

程序设计报告

（ 2019/2020 学年 第 一 学期）

题 目：**ATM仿真软件**

|  |  |
| --- | --- |
| **专 业** | **电子信息工程** |
| **组长 学号姓名** | **B18011834熊凌宇** |
| **组员 学号姓名** | **B18011827赵源** |
|  | **B18011830曹子建** |
| **指 导 教 师** | **俞琼** |
| **指 导 单 位** | **计算机学院、软件学院、网络空间安全学院** |
| **日 期** | **2019.9.9** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **成员分工** | | | | | | | | | |
| 组长（熊凌宇） | | 拆分模块、界面架构、用户菜单界面  用户登录函数、读写文件函数、代码注释  添加交易流水信息函数、获取时间函数 | | | | | | | |
| 组员（赵源） | | 存取款功能、转账功能  信息查询功能、修改密码功能  优化界面、代码注释 | | | | | | | |
| 组员（曹子健） | | 查找用户ID下标函数  优化界面、代码注释 | | | | | | | |
| **教师评价** | | | | | | | | | |
| 评价准则 | | | | 优秀 | 良好 | | 中等 | 合格 | 不合格 |
| 该课题组掌握程序设计开发的相关工程基础知识，并能够针对求解的工程问题，进行合理的分析与综合。 | | | |  |  | |  |  |  |
| 该课题组能够结合计算机软硬件资源，合理选用算法、数据结构、数据存储方式等技术手段，对求解的工程问题进行系统设计、建模及预测。 | | | |  |  | |  |  |  |
| 该课题组能够选择合适的程序设计语言与编程开发平台，对求解的工程问题进行系统模拟、仿真与预测。 | | | |  | √ | |  |  |  |
| 该课题组熟练掌握调试方法与工具，对程序开发过程中出现的问题进行分析、跟踪与调试，并在开发平台上对最终作品进行充分测试。 | | | |  |  | |  |  |  |
| 该课题组具备初步的开发能力，完成了课题要求的各项开发任务，功能实现全面。 | | | |  |  | |  |  |  |
| 该课题组具备初步的界面设计意识。用户使用界面布局合理，交互友好，操作简便。 | | | |  |  | |  |  |  |
| 该课题组正确、完整的回答了指导教师关于课题作品的提问，反映出其对课题内容，以及相关的工程基础知识具有较好的理解和掌握。 | | | |  |  | |  |  |  |
| 该课题组具备一定的表达能力与文字处理能力，能够熟练使用文字处理软件完成课题报告一份。报告完整记录了课题组的工作及总结，内容详实，格式规范。 | | | |  |  | |  |  |  |
| 该课题组具有较好的团队协作精神，能够努力沟通解决遇到的各种开发、工程问题。 | | | |  |  | |  |  |  |
| 该课题组具备一定自学能力与探索创新意识，能够充分利用各种网络与图书馆资源自学新知识与新技能。 | | | |  |  | |  |  |  |
| 该课题组在上机过程中遵守机房规章制度，出勤与平时表现规范。 | | | |  |  | |  |  |  |
| 最终作品和报告反映出该课题组具备较好的解决工程问题的能力。 | | | |  |  | |  |  |  |
| **本次程序设计能力达成评价** | | | | | | | | | |
| 学号姓名 | B18011834熊凌宇 | | B18011827赵源 | | | B18011830曹子健 | | | |
| 贡献度及个人表现 | 45% | | 35% | | | 20% | | | |
| 最终成绩 |  | |  | | |  | | | |
| **备注：**  贡献度及个人表现：好、较好、中等、较差、差  最终成绩：优秀、良好、中等、及格、不及格 | | | | | | | | | |
| **指导教师： 俞琼 2019 年 9 月 11 日** | | | | | | | | | |

**ATM仿真软件**

1. **课题内容和要求**

ATM 的管理系统是银行流程业务中，十分重要且必备的环节之一。在银行业务流程中有着承上启下、起承转合的作用。本课题可以帮助人们更加熟悉 ATM 的各类操作，从而方便生活中的使用。其基本功能如下：密码验证机制；吞锁卡机制；存取款功能；账户查询功能；转账功能等。

要求：

（1）系统内的相关信息文件由程序设计人员预先从键盘上录入，文件中的数据记录不 得少于 20 条；

（2）设计并实现系统的相关界面，使用分级菜单提高人机交互性；

（3）登录系统必须输入正确的卡号和密码，输入错误时提示用户，连续输入错误 3 次则显示锁卡，用户登录后可以修改自己的密码；

（4）取款功能：输入取款金额，核对。若超出账户余额，则输出“余额不足”则输出对应金额的现金，并提示用户是否打印回单，退卡，返回界面；

（5）存款功能：存入现金，核对金额，存入账户，提示用户是否打印回单，返回主界面；

（6）查询账户信息，包括用户名、余额信息，以及各种操作的历史记录；

（7）转账模块，提示用户输入正确的账号，检查转账条件是否满足，并完成转账操作；

（8）退出终端系统，提示用户保存好银行卡。

**二、需求分析**

ATM仿真软件系统的功能框架图如图1所示。

用户登录

存款功能

返回上一级

修改密码

用户信息查询

转账功能

取款功能

查询余额

查询用户信息

图1 功能框架图

（1）提供用户登录一级菜单

（2）登录后显示二级菜单，包含存取款功能、转账功能、用户信息查询功能、修改密码功能。

（3）用户信息查询菜单包含余额查询、用户交易流水记录

**三、概要设计**

1 主要存储结构

struct Account

{

char ID[20]; //账户ID

char Name[20]; //用户姓名

char Password[7]; //密码

double money; //账户余额

int judge; //判断账户是否锁定

};

typedef struct Account Account;

struct Statement //交易流水结构体

{

char accountID[20]; //账户ID

char time[30]; //交易时间

char type[10]; //交易类型

double money; //交易金额

char toAccount[20]; //转账对方账户如果是存取款则为NULL

}；

typedef struct Statement Statement;

