<Group 32>

C++ Arithmetic Calculator
Software Architecture Document

Version <1.2>

Revision History

Date	Version	Description	Author
<06/11/23>	<1.0>	Completion of sections 2, 3, 4, 5, and 6	Derek N., Manu R., Ryan G., Priyatam N.
<11/11/23>	<1.1>	Completion of section 1	Derek N.
<11/12/23>	<1.2>	Final quality check	Derek N.

Table of Contents

Software Architecture Document

1. Introduction

1.1 Purpose

This document is an overview of the planned architectural design to be implemented for the C++ Arithmetic Calculator. Below is a description of the Architectural Representation, Architectural Goals and Constraints, Logical View, Interface Description and Quality.This document will serve as a compass for the design of the software.

1.2 Scope

This document applies to the architectural design of the C++ Arithmetic Calculator which affects the program's general design in terms of class and function design, communication with the user through the command-line, performance of the program, portability of the program, and distribution of the program.

1.3 Definitions, Acronyms, and Abbreviations

C++ - Coding language used in the design of the software
PEMDAS - Order of operations (Parentheses, Exponentiation, Multiplication, Division, Addition, Subtraction

1.4 References

Project Description (09/07/23, Professor Hossein Saiedian, University of Kansas)
Project Plan (09/15/23, Professor Hossein Saiedian, University of Kansas)

1.5 Overview

The rest of the Software Architecture Document contains a guide to the structure of this project's software. It contains sections for the Architectural Representation, Architectural Goals and Constraints, the Logical View, the Interface Description, and Quality. The Architectural Representation section describes what software architecture is for the current system, and how it is represented. The Architectural Goals and Constraints section describes the software requirements and objectives that have some significant impact on the architecture. The Logical View section describes the architecturally significant parts of the design model, such as its decomposition into subsystems and packages. The Interface Description section provides a description of the major entity interfaces. And finally the Quality section provides a description of how the software architecture contributes to all capabilities (other than functionality) of the system.

2. Architectural Representation

The general architectural strategy that we will utilize to develop the C++ Arithmetic Calculator, is parsing through a received input string from the user with the entire arithmetic expression. The program will evaluate each character of the string and determine its arithmetic functionality. Finally the program will return the answer to the arithmetic expression back to the user through the command-line.

3. Architectural Goals and Constraints

The C++ Arithmetic Calculator will be developed with architecture that allows for distribution and portability. Meaning, the program will not be too large, and able to function on different computing platforms. We will utilize an architectural design that optimizes first and foremost functionality of the program, and second, performance of the program. Since no user information is necessary for the functionality of the program, privacy and security will not be considered in the development of the software. We will be constrained by the language we use (C++), mathematical principles (PEMDAS), and the functionalities that we need to implement into the program.

4. Logical View

4.1 Overview

For this design model, there are multiple layers to consider. At the top layer, there will be a user interface package, handling user interaction. Another layer will include expression parsing as a package. The next layer will be the main calculation package. Within this package will be the subpackages for each numerical operation. The last layer will have the error handling package.

4.2 Architecturally Significant Design Modules or Packages

User Interface Package:

Handles user interaction, including input parsing and the display of the result.

Expression Parsing Package:

Tokenizes the input expression, and converts it into a structured format able to be used for the computation of the expression.

Main Calculation Package:

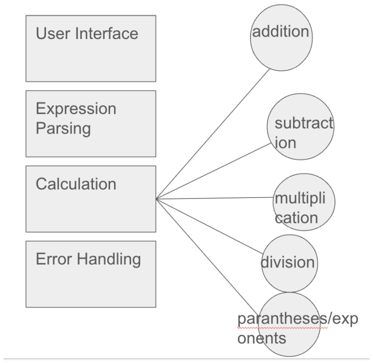
Contains the actual logic for computing the expressions given, according to the PEMDAS rules

Within this package will be the subpackages for each numerical operation, addition, subtraction, multiplication, division, exponents, as well as parentheses

The subpackages will be named **Addition, Subtraction, Multiplication, Division, Exponents, Parentheses**

Error Handling Package:

Accounts for validating inputs by the user, and informs them in case of invalid expressions or computational errors



5. Interface Description

The only interface the user will be interacting with is the command line interface, users will input expressions in the command line and get output in the command line interface as well. The valid operators to be used are the + (addition), - (subtraction), * (multiplication), / (division), % (modulo), and ^ (exponentiation) (** may be added later as the exponentiation operator). Unary negation and addition will also be available as valid input, and the numbers input must be integers only. The resulting output will be the correct numeric calculation of the input expression, in the form of a float or an integer.

6. Quality

The software to be designed will allow the addition of new commands and functions if needed, by adding recognizability for new operators and characters. The software will be made reliable by dealing with invalid expressions (invalid characters, invalid operation sequence, invalid operation order, etc.). The software will be extremely portable given the simple range of its functionality. Given that no personal user information is a necessary input for this software, no safety, security, or privacy considerations are needed for the design of this software.