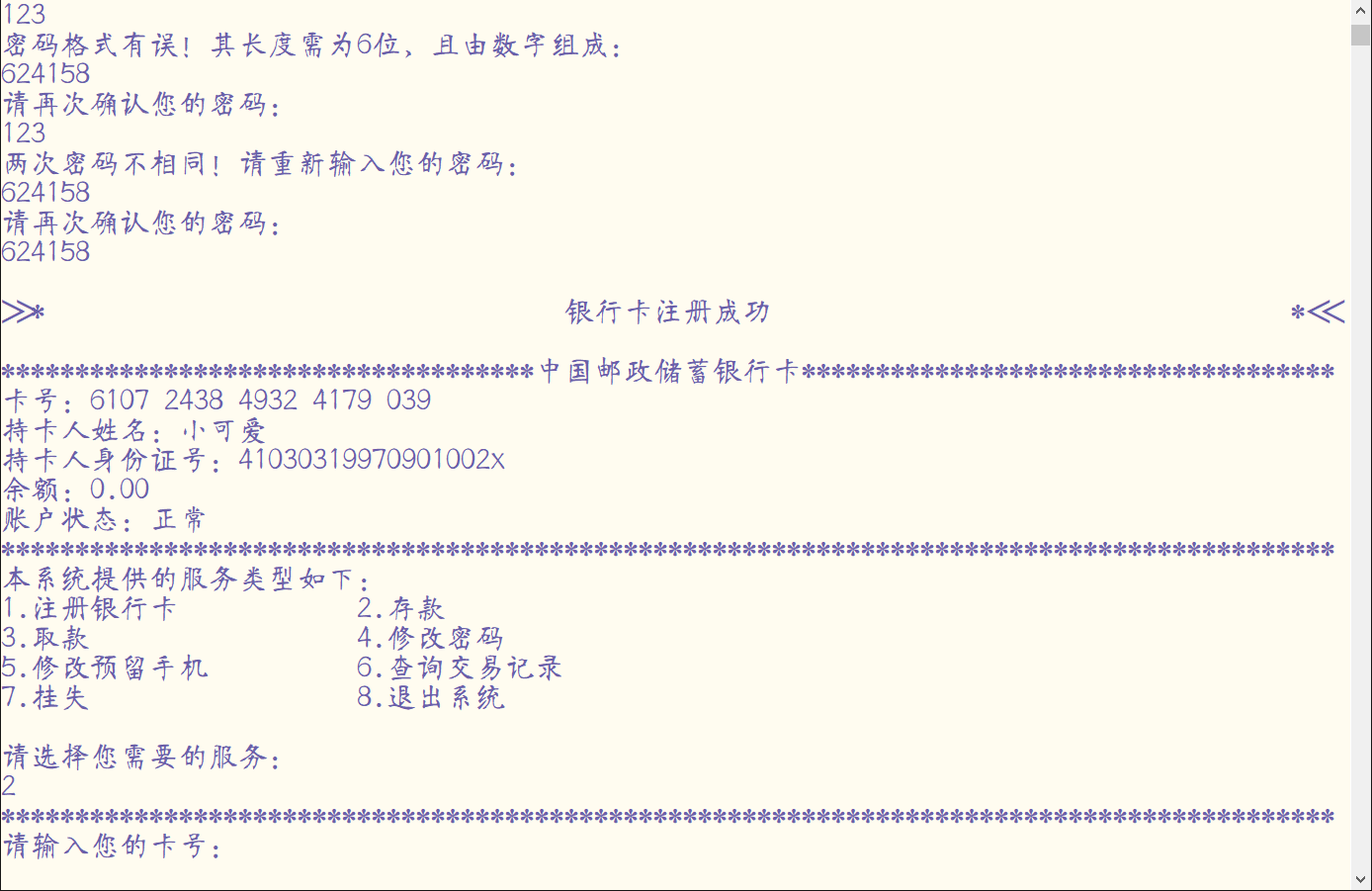
密码需为6位，且仅由数字组成。若输入有误，显示提示信息，并要求重新输入：



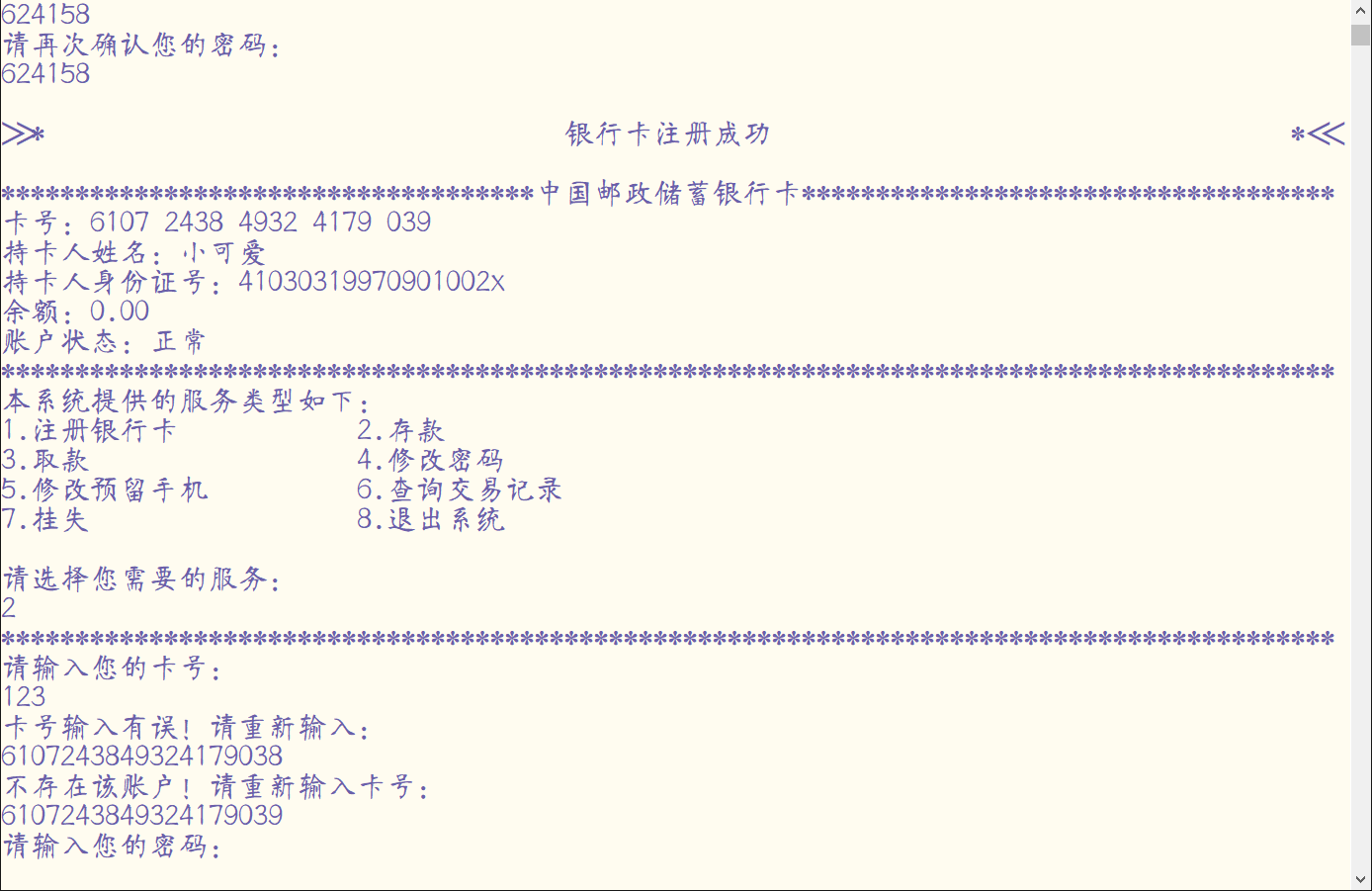
确认密码需与原密码相同，否则显示提示信息，并要求重新设置密码：



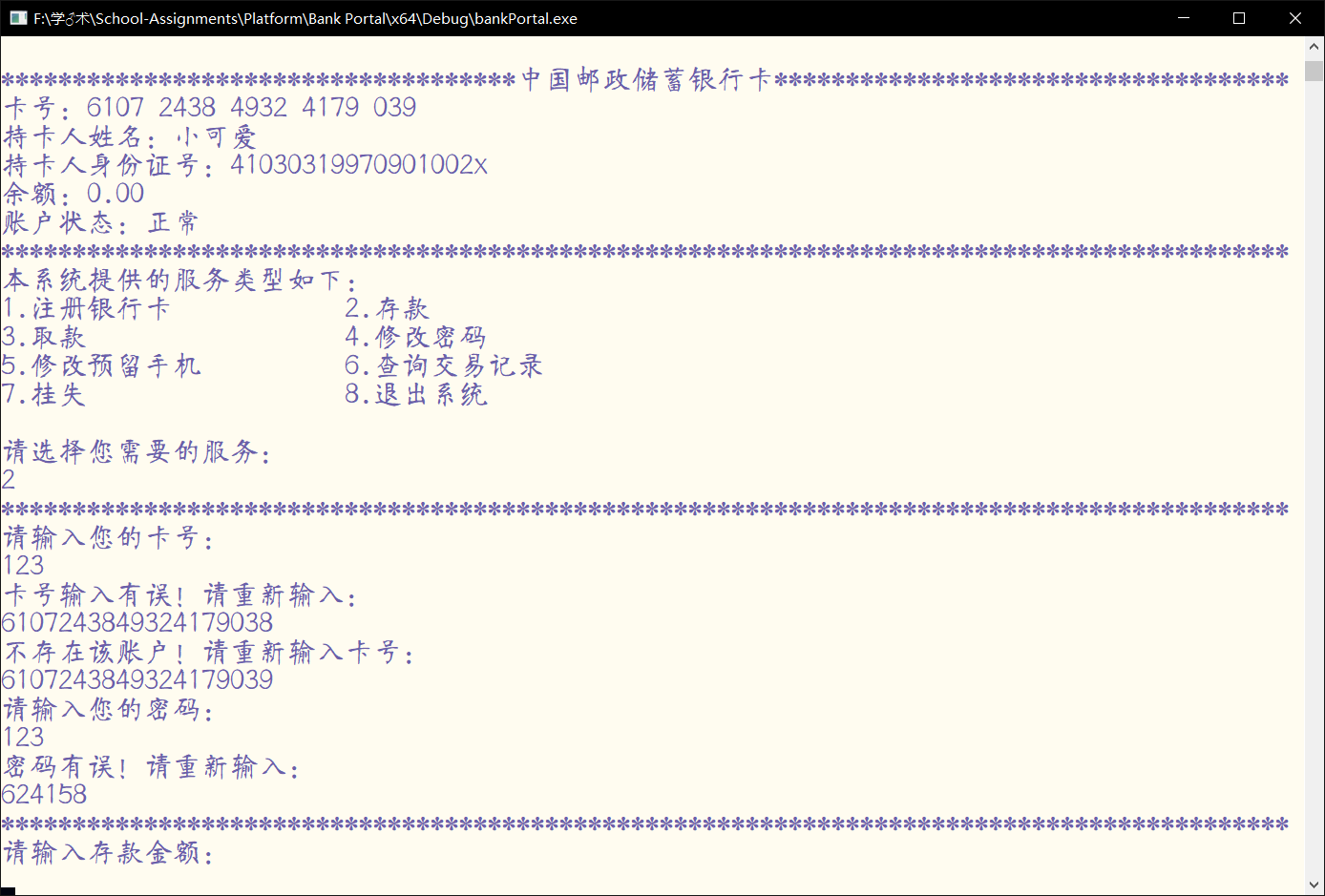
银行卡注册成功。对于存款、取款、修改密码、修改预留手机、查询交易记录、挂失操作，若为初次进入系统，均需输入卡号及密码，若已进行过某一模块的操作，则无需再输入银行卡信息。接下来以存款为例：



