Calculator Web API

Brief discussion on the solution

# Introduction

This document discusses about the implementation of math expression calculator where it converts input string into numbers and arithmetic operations, then calculate the result.

Total time logged: 8 hours

# Development Tools

* Visual Studio 2019
* Postman
* Targeting Framework: Net Core 3.0
* Unit Test: NUnit Framework

# Solution Structure

The solution consists of 2 projects, the Calculator Web API and unit testing. The API is designed with a main HTTP POST method to perform calculation.

Development endpoint: <https://localhost:44324/api/Calculator/>

Example of request body in JSON:

|  |
| --- |
| {  "sum": "1.1 + ( ( 2 + 3 \* ( 7 - 5 ) - 1 ) / 2 ) \* 3 + 2 + 1000"  } |

Response body:

|  |
| --- |
| {      "result": "1013.6",      "error": ""  } |

The result is processed in double data type and rounded to 5 decimal places to prevent arithmetic discrepancies.

In the event of invalid input, a validation error message will be returned. Validation is done by doing regex match on unsupported characters.

|  |
| --- |
| {      "result": null,      "error": "Validation Error: The math expressions consist of unsupported characters."  } |

# Calculator Algorithm

Assume that we have the tokenized math expression and put into a linked list, processing priority will be given to bracketed part and then multiplication/division operations. A nested bracket is evaluated based on the depth of brackets, where the deepest level will be processed first.

Calculation is done by proceeding the linked list items and evaluate based on the math operator.

**1**

(Token #1)

**+**

(Token #2)

**2**

(Token #3)

Three tokens are evaluated at a time, put the result in last token and then remove the remaining tokens.

**3**

(Token #3)

Here is an example of the algorithm steps:

Process the deepest bracket first.

|  |
| --- |
| 1.1 + ( ( 2 + 3 \* ( 7 - 5 ) - 1 ) / 2 ) \* 3 – 2 |

Give priority to multiplication.

|  |
| --- |
| 1.1 + ( ( 2 + 3 \* **2** - 1 ) / 2 ) \* 3 – 2 |

Process next level bracket.

|  |
| --- |
| 1.1 + ( ( 2 + **6** - 1 ) / 2 ) \* 3 – 2 |

Process last bracket and etc.

|  |
| --- |
| 1.1 + ( **7** / 2 ) \* 3 – 2 |

|  |
| --- |
| 1.1 + **3.5** \* 3 – 2 |

|  |
| --- |
| 1.1 + **10.5** – 2 |

|  |
| --- |
| **9.6** |

# Overall Code Flow

