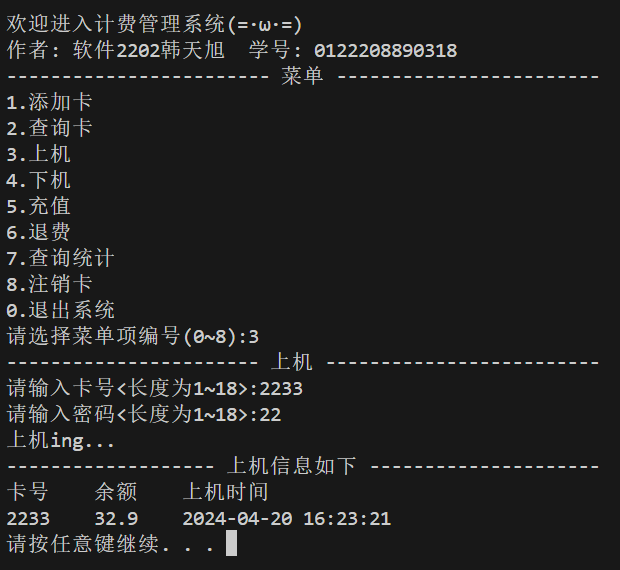
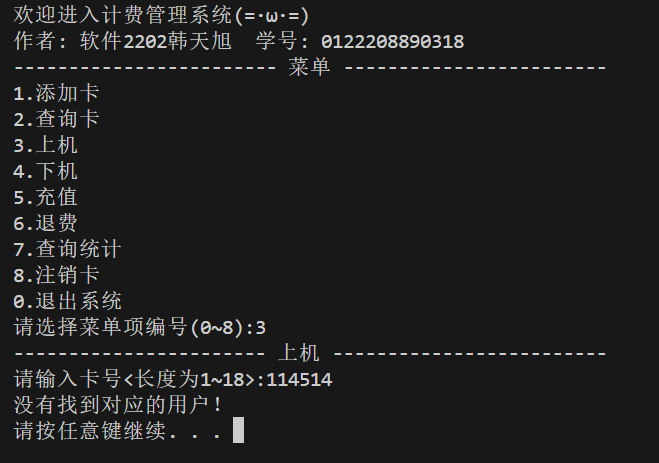
测试用例：

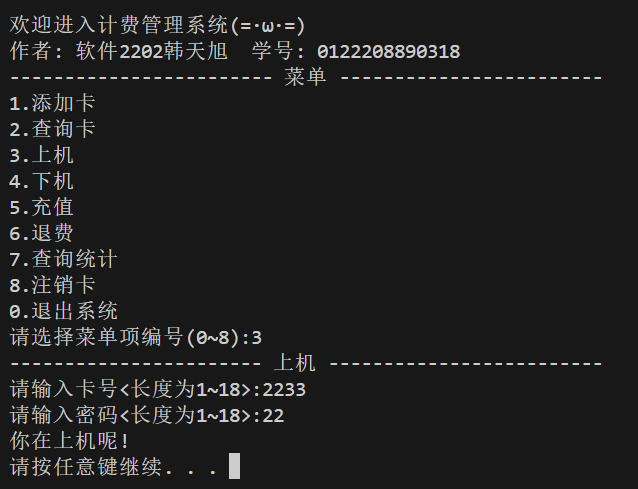
1.



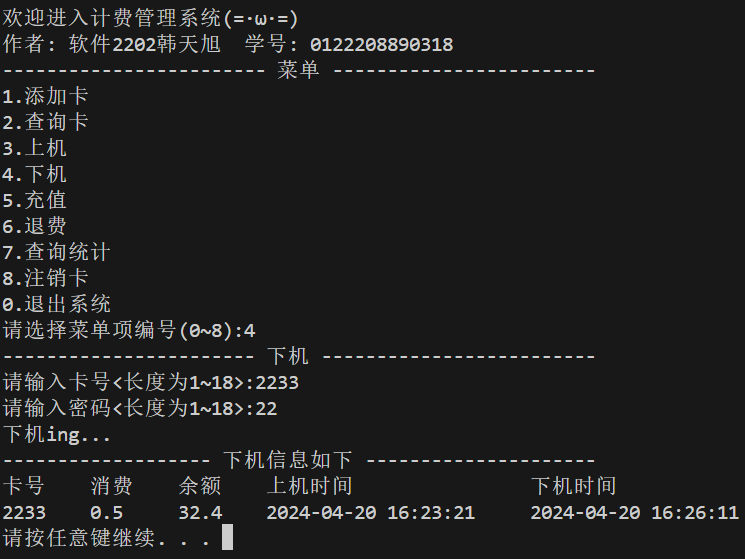
2.



