

车载软件开发基础

课后实践2



**2022至2023学年第 1 学期**

|  |  |
| --- | --- |
| 学号 | 姓名 |
|  | 邹光辉 |
| 任课教师 | 刘骥 |
| 成 绩 |  |

|  |  |
| --- | --- |
| 任务书 | |
| 任务内容 | 请针对课后实践1的内容，将各模块用class改写。各模块的属性保持不变，增加相关操作行为。补充信息如下：   1. 智能小车   增加操作：   1. setID 实现编号录入 2. print 实现智能小车信息的屏幕输出 3. save 实现智能小车信息的本地文件保存 4. 底盘   增加操作：   1. 各属性的set方法 2. print 实现底盘信息的屏幕输出 3. save 实现底盘信息的本地文件保存 4. 轮胎   增加操作：   1. 各属性的set方法 2. print 实现轮胎信息的屏幕输出 3. save 实现轮胎信息的本地文件保存 4. AGX套件   增加操作：   1. 各属性的set方法 2. print 实现AGX套件的屏幕输出 3. save 实现AGX套件的本地文件保存 4. 双目摄像头   增加操作：   1. 各属性的set方法 2. print 实现双目摄像头信息的屏幕输出 3. save 实现双目摄像头信息的本地文件保存 4. 9轴陀螺仪   增加操作：   1. 各属性的set方法 2. print 实现9轴陀螺仪信息的屏幕输出 3. save 实现9轴陀螺仪信息的本地文件保存 4. 多线激光雷达   增加操作：   1. 各属性的set方法 2. print 实现多线激光雷达信息的屏幕输出 3. save 实现多线激光雷达信息的本地文件保存 4. 液晶显示屏   增加操作：   1. 各属性的set方法 2. print 实现液晶显示屏信息的屏幕输出 3. save 实现液晶显示屏信息的本地文件保存 4. 电池模块   增加操作：   1. 各属性的set方法 2. print 实现电池模块信息的屏幕输出 3. save 实现电池模块信息的本地文件保存 4. 学生信息包括：   增加操作：   1. 各属性的set方法 2. print 实现学生信息的屏幕输出 3. save 实现学生信息的本地文件保存   要求：   1. 上述方法的参数自定义。构造函数自定义 2. Main函数中实现小车信息录入、分配、保存和屏幕显示。 3. 小车各模块的保存和显示分别通过相应对象的操作完成 4. 用程序打开保存的文件，然后将相关信息显示在屏幕上（含分配的学生信息）。一次显示1辆小车信息，按n键显示下一辆小车信息，按p显示上一辆小车信息。请注意：显示第1台显示按p无效，显示第10台小车，按n无效。 |
| 程序规范 | （1）所有程序代码采用C++编写，使用git进行源代码管理；  （2）类名、变量名、函数名应符合C++的命名规范，并在代码中前后保持一致；  （3）涉及面向对象的程序，例如自定义的类，应符合面向对象的设计原则；  （4）正确使用头文件和源文件，自定义的头文件应符合头文件的编写原则，例如用条件宏定义确保头文件不被多次引用、不在头文件中进行类和函数的实现（模板除外）； |
| 报告要求 | （1）报告至少应该包括程序设计、程序效果展示、总结分析3个部分；  （2）程序设计描述组成程序的模块、类、函数以及他们之间的相互关系，若有算法，可以描述算法流程；  （3）程序效果展示除了程序运行效果截图之外，应该有必要的文字说明；  （4）总结分析可以分析实现的效果与理想情况的差异，分析导致这些差异的原因，切忌不要写成心得体会；  （5）报告应该格式规范、排版整洁、少语病和错误。 |
| 作业提交 | （1）含有git仓库（有.git目录）的完整源代码；  （2）任务报告。 |
| 评分标准 | 按照五级制打分，分为优秀、良好、中等、及格、不及格，各评分项占总成绩的比例为：  （1）任务完成情况占评分的60%；  （2）报告占评分的40%。  评分老师根据各部分的完成情况，直接给出总成绩。 |

# 一、程序设计

## 1.1程序设计概述

本次程序分为main.cpp，Car.cpp，Car.h,将小车操作的函数以及小车的所有相关信息对象都声明在Car.h中，并且将Car.cpp中定义了void Read\_message(void);以及小车，底盘，摄像头等相关对象的方法， main函数中实现小车信息录入、分配、保存和屏幕显示

