

车载软件开发基础

课后实践1



**2023至2024学年第 1 学期**

|  |  |
| --- | --- |
| 学号 | 姓名 |
| E2020196 | 侯加喜 |
| 任课教师 | 陈恒鑫 |
| 成 绩 |  |

|  |  |
| --- | --- |
| 任务书 | |
| 任务内容 | 某校购买了多台智能小车，并将其分配给同学们实验使用。请使用C++语言编写一个程序，实现对这批智能小车的信息录入及分配，并以文件方式保存相关资料。  具体资料如下：   1. 智能小车信息包括： 2. 编号：自定义，cqusn打头的16位数字+字母 3. 底盘    1. 编号：dp打头的8位数字+字母    2. 型号：SCOUT MINI    3. 轴距：451mm    4. 轮距：490mm    5. 最小离地间隙：115mm    6. 最小转弯半径：0m    7. 驱动形式：四轮四驱    8. 最大行程：10KM    9. 轮胎（4个）       1. 型号：公路轮、麦克纳姆轮       2. 尺寸：175mm 4. AGX套件（1个）    1. 型号：AGX Xavier    2. AI：32 TOPS    3. CUDA核心：512    4. Tensor CORE：64    5. 显存：32G    6. 存储：32G 5. 双目摄像头（1个）    1. 型号：RealSense D435i    2. 摄像头：D430    3. RGB帧分辨率：1920\*1080    4. RGB帧率：30    5. FOV：87\*58    6. 深度帧率：90 6. 多线激光雷达（1个）    1. 型号：RS-Helios-16p    2. 通道数：16    3. 测试范围：100m    4. 功耗：8W 7. 9轴陀螺仪（1个）    1. 型号：CH110    2. 厂家：NXP 8. 液晶显示屏（1个）    1. 尺寸：11.6    2. 型号：super 9. 电池模块（1个）    1. 参数：24V/15Ah    2. 对外供电：24V    3. 充电时长：2H 10. 学生信息包括： 11. 学号 12. 姓名   要求：   1. 共采购了10台，请先分别完成10台小车的信息录入，并完成编号。 2. 根据编号，将每台小车分配给每名同学（含学号和姓名） 3. 将相关信息以文件方式保存到本地硬盘中 4. 用程序打开3中保存的文件，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；  （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）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**

**1.1目录结构**

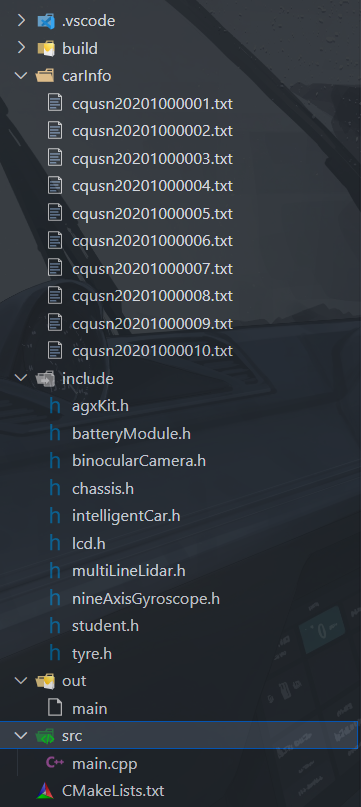
****

图 1.1-1 目录结构

**1.2数据结构**

根据题意可知，智能小车由底盘、Agx套件、轮胎、摄像头等部件组成，并且每位学生分配一辆智能小车。故首先设计了10个struct用于表示这些组件及学生，数值通过程序输入。具体数据结构如下：

小车整体数据结构INTELLIGENT\_CAR:

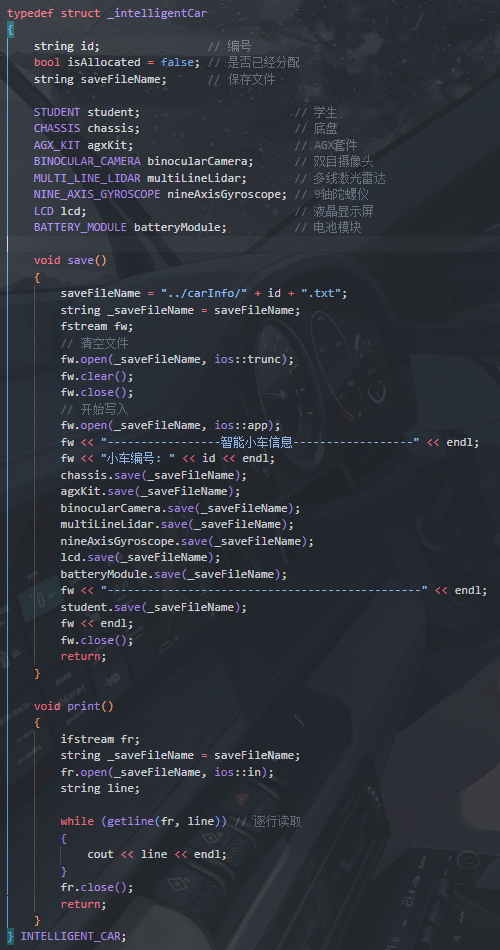


图1.2-1 INTELLIGENT\_CAR

底盘数据结构CHASSIS:

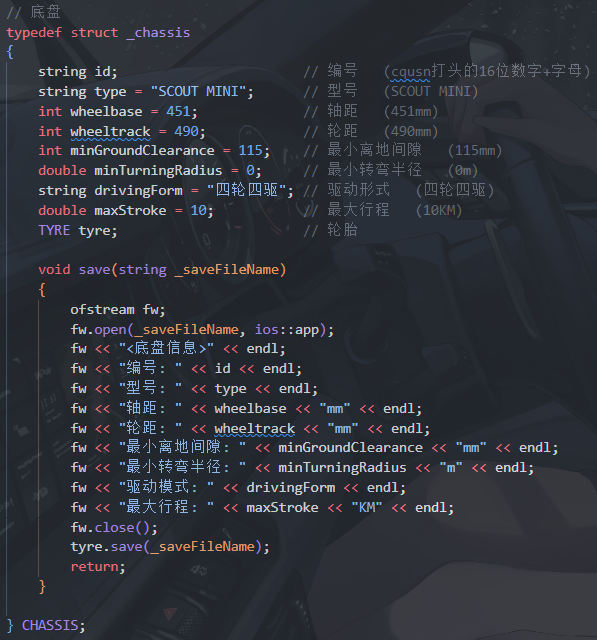


图 1.2-2 CHASSIS

其中轮胎的数据结构TYRE为:

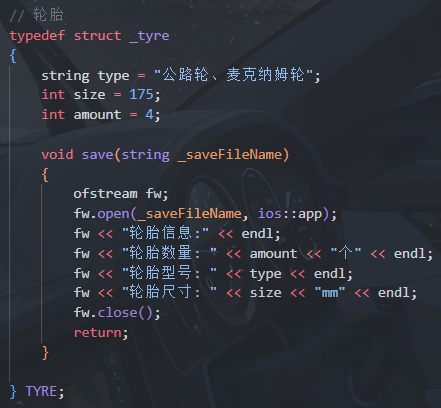


图1.2-3 TYRE

Agx套件数据结构:

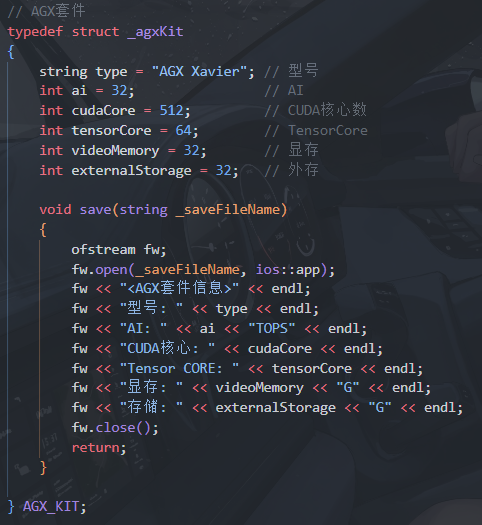


图 1.2-4 AGX\_KIT

双目摄像头数据结构:

