**《程序设计基础》Project报告**

**（CST11103）**

|  |  |  |  |
| --- | --- | --- | --- |
| **姓 名** | **曾钟琪** | **学 号** | **20241407** |
| **年级、专业、班级** | **2024级电气与信息类45班** | | |
| **上课时间** | **2024-2025学年第1学期** | | |
| **Project成绩** | **代码成绩**  （70%） | **报告成绩**  （30%） | **综合成绩** |
|  |  |  |
| **评语** |  | | |
| **评价教师** |  | | |

**《程序设计基础》Project任务书**

|  |  |
| --- | --- |
| 名 称 | ATM模拟程序的设计与实现 |
| 类 型 | □验证性 □设计性 ■综合性 |
| 内 容 | 运用面向对象程序设计思想，基于命令行界面设计并实现一个ATM模拟程序，模拟常见的ATM功能。 |
| 要 求 | ATM模拟程序能够完成ATM的主要功能，包括：  ①显示欢迎词及提示信息；②用户插卡，ATM验证用户账号及密码有效性，输入错误3次即被锁卡；③余额查询：初始余额为10000元；④取款功能：每次取款余额为100的整数倍，有单笔和单日金额限制；⑤转账功能：可将本账户中的存款转入其它账户，转入账户账号需两次输入确认；⑥修改密码：密码为6位数字，新密码需要两次输入确认；⑦退卡。  设计实现命令行界面，界面应友好、方便操作。参考界面如图1所示。程序所涉及到的用户资料、银行帐户、存取款记录等信息保存在数据文件中。其中银行账户的格式如下：  账号 19位数字  姓名  身份证 18位字符  密码 6位数字  余额    图1 ATM模拟程序参考界面 |
| 任务时间 | 2024年12月9日至2024年12月31日 |

**代码评分标准（满分10分）**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **学习态度** | **程序功能** | **程序Bug** | **程序界面** | **异常处理** | **得分** |
| 抄袭、被抄袭 | -- | -- | -- | -- | **0** |
| 自己完成编码  无创新意识 | 没有完成实验要求的所有功能 | -- | -- | -- | **5** |
| 自己完成编码  有一定的创新意识 | 基本完成实验要求的所有功能 | 很多 | 一般 | 无 | **6** |
| 自己完成编码  有一定的创新意识 | 完成实验要求的所有功能 | 较多 | 一般 | 无 | **7** |
| 自己完成编码  有较强的创新意识 | 完成实验要求的所有功能，并有所扩展 | 较少 | 良好 | 有 | **8** |
| 自己完成编码  有较强的创新意识 | 完成实验要求的所有功能，并有所扩展 | 极少 | 友好 | 有 | **10** |

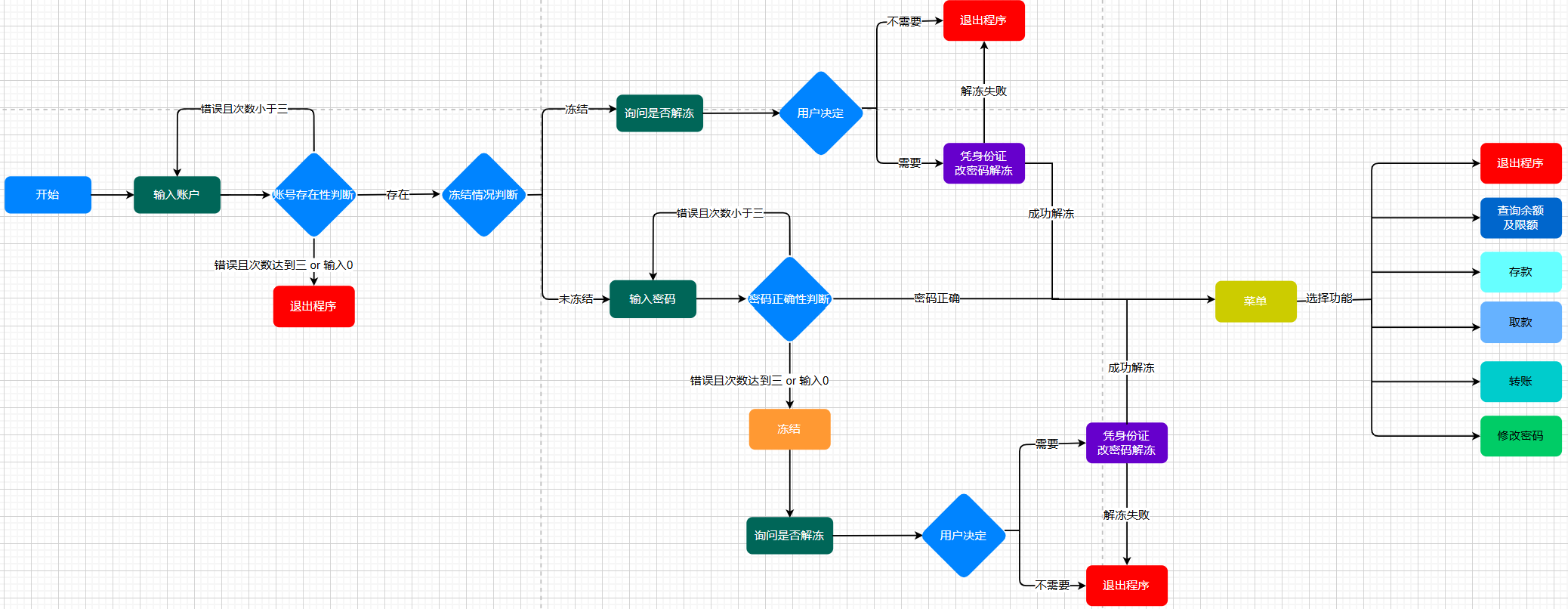
**报告评分标准（满分10分）**

|  |  |  |  |
| --- | --- | --- | --- |
| **文档规范性** | **文档条理性** | **表述准确性** | **得分** |
| 不规范，需求分析、程序设计、程序实现部分不完整。 | 描述缺少逻辑性，没有条理。 | 文字描述不准确。 | **3** |
| 较规范，有需求分析、程序设计、程序实现部分，但实质性描述较少。 | 描述具有一定的逻辑性，条理较清晰。 | 文字描述较准确，但缺少必要的类图、程序流程图等图表说明。 | **5** |
| 规范，有需求分析、程序设计、程序实现部分，描述较充分。 | 描述具有较好的逻辑性，条理清晰 | 文字描述较准确，所描述的类图、程序流程图基本准确、无大错。 | **7** |
| 规范，有需求分析、程序设计、程序实现部分，描述充分。 | 描述具有很好的逻辑性，条理清晰 | 文字描述准确，所描述的类图、程序流程图准确。 | **10** |

# 需求分析

## 1.1 流程图

本项目利用C++语言，运用面向对象程序设计思想，基于命令行界面设计并实现一个ATM模拟程序，模拟常见的ATM功能。下图为程序主要流程图。



## 1.2 主要流程分析

### 1.2.1 登录部分

用户登录账户时，需输入十九位数字的卡号。若输入卡号有误（即输入了不存在的卡号或者输入长度并非十九位），则程序提示“卡号不存在”，用户需再次输入。输入错误三次后程序会自动退出。

