Unit 5 Chapter 14 Assignment

Grading Information: This Program is **due** on **Date Specified**.

Comments are **REQUIRED**; flow charts and pseudocode are **NOT REQUIRED**.

Directions	**Points**
The files must be called <**LiFiUnit5Ch14.java**> (driver program) LiFiAnimal.java LiFiFox.java (which extends LiFiAnimal) LiFiChicken.java (which extends LiFiAnimal) The files must be called as specified above, (LiFi = Your Last Initial Your First Initial) *Proper coding conventions required the first letter of the class start with a capital letter and the first letter of each additional word start with a capital letter*. Only submit the **.java** files needed to make the program run. Do not submit the **.class** files or any other files.	5%
Style Components Include properly formatted prologue, comments, indenting, and other style elements as shown in Chapter 2 starting page 64 and Appendix 5 page 881-892.	5%
Topics covered in chapter Topics with * are covered in this assignment. *Object class *equals method toString method *Polymorphism Abstract *Interfaces Protected access modifier	
Basic Requirements Write a program that simulates the battle between a fox and chickens. Use this class hierarchy: Fox • Kills 1 chicken a day	20%

- Does not reproduce

Chicken
- Have a chance to reproduce as long as conditions are met
- Reproduction only happens when chickens are over 1 and 1 of each sex is present

Simulation Control
- Simulation continues as long as chicken population is greater than 1 and less than or equal 10
  - (if 1 or less, mating can't happen. If > 10, chickens will overrun the fox)

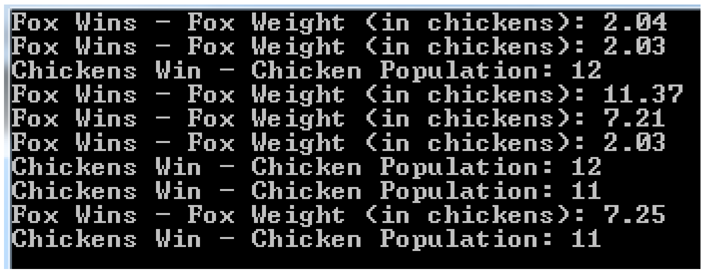
Driver main method should be as shown below: (replacing comment with missing output piece and replacing LiFi with your initials. Add prologue and additional comments to explain functionality.)

```
import java.util.ArrayList;

public class LiFiUnit5Ch14
{
  public static void main(String [] args)
  {
     for(int count=0; count<10; count++)
     {
       LiFiFox foxy = new LiFiFox();
       ArrayList< LiFiChicken > chickens = new
            ArrayList<LiFiChicken>();
       chickens.clearn();
       chickens.add(new LiFiChicken());
       chickens.add(new LiFiChicken ());
       chickens.add(new LiFiChicken ());
       chickens.get(0).setSex(true);
       chickens.get(1).setSex(false);
       chickens.get(2).setSex(false);

       while (chickens.size() >1 && chickens.size() < 10)
       {
          for (LiFiChicken c:chickens)
              c.grow();
```

`foxy.grow();`
`LiFiChicken.mate(chickens);`
`foxy.eat(chickens);`
`}`
`//INCLUDE CODE FOR OUTPUT HERE.`
`}`
`}`
 	
Output code should output:
 	
Depending on if the population of chickens is less than 1 and greater than or equal 10:
 	
Chickens win - Chicken Population: ## (integer value)
or
Fox wins - Fox Weight (in chickens): ##.## (double value, 2 decimal places)
 	
Output should repeat **10 times**.
See sample output below.	

LiFiAnimal.java
 	
Instance variables:
`name (string)`
`age (integer)`
`weight (double)`
`isMale (Boolean)`
 	
LiFiAnimal constructor : (default constructor)
Set age to 1.
 	
grow method :
Increases age of LiFiAnimal by 1.
 	
Accessor / mutator methods for each instance variable above:
Set or returns values as appropriate for data type specified.
	10%

LiFiFox.java class **eat** method: (receive chickens arraylist as argument) Randomly removes a chicken from the population 70% of the time and increases fox weight by the chosen chicken weight. Only increase weight if chicken is	30%

removed/eaten. **grow** method: Set the fox age to the current age plus 1. (use accessor/mutator methods)	
LiFiChicken.java class **LiFiChicken** constructor: (default constructor) Randomly choose sex and assign to isMale as appropriate. Set age to 1. Set weight to 1. **grow** method: Increase age of chicken by 1 and weight of chicken by 1% of current weight. **mate** method: (static method, receive chicken arraylist as argument) Randomly choose 2 chicken objects from arraylist and if conditions are correct, proceed with mating. Successful mating conditions are: • 1 male and 1 female chicken • Both chickens older than 1 day ▪ If successful mating, randomly create between 0-4 chickens and append to arraylist received as argument	30%
NOTE: Complete your activity and submit it by clicking "Submit Assignment"	
Total Percentage	100%
Sample `Your output will vary based on the random numbers generated.` <u>Sample session (requires no user input)</u>: 	

```
Fox Wins - Fox Weight (in chickens): 2.04
Fox Wins - Fox Weight (in chickens): 2.03
Chickens Win - Chicken Population: 12
Fox Wins - Fox Weight (in chickens): 11.37
Fox Wins - Fox Weight (in chickens): 7.21
Fox Wins - Fox Weight (in chickens): 2.03
Chickens Win - Chicken Population: 12
Chickens Win - Chicken Population: 11
Fox Wins - Fox Weight (in chickens): 7.25
Chickens Win - Chicken Population: 11
```