深 圳 大 学 实 验 报 告

课程名称：面向对象程序设计

实验项目名称：实验五 类与对象

学院：医学院

专业：生物医学工程

指导教师：李乔亮、邓云

报告人：陈焕鑫 学号：**2016222042** 班级：生工 **2** 班

实验时间：**2018.11.7**

实验报告提交时间：**2018.11.25**

教务部制

实验目的：

熟练掌握 C++中类与对象的使用

实验内容：

1. 从饭卡管理的角度，抽象并设计学生类:

1) 至少含括学生姓名，ID，余额，卡号，密码等属性。

2) 提供信息打印，饭卡充值，吃饭，消费，挂失/解挂 等接口，根据应用场景合理设计函数形参列表和返回类型。

3) 在主函数中构造学生对象，利用构造函数完成对象初始化，并模拟学生去食堂 圈存/就餐 过程，设计电子“菜单”，根据选择完成用餐。

4） 采用国际标准格式完成 h 和 cpp 文件编写。

参考运行示例：

本餐厅提供：
1. 米饭 2.5/两
2. 芋头 3.0/个
3. 已吃饱，退出
4. 充值
5. 挂失/解挂
6. 查询

请选择— 1 （用户键盘输入）

购买芋头成功！您的当前余额为 2.5 元。

2. 对题 1 的学生类进行优化，假设学生姓名长度不固定，要求能根据实际姓名的长度动态分配内存，从而避免可能的内存浪费。

实验环境与程序代码：

实验环境：win7 系统下的 Visual C++ 6.0

程序代码如下所示：

在 student.h 文件中抽象 CStudent 类，使得该类具有学生姓名、学号、校园卡号、余额和密码等信息，将这些信息作为私有变量。而将信息打印、饭卡充值、用餐、挂失等动作作为函数接口，提供给外部使用。

```
//student.h
#ifndef _STUDENT_H_
#define _STUDENT_H_

#define LOGIN_FAILED -1  //定义登录失败时函数的返回值

typedef struct           //定义用户注册表的节点
{
  char l_cardID[7];      //注册的卡号
  char l_cardPassword[7];  //注册的卡号对应的密码
}User;

typedef enum
{
  ERROR,  //定义错误为 0
  OK      //定义正确为 1
}Status;    //枚举：操作结果

typedef enum
{
  LOCK,       //挂失
  UNLOCK      //解挂
}e_cardStatus;    //枚举：卡的状态

class CStudent
{
private:
  //私有静态变量
  static User s_registry[20];   //生成一个静态的用户注册表数组
  static int s_userMax;       //最大支持的用户数量
  static int s_userNum;       //已有的用户数量
  //私有变量
  char  *name;              //用户名
  unsigned long   id;          //学号
  float balance;            //余额
  char  cardID[7];          //校园卡号
  char  password[7];         //密码
  int tableNumber;          //对应的注册表序号
```

```
  bool cardStatus;          //卡的状态（挂失/解挂）
  //私有函数
  static int cmpPassword(char *i_cardID, char *i_cardPassword); //核对密码函数
  void writeRegisty(void);                          //写注册表函数
public:
  CStudent();     //无参构造函数
  //带形参的构造函数
  CStudent(const char *i_name, const int i_id, const char *i_cardID, const char
*i_password);
  ~CStudent();    //析构函数

  static int searchAccount(char *i_cardID);       //查询用户是否已注册
  static int login(char *i_cardID, char *i_password); //用户登录
  static int getUserNum(void);                //获取已注册用户的数目

  bool registerUser(void);          //注册用户
  void showInfo(void);              //显示用户信息
  bool changeInfo(void);            //更改用户信息
  bool cardRecharge(float amount);    //充值
  bool eating(int choice);            //用餐
  bool lockCard(void);              //挂失
  bool unlockCard(void);            //解挂
  bool getcardStatus(void);         //获取卡的状态
};

bool randString(char starting_c, char ending_c, char *i_string,  int s_length); //生
成随机字符串

#endif
```

