Huang and Co.

Boolean Expression Evaluator
Software Requirements Specifications

Version 1.0

Revision History

Date	Version	Description	Author
02/03/2024	<1.0>	split tasks	Andrew, Keaton, Hunter, Elizabeth, Katharine
23/03/2024	<1.0>	review entirety of document	Andrew, Keaton, Hunter, Elizabeth, Katharine

Table of Contents

Software Requirements Specifications

1. Introduction

This Software Requirements Specification (SRS) document specifies the purpose, the scope, and all the functionalities required to implement a Boolean Expression Evaluator (B.E.E.). The calculator will be designed to interact with the user through a command-line interface.

1.1 Purpose

The purpose of this SRS document is to specify the functional and non-functional requirements for a B.E.E. and the constraints on its implementation. The requirements will be written in an unambiguous, complete, and consistent manner.

1.2 Scope

The B.E.E. is intended to be a single program written in C++ and compiled using a GNU Compiler Collection (GCC) compiler. The program will allow the user to input Boolean logic expressions and will output corresponding values, as well as take care of error handling if the inputs are outside the scope of Boolean logic. The user's input and program's output will be displayed through the command-line interface.

1.3 Definitions, Acronyms, and Abbreviations

SRS - Software Requirements Specification

GCC - GNU Compiler Collection

1.4 References

1. EECS 348 Project Description, Professor Hossein Saiedian, Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence, KS, Spring Semester 2024.

1.5 Overview

The rest of this document provides a high-level description of the product and specifies all its functional and non-functional requirements. Section 2 identifies the interfaces the product interacts with, its functionalities, and its constraints, all of which are designed to satisfy the interests of the stakeholder. Section 3 describes in detail the functionality of the product, and Section 4 classifies these functionalities in order of importance. There are no appendices in this SRS.

2. Overall Description

2.1 Product perspective

2.1.1 System Interfaces

N/A

2.1.2 User Interfaces

There is not a primary user interface. Command Line will be utilized, and if certain packages are necessary they will be described. Whether Windows or Linux must be used will be detailed to a teaching assistant.

2.1.3 Hardware Interfaces

N/A

2.1.4 Software Interfaces

The software requires a GCC compiler to compile the program consisting of C++ code.

2.1.5 Communication Interfaces

N/A

2.1.6 Memory Constraints

The source code has no memory constraints assigned to it by the project description. The operating system used may impose external additional memory constraints during the program's runtime.

2.1.7 Operations

N/A

2.2 Product functions

Logical functions implemented:

AND, OR, NOT, NAND, XOR

Product features:

- Expression Parsing and Parenthesis Handling
- True and false input for variables
- Evaluates functions and outputs correctly

2.3 User characteristics

Professor Saiedian and Liangqiu Ren (Lab TA). The requirements set by the stakeholders are that each deliverable (e.g. Project Plan, SRS) be submitted on time, the product has functionality as specified by requirements, the product works as intended (no bugs), and covers all cases of user error.

2.4 Constraints

Tools:

C++, GitHub

Other:

Able to run on most Linux based or Windows based machines.

Product and deliverables must be delivered in a timely manner.

2.5 Assumptions and dependencies

Requires a machine that can run C/C++ programs.

2.6 Requirements subsets

N/A

3. Specific Requirements

3.1 Functionality

The main purpose of this project is to create a C++ program that takes in a Boolean expression as input from a user and evaluates it as being true or false.

3.1.1 Operator Support

Parse and evaluate expressions containing the following operators:

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

3.1.2 Expression Parsing

Evaluate expressions by implementing the correct order of operations.

3.1.3 Variables

Allow users to specify variables for the equation and define them as true or false using T or F respectively.

3.1.4 Expression Evaluation

Return the total value of the expression as true or false.

3.1.5 Error Handling

Handle errors arising from improperly written equations, such as undefined variables or mismatched/misplaced parentheses.

3.1.6 Parentheses Support

Parse and evaluate expressions containing parentheses that group portions of the expression in order to determine precedence.

3.2 Use-Case Specifications

N/A

3.3 Supplementary Requirements

Clear and readable terminal output for user convenience.

4. Classification of Functional Requirements

Functionality	Type
Operator Support (AND, OR, NOT, NAND, XOR)	Essential
Expression Parsing	Essential
Truth Value Input	Essential
Evaluation and Output	Essential
Error Handling	Essential
Parentheses Handling	Essential
Bracket/Braces Handling	Desirable
Named Operators	Desirable
Variable Initialization	Optional
Truth Table Generation	Optional

5. Appendices

N/A