University of Kansas EECS 348

Arithmetic Expression Evaluator in C++
Software Development Plan
Version 1.7

Revision History

Date	Version	Description	Author
09/12/24	1.0	Filled in basic information about the document	Lillian and Eliza
9/16/24	1.1	Filled out sections 1.1, 2.1, 2.3, 2.4, 4.1, 4.2.1, 4.2.2	Eliza
9/16/24	1.2	Filled out section 1.2	Tyler
9/16/24	1.3	Added to section 1.3	Daniel
9/17/24	1.4	Added to section 1.5 and 3.3	Max
9/19/24	1.5	Added to section 4.8 and 5	Eliza
9/22/24	1.6	Added schedule to 4.2.4 Filled out 3.1	Tyler
9/24/24	1.7	Removed all 1st person language	Lillian
9/24/24	1.8	Added to section section 1.5 and 4.7	Max
9/27/24	1.9	Added to risk management table	Daniel
9/27/24	2.0	Overview of the entire document, small fixes. Updated table of contents	Eliza

Table of Contents *[keep this; say N/A when inapplicable]*

Software Development Plan

1. Introduction

This is the Software Development Plan document. The introduction gives a rundown of the entirety of the document starting from the purpose to the overview of the Software Development Plan.

1.1 Purpose

This *Software Development Plan* holds all general and specific information for the project. This document will be referenced by the project manager to lead the team and by the team members to complete their work as per the plan. Adjustments will be made as necessary, but in general this document will be the top guidance resource throughout the time period that this project persists, which will be until the end of EECS 348 2024 Fall semester.

The *Software Development Plan* outlines a plan for the *Arithmetic Expression Evaluator in C++* project. The project entails developing an arithmetic expression parser in C++ for a larger compiler project. The completed project will be capable of parsing and evaluating arithmetic expressions, including the ability to handle the operators +, -, *, /, %, **, numeric constants, and parentheses within the expression.

The following people use the *Software Development Plan*:

- The project manager uses this plan to establish a schedule, deadlines, and resources for the project.
- The team members reference this document to find their assigned work for the project, and the deadlines for each phase of the project.

1.2 Scope

This *Software Development Plan* outlines how the development team will construct the Arithmetic Expression Evaluator. Additionally, this plan provides an overview of the team structure and responsibilities. The development plans discussed in this document are based on the requirements set in the EECS 348 course syllabus. For more information pertaining to individual iterations, see section 4.2.2.

1.3 Definitions, Acronyms, and Abbreviations

Discord- a messaging application used for communication. For the purposes of this project, the team has a Discord server with channels for meeting, documents, announcements, roles, etc.

Google Docs- Google docs is a word processing program that utilizes the cloud to allow for collaboration and auto-saving changes

GitHub- Github is a large server that hosts git repositories.

AEE- abbreviation for Arithmetic Expression Evaluator in C++

SDP- abbreviation for Software Development Plan

1.4 References

UPEDU Team. (2002). *Time Monitoring Tool Software Development Plan*. Retrieved September 20, 2024, from https://www.upedu.org/templates/cs/PM/upedu_ex_sdp.pdf

Saiedian, Hossein. (2024). *Striving for Successful Team Projects* [PowerPoint slides]. EECS 348 Fall 2024 Canvas.

Saiedian, Hossein. (2024). *Software development process models* [PowerPoint slides]. EECS 348 Fall 2024 Canvas.

1.5 Overview

This *Software Development Plan* contains the following information:

Project Overview — The "Project Overview" section details the project's purpose, scope, and objectives. This section also defines the deliverables that are expected from the project by its completion.

Project Organization — The "Project Organization" section describes the organizational structure of the project team and how the team will deal with external groups. This section also lists everyone on the team, their roles, availability, contact, and experience.

Management Process — The "Management Process" section explains the estimated cost and schedule, defines the major phases and milestones for the project, and describes how the project will monitor risks, quality, and requirements.

Applicable Plans and Guidelines — The project's development follows the Software Development Lifecycle. This lifecycle follows four steps:

Requirements Engineering: This phase is about determining the requirements of the project. This involves analyzing what is needed out of the final product, documenting those needs, and maintaining them throughout the project's lifecycle.

Design: This phase is about blueprinting what the project will likely end up as. It helps the team give a good idea of what they're working to create.

Construction: This phase is about creating the project itself. It involves the team coding and implementing the pre-established requirements.

Testing: This phase involves testing the product. The team will test the product, find issues that need resolving, fix them as needed, and repeat until the product is deemed complete.

