Arithmetic Expression Evaluator in C++
Software Development Plan
Version 1.1

Revision History

Date	Version	Description	Author
09/18/2023	1.0	Group Meeting edits/Personal edits	Hannah
09/22/2023	1.1	Finalize individual sections	Hannah

Table of Contents

Software Development Plan

1. Introduction

1.1 Purpose

This Software Development Plan serves as a document aimed at collecting the essential details needed for the Arithmetic Expression Evaluator project's control and management. It outlines the strategy being used for software development in this project and stands as the primary plan that team members rely on to guide the project's progress.

The following people use the *Software Development Plan*:

- The **project owner** uses it to plan the agendas for each part of the project and the needs of each member's role, and to track progress using the schedule.

- The **utility developer** uses it to identify the key constraints of the project and help the team members overcome any potential obstacles faced while ensuring the quality the project.

- The **project designer** uses it to understand the needs of the project while designing the layout of the program being created by the group.

- The **project leads** use it to gather the information required to collaborate and work with the utility developer and project designer, as well as assign tasks to the team and communicate with the professor to obtain guidance on any future conflicts/technical problems.

- All **project team members** use it to understand their tasks, when to complete them, and how they depend on each other's role in the team.

1.2 Scope

The scope of this document is to take effect of the Arithmetic Expression Evaluator that this team is creating for EECS 348. It will describe the overall plan for the final version that will be turned in at the end of the semester. The final version will be implemented to the specifications of the Project Description.

1.3 Definitions, Acronyms, and Abbreviations

Data structures: various methods used to store and organize data
Heap: data structure specialization of a complete binary tree
PEMDAS: indicates order of operations – parenthesis, exponents, multiplication, division, addition, subtraction
Error handling: how a group recovers from an error found in the software
Debugging: process of removing errors from software
Object-oriented programming: language model that organizes design around objects such as classes or data fields

1.4 References

This Software Development Plan references this artifact list:
Project Description - a description provided by Prof. Saiedian including test cases.
   Agendas – will be sent in the team chat and accessible on the project GitHub
   Meeting Notes – will be accessible on the project GitHub
   Requirements Document – this a deliverable of the project
   Architecture and Design Document - this a deliverable of the project
   Project User Manual - this a deliverable of the project

1.5 Overview

The purpose of this document is to briefly explain the purpose and goals of the project, create clear
organization of the team, and define major deliverables of the project. This software Development Plan
contains:

Project Overview – Brief descriptions of the project purpose, scope, objections, assumptions, constraints   and
project deliverables

Project Organization – Project management plan as well as team members roles, contact and availability.

Management Process – Contains a breakdown of the deliverables and schedules deadlines, a phase plan,
iterations objectives and releases, as well as management plan.

2. Project Overview

2.1 Project Purpose, Scope, and Objectives

The main purpose of this project is to build a calculator that will take an arithmetic expression for an input,
analyze it, and calculate it in the order of PEMDAS. It must be able to use addition, subtraction,
multiplication, division, modulo, and exponentiation, as well as handle parenthesis and numeric constants.
It must be able to handle errors such as a number divided by 0. This project is also expected to be built in
C++ and use object-oriented programming principles with clear organization and documentation.

2.2 Assumptions and Constraints

This plan is based on the assumptions that:

- Each team member has coding knowledge in Python from previous courses and understands basic
coding concepts and terminology.

- All team members have experience implementing basic data structures from previous courses.

- All team members understand the order of operations and basic arithmetic concepts necessary to
identify the correct outputs inputs in the program.

This plan may have constraints including:

- Some team members may not have experience coding in C++, therefore they may have to allocate
more time to learning the programming language.

- Team members may have busy schedules and a lot of work for other courses (especially
midterms/finals) and planning supplementary meetings may be difficult at times.

- Sometimes a team member may not be able to make it to team meetings due to various reasons
(going out of town, other commitments, sickness, emergencies, etc.).

2.3 Project Deliverables

Deliverables for each project phase are identified in the Development Case. Deliverables are delivered
towards the end of the iteration, as specified in section *4.2.4 Project Schedule*.

- Project Management Plan due Sept. 24
- Requirements Document assigned Sept. 19
- Architecture and Design Document assigned Oct. 24
- Implementation assigned Oct. 31
- Test Cases assigned Nov. 14
- Project User Manual assigned Nov. 28
- Final version of the project implementation and the deliverables due Dec. 7

2.4 Evolution of the Software Development Plan

The *Software Development Plan* will be revised prior to November when the Phase Plan will need to be made.

3. Project Organization

3.1 Organizational Structure

The organizational structure of the team will be a skill-based organization with each member working on   code base for the project. There will be a project owner who will organize team meetings and turn in   project deliverables. The Utility developer ensures that all requirements are met and will test and debug the   final project. The Project Designer will work on designing the layout of the program. The Project leads will   take technical issues to the professor or TA and make implementation plans. Everyone will be responsible   for working on the project deliverables.

