CSC 281 – Intro to Computer Science II
Fall 2013 – Programing Project 8 - Due 11/6/2013 @ 11:59

1. Design and implement an Automobile class.

   This class knows about:
   a) The make of the automobile (for example, Ford, Toyota)
   b) The body style of the automobile (for example, wagon, sedan, truck, SUV, van)
   c) The color of the automobile (for example, silver, blue, metallic green)
   d) The number of cylinders in the engine of the automobile (for example, 4, 6 or 8)
   e) The city-driving gas mileage of the automobile.
   f) The distance-driving gas mileage of the automobile.
   g) The cost of the automobile to the dealer
   h) The manufacturer-recommended selling price of the automobile.

   For each of these items, you should declare a private data member in the class.

   This class should have a single constructor which accepts values for a) through h). Once an object has been created, these values cannot be changed (that is, your class provides no way for these values to be changed)

   It should have accessor functions for each of items a through h. These functions should simply return the corresponding value.

2. Declare 6 objects of class Automobile, with the following characteristics:

   A black Buick sedan with an 8 cylinder engine, 10 mpg city/ 13 mpg distance, $12000 to the dealer, $18500 to the public.

   A blue Volvo wagon with a 4 cylinder engine, 20 mpg city/ 24 mpg distance, $14000 to the dealer, $22000 to the public.

   A tan Toyota truck with a 4 cylinder engine, 22 mpg city/ 27 mpg distance, $7800 to the dealer, $12000 to the public.

   A white Ford truck with a 4 cylinder engine, 21 mpg city/ 25 mpg distance, $6500 to the dealer, $10000 to the public.

   A green Mercedes sedan with a 6 cylinder engine, 15 mpg city/ 19 mpg distance, $35000 to the dealer, $51000 to the public.

   A maroon Ford SUV with a 6 cylinder engine, 16 mpg city/ 20 mpg distance, $21000 to the dealer, $32000 to the public.

   You can hard-code these values.

3. Put each of these 6 objects into an array of Automobile objects. You should now have an array of Automobile objects, with array item [0] through array item [5] having values. If you made your array larger, the remaining cells will have undetermined value. If you declared your array with exactly 6 cells, they should all now have values.

4. Write a `main()` function to test your accessor functions by making sure you can call each of them for one of your objects (don't write the code to call each accessor function for each object; it's enough to call each function for a single object to verify that the function is working). When you're sure they work, remove the code that tests them.

5. Design and implement a Customer class. The customer class knows about:

   a. Last name
   b. First name
   c. Preference for automobile make (including the value "any" which signifies that the customer will accept any make that meets his/her other criteria).
   d. Maximum amount the customer is willing to spend (not considering taxes and licensing) for an automobile.

   This class should have a constructor function that accepts values for items a through d. It should also have accessor functions for each of items a through d.

6. The file "customer.dat" has data representing several customers. For each one of these sets of data, there are four values: customer's last name, customer's first name, customer's automobile make preference and maximum amount customer is willing to spend for an automobile.

   Add to your `main()` function to implement the following pseudocode:

```
open the file "customer.dat"
while (more data on "customer.dat")
    construct a Customer object with the data that has just been read;
    call the free function match(Customer, Automobile[]) to determine which, if
        any, automobile in your inventory (the array of Automobiles) matches the
        customer requirements;
    report the details of the automobile which matches the customer requirements.
end while
```

   For the reporting of the automobile details, either call on the individual accessor functions and print out what they return, or write a class member function to do the work instead, and just call that function for the automobile object returned by the `match(Customer)` function.

7. The only other task is to write the free function `match(Customer, Automobile[])`.

   This function should return an object of class Automobile. Therefore, its declaration will be:

   ```
   Automobile match(Customer, Automobile[]);
   ```

   This function's job is to match the requirements of its Customer parameter with the attributes of the six Automobile objects. As soon as you find a match, return the Automobile object which matched. If you don't find a match, return an Automobile object with these attributes: make: "No match"; body type: "No match", color: "No match"; engine cylinders: 0; mpg city: 0; mpg distance: 0; price for dealer: 0; price for public: 0.

8. Your submission on blackboard must include the following components:

   a. Automobile class declaration
   b. Automobile class implementation
   c. Customer class declaration
   d. Customer class implementation
   e. Program listing which has `main()` and your `match` function.
   f. Execution results

Don't forget to document your classes as well as `main()` and your `match` function.

Don't forget to check that your output is nicely and consistently formatted and your spelling is correct.

Bon travail!