

车载软件设计基础

课后实践3



**2022至2023学年第 1 学期**

|  |  |
| --- | --- |
| 学号 | 姓名 |
| E2019051 | 曹代伟 |
| 任课教师 | 陈恒鑫 |
| 成 绩 |  |

|  |  |
| --- | --- |
| 任务书 | |
| 任务内容 | 请针对课后实践2的内容，补充多线激光雷达和底盘模块，使用订阅者模式，实现以下功能。   1. 多线激光雷达作为主题方，发布主题，并接受订阅 2. 底盘模块中增加观察者，向雷达模块进行订阅注册，并接收订阅通知。 3. 当雷达更新障碍物状态时，底盘模块通过订阅者模式接收通知并执行对应操作。 4. 障碍物状态为“前方”，则底盘执行“后退”执行 5. 障碍物状态为“右前方”，则底盘执行“左转”执行 6. 障碍物状态为“左前方”，则底盘执行“右转”执行   注意：  1、底盘执行指令，只需要在屏幕输出类似“左转。。。。”文字即可。  2、障碍物状态可通过命令行方式输入。比如1代表前方障碍，2代表左前方障碍，3代表右前方障碍。 |
| 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。 |

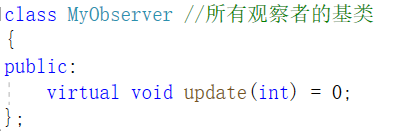
1. **程序设计**

根据报告要求可知，应采用23中设计模式中的观察者模式，其中多线激光雷达（MultilineLidar）应为被观察者，在发现前方有障碍物之后，向它的观察者（底盘模块）发送消息。虽然目前只有一个观察者和一个被观察者，但为了考虑到后期的可扩展性，还是应为所有不同类型的目标类设置一个共同的父类---class MySubject，为所有不同类型的观察者类设置一个共同的父类---class MyObserver。

