HBM Prenscia Calculator Project

1 Project

Implement a Calculator class using C#.NET, VB.NET or Java. If you use Visual Studio, we ask that you send the entire solution, including applicable GUI.

1.1 Part 1: Basic Implementation

The class must implement the following methods:

Calculator(left, right, operator)

- Constructor, all three parameters are strings
- valid values for left and right are numbers
- valid values for operator are: "+", "-", "*", "/"

setLeft(value)

- assigns the left side of the equation
- value is a string representing a number

setRight(value)

- assigns the right side of the equation
- value is a string representing a number

setOperator(value)

- assigns the operator
- value is one of: "+", "-", "*", "/"

calculate()

- performs the calculation using left, right, and operator

getResult()

- returns a string representing the current result
- may return an error message

getPreviousResult(index)

- returns a previous result
- index = 1 returns the result immediately prior, index = 2 the one before that, etc
- index is an integer from 1 to 10

Consider various errors that could occur in your implementation, such as:

- left/right/operator parameters are invalid (empty strings, null, letters, etc)
- arithmetic causes an overflow (passing two very large numbers and "+"))
- invalid operations (params like left:"1", right:"0", operator:"/")

Feel free to implement unit tests if desired. Here are some example parameters and the expected return values from the calculate() method:

left:"9", right:"1", operator:"+" returns "10"

left:"1", right:"7", operator:"-" returns "-6"

left:"8", right:"8", operator:"*" returns "64"

left:"12", right:"3", operator:"/" returns "4"

left:"256", right:"0", operator:"/" returns an error message

left:"", right:"423", operator:"+" returns an error message

left:"foo", right:"14", operator:"+" returns an error message

Errors returned by getResult() are undefined, but should be distinct enough to indicate what happened.

An application would use the Calculator class as follows (pseudo-code):

```
myCalc = new Calculator("5", "1", "+")
myCalc.calculate()
print("5 + 1 = " + myCalc.getResult())
```

```
myCalc.setLeft("3")
myCalc.setRight("9")
myCalc.setOperator("*")
myCalc.calculate()
print("3 * 9 = " + myCalc.getResult())
print("previous result: " + myCalc.getPreviousResult(1))
```

The output from the above program would be:

```
"5 + 1 = 6"
"3 * 9 = 27"
"previous result: 6"
```

1.2 Part 2: Refactoring

The Calculator class is a contrived example, which has a very deliberate API. Implement an alternate class that accomplishes the same goal (simple arithmetic and storing the last 10 results) but with a less verbose API.

This second implementation should implement a method that returns the result of a calculation:

```
calculate(equation)
```

Where equation may contain any number of operands and operators. The format of an equation is "<number> <operator> <number2> <operator2> ... <numberN> <operatorN>", starting and ending with a digit. Support for parentheses is not required.

Retain the previous functionality of storing prior results.