



|  |  |  |  |
| --- | --- | --- | --- |
| **学号** | **19030500024** | **姓名** | **马子豪** |
| **班级** | **1903058** | **任课老师** | **褚华** |
| **实验名称** | **校园一卡通乘车模拟系统** | | |
| **实验学期** | **2020-2021学年第二学期** | | |
| **实验日期** | **2021.5.29** | **实验地点** | **G楼314** |
| **报告成绩** |  | **评分教师** |  |

西安电子科技大学计算机科学与技术学院

# 一、实验目的

采用面向对象程序设计范型、C++语言实现一个完整的简化系统——校园一卡通乘车模拟系统。通过编写该系统熟悉掌握面向对象的编程思想。

# 二、实验环境

* **硬件环境：**

Lenovo Legion Y7000P

处理器：Intel(R) Core(TM) i5-9300H CPU @2.40GHz

Windows 10 64位操作系统

* **集成开发环境**：

Visual Studio 2019

* **数据存储：**

MySQL Ver 8.0.24

* **代码版本管理：**

[Github](https://github.com/ZiHao256/Bus-Card-System)

# 三、实验内容

1. **系统增加对三种一卡通类型的管理**，每种一卡通所有的属性不尽相同。在一卡通持有者刷卡乘车时，根据所持有的一卡通种类的不同，对每张一卡通的属性值（余额、乘车次数等）进行相对应的更新。