2 主要函数流程

（1）查找用户ID下标函数：利用用户输入的ID进行顺序查找，如存在则返回ID所在数组元素下标，否则返回值为-1。查找用户ID下标函数流程图如图2所示。

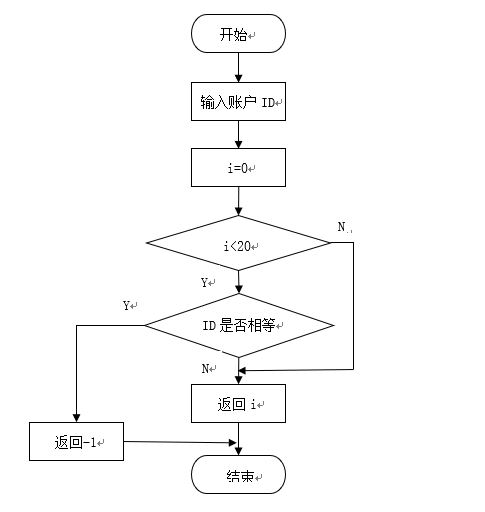


图2 查找用户ID下标函数

（2）用户登录密码验证函数：采用隐藏密码方式输入密码，以及添加了密码加密算法，能够实现输入密码退格功能。流程图如图3所示

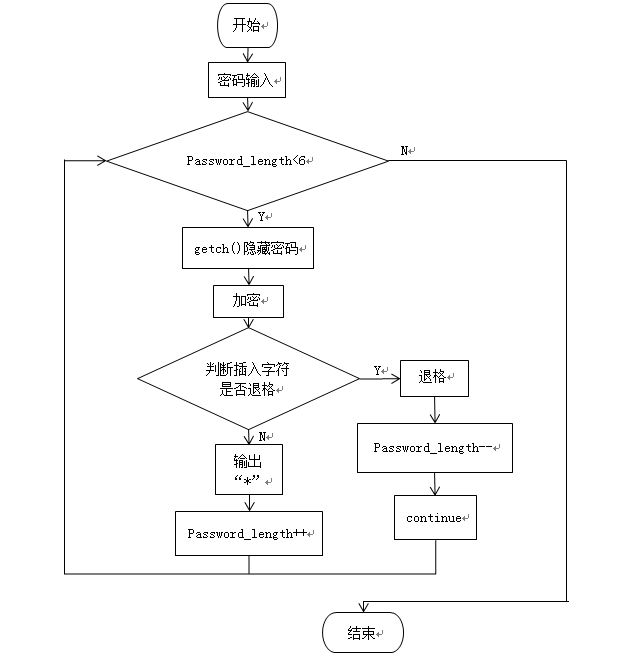


图3 用户登录密码验证函数

（3）装载函数：用于从文件中读取账户信息到数组中。函数流程图如图4所示。

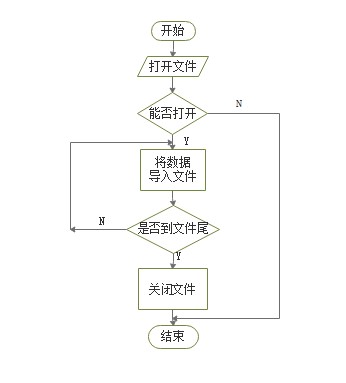


图4 装载函数

（4）更新账户信息函数：利用结构体数组作为中介，用户进行业务时先对数组内元素操作，然后将其操作后的数组覆盖到账户文件中。函数流程图如图5所示。

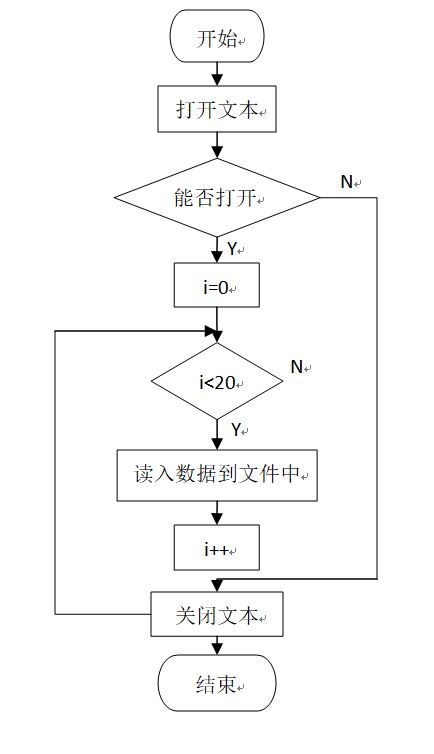


图5 更新账户信息函数

（5）修改密码函数：使用两次输入密码进行判断，如果相等则修改成功，否则修改失败。函数流程图如图6所示。

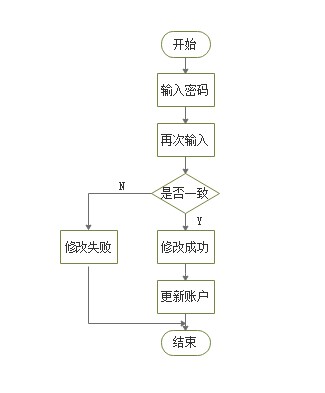


图6 修改密码函数

（6）转账函数： 用户输入金额，若输入金额大于账户余额则从新输入转账金额，对用户输入目标账户是否存在（并且不能给自己转账），存在则转账成功，否则重新输入，然后更新账户信息、添加交易记录。函数流程图如图7所示。

