Math Olympiads

Arithmetic Expression Evaluator
Software Development Plan
Version <1.0>

Revision History

Date	Version	Description	Author
<20/09/24>	<0.5>	<Initial Meeting and Setup>	<Whole Team>
<27/09/24>	<1.0>	<Finish and polish document for submission>	<Whole Team>

Table of Contents

Software Development Plan

1. Introduction

1.1. Purpose

The purpose of the *Software Development Plan* is to gather all information necessary to control the project. It describes the approach to the development of the software and is the top-level plan generated and used by managers to direct the development effort.

Each team member will edit sections relevant to their roles (see section 3.2).

1.2. Scope

This project is a short term, limited project to take no longer than three months. It extends to, but does not surpass, developing and minor deployment of an Arithmetic Expression Evaluator. This is a program which simplifies and solves expressions based on the PEMDAS method, correctly analyzing parentheticals, exponentials, and other relatively complex operations.
The plans as outlined in this document are based upon the product requirements as defined in the *Vision Document*.

1.3. Definitions, Acronyms, and Abbreviations

AEE - Arithmetic Expression Evaluator

1.4. References

Project Measurements Document (AAA-BBB-X.Y.doc)

Vision Statement

Risk List Document (CCC-DDD-X.Y.doc)

Configuration Management Plan (EEE-FFF-X.Y.doc)

1.5. Overview

This *Software Development Plan* contains the following information:

Project Overview — Describes the purpose of the project, detailing the assumptions and constraints of the project, and lists the project deliverables, including target delivery dates.

Project Organization — This section covers the organizational structure of the project team, including management. This section also describes how the project interfaces with external groups, and the roles and responsibilities of each team member.

Management Process — This section details the allocation of time to project phases, identifies major milestones, and outlines iteration objectives. This section also contains a project schedule that contains descriptions and deadlines for the Requirements, Design, Implementation, and Testing phases.

Applicable Plans and Guidelines — Details the type of process our project will follow, outlining methods, tools and techniques to be followed.

2. Project Overview

2.1. Project Purpose, Scope, and Objectives

In this project, we plan to develop and deliver an Arithmetic Expression Evaluator (to be integrated into an operating system at some point in the future) in C++ along with a simple command line user interface. This includes evaluating simple mathematical expressions adhering to PEMDAS.

2.2. Assumptions and Constraints

Assumptions

- The chosen tools, platforms, and technologies will remain stable and supported during the project lifecycle
- Team members and required hardware will be available as planned
- Any third-party systems or integrations will function as expected and on time.
- Adequate time will be allocated for testing and quality assurance activities.

Constraints

- The project staff will be constrained to our team of 6 people
- The equipment for this project will be constrained to the team members' computers
- Our project will be constrained to the schedule outlined in section *4.2.4 Project Schedule*

2.3. Project Deliverables

See section *4.2.4 Project Schedule* for details on project deliverables.

2.4. Evolution of the Software Development Plan

The *Software Development Plan* will be revised prior to the start of each Iteration phase.

Name	Date	Description
(Full Team)	9/20/24	Create document and add general information.

Criteria for Unscheduled Revision and Reissue of this Plan:

If any of the following criteria are met, a revision and reissue of the plan may be necessary.

Significant Scope Change: If project requirements or objectives are altered, introducing new features or removing key functionality, the plan must be revised to reflect these changes.

Critical Milestone Delays: Unanticipated delays in key development phases or the inability to meet deadlines necessitate a reissue of the plan to account for schedule adjustments.

Resource Modifications: Changes in the development team, budget, or available technology/resources that could affect project timelines or deliverables require a plan update.

Risk Emergence: Identification of new, significant risks or issues not previously anticipated, which could impact project success, may trigger an unscheduled revision.

Tooling or Technology Shifts: Adoption of new development tools, platforms, or frameworks that impact the implementation process should prompt a plan reissue to address these shifts.

Quality or Performance Issues: If major defects, performance issues, or failure to meet quality standards are detected, the plan may need to be reissued with revised testing or quality assurance processes.

3. Project Organization

3.1. Organizational Structure

Each member is responsible for their own area of the software (see section 3.2). The product lead will ensure meetings are scheduled so general discussion can occur between the leads. All members have equal say in all decisions, however the lead responsible for that area of work should be consulted first. If a consensus is not able to be reached, the corresponding lead is the one who shall break any ties.

3.2. Roles and Responsibilities

Person	Unified Process for EDUcation Role
Aidan Atwood	Quality Assurance Lead: Responsible for adequate testing of all aspects of the program.
Aidan Bergeman	Product and VCS Lead: Ensures all aspects of the plan are being fulfilled, as well as being responsible for viewing and approving pull requests.
Joel Harrison	Design Lead: Creates the overall design plan for the software, as well as answering design related questions.
Liam Kinghouser	UI/UX Lead: Responsible for the overall user experience and user flow.
Alejandro Sandoval	Technical Lead: Determines technical solutions to requirements and works within constraints.
Tyler Stuckey	Scrum Master: Tracks progress of the team, plans sprints, ensures velocity is sufficient to reach goals.

