The Company Company

<EECS 348 - Calculator Application>
Software Architecture Document

Version <1.0>

Revision History

Date	Version	Description	Author
<06/11/2024>	<1.0>	< Initial Calculator Software Architecture>	<Sionne, Meg, Zeidan, Ash, Jett>

Table of Contents

Software Architecture Document

1. Introduction

This document incorporates the overview and plan to be undertaken and plan to be undertaken for the project Arithmetic Expression Evaluator. The description shall be catered for in the purpose, scope, key definitions, and references that shall be used during the undertaking of the project to ensure that it remains consistent and up to requirements agreed upon.

1.1 Purpose

The purpose of this Software Architecture Document (SAD) will serve as guide for understanding the calculator application's composition and inner workings. It will break down the application's overall design, functionality, and core principles.

This SAD includes an introduction into the project's intended build for the application, what type of architecture the application implements, architectural goals and constraints, built-in subsystems, interface, and the overall quality of the build.

1.2 Scope

The SAD is applicable to the future design of the Company Company's calculator application – or any other future iterations of the application. Moreover, the developers, project managers, and stakeholders may also use this document as a guide for their own respective understanding and use of this specific type of calculator implementation.

1.3 Definitions, Acronyms, and Abbreviations

PEMDAS: stands for order of operation. The basis: Parentheses, Exponents, Multiplication, Division, Addition, and Subtraction. That's the important rule at the time of evaluation of an arithmetic expression. The Compiler: It is what would change the C/C++ code into machine language that the computer will execute. A Debugger finds and fixes bugs in the code. OOP (Object-Oriented Programming): that is the paradigm used in this assignment, and it guarantees modularity and maintainability.

1.4 References

The References subsection contains an entry for every document referenced in the Software Architecture Document. Each document is identified by title, report number, date, and publishing organization. Included is a statement of where each reference can be obtained either by adding an appendix or another document for convenience.

1.5 Overview

The Software Architecture Design document provides an overall structure of the document, including sections such as architectural goals, system views, and key design decisions. The structure will make it easy for stakeholders to find relevant information and understand the basic architecture on which the system is founded.

2. Architectural Representation

The Architectural Representation section views Logical, Development, Process, and Physical views that outline the system structure, code organization, runtime behavior, and hardware mapping for a better understanding by the varied stakeholders.

3. Architectural Goals and Constraints

4. Team – A team goal for the architecture of this project would be to slip the work into different groups such as design and implementation.

5. Structure – A structural goal for this program would be for it to have a well designed and constructed architecture that is easy to process and navigate.
    - Security – A security goal for the architecture would be to have safe coding practices and protecting against errors.
    - Development tools – Visual Studios, C++, and GitHub.
    - Portability – Design a program that can be easily used across multiple computers, such as downloading and then running the executable file onto different computers.

6. Logical View

6.1 Overview

This section describes the various layers and elements of design for the final calculator, like classes and packages within the system.

6.2 Architecturally Significant Design Modules or Packages

Front-End User Interface: This module handles all interactions between the user and the system.
- Classes:
- OutputManager: Outputs correction calculations and handles the format/ handles errors

Back-End Information Handling: This module processes all desired calculations
- Classes:
- CalculatorEngine: Main class for processing mathematical expressions
- Parser: Parses and tokenizes the input expression for evaluation
- ExpressionEvaluator: evaluates parsed tokens while managing PEMDAS

7. Interface Description

The application will begin with a blank screen and a blinking cursor (like that of a typical command line interface). The user may input any basic mathematical operation (+, -, *, /), parenthesis, and numerical values.

- Examples of valid input:
    - (2+1) * 3
    - (1+3) /7
- Examples of invalid input:
    - 8 – 3
    - (44 * 4)

Valid inputs will result in an answer and invalid will print out an error message to the screen.

8. Quality

The C+ calculator is modular in that adding new operations to the system is a relatively simple task. Input validation and error checking improves reliability, especially against the common divisions by zero. Standard libraries make it portable across platforms. Proper secure input handling prevents possible vulnerabilities.