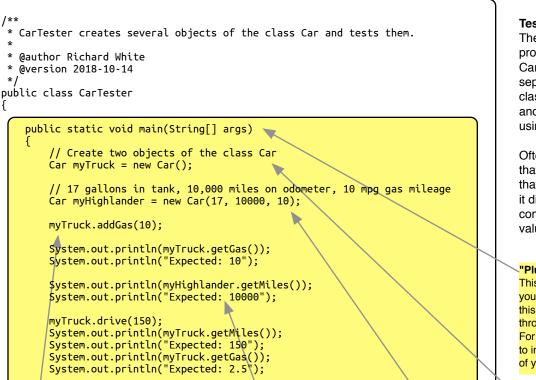
```
44
                                                                                              Class Description
 * The Car class
                                                                                              This entire page is a class
 * @author Richard White
                                                                                              description, which describes to the
 4
   @version 2018-10-14
                                                                                              computer how to construct an
 */
                                        Instance Variables
                                                                                              object of the class Car, what
                                        These variables are always declared as private, meaning the
public class Car
                                                                                              accessor methods can access
                                        user doesn't have direct access to them. Users will only by
                                                                                              information about that object, and
                                        be able to interact with this information via the methods we
    // instance variables 🚄
                                        write for them.
                                                                                              what mutator methods can do to
    private double gas;
    private double odometer;
                                                                                              alter the object.
    private double milesPerGallon;
     * Constructor for objects of class Car
    public Car()
                                                                                                 Constructor Methods
         // initialise instance variables
                                                                                                 One of these two methods will be
        gas = 0;
                                                                                                 used to construct a new Car when
        odometer = 0;
                                                                                                 a program requests it. There are
        milesPerGallon = 20;
                                                                                                 two constructors because there are
                                                                                                 two different ways to make a new
                                                                                                 Car: one default method, and one
       Overloaded Constructor for objects of class Car
                                                                                                 in which the initial characteristics of
       @param initialGas amount of gas in the car
                                                                                                 the car are specified.
       Oparam initialOdometer the initial odometer reading
       @param milesPerGallon the capacity of the car's gas tank
    public Car(double initialGas, double initialOdometer, double milesPerGallon)
        // initialise instance variables
        gas = initialGas;
        odometer = initialOdometer;
        this.milesPerGallon = milesPerGallon;
       getGas method tells how much gas is left in the car
       @return the amount of gas in the car's tank
                                                                                                Accessor Methods
    public double getGas()
                                                                                                These methods are public so that a
                                                                                                program can access them. They
        return gas;
                                                                                                "return" values to a program that
                                                                                                uses them, giving the user access
                                                                                                to current information about the
                                                                                                Car object.
       addGas method adds an amount of gas to the gas tank
       Qparam gasAdded the amount of gas being added to the tank
    public void addGas(double gasAdded)
                                                                                                Mutator Methods
        this.gas = this.gas + gasAdded;
                                               // or just gas = gas + gasAdded
                                                                                                These methods are also public,
                                                                                                and are used to alter the values of
    /**
                                                                                               the private instance variables in
                                                                                                our Car class.
     * drive method drives the car a specified distance
     * @param distance the distance the car is driven
     *,
    public void drive(double miles)
        double gasNeeded = miles / milesPerGallon;
        odometer = odometer + miles;
        gas = gas - gasNeeded;
       getMiles method tells how many miles the car has traveled
       @return the total miles the car has traveled ever (odometer reading)
    public double getMiles()
        return this.odometer;
}
```



### Mutator Method call

}

}

Here we're calling the *addGas()* method for the *myTruck* object. Because it started out with 0 gallons of gas, we'd expect that it has 10 gallons now, but we're going to use the *getGas()* method to confirm that, and output the results along with what we expected to find.

# Accessor Method call

We haven't changed any values in the myHighlander object yet, but let's confirm that the constructor method did what it was supposed to do. We'll call the *getMiles()* method.

### **Tester Class**

The CarTester class is the main program that will be used to run the Car class that we've created. It's a separate program that uses the Car class to establish two Car objects, and then manipulates those objects using their methods.

Often, we'll want to "test" the code that we've written, and demonstrate that it's working correctly by having it display calculated values, and compare them with expected values.

### "Plumbing"

This line is used at the beginning of your main programs. The details of this syntax will be explained throughout the course of the year. For now, just know that you need to include this line at the beginning of your main program or tester.

# Default Constructor

This part creates a new object of the class Car, which we'll refer to as "myTruck."

#### Another Constructor

This one creates a different object, and shows how we can use *parameters* to specify certain initial values.