这个程序设计旨在创建一个用于管理智能小车和学生信息的系统。程序包括自定义对象以及想关操作的方法、函数和它们之间的相互关系，以实现智能小车信息录入、学生信息录入、分配小车给学生、保存信息到文件和从文件读取信息等功能。

## 1.2模块和结构体

程序包括以下模块和类：

1.SmartCar类：表示智能小车，包含编号（carID）、底盘信息（Chassis）、AGX套件信息（AGXKit）、双目摄像头信息（StereoCamera）、9轴陀螺仪信息（Gyroscope）、多线激光雷达信息（Lidar）、液晶显示屏信息（Display）、电池模块信息（Battery）和学生信息（Student）。这个类具有方法来设置智能小车的信息、显示小车信息和保存小车信息到本地文件。

2.Chassis类：表示底盘信息，包括底盘编号（chassisID）、型号（model）、轴距（wheelbase）、轮距（track）、最小离地间隙（groundClearance）、最小转弯半径（minimumTurningRadius）、驱动形式（driveType）、最大行程（maxRange）和轮胎信息（tires）。这个类具有方法来设置底盘信息、显示底盘信息和保存底盘信息到本地文件。

3.AGXKit类、StereoCamera类、Lidar类、Gyroscope类、Display类、Battery类：这些类分别表示AGX套件、双目摄像头、多线激光雷达、9轴陀螺仪、液晶显示屏和电池模块的信息，每个类包括属性和方法来设置信息、显示信息和保存信息到本地文件。

4.Student类：表示学生信息，包括学号（studentID）和姓名（studentName），具有方法来设置学生信息、显示学生信息和保存学生信息到本地文件。

主函数：

主函数包括一个循环，该循环允许用户选择操作（写入信息、读取信息、退出系统）。根据用户的选择，主函数会调用相应的方法来录入或显示智能小车信息。

文件操作：

程序支持将智能小车信息保存到本地文件（"smartcars.txt"），以便后续读取和显示。保存操作在每辆小车的相关类中实现，读取和显示操作在主函数中实现。

总体来说，这个程序设计旨在创建一个管理智能小车信息的系统，通过面向对象的编程风格，将不同类型的信息（底盘、AGX套件、双目摄像头等）组织成各自的类，以便更好地管理和操作这些信息。用户可以选择写入新的信息、读取已有的信息或退出系统。

## 1.3函数

SmartCar类

- `void setID(const string& id)`

- 设置智能小车的编号。

- `void print() const`

- 在屏幕上显示智能小车的信息。

- `void save(ofstream& outputFile) const`

- 将智能小车的信息保存到本地文件。

Chassis类

- `void setChassisInfo()`

- 设置底盘相关信息

- `void print() const`

- 在屏幕上显示底盘的信息。

- `void save(ofstream& outputFile) const`

- 将底盘的信息保存到本地文件。

AGXKit类

- `void setAGXKitInfo() `

- 设置AGX套件的相关信息。

- `void print() const`

- 在屏幕上显示AGX套件的信息。

- `void save(ofstream& outputFile) const`

- 将AGX套件的信息保存到本地文件。

StereoCamera类，Gyroscope类，Lidar类，Display类，Battery类，Student类

（类似AGXKit类，包括 set 方法、print 方法和 save 方法）

void Read\_message(void):用户查询小车的相关信息并且显示在屏幕上。

## 1.4算法流程

用户启动程序后，通过按下W或者R可以选择录入智能小车信息和学生信息，或者从文件读取已保存的信息，或者也可以按Q退出程序

1如果按下W选择录入信息

（1）用户将根据屏幕提示逐步输入智能小车的各个部分的信息，包括底盘、AGX套件、摄像头、激光雷达、陀螺仪、显示屏、电池模块等。

（2） 用户输入学生的学号和姓名。

（3） 程序自动将小车按输入顺序分配给学生，每个学生只能分配一辆小车。

（4）在输入完信息后自动将将录入的信息保存到文件中并回到初始的选择界面

2如果按下R选择读取信息

（1）屏幕会提示相关帮助信息，按n键显示下一辆小车信息，按p显示上一辆小车信息。显示第1台车时按p会显示无效，显示第10台小车，按n会显示无效。

（2）如果读取文件失败会提示“Unable to open the file for reading”。

（2）任意时刻按下q可退出只初始状态

3如果按下Q，直接退出程序

## 1.5总结

这个程序的核心是`SmartCar`结构体，它协调并管理其他类的信息，包括智能小车和学生的信息。通过自定义多个小车相关信息的对象以及方法和void Read\_message(void)函数来实现信息的录入、保存和读取、显示。

