1. Suppose you have a class `Cat` defined as shown below. Fill in the code for the indicated methods, following guidelines given through comments.

```java
public class Cat {
    // instance variables
    String name;
    int age;
    int lives; // number of lives remaining

    // constructor: takes a String as parameter and creates a
    public Cat (String x) {
        name = x;
        age = 0;
        lives = 9; // cats start out with 9 lives
    }

    // birthday(): if the cat is still alive (has least one life left)
    // increases age by 1; otherwise does nothing
    // WRITE METHOD DEFINITION HERE: *****************************************

    // death(): decreases the number of lives by 1; no effect
    // on dead cats (i.e., cats with 0 lives).
    // ********** assume this is already implemented **********

    // getAge(): returns the cat’s age
    // ********** assume this is already implemented **********

    // getLives(): returns the cat’s remaining lives
    // ********** assume this is already implemented **********

    // toString()
    // WRITE METHOD DEFINITION HERE: *****************************************

    }
}
```
2) Suppose you are writing a **driver class** that uses the `Cat` class. Answer the questions below by writing code fragments for this driver class.

a) Instantiate an object of the `Cat` class with name “Luca” and assign it to a variable named `profsCat`.

b) Suppose you have already declared and initialized three variables `cat1`, `cat2`, and `cat3` that refer to `Cat` objects (i.e., like `profsCat`). Let’s pretend that two years have gone by… So their ages need to increase. Meanwhile, `cat3` has done something stupid and loses a life. Write some code that uses the `birthday()` and `death()` methods to model this situation.

c) Print the sum total of the lives of `cat1`, `cat2`, and `cat3`. 
1. Suppose you have a class `Car` defined as shown below. Fill in the code for the missing methods, following guidelines given through comments.

```java
class Car {
    String manufacturer;
    String model;
    int year;
    double price;

    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 (2013 and 0, respectively).
    public Car(String x, String y) {
        manufacturer = x;
        model = y;
        year = 2013;
        price = 0;
    }

    // toString()
    // assume this is already implemented (code omitted)

    // getPrice(): returns the current price of the car
}
```
2) Suppose you are writing a **driver class** that uses the `Car` class. Answer the questions below by writing code fragments for this driver class.

a) Write some code to instantiate an object of the `Car` class (using the second constructor) and to assign this object to a variable named `obscureObjectOfDesire`. Use this information for the car: manufactured by Jaguar, model name “XF Supercharged.”

b) Suppose you have already declared and initialized three variables `car1`, `car2`, and `car3` that refer to `Car` objects. Write some code to find the average of the car prices and to assign it into a variable named `priceAvg`.

c) Write some code to print the results – namely information about `car1`, `car2`, and `car3` and the price average.