2. **系统增加对班车信息的管理**，每辆班车都有各自的基本信息。根据班车的基本信息编制运行时刻表。乘客刷卡上车时，根据班车的载乘人数进行相应的操作。

3. **系统增加对一卡通持卡人信息的管理**。增加对每个人乘车的记录，和对一卡通的业务操作：申请一卡通、注销一卡通、挂失一卡通等。

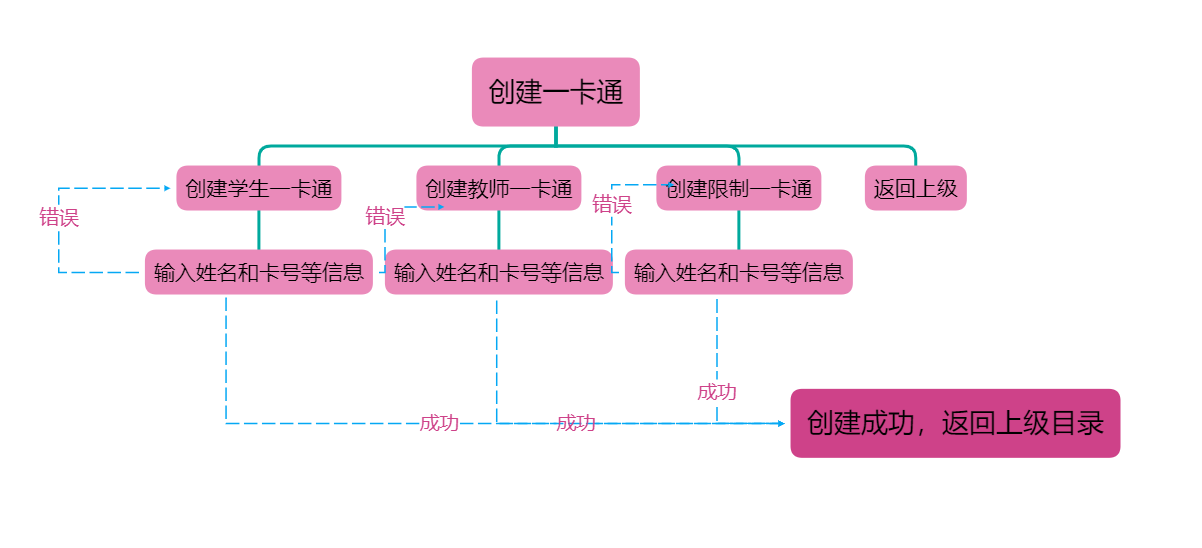
4. **从相应文件中获取信息并完成对班车的仿真**。并设置中途停车点。

# 四、实验步骤

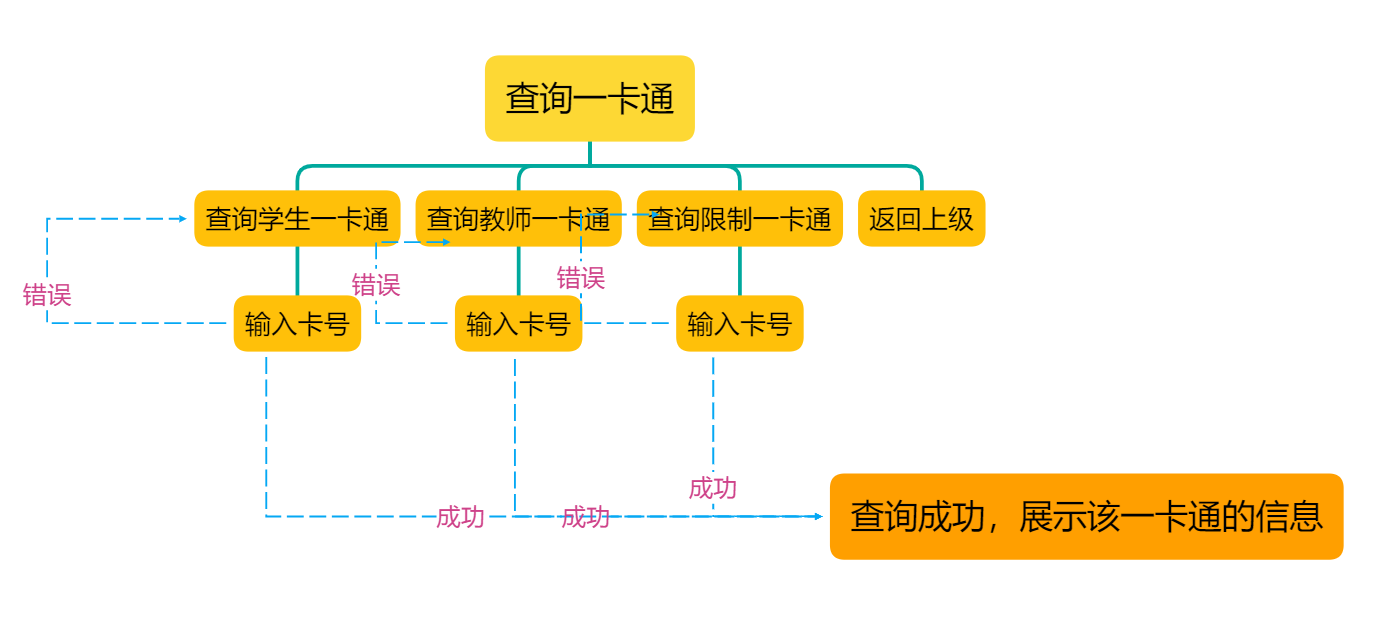
## 4.1 实验思路

### 4.1.1 系统整体结构设计

### 4.1.2 创建一卡通功能逻辑设计

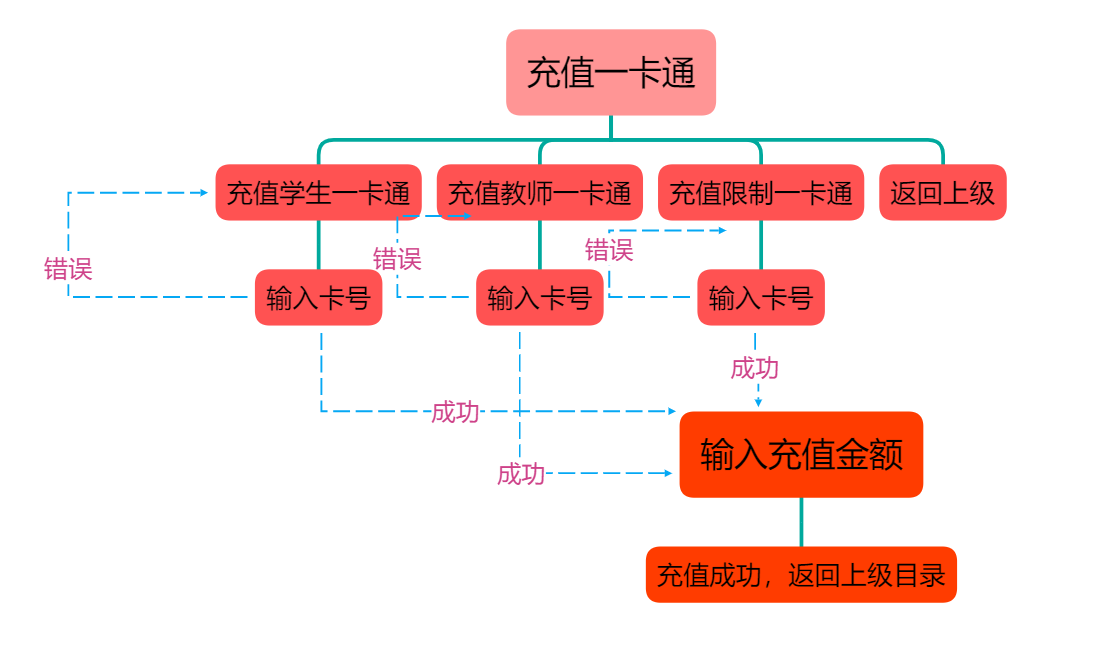


### 4.1.3 查询一卡通功能逻辑设计

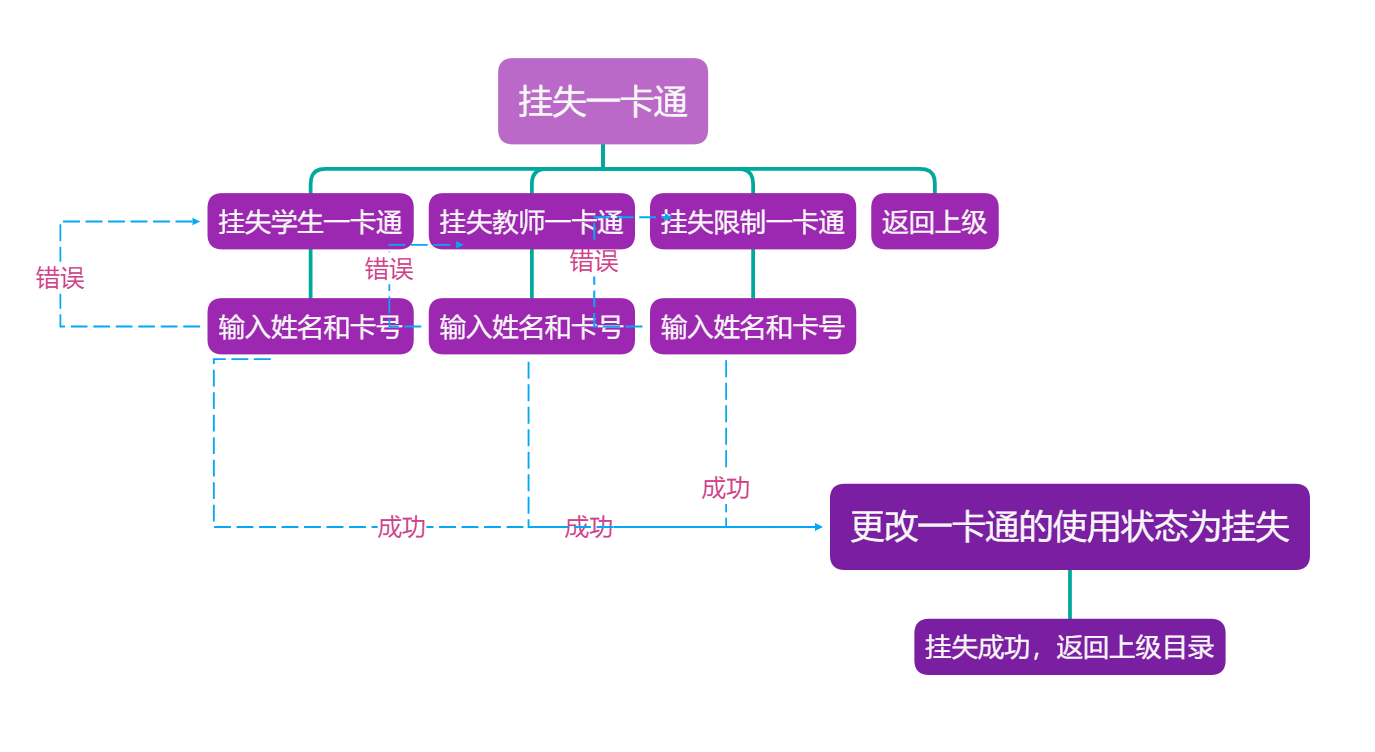


### 4.1.4 操作一卡通功能逻辑设计

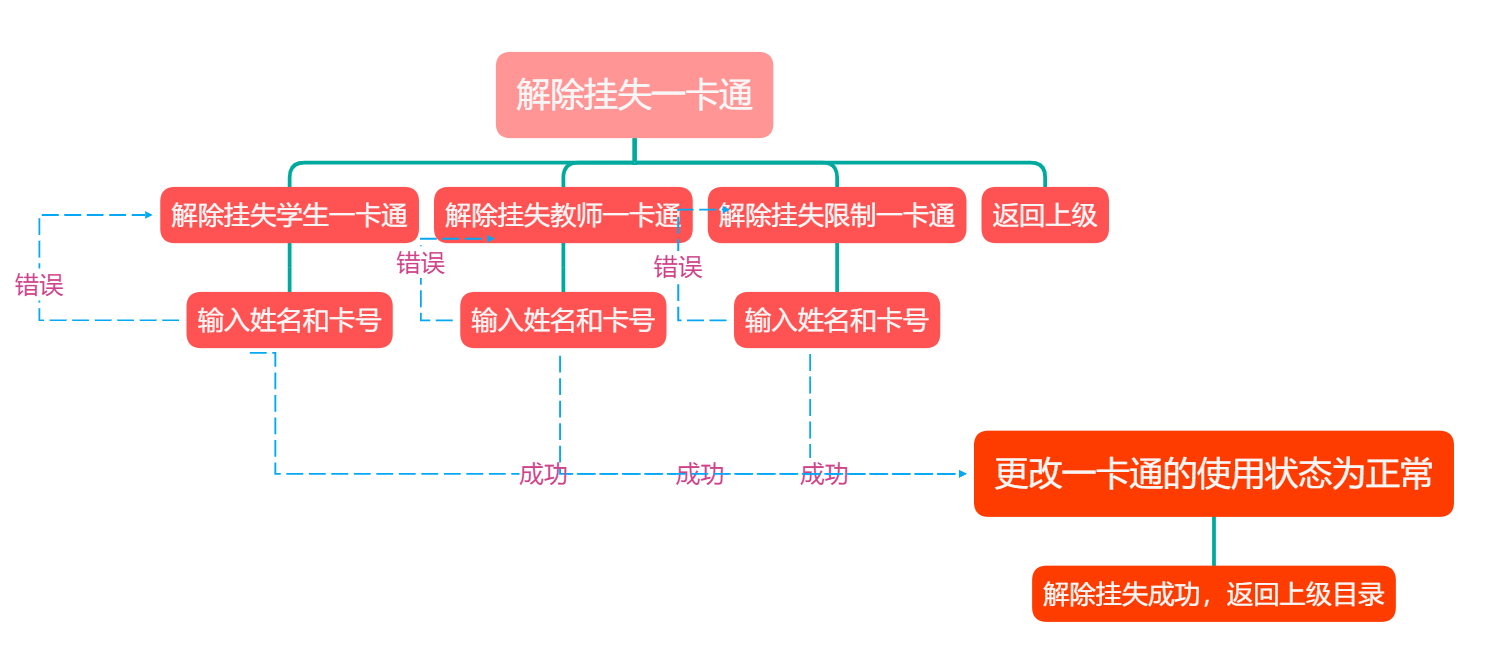
#### 4.1.4.1 充值一卡通功能逻辑设计



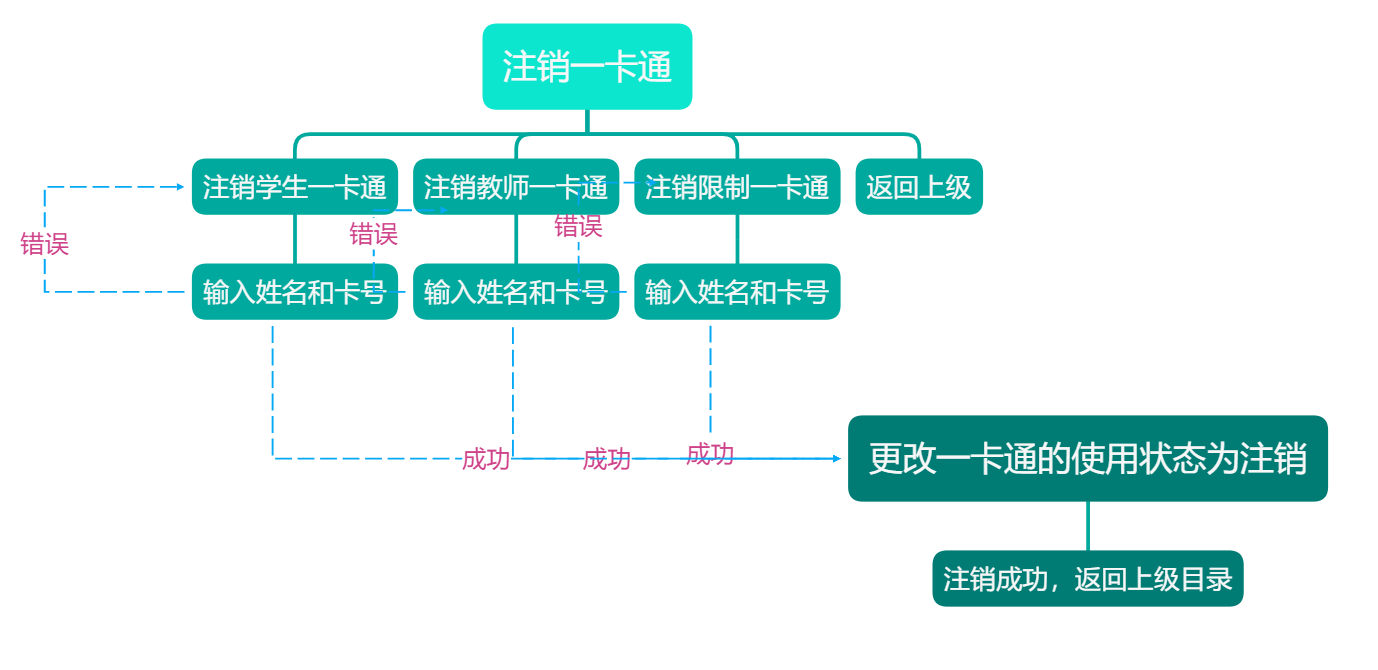
#### 4.1.4.2 挂失一卡通功能逻辑设计



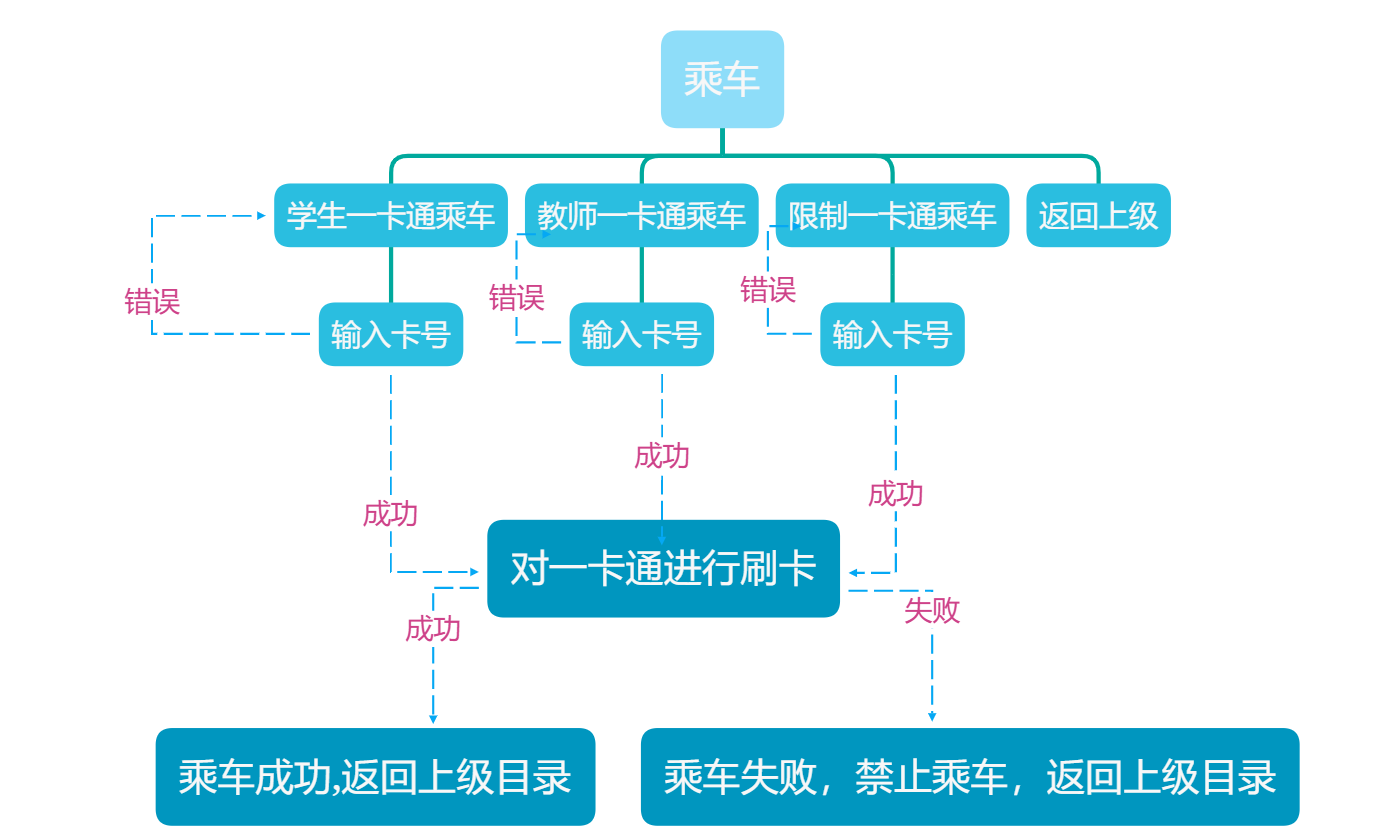
#### 4.1.4.3 解除挂失一卡通功能逻辑设计



#### 4.1.4.4 注销一卡通功能逻辑设计



### 4.1.5 乘车功能逻辑设计



## 4.2 关键数据结构的定义

该校园一卡通乘车模拟系统编写了五个类来实现。

### 4.2.1 class Student 学生一卡通

Student类有8个数据成员和8个成员函数。