若卡号输入形式正确且检测该卡未被冻结，则需输入密码。若输入密码有误（即密码与卡号不匹配或是输入长度并非六位），则程序提示“密码错误”，用户需重新输入。输入错误三次则程序提示“卡已被冻结”，并提示用户进行解冻操作，用户若选择解冻则需输入与该卡绑定的身份证进行解冻并且重设密码。重设密码后用户可直接进入服务菜单。

若卡号输入形式正确且检测该卡被冻结，提示用户进行解冻操作，用户若选择解冻则需输入与该卡绑定的身份证进行解冻并且重设密码。重设密码后用户可直接进入服务菜单。

若账户的卡号及密码皆输入正确且该卡未被冻结，则进入服务菜单。

### 1.2.2 功能部分

服务菜单提供了六种服务：查询余额及限额、存款、取款、转账、修改密码。

若用户选择“查询余额及限额”，则可查看到该卡当前的余额及该卡当日剩余限额。

若用户选择“存款”，则可以输入100的整数倍数字的金额存入账户。若输入数字有误（即不是100的整数倍），则程序提示“输入的数字应该是100的整数倍”，即用户需再次输入正确的数字。若数字大于一百万，则程序提示“数额过大，请前往柜台操作。”若数字输入正确，则存款成功，返回服务菜单。

若用户选择“取款”，则可以输入100的整数倍数字的金额取出账户（单笔默认限额2000，当日默认限额5000）。若输入数字有误（即不是100的整数倍），则程序提示“输入的数字应该是100的整数倍”，即用户需再次输入正确的数字。若数字大于卡内余额或当日剩余限额或单笔限额，则程序提示“余额不足”或“当日限额不足”或“超出单笔限额”，用户需重新输入。若数字大于一百万，则程序提示“数额过大，请前往柜台操作。”若数字输入正确，则取款成功，返回服务菜单。

若用户选择“转账”，则可进行转账服务。首先需输入转入的卡号。若输入卡号有误（即输入了不存在的卡号或者输入长度并非十九位），则程序提示“卡号不存在”，用户需重新输入。若转入账户的卡号输入正确，则需用户再次输入卡号进行确认。若两次卡号输入不一致，则程序提示“输入卡号不一致”，用户需重新输入。若两次账号输入一致，则可进行转账操作。用户可输入转账金额，当输入数字大于用户的卡余额时则程序提示“余额不足”，用户需重新输入若用户输入数字正确则将用户的卡的余额减去转账金额，将目标卡的余额加上转账金额，程序提示“转账成功”并返回服务菜单。

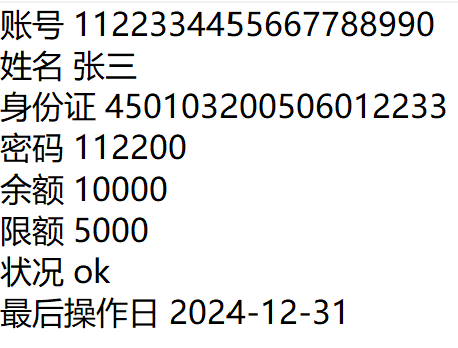
若用户选择“修改密码”，则可对账户的密码进行修改。用户需输入新的六位数字密码，如果输入的并非六位数字密码则程序提示“输入的不是六位数字密码”，用户需重新输入。若密码合法则用户需再次输入密码确认。如果两次新密码输入不一致，则程序提示“密码不一致”，用户需重新输入。若两次新密码输入一致，则密码修改成功，返回服务菜单。

若用户程序“退出程序”，则程序提示“欢迎下次使用”，退出程序。

# 系统设计

## 2.0 数据储存规范样例

数据应当保存在./data/目录下的{卡号}.txt中。由于课设要求故所有示例账户中的初始余额均为10000。



## 登录部分

### 卡号输入部分

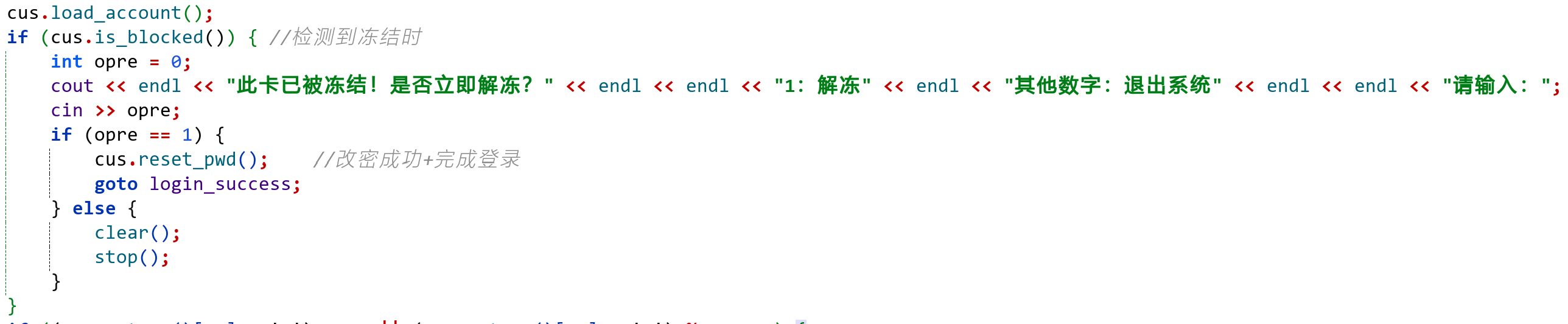
这里检测卡号是否存在的办法是检测该卡号有没有对应的数据文件./data/{卡号}.txt存在。



### 卡号加载及检测部分[5]

这里的数据是从卡号对应的数据文件中./data/{卡号}.txt加载的。当中根据课设要求数据文件中要有卡号一项，故此处仍然加载卡号。理论上加载的卡号应该与文件名中的卡号一致。在后续提到的保存函数中保存的仍是用户输入的正确卡号。当加载卡号与输入的卡号不一致时则认为异常，将提示用户前往柜台操作并终止程序。





