math-parser.md

Math Parser

Quick Install

1. Download the latest release for your environment.
2. Extract the files.
3. Run `MathParser.exe`.
    - If you encounter permission issues, run the following command to grant execution permissions:

      ```
      chmod 700 <path to MathParser.exe>
      ```

    - After that, try running it again.

Build Yourself

1. Clone the repository or download the source code in a suitable .NET development environment.
2. Open and run `Program.cs`.

Features

- This project implements a **math expression parser** supporting the following operations:
    - Addition (`+`), Subtraction (`-`), Multiplication (`*`), Division (`/`), Parentheses (`(` , `)`), and Exponentiation (`^`).
- It uses a **recursive algorithm** pipeline, created by JJDOESIT (me).

How It Works

The parsing and evaluation process occurs in 4 main steps:

1. **Convert user input into a stack:**
   The input expression is tokenized into a stack of operations and numbers.
   - Example:
     `10 (2 + 3) + 1` becomes `[10, (, 2, +, 3, ), +, 1]`.
2. **Add implicit multiplication:**
   Multiplication is assumed where it's not explicitly written.
   - Example:
     `10 (2 + 3) + 1` becomes `10 * (2 + 3) + 1`.
3. **Add parentheses to enforce precedence:**
   Parentheses are added to force the correct order of operations.
   - Example:
     `10 * (2 + 3) + 1` becomes `(10 * (2 + 3)) + 1`.
4. **Calculate the expression:**
   The expression is evaluated recursively from the inside out.

   The algorithm processes each operation according to standard mathematical precedence rules, ensuring that operations inside parentheses are evaluated first.

Example

Here's an example of the entire process:

Input: 10 (2 + 3) + 1

Step-by-step breakdown:

1. Tokenize into stack:
   `[10, (, 2, +, 3, ), +, 1]`
2. Implicit multiplication:
   `10 * (2 + 3) + 1`

3. Add parentheses:
   `(10 * (2 + 3)) + 1`

4. Evaluate expression:

   - Inside parentheses: `(2 + 3) = 5`
   - Multiply: `10 * 5 = 50`
   - Add: `50 + 1 = 51`

Output: 51

Testing

You can easily test the math parser by using the **test case file** that comes with the project. The program allows you to either enter your own custom expression or run a set of pre-defined test cases.

Steps to Run Tests:

1. **Locate the Test File:** The test cases are stored in the `test.txt` file.
2. **Edit the Test File:** Open the `test.txt` file and add your own math expressions, one per line.
3. **Run the Program:** When you run the program (`MathParser.exe`), you will be prompted with 2 options:
   - **Enter a custom expression**: Type in a math expression and press Enter to get the result.
   - **Run the test case file**: Choose this option to automatically evaluate all expressions listed in the `test.txt` file.
4. **View the Results:** The program will print out the result for each test case or custom expression you enter.

This makes it easy to test multiple expressions in bulk and quickly verify the accuracy of the math parser.