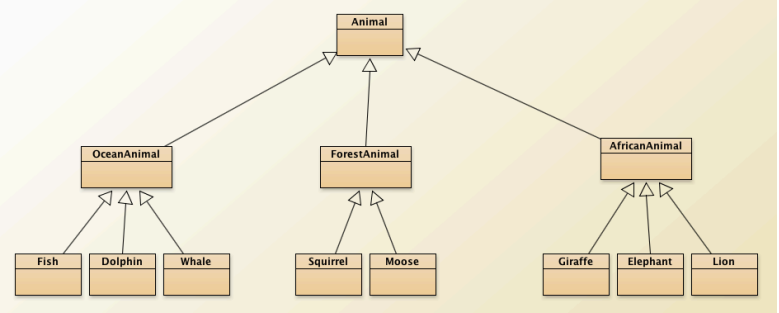
*(30 pts) Approx. 3 days*




This first part of the unit looks at a cool tool for making classes talk and interact with each other: Inheritance. In the same way that you inherit traits from your parents, 'sub' classes can inherit methods, variables, and instructions from other 'parent' classes. In the example on the right, the solid-line arrows suggest inheritance relationships. Logically, Fish (in the bottom row) are a type of "OceanAnimal", and the inheritance relationship shows that. Similarly, "OceanAnimal" is a more specific kind of "Animal". The great thing about this system, is that any method made in the "Animal" class at the top, automatically exists in every class below it… This lets us be very strategic about where methods and variables are made!

1. Take a full page of good notes on the presentations *Interface 1, Interface 2, and Inheritance Tutorial*. Take a good page of notes on these ideas and our class discussions. In particular, look for details on "extends" and "super"!
2. Now, complete the next 6 Java Tasks ***in a new project so you have a clean slate to build on!***
   a. JAVA TASK 59: Make a program that has a class called "Vehicle" that is relatively generic but includes 4 variables of different types. Then, add at least 5 subclasses and properly create their constructors to set the 4 variables in the correct way for that type of vehicle.
   b. JAVA TASK 60: Make a program that starts with a class called "2DShape" (as in geometry) with 3 variables and a constructor. Then, create at least 6 more subclasses with at least 2 levels. Each of these subclasses should have their own constructor that uses the super() constructor to set the included variables appropriately.
   c. JAVA TASK 61: Make a program that creates a parent class called "SchoolPerson" that has 3 variables. Then, create the "Student", "Teacher", "Counselor", and "Principal" classes as subclasses of SchoolPerson. Each of these 4 subclasses should have at least 1 unique variable specific to that class only. Create the proper constructors.
   d. JAVA TASK 62: Make a program that has a main parent class called "Animal" that contains 3 variables and 2 methods. One of the methods should be the toString() method. Then create 6 more classes that fit into 2 more levels of a class hierarchy, and override the toString() method in each. Also create all appropriate constructors.
   e. JAVA TASK 63: Make a program that starts with a parent class called "VideoGameCharacter" and has 4 generic variables as well as 3 generic methods – one of which should be the toString() method. Then, add 12 new classes that build a class hierarchy. These new classes should each have their own unique variable as well as an overridden toString() method.
   f. JAVA TASK 64: Make a program that demonstrates class hierarchy and inheritance with at least 8 different classes organized into at least 3 levels.

| *Part 1: Tasks* | *10-8 points* | *7-4 points* | *3-0 points* |
|---|---|---|---|
| Inheritance Notes | + Watch all presentations<br>+ You took a full page of notes on Inheritance topics<br>+ Your notes include details about the keywords 'extends', 'super', 'abstract'. | - Less than a full page of notes on Inheritance<br>- Your notes are missing important parts | - Very brief or no notes in your notebook |
| | *20-15 points* | *14-8 points* | *7-0 points* |
| Java Tasks 59-64 | + You completed all 6 Java Tasks from this section | - You did not complete all 6 tasks | - You did not complete any tasks |
| Checkoff from Benshoof | + Mr. Benshoof got to see your programs run successfully | N/A | N/A |