### 密码输入部分

输错三次密码则冻结该卡并询问用户是否解冻。



### 2.1.4 解冻部分

此处考虑到ATM机只有数字键盘故解冻要求输入身份证时用“00”代替“X”（身份证最后一位可能为X）。**[7]**

这里调用了**2.2.5 修改密码部分**的函数。

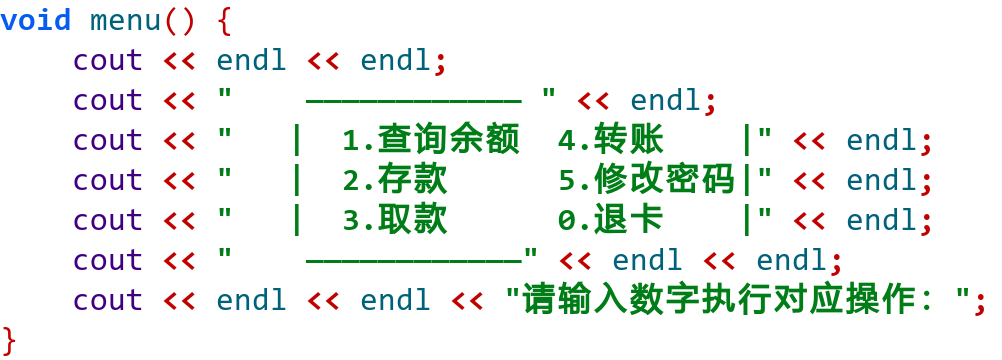


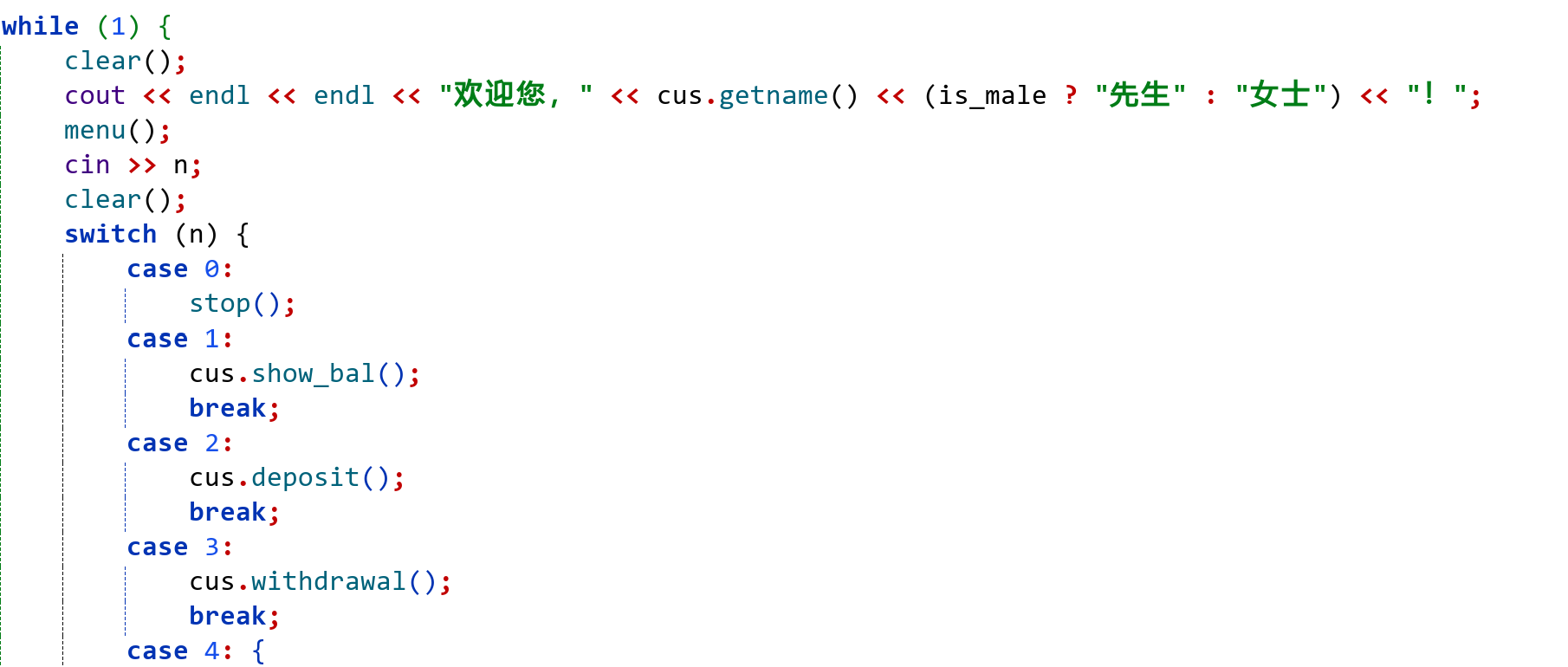
## 2.2功能部分

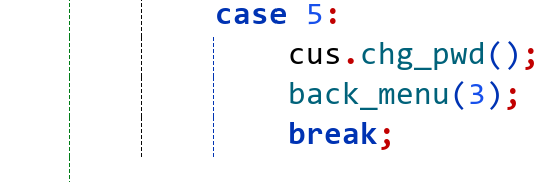
### 2.2.0 菜单及功能选择部分

功能选择主要采用了switch选择语句。

以下图中由于转账部分并非调用成员函数且代码过长故暂时未在**2.2.0**下展出，而是在**2.2.4 转账部分**展出。

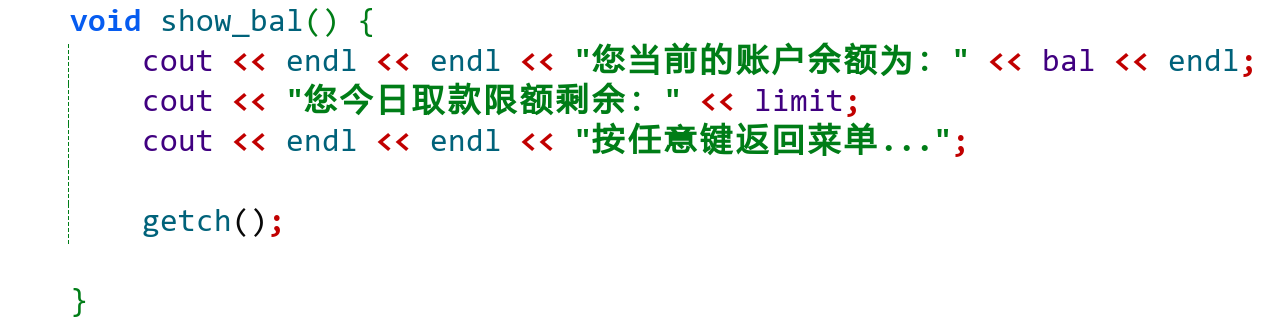






### 2.2.1 查询余额及限额部分[1]

此处采用的方法是getch()而非system("pause")主要考虑为该程序应该兼容多种系统运行，而类Unix系统，如Linux系统及其发行版等大都都没有pause命令，故采用更为包容的getch()。