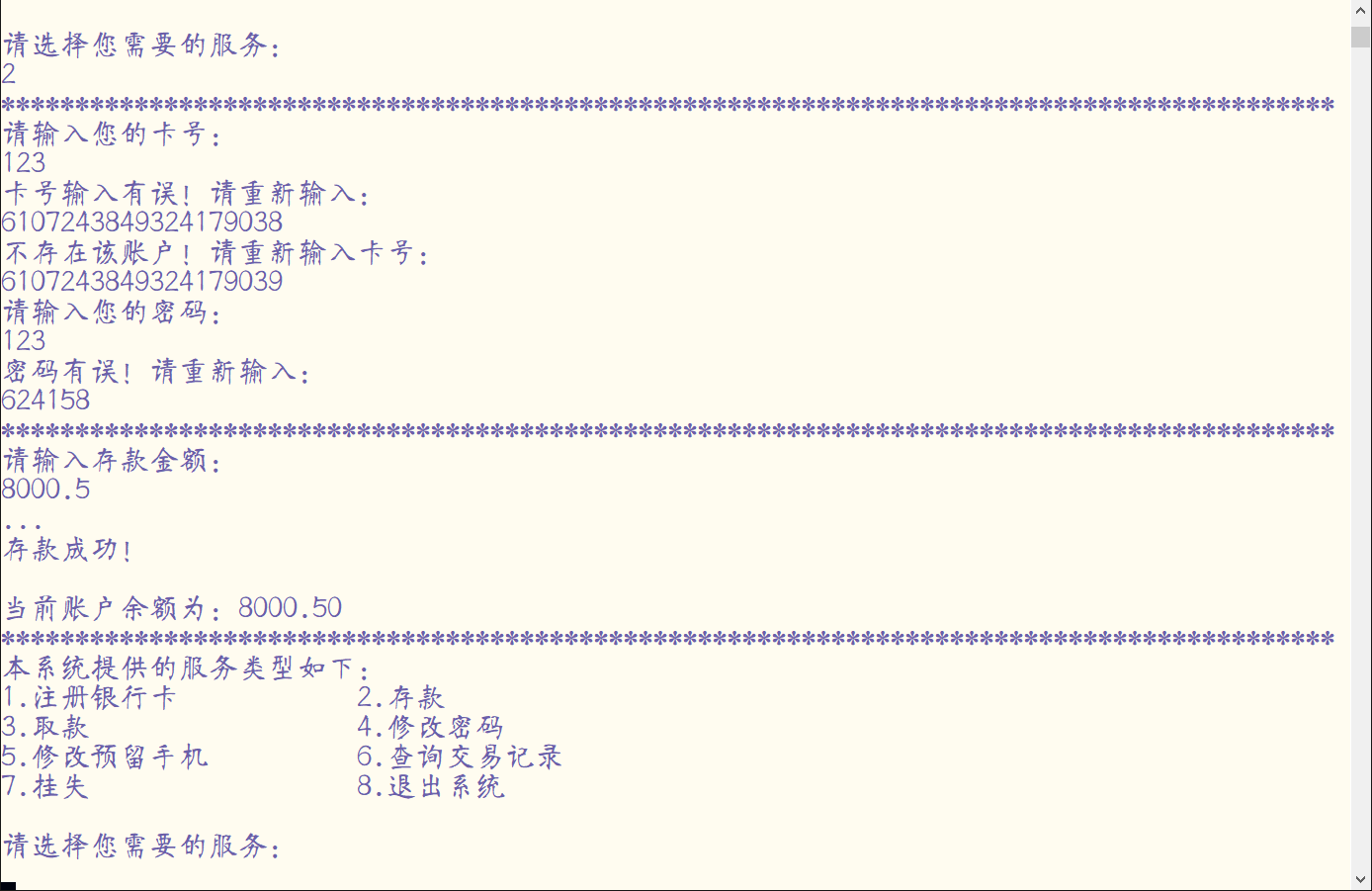
需正确输入卡号。若卡号不存在或形式不正确，显示相应提示信息，并要求重新输入：



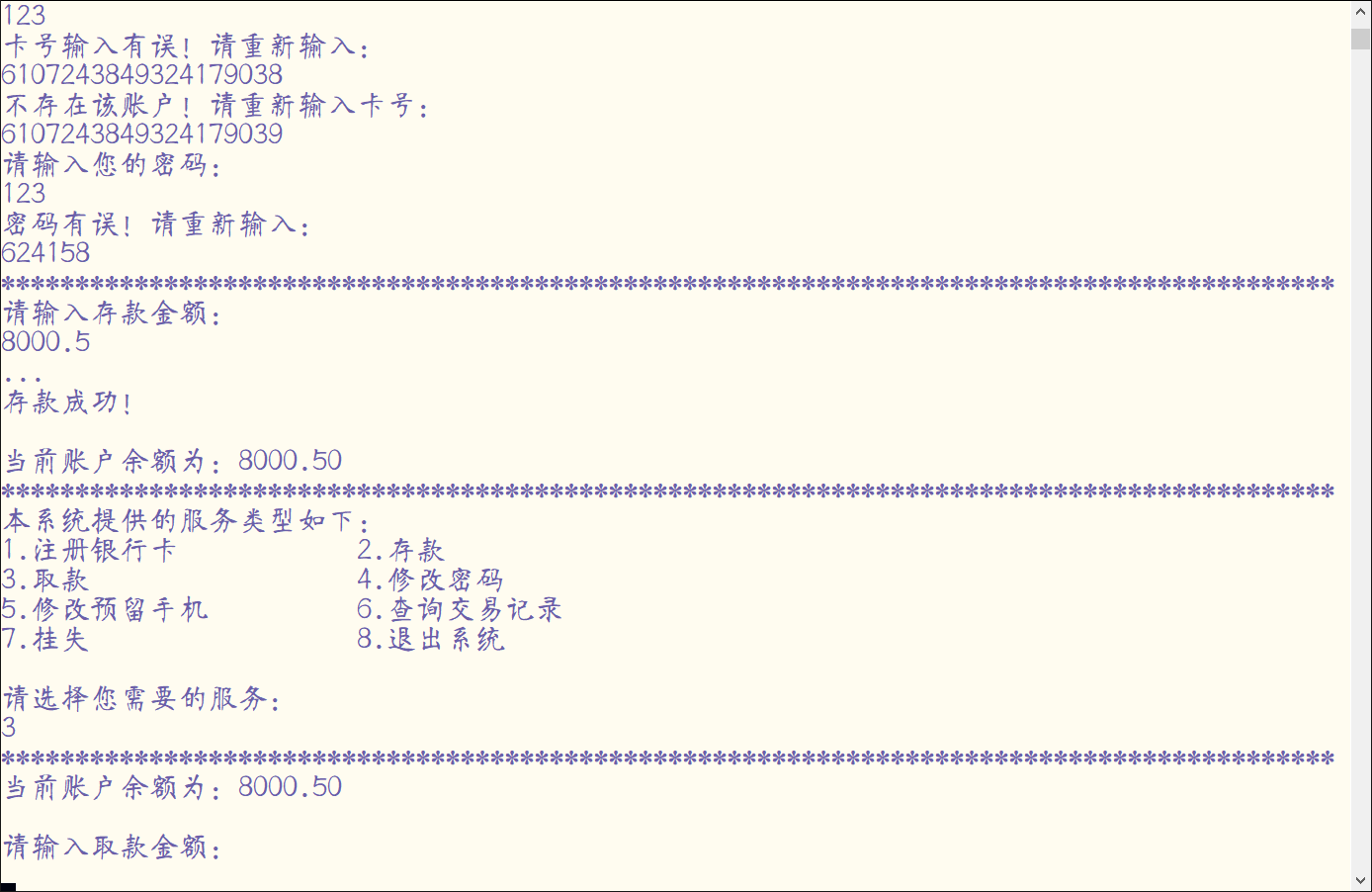
需输入正确密码。若密码不正确，显示提示信息，并要求重新输入：



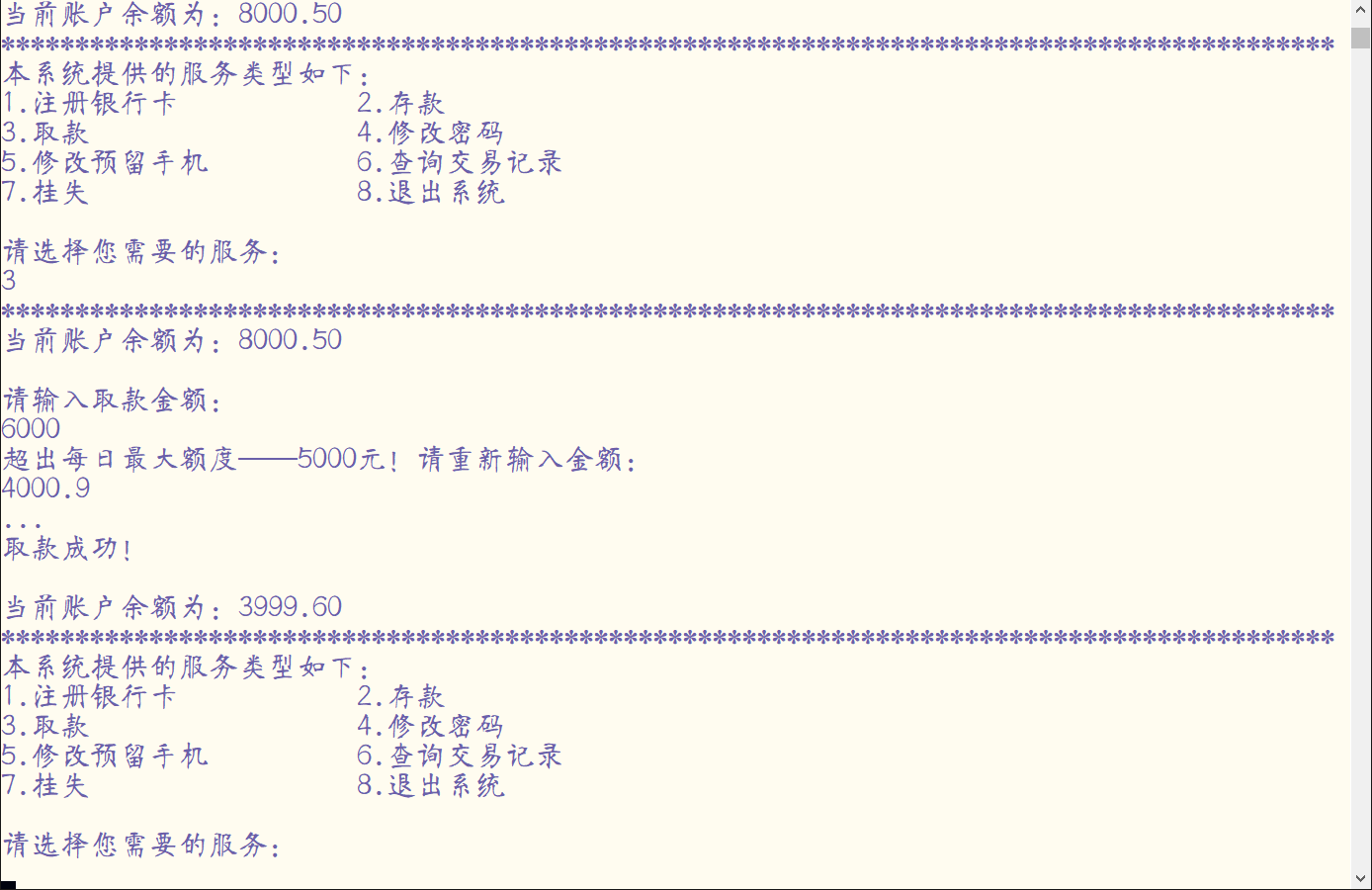
存款金额需为正的小数或整数。若输入有误，显示提示信息，并要求重新输入：



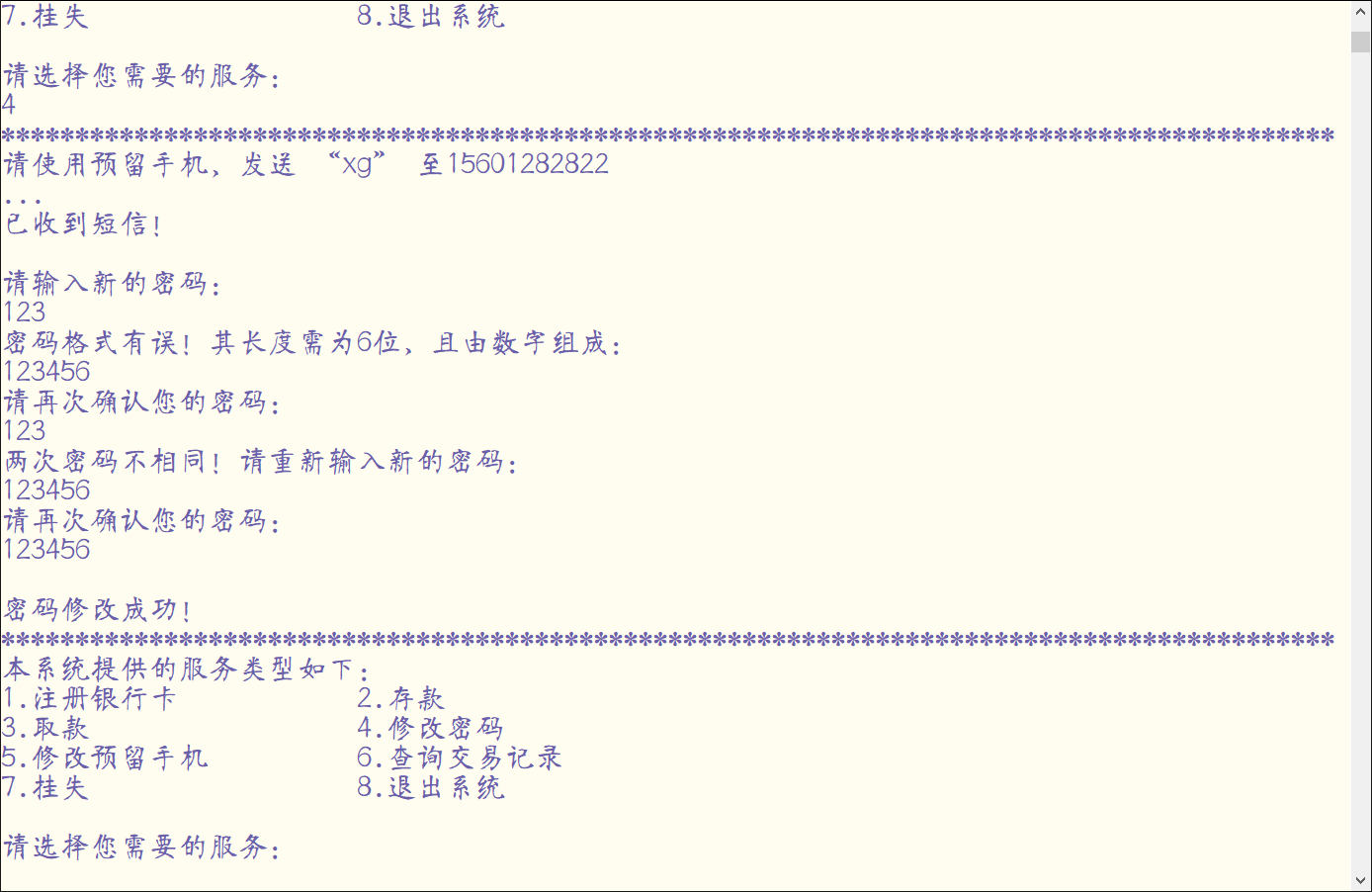
接下来选择取款：



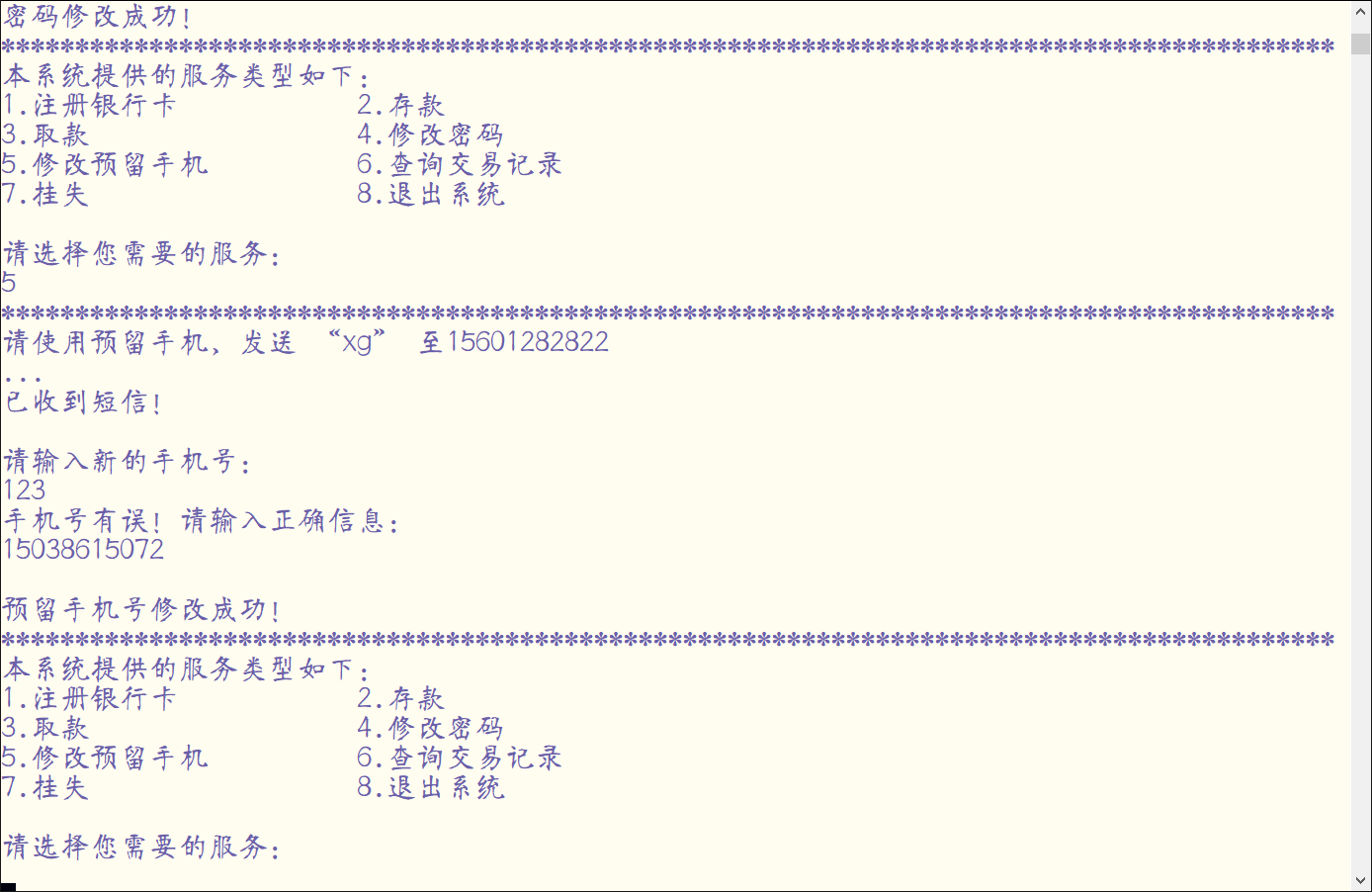
取款金额需为正的小数或整数，且不大于每日最大额度（5000元）和账户余额的较小值，否则显示提示信息，并要求重新输入：



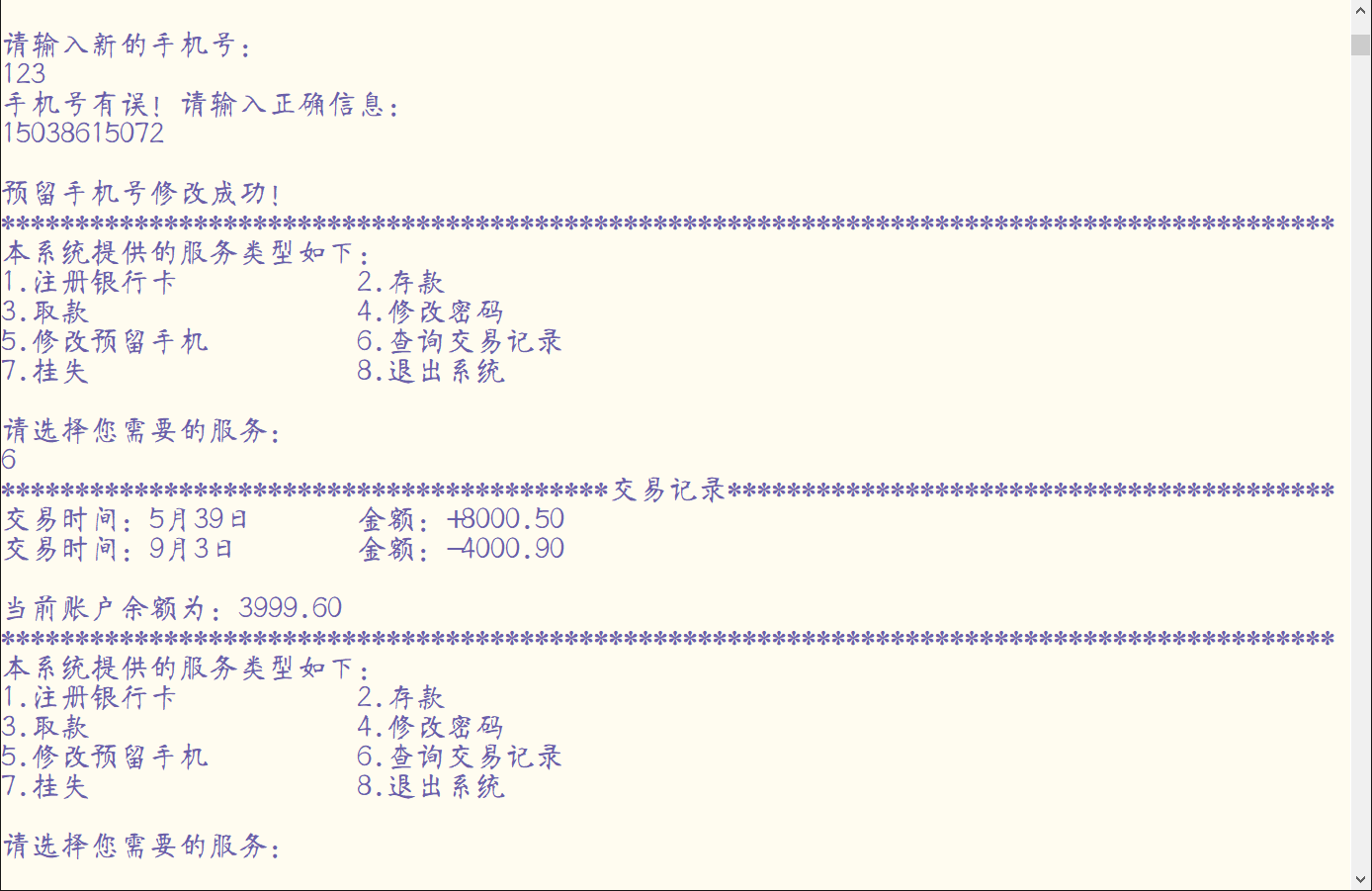
接下来选择修改密码，其过程与注册时的设置密码类似，因而简要演示：



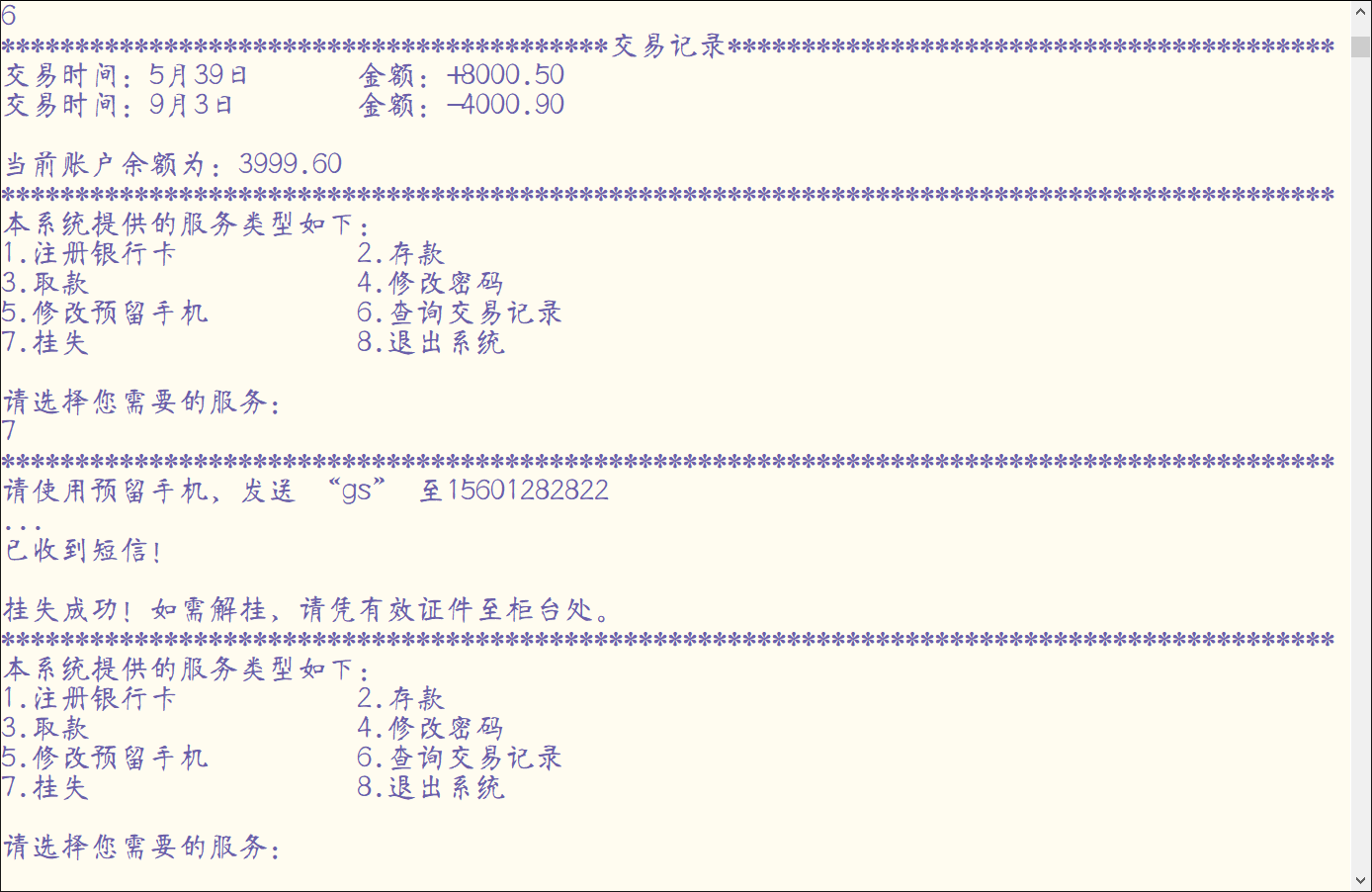
接下来选择修改预留手机，其过程与注册时的设置预留手机类似，因而简要演示：



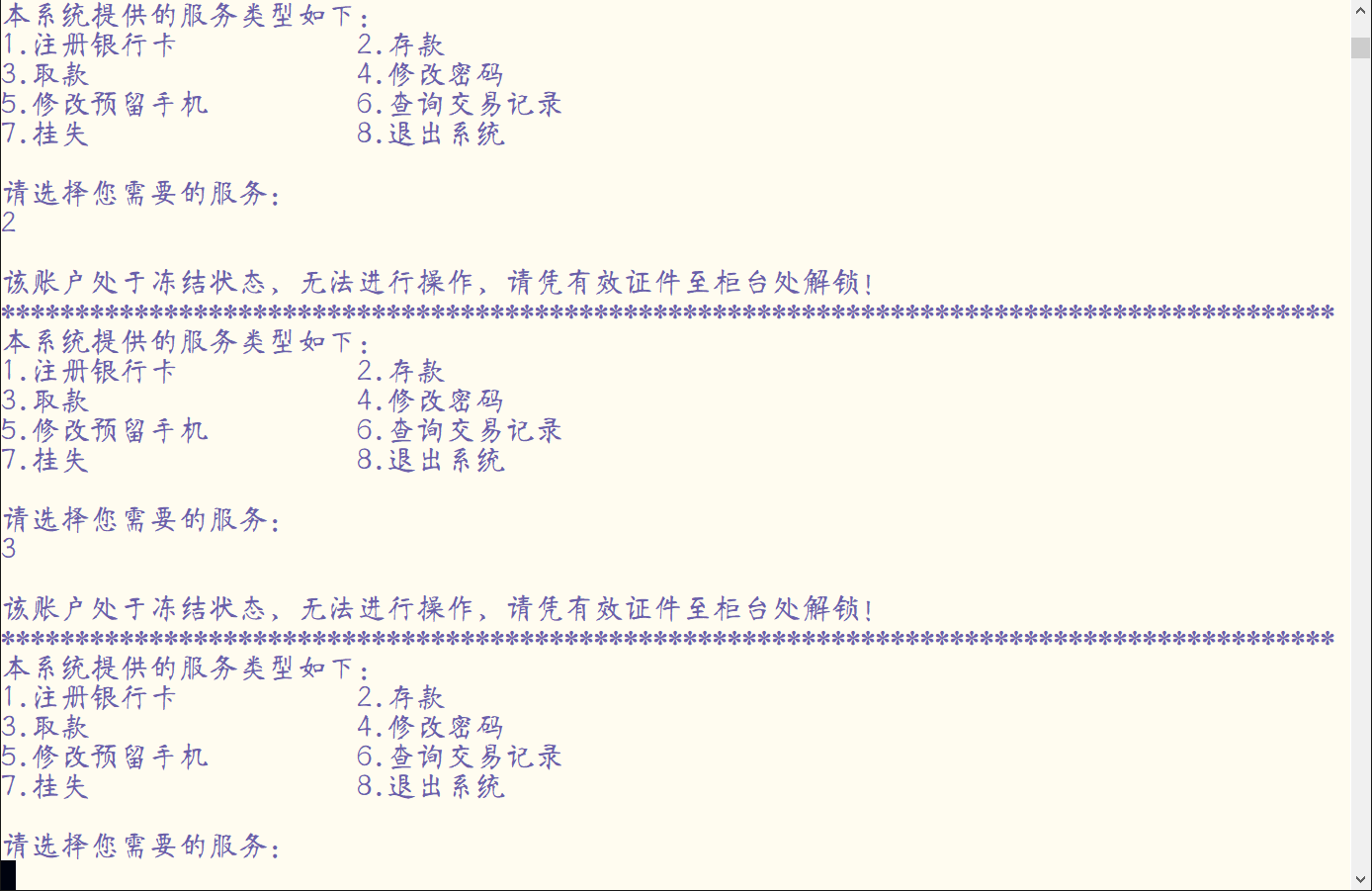
接下来选择查询交易记录：



接下来选择挂失：



一旦挂失成功，2~7操作均无法进行：



以上即为程序功能的大体演示。

电商平台

1. 需求分析

该平台是银行门户之外的一个单独程序，至少支持以下功能：

1. 注册&登录：新用户注册平台账号，已注册用户用平台账号登录。（已注册用户的信息长久保留）；
2. 浏览平台信息；
3. 优惠活动：对同一品类所有产品打折，单笔订单满X减Y；
4. 购买产品：支持用户添加产品到购物车，查看实际应付的价格，提交订单；

附加要求：

1. 平台上至少有三类产品，如食物、服装、图书等。每类至少有三个具体产品，每个产品至少包含产品描述、产品原价、产品剩余量等信息。所有的产品信息存储在数据库或文件，不能写在代码中，平台管理员通过直接修改数据库或文件，管理产品，包括产品的增加和删除，修改数量及具体产品属性信息等。
2. 至少设计一层继承体系（产品基类-产品子类）。设计一个产品基类，然后让图书类、电子产品类和服装类等子类继承它，具体产品是子类的实例对象。基类至少有一个虚函数；
3. 通过为每个产品子类定义“品类折扣系数”来支持对同类所有产品的打折活动（如图书全场5折，则图书类的折扣系数为0.5）；
4. 做好错误场景的处理。
5. 功能说明

|  |  |  |  |
| --- | --- | --- | --- |
| 模块功能 | 输入 | 处理方式 | 输出 |
| 登录模块 | 手机号码  密码 | 判断账户是否存在；  判断密码是否正确 | 若账户不存在，输出提示信息，并要求重新进行操作选择；若密码不正确，输出提示信息，并要求重新进行操作选择；若信息均正确，登陆成功，并输出相应提示信息 |
| 注册模块 | 手机号码  密码 | 判断账户是否存在；  判断手机号是否为11位数字；  判断密码长度是否不少于6位，允许出现数字、字符和特殊符号的任意组合 | 若账户已存在，输出提示信息，并要求直接登陆，输入密码；若手机号不符合规定，输出提示信息，并要求重新输入；若密码不符合规定，输出提示信息，并要求重新输入；若信息均正确，注册成功，并输出相应提示信息 |
| 精准搜索模块 | 搜索词条 | 对已有产品的名称进行子串匹配 | 若无搜索结果，输出提示信息；否则，打印相应搜索结果，并可进行后续操作，如查阅商品信息，加入购物车等 |
| 分区浏览模块 | 分区选择 | 判断分区选择是否正确 | 若分区选择有误，输出提示信息，并要求重新选择；否则，打印该区商品信息，并可进行后续操作，如查阅商品信息，加入购物车等 |
| 购物车模块 | 无 | 遍历账户购物车商品信息 | 若无商品在购物车，输出提示信息；否则，打印商品相关信息，并可进行后续操作，如全选结算，部分结算，修改购买数量，删除部分商品，清空购物车等 |
| 收货地址模块 | 进行修改默认地址、修改收货地址或增添收货地址的选择 | 遍历账户地址信息 | 若无收货地址，输出提示信息；否则，打印收货地址信息，并可进行后续操作 |
| 待发货模块 | 无 | 遍历账户待发货商品信息 | 若无待发货商品，输出提示信息；否则，打印商品相关信息 |
| 待收货模块 | 无 | 遍历账户待收货商品信息 | 若无待收货商品，输出提示信息；否则，打印商品相关信息，并可进行后续操作，如确认收货等 |
| 修改用户名模块 | 新会员名 | 设置账户名称 | 修改成功，输出相应提示信息 |
| 修改密码模块 | 新密码  确认密码 | 判断新密码是否符合规定；  判断确认密码与新密码是否相同 | 若新密码不符合规定，输出提示信息，并要求重新输入；若确认密码与新密码不符，输出提示信息，并要求重新设置密码；若信息均正确，修改密码成功，并输出相应信息 |

注：其中，账户一旦创建，默认用户名为“小可爱”，用户进入系统后，可自行修改。

账户若只有一个收货地址，则系统自动将其设为默认收货地址。

已付款商品，自动归于待发货类别。系统将在1~7天内随机发货，发货后商品归于待收货类别，需要用户手动收货。

1. 软件设计
2. 高层数据结构设计

1）类

class Address { //收货地址

private:

string consignee; //收货人

string contactNumber; //联系电话

string detailedAddress; //具体地址

public:

void readAddress(); //从文件读取地址信息

void writeAddress(); //向文件写入地址信息

void setConsignee(string); //设置收货人

void setContactNumber(string); //设置联系电话

void setDetailedAddress(string); //设置具体地址

string getContactNumber(); //获得联系电话

void priAddress(); //打印地址

Address();

~Address();

};

class Product { //产品基类

private:

static vector<string> categories; //分区

static vector<double> discountOfC; //分区折扣

static vector<string> offerOfC; //分区满减

static int amountOfC; //分区总数量

string name; //产品名称

double originalPrice; //产品原价

double currentPrice; //产品现价

string category; //产品类别

double expressFee; //快递费

int sales; //销量

string storeName; //店家名称

string storeAddress; //店家地址

int productSurplus; //产品剩余量

int purchaseQuantity; //购买数量

int state; //产品状态（1：待发货，2：待收货）

double payTime; //付款时间

double deliveryTime; //发货时间

double discount; //品类折扣系数

string offer; //满减

string intro; //描述

public:

static void readCategories(); //读取分区信息

static void readAnnotation(); //跳过注释信息

static void writeAnnotation(); //向文件写入注释信息

void readProduct(); //读取产品信息

void readForUser(); //读取用户产品信息

void writeProduct(); //向文件写入产品信息

void writeForUser(); //向文件写入用户产品信息

virtual void setCategory(string); //设置产品类别

void setPurchaseQuantity(int); //设置购买数量

void setPurchaseQuantity(int, string); //设置购买数量

void setState(int); //设置产品状态

void setPayTime(double); //设置付款时间

void setDeliveryTime(double); //设置发货时间

void setDiscount(double); //设置品类折扣系数

void setOffer(string); //设置满减信息

static vector<string> getCategories(); //获得分区信息

static vector<double> getDiscountOfC(); //获得分区折扣信息

static vector<string> getOfferOfC(); //获得分区满减信息

static int getAmountOfC(); //获得分区总数量

string getName(); //获得产品名称

double getCurrentPrice(); //获得产品现价

virtual string getCategory(); //获得产品类别

double getExpressFee(); //获得快递费

int getPurchaseQuantity(); //获得购买数量

int getPS(); //获得产品剩余量

int getState(); //获得产品状态

double getPayTime(); //获得付款时间

double getDeliveryTime(); //获得发货时间

double getDiscount(); //获得品类折扣系数

string getOffer(); //获得满减信息

void addSales(int); //增加销量

void decProductS(int); //减少剩余量

void priProduct(); //打印用户商品信息

void priProductInList(); //打印列表中商品信息

void priDetailedProduct(); //打印详细商品信息

void priInfo(); //打印库存信息

void priWF(); //打印待发货或待收货信息

Product();

~Product();

};

class Cosmetic :public Product { //产品子类，美妆

private:

string category; //产品类别

public:

void setCategory(string); //设置产品类别

string getCategory(); //获得产品类别

Cosmetic();

~Cosmetic();

};

class SportsOutdoor :public Product { //产品子类，运动户外

private:

string category; //产品类别

public:

void setCategory(string); //设置产品类别

string getCategory(); //获得产品类别

SportsOutdoor();

~SportsOutdoor();

};

class ArticlesOfDailyUse :public Product { //产品子类，百货

private:

string category; //产品类别

public:

void setCategory(string); //设置产品类别

string getCategory(); //获得产品类别

ArticlesOfDailyUse();

~ArticlesOfDailyUse();