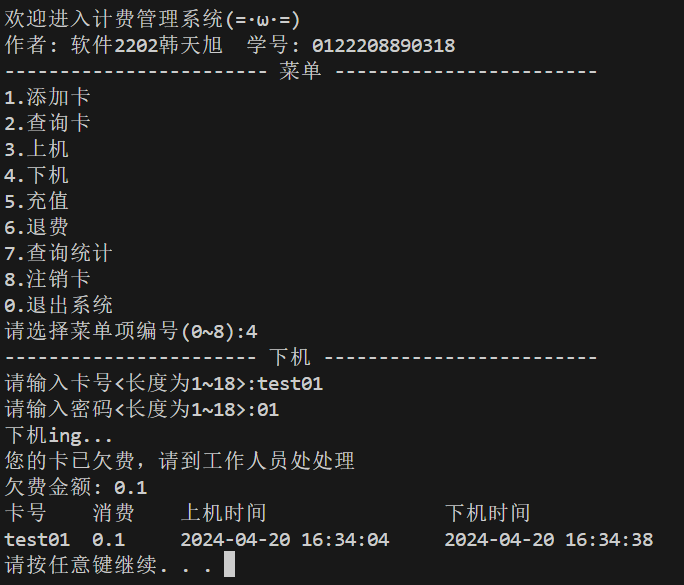
3.



4.



5.



源代码：

main.c

#include <stdio.h>

#include "service.h"

#include "menu.h"

int main()

{

    Cards c;

    Billings b;

    initSystem(&c,&b);

    char ipt[10];

    int choice = 9;

    do

    {

        system("cls");

        showMenu();

        do

        {

            scanf("%s", ipt);

        } while (ipt==NULL);

        choice = ipt[0] - '0';

        if (0 <= choice && choice <= 8)

        {

            switch (choice)

            {

                case 1:

                    ReadCard(&c);

                    break;

                case 2:

                    SearchCard(&c);

                    break;

                case 3:

                    LogIn(&c,&b);

                    break;

                case 4:

                    LogOff(&c,&b);

                    break;

                case 5:

                    InsertMoney();

                    break;

                case 6:

                    GetMoney();

                    break;

                case 7:

                    ShowStatistics();

                    break;

                case 8:

                    SignOff();

                    break;

                case 0:

                    Exit(&c);

                    break;

                default:

                    showDefault();

                    break;

            }

        }

        else

        {

            choice=9;

            showDefault();

        }

    } while (choice);

    return 0;

}

menu.c

#include "menu.h"

void showMenu()

{

    printf("欢迎进入计费管理系统(=·ω·=) \n\

作者: 软件2202韩天旭  学号: 0122208890318 \n\

------------------------ 菜单 ------------------------ \n\

1.添加卡 \n\

2.查询卡 \n\

3.上机 \n\

4.下机 \n\

5.充值 \n\

6.退费 \n\

7.查询统计 \n\

8.注销卡 \n\

0.退出系统 \n\

请选择菜单项编号(0~8):");

}

menu.h

#pragma once

#include <stdio.h>

void showMenu();

service.c

#include <stdio.h>

#include "service.h"

#include "tools.h"

void ReadCard(Cards \*c)

{

    printf("----------------------- 添加卡 -----------------------\n");

    Card\* newc = (Card\*)malloc(sizeof(Card));

    char id[MAXLEN];

    char pwd[MAXLEN];

    float balance;

    InputID(id);

    if(search(c,id)!=NULL)

    {

        printf("这个卡已经有啦\n");

        system("pause");

        return ;

    }

    strcpy(newc->aId,id);

