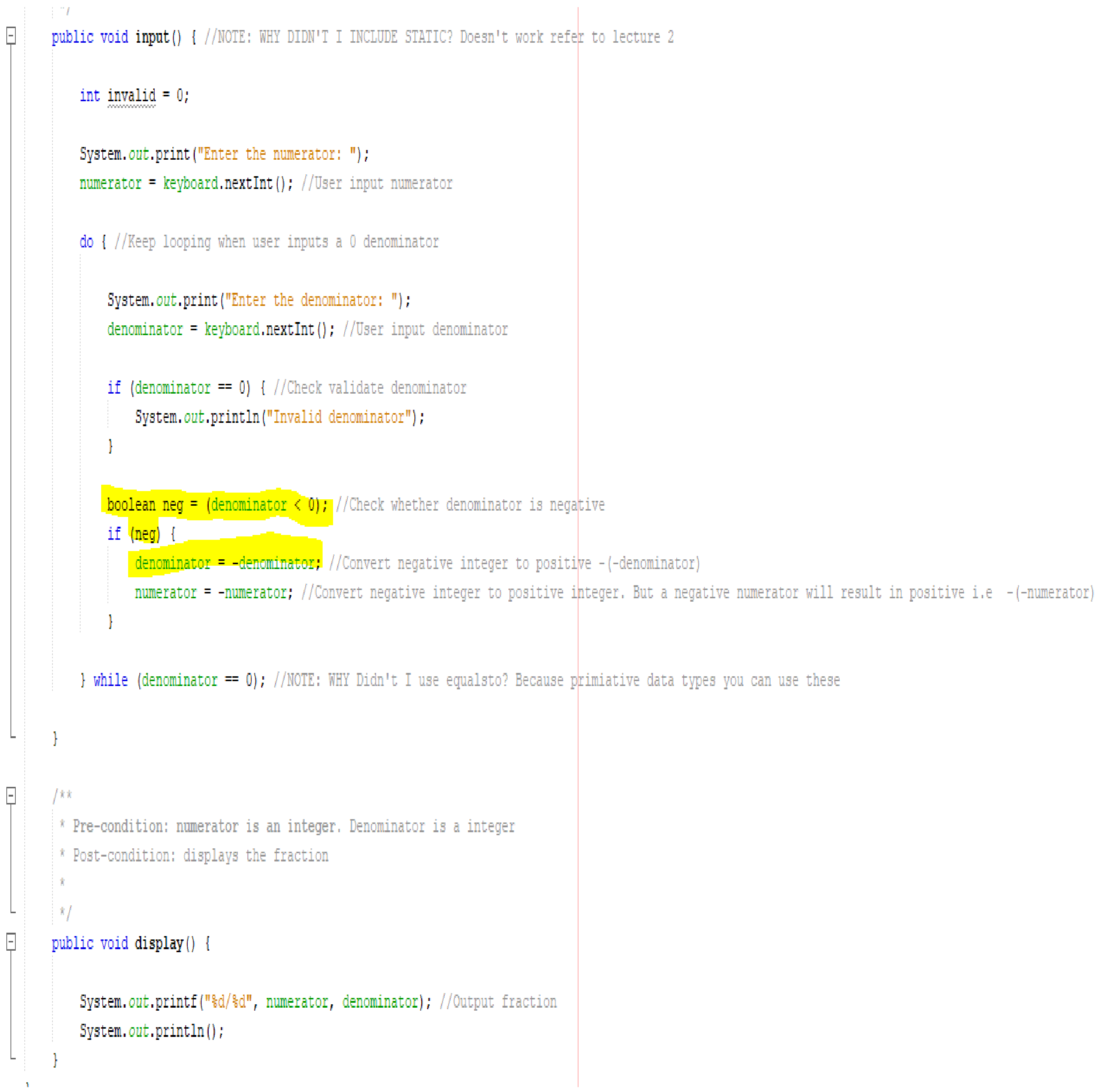
2)

Get two number representing numerator and denominator + Convert negative numbers ←→ positive numbers + Output as fraction