};

class Women :public Product { //产品子类，女装

private:

string category; //产品类别

public:

void setCategory(string); //设置产品类别

string getCategory(); //获得产品类别

Women();

~Women();

};

class Men :public Product { //产品子类，男装

private:

string category; //产品类别

public:

void setCategory(string); //设置产品类别

string getCategory(); //获得产品类别

Men();

~Men();

};

class Food :public Product { //产品子类，食品

private:

string category; //产品类别

public:

void setCategory(string); //设置产品类别

string getCategory(); //获得产品类别

Food();

~Food();

};

class User { //账户

private:

string username; //会员名

int amountOfSA; //收货地址总数

vector<Address> shippingAddress; //收货地址

Address defaultAddress; //默认收货地址

string phoneNumber; //手机号码

string password; //密码

int amountOfSC; //购物车商品总数

vector<Product\*> shoppingCart; //购物车

int amountOfWFD; //待发货商品总数

vector<Product\*> waitForDeliver; //待发货

int amountOfWFS; //待收货商品总数

vector<Product\*> waitForSignature; //待收货

public:

static void readAnnotation(); //跳过注释信息

static void writeAnnotation(); //向文件写入注释信息

void read(); //读取信息

void readAddress(); //读取地址

void writeUser(); //向文件写入用户信息

void setUsername(string); //设置用户名

void setAmountOfSA(int); //设置收货地址总数

void setPhoneNumber(string); //设置手机号码

void setPassword(string); //设置密码

void setAmountOfSC(int); //设置购物车商品总数

void setAmountOfWFD(int); //设置待发货商品总数

void setAmountOfWFS(int); //设置待收货商品总数

string getUsername(); //获得用户名

vector<Address> getShippingAddress(); //获得收货地址

void addShippingAddress(); //增添收货地址

void modifyShippingAddress(); //修改收货地址

void delShippingAddress(); //删除收货地址

void priShippingAddress(); //打印收货地址

Address getDA(); //获得默认地址

void modifyDefaultAddress(); //修改默认地址

string getPhoneNumber(); //获得手机号码

string getPassword(); //获得密码

void modifyPassword(); //修改密码

void modifyName(); //修改用户名

vector<Product\*> getSC(); //获得购物车信息

int getAmountOfSC(); //获得购物车商品数量

void addShoppingCart(Product\*); //加入购物车

int priShoppingCart(); //打印购物车信息

void emptyShoppingCart(); //清空购物车

void delPartOfSC(); //删除购物车商品

void modifyPurchaseQuantity(); //修改购买数量

void clearPartOfSC(); //部分结算

void clearALLShoppingCart(); //全选结算

vector<Product\*> getWFD(); //获得待发货商品信息

int getAmountOfWFD(); //获得待发货商品数量

void priWFD(); //打印待发货商品信息

void updateWFD(); //更新待发货商品（默认7天之内发货）

vector<Product\*> getWFS(); //获得待收货商品信息

int getAmountOfWFS(); //获得待收货商品数量

void priWFS(); //打印待收货商品信息

void confirmSignature(); //确认收货

void priUser(); //打印用户信息

User();

~User();

};

2）常量

#define AMOUNTOFITEMSOFPAGE 4 //页面最大显示数量

#define MAXCATEGORIES 10 //最大品类数

1. 全局变量

vector<Cosmetic> cosmeticForSC; //购物车美妆类商品信息

vector<SportsOutdoor> sportsForSC; //购物车运动户外类商品信息

vector<ArticlesOfDailyUse> articlesOfDUForSC; //购物车百货类商品信息

vector<Women> womenForSC; //购物车女装类商品信息

vector<Men> menForSC; //购物车男装类商品信息

vector<Food> foodForSC; //购物车食品类商品信息

vector<Cosmetic> cosmeticForWFD; //待发货美妆类商品信息

vector<SportsOutdoor> sportsForWFD; //待发货运动户外类商品信息

vector<ArticlesOfDailyUse> articlesOfDUForWFD; //待发货百货类商品信息

vector<Women> womenForWFD; //待发货女装类商品信息

vector<Men> menForWFD; //待发货男装类商品信息

vector<Food> foodForWFD; //待发货食品类商品信息

vector<Cosmetic> cosmeticForWFS; //待收货美妆类商品信息

vector<SportsOutdoor> sportsForWFS; //待收货运动户外类商品信息

vector<ArticlesOfDailyUse> articlesOfDUForWFS; //待收货百货类商品信息

vector<Women> womenForWFS; //待收货女装类商品信息

vector<Men> menForWFS; //待收货男装类商品信息

vector<Food> foodForWFS; //待收货食品类商品信息

vector<Cosmetic> cosmeticList; //美妆类商品信息

vector<SportsOutdoor> sportsList; //运动户外类商品信息

vector<ArticlesOfDailyUse> articlesOfDUList; //百货类商品信息

vector<Women> womenList; //女装类商品信息

vector<Men> menList; //男装类商品信息

vector<Food> foodList; //食品类商品信息

vector<User> userList; //账户信息

double systemTime = 0; //系统时间

ifstream ifp; //输入文件流

ofstream ofp; //输出文件流

1. txt文件

File/ Category.txt 记录类别信息

每个类别的信息从上至下依次为：

类别名称

品类折扣系数

满减信息

例如：

美妆

9.5

300 20

File/ Product.txt 记录产品信息

每个产品的信息从上至下依次为：

产品类别

产品名称

产品原价

产品现价

快递费

月销量

店家名称

店家地址

产品剩余量

品类折扣系数

满减

产品描述

例如：

美妆

科颜氏牛油果眼霜14g 黑眼圈 紧致细纹 保湿补水

255.00

188.99

0.00

796

阿如海外代购

浙江宁波

212

9.50

300 20

富含牛油果树果脂、鳄梨油、维生素E的独特乳霜质地眼霜。使用涂抹时可感受水分释放而出，舒缓眼周干燥，帮助脆弱的眼周肌肤获得理想的保湿度，滋润眼部肌肤。

File/System Time.txt 记录系统时间

登陆系统后启动计时。该文件会保存时间，以便再次启动可以继续计时。

例如：

1.01

File/ User.txt 记录账户信息

每个账户的信息从上至下依次为：

会员名

收货地址总数

收货地址

默认收货地址

手机号码

密码

购物车商品总数

购物车

待发货商品总数

待发货

待收货商品总数

待收货

其中，收货地址格式为：

收货人

联系电话

具体地址

购物车&待发货&待收货商品格式为：

产品类别

产品名称

产品原价

产品现价

快递费

月销量

店家名称

店家地址

产品剩余量

品类折扣系数

满减

购买数量

产品状态（1：待发货，2：待收货，0：其他）

付款时间

发货时间

例如：

小可爱

1

大可爱

15601282822

北邮

大可爱

15601282822

北邮

15601282822

624158

1

美妆

【包顺丰】LA MER海蓝之谜 浓缩修护精华眼霜15ml 经典眼霜

1168.00

1058.00

0.00

412

瑞月坊 桃子代购

北京

308

9.50

300 20

1

0

0.00

0.00

0

0

0

1. 源码文件

E-commercePlatform.h 头文件：声明所有需跨源文件调用的类、常量、函数等

E-commercePlatform.cpp 定义main()和所有全局有效的函数

void readProducts();void readUsers();void readSystemTime();void incSystemTime();void writeSystemTime();

Address.cpp 定义Address类的所有成员函数

Product.cpp 定义Product类的所有成员函数

User.cpp 定义User类的所有成员函数

1. 系统模块划分

各模块函数说明：

|  |  |  |  |
| --- | --- | --- | --- |
| 模块功能 | 函数名称 | | 功能 |
| 主模块 | int main() | | 程序入口 |
| 登录模块 | string User.getPhoneNumber() | | 获得手机号码 |
| string User.getPassword() | 获得密码 | |
| 注册模块 | string User.getPhoneNumber() | 获得手机号码 | |
| void User.setPhoneNumber(string) | 设置手机号码 | |
| void User.setPassword(string) | 设置密码 | |
| 精准搜索模块 | string Product.getName() | 获得产品名称 | |
| void Product.priProductInList() | 打印列表中商品信息 | |
| void Product.priDetailedProduct() | 打印详细商品信息 | |
| void Product.priInfo() | 打印库存信息 | |
| int Product.getPS() | 获得产品剩余量 | |
| void Product.setPurchaseQuantity(int, string) | 设置购买数量 | |
| void User.addShoppingCart(Product\*) | 加入购物车 | |
| 分区浏览模块 | void Product.priProductInList() | 打印列表中商品信息 | |
| void Product.priDetailedProduct() | 打印详细商品信息 | |
| void Product.priInfo() | 打印库存信息 | |
| int Product.getPS() | 获得产品剩余量 | |
| void Product.setPurchaseQuantity(int, string) | 设置购买数量 | |
| void User.addShoppingCart(Product\*) | 加入购物车 | |
| 收货地址模块 | vector<Address> User.getShippingAddress() | 获得收货地址 | |
| void User.priShippingAddress() | 打印收货地址 | |
| void User.modifyDefaultAddress() | 修改默认地址 | |
| void User.addShippingAddress() | 增添收货地址 | |
| void Adress.setConsignee(string) | 设置收货人 | |
| void Address.setContactNumber(string) | 设置联系电话 | |
| void Address.setDetailedAddress(string) | 设置具体地址 | |
| void User.delShippingAddress() | 删除收货地址 | |
| 购物车模块 | int User.priShoppingCart() | 打印购物车信息 | |
| void Product.priProduct() | 打印用户商品信息 | |
| void User.clearALLShoppingCart() | 全选结算 | |
| virtual string Product.getCategory() | 获得产品类别 | |
| static vector<string> Product.getCategories() | 获得分区信息 | |
| double Product.getCurrentPrice() | 获得产品现价 | |
| double Product.getExpressFee() | 获得快递费 | |
| int Product.getPurchaseQuantity() | 获得购买数量 | |
| string Product.getOffer() | 获得满减信息 | |
| double Product.getDiscount() | 获得品类折扣信息 | |
| void Product.addSales(int) | 增加销量 | |
| void Product.decProductS(int) | 减少剩余量 | |
| void Product.setPayTime(double) | 设置付款时间 | |
| void User.clearPartOfSC() | 部分结算 | |
| void User.modifyPurchaseQuantity() | 修改购买数量 | |
| void User.delPartOfSC() | 删除购物车商品 | |
| void User.emptyShoppingCart() | 清空购物车 | |
| 待发货模块 | void User.setAmountOfWFD(int) | 设置待发货商品总数 | |
| void User.priWFD() | 打印待发货商品信息 | |
| void Product.priWF() | 打印待发货或待收货信息 | |
| double Product.getPayTime() | 获得付款时间 | |
| 待收货模块 | int User.getAmountOfWFS() | 获得待收货商品数量 | |
| void User.priWFS() | 打印待收货商品信息 | |
| double Product.getDeliveryTime() | 获得发货时间 | |
| void User.confirmSignature() | 确认收货 | |
| 修改用户名模块 | void User.modifyName() | 修改用户名 | |
| 修改密码模块 | void User.modifyPassword() | 修改密码 | |

