# Module 2 Project - Fraction Calculator

This project is designed to help you practice building your own object class and testing it with a client class. You will be creating two classes, one called Fraction and the other called FractionCalculator. The Fraction class is an object that holds information about a fraction (numerator and denominator). It will have several constructors and both private and public methods implementing the behavior of a fraction. The FractionCalculator class is a class that will allow the user to enter in fractions and operations, calculating and displaying the result. It will run until the user tells it to quit. When this program is complete, you won't have to second guess your fraction arithmetic ever again!

## Part 1 - Fraction Class

Add the following to your Fraction Class:

### Fields

- two private instance variables to hold the numerator and denominator as ints

### Constructors

- two parameter constructor that initializes the numerator and demoninator.
- Fraction frac = new Fraction(4, 5) would create a Fraction with numerator equal to 4 and denominator equal to 5.
- This constructor should throw an IllegalArgumentException if the denominator is zero.
- If the user enters a negative denominator bump the negative sign to the numerator. For example, -3/-2 should be converted to 3/2. Likewise, 5/-3 should be converted to -5/3.
- one parameter constructor that initializes the object equal in value to the integer parameter.
- Fraction frac = new Fraction(3) would create a Fraction with numerator equal to 3 and denominator equal to 1.

- zero parameter constructor that initializes the object to 0, meaning the numerator is 0 and the denominator is 1
- Equivalent to new Fraction(0);

  You should eliminate as much redundancy as possible by letting your constructors rely on one another using the "this" keyword.

## Methods

| Method to implement | parameter | return | description |
|---|---|---|---|
| getNumerator() | none | int | exposes the value of the numerator field to the user |
| getDenominator() | none | int | exposes the value of the denominator field to the user |
| toString() | none | String | "numerator/denominator", a String representation of the Fraction |
| toDouble() | none | double | the result of numerator / denominator |
| add() | Fraction other | Fraction | returns a new Fraction that is the sum of other and this fractions |

| Method to implement | parameter | return | description |
|---|---|---|---|
| subtract() | Fraction other | Fraction | returns a new Fraction that is the difference between the other and this fraction |
| multiply() | Fraction other | Fraction | returns a new Fraction that is the product of the other and this fraction |
| divide() | Fraction other | Fraction | returns a new Fraction that is the division of the other and this fraction, throw an IllegalArgumentException() if the user askes you to divide by 0 |
| equals() | Object other | boolean | must take in an "Object" to properly override the Obect class's equals method, but should ultimately check if two fractions are equal |
| toLowestTerms() | none | none | converts the current fraction to the lowest terms |
| gcd() | int num, int den | int | takes in two ints and determines the greatest common divisor of the two ints, should be a static method |

**toLowestTerms()**

To convert a fraction to lowest terms we have to determine the greatest common divisor (factor) between the numerator and denominator. The greatest common divisor of two numbers a and b, is the largest number that evenly divides both a and b.

The Euclidean Algorithm is a fast method for determining the GCD of two numbers. Here is pseudocode for its implementation:

```
while a and b are not zero
    find the remainder of a divided by b
    set a to b
    set b to the remainder you found
return a
```

Implement a gcd() as a public static method that takes two integers as parameters and returns an int that is their greatest common divisor.

**equals()**

Override the Object equals() method so that it accurately determines whether or not two fractions are equal. In order to have it override, it has to take an Object as a parameter. Your method should check whether or not the parameter is an instanceof Fraction, since if it is not a Fraction it cannot be equal. Don't forget to cast the parameter to a Fraction after you check if it is an Object of type Fraction so that you can access its variables. Two fractions are equal if they represent the same number (i.e. 3/6 = 1/2 and -2/3 = 2/-3).