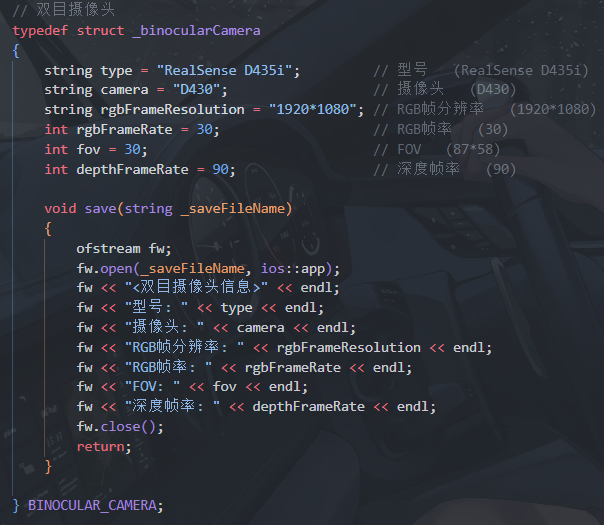


图 1.2-5 BINOCULAR\_CAMERA

多线激光雷达数据结构:

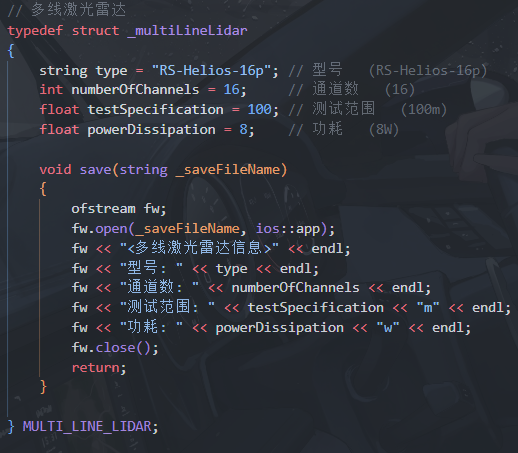


图 2.2-6 MULTI\_LINE\_LIDAR

九轴陀螺仪:

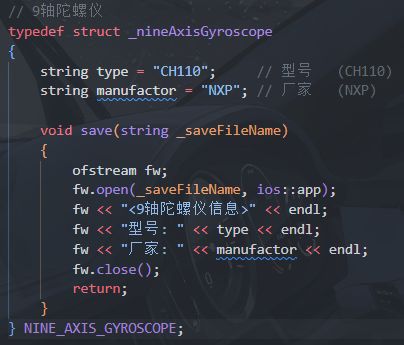


图1.2-7 NINE\_AXIS\_GYROSCOPE

液晶显示屏:

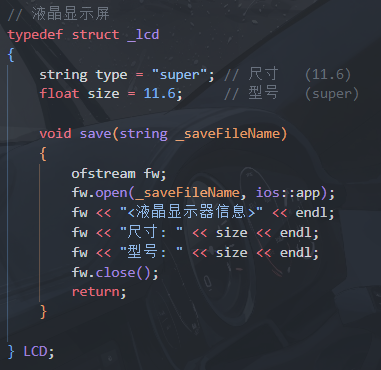


图 3.2-8 LCD

电池模块:

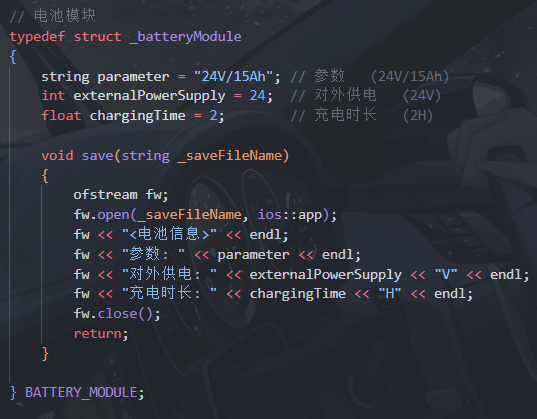


图 4.2-9 BATTERY\_MODULE

学生信息:

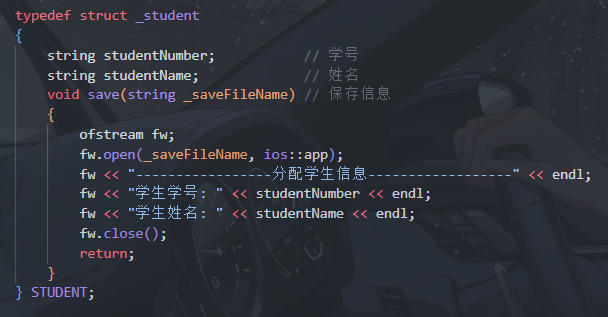


图1.2-10 STUDENT

**1.3关键函数**

(1).保存信息

通过结构体**INTELLIGENT\_CAR**中的**void save()**函数读入信息并将分配信息保存到硬盘中。该函数**依次调用**各个模块结构体中的save保存各自的信息。主要方式是通过ofstream流打开指定路径的文件，以追加模式进行写入。

void save()

{

saveFileName = "../carInfo/" + id + ".txt";

string \_saveFileName = saveFileName;

fstream fw;

// 清空文件

fw.open(\_saveFileName, ios::trunc);

fw.clear();

fw.close();

// 开始写入

fw.open(\_saveFileName, ios::app);

fw << "-----------------智能小车信息------------------" << endl;

fw << "小车编号: " << id << endl;

chassis.save(\_saveFileName);

agxKit.save(\_saveFileName);

binocularCamera.save(\_saveFileName);

multiLineLidar.save(\_saveFileName);

nineAxisGyroscope.save(\_saveFileName);

lcd.save(\_saveFileName);

batteryModule.save(\_saveFileName);

fw << "-----------------------------------------------" << endl;

student.save(\_saveFileName);

fw << endl;

fw.close();

return;

}

(2).读取信息

通过结构体**INTELLIGENT\_CAR**中的**void print()**函数将硬盘中的文本数据读取到控制台，首先通过ifstream流指定文件路径，以ios::in模式打开，按行读取数据不断存储到string buffer中。

void print()

{

ifstream fr;

string \_saveFileName = saveFileName;

fr.open(\_saveFileName, ios::in);

string line;

while (getline(fr, line)) // 逐行读取

{

cout << line << endl;

}

fr.close();

return;

}

1.2.3最后，在**int main()**函数通过while循环不断监听用户输入，当输入n时，读取下一个学生信息，输入p时，读取上一个学生信息。当然，当展示第一组数据时，输入P无效。处于最后一组数据时，输入N无效，以防止数组下标越界异常。

#include <iostream>

#include <fstream>

#include <string>

#include <iomanip>

using namespace std;

#include "student.h"

#include "tyre.h"

#include "chassis.h"

#include "agxKit.h"

#include "binocularCamera.h"

#include "multiLineLidar.h"

#include "nineAxisGyroscope.h"

#include "lcd.h"

#include "batteryModule.h"

#include "intelligentCar.h"

int main()

{

int nums;

cout << "请输入学生人数： ";

cin >> nums;

STUDENT students[nums];

INTELLIGENT\_CAR intelligentCars[nums];

for (int i = 0; i < nums; ++i)