### 2.2.2 存款部分

这里认为ATM没有负号键故此处不考虑负数情况。



### 2.2.3 取款部分

这里认为ATM没有负号键故此处不考虑负数情况。用户取款后该卡当日限额会随之减少。



### 2.2.4 转账部分

这里需要用户输入两次目标卡号以校对并输入金额完成转账。



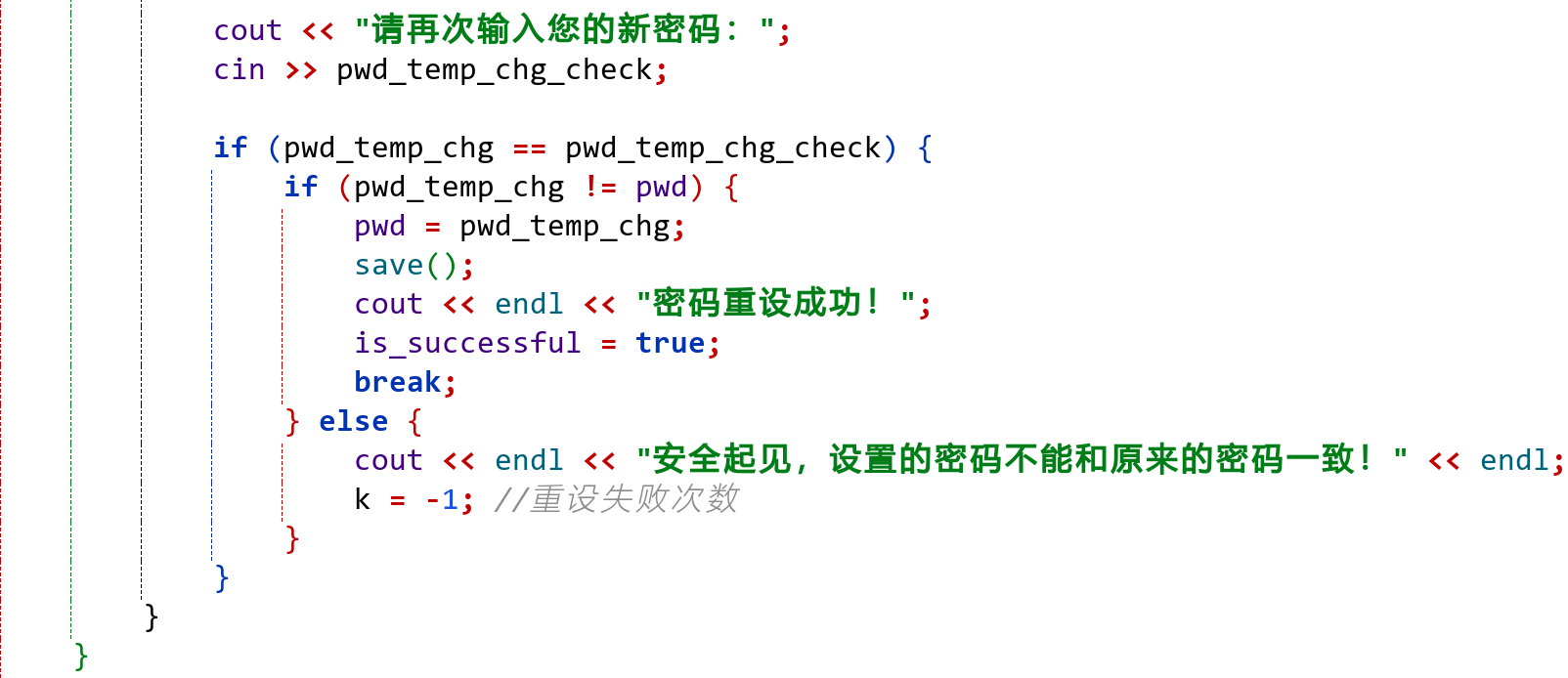




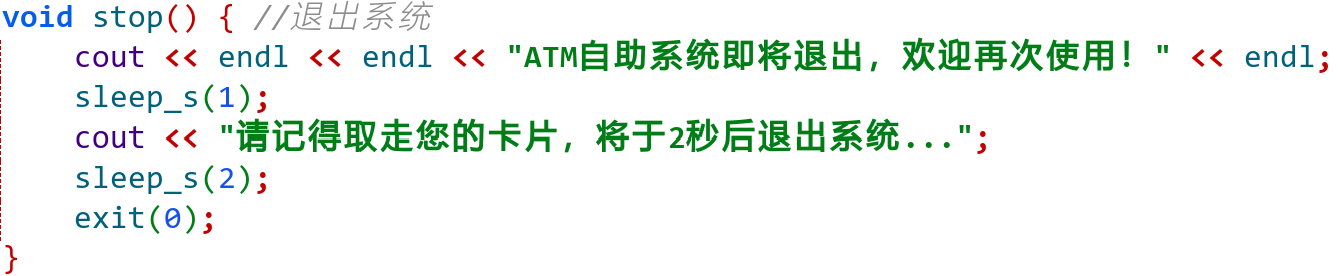
### 2.2.5 修改密码部分

这里根据一般ATM操作逻辑加入了密码不可与原密码重复的设定。





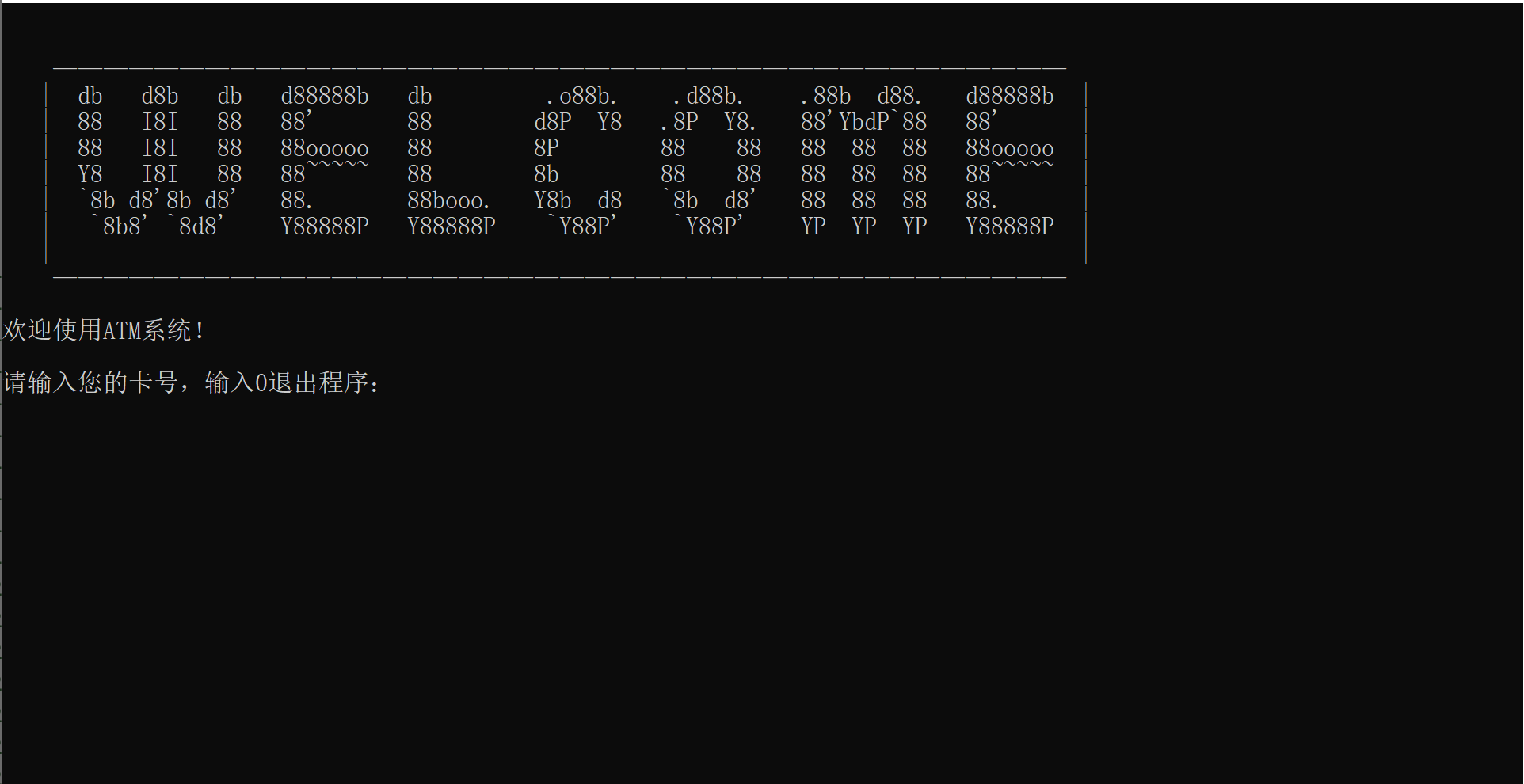
### 2.2.6 退出程序



# 程序实现

## 登录部分

开始时提示用户输入卡号。

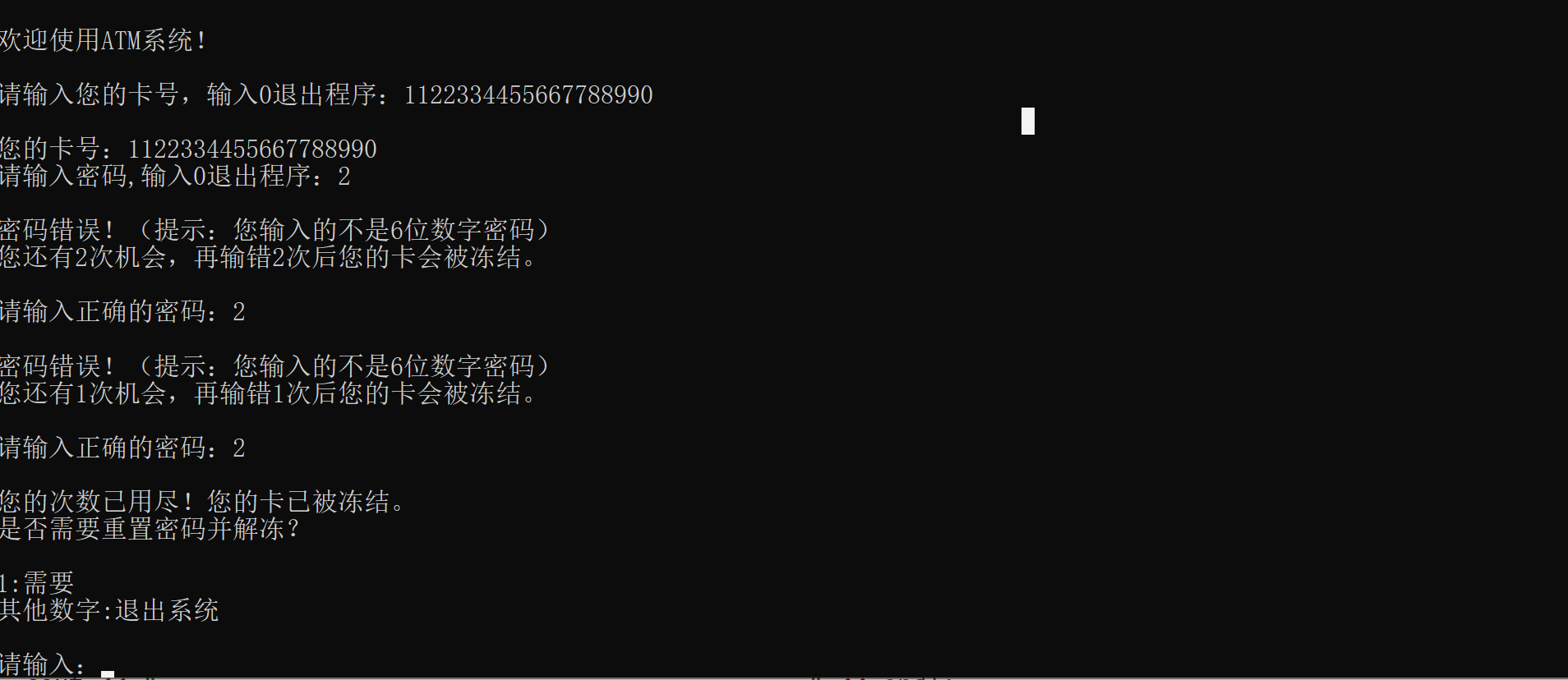


输入正确卡号后，若未冻结提示输入密码。



输入正确密码后即可进入功能菜单。

若被冻结或输入密码错误超过三次则提示解冻。





需输入身份证及重新设置密码密码以解冻。



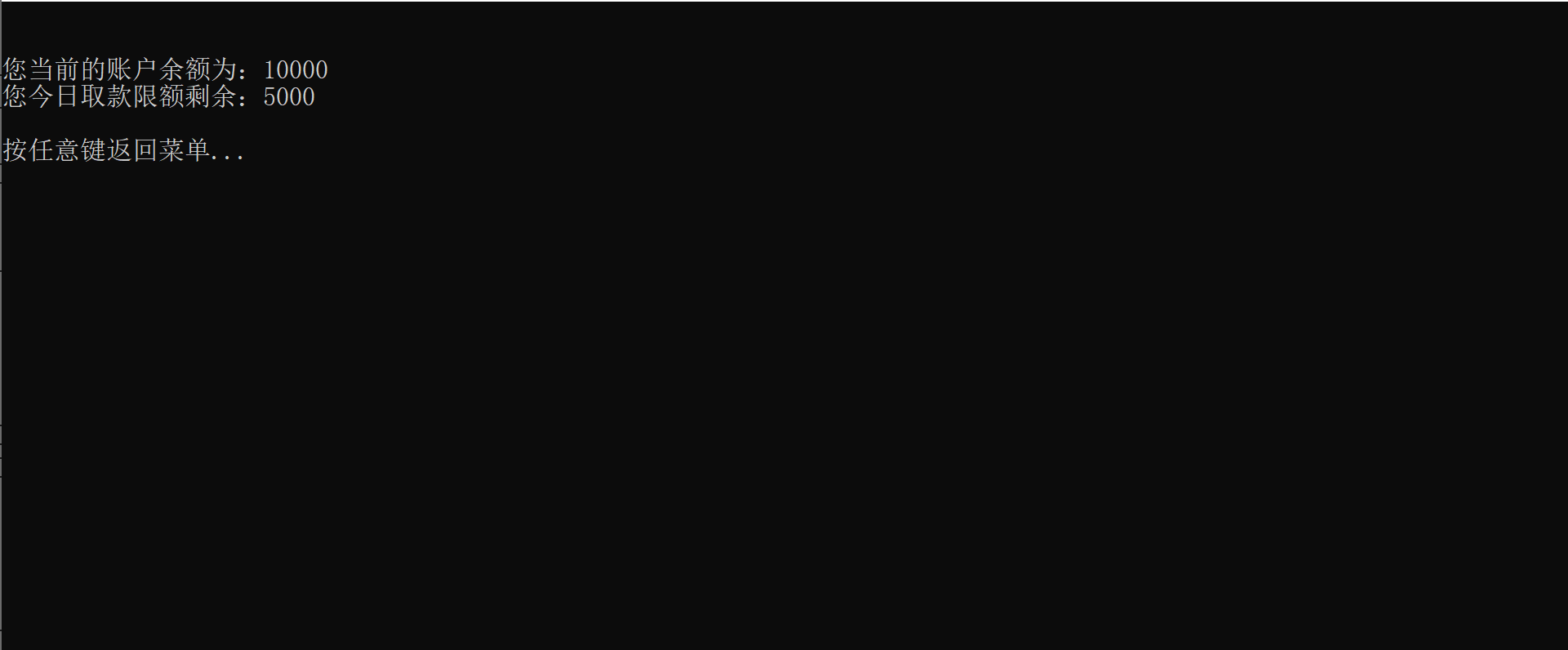
