CS/IT 122 Computer Programming 2 Prelim Group Project 1

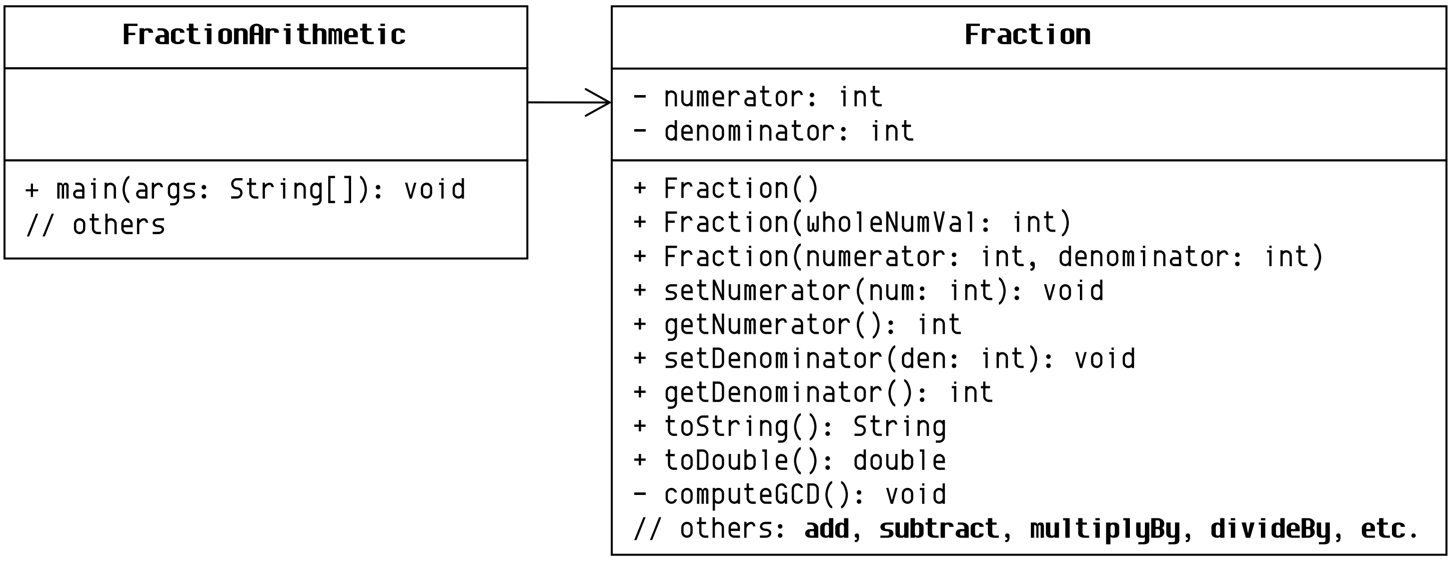
This is a project to be done by a group. Please follow the instructions given by the course facilitator as far as forming your group is concerned.

Problem Description:

1. Create a project in IntelliJ IDEA with the following naming convention:
   <class code><last names of the members (in alphabetical order)>PrelimGroupProject
   e.g. ***9300AranetaAyalaMontelibanoTanPrelimGroupProject***
2. Create a package within the project named "**prog2.pregroup**". All classes needed for this activity should be placed within this package.
3. Create a class called **Fraction**. The **Fraction** class should define a "new data type" such that it will be used as the ***data type*** for a variable that will hold a typical fraction. A typical fraction is composed of a numerator and a denominator.
4. Create a class called **FractionArithmetic**. The **FractionArithmetic** class should be an executable program (class having the ***main*** method) that will allow the user to compute the sum, difference, product and quotient of two fractions. When **FractionArithmetic** is executed, the user will be asked to enter the values of the numerator and denominator of both fractions and then a menu will be shown from which he/she can choose to enter value for fraction 1, enter value for fraction 2, add, subtract, multiply, divide the fractions, reduce a fraction, and quit from the program. The result (sum, difference, product, quotient, or reduce a fraction) should be displayed in the *typical fraction form* and its *equivalent decimal value*.

Requirement:

Complete the codes for **Fraction.java** and **FractionArithmetic.java**. The following diagram depicts a "has-a" relationship between the FractionArithmetic class the Fraction class(i.e. The arrow in the diagrams means that the FractionArithmetic class involves at least one variable that has Fraction as its data type). The following diagram also shows the UML class diagrams for the Fraction class and the FractionArithmetic class.


FractionArithmetic
+ main(args: String[]): void
// others
Fraction
- numerator: int
- denominator: int
+ Fraction()
+ Fraction(wholeNumVal: int)
+ Fraction(numerator: int, denominator: int)
+ setNumerator(num: int): void
+ getNumerator(): int
+ setDenominator(den: int): void
+ getDenominator(): int
+ toString(): String
+ toDouble(): double
- computeGCD(): void
// others: add, subtract, multiplyBy, divideBy, etc.

Notes:

- Your menu may look as follows: (although you're free to modify as you please)

```
+-----------------------------------------+
|  What do you want to do?                |
|     1. Enter value of fraction 1        |
|     2. Enter value of fraction 2        |
|     3. Add fractions                    |
|     4. Subtract fractions               |
|     5. Multiply fractions               |
|     6. Divide fractions                 |
|     7. Reduce a fraction                |
|     8. Quit                             |
+-----------------------------------------+
Enter choice:
```

- The result should show the formats ***numerator/denominator*** and ***decimal*** equivalent.
  Example:

```
result is equal to 7/2 or 3.5.
```

- Debug your programs as you see fit!
- You need to download and unarchive the file posted alongside this file and complete the initial codes provided for your project.

Euclid's Algorithm:

In Euclid's Elements (Book VII) we find a way of calculating the gcd of two numbers, without listing the divisors of either number. It is called Euclid's Algorithm.

For example:

We will find the gcd of 36 and 15. Divide 36 by 15, getting 2 with a remainder of 6. Then we divide 15 by 6 (the previous remainder) and we get 2 and a remainder of 3. Then we divide 6 by 3 (the previous remainder) and we get 2 with no remainder. The last non-zero remainder (3) is our gcd. Here it is in general:

```
a/b gives a remainder of r
b/r gives a remainder of s
r/s gives a remainder of t
...
w/x gives a remainder of y
x/y gives no remainder
```

In this case, **y** is the gcd of ***a*** and ***b***. If the first step produced no remainder, then **b** is the gcd.

Euclid's algorithm comes in handy with computers, because listing divisors is more difficult than the above algorithm. Large numbers are difficult to factor, while they are relatively easy to divide.

Euclid did his proof of his algorithm geometrically, believe it or not, as algebra had not been invented yet (during his time). His algorithm is considered to be one of the best examples of an efficient algorithm.

You are to use this algorithm in the implementation of the **`computeGCD`** method of the **`Fraction`** class.

Source:
http://www.jimloy.com/number/euclids.htm