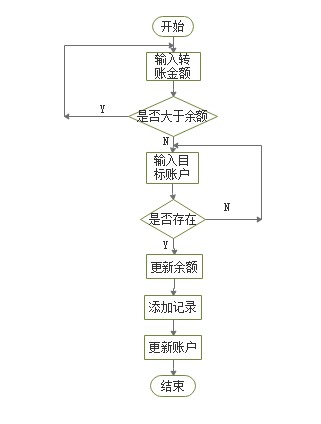


图7 转账函数

**四、源程序代码**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<conio.h> //getch()函数头文件

#include<memory.h>

#include<time.h> //memset函数头文件

#include<math.h>

#include<windows.h> //时间延时头文件

struct Account

{

char ID[20]; //账户ID

char Name[20]; //用户姓名

char Password[7]; //密码

double money; //账户余额

int judge; //判断账户是否锁定

};

typedef struct Account Account;

struct Statement //交易流水结构体

{

char accountID[20]; //账户ID

char time[30]; //交易时间

char type[10]; //交易类型

double money; //交易金额

char toAccount[20]; //转账对方账户（如果是存取款，则为NULL）

};

typedef struct Statement Statement;

//函数声明

void runMain(Account str[],Statement sta[],int choice);

int accountLogin(Account str[]);

int find(Account str[],Account s1);

void menuAccount(Account \*str,Statement \*sta,int account\_Number);

void MoneyChange(Account \*pa,Statement \*ps,int choice,int account\_Number);

void menuSearch(Account \*str,Statement \*sta,int account\_Number);

void searchBalance(Account \*acc,int account\_Number);

void searchManage(Account \*acc,Statement \*sta,int account\_Number);

void transferMoney(Account \*pa,Statement \*ps,int account\_Number);

void ChangePassword(Account \*str,Statement \*sta,int account\_Number);

char\* getDate();

void readFile(Account str[]);

void readFile\_Transactions(Statement sta[]);

void writeFile(Account str[]);

void add\_Account\_Transactions(Statement account\_Message);

void menu()

{

printf("\n\n\n\n\n\n\n\n\n");

printf("\t\t\t\t\t##########################################\n");

printf("\t\t\t\t\t# #\n");

printf("\t\t\t\t\t# #\n");

printf("\t\t\t\t\t# >>>>>>>> 1.Sign in <<<<<<<<<< #\n");

printf("\t\t\t\t\t# >>>>>>>> 0.Quit <<<<<<<<<< #\n");

printf("\t\t\t\t\t# #\n");

printf("\t\t\t\t\t# #\n");

printf("\t\t\t\t\t##########################################\n");

}

//运行主函数

void runMain(Account str[],Statement sta[],int choice) //str为账户信息数组指针，sta为交易流水数组指针，choice为用户业务选择

{

int account\_Number; //存储登录用户账户下标

switch(choice)

{

case 1:account\_Number=accountLogin(str); //账户登录，返回值为用户账户下标

menuAccount(str,sta,account\_Number); //二级菜单

break;

case 0:break;

}

}

//用户登录函数

int accountLogin(Account str[]) //形参为账户数组

{

Account s1; //存储用户登录信息

int password\_Length=0; //判断密码长度

int password\_Number=3; //密码次数（三次将被锁定）

int account\_Number; //账户ID下标

printf("\n\n\n\n\n\n\n\n\n");

printf("\t\t\t\t\tplease input Account\_ID:");

scanf("%s",&s1.ID);

while(find(str,s1)<0) //判断ID是否存在

{

printf("\t\t\t\t\tID input error\n");

printf("\t\t\t\t\tplease input Account\_ID again:");

scanf("%s",&s1.ID);

}

account\_Number=find(str,s1); //查询用户ID数组下标，将其放置在account\_Number中

if(str[account\_Number].judge==1) //判断用户是否被锁定

{

printf("\t\t\t\t\tThe account is locked,please contact with Administrator\n");

exit(1);

}

printf("\t\t\t\t\tplease input passwords:");

while(1)

{

while(password\_Length<6)

{

s1.Password[password\_Length]=getch();

s1.Password[password\_Length]+=51; //加密算法，使文件密码被隐藏

if((s1.Password[password\_Length])=='\b')

{

printf("\b \b");

password\_Length--;

}

printf("\*");

password\_Length++;

}

getch();

s1.Password[6]='\0'; //字符串结束标志位，使s1.Password成为字符串，防止出现乱码

printf("\n");

if(strcmp(str[account\_Number].Password,s1.Password))

{

printf("\t\t\t\t\tpassword input error,the left times you have: %d\n",--password\_Number);

if(password\_Number==0) //错误次数达到三次，用户被锁定

{

printf("\t\t\t\t\tpassword input is equal to three times,the account is locked,please contact with Administrator\n");

str[account\_Number].judge=1;

writeFile(str);

exit(1);

}

printf("\t\t\t\t\tplease input passwords:");

password\_Length=0;

}

else

break;

}

printf("\t\t\t\t\tLogin success! The page will jump in two seconds.\n");

Sleep(2000); //延时2秒

system("cls"); //清屏

return account\_Number; //返回账户在数组中下标

}

//信息查询模块函数

void menuSearch(Account \*str,Statement \*sta,int account\_Number) //str为账户数组指针，sta为交易流水指针，account\_Number为账户在数组下标

