# Inheritance Practice
# ICS4U

*The purpose of this activity is to practice created classes which inherit attributes and methods and to practice accessing those methods and attributes.*

**Part 1: The Person Class**

1. Create a class named `Person` (which extends `Object`). Identify the subclass and superclass in this situation.

2. Name 2 methods that `Person` will inherit from the `Object` class.

3. Add the following attributes to this class: *name, age, height, date of birth*.

4. Create a constructor which takes 4 parameters to initialize the attributes.

5. Create a second constructor which takes no parameters and initialize the attributes to default values of your choosing.

6. Add the following modifiers to your attributes:

   Name should be `public`

   Age and height should be `private`

   Date of birth should be `protected`

7. Create the appropriate methods to access the private attributes.

8. Create a method called `toString(Person p)` which returns a `String` representation of the object.

9. Create a method called `equals(Person p)` which returns `true` if two `Person` objects are exactly the same and `false` otherwise.

10. Create a class with a `main` method to test the `Person` class. You should create several objects and test each attribute and method.

**Part 2: The Student Class**

11. Create a `Student` class which will inherit from `Person`.

12. Add the following private attributes to this class: *ID number, years at school, grade, advisor.*

13. Create appropriate methods to access the private attributes.

14. Create two constructors as you did for the `Student` class.

15. Which attributes from the `Person` class can be directly accessed from `Student`?

16. Add the following methods to the `Student` class: `toString, equals,`

17. Add the following statements to your main method. Which are valid?

```
Student s1 = new Student();

Student s2 = new Person();

Person p1 = new Student();

Person p2 = new Object();

Object o1 = new Student();

Object o2 = new Person();
```

18. Test the valid statements by accessing the methods you created.

19. For the statements that are not valid, correct them by casting, if possible.

20. Last class, we saw a diagram like the one shown below. Create your own version of this diagram and add the methods from each class into the appropriate location.

A Student