1. 高层算法设计

1） 注册模块

void User.setPhoneNumber(string){

判断手机号是否为11位数字

若不是

输出提示信息，并要求重新输入

否则

将手机号设置为输入数字串

}

2） 精准搜索模块

void User.addShoppingCart(Product\*){

判断商品是否已存在

若不存在{

加入购物车

购物车商品数量增加

}

否则

购物中相应商品的购买数量增加

}

3） 收货地址模块

void User.addShippingAddress(){

从标准输入流读入姓名

设置收货人姓名

从标准输入流读入联系电话

判断手机号是否为11位数字

若不是

输出提示信息，并要求重新输入

否则

将联系电话设置为输入数字串

从标准输入流读入具体地址

设置具体地址

收货地址数量增加

判断是否只有一个收货地址

若是

将其设为默认收货地址

增添成功，输出相应提示信息

}

void User.delShippingAddress(){

从标准输入流依次读入需删除的收货地址序号

遍历收货地址，删除对应项

收货地址数量减少

若默认收货地址被删除，且收货地址数量不为0

将收货地址首项设为默认

删除成功，输出相应提示信息

}

4） 购物车模块

void User.clearALLShoppingCart(){

分类别统计消费金额（含邮费）

计算各类别打折、满减后金额，并打印

相加，计算总金额，并打印

对应商品销量增加，剩余量减少

已付款商品记录付款时间，并加入待发货列表

待发货数量增加

购物车清空

购买成功，输出相应提示信息

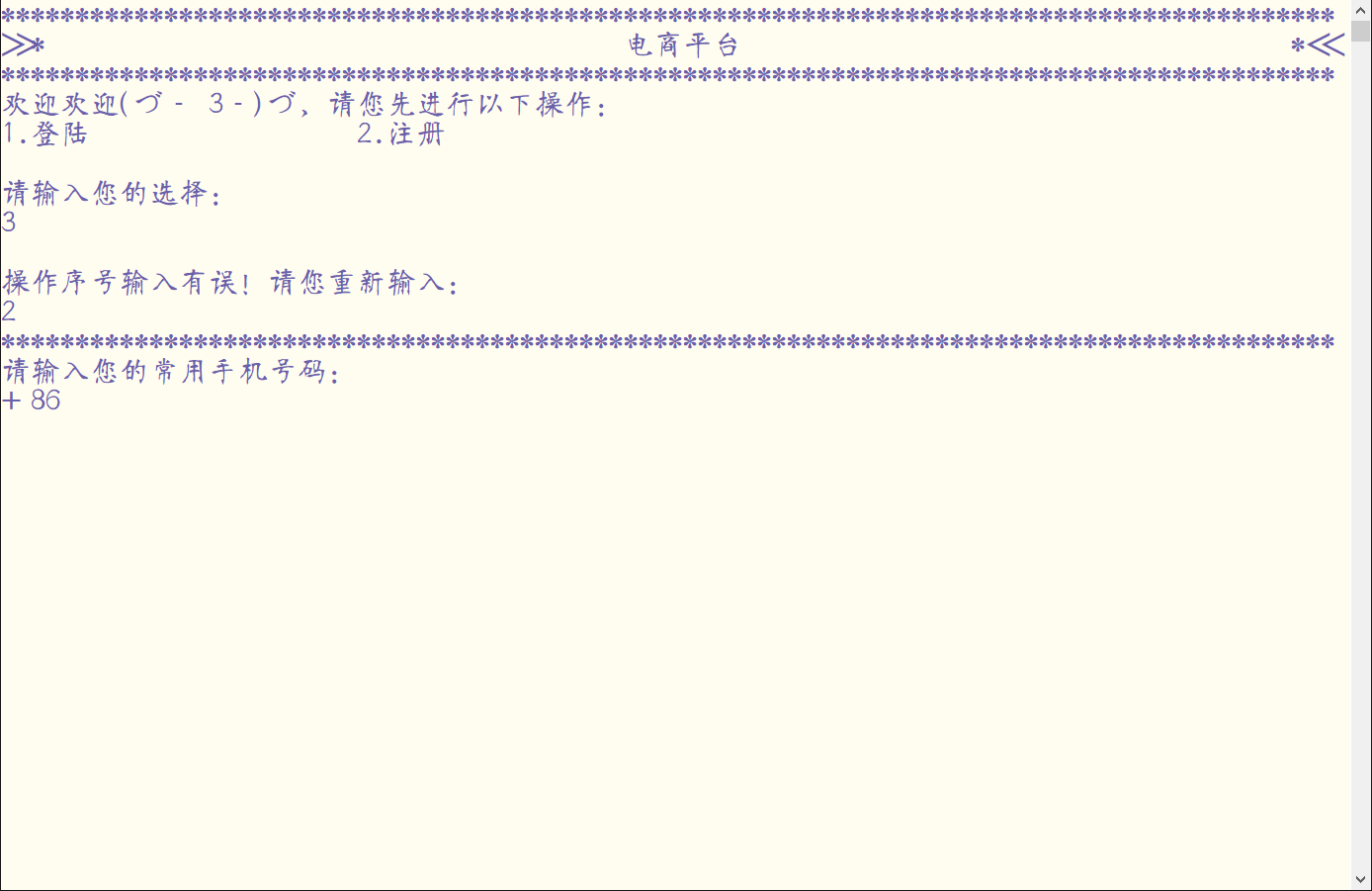
}

1. 性能测试

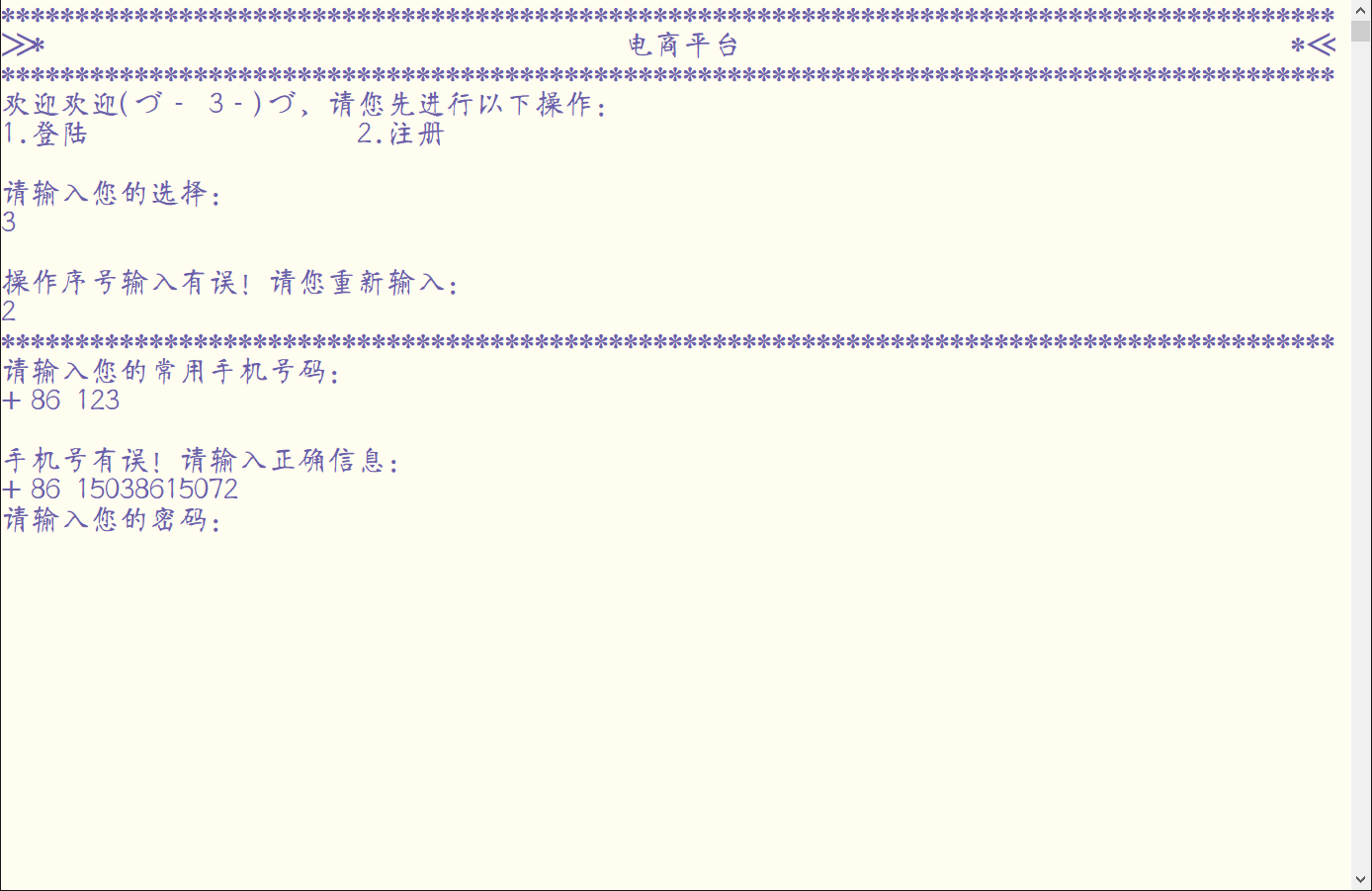
运行程序，界面如下：

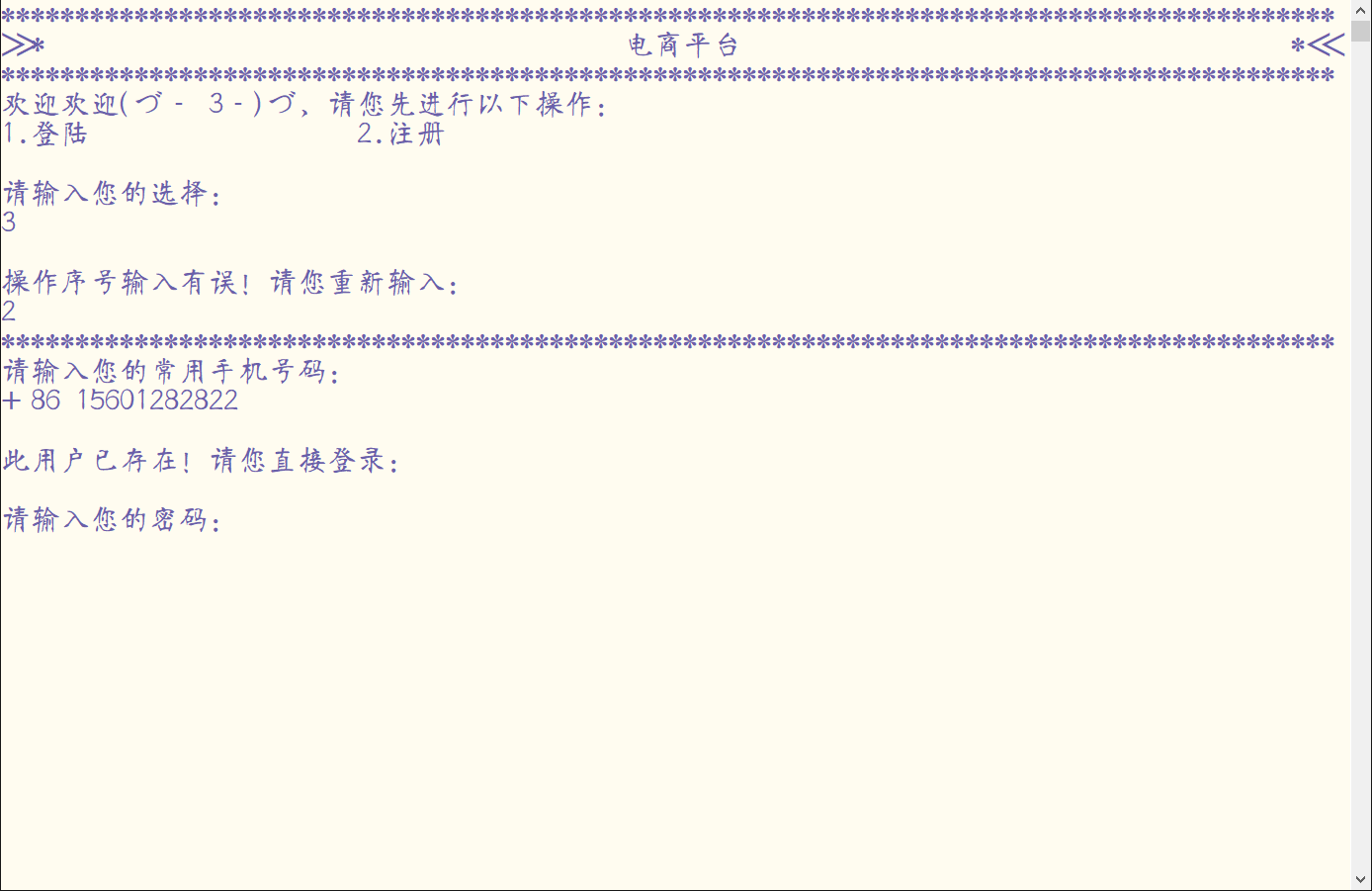


通过输入序号，选择相应操作。若序号输入有误，显示提示信息，并要求重新输入。接下来，逐步演示各模块功能。首先选择注册：

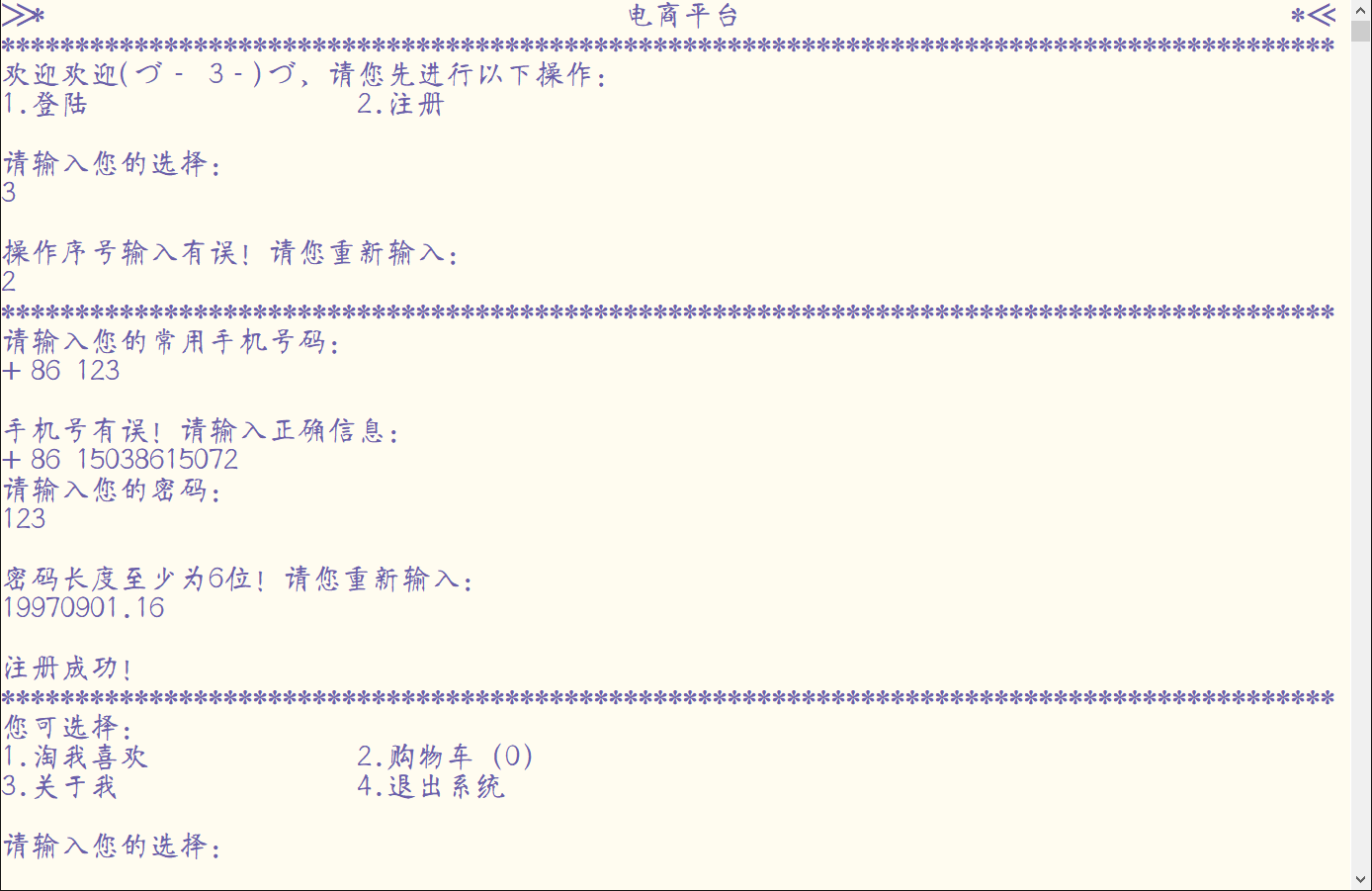


手机号需为11位数字。若输入有误，显示提示信息，并要求重新输入；若账户已存在，输出提示信息，并要求直接登录，输入密码：

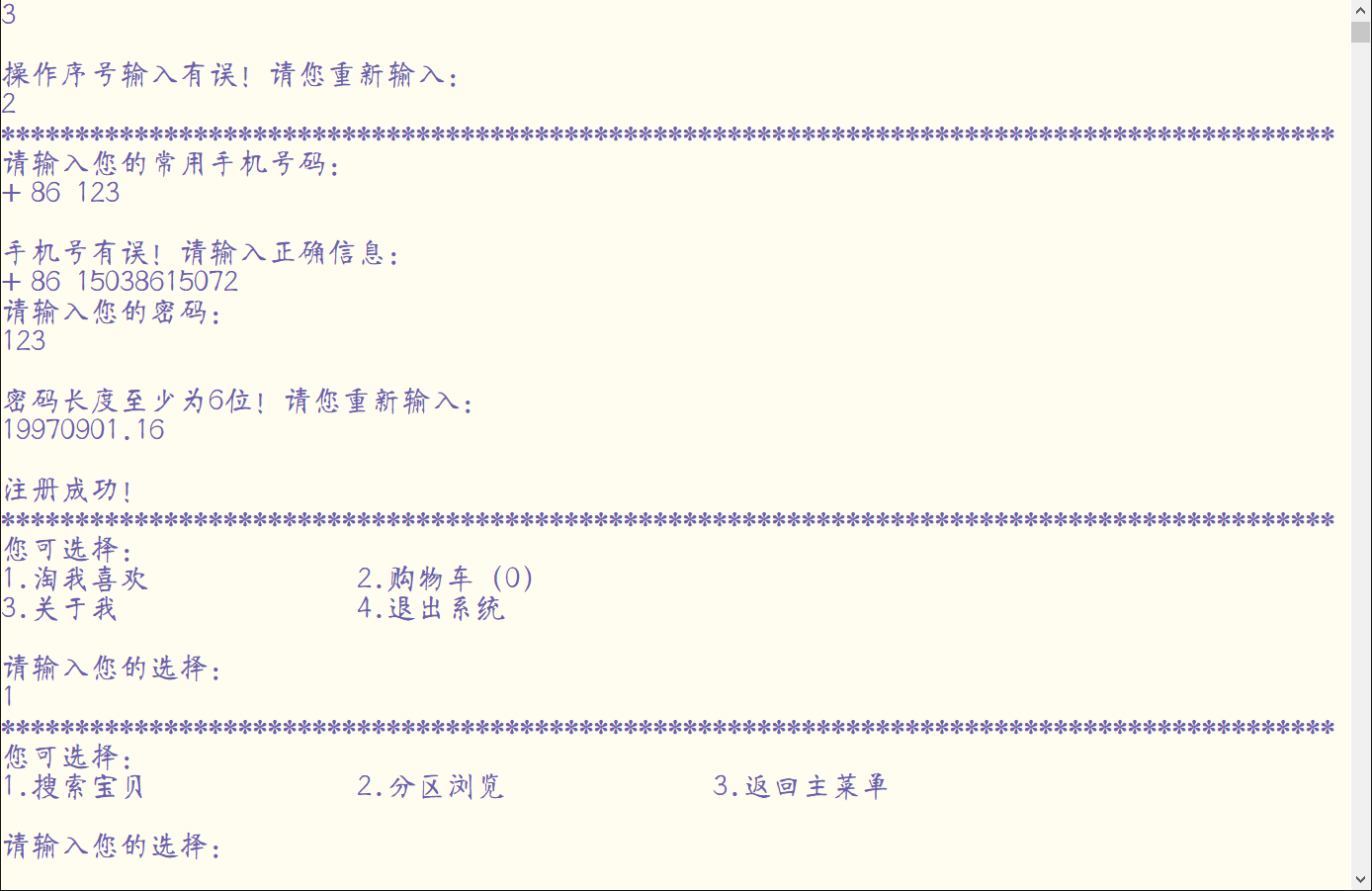




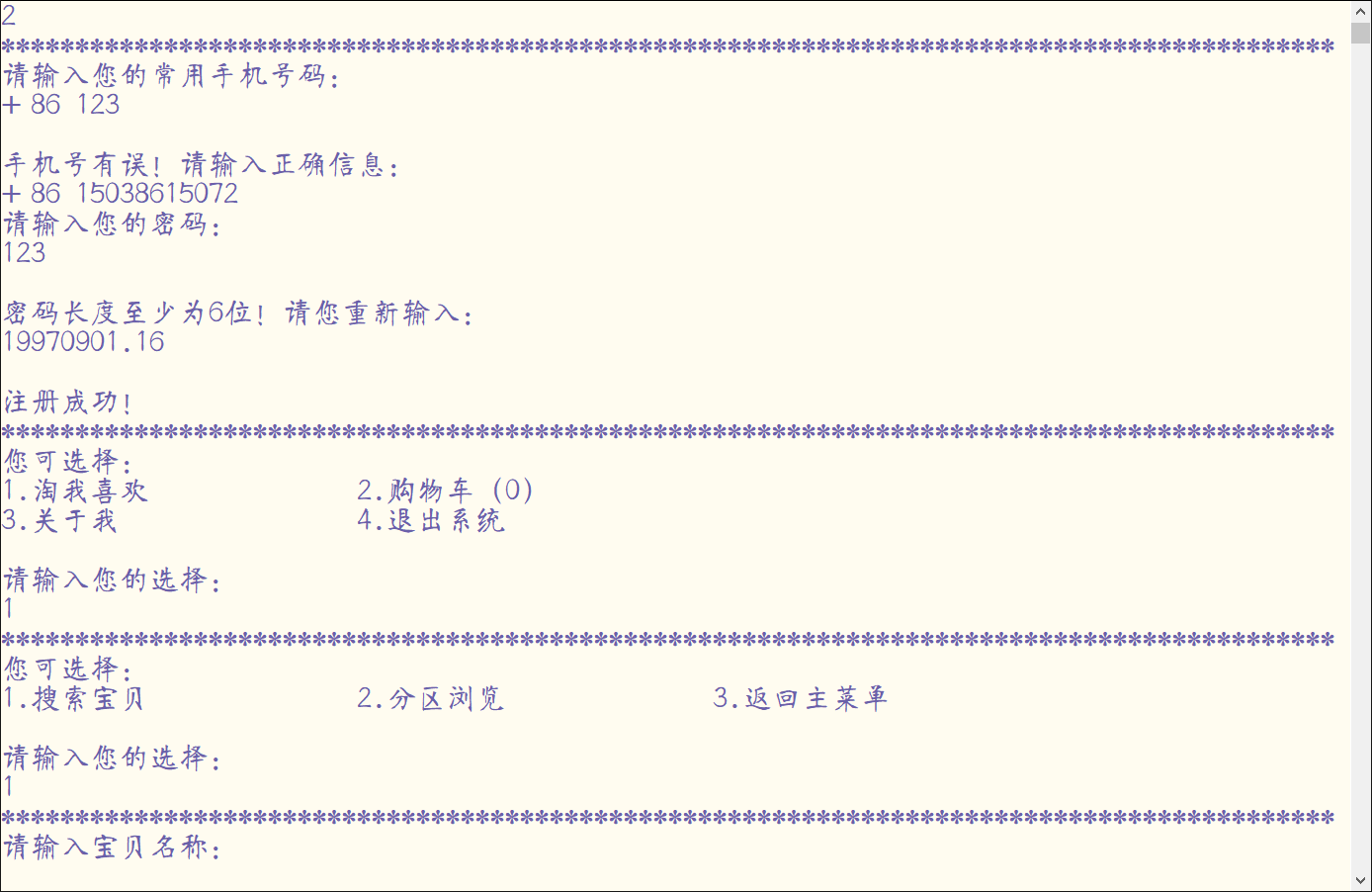
密码需不少于6位，允许出现数字、字母和特殊符号的任意组合。若密码不符合规定，输出提示信息，并要求重新输入：



