Write a program that repeatedly asks for the numerators and denominators of two fractions and then outputs the sum of the two fractions. For full credit program output must look like following, don’t need to turn in algorithm.

This program adds fractions. ‘Y’ continues, any other key exits program

Enter numerator 1 ==> 1
Enter denominator 1 ==> 3

Enter numerator 2 ==> 1
Enter denominator 2 ==> 6

\[
\begin{array}{ccc}
1 & 1 & 1 \\
\hline
3 & 6 & 2
\end{array}
\]

Continue? Y or N! ==> n

Hints and Suggestions:

1) The easiest way to add fractions is to use formula

\[
\frac{a}{b} + \frac{c}{d} = \frac{a \times d + b \times c}{b \times d}
\]

2) After using this formula, the answer still needs to be reduced to lowest terms. One way to do this is to divide the numerator and denominator of the answer by their Greatest Common Divisor, GCD.

3) Program does not have to deal with negative numbers or fractions with large numerators and denominators, let’s use a maximum number of 50 for either numerator or denominator. It should handle fractions with a zero in numerator, but not a zero in denominator, don’t test for this case.

4) Output of the form below is acceptable, but worth 7 less points.

\[
1/3 + 1/6 = 1/2
\]

5) Output not reduced to lowest terms will be worth 7 less points.

6) Output answer may be left in improper fractions, but in reduced terms.

This program must be written using functions and procedures. Each should be logically coherent and designed correctly. This program needs to be written with five total routines; the main, and the four others, one to read the inputted data, one to calculate the sum of the two fractions, one to find the Greatest Common Divisor between the numerator and denominator and the last will display the end result to the screen.

Make sure that all functions and procedures are commented correctly and tell what they do along with parameters and any return value. Remember the main should not contain any messy detail, should be relatively short and concise.

We will go over the algorithm to find the GCD in next class. Turn in only source code and output, don’t need to turn in an algorithm, but you should still write one.