{

cout << "请输入第" << i + 1 << "位学生学号: ";

cin >> students[i].studentNumber;

cout << "请输入第" << i + 1 << "位学生姓名: ";

cin >> students[i].studentName;

}

system("clear");

for (int i = 0; i < nums; i++)

{

cout << "<开始构建第" << i + 1 << " 辆小车>" << endl;

cout << "请输入小车编号: ";

cin >> intelligentCars[i].id;

cout << "请输入底盘编号: ";

cin >> intelligentCars[i].chassis.id;

intelligentCars[i].isAllocated = true;

intelligentCars[i].save();

}

system("clear");

cout << "小车分配完成" << endl;

cout << "按y查看小车信息,按n退出: ";

char ch;

cin >> ch;

if (ch == 'y')

{

intelligentCars[0].print();

}

else

{

return 0;

}

for (int i = 0; true;)

{

cout << "按n查看下一辆小车,按p查看上一辆小车，否则退出: ";

cin >> ch;

switch (ch)

{

case 'n':

if (i == nums - 1)

{

cout << "无下一辆小车,请重新输入：" << endl;

continue;

}

else

{

++i;

}

break;

case 'p':

if (i == 0)

{

cout << "无上一辆小车,请重新输入：" << endl;

continue;

}

else

{

--i;

}

break;

case 'q':

cout << "查看小车信息结束" << endl;

return 0;

}

system("clear");

intelligentCars[i].print();

}

return 0;

}

1. **程序效果展示**

**2.1编译源代码**

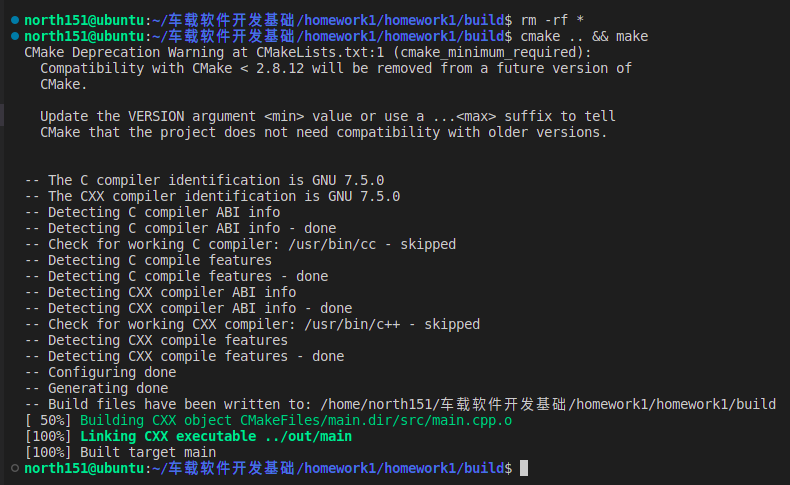


图 2.1-1 编译源代码

**2.2录入效果展示**

程序首先输入学生人数,学生姓名及其学号:

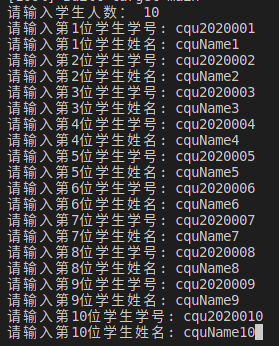


图 2.2-1 输入学生信息

然后开始构建10辆小车:

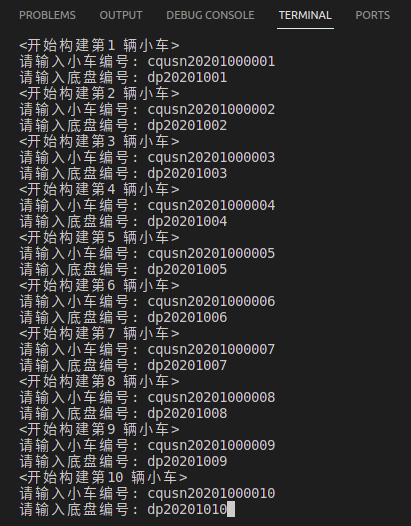


图2.2-2 构建小车

**2.3保存效果展示**

信息录入完毕后,将通过save()函数将分配信息保存至本地硬盘文件中，形式如下：

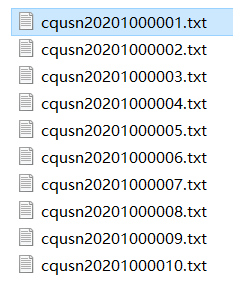


图 2.3-1 carInfo

文件内容:



