CIS 246 – C++ Programming

Program:	**Week 07**
Points:	**20**
Chapter(s):	**7**
File(s) to Submit:	**InventoryDriver.cpp, Car.h (as one zip file)**

Summary

Write a C++ application to handle adding, deleting, and printing cars from a dealer inventory, using the C++ Standard Library class template **vector**.

Description

The JJC Car Dealership needs an inventory management system. Your application will display a menu that prompts the user to add or delete cars from inventory, to print all the car information, or to exit the application.

When printing the inventory, display the car makes and prices in a nicely formatted, aligned chart.

Your code does not have to handle duplicate car types. Test it without using duplicates.

The menu should continue to redisplay until the user chooses to exit the program.

Use the example code file **fig07_21.cpp** as a reference for working with vectors.

Learning Objectives

In this assignment, you will practice:

- Using the vector class template to store objects
- Using the range-based for repetition statement
- Using the appropriate repetition structure to display a menu.

Requirements for the Car Class

Data Members

1. string variable to hold the make of car (Ford, Honda, etc.)
2. integer variable to hold the price

Member Functions

1. **Two-argument constructor**
2. **Getter** functions – one for each data member
3. **No setter functions** are needed for this class.

Requirements for the InventoryDriver Program

1. **main**

   The main function creates the vector using the Car class, and then calls the displayMenu function to display the menu.

   Create the vector without a size, like this: **vector<Car> nameOfVector;**

   Main must use a **sentinel-controlled loop**. Within the loop, call the displayMenu function, and read the user's choice from stdin. Still within the loop, use a **selection statement** to call the appropriate add, delete, or print function based on their choice. Then redisplay the menu by calling displayMenu.

   Be sure to use the most appropriate type of repetition statement for displaying menus.

2. **displayMenu**

   Parameters: none
   Return value: none

   This function displays the menu, which should look like this:

   1. Add a car to inventory
   2. Delete a car from inventory
   3. Print inventory
   4. Exit

   Your choice:

   Note that displayMenu should NOT read the user's choice. It should **only display**. Main will read user input.

3. **addData**

   Parameters: one vector of Car objects
   Return value: none

   This function prompts the user for the type and price of the car to be added. Create a new Car object with this information, and then add this this new Car object to the vector by calling the push_back() function.

4. **deleteData**

   Parameters: one vector of Car objects
   Return value: none

   This function prompts the user for the type of car to delete. This information is entered as a string. Use a **standard counter-controlled for loop to iterate through the vector** until a match is found. Remember, each element in the vector is a Car object.

To find a match, compare the string entered by the user to the string saved in the Car object by calling the **compare** function. The compare function returns a 1 if the comparison is true, and a 0 if the comparison is false.

For example, if carToDelete is the string entered by the user, and you are comparing it to the make of the Car object stored in the first element in a vector called vectorName, then the comparison for equality would look like this:

if (carToDelete.compare(vectorName[0].getMake() == 1)

The comparison for inequality would look like this:

if (carToDelete.compare(vectorName[0].getMake() == 0)

If a match is found, call the erase() function of the vector. If a match is not found, display a warning message.

The erase() function uses the following syntax:

***nameOfVector*.erase(*nameOfVector*.begin() + *the subscript of the Car to be deleted*)**

For example, if the car to be deleted from inventory is at index 6 in the vector, and the vector is called cars, the erase function is called like this:

cars.erase(cars.begin() + 6);

5. **printData**
   Parameters: one vector of Car objects
   Return value: none

   This function uses the **range-based for loop syntax to iterate through the vector** and displays each car's information in a neatly formatted list.

Requirements for Full Credit on This Project

COMPLETE AND ACCURATE – Your program must compile, execute, and give accurate output.

FOLLOW ALL REQUIREMENTS ACCORDING TO THE INSTRUCTIONS – Follow the instructions as written for completing this project, even if you [think you] know a "better" way to do something.

COMMENTS – Include comments in your code. There must be a comment at the top of each source code or header file that includes your name, the assignment number, and a description of the code in that file. There must be comments at each important step in your algorithm that describes that step.

BEST PRACTICES – Follow best practices in C++ programming, including, but not limited to, appropriate use of private/public, appropriate use of classes and/or header files, sets & gets, white space, alignment, meaningful variable names, naming conventions, using statements, etc. Points will be deducted for sloppy code that is hard to read, even if it works, so pay attention to these details.

SUBMIT ALL FILES BEFORE THE DUE DATE – See the due date for this assignment on the course calendar in Canvas and review the submission requirements below.

ALL SUBMISSIONS MUST BE YOUR OWN WORK – Review the syllabus regarding plagiarism and the Joliet Junior College Academic Honor Code. Review the Get Help page in every module if you need help with this assignment.

Submission Requirements

All source code file submissions must be compressed into a Compressed Folder, or zip file. The dropbox for this assignment will only allow compressed files (.zip extension) to be uploaded.

Instructions for creating a Compressed Folder (zip file) are available here: https://support.microsoft.com/en-us/help/14200/windows-compress-uncompress-zip-files

Do not submit your code in any other file format but .cpp and .h (within a zip). I will not accept links to online storage. You must submit the actual file. Do not submit executable files. Do not submit project files from an IDE.

Sample Run

```
Welcome to the JJC Car Dealership Inventory Program

1. Add a car to inventory
2. Delete a car from inventory
3. Print inventory
4. Exit

Your choice: 1

Type of car: Ford
Price of car: 21000


1. Add a car to inventory
2. Delete a car from inventory
3. Print inventory
4. Exit

Your choice: 1
```

```
Type of car: Honda
Price of car: 23000


1. Add a car to inventory
2. Delete a car from inventory
3. Print inventory
4. Exit

Your choice: 3

Make                    Price
----                    -----
Ford                   21,000
Honda                  23,000


1. Add a car to inventory
2. Delete a car from inventory
3. Print inventory
4. Exit

Your choice: 2

Type of car to delete: Ford
Ford deleted from inventory


1. Add a car to inventory
2. Delete a car from inventory
3. Print inventory
4. Exit

Your choice: 3

Make                    Price
----                    -----
Honda               23,000


1. Add a car to inventory
2. Delete a car from inventory
3. Print inventory
4. Exit

Your choice: 2

Type of car to delete: VW
```

```
No such car in inventory


1. Add a car to inventory
2. Delete a car from inventory
3. Print inventory
4. Exit

Your choice: 4
```