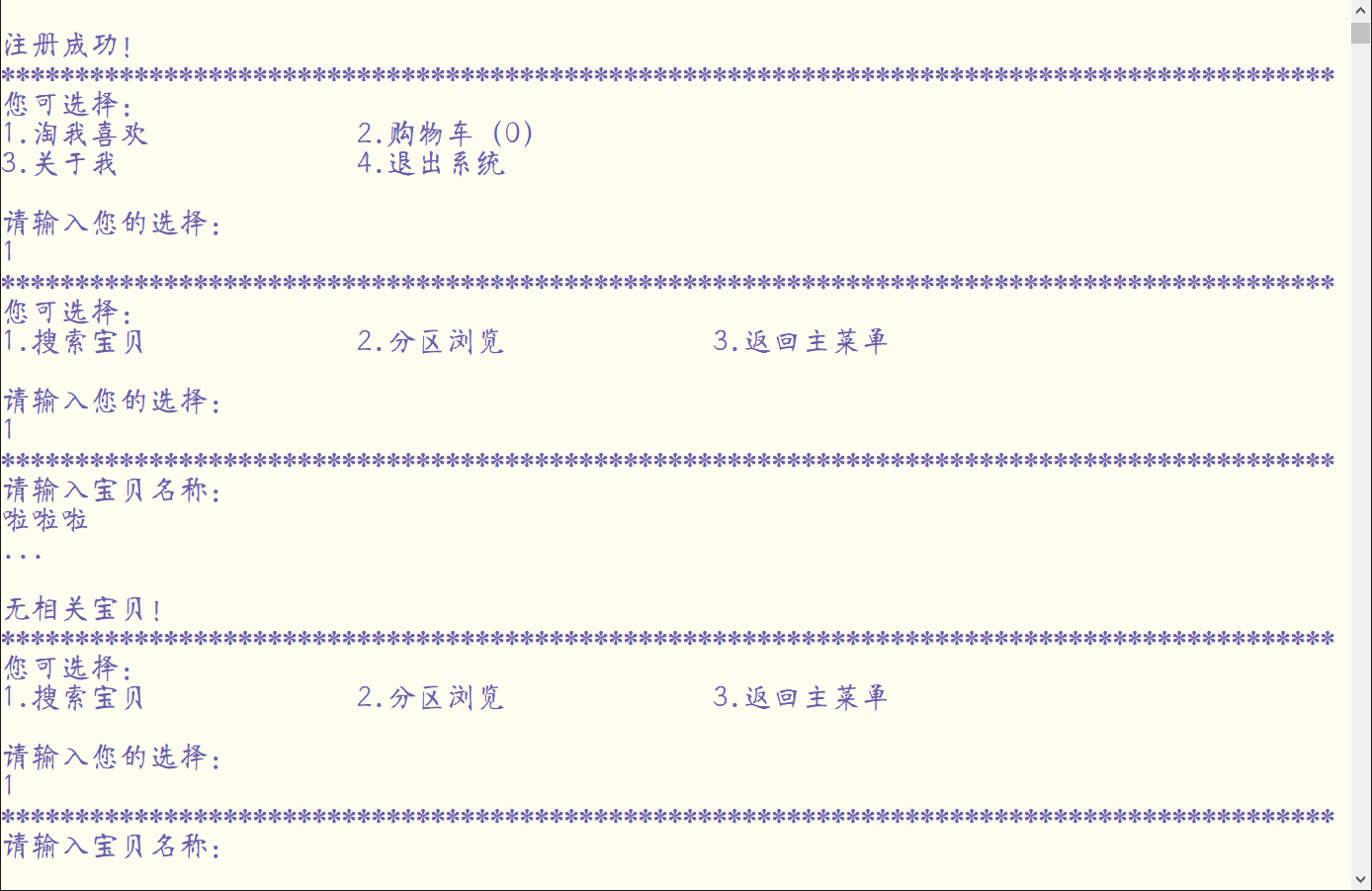
其余功能，先以“淘我喜欢”演示：

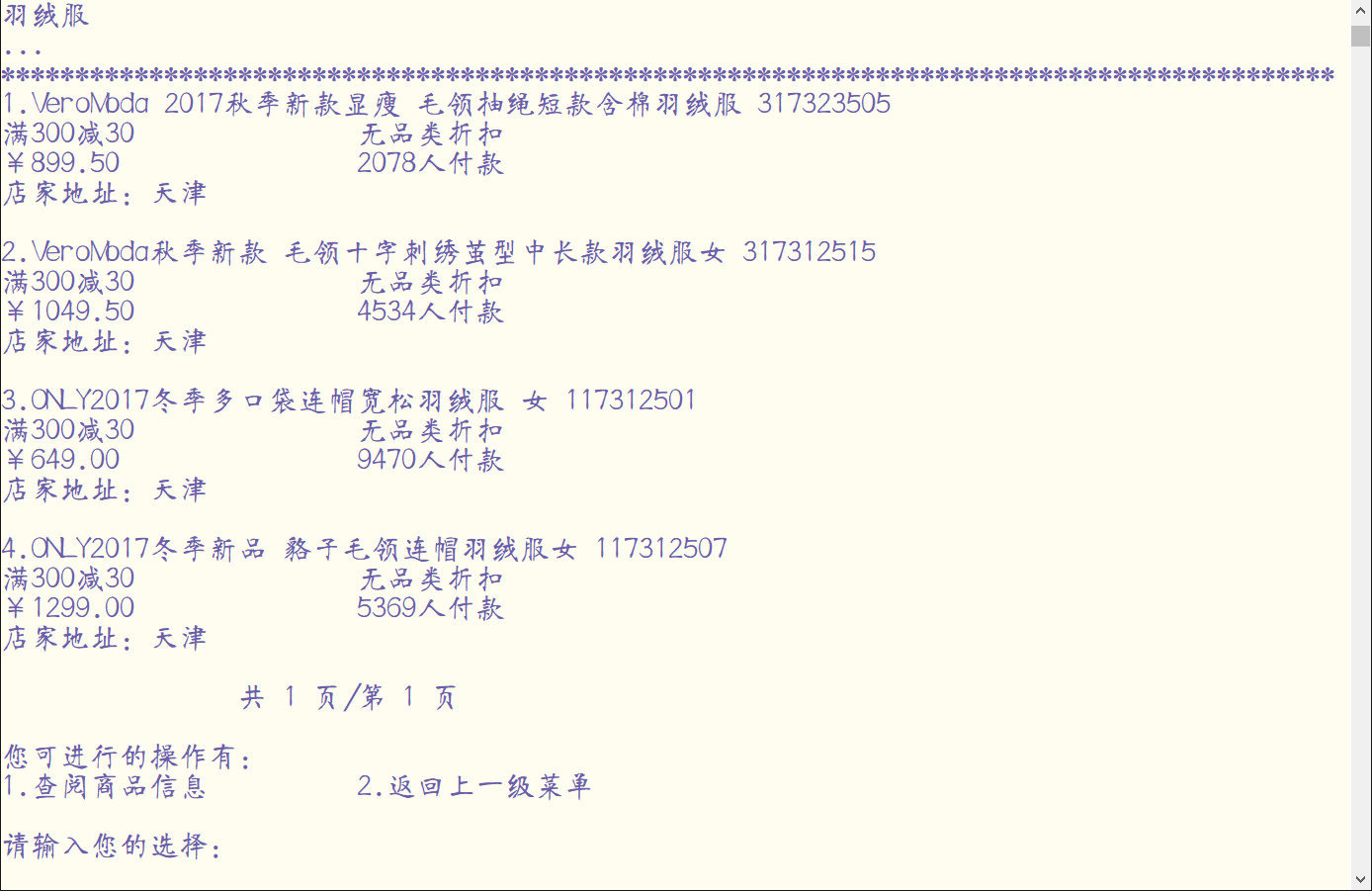


以精确搜索开始：

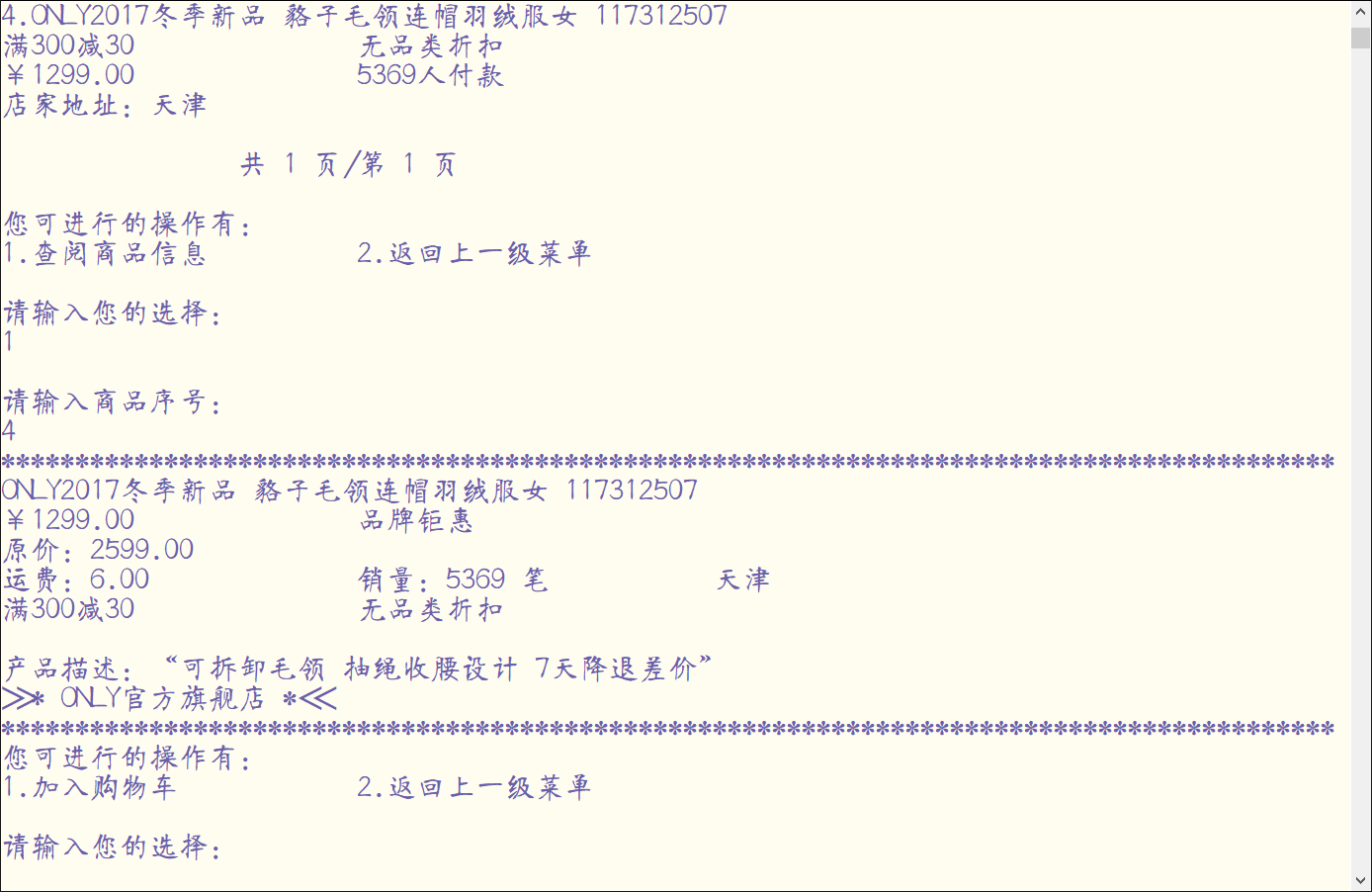


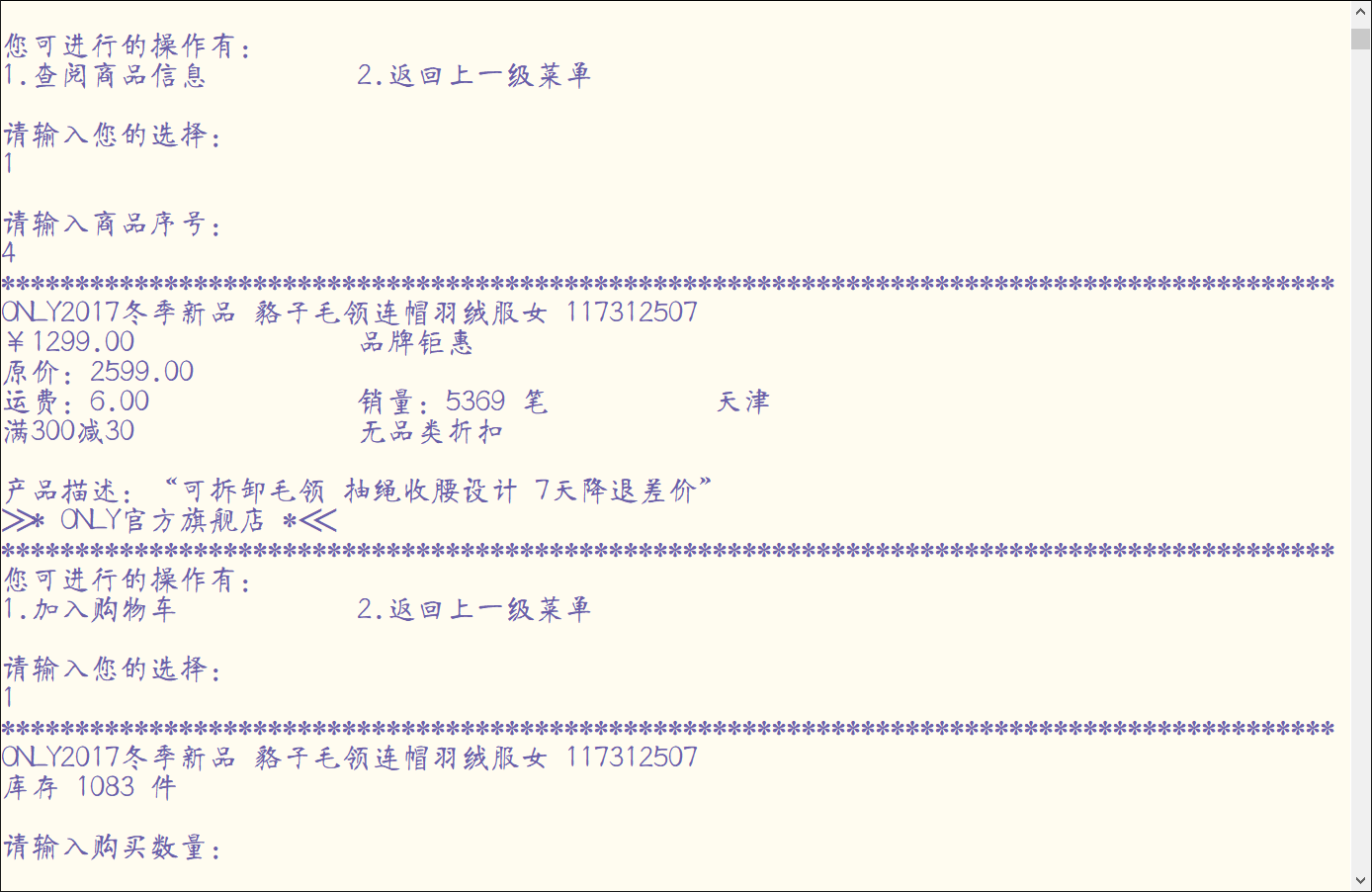
若无搜索结果，输出提示信息；否则，打印相应搜索结果，并可进行后续操作：



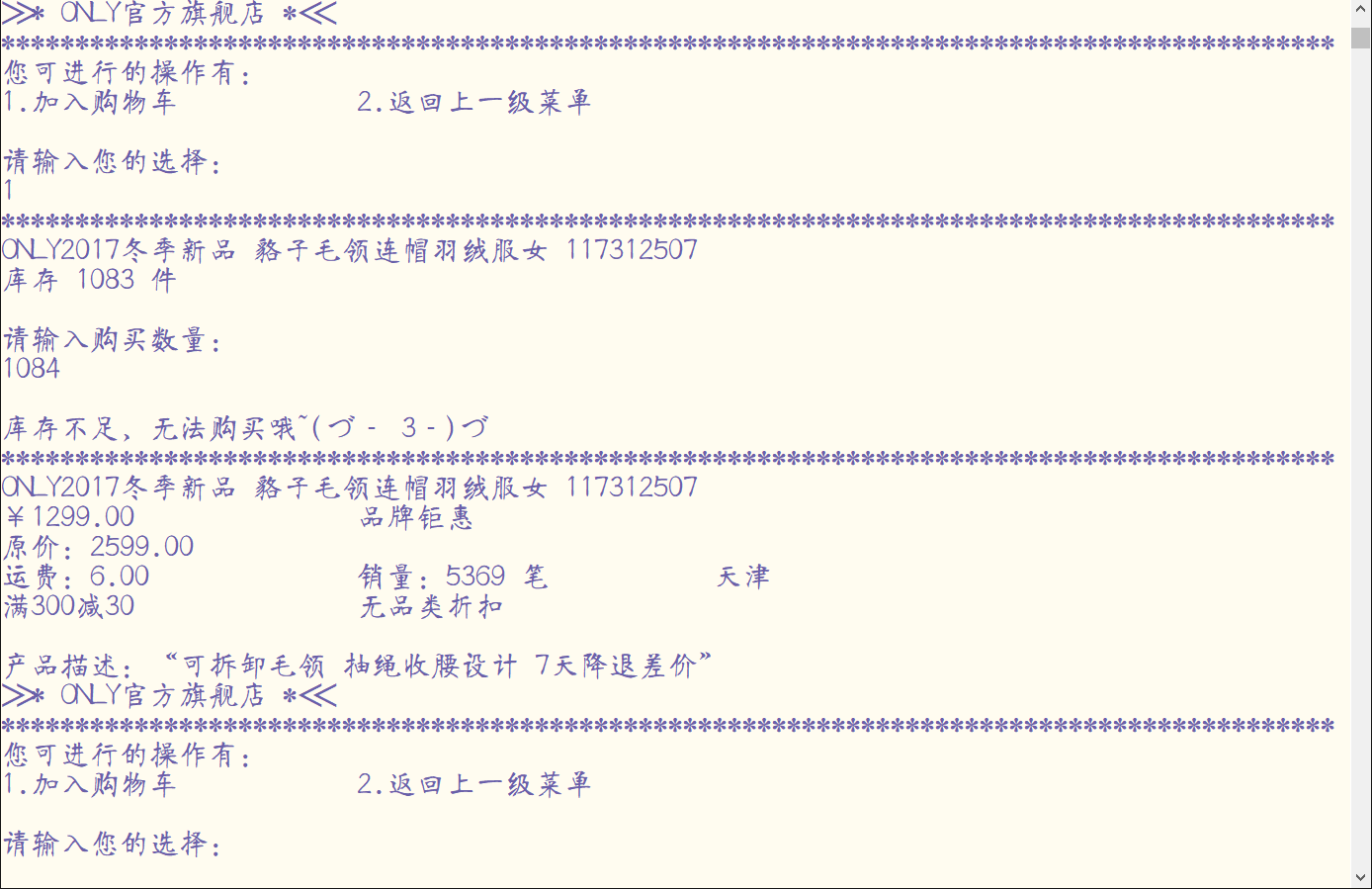


通过输入具体商品序号，查阅其信息：

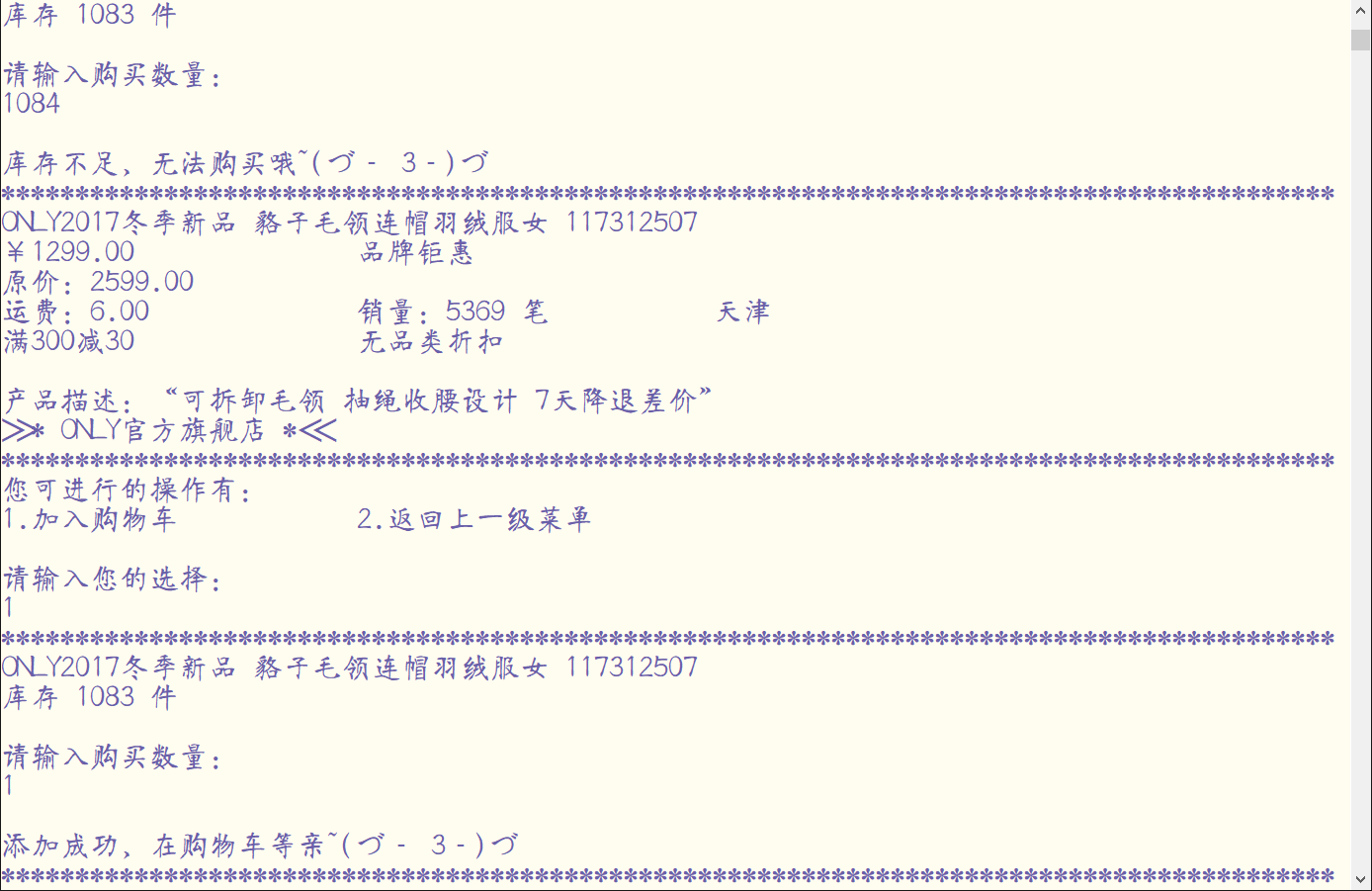




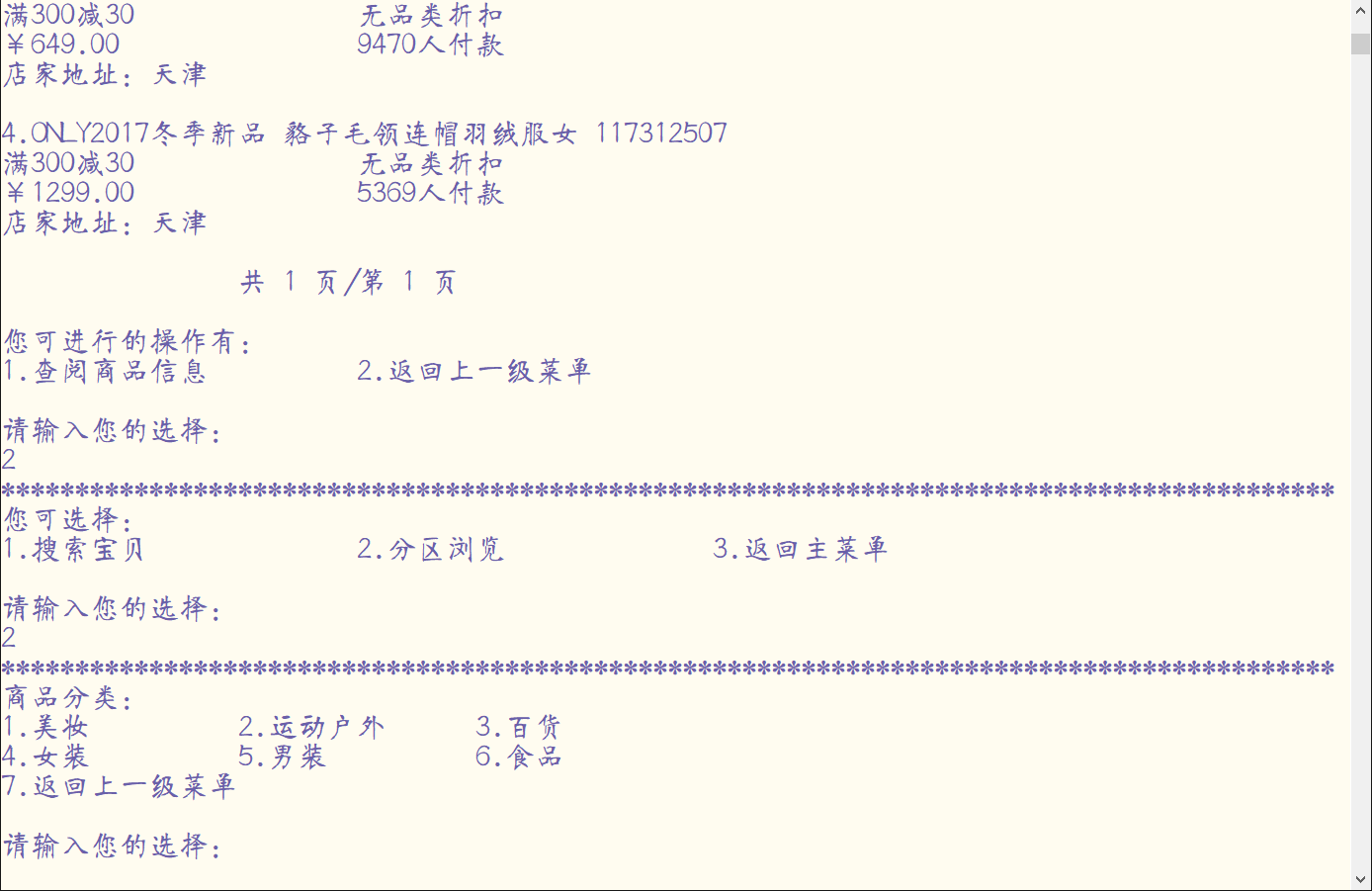
若购买数量超过库存，显示提示信息：

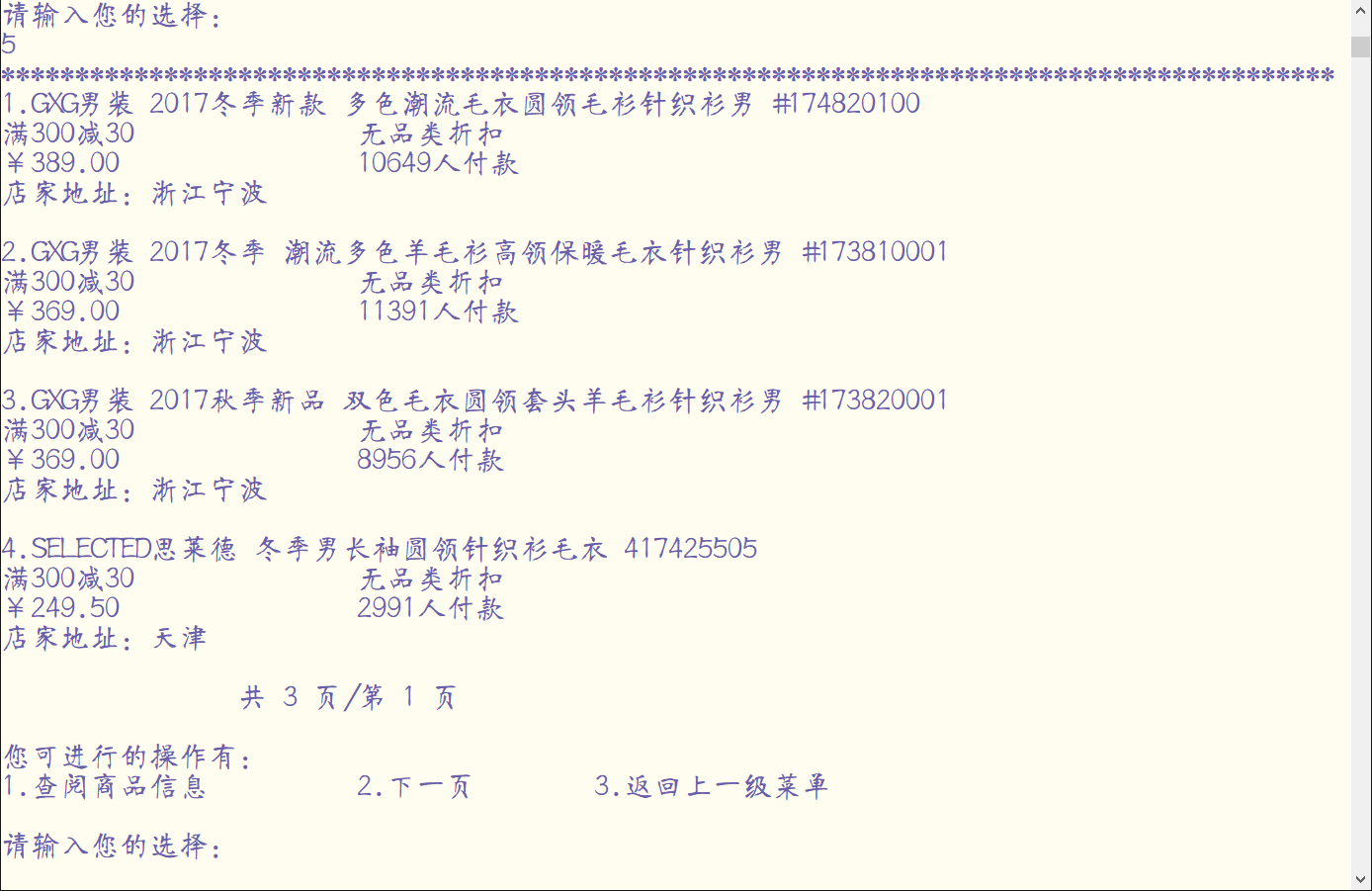


否则，加入购物车成功，输出相应提示信息：

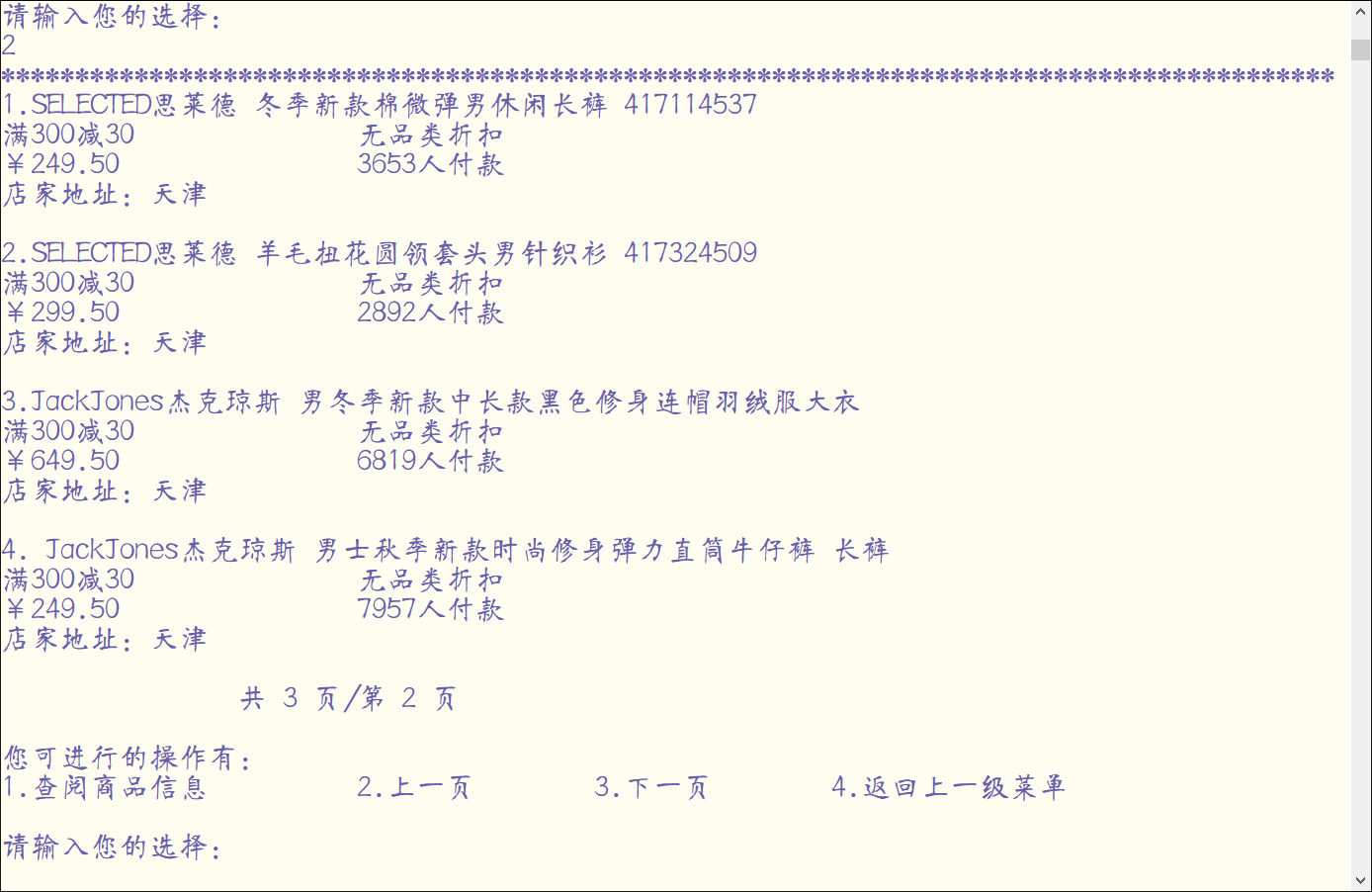


