COMP 2404
Introduction to Software Engineering
Assignment 4

Released: Monday March 12, 2018, 12:00 Noon

Due: Monday March 26, 2018, 12:00 Noon

Instructions

In this assignment you will build upon the program worked on in the previous assignment. You are free to build upon your solution or start from the skeleton code provided.

Task 1: Overloading Operators

In this part of the assignment we will go through and overload some operators for some of the classes in this program. Some of these overloaded operators will replace some functions that are already present.

Vehicle Class

- Overload the **<<** operator so that it prints the contents of the **Vehicle** object in the same manner that it is printed when the “Print Customer Database” option is selected from the main menu of the program.
- Overload the **<** and **>** operators to compare **Vehicle** objects based on their year. Change the code that compares **Vehichle** objects in the **VehicleList** class during insertion to use these operators instead of comparing the **Vehicle** objects’ years directly.

VehicleList Class

- Overload the **+=** operator that takes a **Vehicle** pointer and adds it to the **VehicleList**. Once implemented, remove the **add** function and fix any other code in the program to use this operator instead of that function.
- Overload the **-=** so that it takes a **Vehicle** pointer and removes it from the **VehicleList**. If the **Vehicle** does not exist in the **VehicleList** simply do nothing. This is new functionality for this program that we will use later in this assignment.

- Overload the **[]** operator so that it takes an integer subscript as a parameter and returns a pointer to the **Vehicle** object at that position in the list. If the specified index is invalid, return null.
- Overload the **<<** operator so that it prints the contents of the **VehicleList**. You are to do this by iterating over the list using your newly implemented **[]** operator. You are not allowed to make this operator a friend to the **Node** class, only to the **VehicleList** (so you have to use your **[]** operator). Be sure to make use of the **<<** operator you implemented in the **Vehicle** class. Once this is completed, remove the **toString** function. Update any other code in the program to now use this operator instead.

Customer Class

- Overload the **+=** operator that takes a **Vehicle** pointer and adds it to the **VehicleList**. This should make use of the **+=** operator now available in the **VehicleList** class. Once implemented, remove the **addVehicle** function and fix any other code in the program to use this operator instead of that function.
- Overload the **<<** operator so that it prints the contents of the **Customer** object in the same manner that it is printed when the “Print Customer Database” option is selected from the main menu of the program. In this operation, make use of the **<<** operator now available to you from the **Vehicle** class.
- Overload the **<** and **>** operators to compare **Customer** objects based on their last name. Change the code that compares **Customer** objects in the **CustomerList** class during insertion to use these operators instead of comparing the **Customer** objects’ last name directly.

CustomerList Class

- Overload the **+=** operator that takes a **Customer** pointer and adds it to the **CustomerList**. Once implemented, remove the **add** function and fix any other code in the program to use this operator instead of that function.
- Overload the **-=** so that it takes a **Customer** pointer and removes it from the **CustomerList**. If the **Customer** does not exist in the **CustomerList** simply do nothing. This is new functionality for this program that we will use later in this assignment.
- Overload the **[]** operator so that it takes an integer subscript as a parameter and returns a pointer to the **Customer** object at that position in the list. If the specified index is invalid, return null.
- Overload the **<<** operator so that it prints the contents of the **CustomerList**. You are to do this by iterating over the list using your newly implemented **[]** operator. You are not allowed to make this operator a friend to the **Node** class, only to the **CustomerList** (so you have to use your **[]** operator). Be sure to make use of the **<<** operator you implemented in the **Customer** class. Once this is completed, remove the **toString** function. Update any other code in the program to now use this operator instead. You

should now be able to run your program and when selecting “Print Customer Database”, the output should look more or less the same as it did before.

Shop Class

- Overload the **+=** operator that takes a **Customer** pointer and adds it to the **CustomerList** stored in the **Shop**. This should make use of the **+=** operator defined in the **CustomerList** class. Once implemented, remove the **addCustomer** function and fix any other code in the program to use this operator instead of that function.
- Overload the **-=** so that it takes a **Customer** pointer and removes it from the **CustomerList** stored in the **Shop**. This should make use of the **-=** operator defined in the **CustomerList** class. This is new functionality for this program that we will use later in this assignment.