# 二、程序效果展示

1、程序运行初始状态如下，选择用户需要的操作

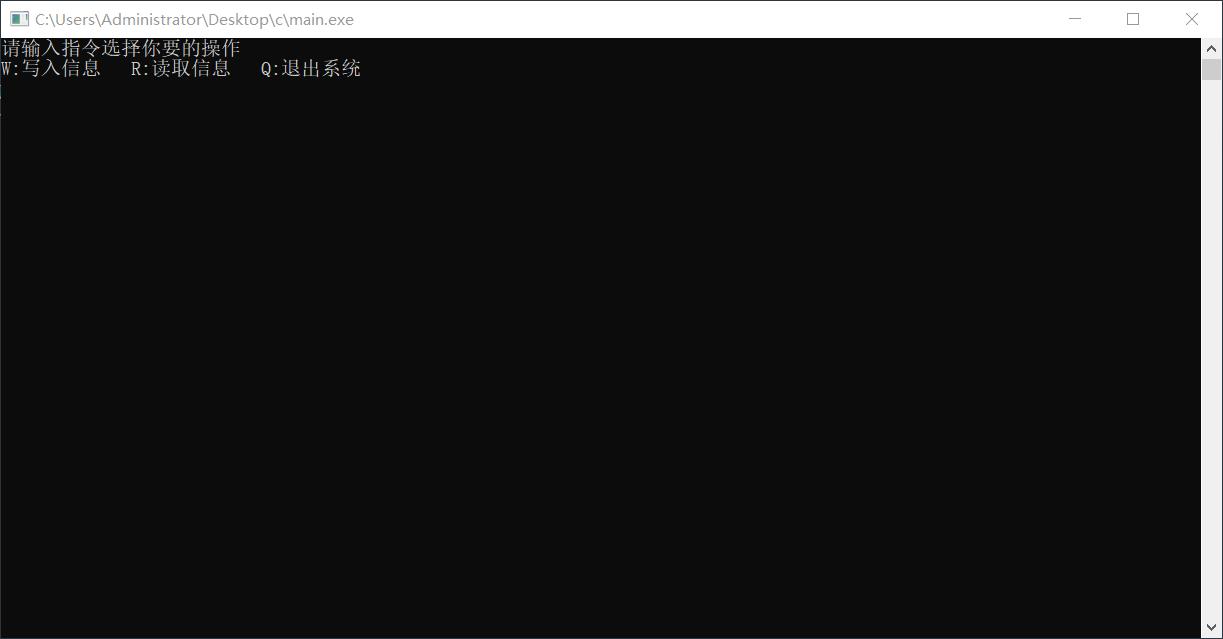


图1初始状态

2、按下W选择写入信息的界面

