**Lab 2 Exercises**

* Create a subfolder of LABS called Lab2 in your git repository.
* Put all your code and answers to the exercises into this subfolder:

**EXERCISES (33 Points each)**

**Exercise 1:** Dogs, just like Birds, Snakes, and Fish, can move() – Add Dog (and, by extension, all of it subtypes, such as Terrier to the inheritance hierarchy you set up in Example Scenario 2 in the lab. Show that you can now add Dog objects and Terrier objects in the array you created in lab…



**Exercise 2:** We have set up a type hierarchy such that a Student is a subtype of Human Being. The class Student defines a new method called getGPA() that is specific to student. We then do the following:

HumanBeing louis = new Student();

Question 1: Can the reference variable louis be used to invoke getGPA() ? Why or Why not?

The question in general, is yes but not directly. If we were to try invoke getGPA through Louis directly, it wouldn't work because it is of type HumanBeing, not Student in which the getGPA method is. However, because Louis is an instance of Student, the typecast us safe and can be done.

Question 2: How can we temporarily change the types of reference variables? Describe the difference between upcasting and downcasting. Which is always safe? How can we ensure that the other was is also safe on a reference-by-reference basis?

We can temporarily change the types of reference variables by typecasting, which can be done either through upcasting, if you are casting a subclass to a superclass, or downcasting, when you are casting a superclass to a subclass. Upcasting is always safe because superclasses are more specific and therefore have all the required information to statisfy the superclass. Downcasting can be "dangerous" as it may not have all the infomration to satisfy the more specific subclass.

Place your answers in a Word document called Lab2Results.docx.

**Exercise 3:** Use your creativity – come up with a simple scenario where inheritance might be useful, and implement it, showing how it works, and, in your comments, explain clearly why you did it this way.

I created a code heirachry that can be used to represent and build character for playing Dungeons and Dragons with your friends. Inheritance is useful as I built a general character creator and then was able to extend that to speific class types for Bards, Clerics, and Barbarians. The class type determines the type of die you use when rolling for your attack damage.

You can find this in the subfolder labeled "Exercise 3". Use \*.javac to compile all of the files and run the "DandDCharacterTester" to test the names of each class types.

**Submission:** Place the Word document (Lab2Results.docx) into your Lab2 subdirectory, alongside Lab2\_starter\_code. Add everything, commit, and push your results.