|  |  |  |  |
| --- | --- | --- | --- |
| 分类 | 名称 | 类型 | 含义 |
| private  数据成员 | status | int | 一卡通状态 |
| name | string | 持卡人姓名 |
| number | string | 卡号 |
| gender | string | 持卡人性别 |
| times | int | 乘车次数 |
| remain | int | 一卡通余额 |
| subordinate | string | 持卡人所属单位 |
| date | string | 一卡通有效期 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 分类 | 名称 | 参数列表 | 返回值 | 功能 |
| public  成员函数 | Student | (int,string,string,string,  int,int,string,string) | — | 构造函数 |
| ~Student | (void) | — | 析构函数 |
| show\_info | (void) | void | 展示一卡通信息 |
| top\_up | (void) | void | 充值一卡通 |
| get\_on | (char[ ]) | void | 使用一卡通乘车 |
| get\_off | (char[ ]) | void | 下车 |
| log\_out | (void) | void | 一卡通注销 |
| report\_loss | (void) | void | 一卡通挂失 |
| unlock\_report\_loss | (void) | void | 一卡通解除挂失 |

### 4.2.2 class Teacher 教师一卡通

Teacher 类有7个数据成员和9个成员函数。

|  |  |  |  |
| --- | --- | --- | --- |
| 分类 | 名称 | 类型 | 含义 |
| private  数据成员 | status | int | 一卡通状态 |
| name | string | 持卡人姓名 |
| number | string | 卡号 |
| gender | string | 持卡人性别 |
| times | int | 乘车次数 |
| subordinate | string | 持卡人所属单位 |
| date | string | 一卡通有效期 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 分类 | 名称 | 参数列表 | 返回值 | 功能 |
| public  成员函数 | Teacher | (int,string,string,string,  int,string,string) | — | 构造函数 |
| ~Teacher | (void) | — | 析构函数 |
| show\_info | (void) | void | 展示一卡通信息 |
| top\_up | (void) | void | 一卡通充值 |
| get\_on | (char[ ]) | void | 使用一卡通乘车 |
| get\_off | (char[ ]) | void | 下车 |
| log\_out | (void) | void | 一卡通注销 |
| report\_loss | (void) | void | 一卡通挂失 |
| unlock\_report\_loss | (void) | void | 一卡通解除挂失 |

