1. Fill in some code for a `Car` class, following guidelines given through comments.

```java
public class Car {
    // instance variables
    String manufacturer;
    String model;
    int year;
    double price;

    // constructor
    public Car(String x, String y, int z, double w)
    {
        manufacturer = x;
        model = y;
        year = z;
        price = w;
    }

    // constructor: another version of the constructor, without
    // parameters for year, or price; sets these to default
    // values (2015 and 0, respectively).
    public Car(String x, String y))
    {
        manufacturer = x;
        model = y;
        year = 2015;
        price = 0;
    }

    //******** Assume the following methods are also implemented ********
    // public double getPrice() // returns the Car’s price
    // public void void setPrice(double x) // sets price to x
    // public String toString(): returns a String representing this Car
    // ************************************************************************
}
```

2) Write client code that uses the `Car` class:

a) Instantiate a `Car` object using the first constructor (make up appropriate values), assign to `myCar`.

```java
Car myCar = new Car("Specialized", "Dolce", 2013, 890); //actually, a bike
```

b) Instantiate a `Car` object using the second constructor, using manufacturer “Jaguar”, model name “XF Supercharged” and assign this object to a variable named `obscureObjectOfDesire`.

```java
Car obscureObjectOfDesire = new Car("Jaguar", "XF Supercharged");
```

c) Print the info of `myCar` and `obscureObjectOfDesire` (use the `toString()` method).

```java
System.out.println(myCar + obscureObjectOfDesire);
```

d) Print the average price of `myCar` and `obscureObjectOfDesire` (use `getPrice()`).

```java
System.out.println("average = " + (myCar.getPrice() + obscureObjectOfDesire.getPrice())/2);
```
1. Fill in some code for a `Car` class, following guidelines given through comments.

```java
public class Car {
    // instance variables
    String maker;
    int year;
    double price;

    // constructor
    public Car(String x, int y, double z) {
        maker = x;
        year = y;
        price = z;
    }

    //constructor: another version of the constructor, without
    // values for year, or price; sets these to default
    // values (2014 and 20000, respectively).
    public Car(String x) {
        maker = x;
        year = 2014;
        price = 20000;
    }

    //******** Assume the following methods are also implemented ********
    // public double getPrice()   // returns the Car’s price
    // public void setPrice(double x)  // sets price to x
    // public String toString(): returns a String representing this Car
    // *************************************************
}
```

2) Write client code that uses the `Car` class:

a) Instantiate a `Car` object using the first constructor (make up appropriate values), assign to `myCar`.
   ```java
   Car myCar = new Car("Specialized", 2013, 890); //actually, a bike
   ```

b) Instantiate a `Car` object using the second constructor, using manufacturer “Nissan” and assign this object to a variable named `otherCar`.
   ```java
   Car otherCar = new Car("Nissan");
   ```

c) Print the average price of `myCar` and `otherCar` (use the `getPrice()` method).
   ```java
   System.out.println("average = " + (myCar.getPrice() + otherCar.getPrice())/2);
   ```

d) Set the price of `myCar` and to 5000 (use the `setPrice()` method).
   ```java
   myCar.setPrice(5000);
   ```