在 student.cpp 文件中编写 CStudent 类的函数实现。

```
//student.cpp

#include <iostream>
#include <stdlib.h>
#include "student.h"

using namespace std;

#define random(x)  (rand()%x)   //产生一个随机数

typedef struct
{
```

```
  char food[20];        //菜名
  float price;          //价格
}FoodNode;

FoodNode foodList[3] = { {"一两米饭", 2.5}, {"一个芋头", 3.0}, {"退出", 0} }; //菜单

int CStudent::s_userNum = 0;    //初始化用户数目
int CStudent::s_userMax = 0;    //初始化最大用户数目
User CStudent::s_registry[20] = { {"000000", "000000"} }; //初始化注册表

CStudent::CStudent()      //无参构造函数
{
  int sizeofname = 0;     //声明用来存放姓名长度的变量
  int match = OK;       //声明用来标记匹配的变量

  sizeofname = strlen("默认");        //默认初始化名称为默认
  name = new char[sizeofname + 1];    //申请内存空间
  if (name == NULL)             //申请失败
  {
    cout << "申请内存空间失败" << endl; //打印信息
    exit(0);                      //终止程序
  }
  strcpy(name, "默认");             //复制字符串到成员变量 name
  id = random(70) + 2016222000;     //生成随机的学号
  balance = 0.0;                  //初始余额为 0
  cardStatus = UNLOCK;              //卡的状态为解挂
  randString('0', '9', cardID, 6);      //生成随机的卡号

  while (1)
  {
    for (int i = 0; match != ERROR && i < s_userMax; i++) //查询注册表
    {
      match = strcmp(cardID, s_registry[i].l_cardID);   //新生成的学号与注册表比较
    }

    if (match == ERROR)           //如果生成的卡号在注册表中已有
    {
      randString('0', '9', cardID, 6);  //再次生成一个随机的卡号，继续比较
    }
    else
    {
      break;                    //跳出循环
    }
  }
```

```
  strcpy(password, "000000");       //默认初始密码为 000000
  tableNumber = s_userMax;          //注册表序号
  writeRegisty();                   //更新注册表
  s_userMax++;                      //注册表最大容量加 1
}
//有参构造函数
CStudent::CStudent(const char *i_name, const int i_id, const char *i_cardID, const char
*i_password)
{
  int sizeofname = 0;

  sizeofname = strlen(i_name);      //获取姓名的长度
  name = new char[sizeofname + 1];    //申请内存空间
  if (name == NULL)             //申请失败
  {
    cout << "申请内存空间失败" << endl; //打印信息
    exit(0);                        //终止程序
  }
  strcpy(name, i_name);         //拷贝姓名到成员变量
  id = i_id;                    //拷贝学号到成员变量
  balance = 0.0;                //初始余额为 0
  cardStatus = UNLOCK;          //卡的状态为解挂
  strcpy(cardID, i_cardID);     //拷贝卡号到成员变量
  strcpy(password, i_password);   //拷贝密码到成员变量
  tableNumber = s_userMax;      //注册表序号
  writeRegisty();               //更新注册表
  s_userMax++;                  //注册表最大容量加 1
}

CStudent::~CStudent()         //析构函数
{
  delete[] name;                //释放指针 name
}

bool CStudent::registerUser(void) //注册用户
{
  if (s_userNum >= s_userMax)   //如果用户数大于最大允许值
  {
    cout << "用户已饱和" << endl;    //打印信息
    return ERROR;               //返回错误
  }

  tableNumber = s_userNum;      //确定用户序号
```

```
  changeInfo();              //修改用户信息
  s_userNum++;               //用户数加 1

  cout << endl << "注册后的账户信息如下 : " << endl;
  showInfo();                //显示注册后的用户信息
  cout << "请务必牢记! " << endl;

  return OK;                 //返回正确
}

bool CStudent::changeInfo(void)   //修改用户信息
{
  if (cardStatus == LOCK)       //如果卡已挂失
  {
    cout << "您的卡已挂失!" << endl;//打印信息
    return ERROR;            //返回错误
  }

  char c = 'y';              //赋值为'y'
  char newString[20];
  int times = 5;             //可以尝试 5 次

  while (c == 'y')              //当 c 的值为'y'时
  {
    cout << "你的名字是 " << name << " , 是否进行修改? (y/n)" << endl; //打印信息
    cin >> c;                //接收用户输入

    while (cin.fail())          //输入引发 cin 异常
    {
      cout << "错误, 请重新输入: ";   //输出错误信息
      cin.clear();              //先清除异常状态
      cin.ignore(999, '\n');             //再清理缓冲区, 清理 999 个字符或者碰到'\n'

      cin >> c;                //再次接受新元素
    }
    cin.ignore(999, '\n');         //再清理缓冲区, 清理 999 个字符或者碰到'\n'

    if (c == 'y')              //如果 c 的值为'y'
    {
      int sizeofname;          //声明变量存放姓名的长度
      char i_name[20];         //声明变量接收用户姓名
      cout << "请输入您的新名字 : ";    //打印信息
      cin >> i_name;              //输入
      sizeofname = strlen(i_name);    //获取姓名的长度
```

```
    delete [] name;              //释放 name 所占的内存空间
    name = new char[sizeofname + 1];    //重新申请一块新的内存空间
    if (name == NULL)               //申请失败
    {
      cout << "申请内存空间失败" << endl;   //打印错误
      exit(0);                      //终止程序
    }
    strcpy(name, i_name);             //拷贝字符串到成员变量
  }
  else
  {
    break;                    //跳出循环
  }
}

c = 'y';           //赋值为'y'，才能进入循环

while (c == 'y')      //循环，直至用户输入'n'
{
  cout << "您的学号为 " << id << " , 是否进行修改？(y/n)" << endl;
  cin >> c;

  while (cin.fail())          //输入引发 cin 异常
  {
    cout << "错误，请重新输入：";   //输出错误信息
    cin.clear();               //先清除异常状态
    cin.ignore(999, '\n');        //再清理缓冲区，清理 999 个字符或者碰到'\n'

    cin >> c;                 //再次接受新元素
  }
  cin.ignore(999, '\n');          //再清理缓冲区，清理 999 个字符或者碰到'\n'

  if (c == 'y')                //用户输入'y'
  {
    cout << "请输入您的新学号 ： ";  //打印信息
    cin >> id;                //接收学号
  }
  else
  {
    break;                  //跳出循环
  }
}

c = 'y';                      //赋值为'y'，才能进入循环
```

```
  while (c == 'y' && times >= 0)        //跳出循环条件：用户输入'n'或者可尝试次数为 0
  {
    cout << "是否要修改您的校园卡密码? (y/n)" << endl;    //打印信息
    cin >> c;                    //接收用户输入

    while (cin.fail())             //输入引发 cin 异常
    {
      cout << "错误，请重新输入: ";   //输出错误信息
      cin.clear();                //先清除异常状态
      cin.ignore(999, '\n');        //再清理缓冲区，清理 999 个字符或者碰到'\n'

      cin >> c;                   //再次接受新元素
    }
    cin.ignore(999, '\n');          //再清理缓冲区，清理 999 个字符或者碰到'\n'

    if (c == 'y')                 //如果用户输入'y'
    {
      cout << "请输入您的原始密码 : " << endl;
      cout << "提示：如果这是您第一次修改密码，则原始密码为 000000" << endl;
      cin >> newString;           //接收用户输入
      bool match = strcmp(password, newString);   //与密码进行匹配
      if (match != true)           //匹配成功
