Huang and Co.

Boolean Expression Evaluator

User's Manual

Version 1.0

[Note: The following template is provided for use with the Unified Process for EDUcation. Text enclosed in square brackets and displayed in blue italics (style=InfoBlue) is included to provide guidance to the author and should be deleted before publishing the document. A paragraph entered following this style will automatically be set to normal (style=Body Text).]

[To customize automatic fields (which display a gray background when selected), select File>Properties and replace the Title, Subject and Company fields with the appropriate information for this document. After closing the dialog, automatic fields may be updated throughout the document by selecting Edit>Select All (or Ctrl-A) and pressing F9, or simply click on the field and press F9. This must be done separately for Headers and Footers. Alt-F9 will toggle between displaying the field names and the field contents. See Word help for more information on working with fields.]

Revision History

Date	Version	Description	Author
27/04/24	1.0	<details>	Andrew Huang, Elizabeth Miller, Hunter Long, Katherine Swann, Keaton Xu

Table of Contents

User's Manual

1. Purpose

This user manual provides a step-by-step guide on how to use the Boolean expression evaluator to solve Boolean logic expressions, and lists the features it contains as well as the steps to take in case of errors.

2. Introduction

The Boolean expression evaluator is made to calculate Boolean logic expressions that are inputted. The truth values of these expressions can be defined by the user, and the evaluator has the ability to handle expressions of any complexity through its parenthesis handling and error handling.

To install the evaluator, navigate to the Github repository: https://github.com/andrewhuang2019/EECS348Project . Clone the repository to your local machine, then navigate inside the project folder that is now on your machine. Use your terminal to run main.cpp.

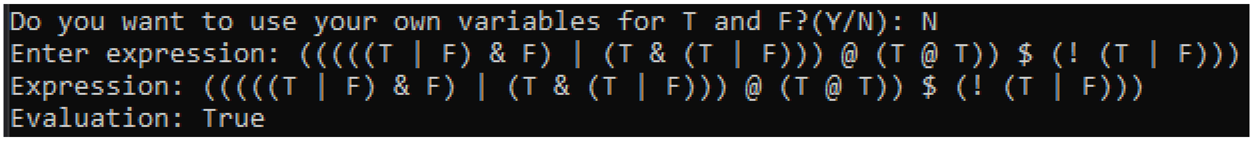
3. Getting started

This section should provide a step-by-step guide on how to use the software to evaluate arithmetic expressions. It should include instructions on how to enter expressions, how to use the various operators and functions, and how to interpret the results.

Usage of the Boolean expression evaluator begins with the opportunity to define custom variables. (For more information on this process, see "Advanced features".)

Following this option, the Boolean expression evaluator will ask the user for a Boolean expression. If the previous option was selected, this expression should be entered with the user-defined variables for true and false; if not, it should be entered with "T" for true and "F" for false. (For more information on operators and grouping with parentheses, see "Examples").

Finally, press "enter" on the keyboard, and the Boolean expression evaluator will automate the hard work of condensing this expression down to a simple "True" or "False"!



```
Do you want to use your own variables for T and F?(Y/N): N
Enter expression: (((((T | F) & F) | (T & (T | F))) @ (T @ T)) $ (! (T | F)))
Expression: (((((T | F) & F) | (T & (T | F))) @ (T @ T)) $ (! (T | F)))
Evaluation: True
```

4. Advanced features

This section should describe any advanced features of the software, such as the ability to save and load expressions, or to define custom variables and functions.

As an alternative to writing "T" for true and "F" for false, the Boolean expression evaluator allows users to instead define custom variables to represent true and false. If this option is desired, the user should select "yes" in response to the first prompt. Following this, the Boolean expression evaluator will then prompt the user to provide one custom variable each for true and false. After these are defined, the user should then provide the equation using these variables.

5. Troubleshooting

This section should provide a list of common problems, if any, that users may encounter, and how to solve them.

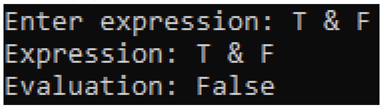
If an invalid expression is entered, an error message will be displayed and the program will abort. Common examples of invalid expressions include missing matching parentheses, unsupported operators, and incorrect variable instantiation. When an error message is encountered, this is indicative of an invalid expression being entered, so the program should be run again with a valid expression.

6. Examples

This section should provide examples of how to use the software to evaluate different types of arithmetic expressions.

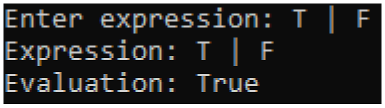
The Boolean expression evaluator allows for numerous logical operations to be used through the use of different symbols. These are:

- AND (&): Returns true if both terms are true



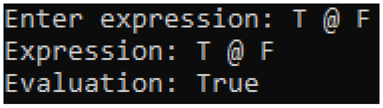
```
Enter expression: T & F
Expression: T & F
Evaluation: False
```

- OR (|): Returns true if at least one of the terms are true



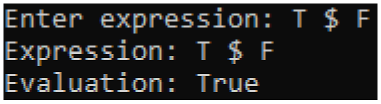
```
Enter expression: T | F
Expression: T | F
Evaluation: True
```

- NAND (@): Returns true unless both terms are true



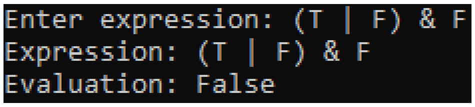
```
Enter expression: T @ F
Expression: T @ F
Evaluation: True
```

- XOR ($): Returns true if exactly one of the terms are true

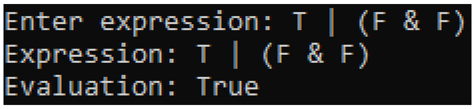


```
Enter expression: T $ F
Expression: T $ F
Evaluation: True
```

Additionally, any section of the expression can be chosen to be calculated first by grouping it within parentheses. These can be nested to give priority as the expression is gradually condensed down, in the fashion of the typical order of operations.

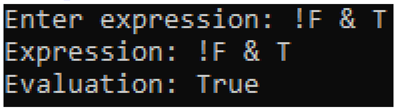


```
Enter expression: (T | F) & F
Expression: (T | F) & F
Evaluation: False
```

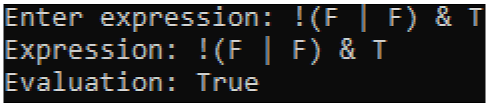


```
Enter expression: T | (F & F)
Expression: T | (F & F)
Evaluation: True
```

Finally, NOT can be used by putting an exclamation mark (!) before a variable or a parentheses-grouped section of the expression.



```
Enter expression: !F & T
Expression: !F & T
Evaluation: True
```



```
Enter expression: !(F | F) & T
Expression: !(F | F) & T
Evaluation: True
```

7. Glossary of terms

This section should define any technical terms that are used in the manual.

- **Boolean:** a data type that holds a value of either True or False.
- **Boolean expression:** an expression consisting of Booleans and operations performed on them, which can be condensed to a single True or False value.

8. FAQ

This section should answer frequently asked questions about the software.

Q. What operating systems does the Boolean expression evaluator support?

A. The Boolean expression evaluator is supported across all operating systems.

Q. Can the Boolean expression evaluator evaluate arithmetic expressions?

A. No.

Q. Does the Boolean expression evaluator cost money to install or operate?

A. No! The Boolean expression evaluator is free and open-source.