Team8 Co

Calc.Co
Software Requirements Specifications

Version <1.0>

Revision History

Date	Version	Description	Author
<dd/mmm/yy>	<x.x>	<details>	<name>

Table of Contents

Software Requirements Specifications

1. Introduction

This Software Requirements Specification (SRS) document presents a comprehensive overview of the Calc.Co project. It aims to clearly define the project's purpose, key functionalities, user interactions, and system constraints. By outlining these elements, this document provides a solid foundation for understanding the overall design and requirements of Calc.Co.

1.1 Purpose

The purpose of the SRS document is to define the functional and non-functional requirements of the Calc.Co program. This document will comprehensively describe how the software behaves, including design constraints and external interfaces with which it will react. The SRS will ensure the project is built in accordance with the outlined specifications and expectations.

1.2 Scope

Calc.Co is software designed to perform advanced mathematical calculations (+, -, *, /, %, ^). We want to support user inputs for calculations and display results in an intuitive user interface. This document covers all the features of our software, which are the user interface, system constraints, and operational requirements.

1.3 Definitions, Acronyms, and Abbreviations

SRS: Software Requirements Specification
Calc.Co: The project name for the calculator software application
UI: User Interface
Use Case: A description of how a user interacts with the system to accomplish a goal
Functional Requirement: A specific behavior or function that the system must perform

1.4 References

Iteration Plans
Date: 26/09/2024
Source: Iteration Plan

Vision Document
Date: 26/09/2024
Source: Calc.Co Vision Plan

Glossary
Date: 09/26/2024
Source: Glossary

GitHub
Date: 09/19/2024
Source: https://github.com/BigIronDestroyer/Calc.Co

1.5 Overview

This SRS document contains a detailed description of the requirements of our software. It includes the following sections:

- ***Description****: Provides the context for the software and its operational environment.*
- ***Specific Requirements****: Lists the functional and non-functional requirements for the software.*
- ***Use-Case Specifications****: Describes key interactions between users and the system.*
- ***Supplementary Requirements****: Captures additional constraints and non-functional requirements not covered by use cases.*

2. Overall Description

2.1 Product perspective

The program parses and evaluates arithmetic expressions that a user enters via a command-line interpreter. Our program enforces operator precedence and handles complex expressions with parentheses. If time on this project permits, we will create a user interface to accompany the project.

2.1.1 System Interfaces

2.1.2 User Interfaces

Our calculator will have a command-line user interface. The user will enter an expression and our program will evaluate the equation and then give a result.

2.1.3 Hardware Interfaces

2.1.4 Software Interfaces

The software uses C++ standard libraries for arithmetic operations.

2.1.5 Communication Interfaces

2.1.6 Memory Constraints

Calc.Co will require minimal system memory to evaluate equations

2.1.7 Operations

2.2 Product functions

The calculator will support various arithmetic operations including addition, subtraction, multiplication, division, modulo, and exponentiation. It will parse arithmetic expressions while handling numeric constraints and respecting operator precedence. Additionally, it will be capable of evaluating expressions with parentheses to ensure correct order of operations. These features will collectively allow for the accurate computation of complex arithmetic expressions in a single calculation.

2.3 User characteristics

The target users for this are students and programmers familiar with C++ basic operations, as no app will directly connect the user to the program. This person also needs a calculator and is willing to input their commands through the command line.

2.4 Constraints

The calculator has to be developed in C++. The program should comply with the PEMDAS rule for order of operations.

2.5 Assumptions and dependencies

Assumes users have a basic understanding of arithmetic expressions and how to input them with a command line.

Assumes that the environment has a working C++ compiler.

2.6 Requirements subsets

3. Specific Requirements

3.1 Functionality

This section describes our functional requirements

3.1.1 Expression Parsing

The program will tokenize the input expression and parse it into an abstract syntax tree (AST) for evaluation. Operator precedence will follow the PEMDAS rule, with parentheses determining the grouping.

