Problem # 1:

We are given a **school**. In the school there are classes of students. Each class has a set of teachers. Each teacher teaches a set of disciplines(courses). Students have name and unique class number. Classes have unique text identifier. Teachers have name. Disciplines have name, number of lectures and number of exercises. Both teachers and students are people. Students, classes, teachers and disciplines could have optional comments (free text block). Your task is to identify the classes (in terms of OOP) and their attributes and operations, encapsulate their fields,
Write a winform application to test your class.

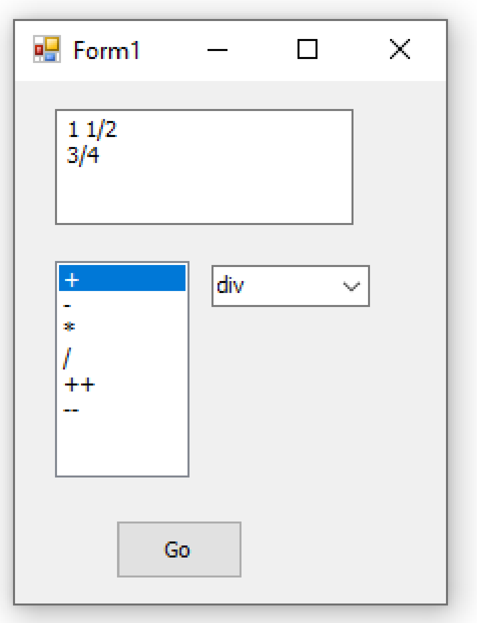
==

Problem # 2:

In C#, Define a class for rational numbers named Rational. A rational number is a number that can be represented as two intergers with division indicated, but the division is not carried out. For example, 1/2, 3/4, 64/2, and so forth. Also, I wanted to be able to express the number both as a simple fraction and as a whole and fractional part. For example "3/2" or "1 1/2". The format string "S" is for the simple style and "L" for the longer.
The default constructor should initialized the numerator to zero and the denominator to 1 (0/1).
Define an overloaded constructor which takes two arguments of the type long. The first argument is the numerator and the second is the denominator. Define a read-only property numerator and one to get the denominator.
Override the ToString method, which provides a string respentation of the data in the object, in such as war that returns a string version of the rational number which includes the numerator, a slash, and the denominator (3/5) (simple format) and the long format that returns a string version of the rational number which includes the whole number ,the numerator, a slash, and the denominator(1 1/2).
Allow simple arithmetic writteing a group of function such as Add, substract, multiply and divide.
Allow simple arithmetic by overloading the following math operators: + (Rational+Rational, long+ Rational, Rational+long), -(Rational-Rational, long-

Rational, Rational-long), *((Rational*Rational, long*Rational, Rational*long)), /((Rational/Rational, long/Rational, Rational/long)),++,--.

Allow for the comparison of two objects of the type Rational, by overloading the following comparison operators: ==, !=, <, >.



1 1/2
3/4