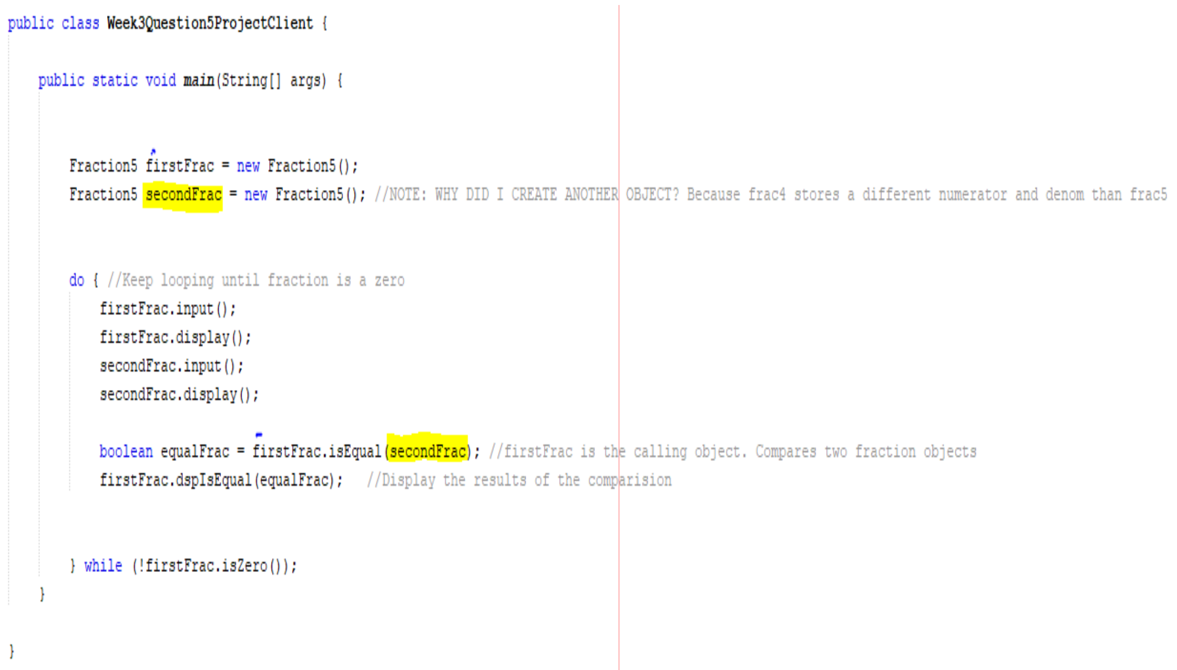


```java
 */
public void input() { //NOTE: WHY DIDN'T I INCLUDE STATIC? Doesn't work refer to lecture 2

    int invalid = 0;

    System.out.print("Enter the numerator: ");
    numerator = keyboard.nextInt(); //User input numerator

    do { //Keep looping when user inputs a 0 denominator

        System.out.print("Enter the denominator: ");
        denominator = keyboard.nextInt(); //User input denominator

        if (denominator == 0) { //Check validate denominator
            System.out.println("Invalid denominator");
        }

        boolean neg = (denominator < 0); //Check whether denominator is negative
        if (neg) {
            denominator = -denominator; //Convert negative integer to positive -(-denominator)
            numerator = -numerator; //Convert negative integer to positive integer. But a negative numerator will result in positive i.e  -(-numerator)
        }

    } while (denominator == 0); //NOTE: WHY Didn't I use equalsto? Because primiative data types you can use these

}

/**
 * Pre-condition: numerator is an integer. Denominator is a integer
 * Post-condition: displays the fraction
 *
 */
public void display() {

    System.out.printf("%d/%d", numerator, denominator); //Output fraction
    System.out.println();
}
```

5)