{

int choice\_number; //进行业务选择

printf("\n\n\n\n\n\n\n\n\n");

printf("\t\t\t\t\t##########################################\n");

printf("\t\t\t\t\t# #\n");

printf("\t\t\t\t\t# 1.Search Blance #\n");

printf("\t\t\t\t\t# 2.Search Transaction #\n");

printf("\t\t\t\t\t# 3.Return #\n");

printf("\t\t\t\t\t# #\n");

printf("\t\t\t\t\t##########################################\n");

printf("\t\t\t\t\tplease input your choice:");

scanf("%d",&choice\_number);

system("cls"); //清屏

switch(choice\_number)

{

case 1:

searchBalance(str,account\_Number); //查询余额

menuSearch(str,sta,account\_Number); //返回菜单

break;

case 2:

searchManage(str,sta,account\_Number); //查询交易记录

menuSearch(str,sta,account\_Number); //返回菜单

break;

case 3:

menuAccount(str,sta,account\_Number); //返回二级菜单

break;

}

}

//查询余额函数

void searchBalance(Account \*acc,int account\_Number) //acc为账户数组指针，account\_Number为账户数组下标

{

printf("\n\n\n\n\n\n\n\n\n");

printf("\t\t\t\t\tNow your account balance is：%.2f\n",acc[account\_Number].money);

printf("\t\t\t\t\tClick enter to return."); //按下回车键返回上一级

getch();

system("cls");

}

//二级菜单，用户办理业务菜单

void menuAccount(Account \*str,Statement \*sta,int account\_Number) //str为账户指针，sta为交易流水指针，account\_Number为账户数组下标

{

int choice\_number; //进行业务选择

printf("\n\n\n\n\n\n\n\n\n");

printf("\t\t\t\t\t##########################################\n");

printf("\t\t\t\t\t# #\n");

printf("\t\t\t\t\t# 1.Deposit 4.Transfer #\n");

printf("\t\t\t\t\t# 2.withdrawal 5.PasswordChange #\n");

printf("\t\t\t\t\t# 3.UserInformation 6.Return #\n");

printf("\t\t\t\t\t# #\n");

printf("\t\t\t\t\t##########################################\n");

printf("\t\t\t\t\tplease input your choice:");

scanf("%d",&choice\_number);

system("cls"); //清屏

switch(choice\_number)

{

case 1:

MoneyChange(str,sta,choice\_number,account\_Number); //调用存取款函数

break;

case 2:

MoneyChange(str,sta,choice\_number,account\_Number); //调用存取款函数

break;

case 3:

menuSearch(str,sta,account\_Number); //用户信息查询函数

break;

case 4:

transferMoney(str,sta,account\_Number); //转账函数

break;

case 5:

ChangePassword(str,sta,account\_Number); //修改密码函数

break;

case 6:

break;

}

}

//查询账户在数组的下标函数以及判断账户是否存在

int find(Account str[],Account s1)

{

int i=0;

for(i=0;i<20;i++)

{

if(strcmp(str[i].ID,s1.ID)==0)

return i; //返回账户下标

}

return -1;//返回-1表示账户不存在

}

//读取账户文件信息函数

void readFile(Account str[])

{

int i=0;

FILE \*fp;

fp=fopen("c language.txt","r");

if(fp==0)

{

printf("source file error");

exit(1);

}

fscanf(fp,"%s%s%s%lf%d",&str[i].ID,&str[i].Name,&str[i].Password,&str[i].money,&str[i].judge); //读入账户数据到str中

while(!feof(fp)) //判断是否到文件末尾

{

i++;

fscanf(fp,"%s%s%s%lf%d",&str[i].ID,&str[i].Name,&str[i].Password,&str[i].money,&str[i].judge);

}

fclose(fp);

}

//读取账户流水信息函数

void readFile\_Transactions(Statement sta[])

{

int i=0;

FILE \*fp;

fp=fopen("account Transaction.txt","r");

if(fp==0)

{

printf("source file error");

exit(1);

}

fscanf(fp,"%s%s%s%lf%s",&sta[i].accountID,&sta[i].time,&sta[i].type,&sta[i].money,&sta[i].toAccount); //读入交易流水数据到sta中

while(!feof(fp)) //判断是否到文件末尾

{

i++;

fscanf(fp,"%s%s%s%lf%s",&sta[i].accountID,&sta[i].time,&sta[i].type,&sta[i].money,&sta[i].toAccount);

}

fclose(fp);

}

//存取款后更新账户金额信息

void writeFile(Account str[])

{

int i;

FILE \*fp;

fp=fopen("c language.txt","w");

if(fp==0)

{

printf("creat new file error\n");

exit(1);

}

for(i=0;i<20;i++)

{

fprintf(fp,"%-12s%-8s%-8s%-12.2f%-4d\n",str[i].ID,str[i].Name,str[i].Password,str[i].money,str[i].judge);

}

fclose(fp);

}

//添加交易记录函数，形参为用户信息结构体

void add\_Account\_Transactions(Statement account\_Message)

{

FILE \*fp;

fp=fopen("account Transaction.txt","a+");

if(fp==0)

{

printf("file open error");

exit(1);

}

fprintf(fp,"%-12s%-16s%-12s%-8.2lf%-8s\n",account\_Message.accountID,account\_Message.time,account\_Message.type,account\_Message.money,account\_Message.toAccount); //添加交易流水到文件中

fclose(fp);

}

//获取当前时间函数

char\* getDate()

