CS151 Lab#2: Inheritance, ArrayList

Reminder to have a TA check off on your lab exercises by the assignment 2 due date.

In this lab, we will practice designing inheritance classes and `ArrayList`. Look in `~dxu/handouts/cs151/labs/02` for code and data files related to this lab.

Exercise 1: Design all the necessary classes in order to make the following driver program work properly (steps have been broken down for you in the sub-parts). You should not change the driver code.

```
public static void main(String[] args){
  Mammal[] mammals = new Mammal[4];
  mammals[0] = new Dolphin();
  mammals[1] = new Platypus();
  mammals[2] = new Human();
  mammals[3] = new CSStudent();

  for (int i=0; i< mammals.length; i++){
    System.out.print("Generally, a " + mammals[i].getName());
    System.out.print(" can be found ");
    if(mammals[i].livesInWater() == false){
      System.out.print("on land, ");
    }
    else {
      System.out.print("in water, ");
    }

    System.out.print("it can ");
    if(mammals[i].laysEggs() == false) {
       System.out.print("not ");
    }
    System.out.print("lay eggs, and is often overheard saying
'");
    mammals[i].speak();
    System.out.println("'");
  }
}
```

For example, this is a sample output that is acceptable:

```
Generally, a Dolphin can be found in water, it can not lay eggs, and is often
overheard saying 'ak, ak, ak, ak'
Generally, a Platypus can be found on land, it can lay eggs, and is often
overheard saying 'errrr'
Generally, a Human can be found on land, it can not lay eggs, and is often
overheard saying 'I'll take a grande latte with a double-shot of espresso'
Generally, a CSStudent can be found on land, it can not lay eggs, and is
often overheard saying 'I love programming!'
```

Specifically, perform the following tasks. In a new directory (say `lab02/ex1`):

1. Design a class `Mammal` with:
    a. two `private String` variables called `name` and `sound`
    b. a constructor that initializes the two variables
    c. getters for the two instance variables
    d. a `void` method `speak()` that prints the object's `sound`
    e. a `boolean` method `laysEggs()`
    f. a `boolean` method `livesInWater()`

2. Design a class called `Dolphin` that extends `Mammal`. Override methods as appropriate.
3. Design a class called `Platypus` that extends `Mammal`. Override methods as appropriate.
4. Design a class called `Human` that extends `Mammal`. Override methods as appropriate.
5. Design a class called `CSStudent` that extends `Human`. Override methods as appropriate.
6. Each class should now be declared `public`, and thus be stored in a separate file that matches the class name, i.e. `Mammal.java`, `Dolphin.java`, etc. The given `main` method should be in a class called `Main` and a file called `Main.java` (or `Lab2` and `Lab2.java`)
7. Compile and check correctness of ouput

Exercise 2: Write a program that reads the text file `nums.txt` into an `ArrayList` called `lst`. Then perform the following operations on `lst`:

1. Print all the numbers out in the following format: `(1, 2, 3, …, 100)`
    a. Do not use `toString` or some other `String` methods to format a single string to print out. Access each element of `lst` individually in a loop and print
2. Print all the numbers out in reverse order
    a. Again via an explicit reverse traversal of `lst`.
3. Compute the average of all numbers in `lst` and print it out
4. Remove all even numbers and print `lst`.
5. Insert 200 to the beginning of `lst`, 300 to the end of `lst` then 400 to the middle of `lst` and print `lst`
    a. Output should look like this : `(200, 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 400, 51, 53, 55, 57, 59, 61, 63, 65, 67, 69, 71, 73, 75, 77, 79, 81, 83, 85, 87, 89, 91, 93, 95, 97, 99, 300)`

Exercise 3: Change the main data structure in **Exercise 1** from array to `ArrayList` and make sure the program works the same.