      {
        cout << "请输入您的新密码 : ";
        cin >> password;          //输入新的密码
        writeRegisty();           //这里需要修改一下
      }
      else
      {
        cout << "密码错误，您只剩下 " << --times << " 次尝试的机会了！" << endl;
      }
    }
  }
  writeRegisty();   //更新注册表

  return OK;
}

int CStudent::login(char *i_cardID, char *i_password) //登录
{
  int match = cmpPassword(i_cardID, i_password);    //匹配密码

  if (match > LOGIN_FAILED)   //匹配成功
  {
```

```
    return match;            //返回匹配到的序号
  }
  else
  {
    return LOGIN_FAILED;     //返回失败
  }
}

int CStudent::searchAccount(char *i_cardID)      //查询账户
{
  for (int i = 0; i < s_userNum; i++)             //注册表中逐项查找
  {
    if (!strcmp(i_cardID, s_registry[i].l_cardID))    //如果找到匹配的
    {
      return i;     //返回序号
    }
  }
  cout << "未注册的用户！" << endl;    //为匹配到，打印信息

  return LOGIN_FAILED;         //返回失败
}

int CStudent::cmpPassword(char *i_cardID, char *i_cardPassword) //匹配密码
{
  int i = searchAccount(i_cardID);    //查询账户

  if (i <= LOGIN_FAILED)          //如果账户不存在
  {
    return LOGIN_FAILED;       //返回失败
  }

  if (!strcmp(i_cardPassword, s_registry[i].l_cardPassword))    //比较密码是否相匹配
  {
    cout << "成功!" << endl;      //打印信息
    return i;              //返回序号
  }
  else
  {
    cout << "密码错误!" << endl;  //密码错误
    return LOGIN_FAILED;    //返回失败
  }
}

inline void CStudent::showInfo()      //打印用户信息
```

```
{
  cout << "**********************" << endl;
  cout << "*  姓名: " << name << endl << "*  学号: " << id << endl;
  cout << "*  余额: " << balance << endl ;
  cout << "*  卡号: " << cardID << endl;
  cout << "**********************" << endl;
}

inline void CStudent::writeRegisty(void)  //更新注册表
{
  strcpy(s_registry[tableNumber].l_cardID, cardID);
  strcpy(s_registry[tableNumber].l_cardPassword, password);
}

bool CStudent::cardRecharge(float amount)
{
  if (cardStatus == LOCK)
  {
    cout << "您的卡已挂失!" << endl;
    return ERROR;
  }

  char c;
  cout << "确定要充值 " << amount << " 元?(y/n) " << endl;
  cin >> c;

  while (cin.fail())          //输入引发 cin 异常
  {
    cout << "错误，请重新输入: ";   //输出错误信息
    cin.clear();               //先清除异常状态
    cin.ignore(999, '\n');        //再清理缓冲区，清理 999 个字符或者碰到'\n'

    cin >> c;                  //再次接受新元素
  }
  cin.ignore(999, '\n');          //再清理缓冲区，清理 999 个字符或者碰到'\n'

  if (c == 'y')
  {
    balance += amount;
    cout << "充值成功!" << endl;
    cout << "您当前的余额为 " << balance << " 元." << endl;
    return OK;
  }
  else
```

```
  {
    cout << "放弃充值!" << endl;
    return ERROR;
  }
}

bool CStudent::eating(int choice)     //用餐
{
  char c;               //声明字符变量

  if (cardStatus == LOCK)   //若卡已挂失
  {
    cout << "您的卡已挂失!" << endl;  //打印信息
    return ERROR;         //返回错误
  }

  cout << "您选择 ";
  cout << foodList[choice].food << endl;  //输出用户选择的菜名
  cout << "确定? (y/n)" << endl;         //打印
  cin >> c;                       //由用户确定

  while (cin.fail())              //输入引发 cin 异常
  {
    cout << "错误，请重新输入: ";   //输出错误信息
    cin.clear();                //先清除异常状态
    cin.ignore(999, '\n');        //再清理缓冲区，清理 999 个字符或者碰到'\n'

    cin >> c;                 //再次接受新元素
  }
  cin.ignore(999, '\n');          //再清理缓冲区，清理 999 个字符或者碰到'\n'

  if (c == 'y')                 //用户确定
  {
    if (foodList[choice].price == 0)  //如果选择退出
    {
      cout << "退出" << endl;
      return ERROR;           //返回错误
    }
    if (balance < foodList[choice].price) //如果余额不足
    {
      cout << "余额不足，请充值! " << endl; //打印结果
    }
    else
    {
```

```
      balance -= foodList[choice].price;    //扣除余额
      //打印消费信息
      cout << "您购买了 " << foodList[choice].food << " , 花费 " << foodList[choice].price
<< " 元" << endl;
      cout << "您的余额还剩 " << balance << " 元." << endl;
    }
    return OK;  //返回成功
  }
  else
  {
    cout << "已放弃 " << foodList[choice].food << " ." << endl;
    return OK;  //返回成功
  }
}

bool CStudent::lockCard()
{
  char newString[20];

  cout << "请输入密码 : ";
  cin >> newString;            //输入密码

  while (cin.fail())             //输入引发 cin 异常
  {
    cout << "错误，请重新输入: ";   //输出错误信息
    cin.clear();               //先清除异常状态
    cin.ignore(999, '\n');        //再清理缓冲区，清理 999 个字符或者碰到'\n'

    cin >> newString;          //再次接受新元素
  }
  cin.ignore(999, '\n');          //再清理缓冲区，清理 999 个字符或者碰到'\n'

  int match = strcmp(password, newString);    //比较密码

  if (match == 0)              //密码正确
  {
    cardStatus = LOCK;          //挂失
    cout << "您的卡已挂失.  " << endl;
    return OK;
  }
  else
  {
    cout << "密码错误!" << endl;
    return ERROR;
```

```
  }
}

bool CStudent::unlockCard()
{
  char newString[20];          //声明字符串数组

  cout << "请输入密码 : ";
  cin >> newString;            //输入密码

  while (cin.fail())              //输入引发 cin 异常
  {
    cout << "错误，请重新输入: ";   //输出错误信息
    cin.clear();                //先清除异常状态
    cin.ignore(999, '\n');        //再清理缓冲区，清理 999 个字符或者碰到'\n'

    cin >> newString;           //再次接受新元素
  }
  cin.ignore(999, '\n');         //再清理缓冲区，清理 999 个字符或者碰到'\n'

  int match = strcmp(password, newString);  //比较密码

  if (match == 0)              //密码正确
  {
    cardStatus = UNLOCK;        //解挂
    cout << "您的卡已解挂!" << endl;
    return OK;
  }
  else
  {
    cout << "密码错误!" << endl;
    return ERROR;
  }
}

bool CStudent::getcardStatus(void)
{
  return cardStatus;    //返回卡状态
}

int   CStudent::getUserNum(void)
{
  return s_userNum;   //返回当前用户数目
}
```

```
bool randString(char starting_c, char ending_c, char *i_string, int s_length)
{
  char c;
  int i;

  if (ending_c <= starting_c)
  {
    cout << "发生错误!" << endl;
    return ERROR;
  }

  for (i = 0; i < s_length; i++)
  {
    //随机生成一个字符
    c = (char)(((double)rand() / RAND_MAX)*(ending_c - starting_c) + starting_c);
    i_string[i] = c;
  }

  i_string[i] = '\0'; //结尾加上结束符

  return OK;
}
```

