Lab 2 – Classes

Overview

The purpose of this assignment is to give you some experience writing classes in C++, including utilizing accessors and constructors to organize data properly.

Description

This program will represent a hypothetical car production line, which consists of a chassis (body of a vehicle) and wheels. These two components will be used to make fully functioning vehicles (just use your imagination). To that end, there are three classes you will be writing:

- Vehicle
- Chassis
- Wheel

For this assignment, main.cpp will be provided for you, so you don't have to worry about the structure of the program. Instead, you can focus solely on the structure of the classes and their interactions. I suggest looking at main closely, to help you understand the structure of the project.

Chassis

The Chassis class is the backbone of your vehicle. You will need to store the following data:

- The size of the chassis (small, medium, large)
- The quality of the chassis (poor, fair, good)
- The number of wheels it can support

The number of wheels a given chassis can support is dependent on the size of the vehicle. If the vehicle is small, it functions with 3 wheels (the tri-wheeled wonder). If it is medium, 4 wheels are required. If the vehicle is large, you will need to have 6 wheels.

The default constructor for the chassis will insure it is a medium sized chassis of fair quality. You will also need to implement two more constructors, one that will allow you to specify a size for the chassis, and one that will allow you to specify a size and quality for the chassis.

Finally, you will need a getter method for the number of wheels called getNumWheels().

Wheels

Your Wheel class should contain variables for the following:

- Mileage Left
- Condition the wheel is in (poor, fair, good)

The default condition for Wheels made will be fair. A fair wheel will have 10,000 miles available to it at the start. A good wheel will have 20,000 and a poor wheel will start with 5,000.

You will once again need a default constructor and a second constructor that overrides the condition of the wheel.

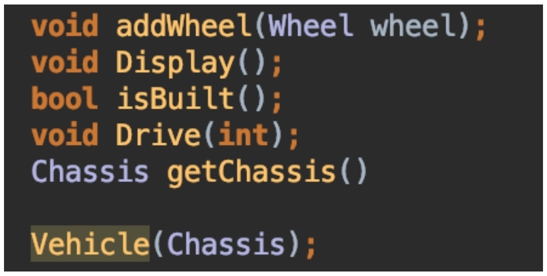
Vehicle

The Vehicle class will be the finished product of your production line, being comprised of the other objects defined below. You will need to store the following data:

- The price
- Wheels
- A Chassis (body of car)
- Is it drivable? (Boolean)

The quality of the chassis and the condition of the wheels will determine the price of the vehicle. The base price of all vehicles is 500. The chassis will apply a multiplier of 5, 8 or 12 to this, depending on its quality. Likewise, the quality of each wheel will apply a multiplier of 1.5, 1.8 or 2.2. (Maybe price should be a float to account for this?). Additionally, a vehicle only becomes drivable once it has the appropriate number of wheels added to it.
An example: A good, small chassis with 2 poor wheels and 1 fair wheel. Bolded is the wheel multipliers (3 of them).
500 * 12 * **(1.5 * 1.5 * 2.2)** = 29,700

In addition to the above, you will need to implement the following methods:

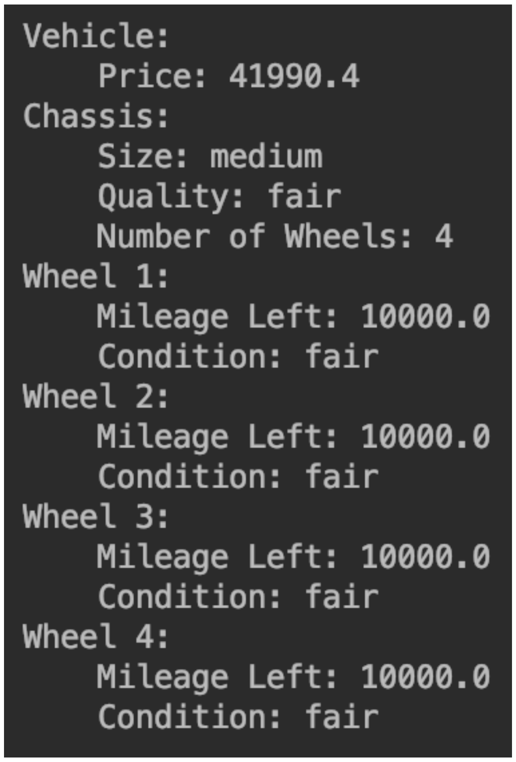


```
void addWheel(Wheel wheel);
void Display();
bool isBuilt();
void Drive(int);
Chassis getChassis()

Vehicle(Chassis);
```

- addWheel() will is fairly self-explanatory, adding a new wheel object to your car (consider using a vector for this). An additional condition is that if you have already added the max number of wheels for the given vehicle (chassis size), the message “You’ve already added all the wheels!” should be displayed
- isBuilt() should return a Boolean as to whether or not the chassis and all wheels have been added to the vehicle.
- Drive() will give your car the ability to go for a test run. The integer it takes in should be the mileage you want the vehicle to travel. If the value entered is greater than the tire with the least mileage left, you should display the output “Broke Down!”. After every drive print “You’ve traveled x miles!” where x is the amount traveled. If you broke down, this value could be different from the value passed into the function. Additionally, make sure to change the condition of the wheels based on their mileage left. If above 10,000 they are good. If above 5,000 they are fair. Below that you have poor wheels. **Keep in mind, a change in condition also means a change in price for the vehicle!** Finally, if you attempt to drive the vehicle before it is built, the message “Vehicle not built. Literally un-drivable” should be displayed.
- getChassis() should just return the current vehicle’s chassis.
- Display() should display all the information of a vehicle and its associated chassis and wheels.

Here is a sample output:



```
Vehicle:
	Price: 41990.4
Chassis:
	Size: medium
	Quality: fair
	Number of Wheels: 4
Wheel 1:
	Mileage Left: 10000.0
	Condition: fair
Wheel 2:
	Mileage Left: 10000.0
	Condition: fair
Wheel 3:
	Mileage Left: 10000.0
	Condition: fair
Wheel 4:
	Mileage Left: 10000.0
	Condition: fair
```

Tips

A few tips about this assignment:

- You can print out a tab character (the escape sequence '\t') to help line up the output.
- Don't try to tackle everything all at once. Work on one class at a time. Can't have a Car without a Chassis.
- You can customize the way numbers are displayed in C++ (particularly floating-point numbers). The header file <iomanip> contains this functionality. Look into std::fixed and std::setprecision()