|  |  |
| --- | --- |
| **Exposure Java** | **Lab 09b** |
| **The Rational Class Program II** | **80 & 100 Point Versions** |
| **Assignment Purpose:**  The purpose of this lab is to demonstrate knowledge of creating a class with object methods, instantiate multiple objects of the created class, and then call the object methods from the main program method. Additionally, this lab requires knowledge of the GUI input output and string to number conversion. | |

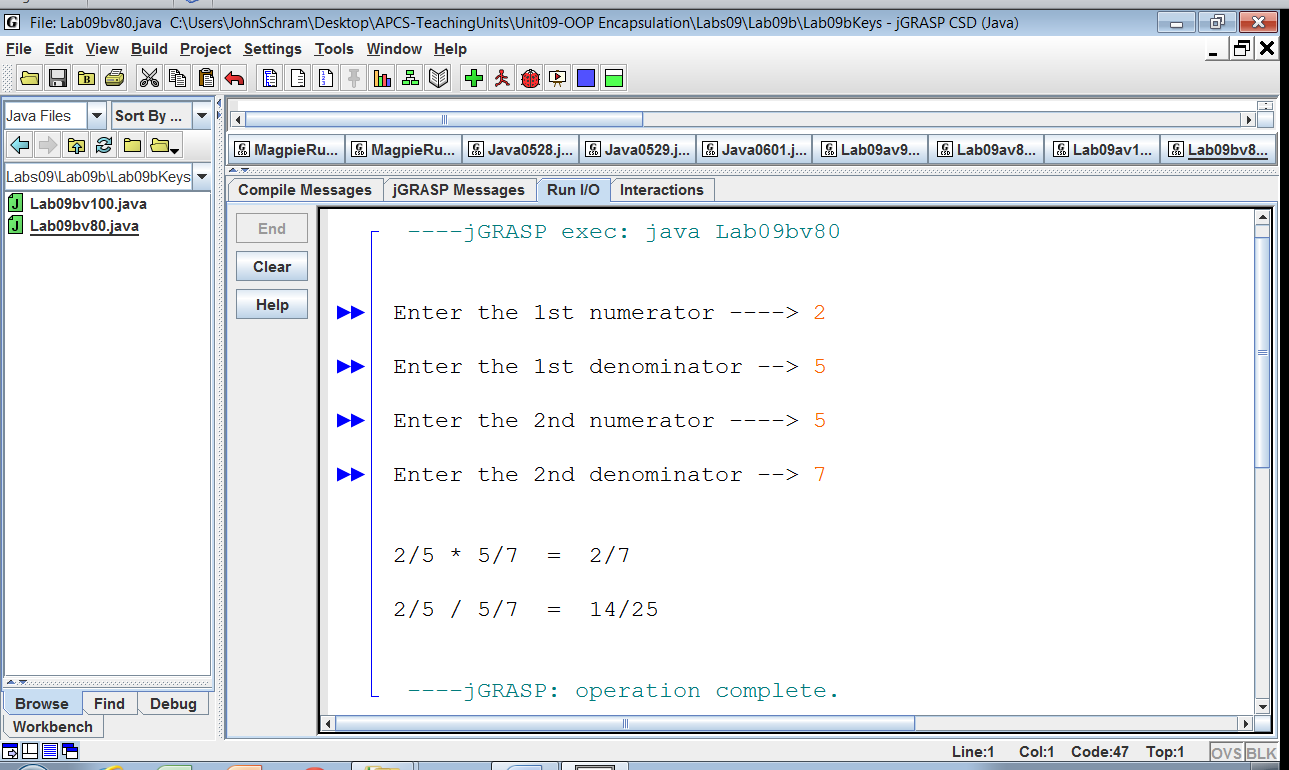
This lab assignment continues the **Rational** class that was started with the **Lab09a** assignment. Now comes the times to add, subtract, multiply and divide fractions with your **Rational** class. For this assignment you will need to complete the **Rational** class and also handle most of the code in the **main** method.