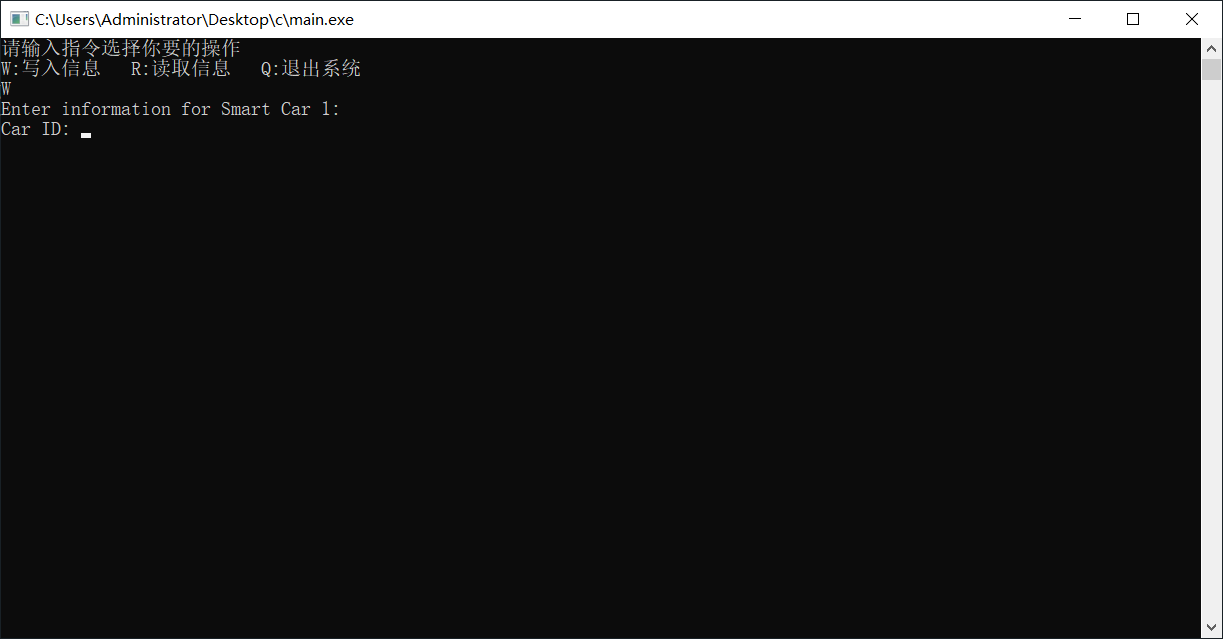
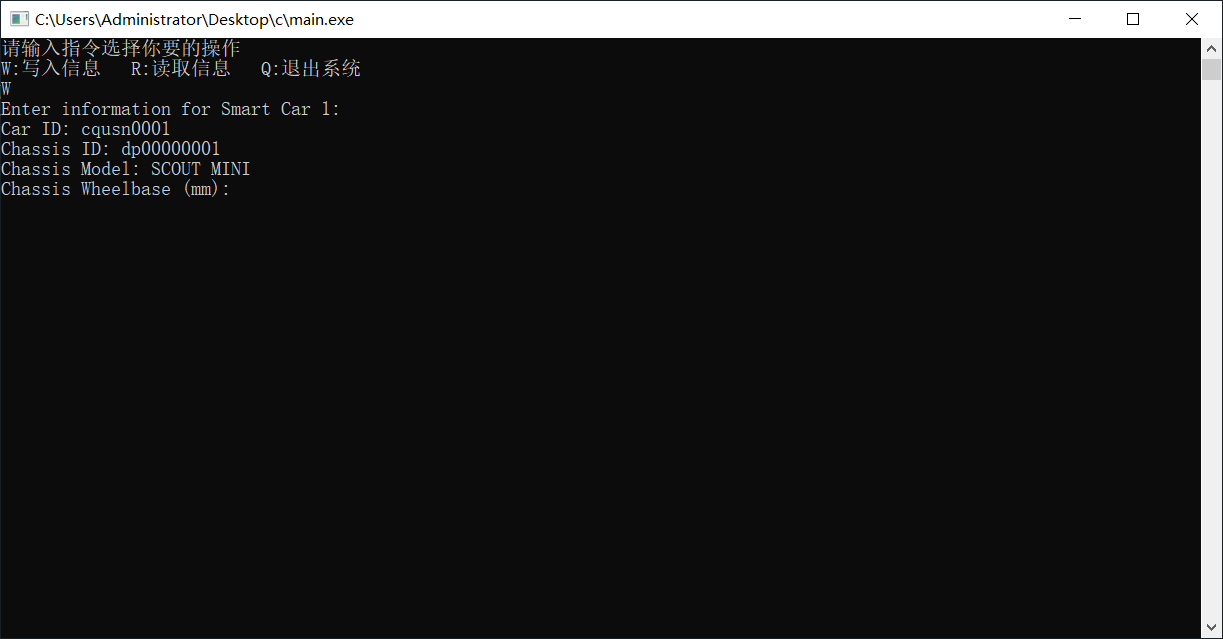
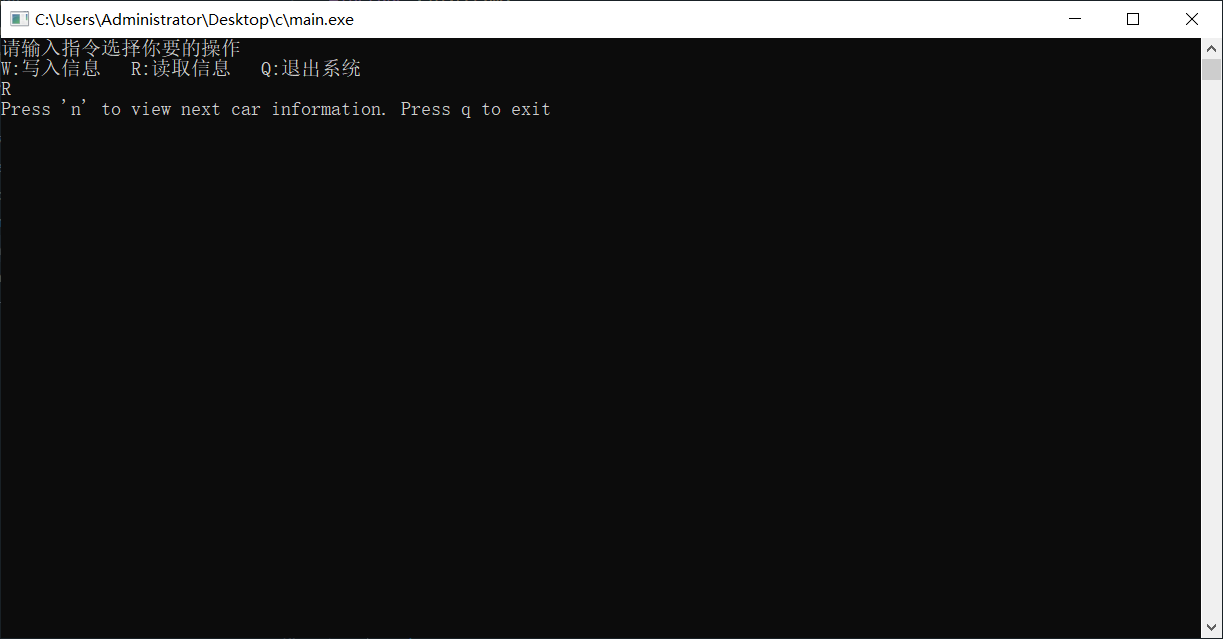