{

char\*current\_time;

time\_t timep;

struct tm \*p;

time (&timep);

p=gmtime(&timep);

current\_time=(char\*)malloc(sizeof(char)\*20);

sprintf(current\_time,"%d/%d/%d/%d:%d",1900+p->tm\_year,1+p->tm\_mon,p->tm\_mday,8+p->tm\_hour,p->tm\_min); //将数据以字符串格式化输出

return current\_time;

}

//修改密码函数

void ChangePassword(Account \*str,Statement \*sta,int account\_Number)

//str为账户指针，sta为交易流水指针，account\_Number为账户数组下标

{

int j=0; //i控制结构体数组下标，j控制密码长度

Account a1,a2;

printf("\n\n\n\n\n\n\n\n\n");

printf("\t\t\t\t\tplease input your new password:");

while(j<=5)

{

a2.Password[j]=getch();

a2.Password[j]+=51; //密码加密处理

printf("\*");

j++;

}getch();

a2.Password[6]='\0';

j=0;

printf("\n\t\t\t\t\tplease input your new password again:");

while(j<=5)

{

a1.Password[j]=getch();

a1.Password[j]+=51; //密码加密处理

printf("\*");

j++;

}getch();

a1.Password[6]='\0'; //字符串结束标志位，使s1.Password成为字符串，防止出现乱码

if(strcmp(a2.Password,a1.Password)!=0)

{

printf("\n\t\t\t\t\tOperation failed(the two passwords entered are different!)\n");

}

else

{

printf("\n\t\t\t\t\tOperation complete!\n");

strcpy(str[account\_Number].Password,a1.Password);

writeFile(str);

}

printf("\t\t\t\t\tClick enter to return.");

getch();

system("cls"); //单击回车键返回上一级

menuAccount(str,sta,account\_Number); //回到二级菜单

}

//转账函数

void transferMoney(Account \*pa,Statement \*ps,int account\_Number)

//pa为账户指针，ps为交易流水指针，account\_Number为账户数组下标

{

double amount; //交易数额

int k; //k为toaccount用户下标

Statement a1,a2;

Account toaccount\_Number;

printf("\n\n\n\n\n\n\n\n\n");

printf("\t\t\t\t\tplease input amount:");

scanf("%lf",&amount);

while(amount>pa[account\_Number].money) //判断账户余额是否大于用户输入的值

{

printf("Insufficient Funds! please Input again:");

scanf("%lf",&amount);

}

printf("\t\t\t\t\tplease input toAccount:");

scanf("%s",toaccount\_Number.ID);

system("cls"); //清屏

k=find(pa,toaccount\_Number); //找到转账账户下标

while(account\_Number==k||k==-1) //判断用户输入的ID是否有效

{

printf("\t\t\t\t\terror input!\n");

printf("\t\t\t\t\tplease input again:");

scanf("%s",toaccount\_Number.ID);

system("cls");

k=find(pa,toaccount\_Number);

}

pa[account\_Number].money=pa[account\_Number].money-amount; //更新账户余额

pa[k].money=pa[k].money+amount; //更新被转帐用户的账户余额

strcpy(a1.accountID,pa[account\_Number].ID); //将登录用户交易记录存储在a1中

strcpy(a1.time,getDate());

strcpy(a1.type,"Transfer");

a1.money=amount;

strcpy(a1.toAccount,toaccount\_Number.ID);

strcpy(a2.accountID,pa[k].ID); //将被转账用户交易记录存储在a2中

strcpy(a2.time,getDate());

strcpy(a2.type,"Input");

a2.money=amount;

strcpy(a2.toAccount,pa[account\_Number].ID);

writeFile(pa); //更新账户金额信息

add\_Account\_Transactions(a1); //将a1中的记录存入文件中

add\_Account\_Transactions(a2); //将a2中的记录存入文件中

printf("\n\n\n\n\n\n\n\n\n");

printf("\t\t\t\t\tTransfer success!\n");

printf("\t\t\t\t\tNow your account balance is:%.2f\n",pa[account\_Number].money);

printf("\t\t\t\t\tThe page will jump in two seconds.\n");

Sleep(2000); //延时两秒

system("cls"); //清屏

menuAccount(pa,ps,account\_Number); //回到业务选择菜单

}

//存取款函数

void MoneyChange(Account \*pa,Statement \*ps,int choice,int account\_Number)

//ps为账户指针，ps为交易流水指针，choice为存取款业务选择，account\_Number为账户数组下标

{

double amount; //存取款金额

int i; //用于下面菜单业务选择

Statement account\_trans; //存储本次交易流水信息

char judge; //用来忽略回车符

printf("\n\n\n\n\n\n\n\n\n");

printf("\t\t\t\t\tplease input the amount:");

scanf("%lf",&amount);

getchar();

if (choice==2) //判断余额增减情况

{

while(amount>pa[account\_Number].money) //判断账户余额是否大于用户输入的值

{

printf("\t\t\t\t\tBalance isn't enough!please input the amount again:");

scanf("%lf",&amount);

}

amount=-amount;

}

pa[account\_Number].money=pa[account\_Number].money+amount; //更新数组账户余额

strcpy(account\_trans.toAccount,"NULL"); //存取款无toaccount

strcpy(account\_trans.accountID,pa[account\_Number].ID); //记录账户ID

account\_trans.money=abs(amount); //记录交易金额

strcpy(account\_trans.time,getDate()); //记录交易时间

switch (choice)

