MacroHard

Boolean Expressions Evaluator

User’s Manual

Version <1.0>

Revision History

Date	Version	Description	Author
<29/04/24>	<0.0>	<details>	Atharva Patil
<01/05/24>	<1.0>	<details>	Atharva Patil

Table of Contents

User's Manual

1. Purpose

The purpose of this document is to let the user of the software know how to utilize its features. It serves as an easy-to-understand guide on what the Boolean operands and operators do, how to use the software to evaluate the expressions in which they are used, and how to interpret the output of the evaluation.

2. Introduction

The Boolean Expressions Evaluator is a software tool designed to process and evaluate Boolean expressions using logical gates such as AND, OR, NOT, NAND, and XOR. The software allows for the input of expressions with variables represented by 'T' (True) and 'F' (False), and utilizes operators to determine the output. It is implemented in C++ and can be run directly from any standard C++ IDE or compiler.

3. Getting started

To use the Boolean Expressions Evaluator:

Launch the Program: Start the software by running the compiled executable or through a C++ development environment.
Input an Expression: Enter your Boolean expression at the prompt. Use 'T' for True and 'F' for False. Operators include:

- **! for NOT**
- **& for AND**
- **| for OR**
- **@ for NAND**
- **$ for XOR**

Parentheses can be used to define the order of operations explicitly.

Evaluate: After inputting the expression, the software evaluates it and displays either "True" or "False" based on the logical calculation, or an error message if the expression is invalid.

4. Advanced features

Currently, the software focuses on fundamental Boolean evaluation without support for saving/loading expressions or defining custom variables and functions.

5. Troubleshooting

Below are common input issues that could lead to errors in the expression's evaluation and how to solve them:

- **Missing Operand: Ensure every operator has the correct number of operands.**
- **Unknown Operator: Only use the specified operators (!, &, |, @, $).**
- **Mismatched Parentheses: Check that every opening parenthesis has a corresponding closing parenthesis.**
- **Empty Expression: Ensure your expression includes at least one operand and operator.**
- **Double Operator: Avoid using two consecutive operators without an operand in between.**
- **Invalid Characters: Use uppercase 'T' and 'F' for True and False; lowercase letters are not recognized.**

6. Examples

- **Expression: (T | F) $ F**
  - **Evaluation: True**
- **Expression: ! (T & T)**
  - **Evaluation: False**
- **Expression: (F @ T) | (T @ F)**
  - **Evaluation: True**
- **Expression: (T $ T) & F**
  - **Evaluation: False**
- **Expression: ! F | ! T**
  - **Evaluation: True**
- **Expression: (((((T | F) & F) | (T & (T | F))) @ (T @ T)) $ (! (T | F)))**
  - **Evaluation: True**
- **Expression: ((F $ ((T | F) & (F @ (T | F)))) | (T $ (T & F)))**
  - **Evaluation: False**
- **Expression: (((! (T $ F)) & (T @ T)) | ((F | T) & (T $ T)))**
  - **Evaluation: False**
- **Expression: (((T @ T) $ (F @ T)) | ((!T) & (T | (!T))))**
  - **Evaluation: True**
- **Expression: ((F @ T) $ (T | (F & F))) & (T & (T @ (!T)))**
  - **Evaluation: False**

7. Glossary of terms

- **Boolean expressions: an arithmetic expression that produces a value of either True or False when evaluated.**
- **C++: the programming language in which the software is coded.**
- **Operand: a value on which an operation is done.**
- **Operator: a character to symbolize an operation being done on one or more values.**

8. FAQ

- **Q: Does the software work as intended?**
  - **A: Yes it does.**