Rational Numbers Class

Introduction

A number n is called a rational number if there are two integers a, b with $b \neq 0$ such that n can be expressed as $n = \frac{a}{b}$. We would like to develop a class called **`RationalNumber`** for performing arithmetic with fractions. The class uses integer variables to represent the private data, namely, the numerator and the denominator.

Program Input

Your program should read two integer values where the first integer represents the numerator and the second integer represents the denominator. The program must verify that the denominator is not zero. If it is zero, the entire input is rejected.

Design Requirements

- A rational number should be stored in reduced form (e.g., the fraction 2/4 would be stored in the object as 1 in the numerator and 2 in the denominator). This should always be the case whether the rational number is initialized, entered, or is the result of some arithmetic operation.
- The class provides:
    1. A constructor function that enables an object of the class to be initialized when it is declared. The constructor should contain default values in case no initial values are provided.
    2. Public utility member functions for each of the following:
        - **set** and **get** member functions for both the numerator and denominator.
        - Printing rational numbers in the form **a/b** where **a** is the numerator and **b** is the denominator.
        - Printing rational numbers in floating point format.
    3. Public member functions to perform the following arithmetic operations:
        - Addition of two rational numbers. The result should be stored in reduced form.
        - Subtraction of two rational numbers. The result should be stored in reduced form.
        - Multiplication of two rational numbers. The result should be stored in reduced form.
        - Division of two rational numbers. The result should be stored in reduced form.
        - The reciprocal of a rational number. The result should be stored in reduced form.
- The driver program enables
    1. Input and output of rational numbers through the overloaded >> and << operators, respectively.
    2. Addition, subtraction, multiplication, and division of two rational numbers as in algebra through overloading the $+$, $-$, $*$, and / operators, respectively.
    3. Assignment of one rational number to another through overloading the $=$ operator.
    4. Comparisons of two rational numbers through overload the $<$, $<=$, $==$, $!=$, $>$, and $\geq$ operators.

- The driver program must use the **`ArrayList`** Class developed in class to manipulate a list of rational numbers.

Problem Statement

Develop the **`RationalNumber`** class as described above. Write a driver program that uses the **`ArrayList`** Class developed in class to manipulate and test the **`RationalNumber`** class.

Submission Instructions

- Zip **ALL** files in the project in one zip file. The zip file must include all necessary files to run and test your program including but not limited to **`RationalNumber.h`**, **`RationalNumber.cpp`**, **`ArrayList.h`**, **`ArrayList.cpp`**, and **`MainDriver.cpp`**.
- Rename the zip file using the following naming convention: **LastName-FristName.zip**
- Submit the zip file in the drop box dedicated for this assignment.