    InputPWD(pwd);

    strcpy(newc->aPwd,pwd);

    while(1)

    {

        printf("请输入业务金额<数值为1~100>:");

        scanf("%f",&balance);

        if(balance<1 || balance>100)

            printf("数据非法，请重新输入!\n");

        else

            break;

    }

    newc->fBalance=balance;

    newc->nStatus=0;

    newc->fTotalUse=0;

    newc->nUseCount=0;

    newc->nDel=0;

    newc->next=NULL;

    newc->next1=NULL;

    time(&(newc->tStart));

    newc->tEnd = newc->tLast = newc->tStart;

    struct tm \* tend = localtime(&newc->tEnd);

    tend->tm\_year += 5;

    newc->tEnd = mktime(tend);

    printf("请确认：\n");

    printf("-----添加卡的信息如下-----\n");

    printf("卡号\t密码\t状态\t开卡金额\n");

    printf("%s\t%s\t%d\t%.1f\n",newc->aId,newc->aPwd,newc->nStatus,newc->fBalance);

    system("pause");

    InsertCard(c,newc);

    printf("开卡成功(=·ω·=)\n");

    system("pause");

}

void SearchCard(Cards\* c)

{

    printf("----------------------- 查询卡 -----------------------\n");

    char id[MAXLEN];

    InputID(id);

    Cards ics;

    int k=vague\_search(c,&ics,id,get\_size(id));

    if(k==0)

    {

        printf("什么也没有找到QAQ\n");

        system("pause");

        return ;

    }

    printf("-----查询到的卡的信息如下-----\n");

    Card\* ic = ics.head;

    while(ic!=ics.tail->next1)

    {

        char time[40];

        time\_to\_string(ic->tLast,time,sizeof(time));

        printf("卡号\t状态\t余额\t累计使用\t使用次数\t上次使用时间\n");

        printf("%s\t%d\t%.1f\t%.1f\t\t%d\t\t%s\n",ic->aId,ic->nStatus,ic->fBalance,ic->fTotalUse,ic->nUseCount,time);

        ic=ic->next1;

    }

    system("pause");

    Card\* ic0 = c->head;

    while(ic0!=NULL)

    {

        ic0->next1=NULL;

        ic0=ic0->next;

    }

}

void LogIn(Cards\* c,Billings\* b)

{

    printf("----------------------- 上机 -------------------------\n");

    char id[MAXLEN];

    InputID(id);

    Card\* cd;

    if((cd=search(c,id))==NULL)

    {

        printf("没有找到对应的用户！\n");

        system("pause");

        return ;

    }

    char pwd[MAXLEN];

    InputPWD(pwd);

    if(strcmp(cd->aPwd,pwd)==0)

    {

        if(cd->fBalance<0)

        {

            printf("您的卡已欠费，请到工作人员处处理\n");

            printf("欠费金额: %.1lf\n",-(cd->fBalance));

            system("pause");

            return ;

        }

        if(cd->nStatus==1)

        {

            printf("你在上机呢!\n");

            system("pause");

            return ;

        }

        printf("上机ing...\n");

        cd->nStatus=1;

        ++cd->nUseCount;

        Billing\* newB = CreateBilling(id);

        cd->tLast = newB->tStart;

        InsertBilling(b,newB);

        LoginInfo lgif;

        strcpy(lgif.aCardName,id);

        lgif.tLogin = newB->tStart;

        lgif.fBalance = cd->fBalance;

        char time[40];

        time\_to\_string(lgif.tLogin,time,sizeof(time));

        printf("------------------- 上机信息如下 ---------------------\n");

        printf("卡号\t余额\t上机时间\n");

        printf("%s\t%.1f\t%s\n",lgif.aCardName,lgif.fBalance,time);

    }

    else

    {

        printf("密码错误!\n");

    }

    system("pause");

}

void LogOff(Cards\* c,Billings\* b)

{

    printf("----------------------- 下机 -------------------------\n");

    char id[MAXLEN];

    InputID(id);

    Card\* cd;

    if((cd=search(c,id))==NULL)

    {

        printf("没有找到对应的用户！\n");

        system("pause");

        return ;

