Cryptic Coders

Boolean Logic Simulator

User’s Manual

Version 1.0

Revision History

Date	Version	Description	Author
04/24/2024	1.00	Started working on the document. Completed the getting started portion and started working on advanced features, as well as the purpose and introduction>	Mark Maloney
04/29/2024	1.01	Started the glossary and the FAQ	Mark Maloney
05/02/2024	1.02	Added the troubleshooting and the examples to the document	Mark Maloney Aiden Burke

Table of Contents

Test Case

1. Purpose

The purpose of this document is to give the user instructions and guidance on how to use the Boolean Logic Simulator. This document goes through the functionality of the simulator to give the user a better experience.

2. Introduction

This software provides the user with a web interface which consists of a webpage which the user can access to use the Boolean calculator. The Boolean Calculator included operations like AND, OR, NOT, NAND and XOR. The user may use T and F for True and False and the calculator will evaluate any expressions using any combination of the listed operators. The software can be downloaded and used on any platform.

Calculator Symbol	Description
T	Represents Boolean value “True”
F	Represents Boolean value “False”
&	Represents “AND” operator
|	Represents “OR” operator
!	Represents “NOT” operator
@	Represents “NAND” operator
$	Represents “XOR” operator
(	Represents open parenthesis
)	Represents close parenthesis

Using this notation, an example input could look like: (T & (F | T))

3. Getting started

Once you navigate to the user interface after downloading, the calculator will display a screen with instructions on which operators are valid and instructs you to use “T” and “F” for true and false, respectively. The interface remind you that the input will be case sensitive as well and parenthesis sensitive.

1) To open the program, you can either use the public replit link, https://replit.com/@aidenburke6/EECS348-Final-Project?v=1, Run, and then send node server.js in the shell terminal to host a server for the interface through replit; or you can run the program from your own computer by navigating to the file directory and sending node server.js in terminal, and opening a page to localhost:3000/
2) If you do not want to host a webserver to run the interface, you can also interact with main directly through the terminal and run main and it will prompt you for an input expression.
3) If you do choose to interact with the web interface, the first step is to enter the Boolean expression in the textbox labeled “Enter a Boolean expression”. There is an example of how to format of the expression as the placeholder in the Textbox.
4) The second step is to double check the expression is entered correctly and press the button labeled “Evaluate”.
5) The result will appear below the box. If there is an error, the text "Error” will appear with an error message.
6) For convenience when entering expressions, there are also buttons with each valid operator or operand as well as buttons for clearing the input box or undoing the most recent input.

4. Advanced features

An advanced feature of this Program is the Web interface. Using the web interface versus the terminal provides the user with a better experience by providing the user with a clean display of the calculator. It also allows the error handling to be displayed more clearly as well as the result, which makes the calculator more practical.

Another advanced feature is that the User Interface provides the user with buttons that upon pressing will add the operator or operand that was pressed to the text box which can then evaluated. There is also a “Clear” button which allows the user to clear the expression that is currently typed and an “Undo” button which removes the most recent operator or operand from the textbox.

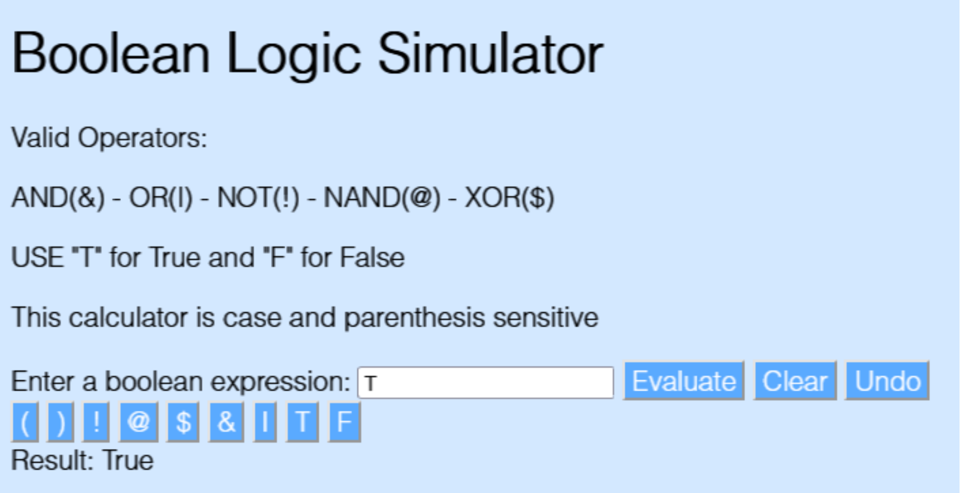
The final advanced feature is some CSS style to give the user the best possible experience. We added a blue theme to the calculator to make it more appealing.