2. Project Overview

2.1 Project Purpose, Scope, and Objectives

The purpose of the project is to create the Arithmetic Expression Parser, which will fit into a larger project of an unspecified language. The Arithmetic Expression Parser will take in a string input with numbers and character operators of +,-,*,/,%,**,(,). The Arithmetic Expression Parser will then use C++ to interpret these characters, making the numbers into integers and floats, and then give an output of appropriate and accurate mathematical evaluation including grouping of parenthesis.

2.2 Assumptions and Constraints

By having assumptions within the project, it is possible to generate solutions for the project. Team members can assume that the development environment will behave as expected, without any bugs. Team members can even assume inputs are valid and wouldn't have to worry about handling errors. Once an assumption is thought out, team members can then input thoughts into a document and work forward from the constraint. Constraints will help team members define the boundaries of what can be acceptable. For example, users may have limitations of what sort of syntaxes are allowed that will work, and other syntaxes that aren't defined within the program and are unusable. Basic operations such as +, -, *, /, %, and ** would be allowed within the project. Any other forms of operations would be unusable. When it comes to the () operation, team members can use assumptions to make sure the program handles it correctly by figuring out what to calculate first and assuming the operations work correctly. In conclusion, assumptions will allow the project to move forward by defining what team members expect to happen.

2.3 Project Deliverables

Deliverables are the product of each product phase, specified in the Development Case, and due date, each of which is specified in 4.2.4 Project Schedule. It is a high priority to finish each deliverable by the due date to ensure a timely and appropriate conclusion and grade.

2.4 Evolution of the Software Development Plan

The Software Development Plan is initially composed in parts by different team members and reviewed for clarity and accuracy in team meetings. The *Software Development Plan* will be revised prior to the start of each Iteration phase.

3. Project Organization

3.1 Organizational Structure

The development team for the *Arithmetic Expression Evaluator* consists of six team members. Eliza is the project manager for the AEE. As the project manager, Eliza is responsible for tracking the team's progress and ensuring the team meets project deadlines. Lillian is the scribe of the development team. As the scribe, Lillian is responsible for taking notes during the team's weekly meetings. Tyler is the Change Control Manager, as outlined in Section 4.8. The remaining team members, Daniel, Johney, and Max, serve as general team members. They work on all tasks related to the project.

The development team holds a weekly in-person meeting on Tuesdays. In this meeting, the team members review tasks for the upcoming week and construct a plan for completing the required work. Team members are also given assignments from the project manager that are to be completed by the next in-person meeting or iterations due date, whichever happens first. Additionally, the team holds an online meeting every Friday evening to discuss progress made on the assigned work. This online meeting also allows members who could not attend the in-person meeting to raise any questions or concerns about the project.

3.2 External Interfaces

In the future, this project will be used either by itself, or within a larger system. Because of the possibility that it will be used within a larger system, it is imperative the output of the arithmetic expression evaluator is in a format that is generalized such that it can be used in a majority of modern systems; specifically speaking, the output of the Arithmetic Expression Evaluator should be in integer or float format.

This product is not for monetary gain and will be assessed by the instructors of EECS 348 Fall 2024.

3.3 Roles and Responsibilities

Person	**Unified Process for EDUcation Role**
Lillian Brooks-Kanost	Team Member, Scribe lbrookskanost@ku.edu Available M/T/R 11:00-13:00, W 11:00-15:00, F 11:00-16:00, everyday after 1800 on Discord Experience in Python/Windows Limited experience in C++/C/Linux/JavaScript
Eliza Malyshev	Project Manager e602m203@home.ku.edu Available: Tuesday/Thursday 12:15 PM-12:50 PM for in-person meetings, after 7pm any day online via discord. Contact: email or discord to set up meeting Experience: C, Python, Windows, Mac Limited Experience: C++, Linux
Tyler Oswald	Change Control Manager tyleroswald@ku.edu Available Tuesday/Thursday 11:30 Am - 12:50 PM for in-person meetings, after 6:00 PM for virtual meetings. Contact: email or discord Experience: Python, Windows, Mac Limited experience: C++, linux
Daniel Harris	Team Member d488h586@ku.edu Available: Tues/Thurs 12:15 PM-12:50 PM for in person, available after 7pm any day for online except Wednesdays. Contact: email, discord for quicker replies. Experience: Python, Windows Limited experience: C++, Linux
Max Toney	Team Member m673t923@ku.edu Available: after 5:30pm any day for virtual meetings if no work. Unavailable for in person meeting time, however can meet in person by request after 5:30pm. Best contacted through Discord or Email. Experience: Python, Windows, Mac Limited Experience: C++, Linux, Java
Johney Makeen	Team Member johneymakeen@ku.edu Available: Upon requests, but not after 7pm on weekdays. Best contacted through email or discord. Experience: Python, Video games, Mac Limited Experience: C++, Linux, Java