在 service.h 中编写函数声明。

```
//service.h

#ifndef _SERVICE_H_
#define _SERVICE_H_

#include "student.h"

typedef enum {
  REGISTER,     //注册
  LOGIN,        //登录
  DINE,         //用餐
  LOGOUT,       //注销
  RECHARGE,     //充值
  LOCK_UNLOCK,  //挂失&解挂
  SEARCH,       //查询
  CHANGE_INFO,  //更改信息
  LEAVE,         //离开
  SERVICE_MAX //枚举最大值
```

```
}ServiceList;

typedef enum {
  RICE,       //米饭
  TARO,       //芋头
  EXIT,       //退出
  MENU_LIST //枚举最大值
}MenuList;

int showMenu(void);     //显示菜单
int showService(void);    //显示服务
void processChoice(int choice, CStudent *&object);  //处理用户选项

#endif // !_SERVICE_H_
```

在 service.cpp 中编写服务菜单和食堂菜单，并显示服务菜单中的各种操作。

```
//service.cpp

#include "service.h"
#include <iostream>

using namespace std;

CStudent *stu = new CStudent[10]; //生成 CStudent 类型的对象数组

int showMenu(void)      //显示食堂菜单
{
  int choice;         //变量用于获取用户选择

  cout << "------------------------- MENU --------------------------" << endl << endl;
  cout << "  ***************" << endl;
  cout << "  |本餐厅提供:  |" << endl;
  cout << "  |1.米饭 2.5/两|" << endl;
  cout << "  |2.芋头 3.0/个|" << endl;
  cout << "  |3.退出       |" << endl;
  cout << "  ***************" << endl << endl;
  cout << "请选择---";

  cin >> choice;
  choice--;
  while (choice >= MENU_LIST)       //当用户输入的数值超出范围时
```

```
  {
    cout << "错误，请重新输入!" << endl;  //提示错误
    cin >> choice;                  //重新输入
  }

  cout << endl << endl;
  return choice;    //返回用户的选择
}

int showService(void)   //显示服务菜单
{
  int choice;         //变量用于获取用户选择

  cout << "-------------------------- SERVICE --------------------------" << endl;
  cout << "请选择服务：" << endl;
  cout << "1.注册" << endl;
  cout << "2.登录" << endl;
  cout << "3.用餐" << endl;
  cout << "4.注销" << endl;
  cout << "5.充值" << endl;
  cout << "6.挂失/解挂" << endl;
  cout << "7.查询" << endl;
  cout << "8.修改信息" << endl;
  cout << "9.退出" << endl;
  cout << "请选择---";

  cin >> choice;
  choice--;
  while (choice >= SERVICE_MAX)       //当用户输入的数值超出范围时
  {
    cout << "错误，请重新输入!" << endl;    //提示错误
    cin >> choice;                    //重新输入
  }

  cout << "-------------------------- FINISH --------------------------" << endl << endl
<< endl;

  return choice;    //返回用户的选择
}

void processChoice(int choice, CStudent *&object)
{
  static bool s_loaded = ERROR;   //标记用户是否登录
  bool result = ERROR;          //接收各类返回结果
```

```
  char c;                    //接收用户的输入
  int getChoice;               //接收用户在食堂菜单的选择
  int i;                     //登录用户的序号

  switch (choice)
  {
  case REGISTER:
    cout << "-------------------------- REGISTER -------------------------" << endl;
    if (s_loaded == OK)
    {
      cout << "请先退出登录!" << endl;
      cout << "-------------------------- FINISH -------------------------" << endl <<
endl << endl;
      break;
    }
    int num;
    num = CStudent::getUserNum();
    result = stu[num].registerUser();
    if (result == ERROR)
    {
      cout << "出错" << endl;
    }
    cout << "-------------------------- FINISH -------------------------" << endl <<
endl << endl;
    break;
  case LOGIN:
    cout << "-------------------------- LOGIN -------------------------" << endl;
    if (s_loaded == OK)
    {
      cout << "您已经登录，请先注销登录!" << endl;
      cout << "-------------------------- FINISH -------------------------" << endl <<
endl << endl;
      break;
    }

    char getcardID[10];
    char getpassword[10];

    cout << "提示：如果您还未拥有账号，请先退出，注册账号!" << endl;
    cout << "请输入您的卡号 (六位数字) : ";
    cin >> getcardID;
    cout << "请输入您的密码 (六位数字) : ";
    cin >> getpassword;
    i = CStudent::login(getcardID, getpassword);
```

```
    if (i > LOGIN_FAILED)
    {
      cout << "登录成功!" << endl;
      s_loaded = OK;
      object = &stu[i];
    }
    else
    {
      cout << "登录失败，请重新尝试" << endl;
      object = NULL;
      s_loaded = ERROR;
    }
    cout << "-------------------------- FINISH --------------------------" << endl <<
endl << endl;

    break;
  case DINE:
    cout << "-------------------------- DINE --------------------------" << endl;
    if (s_loaded == ERROR)
    {
      cout << "请先登录，谢谢!" << endl;
      cout << "-------------------------- FINISH --------------------------" << endl <<
endl << endl;
      break;
    }
    result = OK;
    while (result)
    {
      getChoice = showMenu();
      result = object->eating(getChoice);
    }
    cout << "-------------------------- FINISH --------------------------" << endl <<
endl << endl;
    break;
  case LOGOUT:
    cout << "-------------------------- LOGOUT --------------------------" << endl;
    if (s_loaded == ERROR)
    {
      cout << "您还未登录账号!" << endl;
      cout << "-------------------------- FINISH --------------------------" << endl <<
endl << endl;
      break;
    }
```

```
    cout << "您确定要注销账号吗？ (y/n) ";
    cin >> c;
    while (cin.fail())          //输入引发 cin 异常
    {
      cout << "错误，请重新输入： ";  //输出错误信息
      cin.clear();                          //先清除异常状态
      cin.ignore(999, '\n');                 //再清理缓冲区，清理 999 个字符或者碰到'\n'

      cin >> c;                //再次接受新元素
    }
    cin.ignore(999, '\n');                 //再清理缓冲区，清理 999 个字符或者碰到'\n'

    if (c == 'y')
    {
      cout << "已注销." << endl;
      s_loaded = ERROR;
      object = NULL;
    }
    else
    {
      cout << "您已放弃注销账号!" << endl;
    }
    cout << "------------------------- FINISH -------------------------" << endl <<
endl << endl;
    break;
  case RECHARGE:
    cout << "------------------------- RECHAGER -------------------------" << endl;
    if (s_loaded == ERROR)
    {
      cout << "请先登录，谢谢!" << endl;
      cout << "------------------------- FINISH -------------------------" << endl <<
endl << endl;
      break;
    }
    float money;
    cout << "请输入您的充值金额？ ";
    cin >> money;
    object->cardRecharge(money);
    cout << "------------------------- FINISH -------------------------" << endl <<
endl << endl;
    break;
  case LOCK_UNLOCK:
    cout << "------------------------- LOCK_UNLOCK -------------------------" << endl;
```

```
    char c;
    char getString[10];
    cout << "请输入您的卡号 : ";
    cin >> getString;

    i = CStudent::searchAccount(getString);

    if (i == LOGIN_FAILED)
    {
      break;
      cout << "------------------------- FINISH -------------------------" << endl <<
endl << endl;
    }

    object = &stu[i];
    result = object->getcardStatus();
    if (result == LOCK)
    {
      cout << "您的卡处于挂失，是否解挂？(y/n) ";
    }
    else
    {
      cout << "您的卡处于解挂状态，是否挂失？(y/n) ";
    }

    cin >> c;

    while (cin.fail())          //输入引发 cin 异常
    {
      cout << "错误，请重新输入: ";  //输出错误信息
      cin.clear();                          //先清除异常状态
      cin.ignore(999, '\n');                 //再清理缓冲区，清理 999 个字符或者碰到'\n'

      cin >> c;                //再次接受新元素
    }
    cin.ignore(999, '\n');                //再清理缓冲区，清理 999 个字符或者碰到'\n'

    if (c == 'y' && result == LOCK)
    {
      object->unlockCard();
    }
    else if (c == 'y' && result == UNLOCK)
    {
      object->lockCard();
```

```
    }
    else
    {
      cout << "退出" << endl;
    }
    cout << "-------------------------- FINISH --------------------------" << endl <<
endl << endl;
    break;
  case SEARCH:
    cout << "-------------------------- SEARCH --------------------------" << endl;
    if (s_loaded == ERROR)
    {
      cout << "您尚未登录!" << endl;
      cout << "-------------------------- FINISH --------------------------" << endl <<
endl << endl;
      break;
    }
    object->showInfo();
    cout << "-------------------------- FINISH --------------------------" << endl <<
endl << endl;

    break;
  case CHANGE_INFO:
    cout << "-------------------------- CHANGE_INFO --------------------------" << endl;

    if (s_loaded == ERROR)
    {
      cout << "您尚未登录!" << endl;
      cout << "-------------------------- FINISH --------------------------" << endl <<
endl << endl;
      break;
    }
    object->changeInfo();
    cout << "-------------------------- FINISH --------------------------" << endl <<
endl << endl;
    break;
  case LEAVE:
    cout << "-------------------------- LEAVE --------------------------" << endl;
    cout << "您确定要退出吗?(y/n)" << endl;
    cin >> c;
    while (cin.fail())          //输入引发 cin 异常
    {
      cout << "错误，请重新输入: ";  //输出错误信息
      cin.clear();                          //先清除异常状态
```

```
      cin.ignore(999, '\n');                  //再清理缓冲区，清理 999 个字符或者碰到'\n'

      cin >> c;                 //再次接受新元素
    }
    cin.ignore(999, '\n');                  //再清理缓冲区，清理 999 个字符或者碰到'\n'

    if (c == 'y')
    {
      cout << "-------------------------- FINISH --------------------------" << endl <<
endl << endl;
      exit(0);
    }
    else
    {
      cout << "-------------------------- FINISH --------------------------" << endl <<
endl << endl;
      break;
    }
  default:
    break;
  }
}
```