图2信息录入

3、按照提示依次录入小车的信息

图3 录入提示信息

4、选择读取信息的界面

图4读取界面

5、提示按下n或者p显示上一辆或者下一辆小车的信息，按q退出，由于初始状态时在现实第一辆小车，所以只显示按下n，此时按下p显示无效的输入

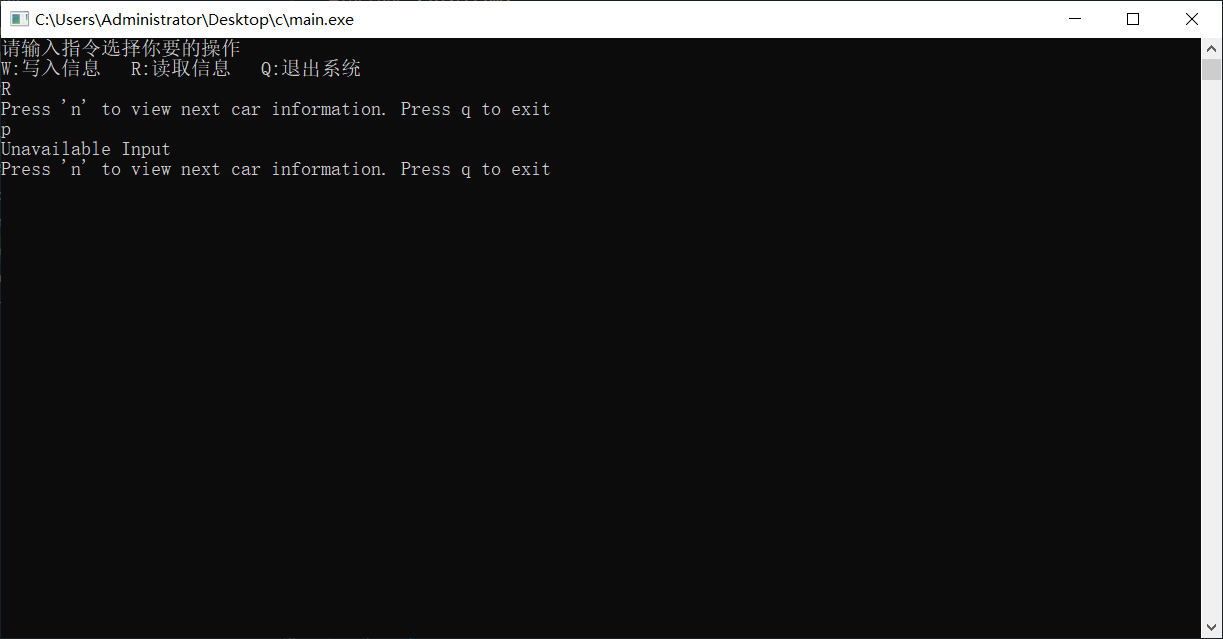


图5 无效输入

6、按下n显示第二辆小车的信息

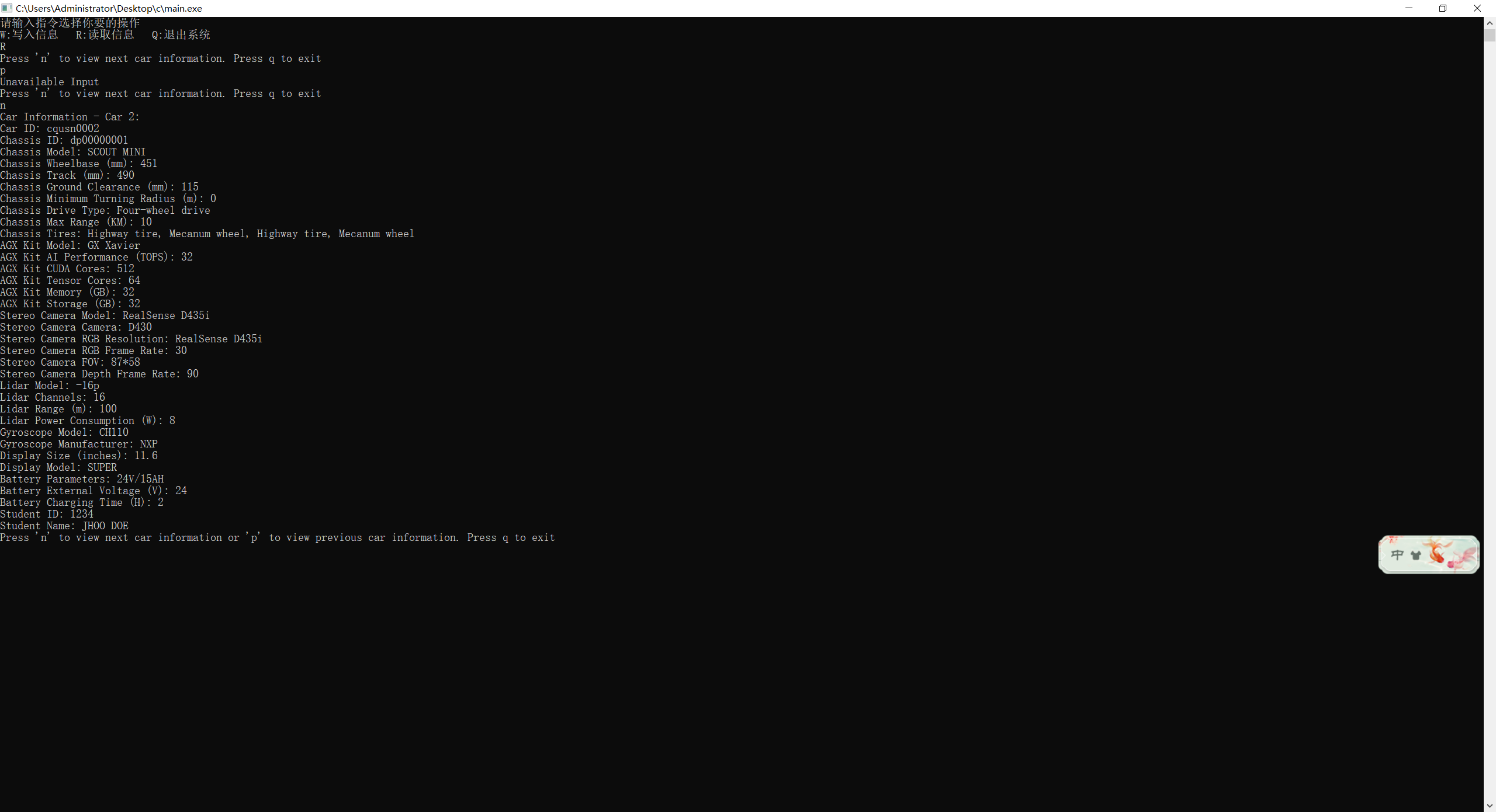


图6查看信息

7、显示到第10辆小车时的情况

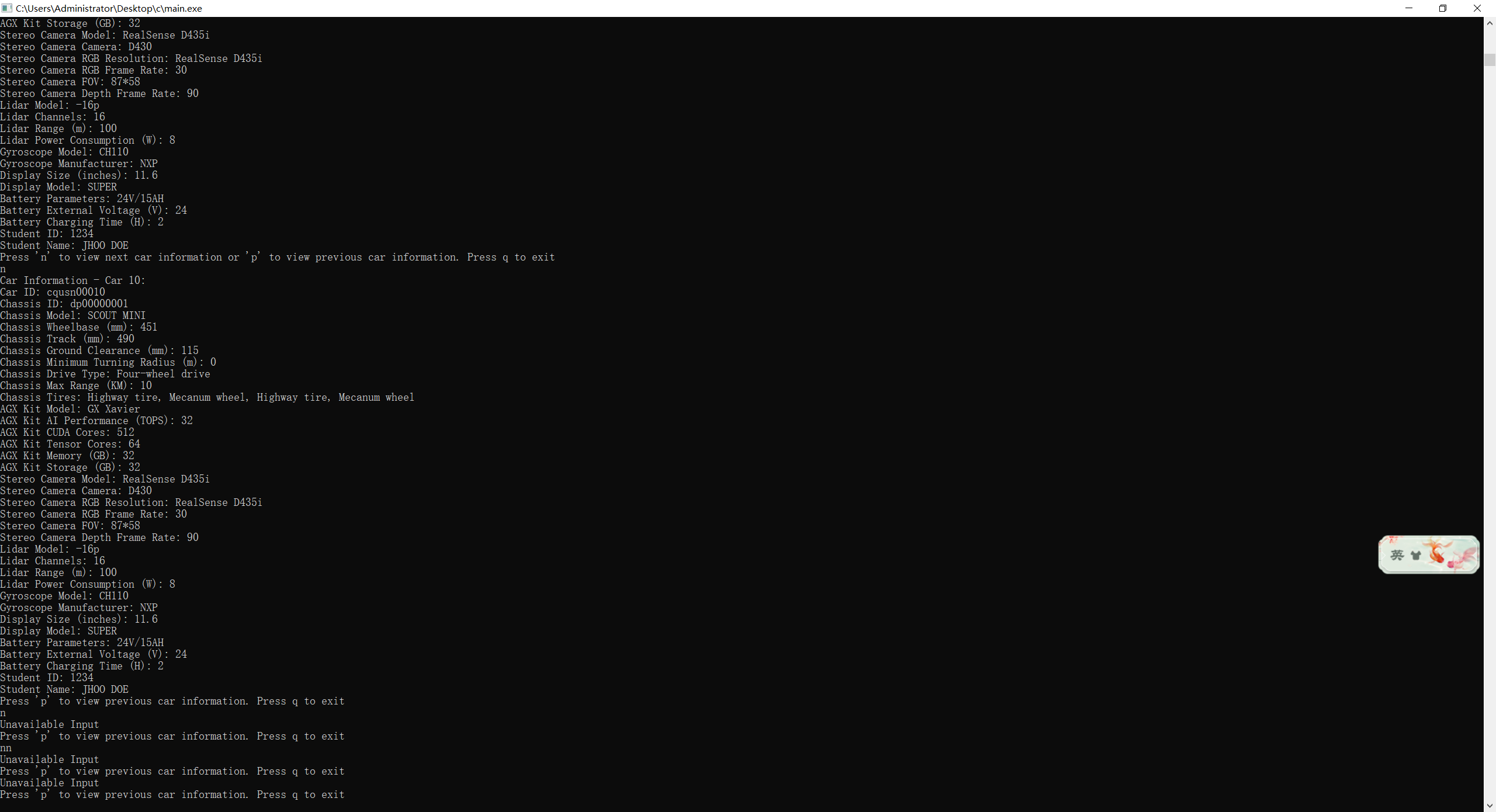


图7 第10辆小车信息

8、按q退出回到初始状态

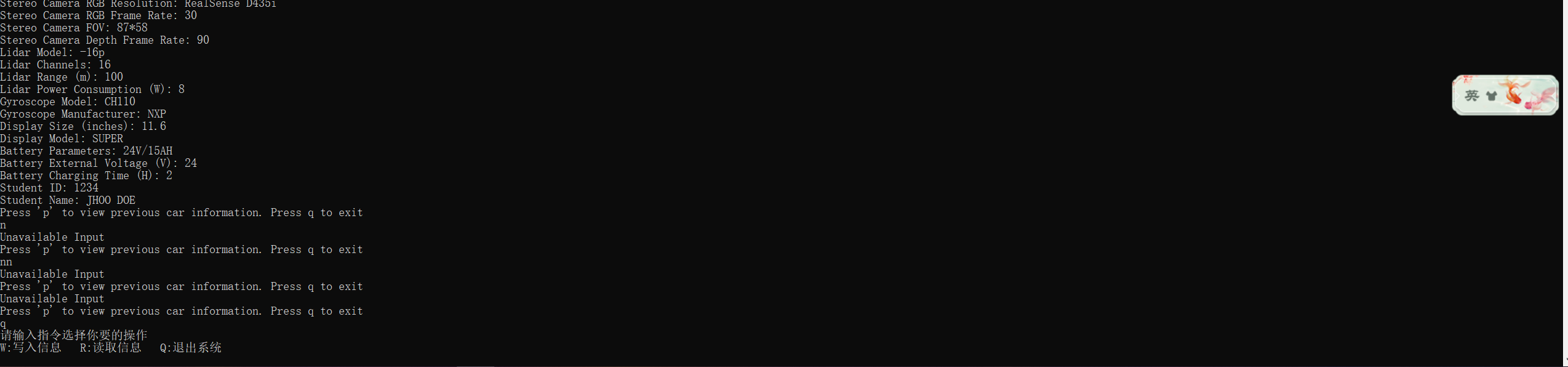


图8退出至原始界面

# 三、总结分析

## 3.1总体评估：

这份代码成功实现了管理智能小车和学生信息的基本功能。用户可以录入小车信息和学生信息，将小车分配给学生，并将信息保存到文件。程序还允许用户从文件中读取信息，并以逐个导航的方式显示小车信息。这个程序设计有助于管理和跟踪多台智能小车的分配情况。

并且在本次开发中使用了github对代码进行版本管理，大大提高了程序开发效率。