{

case 1:

strcpy(account\_trans.type,"Deposit");

break;

case 2:

strcpy(account\_trans.type,"Withdrawal");

break; //记录资金变动类型

}

writeFile(pa); //更新文件账户余额

add\_Account\_Transactions(account\_trans); //添加交易流水

Sleep(1000); //延时1秒

printf("\n");

printf("\t\t\t\t\tDesposit success!\n");

printf("\t\t\t\t\tNow your account balance is:%.2f\n",pa[account\_Number].money);

printf("\t\t\t\t\tWhether to print the return form or not?(Y/N):");

judge=getchar();

while(judge!='Y'&&judge!='N'&&judge!='y'&&judge!='n')

{

printf("\t\t\t\t\tInput error! Whether to print the return form or not?(Y/N):");

getch(); //忽略回车字符

judge=getchar();

}

if(judge=='Y'||judge=='y')

{

printf("\t\t\t\t\tComplete return printing!\n");

}

printf("\t\t\t\t\tThe page will jump in two seconds.");

Sleep(2000); //延时2秒

system("cls"); //清屏

printf("\n\n\n\n\n\n\n\n\n");

printf("\t\t\t\t\t##########################################\n");

printf("\t\t\t\t\t# #\n");

printf("\t\t\t\t\t# 1.Withdraw card 2.Return the main menu #\n");

printf("\t\t\t\t\t# #\n");

printf("\t\t\t\t\t##########################################\n");

printf("\t\t\t\t\tplease input your choice:");

scanf("%d",&i);

system("cls"); //清屏

if(i==2)

menuAccount(pa,ps,account\_Number); //回到二级菜单

}

//流水信息查询函数

void searchManage(Account \*acc,Statement \*sta,int account\_Number) //acc为账户指针，sta为交易流水指针，account\_Number为账户数组下标

{

int i=0; //用于遍历流水信息数组

printf("\n\n\n\n\n\n\n\n\n");

printf("\t\t\t\t\tAccount ID Time Type Amount/$ toAccount ID\n");

readFile\_Transactions(sta); //读取文件流水信息到数组

while(sta[i].money>10e-3&&i<=1000)

{

if(strcmp(sta[i].accountID,acc[account\_Number].ID)==0)

printf("\t\t\t\t\t%-12s%-16s%-12s%-12.2lf%-10s\n",sta[i].accountID,sta[i].time,sta[i].type,sta[i].money,sta[i].toAccount);

i++;

}

printf("\t\t\t\t\tClick enter to return.");

getch();

system("cls"); //单击回车键返回上一级

}

int main()

{

int choice; //用户业务选择

struct Account str[25]; //存储文件用户账户信息

struct Statement sta[1000]; //存储交易流水信息

readFile(str); //读入文件

memset(sta,0,sizeof(Statement)\*1000); //初始化sta数组

do

{

menu(); //一级菜单

printf("\t\t\t\t\tPlease input your choice:");

scanf("%d",&choice);

system("cls"); //清屏

if(choice>=0&&choice<=3)

runMain(str,sta,choice);

else

printf("error input,please input your choice again!\n");

}while(choice);

printf("\n\n\n\n\n\n\n\n\n\n\t\t\t>>>>>>>>>> Exit success,please keep your bank card! <<<<<<<<<<<<\n");

return 0;

}

**五、测试数据及其结果分析**

总帐户信息:

账户ID 姓名 密码（加密后） 账户余额 账户状态（1为锁定）

2018011801 yyding dddddd 800.00 0

2018011802 mma eeeeee 1200.00 0

2018011803 zthang ffffff 1000.00 0

2018011804 zlzhang gggggg 1000.00 0

2018011805 yjli hhhhhh 1000.00 0

2018011806 lling iiiiii 1000.00 0

2018011807 dding jjjjjj 1000.00 0

2018011808 yzhao kkkkkk 1000.00 0

2018011809 wltao llllll 1000.00 0

2018011810 kyxia dddddd 1000.00 0

2018011811 jhluo eeeeee 1000.00 0

2018011812 kyshi ffffff 1000.00 0

2018011813 ybow gggggg 1000.00 0

2018011814 jrjiang hhhhhh 1000.00 0

2018011815 wxlu iiiiii 1000.00 0

2018011816 ypeng jjjjjj 1000.00 0

2018011817 zjcao kkkkkk 1000.00 0

2018011818 gmu llllll 1000.00 0

2018011819 jjcai dddddd 1000.00 0

2018011820 ybzhang eeeeee 1000.00 0

运行后，程序的登录菜单界面如下。

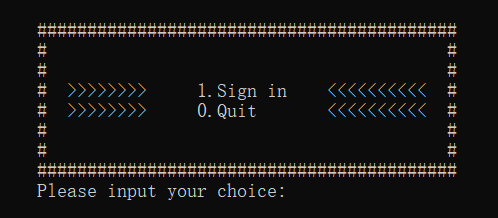


图8用户登录菜单界面

(1)登录和退出测试

1. 输入：0

输出：Exit success! please keep your bank card.

1. 输入：1

输出：please input account ID:

输入：2018011801

输出：please input passwords：

输入：111111（实际看到的是\*\*\*\*\*\*）

输出：Login success! The page will jump in two seconds!

2秒后输出用户菜单：

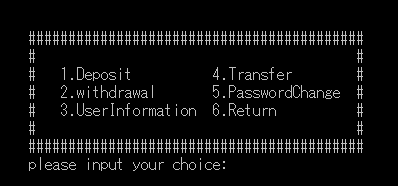


图9 用户菜单界面

（1）存/取款测试

输入：1/2（1为存款，2为取款）

输出：please input amount：

输入：100

输出：Deposit/Withdrawal success!