成功后即可进入功能菜单。

## 功能部分

### 3.2.0 菜单

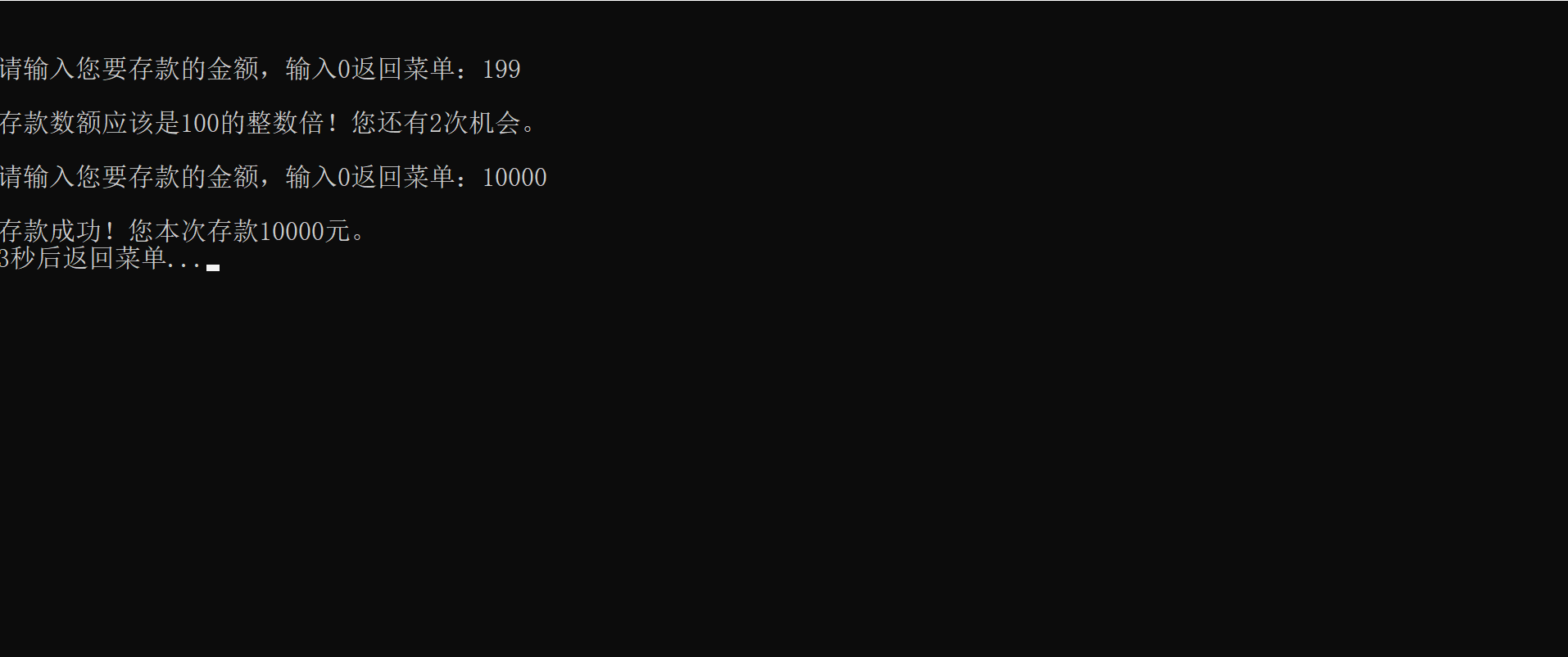


### 查询余额



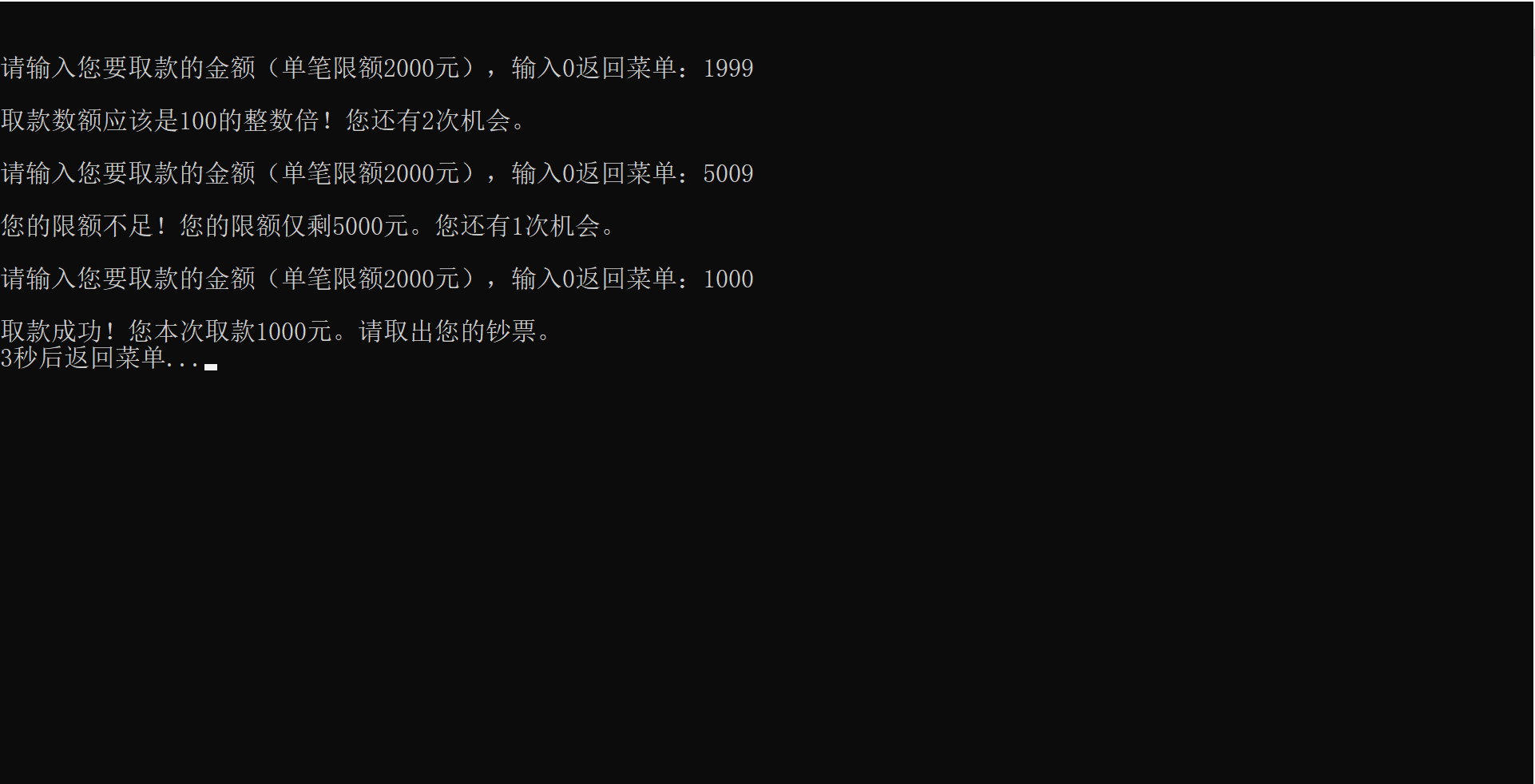
### 存款

这里要求输入一个100的整数倍的数字。



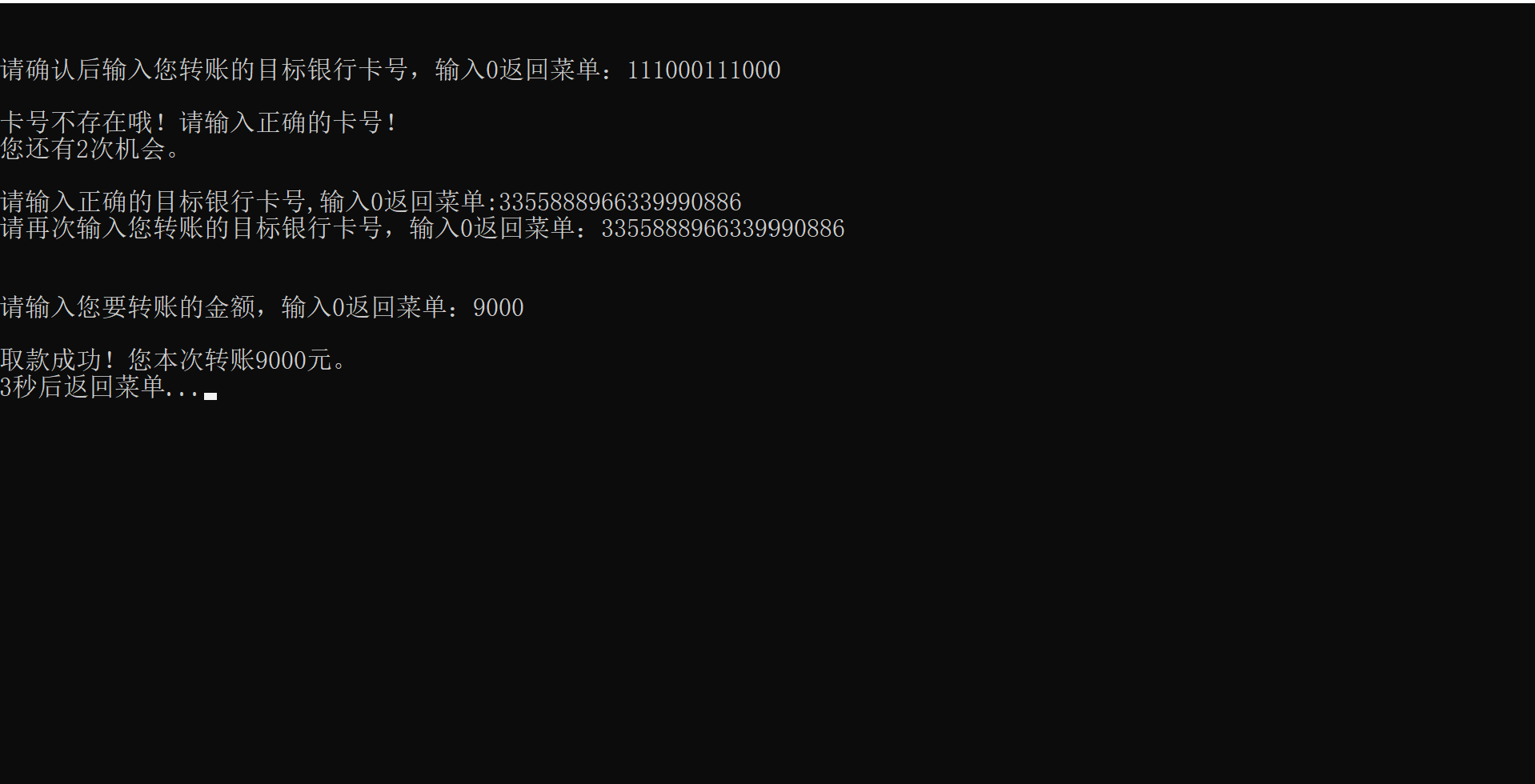
### 取款

这里要求输入一个100的整数倍的数字且小于2000元的单笔限额且小于当日剩余限额且小于余额。



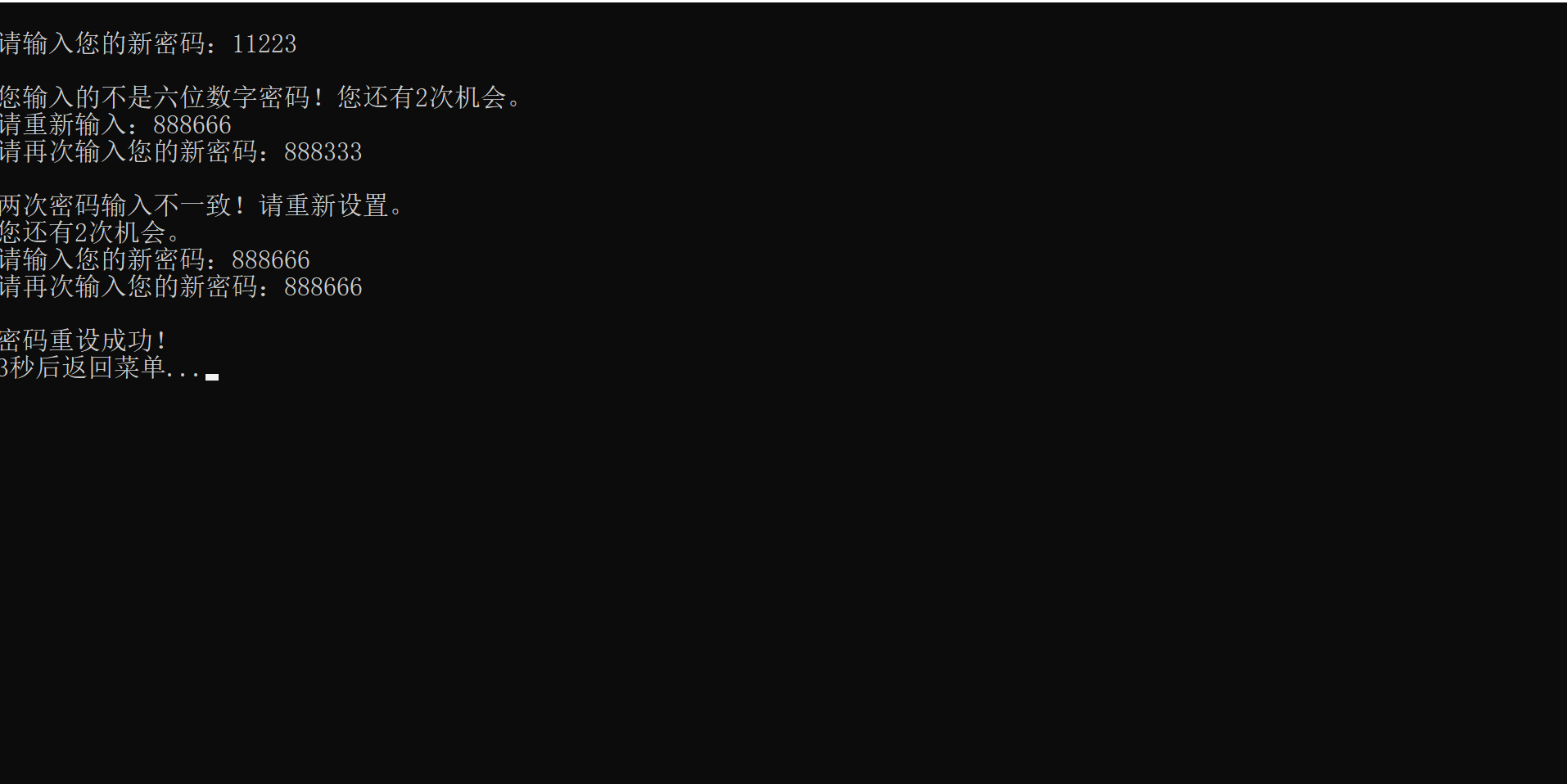
### 转账

这里要求两次输入目标卡号以确认，并输入转账金额。

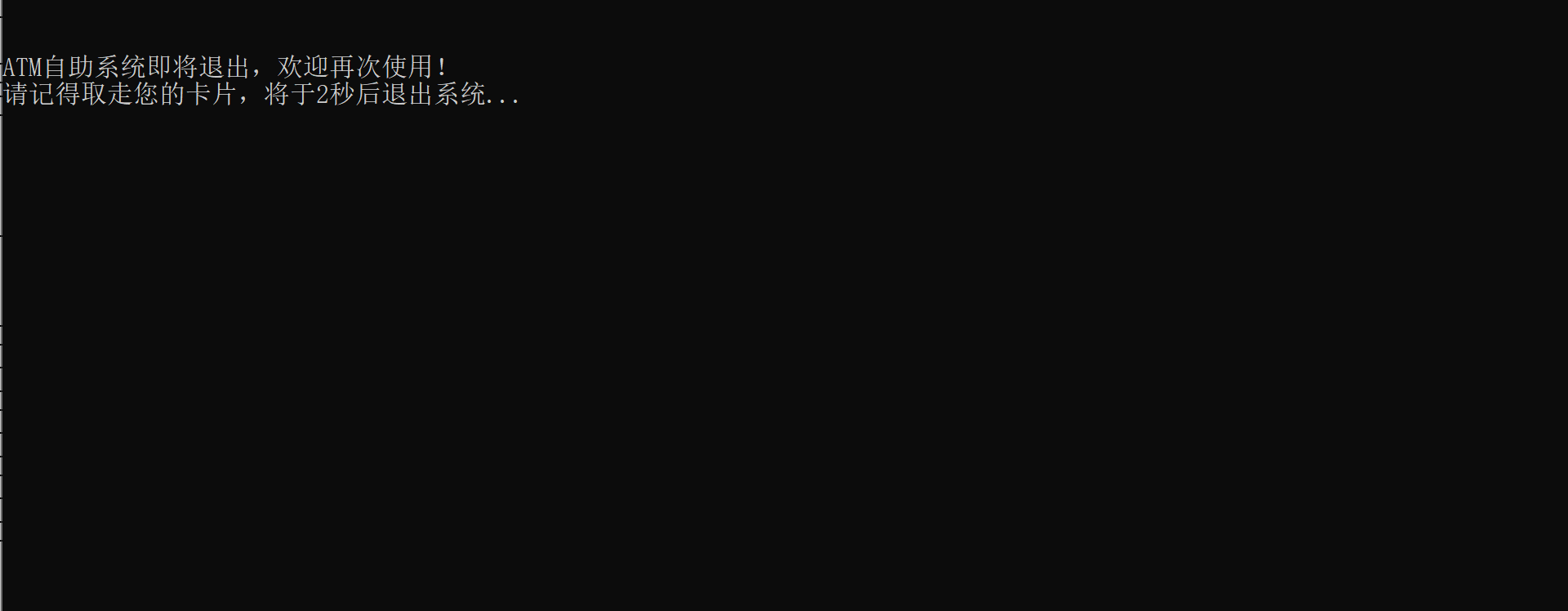


### 修改密码

这里要求两次输入新密码以确认。



### 退出程序



# 4. 总结

## 4.1 项目亮点

该项目**对不同系统做出了兼容**，如clear()函数在Windows系统以及类Unix系统都有对应的实现方式**[2][3]**；程序中等待没有采用常规的在Windows系统上的Sleep(int