接下来，以“分区浏览”演示：

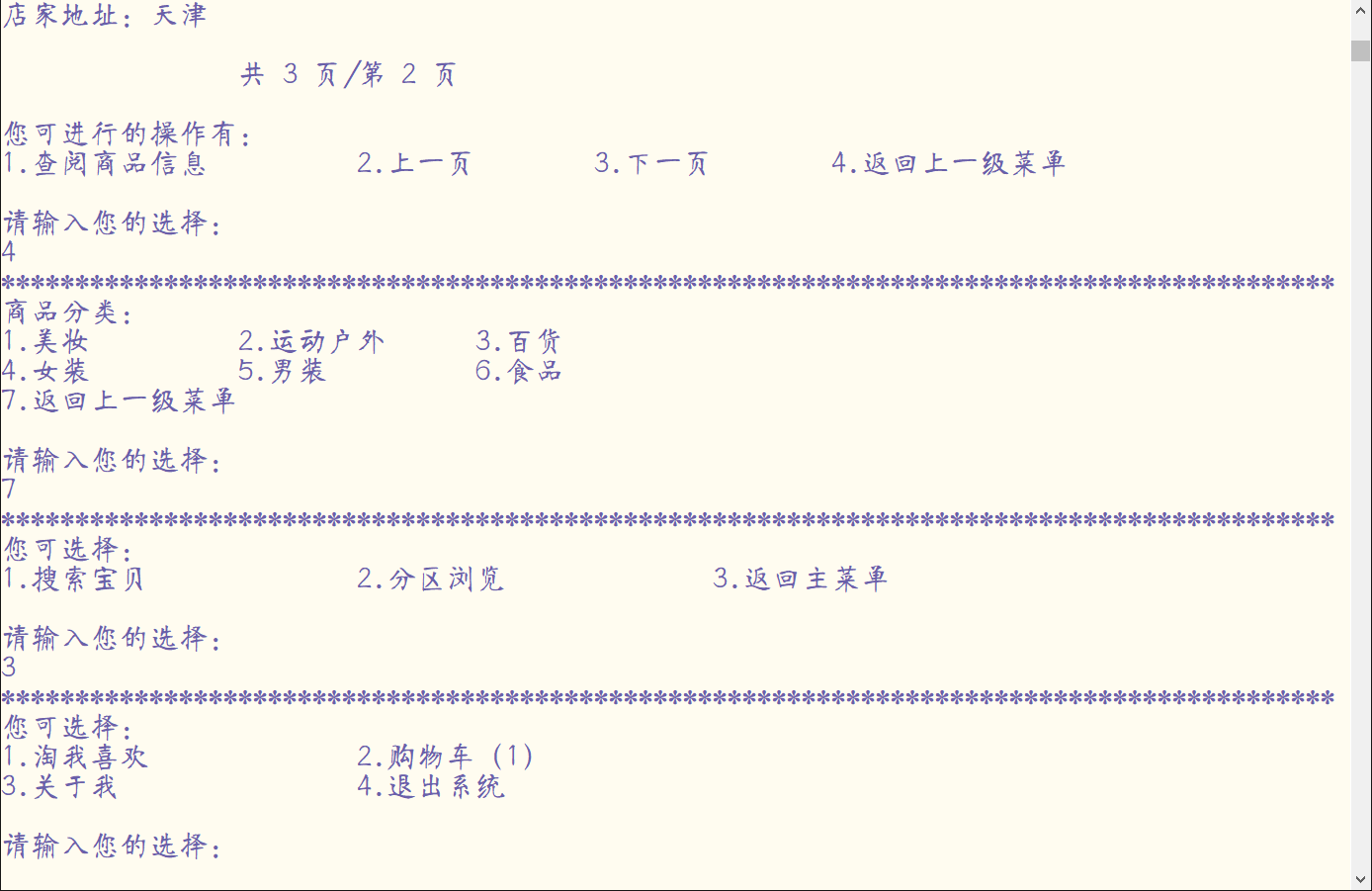




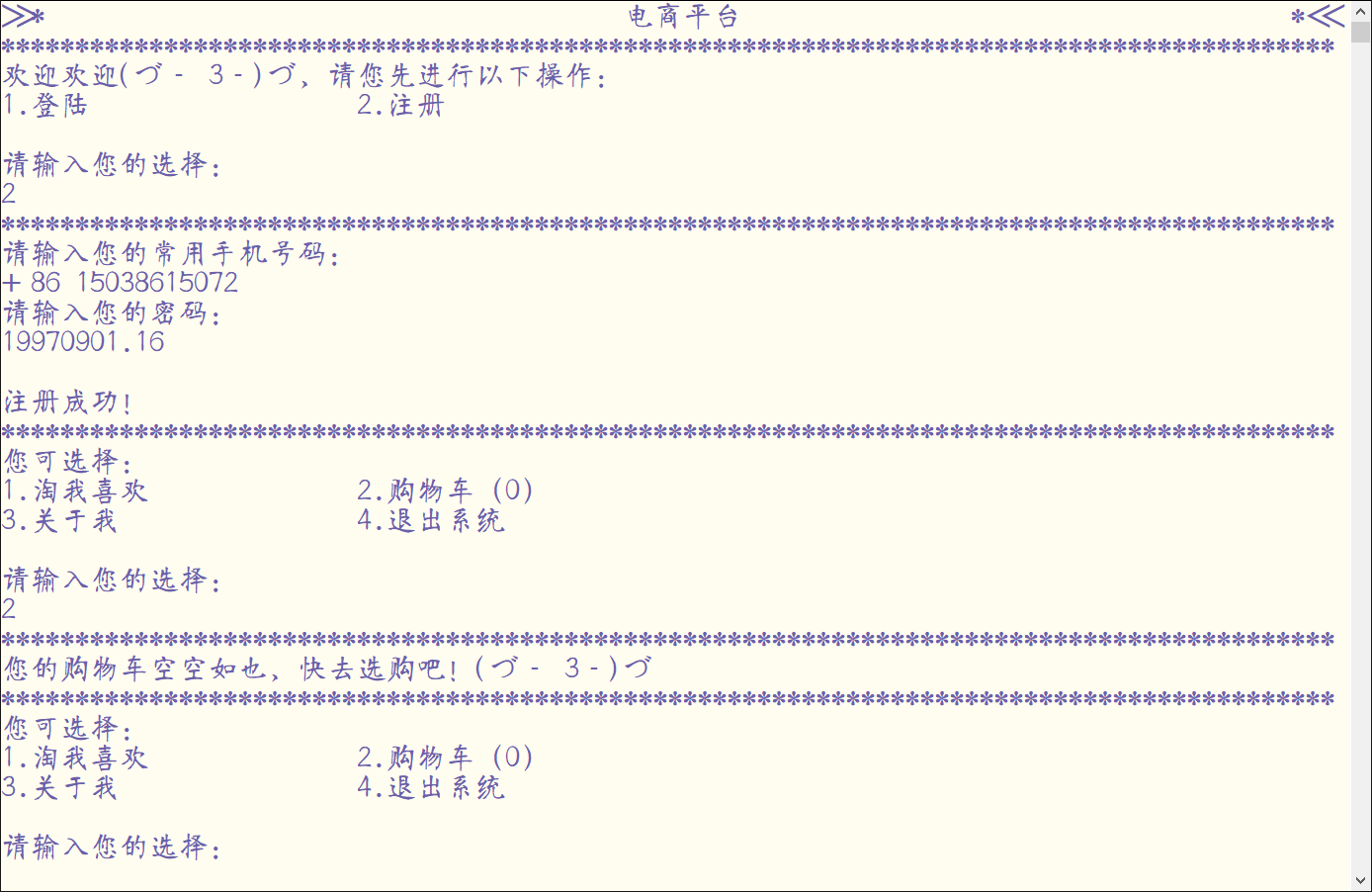
为了较美观，采取每页显示4件商品的方式，并通过手动换页进行浏览：

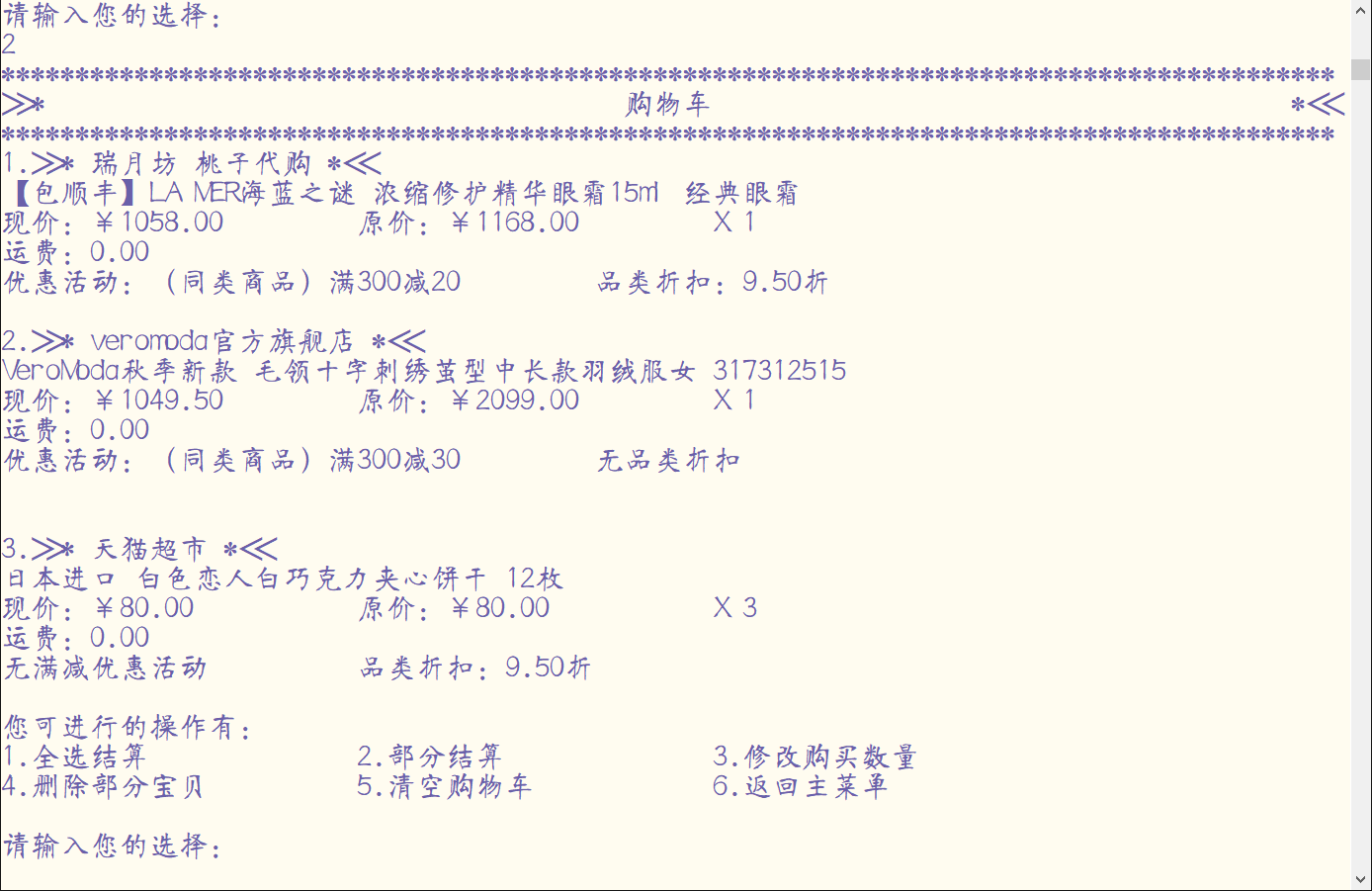


后续操作与精准搜索类似，不再赘述。返回主菜单：

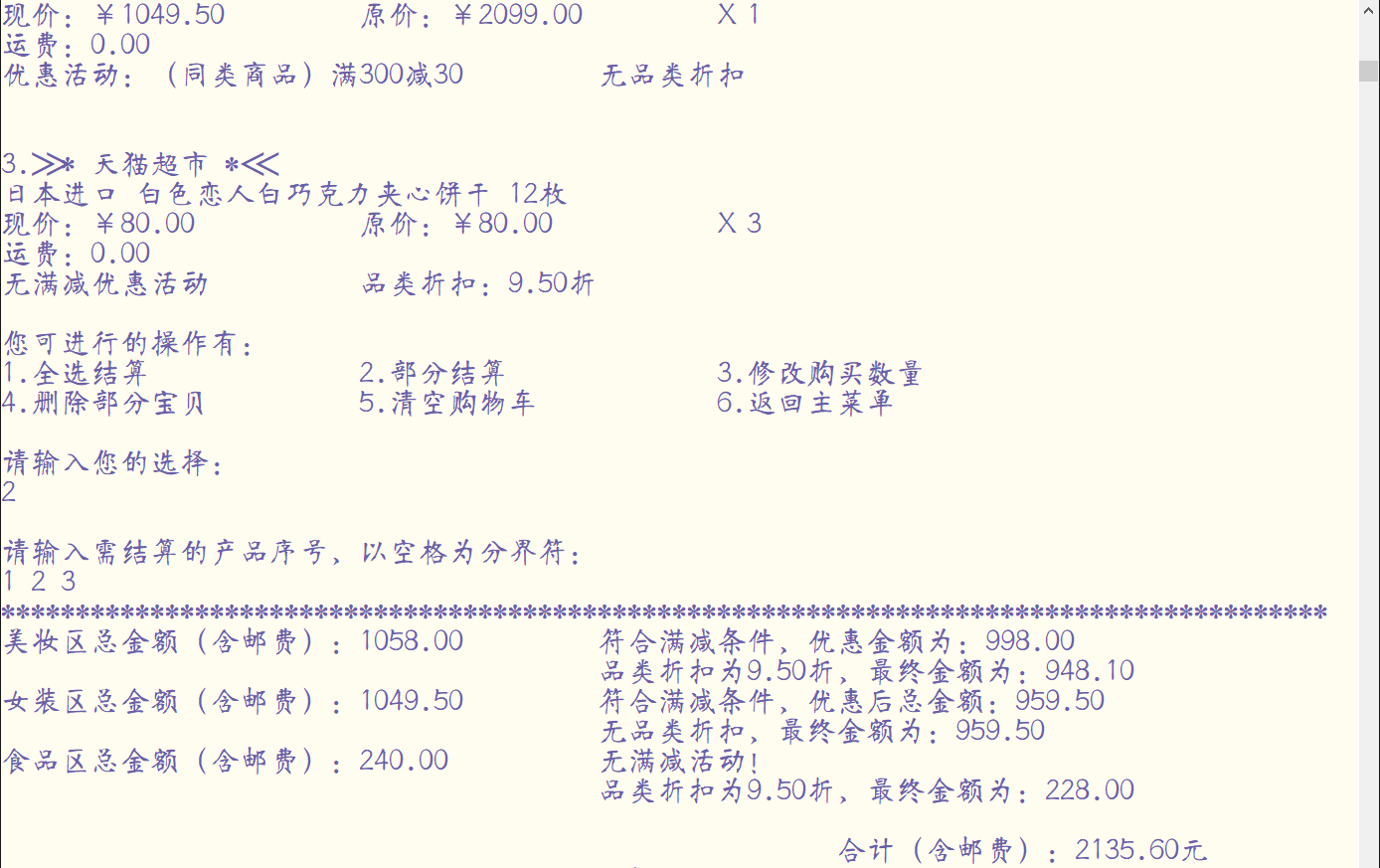


购物车后的数字表示购物车内商品数量。进入购物车，若无商品，输出提示信息，否则打印商品：

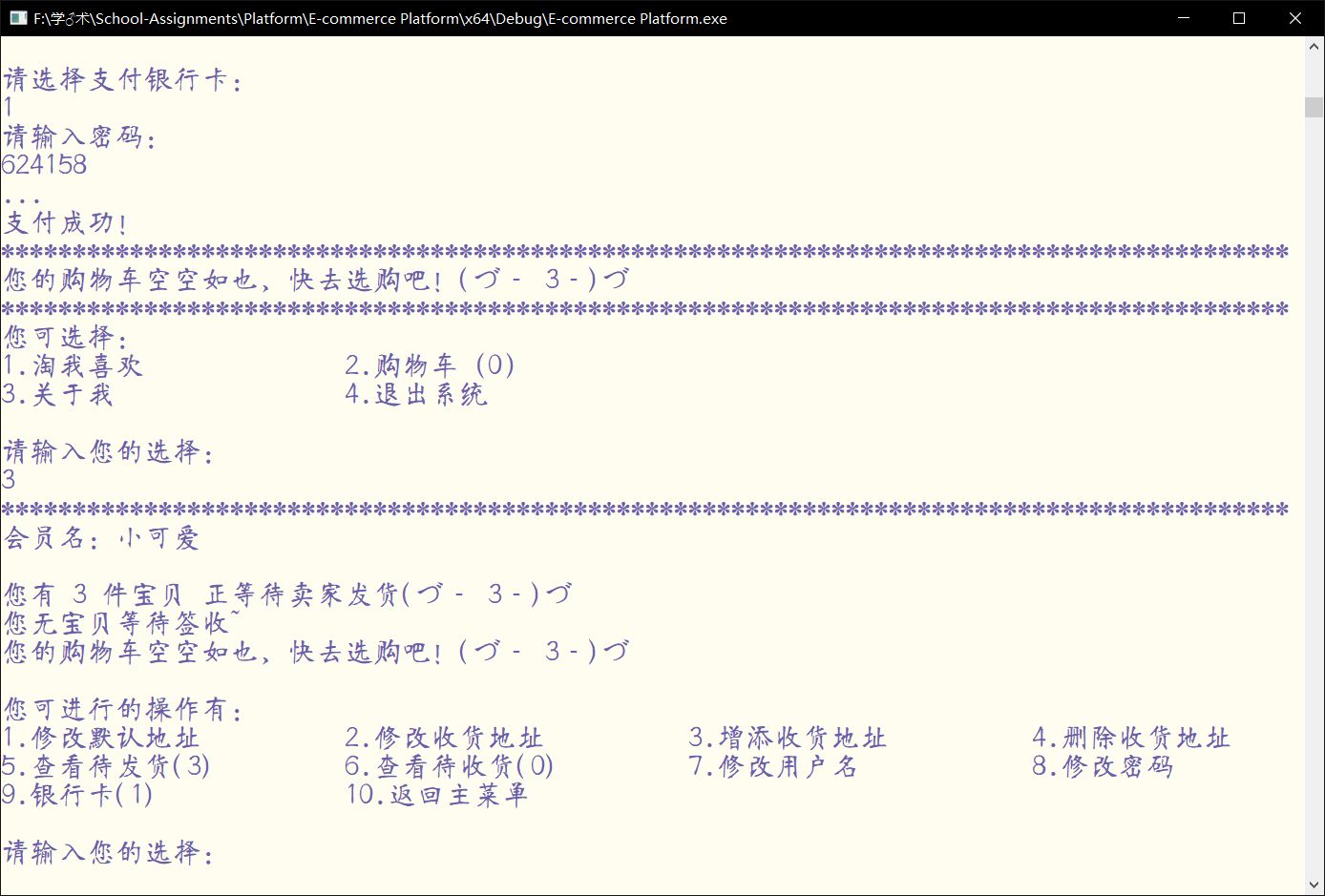


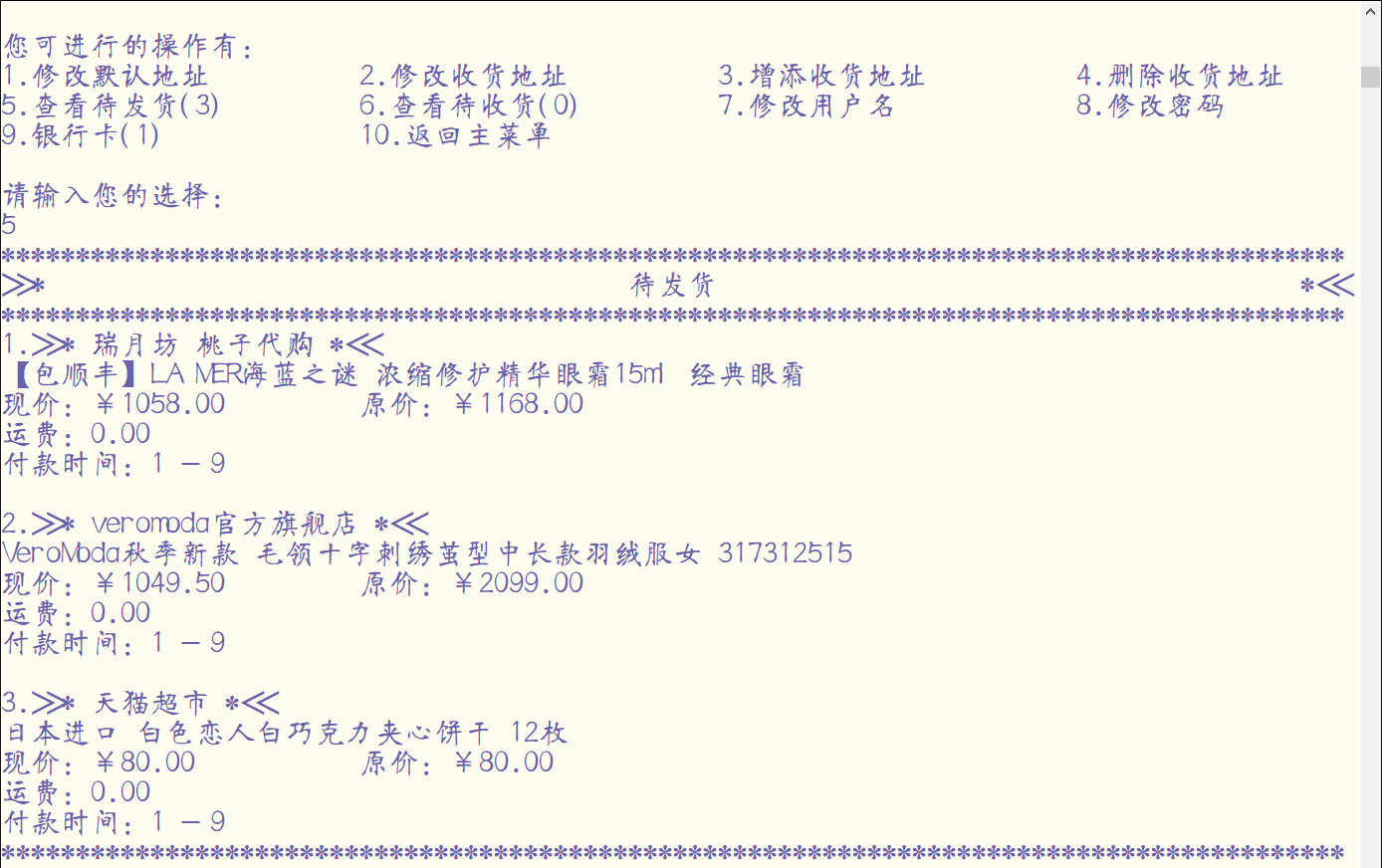


全选结算与部分结算功能类似，以“部分结算”为例：

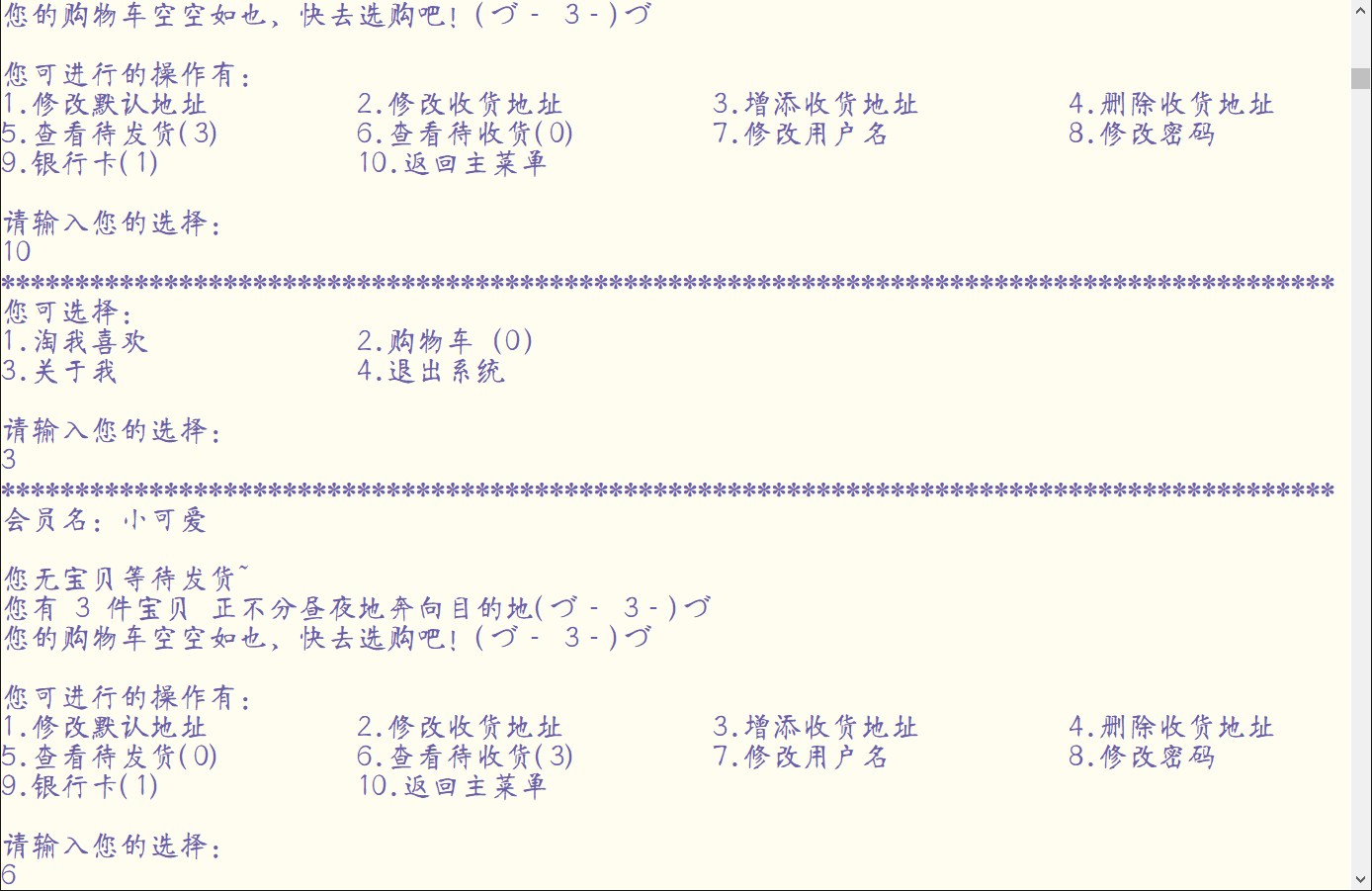


已付款商品，自动归于待发货类别。系统将在1~7天内随机发货：

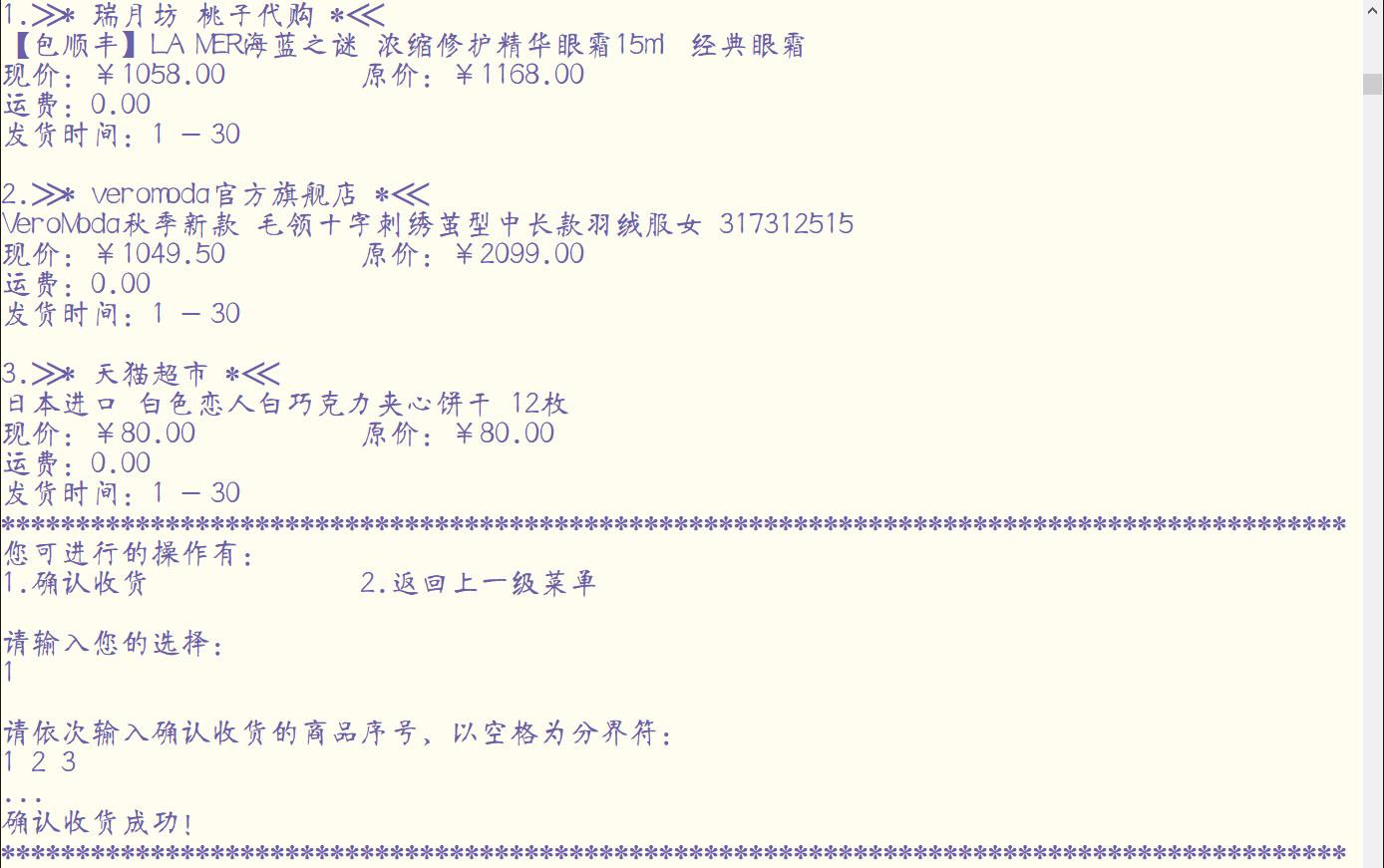