    }

    char pwd[MAXLEN];

    InputPWD(pwd);

    if(strcmp(cd->aPwd,pwd)==0)

    {

        if(cd->nStatus==0)

        {

            printf("你没上机呢!\n");

            system("pause");

            return ;

        }

        Billing\* targetB = searchLoginBill(b,id);

        if(targetB == NULL)

        {

            printf("没有找到对应的消费记录!\n");

            system("pause");

            return ;

        }

        printf("下机ing...\n");

        cd->nStatus=0;

        time(&(targetB->tEnd));

        cd->tLast = targetB->tEnd;

        double price = CalcuPrice(targetB->tStart,targetB->tEnd);

        // printf("price = %.3lf\n",price);

        cd->fTotalUse+=price;

        cd->fBalance-=price;

        targetB->fAmount = price;

        targetB->nStatus = 1;

        char time1[40];

        time\_to\_string(targetB->tStart,time1,sizeof(time1));

        char time2[40];

        time\_to\_string(targetB->tEnd,time2,sizeof(time2));

        if(cd->fBalance>=0)

        {

            printf("------------------- 下机信息如下 ---------------------\n");

            printf("卡号\t消费\t余额\t上机时间\t\t下机时间\n");

            printf("%s\t%.1lf\t%.1lf\t%s\t%s\n",cd->aId,price,cd->fBalance,time1,time2);

        }

        else

        {

            printf("您的卡已欠费，请到工作人员处处理\n");

            printf("欠费金额: %.1lf\n",-(cd->fBalance));

            printf("卡号\t消费\t上机时间\t\t下机时间\n");

            printf("%s\t%.1lf\t%s\t%s\n",cd->aId,price,time1,time2);

        }

    }

    else

    {

        printf("密码错误!\n");

    }

    system("pause");

}

void InsertMoney()

{

    printf("----------------------- 充值 -------------------------\n");

    system("pause");

}

void GetMoney()

{

    printf("----------------------- 退费 -------------------------\n");

    system("pause");

}

void ShowStatistics()

{

    printf("--------------------- 查询统计 -----------------------\n");

    system("pause");

}

void SignOff()

{

    printf("----------------------- 注销卡 -----------------------\n");

    system("pause");

}

void Exit(Cards\* c)

{

    printf("正在保存数据ing。。。\n");

    save(c);

    printf("感谢你使用计费管理系统\n");

    system("pause");

}

void showDefault()

{

    printf("输入有误，请重新输入\n");

    system("pause");

}

service.h

#pragma once

#include <stdio.h>

#include <time.h>

#include <string.h>

#include <stdlib.h>

#define MAXLEN 18

#define SIZE 100

typedef struct card

{

    char aId[MAXLEN];

    char aPwd[MAXLEN];

    int nStatus;

    time\_t tStart;

    time\_t tEnd;

    float fTotalUse;

    time\_t tLast;

    int nUseCount;

    float fBalance;

    int nDel;

    struct card\* next;

    struct card\* next1;

} Card;

typedef struct cards

{

    Card \*head,\*tail;

}Cards;

typedef struct Billing

{

    char aCardName[18];

    time\_t tStart;

    time\_t tEnd;

    float fAmount;

    int nStatus;

    int nDel;

    struct Billing\* next;

}Billing;

typedef struct Billings

{

    Billing \*head,\*tail;

}Billings;

// typedef struct BillingNode

// {

//     Billing data;

//     struct BillingNode \*next;

// }BillingNode,\*lpBillingNode;

typedef struct LoginInfo

{

    char aCardName[18];

    time\_t tLogin;

    float fBalance;

}LoginInfo;

typedef struct SettleInfo

{

    char aCardName[18];

    time\_t tStart;

    time\_t tEnd;

    float fAmount;

    float fBalance;

}SettleInfo;

void ReadCard(Cards \*c);

void SearchCard(Cards\* c);

void LogIn(Cards\* c,Billings\* b);

void LogOff(Cards\* c,Billings\* b);

void InsertMoney();

void GetMoney();

void ShowStatistics();

void SignOff();

void Exit(Cards\* c);

void showDefault();

tools.c

#include "tools.h"

void delString (char\* txt)