## Part 2 – FractionCalculator Class

In this section, you will implement a FractionCalculator class that has a main method and three helper methods. Here is a screenshot from a sample run:

```
This program is a fraction calculator
It will add, subtract, multiply and divide fractions until you type Q to quit.
Please enter your fractions in the form a/b, where a and b are integers.
--------------------------------------------------------------------------------
Please enter an operation (+, -, /, *, = or Q to quit): +
Please enter a fraction (a/b) or integer (a): 1/2
Please enter a fraction (a/b) or integer (a): 1/3
1/2 + 1/3 = 5/6
--------------------------------------------------------------------------------
Please enter an operation (+, -, /, *, = or Q to quit): -
Please enter a fraction (a/b) or integer (a): 4
Please enter a fraction (a/b) or integer (a): 3/2
4 - 3/2 = 5/2
--------------------------------------------------------------------------------
Please enter an operation (+, -, /, *, = or Q to quit): /
Please enter a fraction (a/b) or integer (a): 3/12
Please enter a fraction (a/b) or integer (a): 1/4
3/12 / 1/4 = 1
--------------------------------------------------------------------------------
Please enter an operation (+, -, /, *, = or Q to quit): *
Please enter a fraction (a/b) or integer (a): -2
Please enter a fraction (a/b) or integer (a): 5/2
-2 * 5/2 = -5
--------------------------------------------------------------------------------
Please enter an operation (+, -, /, *, = or Q to quit): =
Please enter a fraction (a/b) or integer (a): 50/100
Please enter a fraction (a/b) or integer (a): 1/2
50/100 = 1/2 is true
--------------------------------------------------------------------------------
Please enter an operation (+, -, /, *, = or Q to quit): q

Process finished with exit code 0
```

Your program should be robust so that it if the user enters invalid input it will continue to re-prompt them until it is valid. Here is an example run where the user is confused and enters invalid input:

```
/Library/Java/JavaVirtualMachines/jdk1.8.0_65.jdk/Contents/Home/bin/java ...
This program is a fraction calculator
It will add, subtract, multiply and divide fractions until you type Q to quit.
Please enter your fractions in the form a/b, where a and b are integers.
--------------------------------------------------------------------------------
Please enter an operation (+, -, /, *, = or Q to quit): foo
Invalid input (+, -, /, *, = or Q to quit): operation
Invalid input (+, -, /, *, = or Q to quit): 1/2
Invalid input (+, -, /, *, = or Q to quit): +-
Invalid input (+, -, /, *, = or Q to quit): / *
Invalid input (+, -, /, *, = or Q to quit): /
Please enter a fraction (a/b) or integer (a): three
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: one/2
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: a/b
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: 1 3
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: 3/0
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: 1/3
Please enter a fraction (a/b) or integer (a): 0/2
1/3 / 0 = Undefined
--------------------------------------------------------------------------------
Please enter an operation (+, -, /, *, = or Q to quit): q

Process finished with exit code 0
```

## Methods

| Method to implement | parameter | return | description |
|---|---|---|---|
| getOperation() | Scanner input | String | Asks the user to enter in a valid mathematical operation. If the user enters anything except “+”, “-“, “/”, “*”, “=”, “q”, or “Q” it should re-prompt them until there is valid input. |
| validFraction() | String input | boolean | returns true if the parameter is in the form "a/b" where a is any int and b is any positive int |

| Method to implement | parameter | return | description |
|---|---|---|---|
| getFraction() | Scanner input | Fraction | It prompts the user for a String that is a validFraction. If they enter any thing that is not a valid Fraction, it should re-prompt them until it is valid |

Here is example output from a call to **getOperation()**:

```
Please enter a fraction (a/b) or integer (a): fraction!
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: three
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: two/one
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: 3/two
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: 1 / 2
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: 0.5
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: 2/-1
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: -31
```

At the end of this run, getOperation would have returned "*".

Some things to be mindful of when implementing the **validFraction()** method:

- The first character may or may not be a '-' character. If a negative shows up anywhere else, then it is not a valid fraction. It may be helpful to remove the '-' character if there is one.
- If there is no '/' character, then every character in the string must be a number (if you removed the '-' sign).
- If there is a '/' character, then it may be helpful to create substrings for the numerator and denominator.
- Both substrings must be non-empty.
- Both must be entirely made of numbers.
- The denominator cannot be "0". **Hint 1**: It may be useful to create a helper method isNumber() that takes a String as input and returns true if every

character in the String is a number 0-9 and false otherwise. This method can also check for empty strings. **Hint 2**: Once you determine whether or not the Strings are numbers, you may find the Integer.parseInt() method helpful.

Here is example output from a call to **getFraction()**. If the user enters any thing that is not a valid Fraction, it should re-prompt them until it is valid:

```
Please enter a fraction (a/b) or integer (a): fraction!
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: three
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: two/one
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: 3/two
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: 1 / 2
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: 0.5
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: 2/-1
Invalid fraction. Please enter (a/b) or (a), where a and b are integers and b is not zero: -31
```

This call would return a new Fraction object equal to -31/1. No user input should throw an exception! If you are getting exceptions, then it is likely your validFraction method isn't correct.

