Team 14

Arithmetic Expression Evaluator in C++

User's Manual

Version 1.0

Revision History

Date	Version	Description	Author
12/12/2024	1.0	Initial Version	Team 14

Table of Contents

Test Case

1. Purpose

This is a user manual for an Arithmetic Expression Evaluator made using C++.

2. Introduction

The Arithmetic Expression Evaluator is a tool accessed via the command line. It is designed to parse through expressions, evaluate them, and return a correct answer to the user.

Key Features:

- Handles parenthesis to determine the order of evaluation; including nested parenthesis
- Follows the PEMDAS order of operations
- Runs from the command line
- Elegant error handling
- Works with these operators: + (addition), - (subtraction), * (multiplication), / (division), % (modulo), **(exponent)

How to Install:

1: Make sure you have a tool like g++ for compiling C++ programs
2: Download the parser source code and all its dependencies
3: Compile the program utilizing the makefile within the src folder. This can be done using the 'make' command on terminal.

3. Getting started

The following are step-by-step instructions on how to use the Arithmetic Expression Evaluator:

Step 1: Launch your terminal and run the .exe file for the Evaluator

Step 2: Input an expression into the command line, for example: (10 – 3) * 20 + 3 first subtracts 10 from 3 within the parenthesis, then multiplies by 20, and finally adds 3. The Evaluator is capable of handling the operators for: addition, subtraction, multiplication, division, modulo, and exponents. Any operators entered will adhere to the rules of PEMDAS.

Step 3: After you enter in your expression, the terminal will return a single floating-point number which is the solution to the expression. For the example given above, the number returned would be 140.0.

If you give the Evaluator an invalid input, it will also return a response letting you know that there was an error and what went wrong. Inputting something like: ((8 % 3) + 5 returns the error response: "Parentheses are mismatched or missing". Clearly displaying that there is a missing parentheses in the expression.

4. Advanced features

Handles Floating-point numbers.

5. Troubleshooting

Problem 1 – Invalid Operators: Using an operator which is not supported by the Evaluator e.g. 2^10
Solution: Only the following operators are recognized by the Evaluator –

- Addition: +
- Subtraction: -
- Multiplication: *

- Division: /
- Modulo: %
- Exponent: **
- Left Parenthesis: (
- Right Parenthesis:)

  Moreover, duplicate operators are also not allowed e.g. 10 // 3. The only exceptions are ‘**’ for exponents, and ‘- -‘ can represent subtracting something negative.

Problem 2 – No Operand: Operators require operands to function. For example: inputting 2+ will result in error because the number 2 is not added to anything
Solution: Make sure that every operator in your expression has the necessary amount of operands

Problem 3 – Mismatched Parenthesis: Opening brackets without closing brackets or vice-versa.
Solution: Every set of parentheses must have an open bracket e.g. ‘(‘ and a closing bracket ‘)’. Be extra careful with nested parentheses.

Problem 4 – Division or Modulo by 0: Attempting to divide or mod a number by 0.
Solution: Numbers may not be divided or mod by 0. Ensure that any divisors do not accidentally equal 0 e.g. 40 / (4 + 9 -13) results in error

6. Examples

Example No.	Expression	Result	Explanation
1	2+8*(-2)	-14	-2 is multiplied by 8 and then 2 is added to the product
2	(35 + 3) % 2**3	6	35 +5 takes the most precedence because it is in parenthesis. The 2 is raised to the power of 3. Then the modulo of 40 and 8 is taken.
3	(((2+3)))	5	Only the operands in the innermost parentheses are added

7. Glossary of terms

PEMDAS: The mathematical order of operations**:** **P**arentheses, **E**xponents, **M**ultiplication, **D**ivision, **A**ddition, **S**ubtraction (modulo goes after division and multiplication and before addition and subtraction)

Command Line: A text-based interface where the user can input commands.

g++: A tool used to compile C++ programs

Arithmetic Expression: Some combination of operators or numbers.

8. FAQ

Can decimal numbers be used in the input?:

Yes, you may use decimal numbers