Life After a CS Degree

Your Professor is fed up with the demands of academic life. Students who skip class and too many department meetings that should have been emails have caused him to quit and go into business for himself. He always loved cars, so he decided to open a car lot.

However, he doesn't want to completely quit programming, so he decides to program the inventory management system himself. He is feeling powerful and excited after quitting his job and going into business for himself, so he decides to use C++. He isn't a C++ pro though, and he has run into some issues. Fortunately, he still knows a lot of very bright students who can help him.

Problem Number One

Your professor starts with the code in the repo. Read this code **before you start this project**. Make sure you understand it completely before you begin. Print it out, and bring the questions to class.

Compile the code. Use the command

```
g++ *.cpp --std=c++11
```

Run it with `./a.out`.

Your professor noticed that all of the Cars have the same id. This is not what we want; each Car should have a unique id, and we should be incrementing the id each time we create a new Car. He wants to understand why this is happening. He emailed you these questions. Please answer them. Turn in a document with your answers with your submission.

1. "I am using a static variable to store the id number, and I know I have that setup correctly. Since the variable is static, every instance of Car will share the same one. Therefore, something else is going on. I am adding one each time I call the constructor, so why do all of my Cars have the same value?"

2. Modify the code to fix the issue. Don't worry if numbers are being skipped for now, just make sure each Car is getting a unique id.

Problem Number Two

If you got this far, you must have remembered that objects are created in different ways (notice I'm not saying in different places, i.e. the stack and the heap - though that happens as well). Now, the problem is trades are not happening correctly. The function that processes the trade is getting called, but afterward the inventory remains the same. This won't do; we need to remove the car we sold and add the one we took in. Your prof emails you again.

3. "Why isn't my function working? When I put print statements in my function to prove everything was working correctly, it all seemed fine. However, after the function finishes I'm left with the same inventory I started with."

4. Modify the code to fix the issue.

Problem Number Three

It is driving your prof crazy that the numbers of the Car ids aren't correct. As it stands, every time a copy is made (even temporary ones), the id is incremented. This means that id numbers seem to be skipped sometimes. Your prof is going crazy; this doesn't seem deterministic, and he can't stop thinkin about it. He emails you again.

5. "Why are some id numbers getting skipped? Also, it seems wasteful and inefficient to pass all these Cars around by copy. Can C++ do pass by reference? How can we make this work with my inventory vector? I know vectors can only store objects and not references. I don't want to deal with pointers though. I learned C++ 35 years ago; has anything new come out that can help us fix this?"

6. Modify the code to fix the issue.

Problem Number Four

If you got this far, well done. Your prof finally has a usable system. However, he wants you to extend it in a few ways. He wants to be able to compare the cost of ownership of vehicles, and to sort vehicles accordingly. To do this, you will need to override the `operator<` function for the Car class. The signature looks like

```
bool operator<(const Car& other)
```

A Car is considered less than another Car if the value of the ServiceRecords is less than the value of the ServiceRecords for the other. So, if Car a had two records totalling $27.43 and Car b had three records totalling $54.32, `a < b` would return "true". Add this code to your Car class.

Finally, create five different cars. Put them all in a vector. Use the `std::sort` function to sort them with your `operator<`.

Submission

You may work on this with **1 other person**. Additionally, you **may** use ChatGPT or other helper with getting your code working correctly. You **may not** ask ChatGPT to answer any of the questions for you. I want you to reason about the problems your professor is having. If you have multiple theories, list them, and list what you tried and didn't work.

I don't care that you write perfect C++ code. I want to see that you learned how C++ handles memory, and that you understand the related issues you may encounter when programming in this language. While I will grade your code, the answers you provide to the questions are more important to me for assessing your learning.

Please submit **only** your Car.cpp, Car.h, and main.cpp files (unzipped) to Blackboard. **Put all group members names on the submission and in the code.**