Homework 3 – Classes

Due next week class 4a at beginning of class

Orders for Truck & Car

Objective

This program will give you practice working with:

- Wrapper classes
- Parsing strings
- Array usage
- OOD – Object Oriented Design (UML)
- ArrayList or Vector
- "instanceof"
- toString()
- Java Docs - Documentation
- Other Java programming concepts

Description

In this set of programs you will write classes that work together.

The classes are:

1. Truck
2. Car
3. Orders

To start describing this assignment, there are some things done differently this program than previous homeworks. This homework REQUIRES user input and printing to be done in the classes other than the "test (or main) class". The reason of this is, the (non-main) classes will "take care of themselves". That means the classes have all the knowledge to prompt the user for their information, store and retrieve their information, along with the accessors and mutators to return this information to the main program as requests are made.

The Order class accepts from the user what kind of vehicle they want to purchase, a Car (C/c) or a Truck (T/t). From this simple choice, Order calls the default constructor for the class requested. The default constructor is to call a method(s), name the methods the same in both the Car or Truck class.

From this constructor, the Orders class creates and object of Truck and of Car, per user's request. Each Truck and Car selected will ask the user for some basic information; these are Model, Color, and Cost.

In addition to these three basic pieces of information when a:

- Truck order; you will use a menu system to ask the user if:
  - Is it a “half ton”, or “one ton” truck?
  - Engine size, choices could be. “Really big”, “Not so big” (Please choose other engine sizes than these.)
- Car order; you will use a menu system to ask the user:
  - The type of car: sedan, coupe, or wagon?
  - Does it have a towing package 1=yes, 2=no?

These options are numbered, 1, 2, etc. See the output examples for how the selections of these are to work.

The Orders class uses the Truck and Car classes as needed. Creating a new object of the Truck or Car and adding the appropriate object to a Collection maintained by Orders.

Once the user says there are no more orders, have Orders iterate through Order’s Collection. In this loop determine if the object is a Truck or Car by using the “instanceof” operator. If it is a Car, then cast the object as Car and execute the toString() to have it return the formatted information about the car (see output). Similarly for if the object is of the Truck class.

Requirements:

- Truck and Car classes have a default constructor that calls methods which request input to that class.
- Classes must have Accessors and Mutators for all attributes, along with a toString().
- Must store the orders for Truck and Car in ONE common Collection.
- Javadoc – make sure all the methods have at some javadoc comments to them. Comment any *somewhat* complex code segments in the methods.
- UML Diagram. Do not need to specify accessors and mutators in UML diagram. Make sure connectors are correct.
- Inputs MUST be in order and data type as shown (or 15 points deduction). That is, menu’s must accept numeric entries for the menu choices. For example, you may set the Truck Engine Size to say any two size options you want, but they must be asked at the correct time and the number choices must be 1 and 2.

Strongly suggest:

- In similar methods of Truck and Car, give them the same name. For example, they both will have a setCost, and getCost method.
- Use an accessor or mutators to access instance variables. Even within the class itself.

Submitting your work:

- Submit all of your java and class files to the MyCourses homework dropbox.
- Do not include your JavaDoc files in what you send to the dropbox.

Sample Output: (Bolded text was entered by the user)

```
C:\>java Orders
Your Name Here's Ordering System (218-HW2)

Do you want to order a Truck (T/t) or Car (C/c)? c
Entering Car order:
     Model: Ford
     Color: Green
     Cost: 19875.95
What type of Car is this?
     1. Sedan
     2. Coupe
     3. Wagon
     Choice: 3
Does this car have a towing package?
     1.   Yes
     2.   No
     Choice: 1

Do you want to order another vehicle? y
Do you want to order a Truck (T/t) or Car (C/c)? t
Entering Truck order:
     Model: Dodge RamTruck
     Color: red
     Cost: 25000.95
What size truck is this?
     1. Half-ton
     2. Full ton
     Choice: 1
What is the engine size of the truck?
     1. Really big
     2. Not so big
     Choice: 2

Do you want to order another vehicle? Y
Do you want to order a Truck (T/t) or Car (C/c)? C
Entering Car order:
     Model: Toyota
     Color: white
     Cost: 12543.21
What type of Car is this?
     1. Sedan
     2. Coupe
     3. Wagon
     Choice: 1
Does this car have a towing package?
     1. Yes
```

```
     2. No
     Choice: 2

Do you want to order another vehicle? n

Car:
     Model:  Ford
     Color:  Green
     Cost:   $19875.95
     Type:   Wagon
     Towing: included

Truck:
     Model:  Dodge RamTruck
     Color:  red
     Cost:   $25000.95
     Load:   Half-ton
     Engine: Not so big

Car:
     Model:  Toyota
     Color:  white
     Cost:   $12543.21
     Type:   Sedan
     Towing: not included


Thank you for using Your Name's Ordering System.

C:\>
```

When no more input, print the information entered.

Grading

Your grade will be determined based on these criterions:

- Using a collection: Placing items in, and getting them out.
- Being able to read the data, use mutators to store the values.
- Using accessors to obtain the values and print them out.
- Printing your name and the heading as shown.
- Following coding standards handed out in class. This includes, but not limited to:
    - Define all numbers and strings used in your code as constants.
    - Comment blocks: in the header of each class, prior to each method, and where any code is written that may not be obvious to a Java programmer.

Other Notes for this and all homework:

- Zip then place all work in the homework dropbox. To place your files in the dropbox, attach all *.java files, and any others you need to run the program.

- There are programs out there that will draw the UML from your completed code. These are not acceptable ways of doing the work. Any use of them will be considered a violation of the Academic Dishonesty policy, and will be dealt with accordingly. Writing the UML before writing the code is like looking at a map before you take a long and complex drive in your car. You don't gain any learning by writing the UML after the code is completed, or looking at a map after you arrive at your destination.

- In no case should any code generators be used. This means if your development environment produces ANY Java code for you, that is wrong and also a violation of the Academic Dishonesty policy.

- Your code may be sent to a comparison program to test for similarities between it and other code. Besides the teacher, be assured your code will not be distributed or made available to any other person.

Homework 3 – Classes, methods

Item	Points possible	Points earned
Order.java		
Main driver of code	5	
Accepts choices of truck or car	5	
Adds truck or car to a collection	5	
Prints the truck/car summary	5	
Javadoc comments	5	
Works as expected	5	
Truck.java		
Proper definitions of attributes Define arrays for input	5	
Accept input for this object	5	
Accessors	5	
Mutators validate input, only setting the value if input is valid	5	
Proper toString() method that uses accessors to get the data	5	
Javadoc comments	5	
Works as expected	5	
Car.java		
Proper definitions of attributes, Define arrays for input	5	
Accept input for this object	5	
Accessors	5	
Mutators validate input, only setting the value if input is valid	5	
Proper toString() method that uses accessors to get the data	5	
Javadoc comments	5	
Works as expected	5	
Deductions: Violations of coding standards or other violations		
Total	100	