5. Troubleshooting

1) Order of Operations: This calculator evaluates Parenthesis first, followed by the not command. The commands AND, NAND and OR are then next, and will be evaluated from left to right if they are in the same expression. The last operator is XOR. If there is any confusion in the order of the operators, we recommend the use of parenthesis.
2) Parenthesis: Many Errors can occur from having unmatched parenthesis. Please make sure for each open parenthesis, there is exactly ONE close parenthesis.

6. Examples

Example 1:



Boolean Logic Simulator

Valid Operators:

AND(&) - OR(|) - NOT(!) - NAND(@) - XOR($)

USE "T" for True and "F" for False

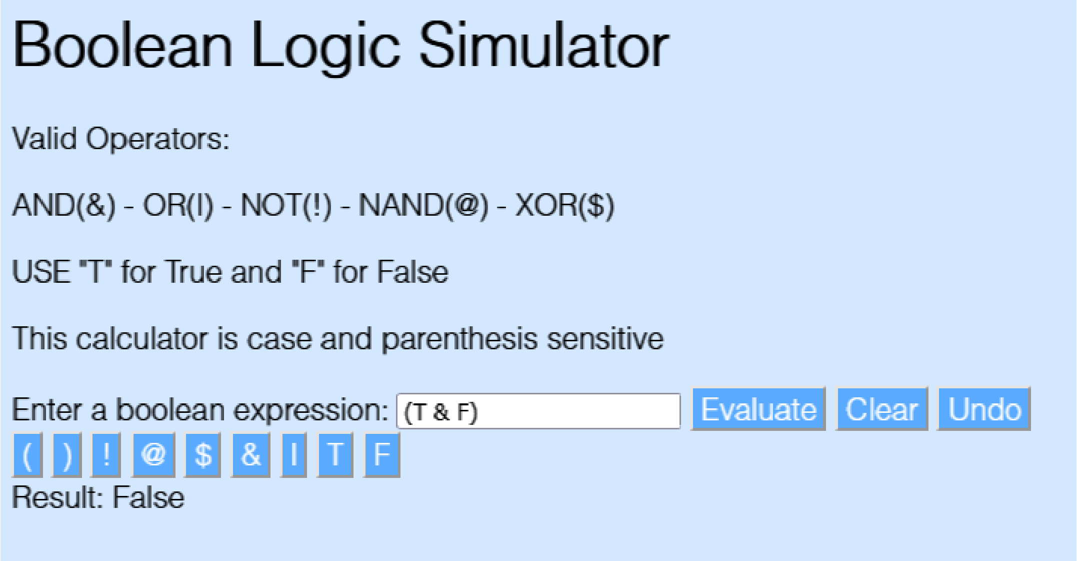
This calculator is case and parenthesis sensitive

Enter a boolean expression: T

Result: True

Expression: T
Output: True
This is an example of the program working without any operators.