## Part 3 - Putting it all together!

- Write a short introduction method that describes the calculator program and welcomes your user
- Ask the user to enter in an operation
- As long as the user enters something that's not "q" or "Q" when asked for an operation you should run the calculator.
- Get two fractions from the user and then perform whichever operation they ask for

- Printing the result of the operation Here is an example run of the entire program:

```
This program is a fraction calculator
It will add, subtract, multiply and divide fractions until you type Q to quit.
Please enter your fractions in the form a/b, where a and b are integers.
--------------------------------------------------------------------------------
Please enter an operation (+, -, /, *, = or Q to quit): +
Please enter a fraction (a/b) or integer (a): 1/2
Please enter a fraction (a/b) or integer (a): 1/3
1/2 + 1/3 = 5/6
--------------------------------------------------------------------------------
Please enter an operation (+, -, /, *, = or Q to quit): -
Please enter a fraction (a/b) or integer (a): 4
Please enter a fraction (a/b) or integer (a): 3/2
4 - 3/2 = 5/2
--------------------------------------------------------------------------------
Please enter an operation (+, -, /, *, = or Q to quit): /
Please enter a fraction (a/b) or integer (a): 3/12
Please enter a fraction (a/b) or integer (a): 1/4
3/12 / 1/4 = 1
--------------------------------------------------------------------------------
Please enter an operation (+, -, /, *, = or Q to quit): *
Please enter a fraction (a/b) or integer (a): -2
Please enter a fraction (a/b) or integer (a): 5/2
-2 * 5/2 = -5
--------------------------------------------------------------------------------
Please enter an operation (+, -, /, *, = or Q to quit): =
Please enter a fraction (a/b) or integer (a): 50/100
Please enter a fraction (a/b) or integer (a): 1/2
50/100 = 1/2 is true
--------------------------------------------------------------------------------
Please enter an operation (+, -, /, *, = or Q to quit): q

Process finished with exit code 0
```

## Part 4 - Hacker Problem - FractionCalculatorAdvanced

Create another class called FractionCalculatorAdvanced. You may cut and paste useful methods from FractionCalculator. The key difference between FractionCalculator and FractionCalculatorAdvanced is that the user can enter in their operations on a single line. Allow the user to enter their input onto a single line. Your program must be robust so that if the user enters in invalid input, it will re-prompt them until they either enter a q to quit or a valid operation. It is possible to do this without try/catch, but it is quite difficult. You may read about try/catch blocks here: http://beginnersbook.com/2013/04/try-catch-in-java/.

Here is sample output from a run of FractionCalculatorAdvanced:

```
/Library/Java/JavaVirtualMachines/jdk1.8.0_65.jdk/Contents/Home/bin/java ...
This program is a fraction calculator
It will add, subtract, multiply and divide fractions until you type Q to quit.
Valid operations are of the form "[FRAC] [OPERATION] [FRAC]".
[FRAC] can be either a single integer or two integers separated by "/".
[OPERATION] can be +, -, *, / or =.
Enter an operation (q to quit): this
Invalid operation. Must be "[FRAC] [OPERATION] [FRAC]".
Enter an operation (q to quit): isn't
Invalid operation. Must be "[FRAC] [OPERATION] [FRAC]".
Enter an operation (q to quit): a valid
Invalid operation. Must be "[FRAC] [OPERATION] [FRAC]".
Enter an operation (q to quit): op + ation
Invalid operation. Must be "[FRAC] [OPERATION] [FRAC]".
Enter an operation (q to quit): 2 + foo
Invalid operation. Must be "[FRAC] [OPERATION] [FRAC]".
Enter an operation (q to quit): 3 - zee
Invalid operation. Must be "[FRAC] [OPERATION] [FRAC]".
Enter an operation (q to quit): 45 / 22
45 / 22 = 45/22
Enter an operation (q to quit): 33 / 11
33 / 11 = 3
Enter an operation (q to quit): 1/2 * 4/3
1/2 * 4/3 = 2/3
Enter an operation (q to quit): 11/14 / 7/11
11/14 / 7/11 = 121/98
Enter an operation (q to quit): q

Process finished with exit code 0
```