Refactoring exercise

You are given a "working" calculator program. It is a quick and dirty programming job. Make it as pretty as possible!

What is an RPN calculator?

Hewlett Packard calculator: HP 65 - https://www.cuveesoft.ch/rpn65/
RPN = reverse Polish notation.
Instead of writing complex expressions with parentheses, let the operator come last

Normal notation	RPN
1 + 2	1 2 +
3 * 5 + 1	3 5 * 1 +
3 * (5 + 1)	3 5 1 + *

The calculator keeps the numbers on a stack.
An entered number is pushed on top of the calculator stack.
An entered operator pops the two top numbers, performs the operation and pushes back the result. Play with it, and you will understand.
(try adding one number, enter, then one more number, enter then one operator!)

Refactoring

Code smells

Find as many smells as you can find, e.g.

- Substandard naming
- Duplicated code
- Long methods with all kinds of responsibilities
- Why have our own DoubleStack ?
- Get rid of the switch?
- Is the DoubleStack.ToString really OK?

Refactor to beauty!

Separation of concerns

The user interface fixed to be Console.Read/WriteLine.
What if we wanted to have another user interface?
Can you separate the calculator logic from the presentation code with an interface?
And then write another kind of user interface? Web, WebForms etc.

Separate the actual calculator logic from the code that controls the flow of data from UI to calculator. Introduce a Controller to do that!

Fragile code

Handling of incorrect input is fragile, only works in sunshine. Fix!
Binary operators invoked with only 1 value on the stack behaves badly. Fix!

This is an exercise. There are many different beautiful solutions!