Reverse Polish Calculator

Rules

1. Strictly practice TDD
    a. Follow the: Red-Green-Refactor cycle
    b. Stop and read your code before refactoring.
2. Practice wishful coding
    a. Do not write any production code until you have a failing test.
    b. Or compilation failures to drive the need to implement the method.

The Kata

Implement each requirement without looking forward to the next.
Use the following interface:

```
public interface ICalculator
{
   int Calculate(string input);
}
```

Requirement 1

Calculate() will be given a string with two integers, separated by spaces, and should return the result.
E.g. `calculator.Calculate("1 2"); // should return 3`

Requirement 2

This calculator is going to be used to display a sum on a limited-size display. We cannot display more than 8 characters on this display and the Calculator should throw an InvalidOperationException if the value cannot be displayed.

- The negative sign counts as a character! Can your calculator cater for this?

Requirement 3

The consumer of this calculator (Jane) is truly pleased not to have to deal with the math. She is rather fond of Reverse Polish notation, so she asked for the following change: she would like to specify the operation as a third parameter in the string so that she could use the same class to add and find the difference between two integer values. Jane only requires support for + and – operations.

```
calculator.Calculate("12 23 +"); // should return 25
calculator.Calculate("43 13 -"); // should return 30
```

Jane would like the default operator to be +. E.g. `calculator.Calculate("23 4"); // should still return 27`

Requirement 4

Some end-users are inputting comma-separated numbers instead, so Jane would like to support that.
Valid inputs could now include:
"4 67"
"6 9 +"
"23, 45 -"
"8,5 +"

Requirement 5

Some end-users are forgetting the space before the operand, so we also need to support, for example: "4,5+";

Requirement 6

Jane would like to be able to multiply by the same method as + and –

Requirement 7

If you used a switch or series of if statements, refactor your code to remove the switch and/or if statements to handle the various operations (+ - *).

Requirement 8

Implement division ("/") for the calculator.

Requirement 9

Implement modulo ("%") for the calculator

Requirement 10

Implement exponent ("^") for the calculator. E.g. "2 4 ^" = 16