实验结果与分析：

在 CStudent 类中声明了一个注册表用于注册用户信息，在以后登录的时候我们可以使用注册表来对照用户输入的账户和密码是否匹配，注册表是类私有的，外部不能进行访问，只能调用函数来检测账户和密码是否匹配。对象的属性包括了用户名、学号、余额、校园卡号、密码、卡状态（挂失或者解挂）和对应的注册表序号（该注册表序号可以使得注册表查找更加方便，在用户已登录状态需要密码时，可以直接在注册表找到定位）。这里的用户名直接使用了字符串指针而不是数组，可以在获得用户名的情况下根据用户名的长度动态分配内存而不是使用一段固定不变的内存单元，从而可以避免内存浪费。在需要分配内存的时候，使用语句

```
sizeofname = strlen(用户名字符串);      //获取姓名的长度
name = new char[sizeofname + 1];    //申请内存空间
```

再使用 strcpy 函数将用户名的内存复制给 name。在无参构造函数值中，调用了一个自己编写的生成随机字符串数组的函数 randString，该函数的形参列表是字符串所要生成字符范围的最小值、字符串所要生成字符范围的最大值、存放结果的字符串指针、生成的字符串的长度。调用这个函数我们可以生成一个内容全为阿拉伯数字的 10 位字符串，符合校卡卡号的标准。

CStudent 类中包含两个私有的函数分别是核对密码函数 cmpPassword 和写注册表函数 writeRegisty。每一次用户修改了账户或者密码之后都应该重新调用一次写注册表函数，该函数中使用到了注册表序号这个变量。

