The University of Kansas

Arithmetic Expression Evaluator User Manual

Version <1.0>

Revision History

Date	Version	Description	Author
06/11/23	1.0	Initial version of User Manual	Hannah Smith

Table of Contents

Software Architecture Document

1. Introduction

1.1 Purpose

The Arithmetic Expression Evaluator User Manual has been created to provide all users, regardless of their level of experience, with comprehensive guidance on effective software utilization. It aims to equip individuals with the knowledge required for successful arithmetic expression evaluation, covering installation, troubleshooting, and advanced features. Serving as a reliable reference, the manual ensures users can confidently interact with the Arithmetic Expression Evaluator, fostering a seamless and informed user experience.

1.2 Features

The Arithmetic Expression Evaluator boasts a range of powerful features designed to streamline mathematical calculations. Users can effortlessly evaluate arithmetic expressions, ensuring precise and swift results. The software's intelligent error feedback mechanism provides users with clear guidance if an incorrect character is entered, aiding in prompt correction.

1.3 How to install and run it

To install and run the Arithmetic Expression Evaluator, start by downloading the folder named "Arithmetic Expression Evaluator" from the project GitHub. Open the terminal on your computer and navigate to the Downloads directory by typing "cd Downloads." Once there, access the specific folder for the Arithmetic Expression Evaluator using the command "cd arithmeticExpressionEvaluator." To compile the software, enter the command "make" in the terminal, initiating the compilation process. After successful compilation, you can run the program by executing the command “./Calculator”. Following these steps ensures the proper installation and functionality of the Arithmetic Expression Evaluator on your computer. If any issues arise, consult the troubleshooting section in the user manual for assistance.

2. Getting Started

To begin utilizing the Arithmetic Expression Evaluator, take the following steps:

1. Have and understand the expression to be entered. To get a correct output, the expression must be entered correctly, and this requires an understanding of the desired order of operations.
2. Run the program and carefully read the instructions.
3. When prompted, begin entering the expression into the terminal, utilizing parentheses where needed to clearly define precedence. Note that implicit multiplication is not supported. Permitted entries are as follows:
    i. All numeric symbols
    ii. Operators: +, -, *, /, ^, %, (,)
    iii. “exit” or “history”
4. Once the expression has been inputted, press the “enter” key and wait a moment for the program to process the expression.
5. The proper result will be printed to the terminal. If there is a nonnumerical result, refer to Section 4 (Troubleshooting) for what your result may mean.
6. If you enter “exit”, the program will end.

3. Advanced Features

In addition to the basic features mentioned earlier in this document, the Arithmetic Expression Evaluator has two other advanced features. First, users are not limited to integer values as the Arithmetic Expression Calculator can parse floating point numbers. To do so, users can enter an integer followed by a period (.) and the decimal value. Second, users can view a list of the previously entered expressions and their results by entering “history” instead of a new expression.

4. Troubleshooting

- If you get the error “Mismatched Parenthesis”, check to see if you have the same amount of opening parenthesis as closing parenthesis.
- If you get the error “Incorrect Operator Usage” there could be multiple causes:
    - Dividing (/) by 0
    - Modding (%) by 0
- If you get the error “Invalid character” check to see if you have only used the characters +, -, *, /, ^, %, (, and).
- If you get the error “Missing Operator” check to see you have an operator between all numbers and parentheses.
- If you get the error “Operator Without Operands” check to see if you have an operator that does not have a number before or after it.
- If your terminal doesn’t recognize the file, check to see if the folder is downloaded and if you’re in the correct folder.

5. Glossary of Terms

- Compilation - process by which a high-level programming language code is translated into machine code or an intermediate code by a compiler to make the code understandable and executable by a computer's hardware

- Expression - combination of numbers, symbols, and operators (such as +, -, *, /) that represents a mathematical computation

- Implicit multiplication - refers to multiplication that is not explicitly indicated using a multiplication symbol or an operator

- Mod - refers to the modulo operation. For example, "a mod b" is the remainder when a is divided by b

- Parse - refers to the process of analyzing a sequence of symbols or data to understand its structure and meaning

- Precedence - refers to the order in which operations are carried out in an expression

- Terminal - refers to a command-line interface or a device that allows you to interact with a computer system using text-based commands

6. Examples

When the program starts:

```
Welcome! Below you'll be able to insert a function to calculate. Here are the
 available operators:
( ) - Parenthesis
^ - Exponent
* - Multiplication
/ - Division
% - Modulus
+ - Addition
- Subtraction

Please enter your expression, type 'history' to print history, or type 'exit'
 to end the program.
Enter expression:
```

Enter the expression:

```
Welcome! Below you'll be able to insert a function to calculate. Here are the
 available operators:
( ) - Parenthesis
^ - Exponent
* - Multiplication
/ - Division
% - Modulus
+ - Addition
- Subtraction

Please enter your expression, type 'history' to print history, or type 'exit'
 to end the program.
Enter expression: 3+4
```

Results will be shown in the next line:

```
Welcome! Below you'll be able to insert a function to calculate. Here are the
 available operators:
( ) - Parenthesis
^ - Exponent
* - Multiplication
/ - Division
% - Modulus
+ - Addition
- Subtraction

Please enter your expression, type 'history' to print history, or type 'exit'
 to end the program.
Enter expression: 3+4
Result: 7.000000
```

Errors will be shown with Invalid Expressions:

```
Welcome! Below you'll be able to insert a function to calculate. Here are the
 available operators:
( ) - Parenthesis
^ - Exponent
* - Multiplication
/ - Division
% - Modulus
+ - Addition
- Subtraction

Please enter your expression, type 'history' to print history, or type 'exit'
 to end the program.
Enter expression: 3+4
Result: 7.000000
Enter expression: /8
Result: Operator Without Operands
```

View history:

```
Welcome! Below you'll be able to insert a function to calculate. Here are the
 available operators:
( ) - Parenthesis
^ - Exponent
* - Multiplication
/ - Division
% - Modulus
+ - Addition
- Subtraction

Please enter your expression, type 'history' to print history, or type 'exit'
 to end the program.
Enter expression: 3+4
Result: 7.000000
Enter expression: /8
Result: Operator Without Operands
Enter expression: 7%2
Result: 1.000000
Enter expression: 4*5
Result: 20.000000
Enter expression: history
User History:
3+4=7.000000
7%2=1.000000
4*5=20.000000
```

Exit Program:

```
Welcome! Below you'll be able to insert a function to calculate. Here are the
 available operators:
( ) - Parenthesis
^ - Exponent
* - Multiplication
/ - Division
% - Modulus
+ - Addition
- Subtraction

Please enter your expression, type 'history' to print history, or type 'exit'
 to end the program.
Enter expression: exit
t772n289@cycle2:~/Documents/EECS_348/Arithmetic-Expression-Evaluator-main/ari
thmeticExpressionEvaluator$
```