3.2 External Interfaces

N/A

3.3 Roles and Responsibilities

Person	Unified Process for EDUcation Role	Contact and Availability

Hannah Smith	Project Owner
	• Schedule any extraneous meetings
	• Compile and turn in all artifacts
	• Compile and send out meeting agendas
	• Work on Deliverables
		Contact: hrsmith123@ku.edu
		  (913)226-3861
		Availably:
		   Monday – after 12pm
		   Tuesday – after 6pm
		   Wednesday – after 3pm
		   Thursday – after 4pm
		   Friday – after 12pm
		   Saturday – All Day
		   Sunday – All Day

Madeline Watson	Utility Developer
	• Ensure key results/constraints are taken care of
	• Represent users and customers
	• Test and debug product results
	• Working on project deliverables
		Contact: madeline.watson@ku.edu
		   (913) 203-0491
		Availability:
		   Monday – 1-2pm, after 3pm
		   Tuesday – after 12:30pm
		   Wednesday – 11-2pm, after 3pm
		   Thursday – 11:30-2pm, after 4pm
		   Friday – 11-2pm, after 3pm
		   Saturday – All day
		   Sunday – All day

Christina Sorensen	Project Designer
	• Take meeting notes (alternate with Project Lead)
	• Designs the layout of the program based off needs of the program
	• Working on project deliverables
		Contact: christinasorensen@ku.edu
		(785)521-0246
		Availability:
		   Monday – Before 12 PM, 1-2 PM
		   Tuesday – After 6 PM
		   Wednesday – 1-2 PM
		   Thursday – Before 12 PM
		   Friday – Before 12 PM, 1-2 PM
		   Saturday – All Day
		   Sunday – N/A

Tejaswi Nimmagadda	Project Lead • Take meeting notes (alternate with Project Designer) • Work with Utility Developer to test and debug the program • Reporting technical issues to Professor/TA • Working on project deliverables	Contact: tnimmagadda@ku.edu (804)937-2760 Availability (can vary): Monday – 12-5 PM, After 7 PM Tuesday – Before 11 AM, After 4 PM Wednesday – 12-1 PM, 3-6PM Thursday – Before 11 AM, After 4 PM Friday – 12-2 PM, 4-5PM Saturday – All Day Sunday – 2-4 PM
Magaly Camacho	Project Lead • Work with Project Designer to design project to meet requirements • Make implementation plan and assign tasks and goals • Work on project deliverables	Contact: magaly@ku.edu (915)630-0239 Availability: Monday – After 12 PM Tuesday – 12-2 PM, After 4 PM Wednesday – After 3PM Thursday – 12-2 PM, After 4 PM Friday – 12-2 PM, After 3 PM Saturday – All Day Sunday – All Day

4.   Management Process

4.1   Project Estimates

N/A

4.2   Project Plan

The project will be broken down into several deliverables. The first of which is the project management plan, then the project requirements document, followed by a project architecture and design document. The project will be wrapped up with the implementation of the code to meet the previously laid out plan. The project will be complete when the code completes all the testing. Then a revised version of any previously completed documentation will be turned in.

4.2.1   Phase Plan

Can Leave blank until November

4.2.2   Iteration Objectives

Iteration I:
• Implement a function to tokenize the input expression
• Create a data structure to represent the expression's structure
• Define the precedence of operators according to PEMDAS

Iteration II:
• Implement logic to evaluate expression
• Develop mechanism to identify and evaluate expressions with parenthesis
• Recognize numeric constants in input

Iteration III:
• Allow error handling
• Develop user-friendly command line interface

4.2.3 *Releases*

Demo – create a data structure to house the expression, define precedence of operators

Beta – use logic to evaluate expression with parenthesis, without parenthesis, and with numeric constants   Final –
allow for error handling and create a user-friendly interface

4.2.4 *Project Schedule*

Recurring project meetings every Friday at 1pm.

Requirements Document assigned Sept. 19
Architecture and Design Document assigned Oct. 24
Implementation assigned Oct. 31
Test Cases assigned Nov. 14
Project User Manuel assigned Nov. 28
Project due Dec. 7

4.2.5 *Project Resourcing*

N/A

4.3   **Project Monitoring and Control**

Quality Control, Risk Management and Configuration Management will be briefly discussed in the
following sections. The Quality Control section will describe the timing and methods used to control the
quality of the project deliverables and how to take corrective action when needed. The Risk Management
section will describe the approach used to identify, assess, and mitigate risks. The Configuration
Management section will describe the actions to be taken when problems or changes arise, and it will
describe the stylistic and naming conventions for project artifacts. Requirements Management, Reporting
and Measurement will not be discussed as they are not applicable for this project.

4.4   Requirements Management

N/A

4.5   Quality Control

Defects will be recorded and tracked as Change Requests, and defect metrics will be gathered (see
Reporting and Measurement below).

All deliverables are required to go through the appropriate review process, as described in the Development
Case. The review is required to ensure that each deliverable is of acceptable quality, using guidelines and
checklists.

Any defects found during review which are not corrected prior to releasing for integration must be captured
as Change Requests so that they are not forgotten.

4.6   Reporting and Measurement

N/A

4.7   Risk Management

One of the main risks would be losing points since this project will impact a big part of the overall course
grade of each team member. Risks will be managed by reflecting on the feedback from instructors after
each deliverable is graded. As a part of the agenda at meetings, a plan for improvements will be made to
lower the risk of losing points and some or all the team members' roles may be updated.

4.8   Configuration Management

Appropriate tools will be selected which provide a database of Change Requests and a controlled versioned
repository of project artifacts.

All source code, test scripts, and data files are included in baselines. Documentation related to the source code is also included in the baseline, such as design documentation. All customer deliverable artifacts are included in the final baseline of the iteration, including executables.

The Change Requests are reviewed and approved by one member of the project, the Change Control Manager role.

5. Annexes

Other applicable process plans are listed in the references section, including Project Description.