Part 1: Car’s Constructor

Write the constructor body for the Car class. This will initialize the instance variables. The instance variables have been provided for you.

2: getYear(), getModel(), getMiles(), getFuelTankLevel()

Finish writing the accessors for each of the instance variables. Return the information for each respective instance variable.

Testing getYear(), getModel(), getMiles(), getFuelTankLevel()

When you run the program, it will automatically test the method by calling gettersTests(). You should find that method in CarMain.java. You should add your own tests, as we only have a limited amount of tests.

3: addMiles(int milesToAdd)

The addMiles method is a method that will add to the current amount of miles already driven by a car. The miles instance variable holds the total amount of miles for a given car. For example, with the following code the miles variable for the car should end up being 200,150.

**public** **static** **void** **main(String** args**[]){**

**Car** vw **=** **new** **Car(**1969**,** "Volkswagon Beetle"**,** 200000**,** 100**);**

vw**.**addMiles**(**150**);**

**}**

Testing addMiles(int milesToAdd)

There is a prewritten method called addMilesTests() method found in CarMain.java. You should add your own tests, as we only have a limited amount of tests. Think about what happens if addMiles is called more than once on a car?

4: isFuelTankEmpty()

This method returns true or false based on if a given cars fuel tank level is 0.

* **Hint**: Which instance variable can give information on a fuel tank level?

Testing isFuelTankEmpty()

After writing this method check the method isFuelTankEmptyTests() in CarMain.java. There are prewritten test here. Make sure to add your own tests as well.

5: milesToFuelLevel(int miles)

For this next method, you will write a conversion from miles to fuel level percentage. Since the fuelTankLevel variable represents a percentage, you will need to convert accordingly. For this method, assume that every car fuel tank has a maximum capacity of 400 miles. Return a ratio between miles and the fuel tank capacity as a percentage. (i.e. 62.5)

Testing milesToFuelLevel(int miles)

After writing this method check the method milesToFuelLevelTests() in CarMain.java. There are prewritten tests here. Make sure to check if you are returning a percentage or a decimal representation.

Part 6: modifyFuelTankLevel(int miles)

For this method, you will write a method that changes a car’s fuel tank level depending on the amount of miles inputted. Only change the fuel tank level if the fuel tank is not empty. If the miles inputted causes the fuel tank level to be negative, make the fuel tank level 0.

* **Hint:** What previous methods could help with this?

Testing modifyFuelTankLevel(int miles)

After writing this method check the method modifyFuelTankLevelTests() in CarMain.java. Make sure to add your own tests. If a fuel tank becomes negative is it set appropriately? Do your previous methods reflect the same information?

Part 7: drive(int miles)

For the last part of this assignment, you will write a that method “drives” a car. This method returns a string with a status of either the cars total mileage and fuel tank level, or if the fuel tank is empty. This method will also change a cars miles and fuel tank level. For example, calling drive with an input of 150, with a car that is a 2019 Honda CRV with 11,000 miles and a fuel tank level of 100%, will result in the following output.

The 2019 Honda CRV drove 150 miles, for a total mileage of 11150 and a fuel tank level of 62.5%.

If the Car object’s fuel tank level is empty, then the following would be outputted.

The 2019 Honda CRV's fuel tank is empty!

* **Hint:** Take a look at the previous written methods. Could these help?

Overload drive()

This method will "drive" the Car a random amount of miles from 1-10. It will display the same information as **drive(int miles).**

Testing drive(int miles)

Since this method is returning two different strings depending on a cars instance variables, it would be best to test for each string separately. In the while loop set a cars fule level to reach 0 before the car can drive 100 miles. (For example, first write the code if a cars fuel tank is empty. Is it returning the correct string and format? Next, write the code for a fuel tank that is not empty. Is this string returned correctly?) Then race the cars to 100 miles and see who wins.