### 4.2.3 class Limit 限制一卡通

Limit 类有9个数据成员和9个成员函数。

|  |  |  |  |
| --- | --- | --- | --- |
| 分类 | 名称 | 类型 | 含义 |
| private  数据成员 | status | int | 一卡通状态 |
| name | string | 持卡人姓名 |
| number | string | 卡号 |
| gender | string | 持卡人性别 |
| times | int | 乘车次数 |
| remain | int | 余额 |
| free\_times | int | 剩余免费次数 |
| subordinate | string | 持卡人所属单位 |
| date | string | 一卡通有效期 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 分类 | 名称 | 参数列表 | 返回值 | 功能 |
| public  成员函数 | Limit | (int,string,string,string,  int,int,int,string,string) | — | 构造函数 |
| ~Limit | (void) | — | 析构函数 |
| show\_info | (void) | void | 展示一卡通信息 |
| top\_up | (void) | void | 充值一卡通 |
| get\_on | (char[ ]) | void | 使用一卡通乘车 |
| get\_off | (char[ ]) | void | 下车 |
| log\_out | (void) | void | 一卡通注销 |
| report\_loss | (void) | void | 一卡通挂失 |
| unlock\_report\_loss | (void) | void | 一卡通解除挂失 |

### 4.2.4 class Bus 班车

Bus 类有6个成员函数和9个成员函数。

|  |  |  |  |
| --- | --- | --- | --- |
| 分类 | 名称 | 类型 | 含义 |
| private  数据成员 | number | string | 车牌号 |
| type | string | 车类型 |
| driver | string | 司机名字 |
| max\_num | int | 最大乘客数 |
| cur\_num | int | 现有乘客数 |
| start\_time | int | 发车时间 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 分类 | 名称 | 参数列表 | 返回值 | 功能 |
| public  成员函数 | Bus | (string,string,string,int,int,int) | — | 构造函数 |
| ~Bus | (void) | — | 析构函数 |
| show\_bus | (void) | void | 展示该车的信息 |
| get\_cur\_num | (void) | int | 获得当前该车上乘客数 |
| check\_bus | (void) | void | 乘客乘车 |
| add\_people | (string,string) | void | 增加乘客 |
| delete\_people | (string,string) | void | 删除乘客 |
| bus\_detail | (void) | void | 展示该车上当前乘客具体信息 |
| clear\_bus | (void) | void | 清空该车 |

### 4.2.5 class System 系统

System 类较为特殊，无数据成员，只有成员函数。由于该系统中并未对System类进行实例化，因此仅有静态成员函数。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 分类 | 名称 | 参数列表 | 返回值 | 功能 |
| public  成员函数 | start | (void) | static void | 进入系统主页 |
| moni | (void) | static void | 模拟班车运行 |
| get\_student | (char[]) | static Student | 获得一个Student对象 |
| get\_teacher | (char[]) | static Teacher | 获得一个Teacher对象 |
| get\_limit | (char[]) | static Limit | 获得一个Limit对象 |
| **create\_card** | (void) | static void | 进入创建一卡通页面 |
| create\_student | (void) | static void | 创建学生一卡通 |
| create\_teacher | (void) | static void | 创建教师一卡通 |
| create\_limit | (void) | static void | 创建限制一卡通 |
| **show\_card** | (void) | static void | 进入展示一卡通页面 |
| show\_student\_card | (void) | static void | 展示学生一卡通 |
| show\_teacher\_card | (void) | static void | 展示教师一卡通 |
| show\_limit\_card | (void) | static void | 展示限制一卡通 |
| **top\_up** | (void) | static void | 充值一卡通 |
| top\_up\_student | (void) | static void | 充值学生一卡通 |
| top\_up\_teacher | (void) | static void | 充值教师一卡通 |
| top\_up\_limit | (void) | static void | 充值限制一卡通 |
| **get\_on** | (void) | static void | 进入乘车页面 |
| get\_on\_student | (char[],char[]) | static void | 学生一卡通乘车 |
| get\_on\_teacher | (char[],char[]) | static void | 教师一卡通乘车 |
| get\_on\_limit | (char[],char[]) | static void | 限制一卡通乘车 |
| **operate\_card** | (void) | static void | 进入操作一卡通页面 |
| **log\_out** | (void) | static void | 进入注销一卡通页面 |
| log\_out\_student | (void) | static void | 学生一卡通注销 |
| log\_out\_teacher | (void) | static void | 教师一卡通注销 |
| log\_out\_limit | (void) | static void | 限制一卡通注销 |
| **report\_loss** | (void) | static void | 进入挂失一卡通页面 |
| report\_loss\_student | (void) | static void | 学生一卡通挂失 |
| report\_loss\_teacher | (void) | static void | 教师一卡通挂失 |
| report\_loss\_limit | (void) | static void | 限制一卡通挂失 |
| **unlock\_report\_loss** | (void) | static void | 进入解除挂失一卡通页 |
| unlock\_student | (void) | static void | 学生一卡通解除挂失 |
| unlock\_teacher | (void) | static void | 教师一卡通解除挂失 |
| unlock\_limit | (void) | static void | 限制一卡通解除挂失 |
| get\_bus | (char[]) | static Bus | 获得一个Bus对象 |
| add\_bus | (void) | static void | 添加一个班车 |