Example 2:



Boolean Logic Simulator

Valid Operators:

AND(&) - OR(I) - NOT(!) - NAND(@) - XOR($)

USE "T" for True and "F" for False

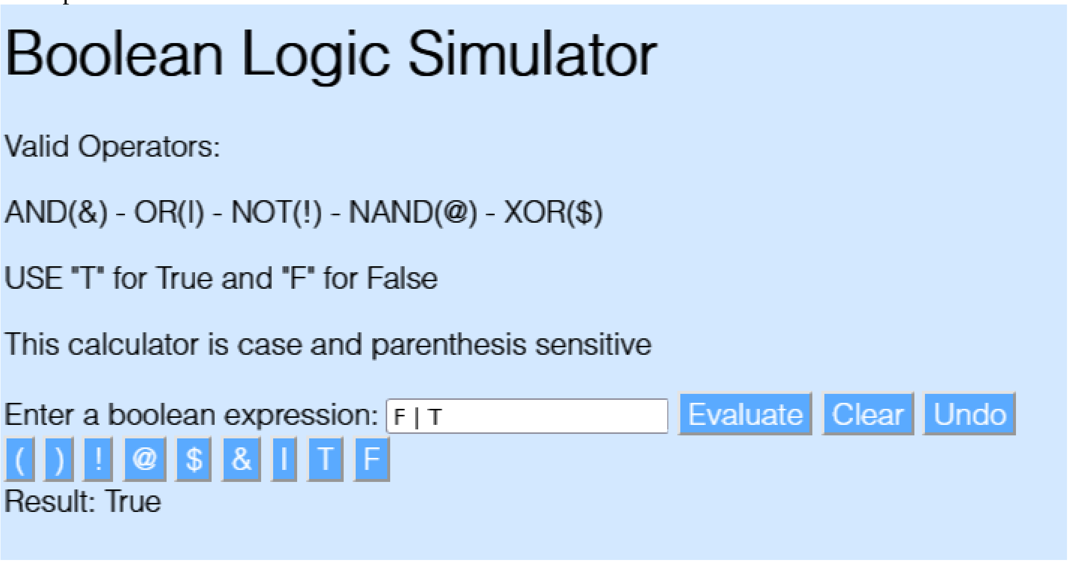
This calculator is case and parenthesis sensitive

Enter a boolean expression: (T & F)

Result: False

Expression: (T & F)
Output: False
This is an example of the program working using the AND operator.

Example 3:



Boolean Logic Simulator

Valid Operators:

AND(&) - OR(I) - NOT(!) - NAND(@) - XOR($)

USE "T" for True and "F" for False

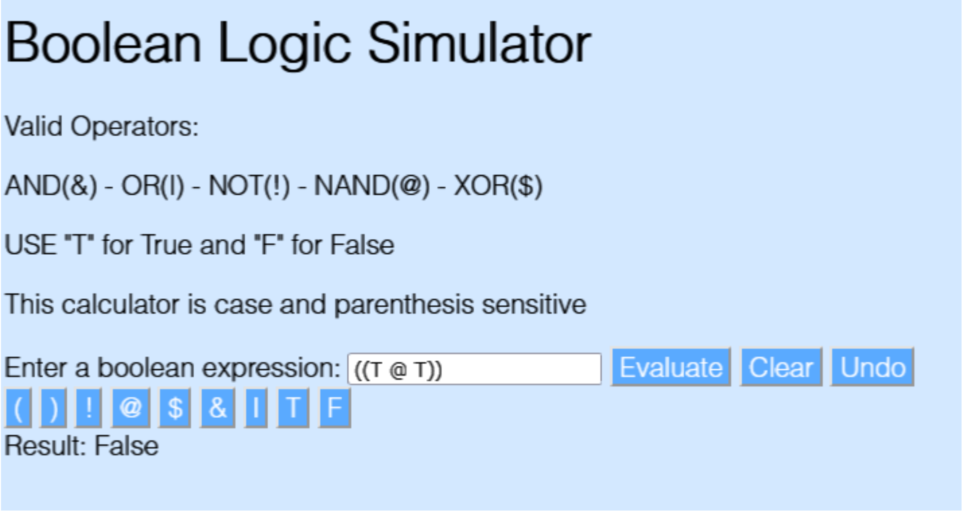
This calculator is case and parenthesis sensitive

Enter a boolean expression: F | T

Result: True

Expression: F | T
Output: True
This is an example of the program working using the OR operator.