CStudent 类中提供的函数接口有查询用户是否注册函数 searchAccount、用户登录函数 login、获取已注册用户数目函数 getUserNum、注册用户函数 registerUser、显示用户信息函数 showInfo、更改用户信息函数 changeInfo、充值函数 cardRecharge、用餐函数 eating、挂失函数 lockCard、解挂函数 unlockCard 和获取卡状态函数 getcardStatus。通过调用这些函数可以实现校园卡的功能。

在 service.h 文件中声明了食堂菜单 showMenu、服务菜单 showService 两种显示函数，还有一个处理用户的选择的函数 processChoice。服务菜单中列出了如图 1 所示界面，用户输入编号后，函数可以接收编号并返回编号。返回来的编号可以作为实参传入处理用户选择函数 processChoice 中，processChoice 含有 switch…case…语句来处理各种结果。在用餐下调用了食堂菜单。如果用户选择退出并最终确定，则会调用 exit(0)来结束整个程序。值得注意的时，processChoice 函数中的第二个形参应该写成 CStudent 指针的引用，否则，每一次在 processChoice 函数中改变了该指针指向之后，跳出函数并不能生效，还是

在 main.c 中，声明一个 CStudent 类型的指针，然后在 while 死循环里面调用 showService 和 processChoice 函数，不断地接收服务，处理服务内容，知道用户选择退出，直接结束整个程序。

下图中，图 1 是服务菜单的界面，在服务菜单下输入 1 后回车进入注册界面，如图 2 所示。注册后默认的名字为“默认”，程序将询问用户是否修改用户名，输入“y”选择修改，输入“n”放弃，其他也类似。各个服务的界面图片都如下各图所示。

```
-------------------------- SERVICE --------------------------
请选择服务：
1.注册
2.登录
3.用餐
4.注销
5.充值
6.挂失/解挂
7.查询
8.修改信息
9.退出
请选择---
```

图 1 - 初始界面

```
-------------------------- REGISTER --------------------------
你的名字是 默认 , 是否进行修改? (y/n)
y
请输入您的新名字 : 小明
你的名字是 小明 , 是否进行修改? (y/n)
n
您的学号为 2016222041 , 是否进行修改? (y/n)
y
请输入您的新学号 : 2016222042
您的学号为 2016222042 , 是否进行修改? (y/n)
n
是否要修改您的校园卡密码? (y/n)
y
请输入您的原始密码 :
提示：如果这是您第一次修改密码，则原始密码为000000
000000
请输入您的新密码 : 543432
是否要修改您的校园卡密码? (y/n)
n

注册后的账户信息如下 :
**********************
*  姓名: 小明
*  学号: 2016222042
*  余额: 0
*  卡号: 517543
**********************
请务必牢记!
-------------------------- FINISH --------------------------
```

图 2 - 注册

```
-------------------------- LOGIN --------------------------
提示：如果您还未拥有账号，请先退出，注册账号!
请输入您的卡号 (六位数字) : 517543
请输入您的密码 (六位数字) : 543432
成功!
登录成功!
-------------------------- FINISH --------------------------
```