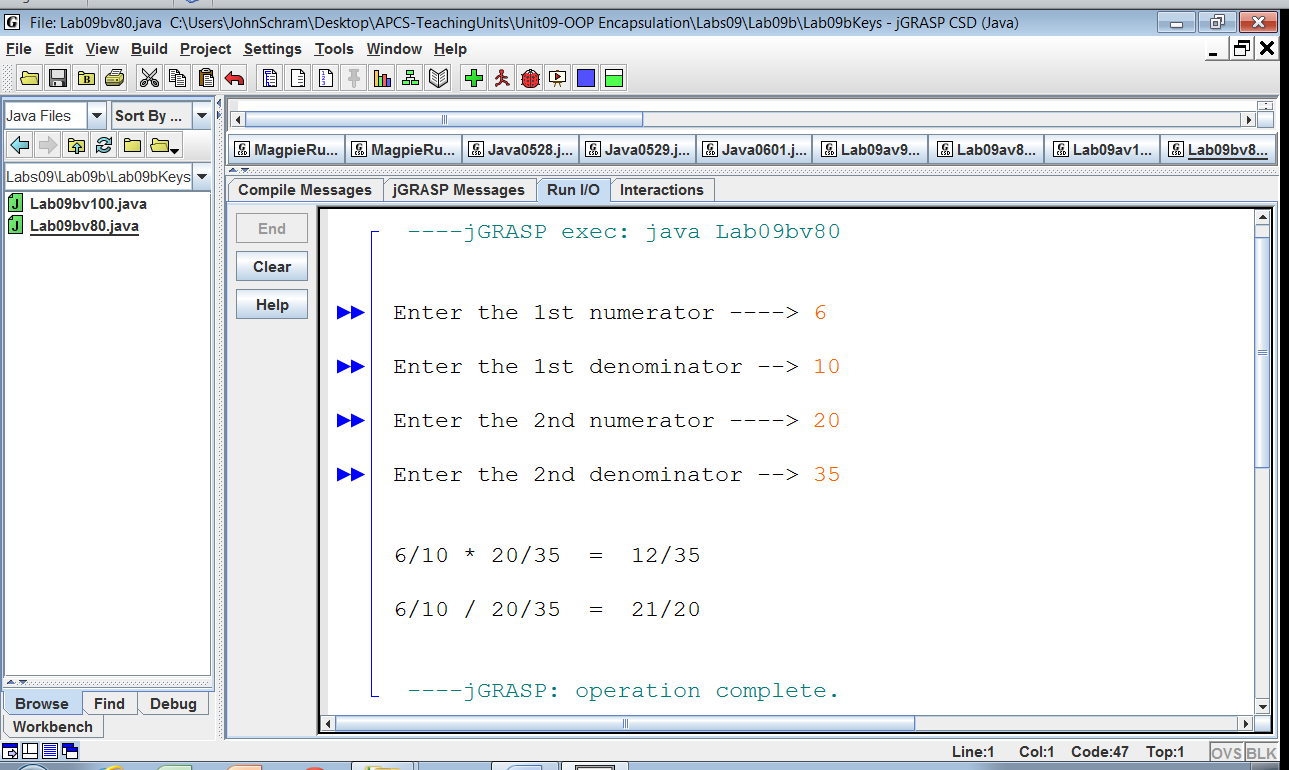
|  |  |
| --- | --- |
| **Lab09bvst Student Version** | **Do not copy this file, which is provided.** |
| // Lab09bvst.java // The Rational Class Program II // This is the student, starting version of the Lab09b assignment.   import java.util.Scanner;   public class Lab09bvst {  private static int num1, den1; // numerator and denominator of the 1st rational number  private static int num2, den2; // numerator and denominator of the 2nd rational number   public static void main (String[] args)  {  enterData();   Rational r1 = new Rational(num1,den1);  Rational r2 = new Rational(num2,den2);  Rational r3 = new Rational();   r3.multiply(r1,r2);  System.out.println("\n\n" + r1.getOriginal() + " \* " + r2.getOriginal() + " = " + r3.getReduced());  r3.divide(r1,r2);  System.out.println("\n" + r1.getOriginal() + " / " + r2.getOriginal() + " = " + r3.getReduced());  // 100 Point Version Only // r3.add(r1,r2); // System.out.println("\n" + r1.getOriginal() + " + " + r2.getOriginal() + " = " + r3.getReduced()); // r3.subtract(r1,r2); // System.out.println("\n" + r1.getOriginal() + " - " + r2.getOriginal() + " = " + r3.getReduced());  System.out.println();  }  public static void enterData()  {  Scanner input = new Scanner(System.in);  System.out.print("\nEnter the 1st numerator ----> ");  num1 = input.nextInt();  System.out.print("\nEnter the 1st denominator --> ");  den1 = input.nextInt();  System.out.print("\nEnter the 2nd numerator ----> ");  num2 = input.nextInt();  System.out.print("\nEnter the 2nd denominator --> ");  den2 = input.nextInt();  } }   class Rational {  private int firstNum; // entered numerator  private int firstDen; // entered denominator  private int reducedNum; // reduced numerator  private int reducedDen; // reduced denominator  private int gcf; // greatest common factor     private void getGCF(int n1,int n2)  {  int rem = 0;  do  {  rem = n1 % n2;  if (rem == 0)  gcf = n2;  else  {  n1 = n2;  n2 = rem;  }  }  while (rem != 0);  }  } | |

**80 Point Version Specifics**

For the 80-point version you need to write two constructor **Rational** methods and methods **reduce**, **getReduced**, and **getOriginal**. You will also need to write methods **multiply** and **divide**. Your fractions need to be reduced. Not only do you need to write the **reduce** method, but you also need to call it in the appropriate places so that the product and quotient will be displayed in lowest terms.

