Implementing a Simple Class

Syntax of a class with an instance variable:

```
public class ClassName
{
      private typeName variableName;
      …
}
```

Yesterday, we looked at the difference of classes and objects. You took the time to create 5 objects, defined their variables which described the state of the object, and defined methods to describe their behavior.

Today's activity will look at an example of implementing a Dog class.

When an object of a class is created, the class is said to be instantiated. All the instances share the attributes and the behavior of the class. But the values of those attributes are unique for each object. A single class may have any number of instances, and in this case we can have a number of different dogs.

The following is an example of a class for Dogs that requires the name of the dog, the breed, and the age. There are two files (1) Dog.java that contains the Dog class, and (2) TestDog.java that allows us to create a instance for dog1 and print the results. These two files are both located in the src folder of my project.

File name: Dog.java

```
//Class Declaration
public class Dog {

        /* Instance Variables – the variables which hold data that describes the attributes
        * of any object constructed from it.*/

        private String name;
        private String breed;
        private int age;

        /*Class constructor – used to create a new object from the class each time it is
         * called. The constructor of a class must have the same name as the class and is case
         * sensitive. The constructor can accept information from the caller as parameters
         * that will be assigned to the instance fields of the instance being created.*/

        public Dog(String name, String breed, int age) {
               this.name = name;   //sets this object's name with the provided name
               this.breed = breed; //sets this object's breed with the provided breed
               this.age = age;     //sets this object's height with the provided age

               /*Note that the keyword 'this' tells the compiler that we are referring to the
                * current instance of the class.*/
        }

        /* The following methods are Instance Methods -- objects in Java programs perform
         * actions by use of their methods*/

        public String getName() {
               return name;
```

```
        }

        public String getBreed() {
               return breed;
        }

        public int getAge() {
               return age;
        }

        public int getPersonYears() {
               return getAge() * 7;       //calculates the age of the dog in human years
        }

        public String toString() {
               return getName() + " is a " + getBreed() +
               " and is " + getPersonYears() + " in human years.";
        }
}
```

File name: TestDog.java

```
public class TestDog {

      public static void main(String[] args) {

            //Call on the constructor to create a new instance of Dog.
            //name the instance dog1

            Dog dog1 = new Dog("Gina", "German Sheppard", 4);

            //Gina is the name of the dog
            //German Sheppard is the breed of the dog
            //4 is the age of the dog

            System.out.print(dog1.toString());

      }
}
```

Assignment: (Upload your two Java files to GitHub).

Practice coding your own class and test file. Choose one of the topics you wrote about yesterday.

Class requirements: Your class should follow the format above – instance variable, class constructor, instance methods, toString method.

Include at least 3 variables, and create a method that can manipulate one of your variables like I have above with getPersonYears().

Test file requirements: create 3 instances of your object and print out the results.