3.1.2 Supported Operators

- **Addition (+)**: Adds two numbers.
- **Subtraction (-)**: Subtracts the second number from the first.
- **Multiplication (*)**: Multiplies two numbers.
- **Division (/)**: Divides the first number by the second (except division by zero).
- **Modulo (%)**: Returns the remainder after division.
- **Exponentiation ()**: Raises the first number to the power of the second.

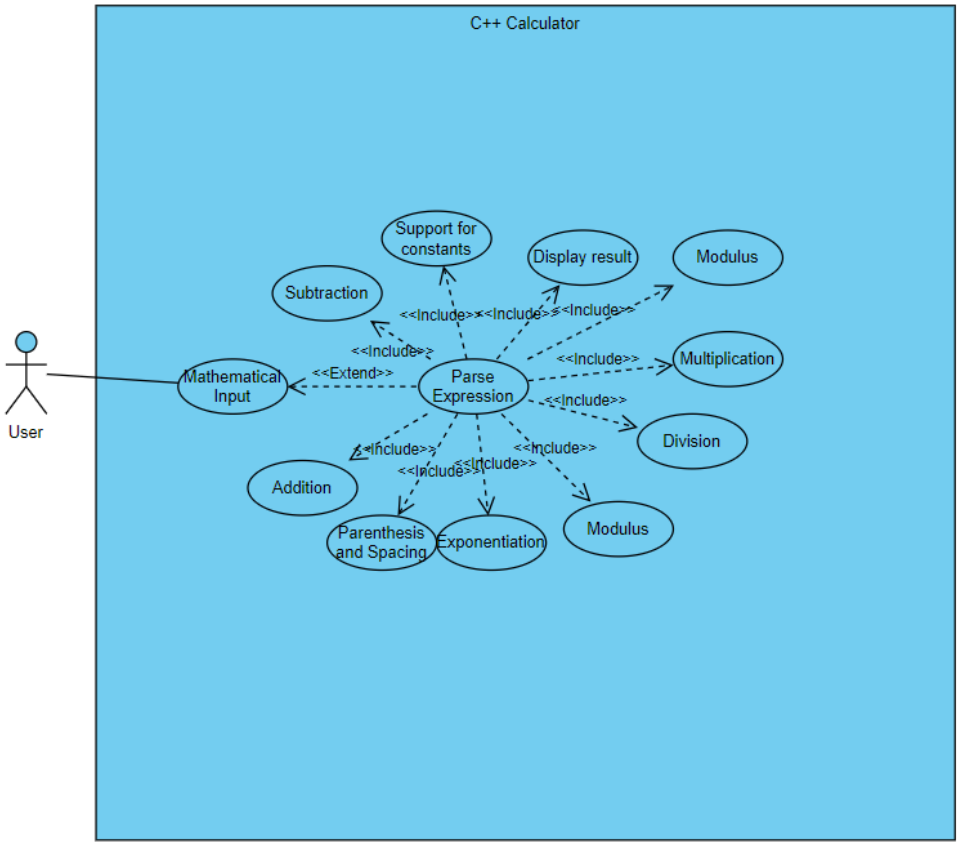
3.1.3 Parentheses

The program must handle parentheses, it has to ensure correct grouping and precedence of operations.

3.1.4 Error Handling

- Division by zero should prompt a message
- Invalid inputs (unsupported operators) should also prompt a message

3.2 Use-Case Specifications



3.3 Supplementary Requirements

The software must be easy to add future features (e.g., floating-point numbers).

The command-line interface must be intuitive for users familiar with basic terminal operations.

4. Classification of Functional Requirements

Functionality	Type
Expression Parsing	Essential
Operator Support	Essential
Parentheses Handling	Essential
Error Handling	Essential

Floating Points	Optional

5. Appendices

*[When appendices are included, the **SRS** should explicitly state whether or not the appendices are to be considered part of the requirements]*