## 4.3 关键算法流程

### 4.3.1 class Student

类Student中，涉及的关键算法是对实例的析构函数。

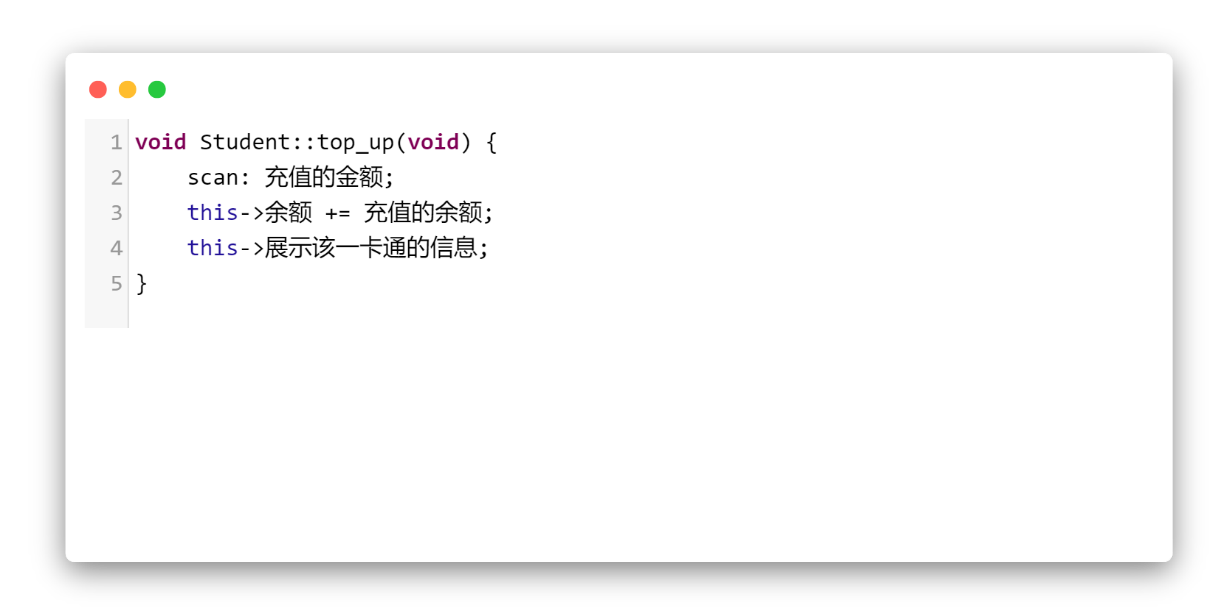
* **~Student 算法：**



* **get\_on 算法：**

****

* **top\_up 算法**



### 4.3.2 class Teacher

* **~Teacher** **算法**

和Student类的析构函数算法类似。

* **get\_on 算法**

和Student类的get\_on函数算法相似

### 4.3.3 class Limit

* **~Limit:**

和Student类的析构函数相似

* **get\_on**

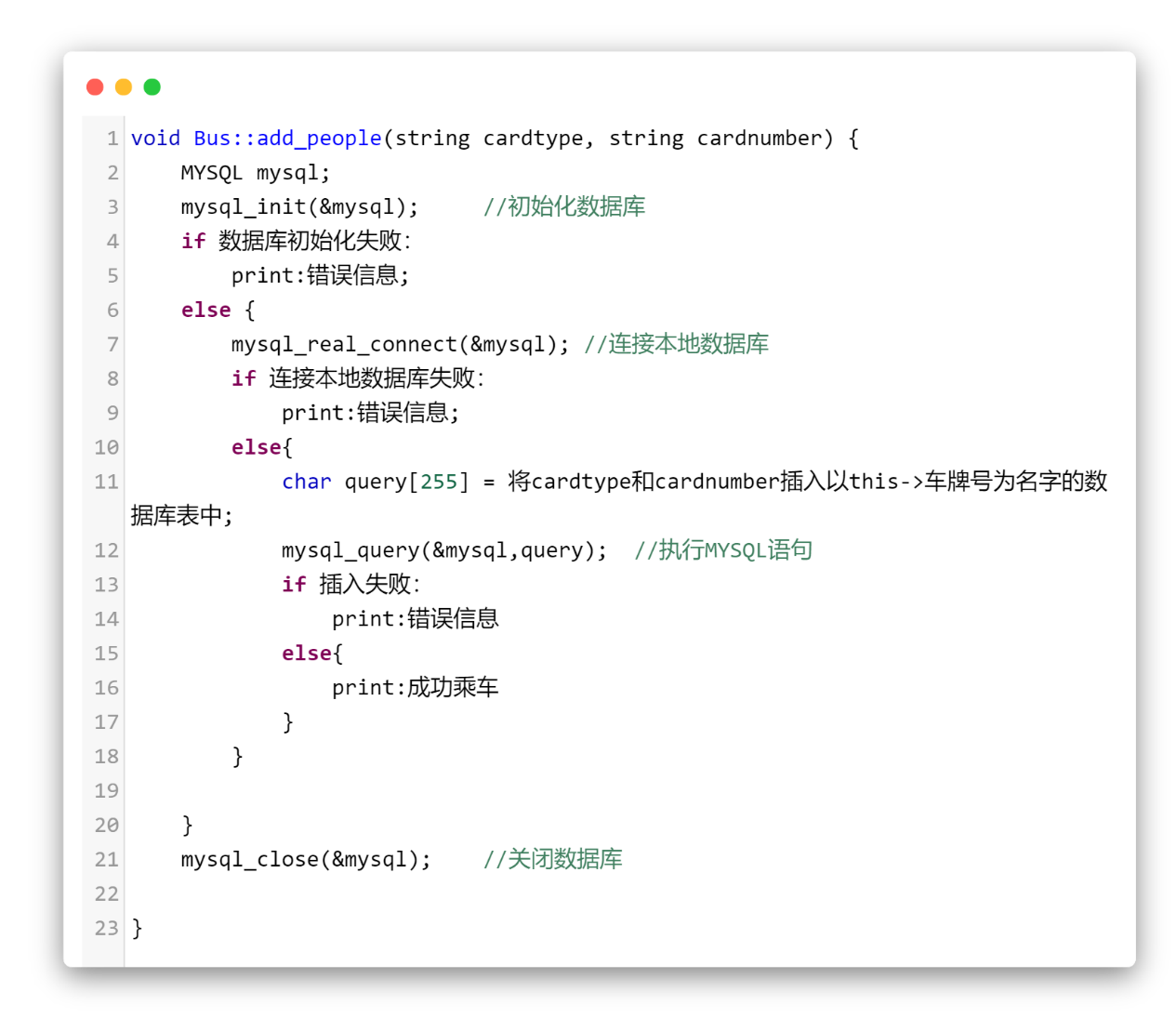
和Student类的get\_on函数相似

### 4.3.4 class Bus

* **get\_cur\_num**



* **add\_people**



* **bus\_detail**



### 4.3.5 class System

* **get\_setudent**



* **get\_teacher**

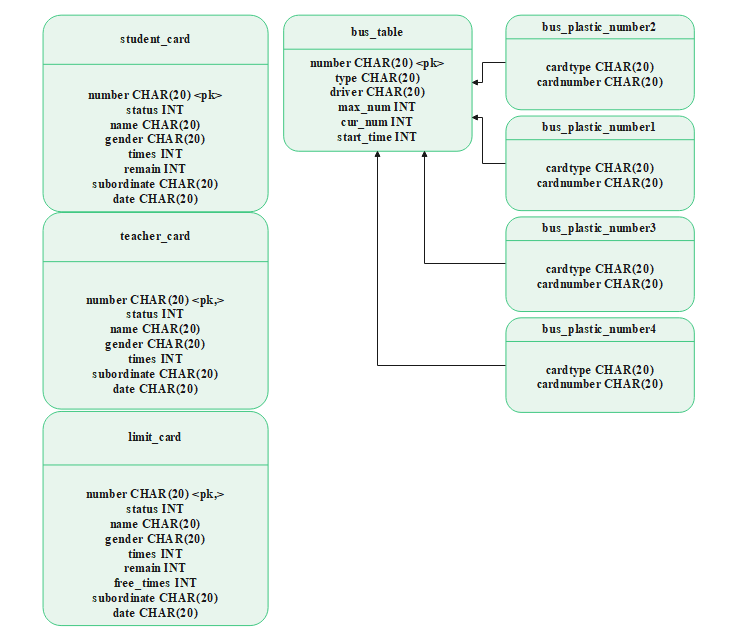
和get\_student类似

* **get\_limit**

和get\_student类似

## 4.4 数据存储

使用Visual Studio 2019连接本地MySQL数据库，来存储学生一卡通、教师一卡通、限制一卡通、班车信息以及每辆班车当前所载的乘客信息。



## 4.5 遇到的问题

* 系统数据的存储问题