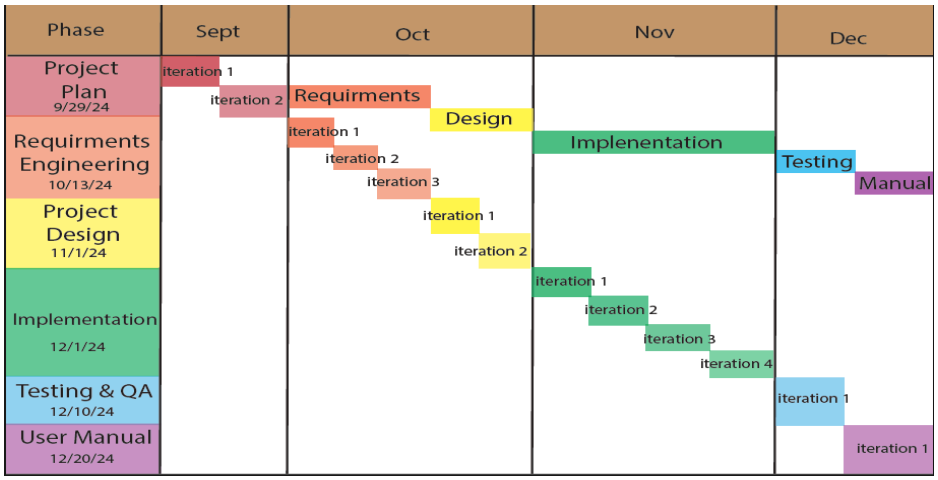
4. Management Process

4.1 Project Estimates

Cost: No cost will be incurred.
Schedule: Estimated time for the end of this project is before December 20th, 2024.

4.2 Project Plan

4.2.1 Phase Plan



There will be main phases for this project, outlined below, and multiple iterations for each phase.

1. Project Plan: all group members filled out sections assigned by project manager and each document section was completed by September 29, 2024, 6pm. Two iterations.
2. Project Requirements: requirements of election, documentation, and validation are all detailed and completed as per EECS 348 lecture. This will be finished by October 13, 2024 6pm. Tree iterations.
3. Project Architecture & Design: develop a detailed plan for the Arithmetic Expression Evaluator that addresses requirements. This will be completed by November 1st, 2024, 6pm. Two iterations.
4. Project Implementation: convert details of architecture and design into functioning C++ code, made of several parts, each of which will be completed by one or more team members and reviewed by the team as a whole. This will be completed by December 1tst, 2024, 6pm. Four iterations.
5. Project Test Cases: probe & test every possible input/output and scenario that could happen within scope and appropriate domain of reason. This should be done by December 10th, 2024, 11pm. One iteration.
6. Project User Manual: the manual will detail how to use the Arithmetic Expression Evaluator either alone or within a larger project. This will be done by December 20th, 2024, 11pm. One iteration.

4.2.2 Iteration Objectives

Project Plan- to be completed by 09/29/24

Iteration 1: Project plan document analyzed; sections assigned to project members by the project manager.

Iteration 2: Any necessary edits to the sections are made by the project manager.

Project Requirements- to be completed by 10/13/24

Iteration 1: Requirements of election and documentation are detailed.

Iteration 2: Requirements of validation are detailed.

Iteration 3: Previous iterations are refined and finalized.

Project Architecture and Design- to be completed by 11/01/24

Iteration 1: Initial plan for the AEE is drafted.

Iteration 2: Plan is refined and finalized.

Project Implementation- to be completed by 12/01/24

Iteration 1: Code assignments are determined by the project manager.

Iteration 2: Basic functionality is added to the project; prototype is released.

Iteration 3: Additional functionality is added to the project according to the project requirements. Test cases for the project are drafted.

Iteration 4: Test cases for the project are finalized. Implemented code is refined and finalized.

Project Test Cases- to be completed by 12/10/24

Iteration 1: Previously drafted test cases are used to test the project; any necessary edits to the project are made. Final release of the project is released.

Project User Manual- to be completed by 12/20/24

Iteration 1: User manual for all possible use cases is drafted and added to the project repository.

4.2.3 Releases

Prototype: First version of the drafted code pre-testing.

Final Release: Completed project.