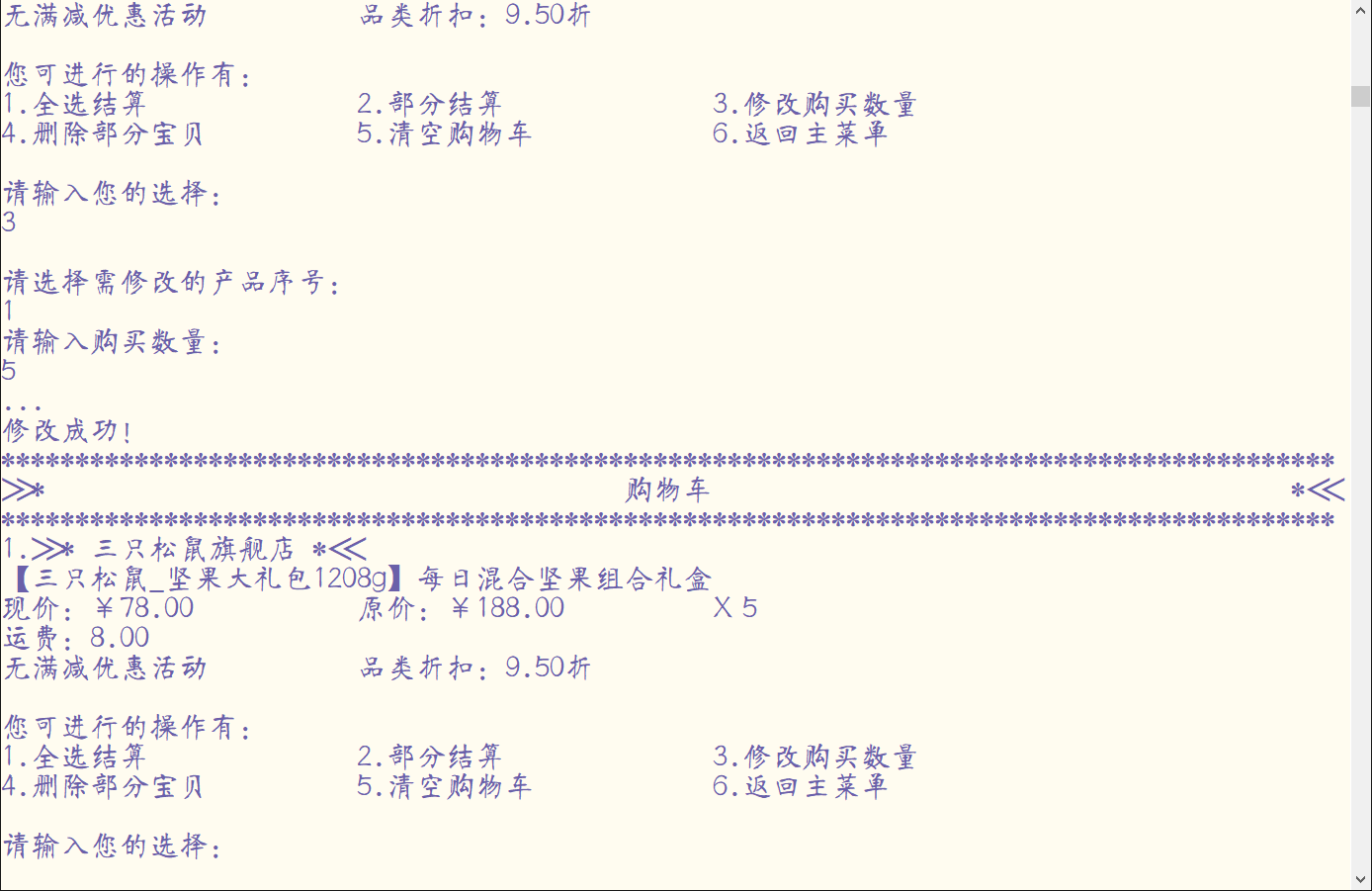
发货后商品归于待收货类别，需要用户手动收货：



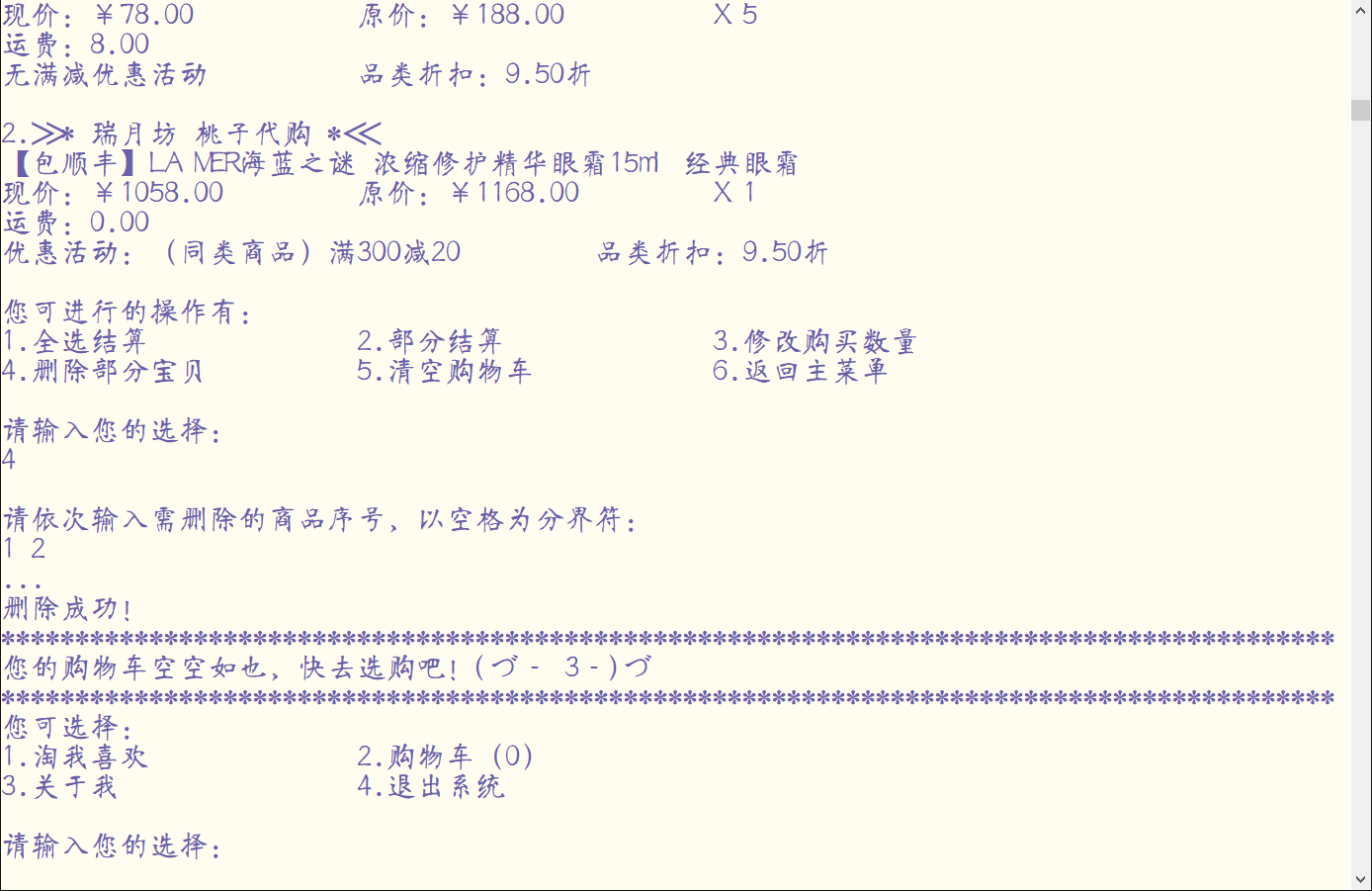




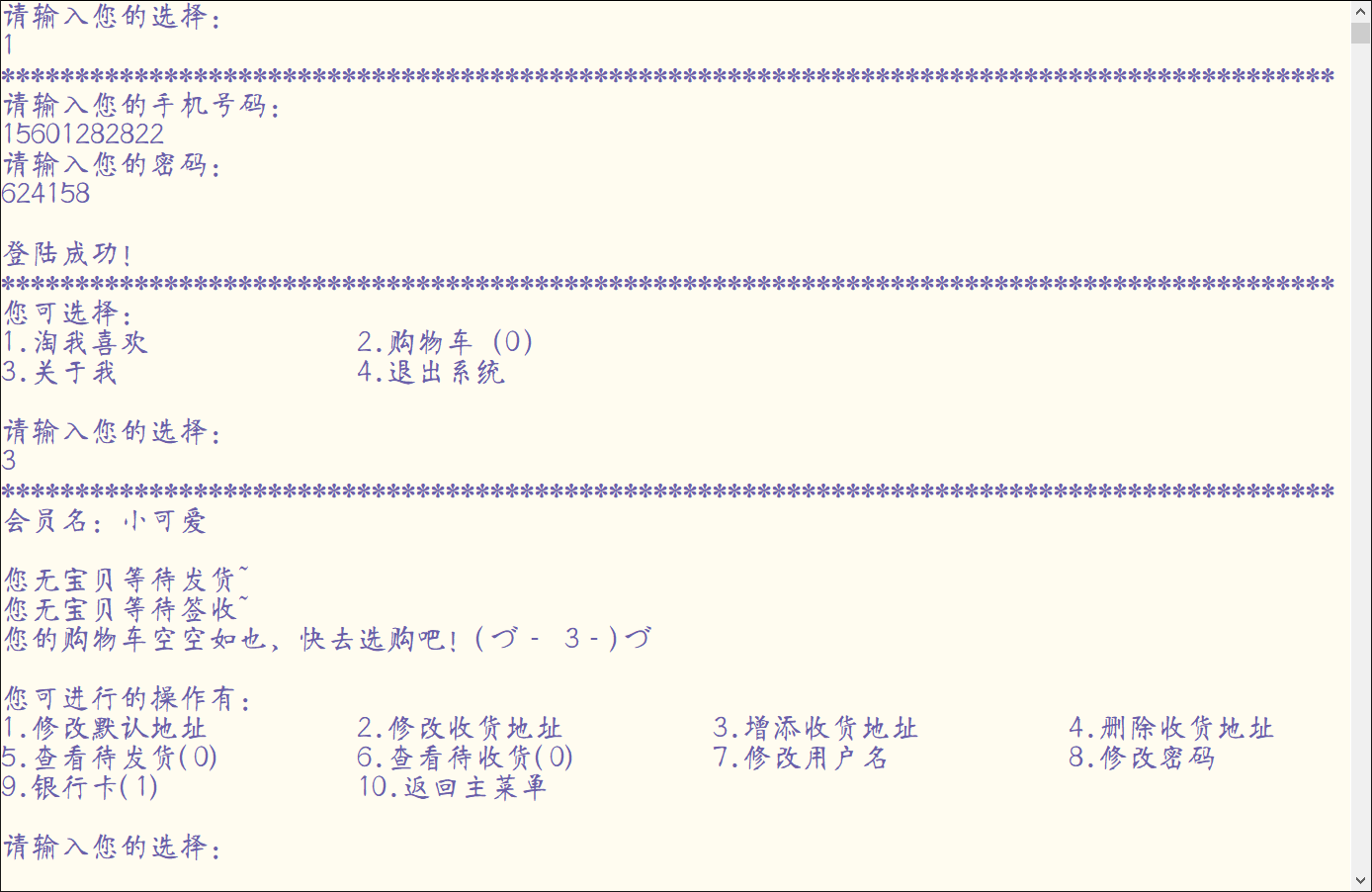
若在购物车菜单选择“修改购买数量”：



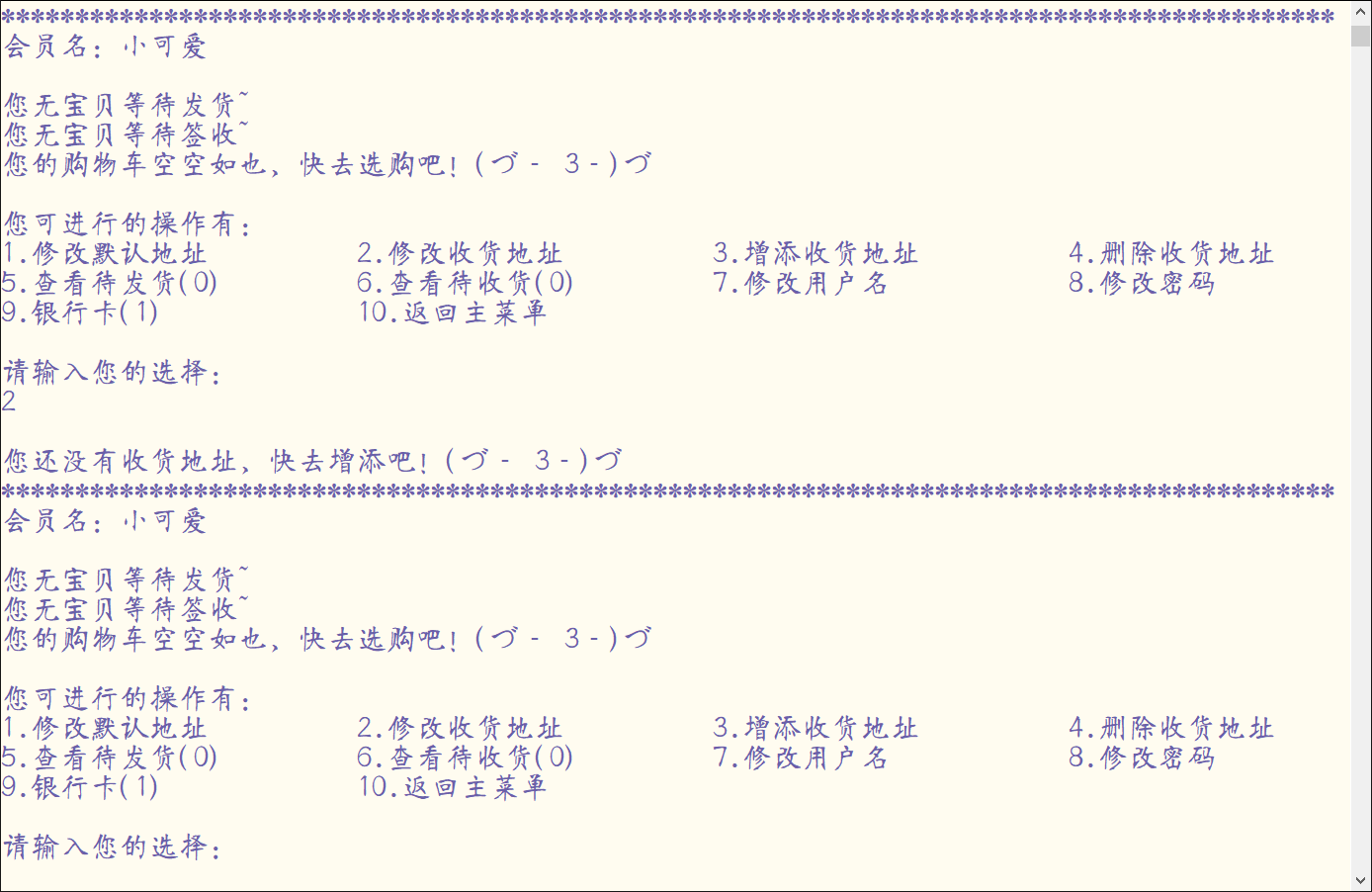
清空购物车与删除部分宝贝功能类似。以“删除部分宝贝”为例：



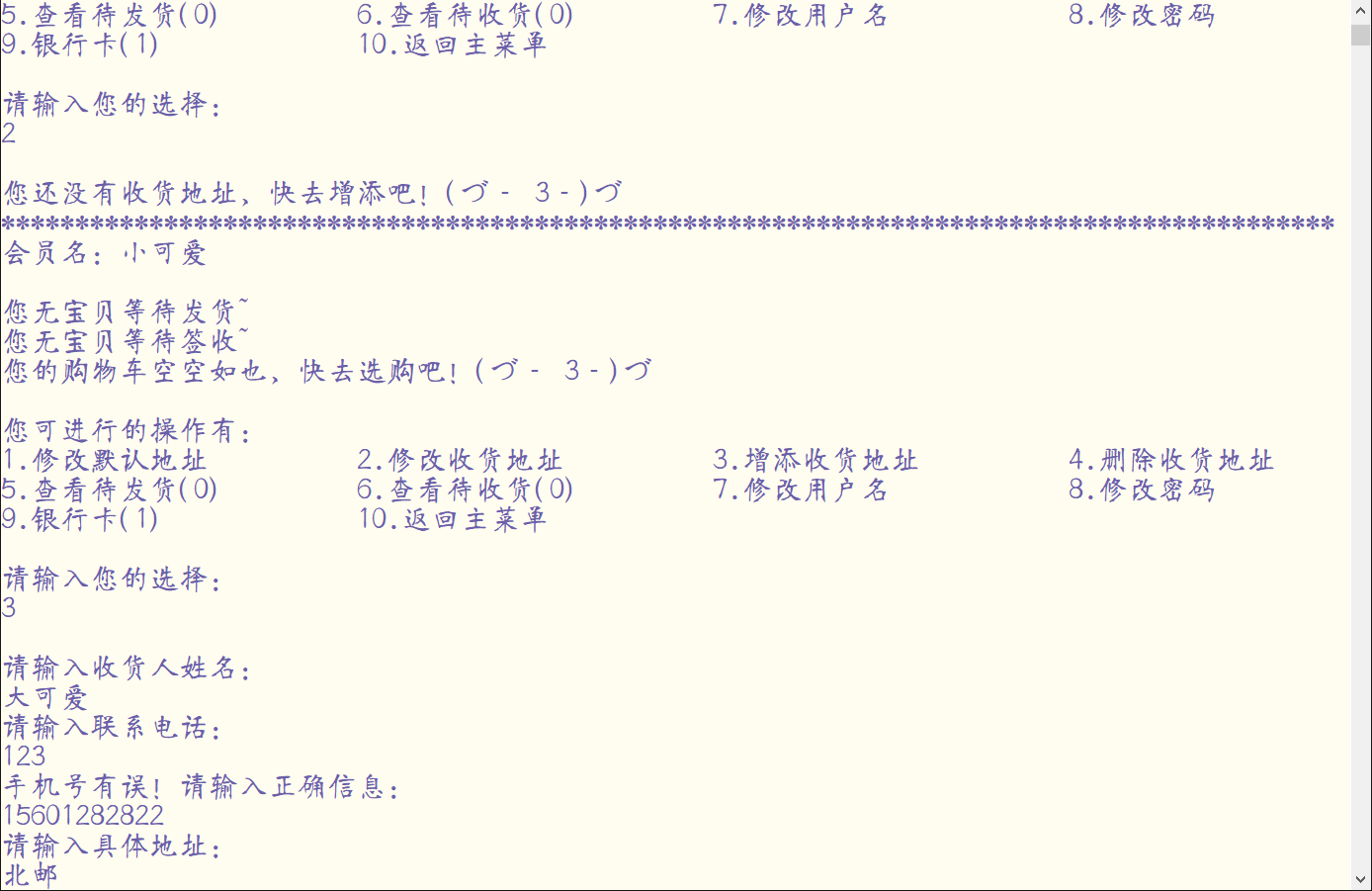
接下来，进入账户信息界面：



在无收货地址情况下，若选择修改默认地址或收货地址，会输出相应提示信息：



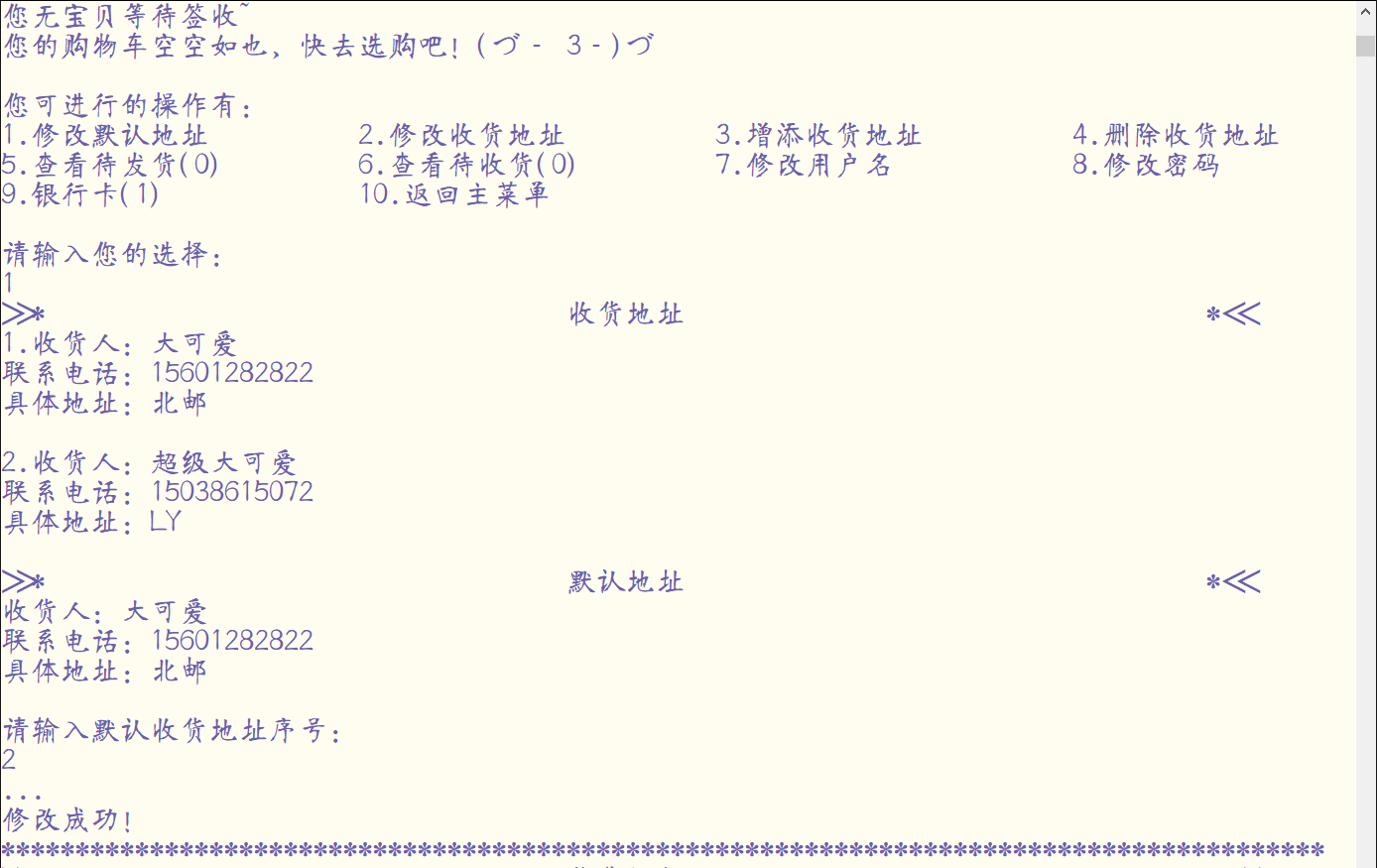
因而，先演示“增添收货地址”：

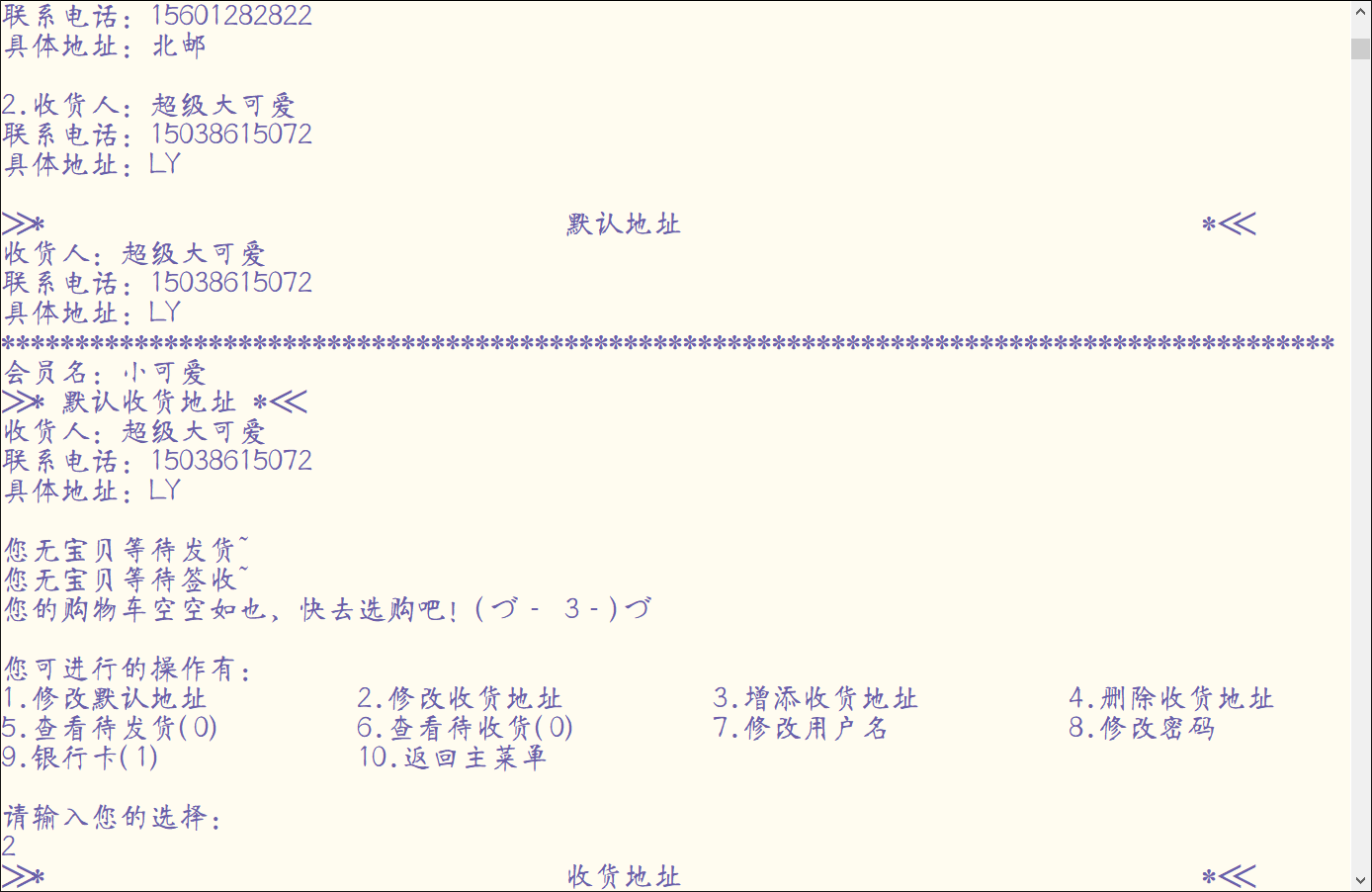


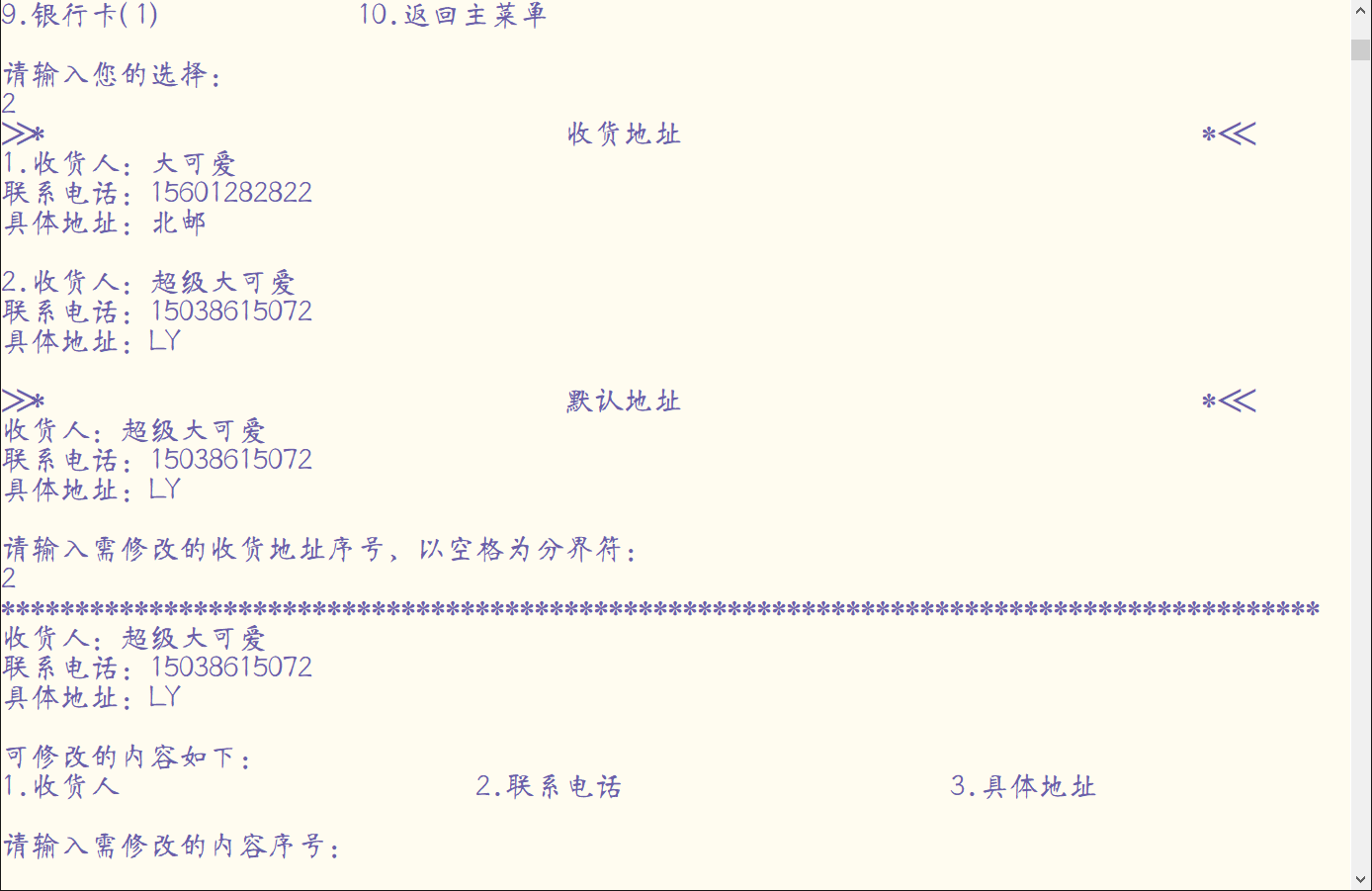
若只有一个收货地址，则系统自动将其设为默认收货地址：