## 3.2实现效果：

（1）信息录入和显示：信息录入和显示功能运行良好，用户可以逐步录入小车信息和学生信息，并以逐个导航的方式查看小车信息。

（2）信息保存和读取：信息可以成功保存到文件中，并且可以从文件中读取。这提供了数据的持久性，允许用户在不同会话中访问相同的信息。

（3）分配小车：用户可以将小车分配给学生，建立了小车和学生之间的关联。这有助于跟踪哪位学生使用了哪辆小车。

## 3.3差异和原因：

（1）用户友好性：程序在信息录入方面做得很好，但在导航小车信息时存在一些限制。用户在初始化时只能按n查看第二辆小车的信息，之后再按p才能查看第一辆小车的信息，这可能会限制用户在导航小车信息时的便捷性，但是考虑到这个综合性。但为了用户选择查看信息时观感效果，最终选择进入默认为第一辆小车信息并且不显示。

（2）输入格式：程序目前假设用户会按照预期的格式输入信息，但没有处理用户可能输入的不良数据。更强大的输入验证机制将有助于提高程序的稳定性。

（3）在输入数据时只能一次性按照提示信息完整输入所有小车和学生信息后才能回到初始菜单界面，无法像读取数据时随时可以按q退出界面。出现这种情况的原因是由于在读取数据的输入时使用了大量的cin或者getline读取一行字符串或者浮点（整数）类型数据，同时每次输入前需要显示输入提示信息，想要做到随时退出就必须每一次读取的输入都进行if判断，这样会大大增加开发时间和复杂性并且意义不大。

