```java
package structure;

import structure.exceptions.FractionException;

public class Fraction
{
	public int m; //numerator
	public int n; //denominator

	public Fraction (int m, int n) throws FractionException
	{
		if (n == 0)
		{
			throw new FractionException(new String("Divided by
zero.(n != 0)"));
		}

		this.m = m;
		this.n = n;

	}
	public Fraction ()
	{
		this.m = 0;
		this.n = 1;

	}

	public Fraction (int m) throws FractionException
	{
		this.m = m;
		this.n = 1;
		if (n == 0)
		{
			throw new FractionException(new String("Divided by
zero.(n != 0)"));
		}
		Short(this);
	}


	public Fraction operator (Fraction frct1, char symbolOperator,
Fraction frct2 ) throws FractionException
	{
		if ( (frct1 == null) || (frct2 == null))
		{
			throw new FractionException(new String("Why are u
kidding me bro it's not fraction it's null!"));
		}
```

```
        switch (symbolOperator){
            case '-' :
            {
                frct1.m = (frct1.m*frct2.n - frct2.m*frct1.n);
    // a/b - c/d = (a*d - c*b) / (b*d)
                frct1.n = frct1.n*frct2.n;
    //b*d
                Short(frct1);//
                break;
            }

            case '+' :
            {
                frct1.m = (frct1.m*frct2.n + frct2.m*frct1.n);
    // a/b + c/d = (a*d + c*b) / (b*d)
                frct1.n = frct1.n*frct2.n;
    //b*d
                Short(frct1);//
                break;
            }

            case '*' :
            {
                frct1.m = frct1.m*frct2.m ;
        // a/b * c/d = (a * c) / (b*d)
                frct1.n = frct1.n*frct2.n;
    //b*d
                Short(frct1);//
                break;
            }

            case '/' :
            {
                frct1.m = frct1.m*frct2.n ;
        // a/b / c/d = (a * d) / (b*c)
                frct1.n = frct1.n*frct2.m;
    //b*c
                Short(frct1);//
                break;
            }

            default: {throw new FractionException(new
String("Invalid operator (+ - / *)"));}
        }

        return frct1;

    }
```

```
	private void Short (Fraction frct)  throws FractionException
	{
		if (frct == null)
		{
			throw new FractionException(new String("Why are u 
kidding me bro it's not fraction it's null!"));
		}

		int[] primeNumbers;
		int size;

		if (n<0) {

			frct.m = frct.m*(-1);
			frct.n = frct.n*(-1);

		}

		int max;
		if(m>n){
			max=m;
		}
		else{
			max=n;
		}
		size = (int) Math.round(Math.sqrt((double) max)) + 1;
		primeNumbers = new int[size];

		primeNumbers[0] = 2;
		int count=0;
		for(int  i=2; i<size ; i++){

			for(int j=0; j<i; j++)
			{
				if (primeNumbers[j] == 0)
					{
						primeNumbers[j] = i;
						count= j;
						break;
					}

				else if (i % primeNumbers[j] == 0) break;
			}
		}

		primeNumbers[++count] = max;

		int i = 0;
		while(i < count){
```

```
                        if((m%primeNumbers[i] == 0) && (n%primeNumbers[i] ==
0))
                        {
                                frct.m = m / primeNumbers[i];
                                frct.n = n / primeNumbers[i];
                        }
                        else i++;
                }

        }

        public String toString(Fraction frct) throws FractionException
        {
                if (frct == null)
                {
                        throw new FractionException(new String("Why are u
kidding me bro it's not fraction it's null!"));
                }
                String s = new String();

                s += m;
                s += '/';
                s += n;

                return s;
        }
}
```