第一次实验时，使用txt文本文件进行存储一卡通信息以及班车信息，使用C++的头文件 fstream 可以对文件进行读写。

但是C++对于文本文件的操作十分繁琐。不能直接在txt文件中查找某个字段中的指定字符串，只能先将txt文件中每一行读入C++的缓冲区中，然后在进行对比某个字段的值。除此之外，对于txt文件某条记录的更新，需要重写该txt文件，即先将原先内容保存，再修改内容，最后写入txt文件。

* 解决方法：

因此，我最终选择了使用Visual Studio2019集成开发环境连接本地MySQL数据库，来进行数据的存储。只需要更改Visual Studio的项目设置中属性的 VC++ **包含目录**和**库目录**为MySQL的包含目录和库目录所在路径即可。

然后，再从MySQL官方网站学习MySQL给予C++的API接口即可。

* C++ 中使用MySQL API接口时遇到的问题

1. 在使用mysql\_query()查询数据库中的信息时，所显示出的中文乱码。

2. 使用sprintf\_s( ) 将C++中的变量传入MySQL语句时报错

3. 访问数据库冲突

* 解决方法

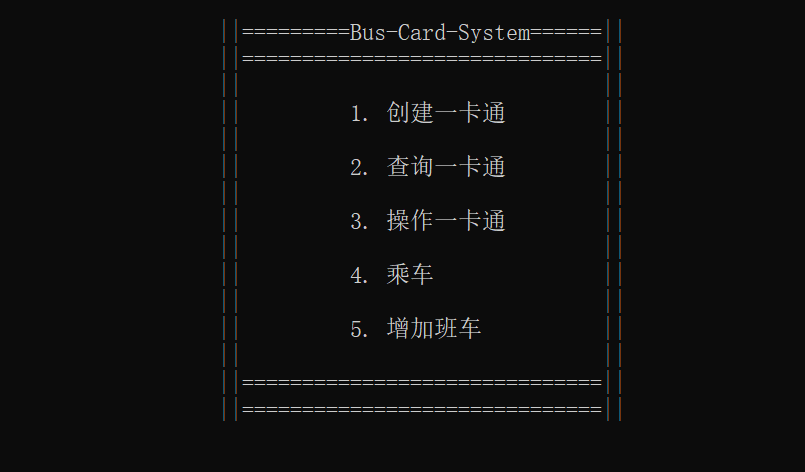
1. 使用 mysql\_options() 语句将数据库编码方式设置为 gbk

2. 传入变量为字符串时，需要传入char[]字符串，须将string转换成char[]。此外，在传入数据库表名时，不需要加单引号

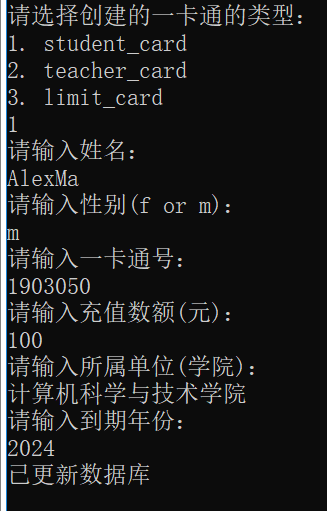
3. 在使用完MySQL本地数据库后，需要将数据库使用API 接口函数mysql\_close() 关闭，和mysql\_real\_connect() 前后呼应，不然会引发访问冲突。

