KI-69 Calculator

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 in Microsoft Word (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** |
| <dd/mmm/yy> | <x.x> | <details> | <name> |
| 13/10/23 | 1.0 | Filled out section 1, 1.1, and 1.2 | Ginny Ke |
| 14/10/23 | 1.1 | Filled out section 1.3, 1.4, 1.5 and section 3 | Hayden Roy |
| 15/10/23 | 1.2 | Filled out sections 4 and 5 | Harrison Wendt |
| 15/10/23 | 1.3 | Filled out sections 2 and 2.1 | Nikka Vuong |
| 15/10/23 | 1.4 | Filled out sections 2.2, 2.3, 2.4, 2.5, and 2.6 | Dylan Sailors |

Table of Contents

1. Introduction 4

1.1 Purpose 4

1.2 Scope 4

1.3 Definitions, Acronyms, and Abbreviations 4

1.4 References 4

1.5 Overview 4

2. Overall Description 4

2.1 Product perspective 5

2.1.1 System Interfaces 5

2.1.2 User Interfaces 5

2.1.3 Hardware Interfaces 5

2.1.4 Software Interfaces 5

2.1.5 Communication Interfaces 5

2.1.6 Memory Constraints 5

2.1.7 Operations 5

2.2 Product functions 5

2.3 User characteristics 5

2.4 Constraints 5

2.5 Assumptions and dependencies 5

2.6 Requirements subsets 5

3. Specific Requirements 5

3.1 Functionality 6

3.1.1 <Functional Requirement One> 6

3.2 Use-Case Specifications 7

3.3 Supplementary Requirements 7

4. Classification of Functional Requirements 7

5. Appendices 8

# Introduction

The software requirements specification aims to gather all the information necessary to create a program that evaluates arithmetic expressions. The purpose, scope, definitions, references, and overview are all detailed below. This document aims to specify requirements for the new KI-69 calculator software for Kansas Instruments. The document’s format is based on IEEE standard 830-1998 and reflects on those requirements as accurately as possible.

## Purpose

The basic requirements detailed by this document, include:

* Functionality: The tasks the software is required to perform
* External Interfaces: How the system interacts with the user.
* Performance: Speed and output time.
* Attributes: Accessibility, correctness, and adaptability,
* Design Constraints: Required standards, implementation language, operating language, etc.

The requirements in this document define the specific functionality that must be met in order to successfully establish a new arithmetic expression evaluator software for Kansas Instruments.

The intended audience for this SRS includes technical personnel involved in the design, coding

and testing of the new software as well as the stakeholders involved in Kansas Instruments.

## Scope

The software to be created is an updated arithmetic expression evaluator that operates accurately and efficiently. For the purposes of the SRS, the new system will be called KI-69. The stakeholders want an user-friendly software that is easily navigated and correct. The objective of this SRS is to create an easily maintainable, adaptable, calculation time-reduced, and mathematically correct software that implements various arithmetic expressions.

## Definitions, Acronyms, and Abbreviations

* + User: individual that uses the calculator (actor in use case)
  + Operation: user chooses an operator to be performed

## References

Refer to the appendix.

## Overview

In the following sections, the requirements for the system are described. The specific requirements are detailed in Section 3, and functional requirements classified by type can be found in Section 4. The specific requirements will describe functional requirements, use-case specifications, and the non-functional requirements.

# Overall Description

Section 2 will cover details about the specific details of the project, such as product perspective, interfaces, operations, function, characteristics, constraints, assumptions, dependencies, and requirements need for the product.

[This section of the **SRS** describes the general factors that affect the product and its requirements. This section does not state specific requirements. Instead, it provides a background for those requirements, which are defined in detail in Section 3, and makes them easier to understand. Include such items as:

## Product perspective

The Arithmetic Expression Evaluator is meant to evaluate arithmetic expressions containing operators +, -, \*, /, %, and ^ as well as numeric constants.

### System Interfaces

The System Interfaces that will be used for the arithmetic expression evaluator is the programming language C++ as well as the C++ compiler. The program will be used on any system that supports the

compiler.

### User Interfaces

The User Interfaces that will be used include the operators +, -, \*, /, %, and ^ as well as numeric constants. The program has a user-friendly command-line interface that allows users to enter expressions and displays the calculated results.

### Hardware Interfaces

No Hardware Interfaces required.

### Software Interfaces

The Software Interface that will be used for is the programming language C++ and the C++ compiler.

### Communication Interfaces

No Communication Interface Required.

### Memory Constraints

The program will be used on any system, so memory constraints will depend on the system used.

### Operations

Software operations will include production software applications, monitoring system performance, making defect repairs, and testing the application after any changes are made.