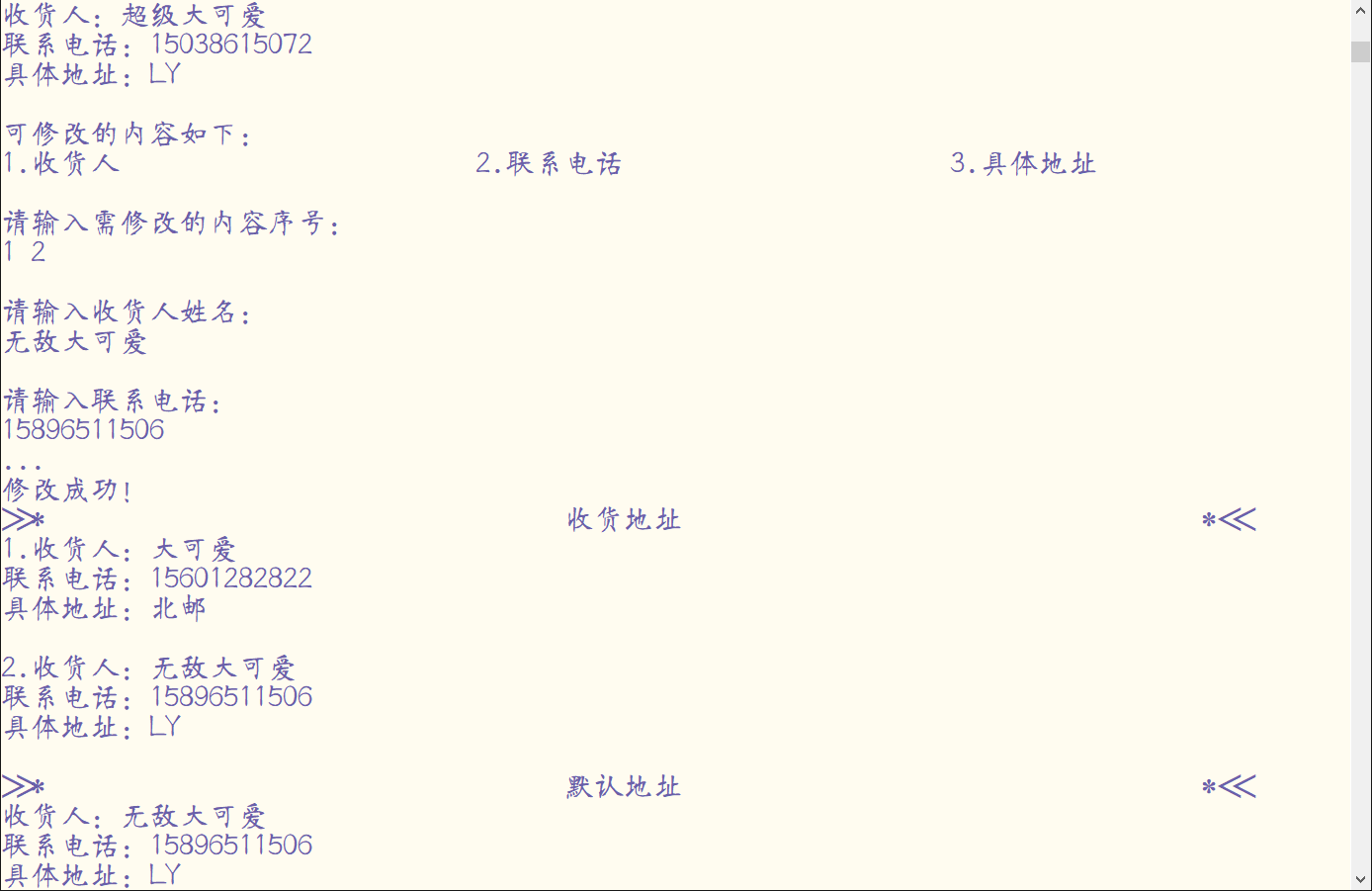


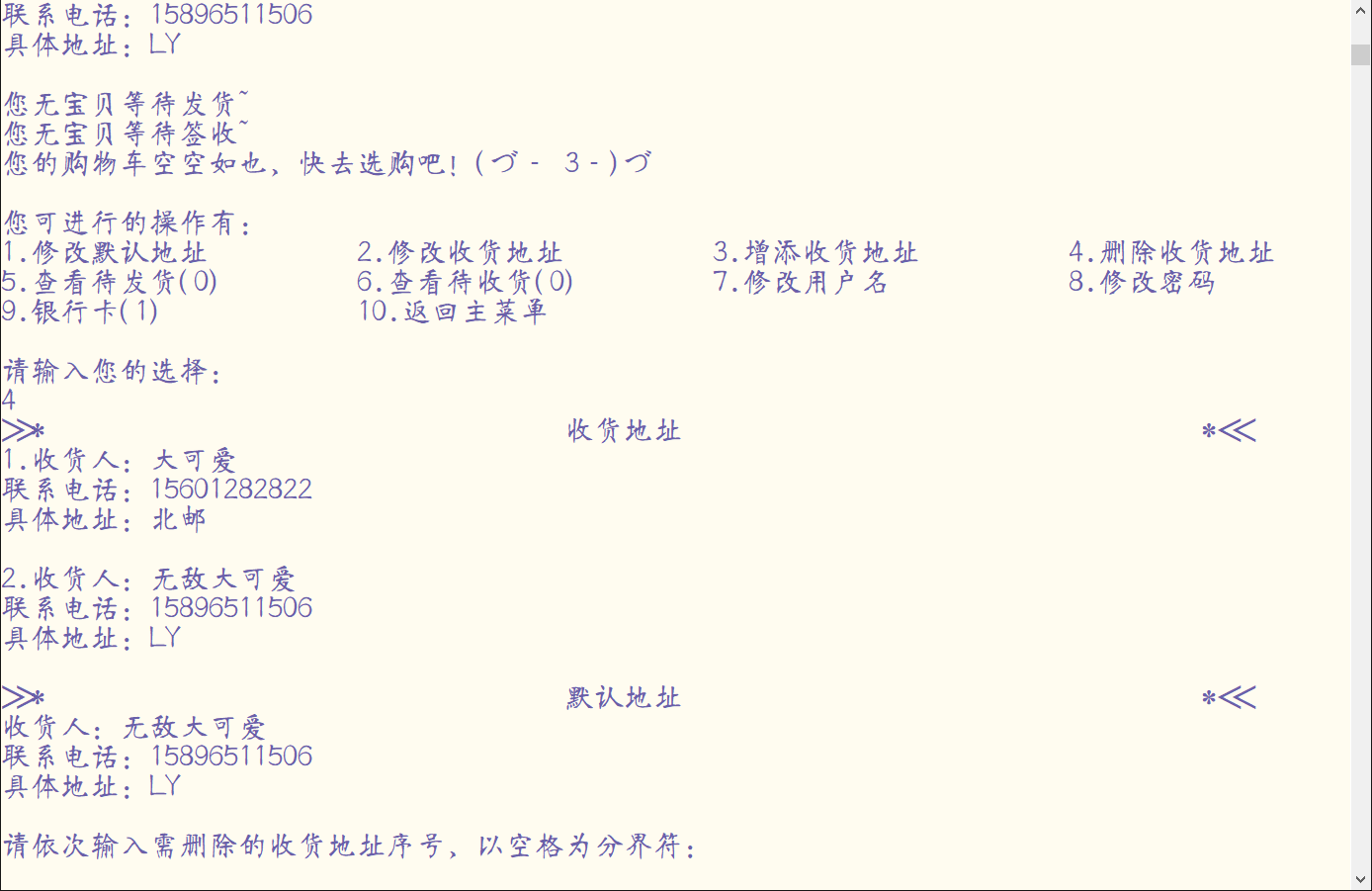
在此基础上，可以进行修改默认地址、收货地址或删除收货地址的操作：

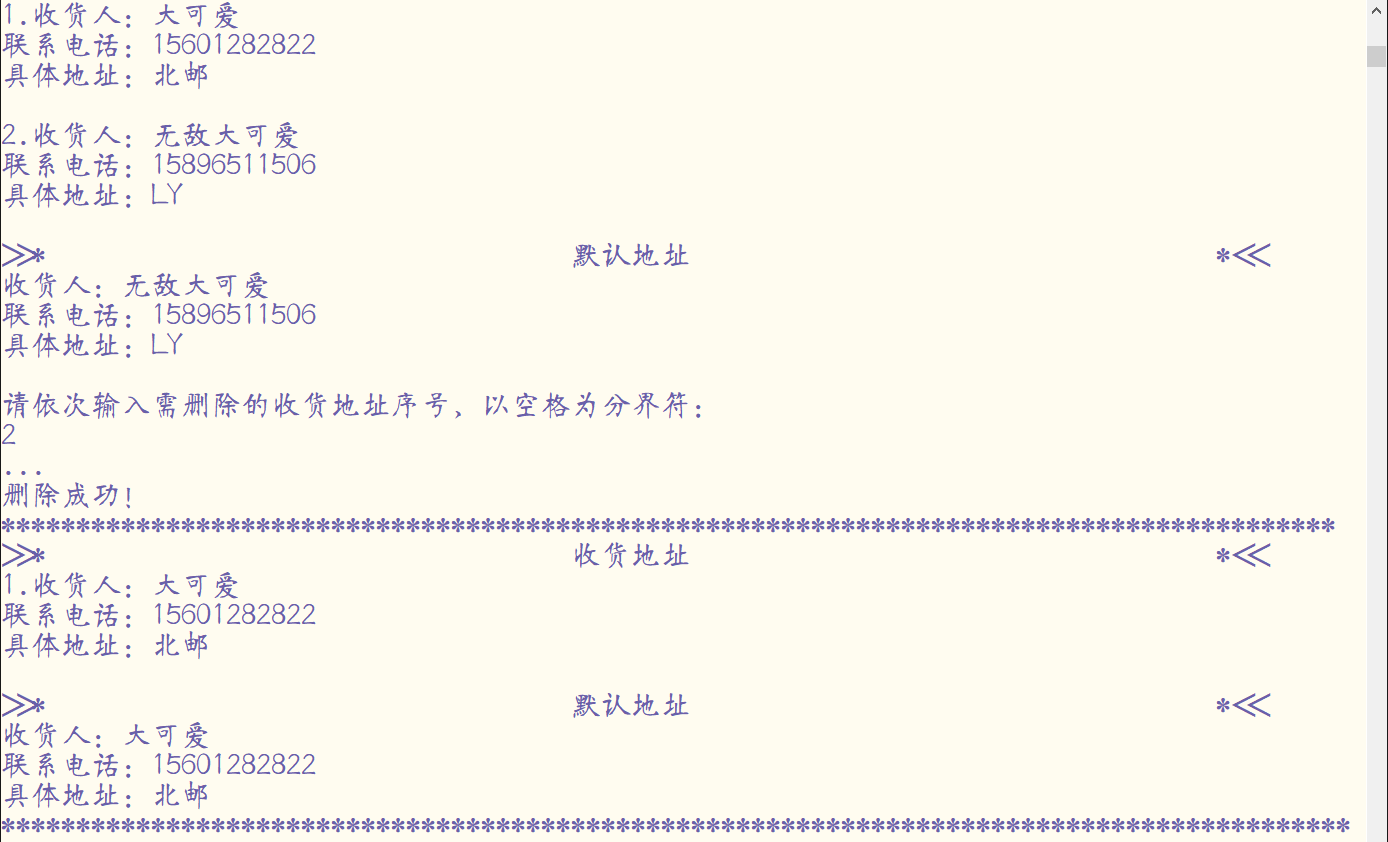




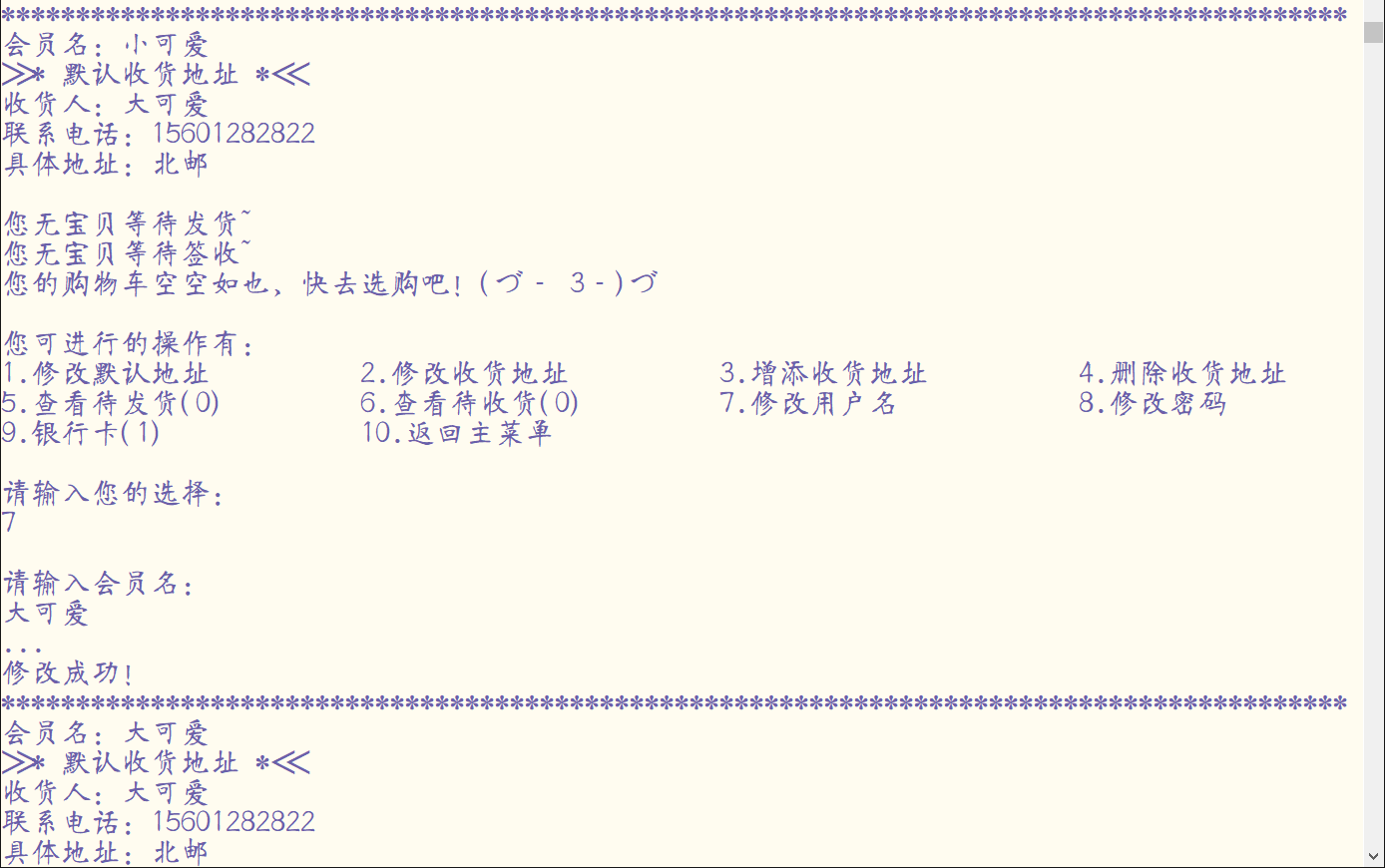




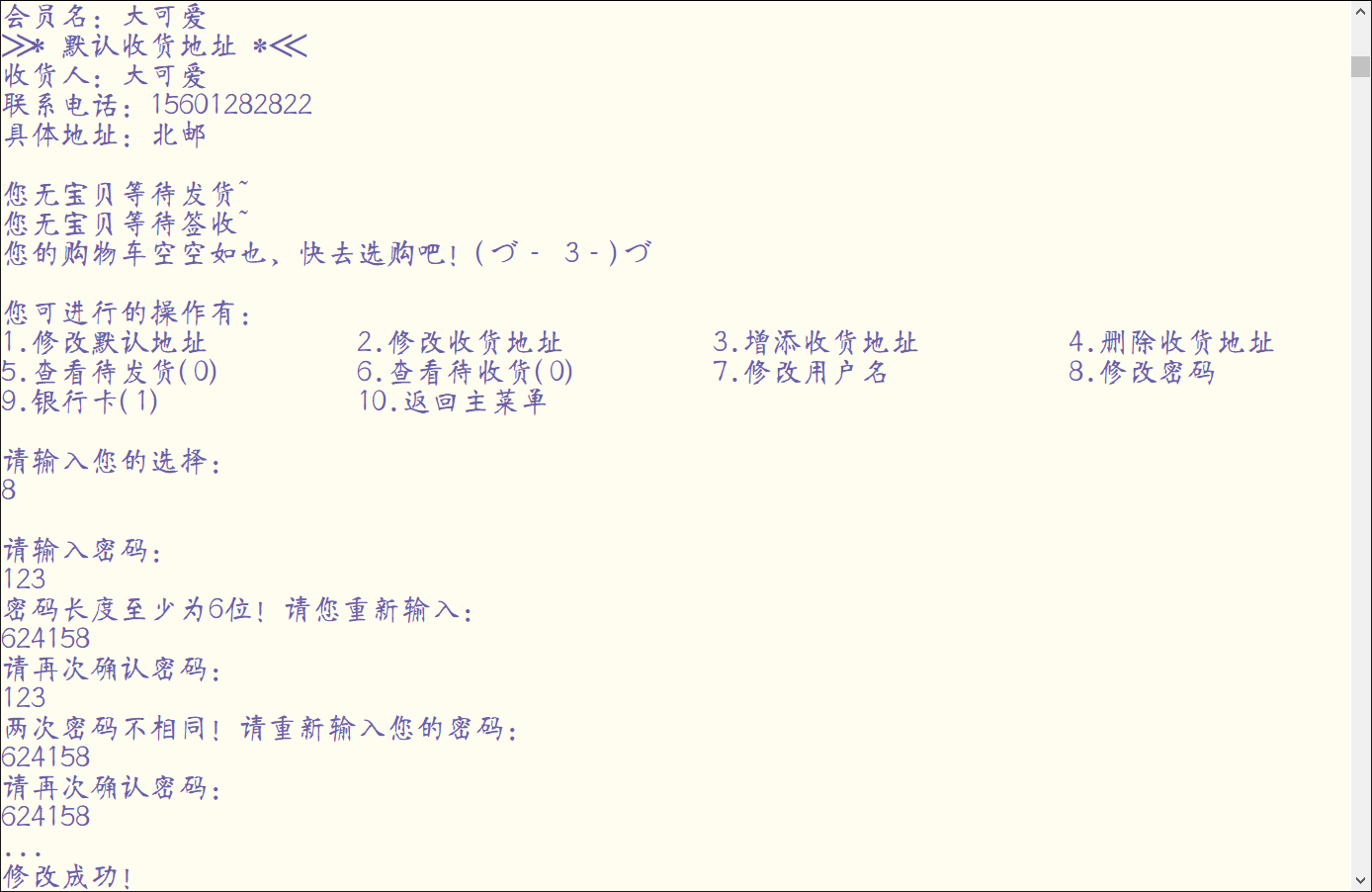




接下来，进行“修改用户名“”的演示：



“修改密码”的演示：



以上即为程序功能的大体演示。

其一，文件操作的改进。数据库非本学期课程，在专业选修JAVA和Python的压力下没有抽出时间进行相关的自学。C++对于文本文件的直接操作较C方便许多，但仍繁琐且效率不高。若搭载在数据库的框架下，效率会有显著提升，代码量也会相应降低，需改进；

其二，用户界面的美化。银行门户的设计用时不多，功能增添较少。本次大作业重心在电商平台的设计，基于实际和人性化的考虑，加入较多额外功能。

其三，解挂失的处理。