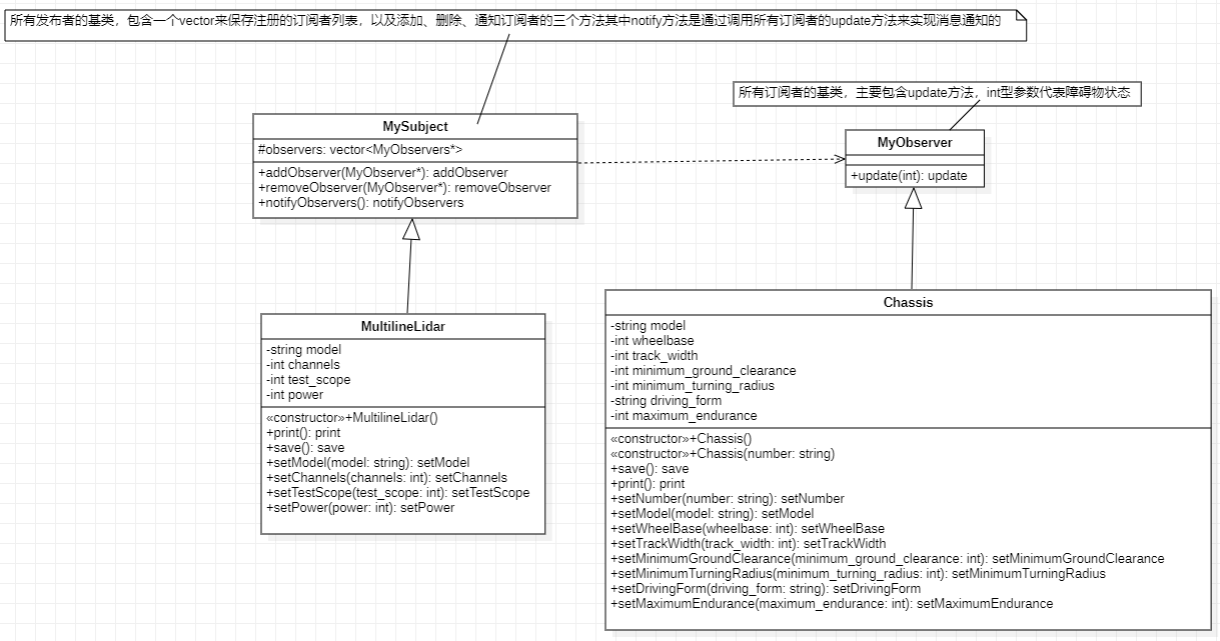
其中class MySubject结构如下：



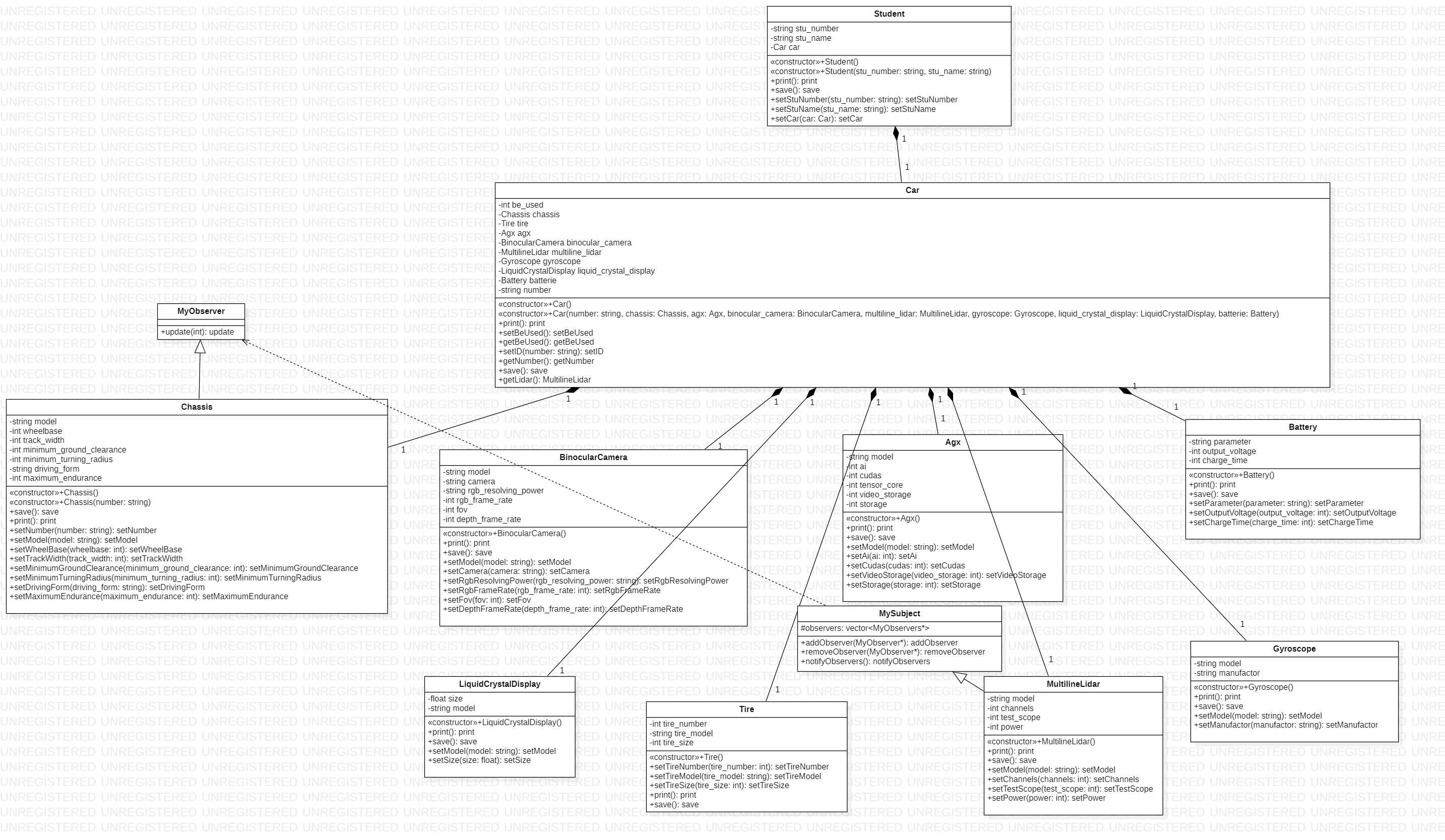
class MyObserver结构如下：



观察者模式中类结构设计如下：

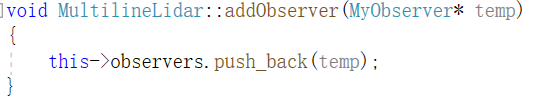


整体类结构图如下：

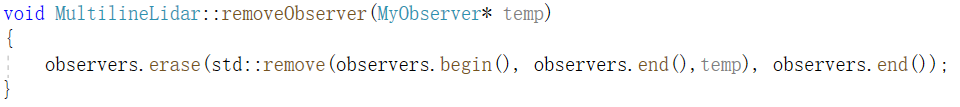


**关键函数：**

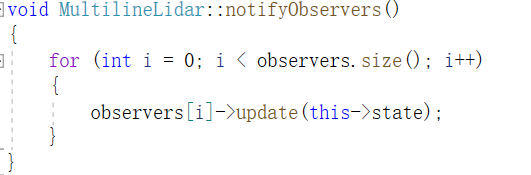
1. addObserver：为目标类添加观察者对象，只需传入观察者对象的基类指针即可，无需指定具体的观察者类型，然后通过vector<MyObserver\*> observers;自带的push\_back方法添加至vector尾部即可。



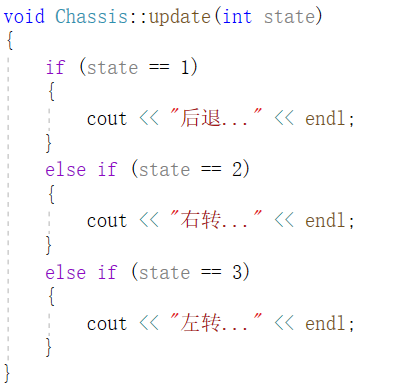
1. removeObserver:从vector<MyObserver\*> observers；中移除指定的观察者对象，同样只需传入基类指针即可，然后遍历整个观察者对象列表，找到相符的元素后直接移除。



