Project 2: Sunset Auto

CPSC 131 Fall 2020

|  |  |
| --- | --- |
| Image taken from https://commons.wikimedia.org/wiki/File:Mechanic\_and\_Customer.jpg | Image taken from https://www.pickpik.com/car-repair-car-workshop-repair-shop-garage-repairs-car-32501 |

### Introduction

Welcome to Sunset Auto, the automotive expert in Orange County. Since 1998 we've provided the Orange County area with reliable, affordable automotive maintenance and repair services. Sunset Auto offers auto repair services you can count on. While our motto is “first come, first serve” we also allow advance appointments, and we offer priority services to our loyal customers. We have experience servicing all makes and models of domestic and import vehicles. We even provide a shuttle service to get you to and from your home or work!

Your project is about maintaining a daily list of repairs requested by customers for their vehicles. Since we do not know how many repairs we will have in a day, we will use a linked list to maintain the list. For each repair requested we keep track of: the customer and vehicle data, the date and time when the repair was requested, the services requested, and the estimated cost of the repair.

You are in charge of implementing this linked list using the STL container class list.

**Usage of container class std::list is a must**. **It is already provided in class definition.**

As repairs are being made by our professional mechanics, you will keep an iterator to track the current repair request to be serviced.

For the advance appointments, you will load in the list a number of repairs from a given CSV file. For the first come, first serve customers, you will append the repair request to the end of the list. For our loyal customers, you will insert the repair request right after the current repair in the list, aka becoming the next to be serviced after the current repair.

This project is to reinforce the new data structure learnt, doubly linked list.

### Objective

You are given two header files (containing classes and their implementations): Repair.hpp and RepairList.hpp, one monday.csv file, and one main.cpp file. **The file RepairList.hpp needs to be completed.**

Repair.hpp holds information about each repair request: the customer and vehicle data, the date and time when the repair was requested, the services requested, and the estimated cost of the repair. **Repair.hpp is already complete.**

RepairList.hpp keeps track of all repair requests in a given day and maintains an iterator pointing to the current repair request. **RepairList.hpp is incomplete and you should add the implementation (code) for all the function members where it says : TO DO**.

Complete the RepairList.hpp, and make sure all tests pass. Your code is tested in the provided main.cpp.

For RepairList.hpp:

**Data members are already provided:**

**std::list<Repair> \_dailyList;**

**std::list<Repair>::iterator \_nowServicing;**

You will need to implement (write the code for) the following function members of the class RepairList:

* **std::size\_t size():** returns the number of repair requests in the list
* **Repair currRepair()**: returns the current repair request
* **void next()**: go to the next repair request in the list, if any
* **void prev()**: go to the previous repair request in the list, if any
* **void addToList(const Repair& newRequest)**: append a new repair request to the end of the list and it does not change the current position in the list
* **void insertLoyal(const Repair& newRequest)**: insert a new repair request after the current position of the list, but it does not change the current position in the list

Initially the given code may not compile or not execute. As you complete the code, the tests should start to pass in main.cpp.

### Source Code Files

You are given “skeleton” code files with declarations that may be incomplete and without any implementation. Implement the code and ensure that all the tests in main.cpp pass successfully.

* LICENSE: This is completed. It contains the MIT license that allows you to use the instructor’s provided code as the skeleton code.
* Repair.hpp: This file is completed
* **RepairList.hpp: This file needs to be completed**
* main.cpp: The main function tests the output of your functions. You may wish to add additional tests. During grading your main.cpp file will be replaced with the one you were provided with.

### Obtaining and submitting code

The skeleton code is available in Project 2 folder in Titanium. When you have completed the project, upload the completed file in the Project 2 folder.

#### Submitting the project

**Only submit the following one file on Titanium in the link provided for Project 2:**

**1) RepairList.hpp**

### Testing (either of the two below)

1. **On Tuffix:**

Use the following command to compile your program:  
**clang++ -g -std=c++17 main.cpp -o test**  
  
To attempt to run the compiled test program, use the following command:  
**./test**

##### **You can use Visual Studio:**

Free download: Community 2019

<https://visualstudio.microsoft.com/downloads/>

### You can write, debug, run your project in Visual Studio.

### Grading rubric

Your grade depends on how correct the code is. While passing tests is a good indicator, I still look at the code mark on the basis of how correct it is.

**You absolutely HAVE to use std::list : It is already provided in class definition.**

**No points for using anything else**

### Deadline

The project deadline is **Nov 1, 2020**, 11:59 PM.

**Your code must compile/build for it to be tested and graded. If you only complete part of the project, make sure that it compiles before submitting.**

### Blurb for your resume

Use your GitHub account as a ready-to-show portfolio of your programming projects to potential employers. You can also add this blurb to your resume after successfully completing this project:

*Completed from a point-of-departure baseline the implementation of a C++ program that reads car repair requests from a CSV file, adds and inserts requests as they are made by ordinary or loyal customers, and then schedules them for repairs.*