{

    for(int i=0;i<strlen(txt);++i) txt[i]='\0';

}

void printTime(time\_t tt)

{

    struct tm \* timeinfo;

    timeinfo = localtime(&tt);

    char strTime[40];

    strftime(strTime,sizeof(strTime),"%Y年%m月%d日 %H:%M",timeinfo);

    printf(strTime);

}

void time\_to\_string(time\_t tt,char\* strTime,int size)

{

    struct tm \* timeinfo;

    timeinfo = localtime(&tt);

    strftime(strTime,size,"%Y-%m-%d %H:%M:%S",timeinfo);

}

void cardcpy(Card\* b,Card\* a)

{

    strcpy(b->aId,a->aId);

    strcpy(b->aPwd,a->aPwd);

    b->fBalance = a->fBalance;

    b->fTotalUse = a->fTotalUse;

    b->nDel = a->nDel;

    b->nStatus = a->nStatus;

    b->nUseCount = a->nUseCount;

    b->tEnd = a->tEnd;

    b->tLast = a->tLast;

    b->tStart = a->tStart;

}

void initSystem(Cards\* c,Billings\* b)

{

    c->tail=NULL;

    c->head=c->tail;

    b->tail=NULL;

    b->head = b->tail;

    FILE \*fp;

    if((fp=fopen("card\_file.txt","r"))==NULL)

    {

        return ;

    }

    int i=0;

    char line[1024];

    long int Start,End,LastTime;

    while(fgets(line,sizeof(line),fp)!=NULL)

    {

        Card\* ic = (Card\*)malloc(sizeof(Card));

        sscanf(line,"%[^##]##%[^##]##%d##%ld##%ld##%f##%ld##%d##%f##%d\n",ic->aId,ic->aPwd,&ic->nStatus,&Start,&End,&ic->fTotalUse,&LastTime,&ic->nUseCount,&ic->fBalance,&ic->nDel);

        ic->tStart=(time\_t)Start;

        ic->tEnd=(time\_t)End;

        ic->tLast=(time\_t)LastTime;

        ic->next1=NULL;

        InsertCard(c,ic);

        ++i;

    }

    system("cls");

    printf("好像连上了，获得了%d条数据\n",i);

    system("pause");

}

void save(Cards\* c)

{

    FILE \*fp,\*fd;

    Card\* ic = c->head;

    int i=0;

    if(ic==NULL)

    {

        return ;

    }

    if((fp = fopen("card\_file.txt","w")) == NULL)

    {

        printf("数据保存失败！\n");

        return ;

    }

    while(ic!=NULL)

    {

        ic->next1=NULL;

        long int sstart=ic->tStart;

        long int send=ic->tEnd;

        long int slast=ic->tLast;

        fprintf(fp,"%s##%s##%d##%ld##%ld##%.1f##%ld##%d##%.1f##%d\n",ic->aId,ic->aPwd,ic->nStatus,sstart,send,ic->fTotalUse,slast,ic->nUseCount,ic->fBalance,ic->nDel);

        Card \*p = ic;

        ic=ic->next;

        free(p);

        ++i;

    }

    fclose(fp);

    printf("主人，保存了%d条数据\n",i);

}

Card\* search(Cards\* c,char\* id)

{

    Card\* p=c->head;

    while(p!=NULL)

    {

        if(strcmp(p->aId,id)==0) return p;

        p=p->next;

    }

    return NULL;

}

Billing\* searchLoginBill(Billings\* b,char\* id)

{

    Billing\* p=b->head;

    while(p!=NULL)

    {

        if(strcmp(p->aCardName,id)==0 && p->nStatus==0) return p;

        p=p->next;

    }

    return NULL;

}

void get\_next(char \*txt,int \*next,int txt\_size)

{

    int j=0,k=-1;

    next[j]=-1;

    while(j<txt\_size-1)

    {

        if(k==-1 ||txt[j]==txt[k])

        {

            ++j;++k;

            next[j]=k;

        }

        else k=next[k];

    }

}

int index\_KMP(char\* txt,char \*id,int id\_size,int txt\_size)