Task 2: Adding to the Main Menu

We will now add two new options to our main menu: **4. Remove Customer** and **5. Remove Vehicle**. These functions will work similarly to the respective add functionality and will maintain the design principles of this program. You will need to implement functionality in a few classes to achieve this.

If the user chooses **Remove Customer**, they are prompted for a Customer ID. If they enter an invalid ID, this is reported and nothing happens. Otherwise, the **Customer** identified is removed.

If the user chooses **Remove Vehicle**, they are prompted for a Customer ID. If they enter an invalid ID, this is reported and nothing happens. Otherwise, the user is prompted to indicate which vehicle (a number from 0 - the number of vehicles the customer has registered -1). If they enter an invalid number, this is reported and nothing happens. Otherwise the **Vehicle** identified is removed.

In both cases the objects should be completely destroyed and all allocated memory associated with those objects should be freed.

Task 3: Introducing a Class Hierarchy

We will now introduce a relatively simple class hierarchy into this program. We will create a **Person** class that has the following members: **firstName**, **lastName**, **address** and **phoneNumber**. It will also have a constructor that is responsible for initializing these members and a getter function for each member. It is up to you whether you make these members private, protected or public, keeping with the idea of encapsulation we have been talking about in class.

Once implemented, you will update the **Customer** class to inherit from the **Person** class properly.

Task 4: Introducing the Mechanic Class

On top of keeping track of our customers, we also want to keep track of our Mechanics. Implement a **Mechanic** class which inherits from the **Person** class. On top of the inherited members, the **Mechanic** class needs to have the following:

- A constant integer ID member. This ID is taken from a static integer declared in the class which is initialized to start at 5000, and is incremented every time a new **Mechanic** is created, similar to the id in the **Customer** class.
- A **salary** data member, stored as an integer.
- A constructor which initializes the object properly.
- Overloaded **<<** operator which prints a **Mechanic** object in a similar manner to a **Customer** object, replacing the **Vehicles** with the **salary**.

Once this is created, add an array of pointers for **Mechanic** objects to the **Shop** class. Since our shop is relatively small, this will have a fixed size of 5. Overload the **+=** operator in the **Shop** class to take a pointer to a **Mechanic** object and add it to this array, if there is room. If there isn't, do nothing.

In the **initCustomers** function in the **ShopController** class, create at least 3 **Mechanic** and add them to the **Shop** object using your **+=** operator. Keep in mind that these objects should be dynamically allocated and you will therefore decide where and when this memory needs to be free. We will rename this function **initShop** which better aligns with its new functionality.

We will now add one more option to our main menu: **6. Print Mechanics**. This functions will work similarly to the **Print Customer Database** option and will maintain the design principles of this program. You will need to implement functionality in a few classes to achieve this.

Constraints

- your program must not have any memory leaks (don't worry if valgrind reports that some memory on the heap is "still reachable")
- do not use any global variables
- your program must reuse functions everywhere possible
- your program must be thoroughly commented
- **your program must compile and run in COMP2404-2406- W18 Virtual Machine**

Submission

You will submit in cuLearn, before the due date and time, one **tar** file that includes all the following:

- all source code, including the code provided
- a readme file that includes:
    - a preamble (program author, purpose, list of source/header/data files)
    - the exact compilation command
    - launching and operating instructions

Grading [out of 50 marks]

Marking components:

- 28 marks: operator overloading
    - **Vehicle** class
        - 2 marks: <<
        - 2 marks: < and >
    - **VehicleList** class
        - 2 marks: +=
        - 2 marks: -=
        - 2 marks: []
        - 2 marks: <<
    - **Customer** class
        - 2 marks: +=
        - 2 marks: <<
        - 2 marks: < and >
    - **CustomerList** class
        - 2 marks: +=
        - 2 marks: -=
        - 2 marks: []
        - 2 marks: <<
    - **Shop** class
        - 2 marks: += and -=
- 8 marks: **4. Remove Customer** and **5. Remove Vehicle** Menu options
    - 4 marks: Remove Customer
    - 4 marks: Remove Vehicle
- 3 marks: **Person** class
- 2 marks: **Customer** class
- 4 marks: **Mechanic** class
    - 2 marks: overall consturction
    - 2 marks: <<
- 2 marks: **Mechanic** array in **Shop** class
- 3 marks: **6. Print Mechanics** Menu option

Notes:

In order to get credit for a marking component, the program must prove that the marking component executes successfully. This is usually accomplished by printing out correct data.