图 2.3-2 cqusn20201000001.txt

**2.4读取效果展示**

完成10辆小车的构建后，开始读取通过void print()函数读取之前保存到硬盘的文本文件，打印输出到命令行窗口，并持续监听用户输入，按下N展示下一组数据，按下P展示上一组数据。小车信息。

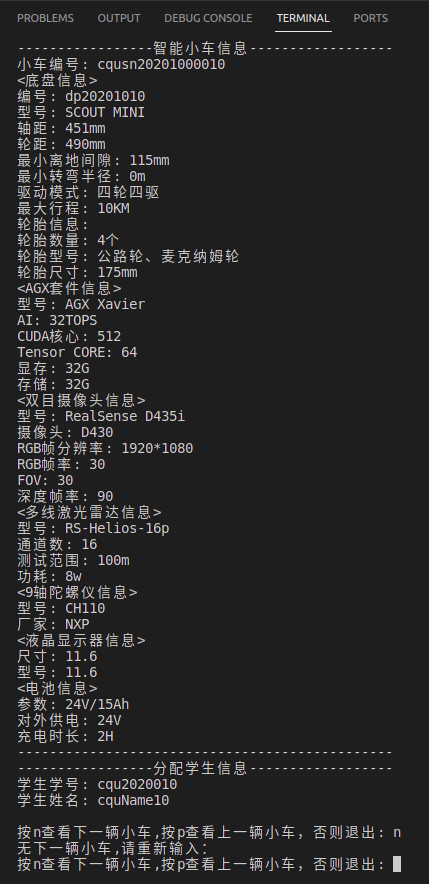


图2.4-1 读取效果展示

1. **总结和分析**

[Github](https://github.com/north151/carTask): https://github.com/north151/carTask

[Gitee](https://gitee.com/north151/homework1): https://gitee.com/north151/homework1

此次报告采用面向过程的思想进行编写，面向过程的最大特点就是分析出解决问题所需要的步骤，然后用函数把这些步骤一步一步实现，使用的时候一个一个依次调用就可以了。其中需要频繁使用结构体struct，它是一种值类型，通常用来封装小型相关变量组，例如CHASSIS就封装了底盘的编号、型号、轴距、驱动方式等相关性很强的变量。struct的优点是性能高，因为不需要像class一样进行实例化。缺点就是复用性、扩展性、可维护性都较差，一般struct只会用于存储结构化数据，虽然在C++中struct也可以像class一样实现继承，但Google编程规范中明确指明不应该这样做。此外,为方便调用,本次实验在car.h和student.h中分别声明与小车和学生信息有关的结构体。虽然通过面向过程思想也较好地完成了此次任务，但代码的复用性与扩展性始终不尽人意，这也是struct结构体特性所限制的，未来可以通过面向对象的思想进一步改进！

1. **参考文献**

[1]面向对象和面向过程的区别详解.http://t.csdn.cn/8cuvP,2022-03-28.

[2]struct结构体详述.http://t.csdn.cn/kuW63,2020-04-21.

[3]C++中struct与class的区别.http://t.csdn.cn/MZY0A,2021-09-20.

[4]class和struct的优缺点.https://www.jianshu.com/p/cc626dad2bac,2022-01-03.

[5][美]Stephen Prata.C++ Primer plus[M].张海龙 袁国忠.北京:人民邮电出版社,2015:20