{

    int next[txt\_size-1];

    get\_next(txt,next,txt\_size);

    int i=0,j=0;

    while(i<id\_size-1 && j<txt\_size-1)

    {

        if(j==-1 || id[i]==txt[j]) ++i,++j;

        else j=next[j];

    }

    if(j>=txt\_size-1) return i-txt\_size+1;

    else return -1;

}

int get\_size(char\* txt)

{

    int i=0;

    while(txt[i]!='\0') ++i;

    return i+1;

}

int vague\_search(Cards \*c,Cards\* ret,char\* txt,int txt\_size)

{

    Card\* p=c->head;

    int sign=1;

    ret->tail=NULL;

    ret->head=ret->tail;

    while(p!=c->tail->next)

    {

        int idx=index\_KMP(txt,p->aId,get\_size(p->aId),txt\_size);

        // printf("%d\n",idx);

        if(idx!=-1)

        {

            InsertCard1(ret,p);

            sign=0;

        }

        p=p->next;

    }

    if(sign) return 0;

    return 1;

}

void PrintCard(Cards \*c)

{

    Card\* p=c->head;

    while(p!=NULL)

    {

        printf("%s\t%s\t%d\t%.1f\n",p->aId,p->aPwd,p->nStatus,p->fBalance);

        p=p->next;

    }

}

void InsertCard1(Cards\* c,Card\* newc)

{

    if(c->head)

    {

        c->tail->next1=newc;

        c->tail=c->tail->next1;

        c->tail->next1=NULL;

    }

    else

    {

        c->head=c->tail=newc;

    }

}

void InsertCard(Cards\* c,Card\* newc)

{

    if(c->head)

    {

        c->tail->next=newc;

        c->tail=c->tail->next;

        c->tail->next=NULL;

    }

    else

    {

        c->head=c->tail=newc;

    }

}

void InsertBilling(Billings\* b,Billing\* newb)

{

    if(b->head)

    {

        b->tail->next=newb;

        b->tail=b->tail->next;

        b->tail->next=NULL;

    }

    else

    {

        b->head=b->tail=newb;

    }

}

Billings\* CreateBilling(char \*id)

{

    Billing\* newB = (Billing\*)malloc(sizeof(struct Billing));

    strcpy(newB->aCardName,id);

    time(&(newB->tStart));

    time(&(newB->tEnd));

    newB->fAmount = 0;

    newB->nStatus = 0;

    newB->nDel = 0;

    return newB;

}

void InputID(char\* id)

{

    while(1)

    {

        printf("请输入卡号<长度为1~18>:");

        scanf("%s",id);

        if(strlen(id)>MAXLEN-1)

        {

            printf("卡号过长，请重新输入!\n");

            delString(id);

        }

        else

            break;

    }

}

void InputPWD(char\* pwd)

{

    while(1)

    {

        printf("请输入密码<长度为1~18>:");

        scanf("%s",pwd);

        if(strlen(pwd)>MAXLEN-1)

        {

            printf("密码过长，请重新输入!\n");

            delString(pwd);

        }

        else

            break;

    }

}

double CalcuPrice(time\_t tStart,time\_t tEnd)

{

    double deltaTime = tEnd - tStart;

    int price = 10;

    return deltaTime \* price / 3600;

}

tools.h

#pragma once

#include "service.h"

void delString (char\* txt);

void printTime(time\_t tt);

void time\_to\_string(time\_t tt,char\* strTime,int size);

void initSystem(Cards\* c,Billings\* b);

void save(Cards\* c);

Card\* search(Cards\* c,char\* id);

void PrintCard(Cards \*c);

void InsertCard(Cards\* c,Card\* newc);

void InsertBilling(Billings\* b,Billing\* newb);

Billings\* CreateBilling(char \*id);

Billing\* searchLoginBill(Billings\* b,char\* id);

void InsertCard1(Cards\* c,Card\* newc);

int vague\_search(Cards \*c,Cards\* ret,char\* txt,int txt\_size);

int get\_size(char\* txt);

void InputID(char\* id);

void InputPWD(char\* pwd);

double CalcuPrice(time\_t tStart,time\_t tEnd);