Now your account balance is : 900.00/1100.00

Whether to print the return form or not?(Y/N)

输入：Y

输出：complete printing！The page will jump in two seconds.（若输入N则不输出此条语句，直接跳转下一界面）

两秒后：跳转下一界面：

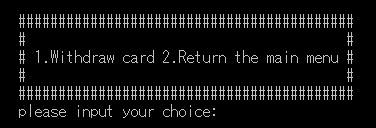


图10 打印回单返回界面

输入：1 则跳转至登陆菜单界面

输入：2 则跳转至用户菜单界面

（2）信息查询测试

输入：3

输出：

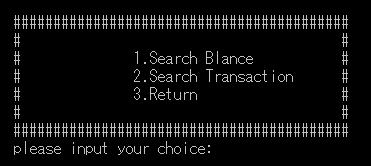


图11 用户信息查询界面菜单

A.查询余额测试

输入：1

输出：Now your balance is:900

Click enter to return.

B.查询交易记录测试

输入：2

输出：

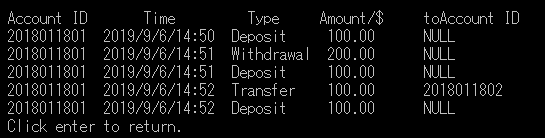


图12 用户交易流水记录

C．退出测试

输出：3

输出：

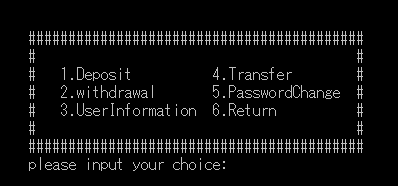


图13 用户菜单界面

（3）转账测试

输入：4

输出：please input amount:

输入：200

输出：please input toAccount ID：

输入：2018011802

输出：Transfer success!

Now your account balance is:700

The page will jump in two seconds.

两秒后：跳转至用户菜单界面

（4）修改密码测试

输入：5

输出：Please input your new password:

输入：222222（实际看到的是\*\*\*\*\*\*）

输出：Please input your new password again:

输入：222222（实际看到的是\*\*\*\*\*\*）

输出：Complete Operation!

Click enter to return.

（5）返回上一级测试

输入：6 则跳转至登录菜单界面

结果分析如下。

（1）：在各个菜单用户选择时，若输入的不是菜单中存在的数选项，则系统会提示用户，选项输入错误，重新输入。

（2）：若用户登录时输入不存在的账户，则系统会提示用户输入错误，重新输入。用户输入密码时有三次机会，每次输入错误后系统会提示用户剩余次数，若三次密码均输入错误，则会锁卡，提示用户联系管理员解锁。

（3）：若用户取款和转账时输入的金额大于账户余额，则系统提示用户账户余额不足，请重新输入。

（4）：存取款完成后打印回单时，若用户输入Y、y、N、n以外的字符，则系统提示用户输入的字符不正确，请用户重新输入。

（5）：转账时若用户输入了不存在的用户，则系统提示用户账户不存在，请重新输入。

（6）：修改密码时若两次输入的密码不一致，则系统会提示用户两次输入密码不一致，并返回上一级菜单。

**六、调试过程中的问题**

（1）获取时间函数问题：刚开始不知道如何获取系统时间，后面通过查找资料，得到返回值为整型的时间。然后通过fprintf函数，使类型为整型的时间变成字符串类型。

（2）密码隐藏问题：开始我们的密码是明码，经过老师提醒后意识到了这个问题。然后我们使用getch函数与printf函数使得输入密码时别人只能看见“\*”而开不见我们的密码。

（3）密码验证问题：用户登录输入密码验证时，即使用户输入正确密码也无法登录，原因是由于密码隐藏功能，使得字符密码数组不是一个完整字符串，导致进行比较时在密码后面还有许多乱码字符（烫烫烫）。解决方法：在密码输入完成后，添加“\0”字符作为字符串结束标记。

（4）存取款、转账后账户更新问题：当用户进行交易时，账户余额势必会发生变化，如何将更新后的账户信息存储到文件中遇到了麻烦。当时初步想法是通过文件指针定位到账户金额位置然后修改，然后我们发现了定位的复杂性，以及考虑到账户信息的数量，最后选择通过重新覆盖的方式，将发生交易后被修改的数组立即写入到文件中，覆盖之前的账户信息，从而达到更新账户信息效果。

（5）输入密码无法退格问题：当用户密码输入错误需要退格删除时，按下退格无法执行退格，相反还会被认为是密码而输入进去。解决方法：通过if语句判断输入字符是否是退格键，如是，则执行退格功能，然后使用continue函数退出此次循环。

（6）界面交互问题：如何实现用户业务选择后界面能够进行跳转，提高人机交互性。解决办法：使用清屏函数删除控制台上一次的运行纪录，再配合Sleep延时函数延长时间，使得跳转界面过程变得可见。

**七、课程总结**

程序设计心得

-----B18011834熊凌宇

此次的程序设计周让我们小组每个人五轮是在专业知识的掌握还是团队合作完成项目的锻炼都受益良多，同时又有了第一次项目开发经验，对于以后的项目开发能够提供宝贵经验。在选题时，我们选择了与生活息息相关的项目，这样能够站在用户的角度去优化我们界面、增强用户体验感。