## Product functions

The product will take an expression given by the user and calculate an answer based off of its input/output interface, parenthesis handling, exponent handling, multiplication and division handling, addition and subtraction handling, and error handling.

## User characteristics

The user should have general knowledge of math and how to form expressions. Education level for someone using this product varies per user.

## Constraints

Hardware constraints depend on the system being used by the user. No other constraints.

## Assumptions and dependencies

We assume the program will be run on a modern Windows, Mac, or Linux system. The calculator assumes and depends on the user to input a valid expression with no errors. If error is thrown, it will be handled.

## Requirements subsets

1. Accuracy
2. Performance
3. Input/Output Interface
4. Parentheses Handling
5. Multiplication/Division Handling
6. Addition/Subtraction Handling
7. Error Handling

# Specific Requirements

The requirements for this project includes:

1. Accuracy

* The software must generate the correct answer to the expression given by the user.
* The software must calculate the expression in the correct order of operations.

1. Input/Output Interface

* The software should have a clear and clean input field to prompt the user for an expression.
* The software should output a clear and clean answer to the expression given by the user.

1. Parentheses Handling

* The software must recognize parentheses and prioritize operations within them.

1. Error Handling

* The software must be able to recognize and handle errors, such as division by zero, syntax errors, and invalid expressions.

## Functionality

The functionality of this projects includes:

|  |  |
| --- | --- |
| Function # | Function: |
| 1 | Input/Output Interface |
| 2 | Parentheses Handling |
| 3 | Exponent Handling |
| 4 | Multiplication and Division Handling |
| 5 | Addition and Subtraction Handling |
| 6 | Error Handling |

### Input/Output Interface

The program must be able to take input from the user to get the expression to calculate. This input should be prompted for neatly and clearly. The output of the program must also be displayed neatly and clearly for the user.

### Parentheses Handling

The program must recognize parentheses and nested parentheses and evaluate expressions inside the parentheses first.

### Exponent Handling

The program must be able to compute exponents within given expressions and compute the exponent in the correct order of operations.

### Multiplication and Division Handling

The program must be able to compute multiplication and division within given expressions and compute the exponent in the correct order of operations.

### Addition and Subtraction Handling

The program must be able to compute addition and subtraction within given expressions and compute the exponent in the correct order of operations.

### Error Handling

The program must be able to recognize invalid inputs such as syntax errors like uneven number of open and closed parentheses, division by zero, and non-mathematical input.

## Use-Case Specifications

|  |  |
| --- | --- |
| Use Case: Evaluate Expression | |
| Overview | The main purpose of this use case is to evaluate a mathematical expression given by the user. The program prompts an expression and prints the result of the expression |
| Primary Actor | User |
| Secondary Actor | None |
| Starting Point | The use case starts when the program is run |
| Ending Point | The expression is either evaluated or an error is displayed due to invalid input |
| Measurable Results | The program prints the evaluation of the expression |
| Flow of Events | The use case begins with the program being run. The program will prompt for the primary actor to input a mathematical expression. The program will check for errors and if there are none the program will parse through the expression computing in the order of operations. The program will then output the evaluation of the expression. |
| Alternative Flow of Events | The primary actor gives an invalid expression which is recognized by the program and an error message is printed. |

## Supplementary Requirements

The supplementary requirements consist of:

1. Accuracy

* The program must accurately evaluate the expression.

1. Performance

* The program must perform the evaluation of the expression within a reasonable time.

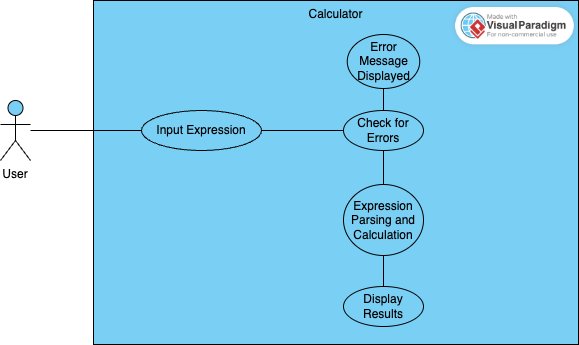
# Classification of Functional Requirements

|  |  |
| --- | --- |
| **Functionality** | **Type** |
| Input/Output Interface | Essential |
| Parentheses Handling | Essential |
| Exponent Handling | Essential |
| Multiplication and Division Handling | Essential |
| Addition and Subtraction Handling | Essential |
| Error Handling | Essential |

# Appendices

The appendices are to be considered as part of the requirements.

*5.1 Use Case Diagram*



*5.2 References*

* [EECS348: Term Project in C++](https://canvas.ku.edu/courses/104607/files/folder/Project?preview=7471579) - Professor Saiedian