CIS 1068: Practice Problems 18

Practice designing classes with constructors, instance fields, and instance methods.

1. Tracing programs

   a. What gets printed to the screen when we execute the MysteryClient class on the left?

```java
public class MysteryClient {
    public static void main(String[] args) {
        Mystery m = new Mystery("hello");
        m.display(" again");
    }
}

public class Mystery {
    public String str = null;
    public Mystery(String s) {
        str = s;
    }
    public void display(String s) {
        System.out.println(str + s);
    }
}
```

   b. What gets printed to the screen when we execute the MysteryClient class on the left?

```java
public class MysteryClient {
    public static void main(String[] args) {
        Mystery m = new Mystery(15, 27);
        m.display(3);
        Mystery m2 = new Mystery(16, 28);
        m2.reduce(4);
        m2.display(4);
    }
}

public class Mystery {
    public int x = 0;
    public int y = 0;
    public Mystery(int num1, int num2) {
        x = num1;
        y = num2;
    }
    public void display(int z) {
        if(x%z == 0 && y%z == 0) {
            System.out.println("divides");
        } else {
            System.out.println("too bad");
        }
    }
    public void reduce(int z) {
        x = x / z;
        y = y / z;
    }
}
```
2. Creating your own classes
   a. Create a Point3D class. This class will represent points in three-dimensional space (as opposed to the Point class that is built in to Java, which represents 2D points).

      What attributes do 3D points have? For each one, create an appropriate instance field in the Point3D class.

      Create a constructor to initialize a Point3D object.

      Create a "translate" method that moves a Point3D object by a specified amount.

      Create a "toString" method that returns a String containing the values of each attribute of a Point3D object.

      Create a main method that constructs some Point3D objects, and tests out the translate and toString methods.

   b. Create a Fraction class. Each Fraction object should have two attributes: a numerator and denominator.

      Create a constructor to initialize these attributes. If the parameter for the denominator is less than 1, the constructor should create an IllegalArgumentException object and throw it. (We'll allow the numerator to be zero or negative, but not the denominator.)

      Write a method called "toString" that returns a String containing the numerator and denominator separated by "/".

      Write a method called "greatestCommonDivisor" that returns the largest integer that divides both the numerator and the denominator.

      Write a method called "reduce" that resets the numerator and denominator by dividing them by their greatest common divisor.

      Finally, write a main() method that constructs some Fraction objects and tests out the methods you've written.