Deductions

- Packaging errors:
  - 10% for missing readme
- Major programming and design errors:
  - 50% of a marking component that uses global variables
  - 50% of a marking component that consistently fails to use correct design principles, including separate functions
  - 50% of a marking component where unauthorized changes have been made to provided code or prototypes
- Minor programming errors:
  - 10% for consistently missing comments or other bad style
  - 10% for consistently failing to perform basic error checking
- Execution errors:
  - 100% of a marking component that can't be tested because the code doesn't compile or execute in the VM
  - 100% of a marking component that can't be tested because the feature isn't used in the code
  - 100% of a marking component that can't be proven to run successfully because data is not printed out

Sample Console Output

User input highlighted.

```
> make
g++ -c main.cc
g++ -c ShopController.cc
g++ -c View.cc
g++ -c Shop.cc
g++ -c CustomerList.cc
g++ -c VehicleList.cc
g++ -c Customer.cc
g++ -c Vehicle.cc
g++ -c Mechanic.cc
g++ -c Person.cc
g++ -o mechanicshop main.o ShopController.o View.o Shop.o CustomerList.o VehicleList.o
Customer.o Vehicle.o Mechanic.o Person.o
> ./mechanicshop



      **** Toby's Auto Mechanic Information Management System ****

                                MAIN MENU

      1. Print Customer Database
```

```
       2. Add Customer

       3. Add Vehicle

       4. Remove Customer

       5. Remove Vehicle

       6. Print Mechanics

       0. Exit

Enter your selection:  1

CUSTOMERS:

Customer ID 1001

       Name:                           Abigail Atwood
       Address:                        43 Carling Dr.
       Phone Number:                   (613)345-6743

       1 vehicle(s):

      Green 2016   Subaru Forester (40000km)


Customer ID 1002

       Name:                           Brook Banding
       Address:                        1 Bayshore Dr.
       Phone Number:                   (613)123-7456

       2 vehicle(s):

      White 2018    Honda Accord (5000km)
      White 1972 Volkswagon Beetle (5000km)


Customer ID 1004

       Name:                           Eve Engram
       Address:                        75 Bronson Ave.
       Phone Number:                   (613)456-2345

       3 vehicle(s):

       Blue 2017    Toyota Prius (10000km)
       Gold 2015    Toyota Rav4 (20000km)
```

Green 2013 Toyota Corolla (80000km)

Customer ID 1003

Name: Ethan Esser
Address: 245 Rideau St.
Phone Number: (613)234-9677

1 vehicle(s):

Black 2010 Toyota Camery (50000km)

Customer ID 1000

Name: Maurice Mooney
Address: 2600 Colonel By Dr.
Phone Number: (613)728-9568

1 vehicle(s):

Red 2007 Ford Fiesta (100000km)

Customer ID 1005

Name: Victor Vanvalkenburg
Address: 425 O'Connor St.
Phone Number: (613)432-7622

4 vehicle(s):

Black 2016 GM Escalade (40000km)
Red 2015 GM Malibu (20000km)
Purple 2012 GM Envoy (60000km)
Orange 2012 GM Trailblazer (90000km)

Press enter to continue...

**** Toby's Auto Mechanic Information Management System ****

MAIN MENU

1. Print Customer Database

```
        2. Add Customer

        3. Add Vehicle

        4. Remove Customer

        5. Remove Vehicle

        6. Print Mechanics

        0. Exit

Enter your selection:  4
Customer ID: 1006

Invalid choice.

Press enter to continue...



        **** Toby's Auto Mechanic Information Management System ****

                                MAIN MENU

        1. Print Customer Database

        2. Add Customer

        3. Add Vehicle

        4. Remove Customer

        5. Remove Vehicle

        6. Print Mechanics

        0. Exit

Enter your selection:  4
Customer ID: 1000

Press enter to continue...



        **** Toby's Auto Mechanic Information Management System ****

                                MAIN MENU
```

1. Print Customer Database

2. Add Customer

3. Add Vehicle

4. Remove Customer

5. Remove Vehicle

6. Print Mechanics

0. Exit

Enter your selection: 4
Customer ID: 1001

Press enter to continue...