4.2.4 Project Schedule

PROJECT TIMELINE

PROJECT TITLE	Arithmetic Expression Evaluator in C++	COMPANY NAME	EECS 348 Development Team
PROJECT MANAGER	Eliza Malyshev	DATE	9/22/24

PHASE		DETAILS	Q3		Q4			
			SEPT		OCT	NOV	DEC	
			15th - 29th	1st -13th	14th - 31st	1st - 30th	1st - 10th	11th - 20th
1	**Project Plan** **Due: 9/29/24**	- Analize project plan - Review and edit project plan	Project Plan					
2	**Requirements Engineering** **Due: 10/13/24**	- Requirements are detailed - Requirements are validated - Requirements are reviewed and finalized		Requirements				
3	**Project design** **Due: 11/01/24**	- Draft inital plan for the AEE - Review and finalize design			Project Design			
4	**Project implemintaion** **Due: 12/01/24**	- Coding assinments are created - Basic features are added - Additional features are added and test cases are created				Project Implemintation		
5	**Project testing and QA** **Due: 12/10/24**	- Tests defined in project implemintation are conducted - Final version of the project is released					Testing	
6	**Create user manual** **Due: 12/20/24**	- User manual is created and added to the project repository						Manual

PROJECT END

4.2.5 Project Resourcing

Overall, the development team requires little to no outside training to complete the AEE. However, if any team members have questions about C++ syntax or structure, they can refer to John Gibbon's C++ boot camp, which is listed below. Additionally, if a member of the AEE team has any general questions about C++, they can visit the website Stack Overflow.

John Gibbon's C++ boot camp Stack Overflow

4.3 Project Monitoring and Control

- *Requirements Management:* Lots of documentation, meeting notes and assigning roles to team members will be used to keep track of requirements and progress done to them
- *Quality Control:* Weekly meetings and deadlines to make sure progress is being made and to ensure that the project is running smoothly. Biweekly meetings will also be held over Discord where work will be shared over video and discussed as the group works through it.
- *Reporting and Measurement:* Updates to documents and work will be made at the end of each iteration. Metrics will be added to section 1.3 and if any other changes arise this document will be changed to accommodate that.
- *Risk Management:* We have systems in place to ensure deadlines are met by holding others accountable and allowing for role swapping. The group will also keep the program well documented by enforcing descriptions to be submitted for commits.
- *Configuration Management:* Change requests will be tracked on Google Docs, Discord, and Github. Docs will be used to track documentation, and Github for different versions of code.

4.4 Requirements Management

Requested changes to requirements are captured in Change Requests, and are approved as part of the Configuration Management process.

4.5 Quality Control

Any issues found will be documented in Change Requests to the and reported to the Change Control Manager, Tyler.

Each deliverable will be reviewed by the team as a whole to ensure quality.

All defects found during review will be promptly fixed before the deadline of the deliverable.

Quality Control helps fulfills the program's task correctly without any errors. The project's goal is to create a parser that will evaluate arithmetic operations, and if the operations were to have a single error within the program, this will undermine the project. Team members can create test cases to ensure the parser handles given inputs correctly (i.e. 2+2, (3 *5) + 7, etc.). If the test case was to receive an invalid input, the program must handle this. Team members can achieve this goal by having others review the program's codes and make sure the program follows the standards. It is crucial the parser is tested and reviewed after each use,

4.6 Reporting and Measurement

Updates to any documents and work will be made at the end of each iteration. Any metrics will be added to definitions and abbreviations in section 1.3. If any changes arise to budget or schedule, this document will be updated to accommodate that.

The main metrics used in this project is time and amount of work done till completion. These metrics will be evaluated per iteration, which happens weekly or biweekly (see section 4.2 for details).

4.7 Risk Management

Risks will be identified at every iteration as they arise. Team members will identify risks and share them with each other. Together, a solution will be made and the risk will promptly be resolved.

Risks will be addressed, described, and given a solution in the chart below.

Risk	Description	Solution
Not meeting deadline	Not being able to meet deadlines for any reason	Hold groupmates accountable as well as allowing group members to swap roles if necessary
Lack of documentation	Having a messy/mostly undocumented program	Ignore all commits that don't have a description for changes made attached

4.8 Configuration Management

Tools used to keep track of Change Requests will be Discord, Google Docs, and sometimes Github. For versioned repositories of documents look to Google Docs, and for versioned repositories of code and artifacts look to GitHub.

All source code, test scripts, and data files are included in baselines and Github. All documents are in Google Docs shared folder for this project. All customer deliverable artifacts are included in the final baseline and the GitHub Repository.

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

5. Annexes

The project will follow the UPEDU process that was detailed in EECS 348 lectures and that is summarized in the UPEDU project outline document provided in the Discord Documents Archive for reference.

Other applicable process plans are listed in the references section, including Programming Guidelines.