## 3.4改进建议：

（1）导航改进：改进导航功能，使用户可以在任何时间点跳转到任何小车的信息，而不受限制。

（2）错误处理：增加更多的错误处理机制，包括检查重复分配和输入验证，以确保数据的一致性。

（3）用户友好性：改善用户界面，提供更友好的菜单和反馈，以提高用户体验，甚至可以直接使用QT结合C++进行软件开发，大大提高与用户的交互性，并且还利于一直到各类嵌入式产品中。

（4）更多功能：考虑添加其他功能，如搜索、编辑和删除小车信息，以增强程序的实用性。

总的来说，这个程序是一个很好的起点，但仍有改进的空间，以提高用户友好性和稳定性。

**4参考文献**

1. Stroustrup, B. (2013). The C++ Programming Language. Addison-Wesley Professional.
2. Meyer, B. (1997). Object-Oriented Software Construction. Prentice Hall.
3. Martin, R. C. (2008). Clean Code: A Handbook of Agile Software Craftsmanship. Prentice Hall.
4. Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Professional.
5. Fowler, M. (2018). Refactoring: Improving the Design of Existing Code. Addison-Wesley Professional.
6. Lippman, S. B., Lajoie, J., & Moo, B. E. (2012). C++ Primer. Addison-Wesley Professional.
7. McConnell, S. (2004). Code Complete: A Practical Handbook of Software Construction. Microsoft Press.
8. Sutter, H., & Alexandrescu, A. (2004). C++ Coding Standards: 101 Rules, Guidelines, and Best Practices. Addison-Wesley Professional