1. notifyObservers：遍历所有观察者列表，调用它们的update方法，传入状态码，达到通知的效果。



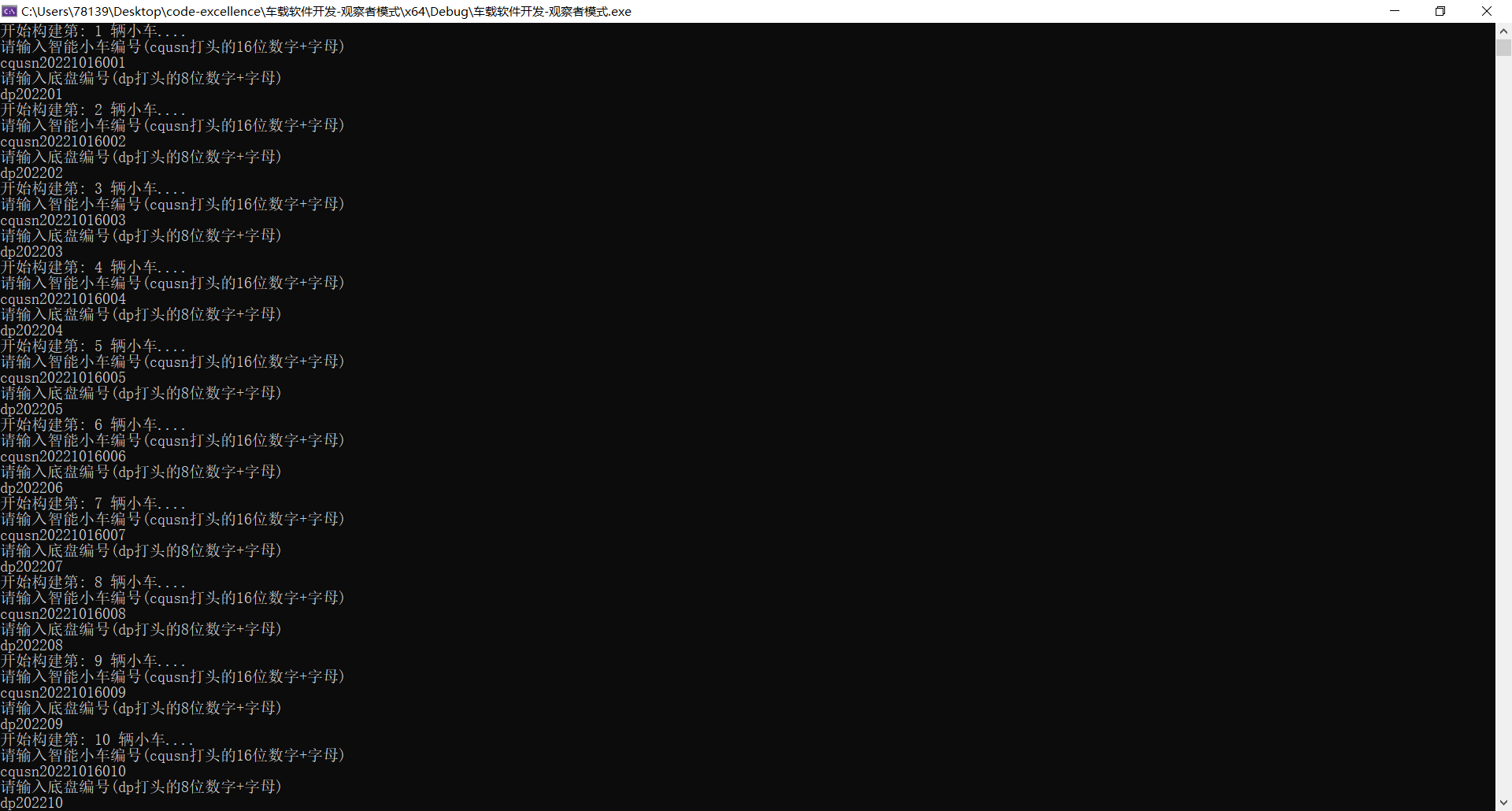
1. update：接收目标类传入的状态码并做出相应操作。

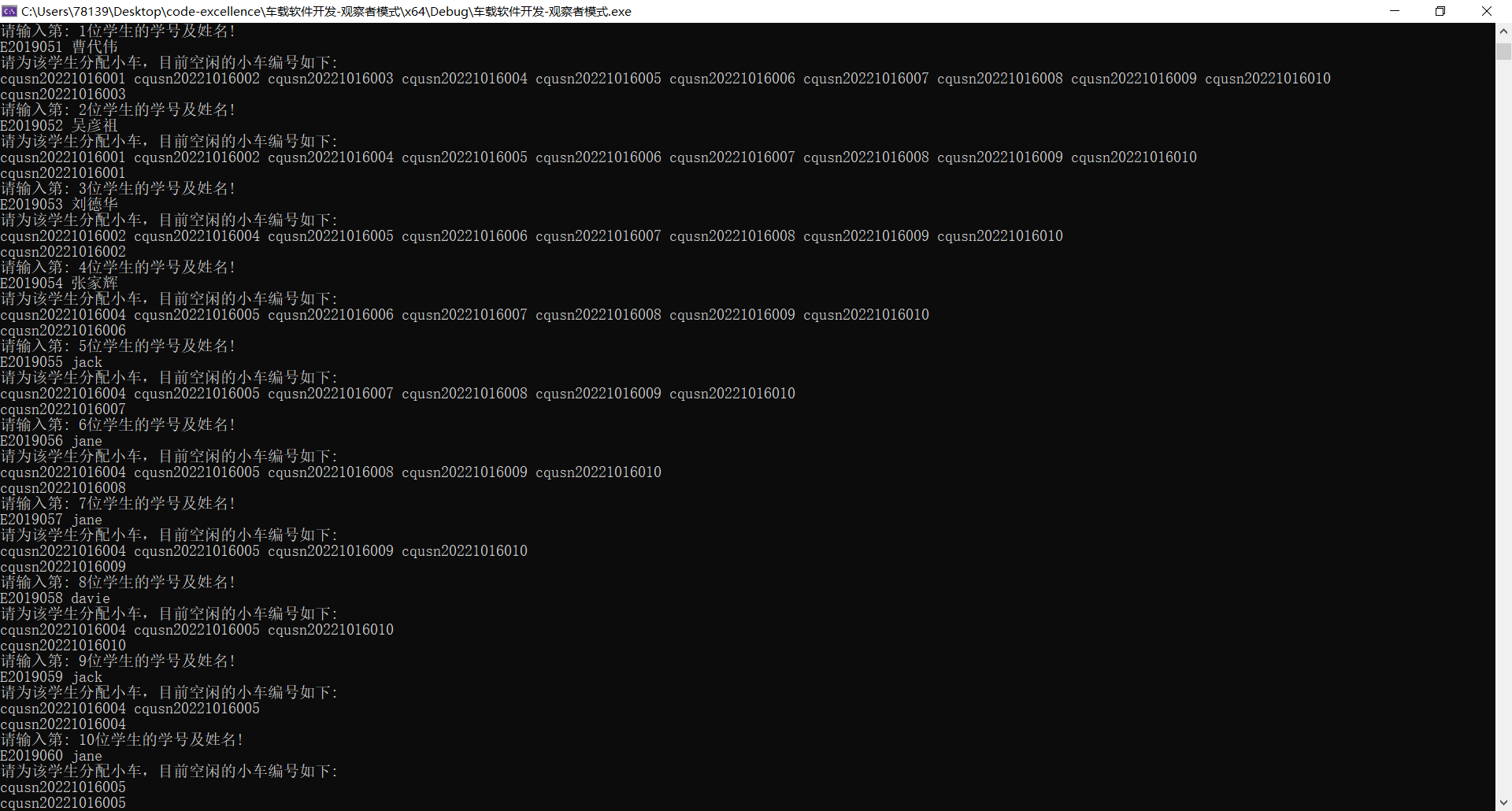


1. void startCar(Car car)：传入一辆Car对象，代表启动了这辆小车。然后程序不断监听用户输入的障碍物状态，小车的雷达组件接收到障碍物状态信息后便调用notifyObservers方法来通知所有监听它的观察者对象。

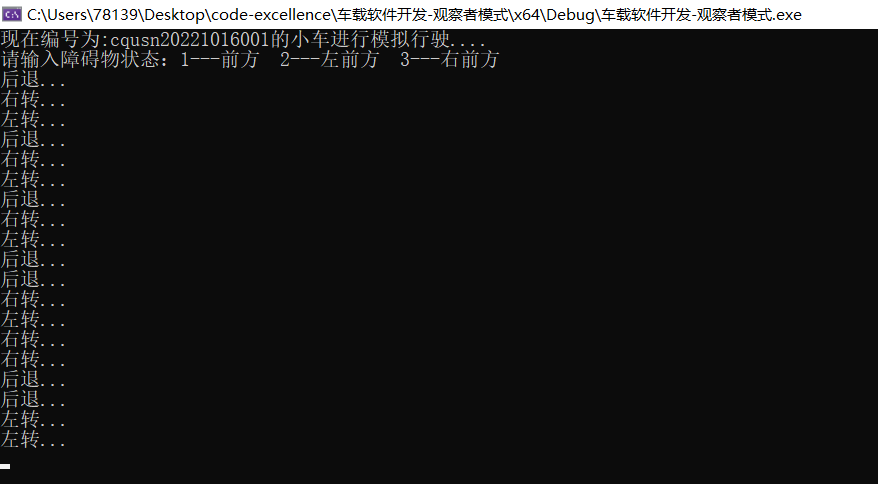


1. **程序效果**
2. 首先还是要初始化十辆小车，输入它们的编号，然后初始化十位学生，输入学号、姓名及分配给他的小车编号。





2.然后通过void startCar(Car car)方法启动一辆小车（默认启动第一辆），持续监听用户输入的障碍物状态并作出相应操作。



**三、总结和分析**

此次实验基本达到目标，小车的激光雷达在探测到前方障碍物后能够向订阅它的底盘组件发出障碍物状态信息，然后进行后退、转弯操作。最核心的是要理解观察者模式的特点，观察者模式又叫发布-订阅模式，其中的订阅表示这些观察者对象需要向目标对象进行注册，这样目标对象才知道有哪些对象在观察它。