Homework Assignment #5

CS5004 – Object-Oriented Design
Northeastern University – Silicon Valley
Summer 2020

Due Sunday 06/14 at 11:00pm PT

Grading: Each programming problem is graded as follows

- A submission which does not compile gets 0.
- A submission which compiles but does something completely irrelevant gets 0.
- A submission which works (partially) correctly, gets (up to) %80 of the total credit.
- %20 is reserved for the coding style. Follow the coding style described in the book.

Problem 1 [30pts]. Create a class to represent a container. The class Container should have the following properties

1. Maximum capacity of the container in liters.
2. Quantity of liquid at any given time in liters.

The following operations can be performed on the containers:

1. Completely fill a container.
2. Completely empty a container.
3. Transfer liquid from one container to another.

Define the class named `Container` that implements the properties and operations defined below. Create a constructor of the `Container` class that allows the user to specify the maximum capacity of the container in liters. Initially, assume that all the containers are empty.

Implement the following methods in the `Container` class.

- `quantity()` to return the current quantity of liquid at any given time in liters.

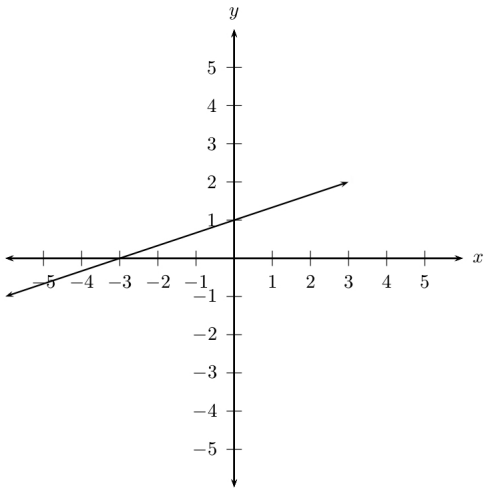
- `leftOver()` to return the quantity of liquid that can be filled in the current container before it is full.
- `full()` to fill the current container fully.
- `empty()` to make the container completely empty.
- `transfer()` to transfer a certain amount of liquid from one container to another.
- `displayQuantity()` to display the current quantity in liters.

Note: While transferring liquid from one container to another, check the maximum capacity of the container.
Submission format: A file `ContainerTest.java` which contains the non-public `Container` class and a public test class `ContainerTest` containing a `main()` method which instantiates a few `Containers` and performs test for each required functionality. Your tests *must* include corner cases (e.g. transferring more than capacity) as well.

Problem 2 [40pts]. In this problem, you will be using the `Rational` class from PS4. The goal is to design a class called `Line` which represents a line on a 2-dimensional plane.* Recall that the equation for a line can be written as

$$y = rx + q, \tag{1}$$

where r and q, both rational numbers, are the slope and intercept respectively. For example, the figure below shows a line whose equation is $y = \frac{1}{3}x + 1$.



Given a point $p = (a, b)$, where a and b are rational numbers, we would like to know if p is on the line. Fortunately, the method for doing so is easy: just plug in a for x and see if $y = b$ or not. For example, consider the line above. The point $p = (1, 2)$ is not on the line (the line does not pass through it) since setting $x = 1$ and computing $y = \frac{1}{3} * 1 + 1$ we get $\frac{4}{3} \neq 2$. On the other hand, the point $p = (3, 1)$ is on the line since $y = \frac{1}{3} * 3 + 1 = 2$.

*No special math background is required for this problem. Read on.

Your code will be tested with the program below using the command `java ea Test`.[†] The `ea` switch instructs JVM to enable assertions. The output should be `"Passes."`.

```
public class Test {
    public static void main(String[] args) {

        try {
            Line l1 = new Line(new Rational(1,3), new Rational(1));
            assert(l1.Includes(new Rational(1), new Rational(2)) == false);
            assert(l1.Includes(new Rational(3), new Rational(2)) == true);

            Rational slope = new Rational(1,2);
            Rational intercept = new Rational(0);
            line l2 = new Line(slope, intercept);
            assert(l2.Includes(new Rational(-1), new Rational(1)) == false);
            assert(l2.Includes(new Rational(0), new Rational(0)) == true);

            slope = new Rational(-2);
            intercept = new Rational(0);
            Line l3 = new Line(slope, intercept);
            assert(l3.Includes(new Rational(0), new Rational(1)) == false);
            assert(l3.Includes(new Rational(-2), new Rational(4)) == true);

            slope = new Rational(5);
            intercept = new Rational(-5);
            Line l4 = new Line(slope, intercept);
            assert(l4.Includes(new Rational(0), new Rational(5)) == false);
            assert(l4.Includes(new Rational(1), new Rational(0)) == true);

            slope = new Rational(0);
            intercept = new Rational(12);
            Line l5 = new Line(slope, intercept);
            assert(l5.Includes(new Rational(0), new Rational(0)) == false);
            assert(l5.Includes(new Rational(-100), new Rational(12)) == true);
        } catch(AssertionError e) {
            System.err.println("Does not pass.");
            System.exit(1);
        }

        System.out.println("Passes.");
    }
}
```

[†]We will copy and paste the contents of your submission in a file `Test.java` which already has the test code seen above and run it with assertions enabled.

Required methods are

- A constructor `Line(Rational, Rational)`,
- A pair of accessors `getSlope()` and `getIntercept()`,
- A pair of mutators `setSlope(Rational)` and `setIntercept(Rational)`,
- A method `Includes(Rational, Rational)` which takes the coordinates of a point and returns a boolean (`true` if the line passes through the point),
- `toString()` which returns the equation of the line like Equation (1)–that is, "y=slopex + intercept",
- `equals(Line)` which returns `true` if and only if the two lines are identical.

Submission format: A file `Line.java` which contains non-public classes `Line` and `Rational`. You are not required to provide any tests but are encouraged to test your code before submission.
Question 1: Can `Includes()` be made `static`? No, since this method needs access to the instance variables. To see if a line includes a point, there must be a line to begin with!
Question 2: What is the return value of setters? Shouldn't we perform input validation? Use `void`. No validation is required since any rational number can be used as slope or intercept.