**** Toby's Auto Mechanic Information Management System ****

MAIN MENU

1. Print Customer Database

2. Add Customer

3. Add Vehicle

4. Remove Customer

5. Remove Vehicle

6. Print Mechanics

0. Exit

Enter your selection: 4
Customer ID: 1002

Press enter to continue...

**** Toby's Auto Mechanic Information Management System ****

MAIN MENU

1. Print Customer Database

2. Add Customer

3. Add Vehicle

4. Remove Customer

5. Remove Vehicle

6. Print Mechanics

0. Exit

Enter your selection: 4
Customer ID: 1003

Press enter to continue...

**** Toby's Auto Mechanic Information Management System ****

MAIN MENU

1. Print Customer Database

2. Add Customer

3. Add Vehicle

4. Remove Customer

5. Remove Vehicle

6. Print Mechanics

0. Exit

Enter your selection: 4
Customer ID: 1004

Press enter to continue...

**** Toby's Auto Mechanic Information Management System ****

MAIN MENU

```
       1. Print Customer Database

       2. Add Customer

       3. Add Vehicle

       4. Remove Customer

       5. Remove Vehicle

       6. Print Mechanics

       0. Exit

Enter your selection:  1

CUSTOMERS:

Customer ID 1005

       Name:                     Victor Vanvalkenburg
       Address:                  425 O'Connor St.
       Phone Number:                    (613)432-7622

       4 vehicle(s):

      Black 2016    GM Escalade (40000km)
       Red 2015     GM Malibu (20000km)
     Purple 2012          GM Envoy (60000km)
     Orange 2012   GM Trailblazer (90000km)



Press enter to continue...



       **** Toby's Auto Mechanic Information Management System ****

                                MAIN MENU

       1. Print Customer Database

       2. Add Customer

       3. Add Vehicle

       4. Remove Customer

       5. Remove Vehicle
```

```
       6. Print Mechanics

       0. Exit

Enter your selection:  5
Customer ID: 1005

Vehicle (0 - 3): 4

Invalid choice.

Press enter to continue...



       **** Toby's Auto Mechanic Information Management System ****

                                MAIN MENU

       1. Print Customer Database

       2. Add Customer

       3. Add Vehicle

       4. Remove Customer

       5. Remove Vehicle

       6. Print Mechanics

       0. Exit

Enter your selection:  5
Customer ID: 1005

Vehicle (0 - 3): 3

Press enter to continue...



       **** Toby's Auto Mechanic Information Management System ****

                                MAIN MENU

       1. Print Customer Database

       2. Add Customer
```

```
       3. Add Vehicle

       4. Remove Customer

       5. Remove Vehicle

       6. Print Mechanics

       0. Exit

Enter your selection:  1

CUSTOMERS:

Customer ID 1005

       Name:                       Victor Vanvalkenburg
       Address:                    425 O'Connor St.
       Phone Number:                      (613)432-7622

       3 vehicle(s):

      Black 2016    GM Escalade (40000km)
       Red 2015     GM Malibu (20000km)
     Purple 2012           GM Envoy (60000km)



Press enter to continue...



       **** Toby's Auto Mechanic Information Management System ****

                                MAIN MENU

       1. Print Customer Database

       2. Add Customer

       3. Add Vehicle

       4. Remove Customer

       5. Remove Vehicle

       6. Print Mechanics

       0. Exit
```

```
Enter your selection:  6

MECHANICS:

Employee ID 5000

       Name:                           Bill Taylor
       Address:                        54 Park Place
       Phone Number:                   (613)826-9847
       Salary:                               75000

Employee ID 5001

       Name:                           Steve Bane
       Address:                        77 Oak St.
       Phone Number:                   (613)223-4653
       Salary:                               60000

Employee ID 5002

       Name:                           Jane Smyth
       Address:                        10 5th Ave.
       Phone Number:                   (613)762-4678
       Salary:                               71000


Press enter to continue...



       **** Toby's Auto Mechanic Information Management System ****

                                 MAIN MENU

       1. Print Customer Database

       2. Add Customer

       3. Add Vehicle

       4. Remove Customer

       5. Remove Vehicle

       6. Print Mechanics

       0. Exit

Enter your selection:  0
```