

**Directions: SHOW ALL YOUR WORK. REMEMBER THAT PROGRAM SEGMENTS ARE TO BE WRITTEN IN JAVA.**

Notes:

- Assume that the classes listed in the Quick Reference found in the Appendix have been imported where appropriate.
- Unless otherwise noted in the question, assume that parameters in method calls are not `null` and that methods are called only when their preconditions are satisfied.
- In writing solutions for each question, you may use any of the accessible methods that are listed in classes defined in that question. Writing significant amounts of code that can be replaced by a call to one of these methods may not receive full credit.

1. An improper fraction in mathematics is represented by two quantities - the numerator and the denominator. In a reduced fraction, the numerator can be any integer, while the denominator must be a positive integer greater than 0. Below is the outline of a Fraction class.

```
public class Fraction
{
    private int numer, denomer;

    /** @param n is the numerator of a Fraction, d is the denominator
        */
    public Fraction(int n, int d)
    { /* to be implemented in part (c) */

    /**
     * reduces completely the fraction represented by numer and denomer
     *      Precondition: denomer != 0
     *      Postcondition: denomer > 0, Fraction is reduced
     */
    public void reduce()
    { /* to be implemented in part (b) */ }

    /**
     *      Precondition: denomer > 0
     * @return the fraction represented as a decimal value
     */
    public double getDecimalValue()
    { /* to be implemented in part (a) */ }

    /**
     * automatically quits the program
     */
    public void generateErrorAndQuit()
    { /* implementation not shown */ }
}
```

(a) Write method `getDecimalValue`. This method returns the decimal value represented by a fraction. For example, the result of calling `new Fraction(2, 4).getDecimalValue()` should be 0.5.

Complete method `getDecimalValue` below.

```
/**
 *      Precondition: denomer > 0
 *  @return the fraction represented as a decimal value
 */
public double getDecimalValue()
```

(b) Write method `reduce`. This method reduces the fraction to simplest form. For example, the fraction 10/12 would reduce to 5/6, the fraction 6/-8 would reduce to -3/4, and the fraction 0/5 would reduce to 0/1. HINT: Try to write code that will find the largest number that divides both the numerator and denominator first (the gcd, ignoring the sign), then use that number to simplify the numerator and the denominator. Then worry about special cases afterwards.

Complete method `reduce` below.

```
/**
 *  reduces completely the fraction represented by numer and denomer
 *      Precondition: denomer != 0
 *      Postcondition: denomer > 0, Fraction is reduced
 */
public void reduce()
```

(c) Write constructor `Fraction`, which takes two integers as its parameters. This constructor sets up the parts of the `Fraction` based on these values. The constructor should also deal with invalid inputs (the denominator being equal to 0 should cause the program to automatically quit), as well as make sure that the `Fraction` is in REDUCED form.

In writing `Fraction`, assume that `reduce` and `getDecimalValue` works as specified, regardless of what you wrote in part (a) or part (b).

Complete constructor `Fraction` below.

```
/** @param n is the numerator of a Fraction, d is the denominator
 */
public Fraction(int n int d)
```