COMP2710: Homework 2

Due: **Fri, November 14th, 2014 at 11:55 pm on Canvas;** Points Possible: 100

Goals:

- To learn how to define C++ classes and the member functions
- To learn how to define inheritance using C++ derived classes

Description:

Write a program that will register and record the ownership of vehicles. The vehicles can be of type truck, car, or sportscar. The different types of vehicles must be defined as different derived C++ classes as follows:

1. Create a base class called `Vehicle` that has the manufacturer's name (type `string`), number of cylinders in the engine (type `int`), and owner (type pointer to a `Person` object, given below).
2. Then create a class called `Truck` that is derived from `Vehicle` and has additional properties: the load capacity (type `double`, since it may contain a fractional part) and towing capacity in pound (type `int`).
3. Next, define a `Car` class that is derived from `Vehicle` and has additional properties: the number of doors (type `int`) and the size of the engine in liters (type `int`).
4. Finally, define a class called `SportsCar` that is derived from the `Car` class and has additional properties: whether it has sunroof (type `bool`) and whether the top can be popped (type `bool`).

Be sure that your classes have reasonable complement of constructors, accessor, and mutator functions, an overload assignment operator, and a copy constructor. *Write a driver program that tests all member functions for each of the above four classes*: `Vehicle`, `Truck`, `Car`, and `SportsCar`.

The definition of the class `Person` follows.

```
class Person
{
public:
    Person();
    Person (string the_name);
    Person (string the_name, string the_address);
    Person (const Person& the_object);
    string get_name() const;
    int set_name(string the_name);
    string get_address() const;
    int set_address(string the_address);
    Person& operator = (const Person& rt_side);
private:
    string name;
    string address;
};
```

Write all the definition of the member function of the above `Person` *class and write a driver program to test all these functions*.

Write a program that will allow the owner and vehicle information to be entered and stored. The program will then allow the owner and vehicle information to be retrieved later, i.e. it must allow the user to do the following:

1. Print all the owners (and all related information) and their vehicles (and all related information)
2. Print a particular owner (by name), all the owner's related information and their vehicles (and all their related information)
3. Print all the vehicles (and all related information) and their owners
4. Print all the vehicles (and their owners) of a specific type, e.g. sportscar.

There must be only one `Person` object for each owner or person, but a `Person` object may own more than one vehicle of different type.

Your program may consist of two other C++ classes called `System` and `Menu`. Each of these classes must define an Abstract Data Type that maintains a set of data and related functions. You are required to define the set of functions described below for each class and may also add other functions and other member variables for storing data. You must design the C++ classes to include private and public member functions and variables; you must correctly decide which functions and variables should be public and which are private.

The `System` class is responsible for managing the main operations of the system. Objects of the other classes are created in this `System` class. It contains a function that has a loop that will prompt for the user option and perform the appropriate operations. The loop will terminate when the user enters the quit option. The function may use the `Menu` class to display and determine the option selected by the user and perform the related functions.

The `Menu` class is responsible for displaying the menu for all the possible options and accepts the options that are selected by the user.

Use your classes with a test program that creates several owners and all the four different types of vehicles. Your test program must show clearly that your program is capable of performing all the four main functions above, e.g. print all owners, print all vehicles, etc.

Programming Environment:

Your program must compile correctly and run correctly using the g++ compiler on a tux Linux computers (in Davies 123) You may test the program elsewhere, e.g. computer labs in Shelby, your home Linux machine, a Linux box on a virtual machine, or using an emulator like Cygwin, but your final program must compile and execute correctly in the Linux computers (in Davies 123).

Requirements:

1. **Use comments to provide a heading at the top of your code** containing your name, Auburn Userid, filename, and how to compile your code.
2. Your source code file should be named as "<username>_hw2.cpp" (for example, mine would read "lim_hw2.cpp").
3. No compilation error and no warning messages
4. Usability of your program (e.g., input and output)
5. You must define a function.
6. You must use a loop.
7. Quality of your source code.

You will **lose points** if you: do not use the specific program file name, or do not have a comment block on **EVERY** program you hand in.

Deliverables:

- Submit your source code file named as "<username>_hw2.cpp" through the Canvas system.
- Submit the execution script of your program to show that it is capable of performing all the functions as specified, including the following:
  - All the results of the driver tests for all the member functions of the `Vehicle`, `Truck`, `Car`, `SportsCar` and *`Person`*.
  - All the results showing that your program can perform the following functions:
    - Print all the owners (and all related information) and their vehicles (and all related information)
    - Print a particular owner (by name), all the owner's related information and their vehicles (and all their related information)
    - Print all the vehicles (and all related information) and their owners
    - Print all the vehicles (and their owners) of a specific type, e.g. sportscar.