Example 4:



Boolean Logic Simulator

Valid Operators:

AND(&) - OR(I) - NOT(!) - NAND(@) - XOR($)

USE "T" for True and "F" for False

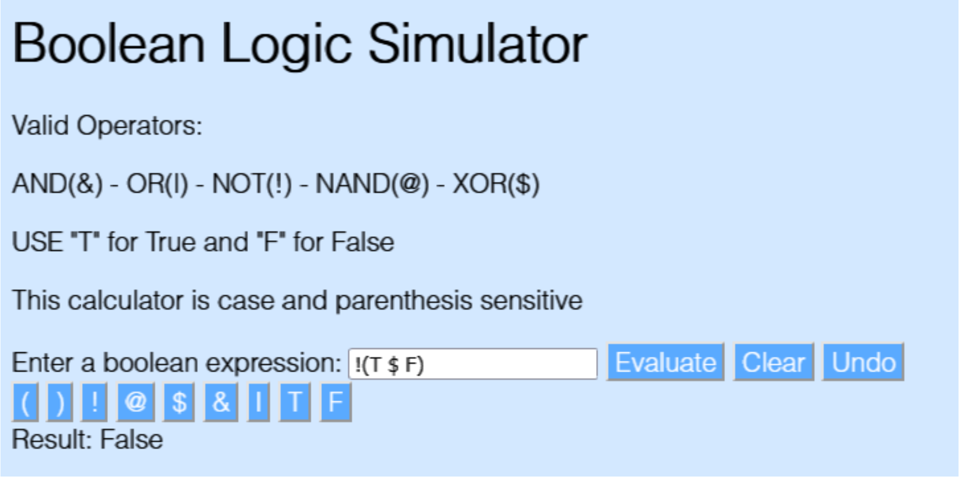
This calculator is case and parenthesis sensitive

Enter a boolean expression: ((T @ T))

Result: False

Expression: ((T @ F))
Output: False
This is an example of the program working using the NAND operator.

Example 5:



Boolean Logic Simulator

Valid Operators:

AND(&) - OR(I) - NOT(!) - NAND(@) - XOR($)

USE "T" for True and "F" for False

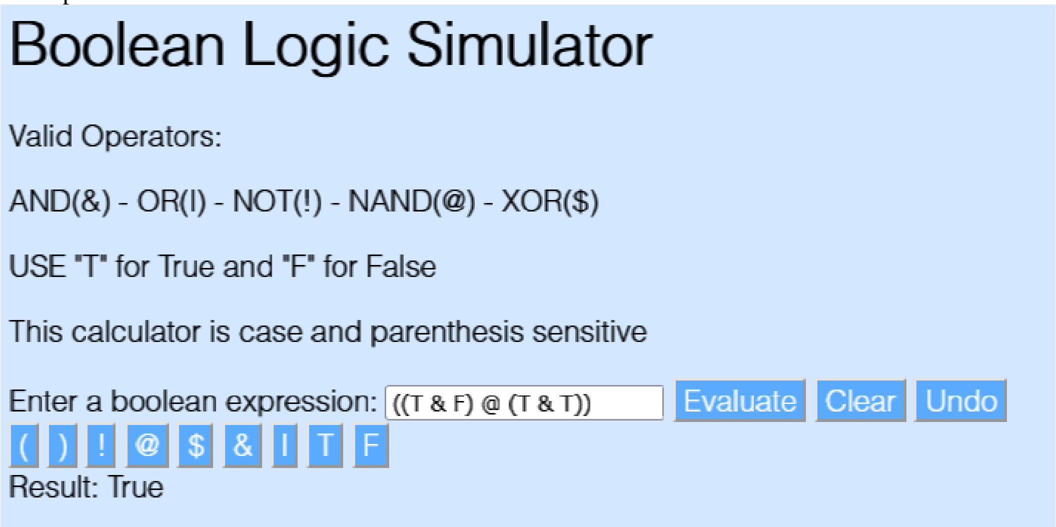
This calculator is case and parenthesis sensitive