# 五、实验结果

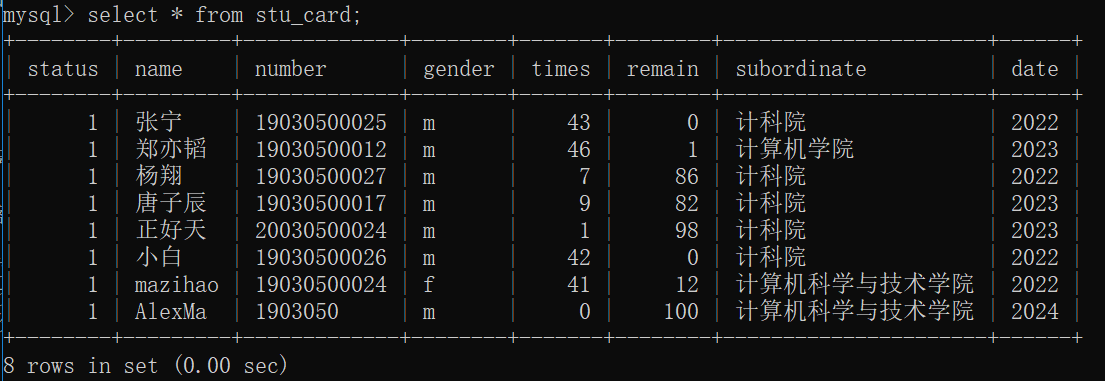
* **主页面**



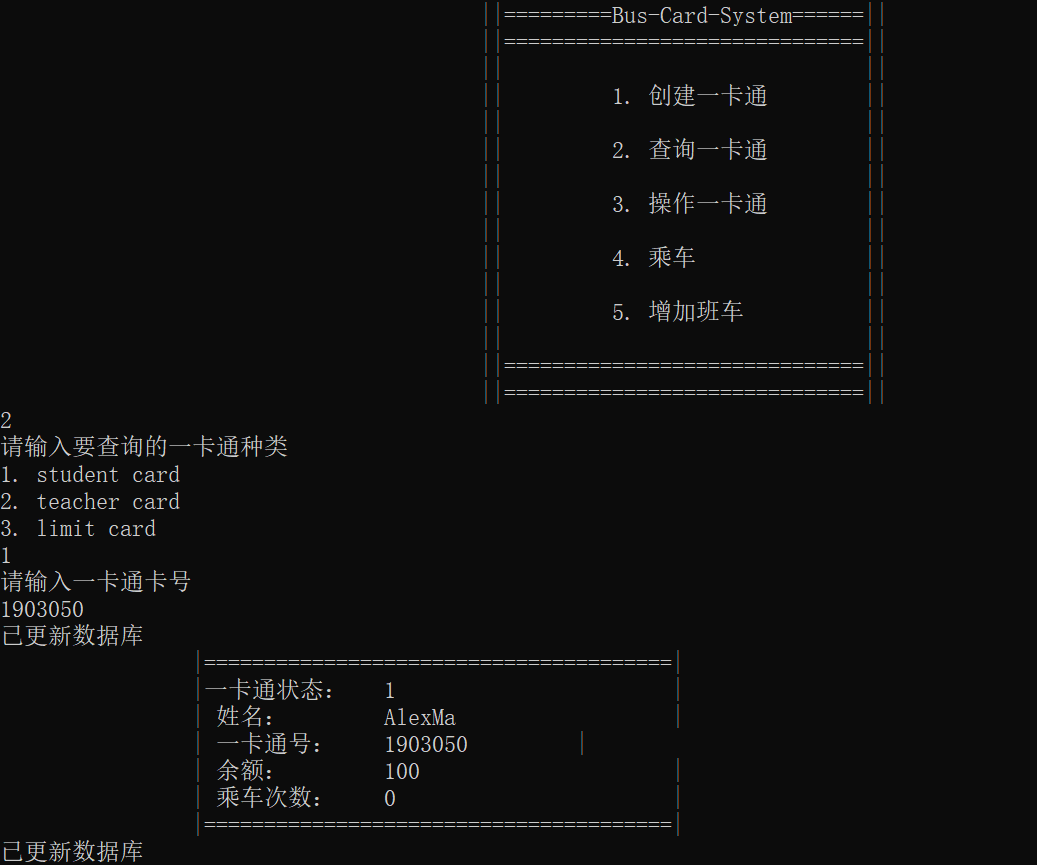
* **创建一卡通**



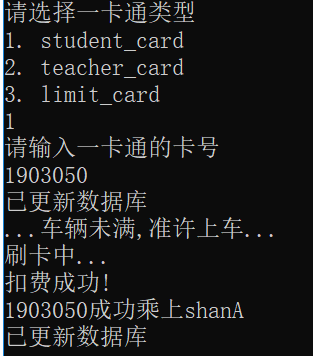
* **创建后，本地数据库表stu\_card中：**



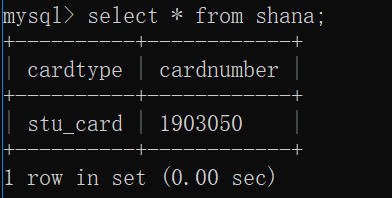
* **查询一卡通**



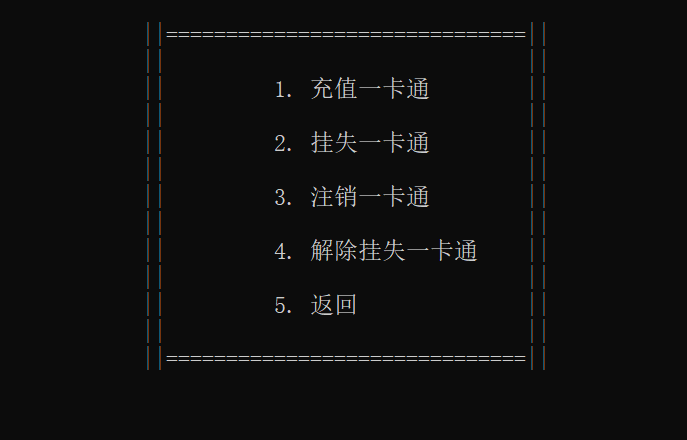
* **使用一卡通乘车**



* **乘车后shanA的数据库表**



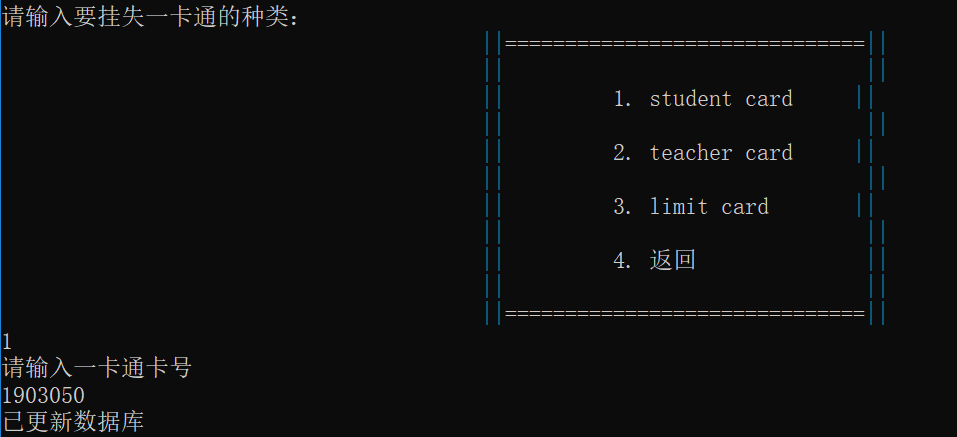
* **操作一卡通页面**



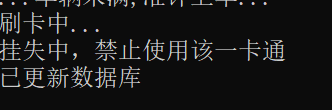
* **充值一卡通**



* **挂失一卡通**



* **挂失一卡通后乘车**



# 六、实验总结

通过本次使用C++对西电乘车一卡通系统的模拟，对C++的面向对象程序设计范型有了更深入的理解。面向对象你的程序设计范型和封装机制将定义和实现分离，将复杂的内部细节隐藏，提高了程序的可复用性和可维护性。这次实验中，主函数main() 之外，基本只定义了类、类的非静态成员函数、类的静态成员函数。

类中最特别的**成员函数**应该就是**构造函数、析构函数和虚拟函数**了，本实验中最令我印象深刻的是，对三个一卡通类的析构函数的定义。由于每次对一卡通对象的信息的编辑，**一定**需要及时更改到相应的数据库表中，因此我想到可以直接在析构函数中将该一卡通对象生命周期的最后的状态更新到相对应的数据库表中。这样不需要繁琐地在每次对一卡通的操作后更新数据库。

除此之外，之前只是将Hexo博客部署在Github上，在本次实验时我首次使用Github用于项目的版本管理。一个人的使用优点并不突出，但是当一个团队来维护一个项目时，我觉得会极大的提高开发项目的效率和正确性。