i)函数而是采用了对多系统更包容的this\_thread::sleep\_for(chrono::seconds(i))函数并重新定义成sleep\_s(int i)函数**[4]**；“查询余额及限额”中的“按任意键返回菜单”用的方法是getch()而不是常规Windows系统中用的system("pause")函数。该项目可以更好的兼容大多数系统。

该项目考虑客户需求及感受，设计了更好的用户菜单以及在其他方面做出了更好的处理。如考虑到解冻时用户的身份证可能含有X而设计了使用“00”取代“X”的方案。又如优化过的提示语的内容、换行及空行等能更好且更快地让用户理解ATM的操作方式。

该项目亦在细节上进行了处理，如几乎每个步骤都有的**最大试错次数规定以及试错次数超出后的处理**，还有几乎所有功能都有的输入0返回菜单的设定、几乎每个登录步骤都有的输入0退出程序的设定，以及欢迎语中根据身份证倒数第二位判断性别而称“先生”或“女士”。

该项目具有**相应的异常处理**，如数据文件存在但无法打开时的异常提示并终止程序，以及数据文件内卡号与文件名包含卡号不一样时的异常提示并终止程序。

## 4.2 项目创新点

## 该项目中加入了**冻结时可以重置密码并解冻**的功能，给原本课设要求的数据文件中相对冗余的身份证给予了作用，并加入了一个实用的新功能。

该项目加入了**自动重置限额**的操作，当今日日期与从文件中读取的最后操作日期不一致时，限额自动重置为5000并将最后操作日期改为今日。**[6]**

该项目采用了**从外部文件读取数据并将数据保存在外部文件**的操作，并将文件名称定为{卡号}.txt，便于统一管理，查找及修改，删除用户数据。

## 4.3 不足之处

该项目代码可能存在冗余并且各个关于试错的循环操作的写法并不完全一致，且可能提示语过多导致运行时占用内存可能相对较大。后续优化可以进一步地查找不必要的代码并修改、删除，统一循环操作的写法，以及编写提示语函数来实现统一提示语规范及减少代码中不断出现的提示语。

## 4.4 未来改进方向

目前的项目仍然是在命令行当中运行，未来可以改进界面化操作。

目前的项目仍然无法像现实中真正的ATM一样联网并从互联网中的数据库读取数据，未来可以尝试将程序联网，以及将数据保存在数据库而非当前模拟的文件当中。

目前的项目对用户数据没有加密措施，未来可尝试引入加密方法加密保存用户数据。

# 5. 参考文献

[1]C++中如何实现按任意键继续

<https://blog.csdn.net/rgsongzh/article/details/131627403>

[2]C++如何清除屏幕上的内容？

<https://blog.csdn.net/Code_easy/article/details/126678558>

[3]关于 linux c 清屏命令

<https://blog.csdn.net/silno/article/details/109678293>

[4]C++睡眠延时函数 Sleep() usleep()（windows.h、unistd.h）

<https://blog.csdn.net/Dontla/article/details/120304529>

[5]C++ 文件和流

<https://www.runoob.com/cplusplus/cpp-files-streams.html>

[6]C/C++ 获取当前时间(最全方法)

<https://blog.csdn.net/weixin_50964512/article/details/123309596>

[7]c++string截取字符串

<https://blog.csdn.net/scarificed/article/details/117229978>