**80 Point Version Outputs**

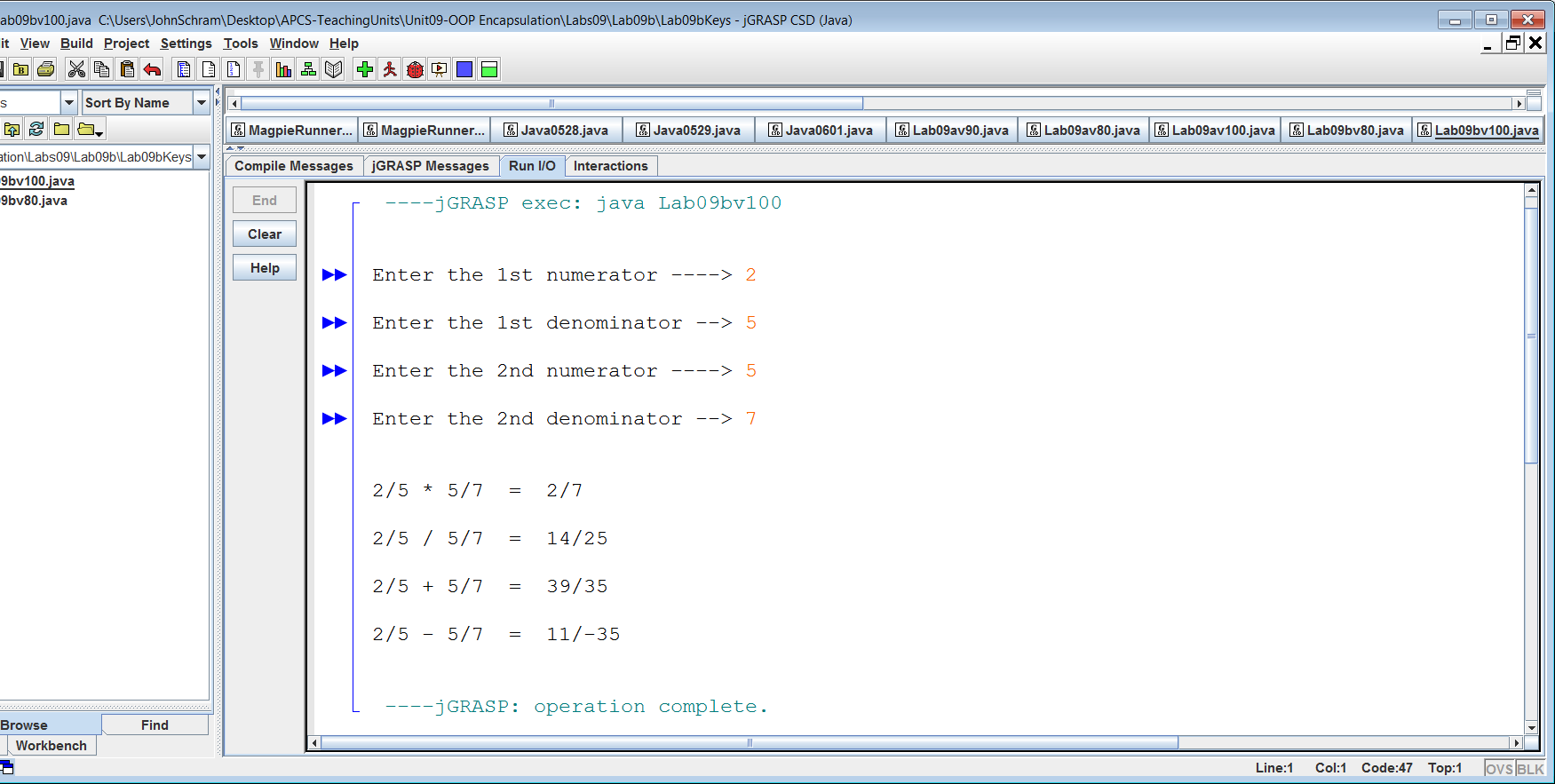




**100 Point Version Specifics**

The 100-point version completes the **Rational** class with methods **add** and **subtract**. The sum and difference also need to be displayed in lowest terms.

**100 Point Version Outputs**



NOTE: In the subtraction answer, the negative sign might show up in the numerator.

It also might show up in the denominator. Either way is fine.