图 3 - 登录

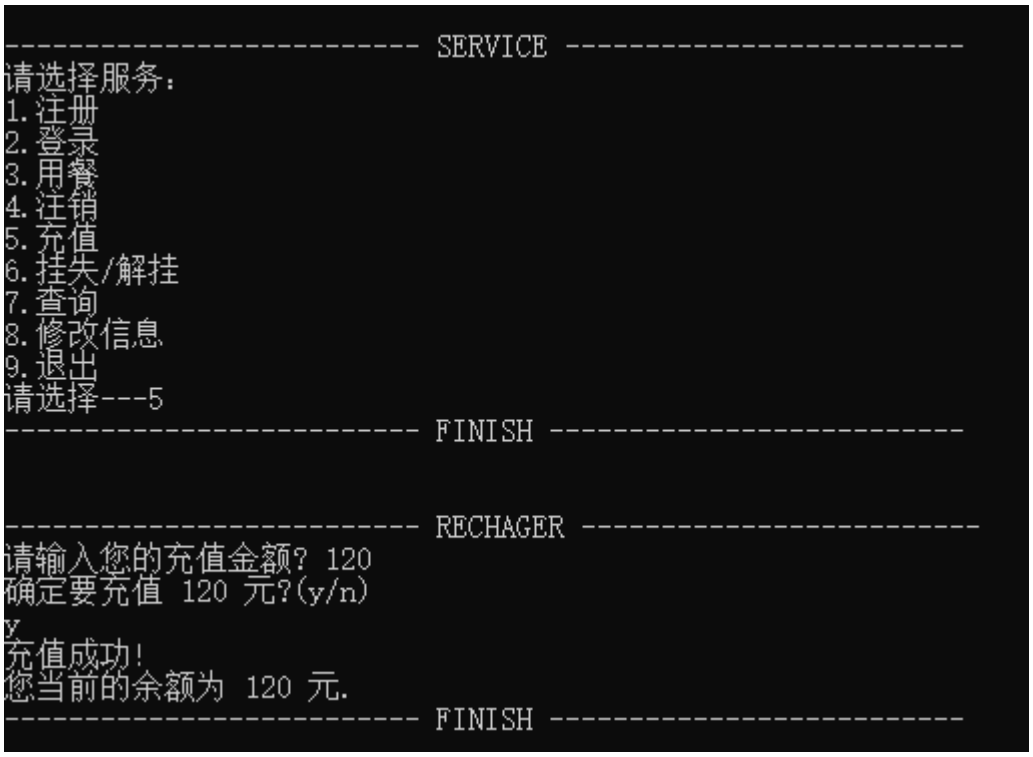

图 4 - 充值

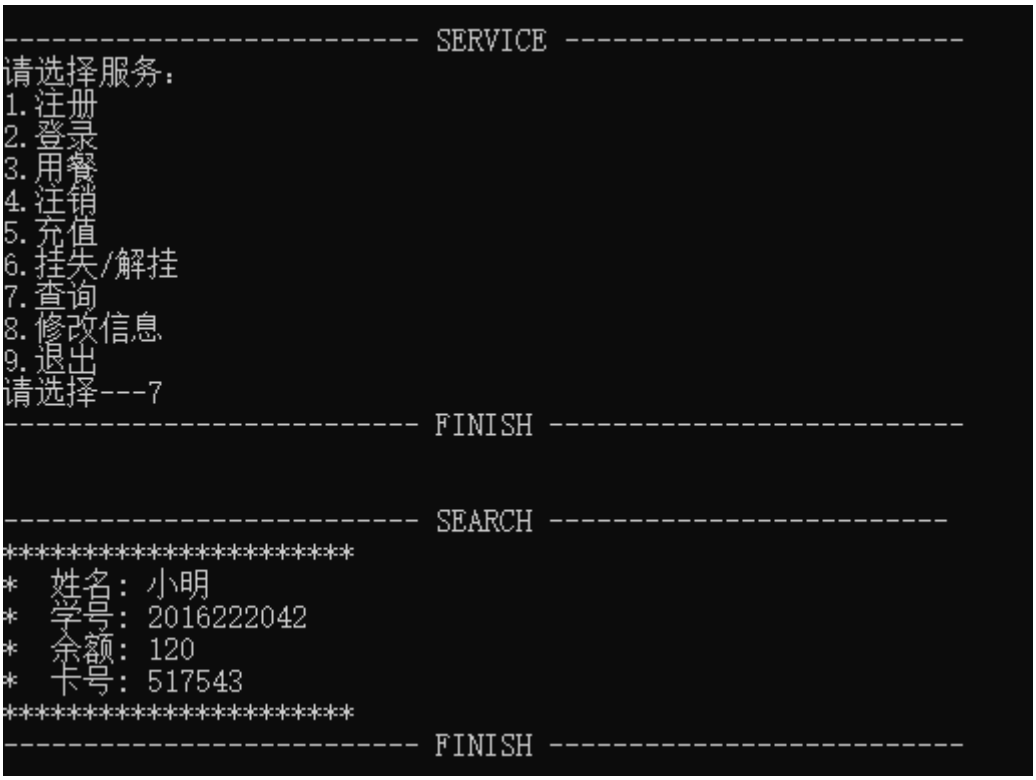

图 5 - 查询


SERVICE
请选择服务:
1.注册
2.登录
3.用餐
4.注销
5.充值
6.挂失/解挂
7.查询
8.修改信息
9.退出
请选择---3
FINISH
DINE
MENU
本餐厅提供:
1.米饭 2.5/两
2.芋头 3.0/个
3.退出
请选择---1
您选择 一两米饭
确定?(y/n)
y
您购买了 一两米饭 , 花费 2.5 元
您的余额还剩 117.5 元.
MENU
本餐厅提供:
1.米饭 2.5/两
2.芋头 3.0/个
3.退出
请选择---

图 6 - 用餐

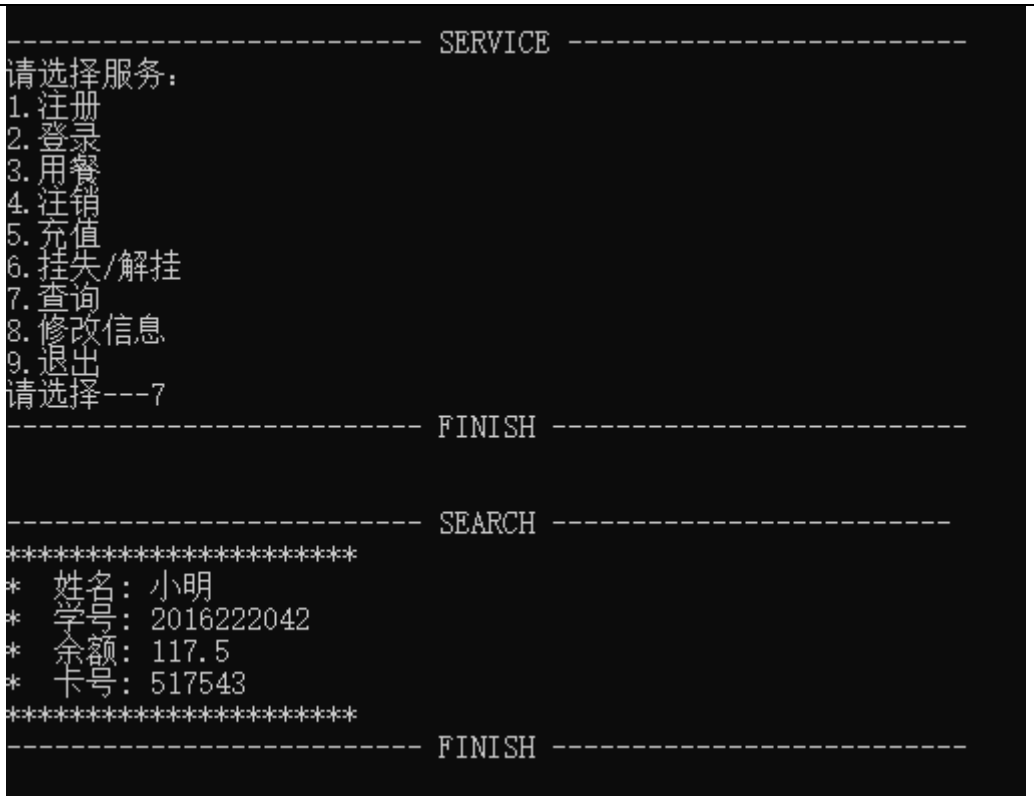
---------- SERVICE ----------
请选择服务：
1.注册
2.登录
3.用餐
4.注销
5.充值
6.挂失/解挂
7.查询
8.修改信息
9.退出
请选择---7
---------- FINISH ----------
---------- SEARCH ----------
****
* 姓名：小明
* 学号：2016222042
* 余额：117.5
* 卡号：517543
****
---------- FINISH ----------