发布指的是当目标对象的状态改变时，它就向它所有的观察者对象发布状态更改的消息，以让这些观察者对象知晓。其中**MySubject**代表目标类，它是一个抽象类，也是所有目标对象的父类。它用一个列表记录当前目标对象有哪些观察者对象，并提供增加、删除观察者对象和通知观察者对象的接口。虽然此处只有一个目标类（激光雷达），但是为了考虑后期的可扩展性，还是应当设置这样一个基类供所有目标类继承。同理，**MyObserver**它也是一个抽象类，是所有观察者对象的父类；它为所有的观察者对象都定义了一个名为update的方法。当目标对象的状态改变时，它就是通过调用它的所有观察者对象的update方法来通知它们的。

**四、参考文献**

[1]详细解读设计模式：观察者模式.http://t.csdn.cn/XUqVc,2020-07-22.

[2]C/C++编写命名规范.http://t.csdn.cn/VzFcz,2022-05-21.

[3]C++ 保存数据的四种方式.http://t.csdn.cn/6g3xK,2021-03-20.

[4]C++面向对象设计原则详解.http://t.csdn.cn/KkfqZ,2020-03-29.

[5][美]Stephen Prata.C++ Primer plus[M].张海龙 袁国忠.北京:人民邮电出版社,2015:96-183.

[6]设计模式之观察者模式（C++）.www.cnblogs.com/carsonzhu/p/5770253.html,2019-07-14.