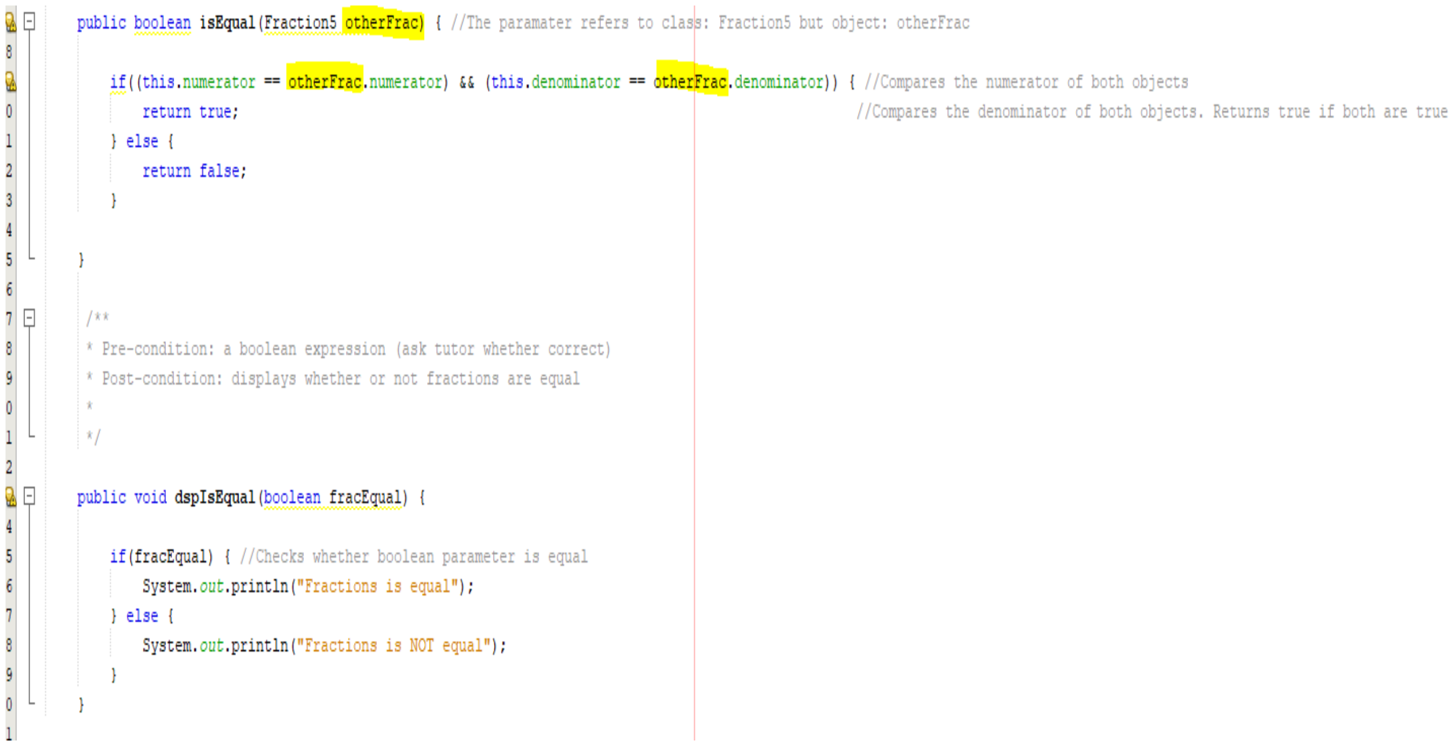
Create two fraction objects + <u>Compare two objects from same class </u>+ Output fraction object equal

Client program→



```
public class Week3Question5ProjectClient {

    public static void main(String[] args) {


        Fraction5 firstFrac = new Fraction5();
        Fraction5 secondFrac = new Fraction5(); //NOTE: WHY DID I CREATE ANOTHER OBJECT? Because frac4 stores a different numerator and denom than frac5


        do { //Keep looping until fraction is a zero
            firstFrac.input();
            firstFrac.display();
            secondFrac.input();
            secondFrac.display();

            boolean equalFrac = firstFrac.isEqual(secondFrac); //firstFrac is the calling object. Compares two fraction objects
            firstFrac.dspIsEqual(equalFrac);   //Display the results of the comparision


        } while (!firstFrac.isZero());
    }

}
```

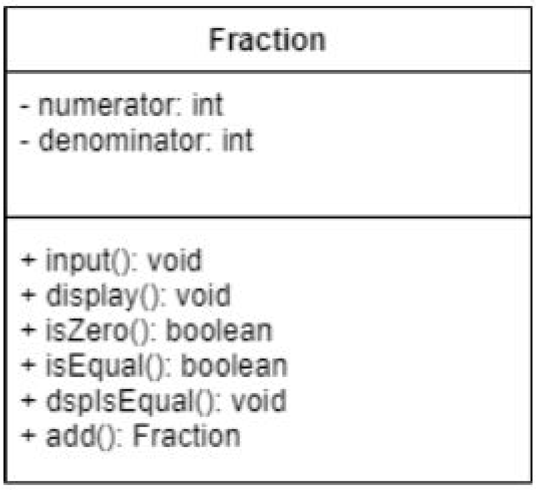
Class →



```
    public boolean isEqual(Fraction5 otherFrac) { //The paramater refers to class: Fraction5 but object: otherFrac

        if((this.numerator == otherFrac.numerator) && (this.denominator == otherFrac.denominator)) { //Compares the numerator of both objects
            return true;                                                              //Compares the denominator of both objects. Returns true if both are true
        } else {
            return false;
        }

    }

    /**
     * Pre-condition: a boolean expression (ask tutor whether correct)
     * Post-condition: displays whether or not fractions are equal
     *
     */

    public void dspIsEqual(boolean fracEqual) {

        if(fracEqual) { //Checks whether boolean parameter is equal
            System.out.println("Fractions is equal");
        } else {
            System.out.println("Fractions is NOT equal");
        }
    }
```

3)

Draw UML Class:


Fraction
- numerator: int
- denominator: int
+ input(): void
+ display(): void
+ isZero(): boolean
+ isEqual(): boolean
+ dspIsEqual(): void
+ add(): Fraction