选题之后，首先我将项目进行拆分，拆分成多个模块，确定项目架构，然后每个模块需要哪些函数，确定好要使用哪些函数。由于第一次开发项目并没有太多经验，所以拆分模块和封装函数花费了较多时间。之后我们便分工明确，我负责界面菜单构建、用户登录函数、读取和存储文件函数、添加交易流水记录函数的封装，他们负责存取款函数、转账函数、查询交易记录函数的封装。当我将界面做好调试完成且各界面交互性良好时，内心无比喜悦，这也让我对接下来的工作充满了热情。

第二天开始后就遇到了一些问题，用户登录进行密码验证时，即使用户输入正确密码后仍然显示密码错误。当时我们百思不得其解，最后通过跟踪发现了问题。由于实现密码隐藏功能的原因，所以密码以单个字符的方式逐个读入，此时数组中存储的并不是字符串，只是字符，所以结尾会出现各种乱码，使用”strcmp”函数时就会发生错误。解决方法是在密码之后添加 ‘\0’，代表字符串终止。另一个问题就是发生交易后，账户金额改变后账户如何更新问题。最初我们打算使用指针定位到文件中的金额位置，然后修改。我们当时感觉实现难度颇大，通过询问朋友，最后确定了这样的方式：直接在结构体数组中改变，此时数组中的信息已更新，再将其覆盖到之前文件中从而达到文件更新效果。

第四天代码基本写完，剩下的就是进行调试、发现其中bug，然后针对性进行解决。我们还添加了清屏函数，使得我们界面菜单间能够做到独立显示，添加了延时函数，极大真实模拟ATM取款机，提高用户体验感。还有就是开始进行代码注释，用户界面美化、优化。

现在回顾整个过程，感觉所有的努力都得到了回报。不仅是在使用自己开发的程序时的喜悦，更重要的是在开发时碰到的问题被解决后的那一瞬间，内心的喜悦溢于言表。这一周，我们遇到了许多问题，也解决了许多问题。这一周的程序设计，将成为我们每个人的珍贵回忆。

程序设计心得

-------B18011830曹子健

为期一周的实习结束了。这一周里我深刻的体验到了一些感触。

上上学期学习的C语言已经被我忘记了很多，而且并非考试重点的文件和字符串部分反而是本次实习的重点部分，可以说我的上手相当麻烦。幸好借助书籍和网络还是勉强能写出部分代码并提出一些解决方案。这种自由设计和作业差距非常大，在大一时我的C语言成绩还不错，但在此次实习中我深刻体验到了实际应用的困难。

有一个段子说，程序员就是解决了一个bug又冒出来七个bug的职业，我曾经以为这只是一个笑话，现在我觉得这句话说得很对。因为组长精益求精的加密设计，本来可以使用的退格功能失去了效力，最终求助老师得到了解决。由于没有注意到返回值是自然数而非正整数的问题，我花了相当长的时间才想到把另一种情况下的返回值调整。一步一步调试把握每一个细节的感觉让人非常抓狂，我至少五次抑制住了想要砸电脑的冲动（尴尬而不失礼貌的笑）。不过，在最终完成的时候我还是挺兴奋的——虽然组长熊凌宇和另一名组员赵源的贡献比我大很多。所谓带妹上分，应该就是说的我，嗯，我就是那个妹。

组长的设计非常简明扼要而追求完美，他的分析切中肯綮，把握住了问题的本质，将其划分为易于理解的部分，将抽象转化为具体。经过这次实习，我愈发意识到形成一种思维的重要性。只有形成了那种特有的思维，才可以拥有解决问题的钥匙。形成思维的先决条件是不断的练习和归纳总结，所以……大一下来，我们的差距已经很大了，在深深敬佩组长的同时，我也该继续努力才行。

程序设计心得

------B18011827赵源

不同于平日里稍显松散的上课周，程序设计周需要我们在规定时间内完成任务，但它亦不同于期末逼近时单调压抑的复习周，程序设计周在完成任务的过程中又时常跌宕起伏，既有面对困难久攻不下的焦急难耐，又有历尽艰辛终得回报的欣慰欢喜。虽然整个过程的时间只有短短五天，但其中所包含的汗水、喜悦与进步足以令我永久铭记。

要说让我印象最为深刻的，那一定是编写代码时出的五花八门的漏洞。由于我们编写的程序时ATM仿真系统，所以安全机制问题需要我们格外注意，如何判断账户取款转账时余额是否充足、如何实现用户输入密码时屏幕上不直接显示密码、甚至于如何在记录账户信息中的原文件中隐藏真实密码等等…这些都是现实生活中的ATM机必须要具备的功能。面对这些困难，我们小组热烈讨论，各抒己见，最终将它们一一解决：我们先是利用while循环语句和if语句的嵌套解决了大多类似判断余额是否充足、判断ID是否存在的问题，又上网学习getch函数和printf函数的组合，使用户在输入字符时显示在屏幕上的为“\*”号，并以此解决密码被他人偷看的问题，而最后的密码加密机制则通过一个小算法解决。此外，实现各个基本功能的函数以及实现界面交互的代码在最初编好之后漏洞百出，如何精准定位登陆用户的下标，以下标来调用用户结构体数组中的元素，实现对账户信息的修改，如何实现界面的跳转等等，解决这些漏洞既需要我们参考教材，学习书上相似的例子，还需要我们上网学习教材上没有的知识点。

程序设计周中所遇到的问题远不止上面提到的那些，我也不再一一赘述。这些问题虽然难易程度不同，有的棘手有的简单，但都让我们得到了一次发现问题并解决问题的锻炼，给了我们一个提升自身能力的机会。程序设计周已然在我的大学生活中留下了不可磨灭的痕迹，我将记得它带给我的每一次挑战，每一次进步和每一次喜悦。