CS3330: Lab 7

Due 48 hours after your lab ends

Objective:

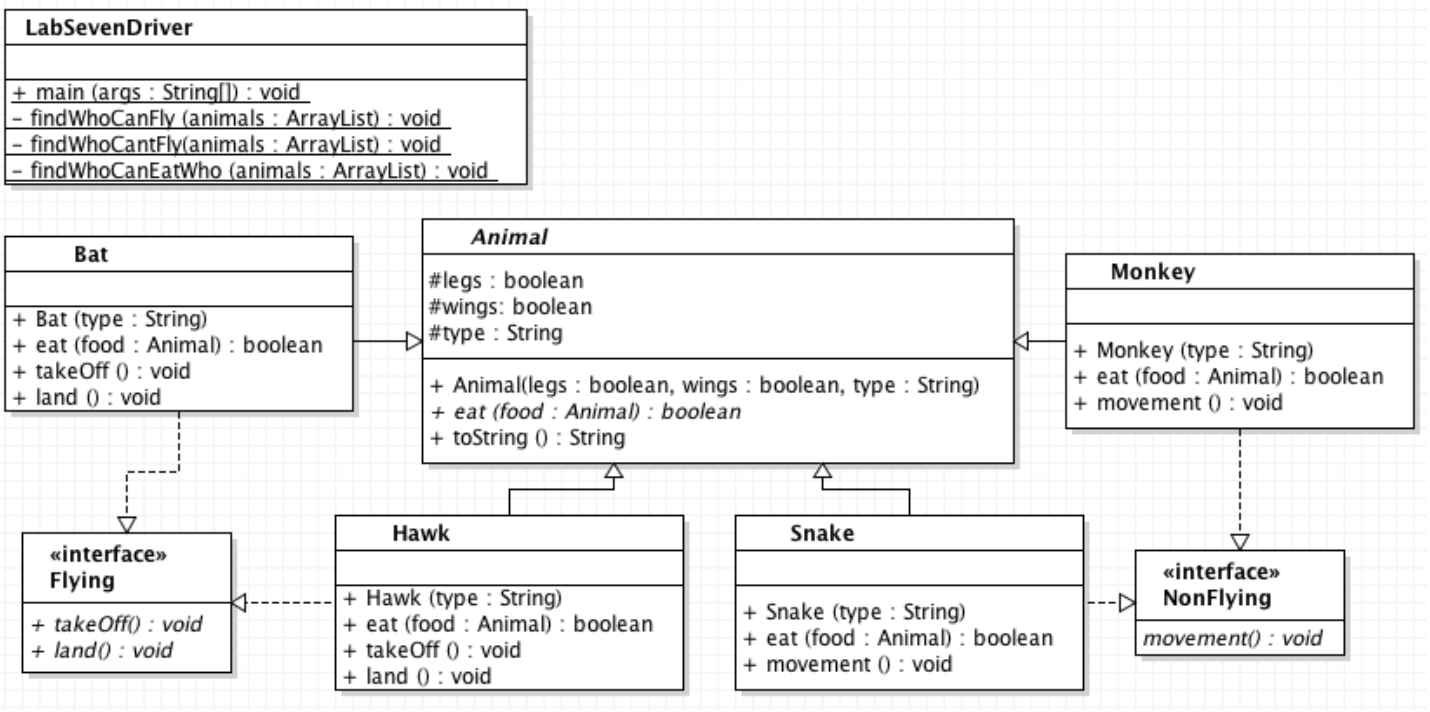
- More practice with **instanceof**
- Implement a project with Polymorphism and Interfaces

Submission Information:

cs_submit CS3330_LAB-<section_letter> LAB7 <yourpawprint>.cs3330.lab7.zip

This lab consists of 6 classes and 2 interfaces: LabSevenDriver.java, Animal.java, NonFlying.java, Flying.java, Bat.java, Monkey.java, Hawk.java, and Snake,java. The main method in LabSevenDriver.java is given!

UML:



*All getters are public and all setters are protected.

****The getters and setters are NOT supplied in the diagram above.**

LabSevenDriver.java

findWhoCanFly(animals : ArrayList) - finds all the animals that can fly; then for each animal, displays the type, and the animal taking off and landing. Check output for help.
findWhoCantFly(animals : ArrayList) - finds all the animals that can't fly; then for each animal, displays the type, and what happens when that animals moves. Check output for help.
findWhoCanEatWho(animals : ArrayList) - Figures out and displays which animal can eat another animal. Check output for help.

Animal.java (Abstract Class)

Animal(wings : boolean, legs : boolean, type : String) - Sets all the attributes
toString() - Returns a string unique string based off the boolean values of legs and wings. The toString will return the type, if it has legs, and if it has wings. (Check output for formatting)

Monkey.java

Monkey(type : String) - calls the constructor of it's superclass with the necessary values.
eat (food : Animal) - Only allows a Monkey to eat an Animal of type Snake.
movement() - allows the Monkey to run

Bat.java

Bat(type : String) - calls the constructor of it's superclass with the necessary values.
eat(food : Animal) - Only allows a Bat to eat an Animal of type Snake or Bat.
takeOff() - void allows the Bat to take off the ground
land() - allows the Bat to land on the ground

Hawk.java

Hawk(type : String) - calls the constructor of it's superclass with the necessary values.
eat(food : Animal) - Only allows a Hawk to eat an Animal of type Snake or Bat.
takeOff() - allows the Hawk to take off the ground
land() - allows the Hawk to land on the ground

Snake.java

Snake(type : String) - calls the constructor of it's superclass with the necessary values.
eat(food : Animal) - A Snake can't eat any Animal.
movement() - allows the Snake to slither

The takeOff, land, and movement methods only display to the user what happened - nothing else. Check output for guidance.

OUTPUT

```
Editable Animals
Hawk has wings and has legs and ate a Bat
Hawk has wings and has legs and ate a Snake
Bat has wings and has legs and ate a Bat
Bat has wings and has legs and ate a Snake
Monkey has no wings, but has legs and ate a Snake

Animals that can fly
Hawk has wings and has legs
The hawk has taken off!
The hawk has landed on the ground
Bat has wings and has legs
The bat has taken off!
The bat has landed on the ground

Animals that can't fly
Snake has no wings and no legs
The snake slithers on the ground
Monkey has no wings, but has legs
The monkey is running on the ground
```

Grading:

General:

If your program does not compile, has runtime errors, or doesn't produce any I/O; your lab will have a grade of zero.

If you do not have Header Comments (including lab submission code) in EVERY file you create, this will also result in a grade of ZERO.

You also may not create ANY other instance variables or methods. Doing this will take 10 points off your grade.

Rubric:

- **5 points -** Comments (ie. Javadoc comments and inline comments)
- **5 points -** For following the UML exactly and having the same output as the given output sample
- **5 points -** findWhoCanEatWho
- **5 points -** findWhoCanFly and findWhoCantFly
- **5 points -** NonFlying.java, Flying.java, and Animal.java
- **5 points -** Monkey.java, Bat.java, Snake.java, Hawk.java

LAB BONUS QUIZ