- ### 3.3 Team Information

Person	Contact Information and Technical Experience
Aidan Atwood	Programming Languages: Python, HTML, Java, Bash OS Experience: Windows, Linux Contact: aidan.atwood@ku.edu
Aidan Bergeman	Programming Languages: Python, Rust, Java, Bash OS Experience: Ubuntu, MacOS, Windows Contact: aidanberegman@ku.edu
Joel Harrison	Programming Experience: C#, Python, HTML, CSS OS Experience: Windows, MacOS, SteamOS Contact: joel.harrison@ku.edu
Liam Kinghouser	Programming Experience: Java, Python, HTML, CSS, JavaScript OS Experience: MacOS, Windows Contact: lkinghouser@ku.edu
Alejandro Sandoval	Programming Experience: Python, Javascript, C, Lua OS Experience: Windows, Linux Contact: alesan@ku.edu
Tyler Stuckey	Programming Experience: Java, Python, HTML, PHP OS Experience: Windows, Linux Contact: tylerstuckey@ku.edu

4. Management Process

4.1. Project Estimates

N/A

4.2. Project Plan

4.2.1. Phase Plan

N/A

4.2.2. Iteration Objectives

Features will be added in order of importance. Some features cannot be added until fundamental functionality is added first, therefore the following order will be followed to allow for an iterative approach:

- Recognize Numeric Constants
- Expression Parsing and Operator Precedence
- Parenthesis Handling
- User Interface
- Error Handling

4.2.3. Releases

- <0.1> Demo: Basic calculator functionality will be added at this stage. It may not contain all the features we want and it will
- <0.2> Beta: This version will contain all requirements. The features will be refined as this version is tried out.
- <0.3> Release Candidate: Implementation is fully finished at this stage. Final testing will start here.
- <1.0> Release: Product can be released once testing is done and all bug-fixes are completed, if needed. This will be the final version that will be submitted.

4.2.4. Project Schedule

Joint Meetings: Meetings to discuss current progress and verify feasibility of the schedule will occur each Friday at 5:00PM CST in LEEP2 2320 unless otherwise communicated via messaging.

Each meeting will consist of a follow up on last week's discussion, as well as address any issues that might have come up. Goals for the next week will be set, primarily remaining small in scope to be reasonably accomplished in a short term. Before ending the meeting, any additional questions can be asked so everyone is on the same page. Meeting notes are present in the GitHub repository under the directory notes.

	Completion Target Date	**Description**
Requirements	10/20/2024	The requirements for the project need to be gathered as soon as possible in order to proceed in a timely manner.
Design	10/26/2024	A clear blueprint will be created for implementation.
Implementation	11/9/2024	Implementation will take a long time. Development will proceed iteratively, with each release functioning as a development phase.
Testing	12/1/2024	Before release, testing will be done to find all issues and enough time will be provided to fix them before the submission due date.

4.2.5. Project Resourcing

N/A

4.3. Project Monitoring and Control

4.4. Requirements Management

The requirements for this system are captured in the Vision document. Requested changes to requirements are captured in Change Requests, and are approved as part of the Configuration Management process.

Requirements are listed in a way that facilitates the project's iterative approach, which means basic functionality will be the main priority and more detailed functionality will be considered extensions. Emphasis will be put on finding possible edge cases.

4.5. Quality Control

Defects will be recorded and tracked as using GitHub issues, and defect metrics will be gathered (see Reporting and Measurement below).

The QA Lead is responsible for ensuring all issues are brought forward to the team, as well as testing the software to catch future issues.

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.

The quality of the project will be ensured through code reviews, requirement reviews, and testing. The codebase of the program must remain readable to facilitate collaboration and to speed up bug fixing within our limited time frame.

4.6. Reporting and Measurement

Updated schedule estimates, and metrics summary reports, will be generated at the end of each iteration.

The Minimal Set of Metrics, as described in the RUP Guidelines: Metrics will be gathered on a weekly basis. These include:

Earned value for completed tasks. This is used to re-estimate the schedule and budget for the remainder of the project, and/or to identify need for scope changes.

Total defects open and closed – shown as a trend graph. This is used to help estimate the effort remaining to correct defects.

Acceptance test cases passing – shown as a trend graph. This is used to demonstrate progress to stakeholders.

Refer to the Project Measurements Document (AAA-BBB-X.Y.doc) for detailed information.

4.7. Risk Management

Risks will be identified in the Inception Phase using the steps identified in the RUP for Small Projects activity "Identify and Assess Risks". Project risk is evaluated at least once per iteration and documented in this table.

Making sure quality control is able to take place continuously is an important part of risk management.

Refer to the Risk List Document (CCC-DDD-X.Y.doc) for detailed information.

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 at least one member of the project, the VCS Lead.

Tools to be used:

- gcc >= 11.4.0
- git >= 2.34.1
- GitHub

-

5. Annexes

The project will follow the UPEDU process.

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