Enter a boolean expression: !(T $ F)

Result: False

Expression: !(T $ F)
Output: False
This is an example of the program working using the XOR and NOT operator, as well as parenthesis.

Example 6:



Boolean Logic Simulator

Valid Operators:

AND(&) - OR(I) - NOT(!) - NAND(@) - XOR($)

USE "T" for True and "F" for False

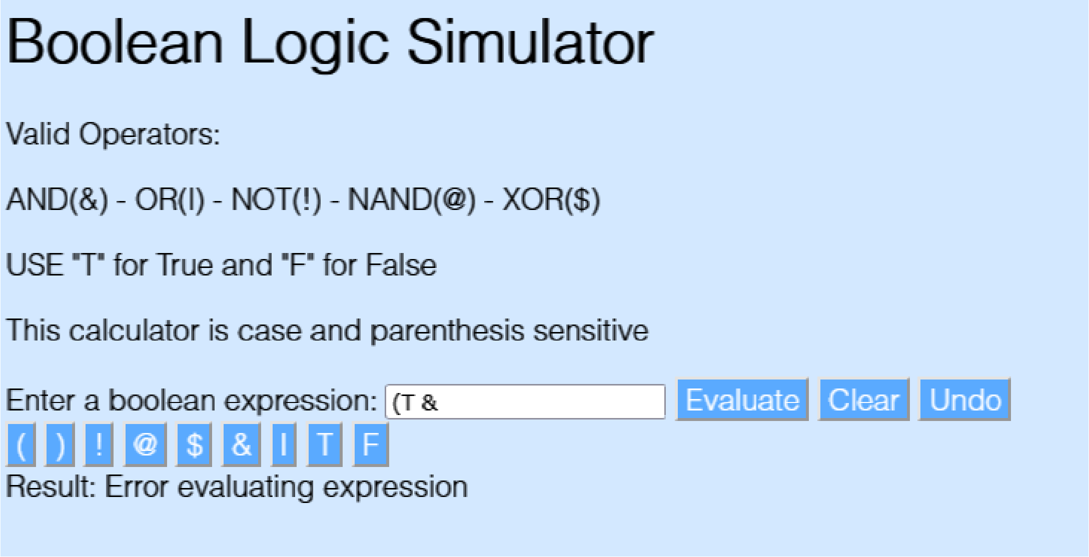
This calculator is case and parenthesis sensitive

Enter a boolean expression: ((T & F) @ (T & T))

Result: True

Expression: T
Output: True
This is an example of the program working with a complex expression.

Example 7:



Boolean Logic Simulator

Valid Operators:

AND(&) - OR(I) - NOT(!) - NAND(@) - XOR($)

USE "T" for True and "F" for False

This calculator is case and parenthesis sensitive

Enter a boolean expression: (T &

Result: Error evaluating expression

Expression: T
Output: Error
This is an example of the program's error handling working properly.

7. Glossary of terms

FAQ – Frequently Asked Questions

8. FAQ

This section addresses frequently asked questions.

Q: Does this calculator only solve Boolean expressions?

A: Yes, this calculator only solves Boolean expressions, it will not work for arithmetic expressions.

Q: Can I use other Symbols instead of the ones in the User Manual?

A: No, only symbols mentioned in the User Manual will work.

Q: How are truth values represented in the Boolean Expression Evaluator?

A: Truth values are represented as characters either ‘T’ for True or ‘F’ for False.

Q: Will downloading this software harm my machine?

A: No, the software is perfectly safe.