图 7 - 餐后余额

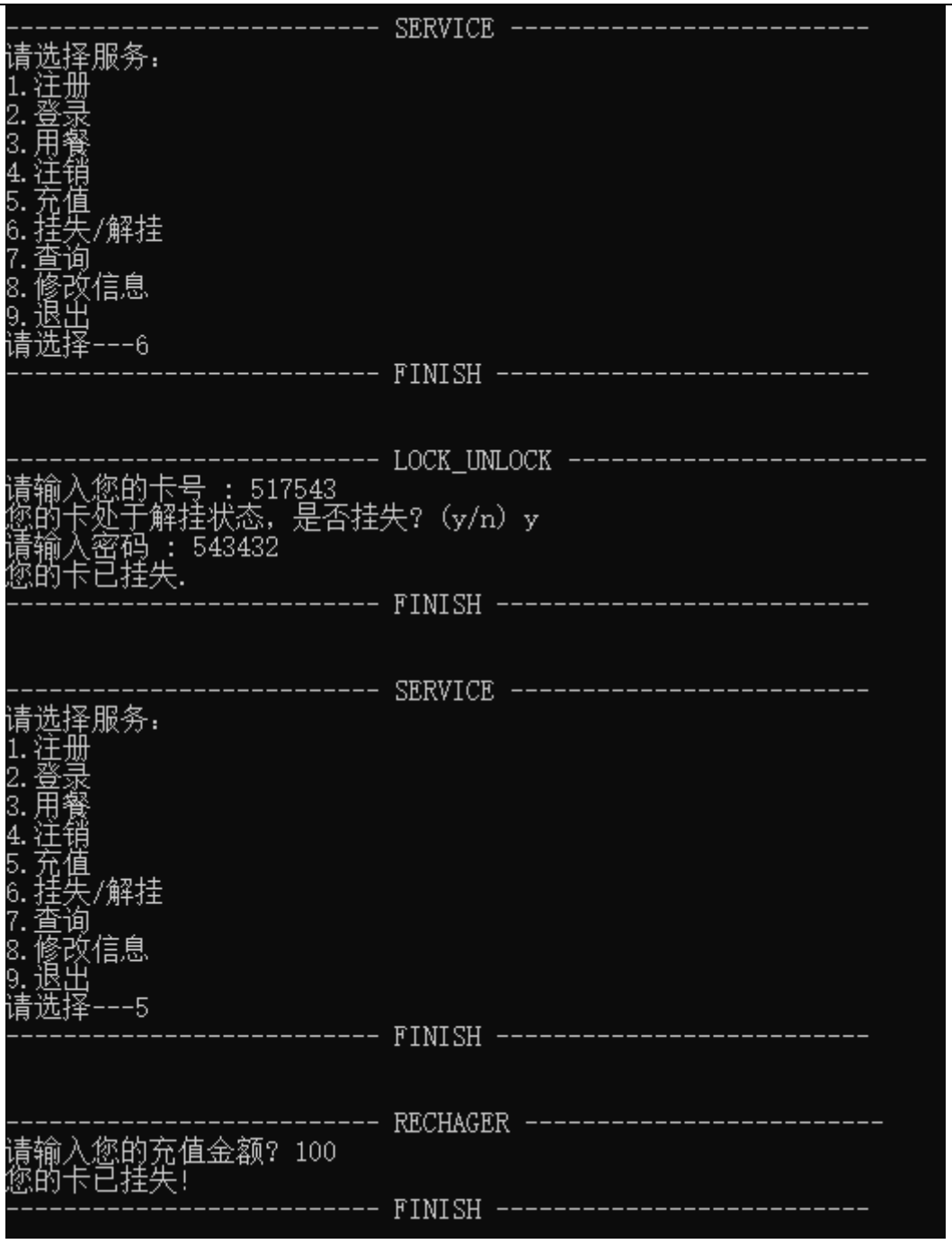
---------- SERVICE ----------
请选择服务：
1.注册
2.登录
3.用餐
4.注销
5.充值
6.挂失/解挂
7.查询
8.修改信息
9.退出
请选择---6
---------- FINISH ----------
---------- LOCK_UNLOCK ----------
请输入您的卡号 ： 517543
您的卡处于解挂状态，是否挂失？（y/n） y
请输入密码 ： 543432
您的卡已挂失.
---------- FINISH ----------
---------- SERVICE ----------
请选择服务：
1.注册
2.登录
3.用餐
4.注销
5.充值
6.挂失/解挂
7.查询
8.修改信息
9.退出
请选择---5
---------- FINISH ----------
---------- RECHAGER ----------
请输入您的充值金额？ 100
您的卡已挂失!
---------- FINISH ----------

图 8 - 挂失

```
-------------------------- SERVICE --------------------------
请选择服务：
1.注册
2.登录
3.用餐
4.注销
5.充值
6.挂失/解挂
7.查询
8.修改信息
9.退出
请选择---6
-------------------------- FINISH --------------------------


-------------------------- LOCK_UNLOCK --------------------------
请输入您的卡号 : 517543
您的卡处于挂失，是否解挂？(y/n) y
请输入密码 : 543432
您的卡已解挂!
-------------------------- FINISH --------------------------
```

图 9 - 解挂

```
-------------------------- SERVICE --------------------------
请选择服务：
1.注册
2.登录
3.用餐
4.注销
5.充值
6.挂失/解挂
7.查询
8.修改信息
9.退出
请选择---4
-------------------------- FINISH --------------------------


-------------------------- LOGOUT --------------------------
您确定要注销账号吗？(y/n) y
已注销.
-------------------------- FINISH --------------------------
```

图 10 - 注销

```
-------------------------- LOGOUT --------------------------
您确定要注销账号吗？（y/n）y
已注销.
-------------------------- FINISH --------------------------


-------------------------- SERVICE --------------------------
请选择服务：
1.注册
2.登录
3.用餐
4.注销
5.充值
6.挂失/解挂
7.查询
8.修改信息
9.退出
请选择---5
-------------------------- FINISH --------------------------


-------------------------- RECHAGER --------------------------
请先登录，谢谢!
-------------------------- FINISH --------------------------
```

图 11 - 注销后尝试进行操作

```
-------------------------- SERVICE --------------------------
请选择服务：
1.注册
2.登录
3.用餐
4.注销
5.充值
6.挂失/解挂
7.查询
8.修改信息
9.退出
请选择---9
-------------------------- FINISH --------------------------


-------------------------- LEAVE --------------------------
您确定要退出吗?(y/n)
y
-------------------------- FINISH --------------------------
```

图 12 - 退出服务

指导教师批阅意见：

成绩评定：

指导教师签字：

年　　月　　日

备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。