Kansas Instruments™ (Not Really)

Arithmetic Expression Evaluator in C++
Software Development Plan

Version 1.4

Revision History

Date	Version	Description	Author
09/15/2023	1.0	This was the first day we worked on the Software Development Plan, got sections one and two done.	Benjamin Kozlowski, Steve Gan, Jacob Leehy, MJ McGee, Nicholas Hausler
09/19/2023	1.1	This was our second day of working on it, we finished section three and much of section four.	Benjamin Kozlowski, Steve Gan, Jacob Leehy, MJ McGee, Nicholas Hausler
09/22/2023	1.2	This was our third day of working on the project plan, which is now finalized. There will be additions in the future, but the initial plan is now complete.	Benjamin Kozlowski, Steve Gan, Jacob Leehy, MJ McGee, Nicholas Hausler
10/13/2023	1.3	Added the SRS to the references.	Benjamin Kozlowski
12/01/2023	1.4	Added final touches to references.	Benjamin Kozlowski

Table of Contents

Software Development Plan

1. Introduction

1.1 Purpose

The purpose of this *Software Development Plan* is to organize and structure the project, so that the development team can have a reliable framework from which to work off. It describes the development approach for the software and has been written by the team, so that in the future it can be updated and used as a guide by the team members.

The following people will be using the *Software Development Plan:*

- The **Project Manager** will use it to arrange and schedule the progress the software should be making.
- The **Project Team Members** will use it as a framework for their scheduled work on the software and to know what they should be doing.

1.2 Scope

This *Software Development Plan* describes the plan to be implemented by the *Arithmetic Expression Evaluator in C++* project, including deployment of the product. The details of the individual iterations will be described in the Iteration Plans.
The plans as outlined in this document are based on the product requirements as defined in the EECS 348 Term Project Description file.

1.3 Definitions, Acronyms, and Abbreviations

See the Project Glossary.

Operators:

- + : addition
- - : subtraction
- * : multiplication
- / : division
- % : modulo
- ^ : exponential

EECS

KU

1.4 References

Meeting logs: Notes taken by the Recording/Secretarial Engineer describing the activities within each meeting and other miscellaneous information.

GitHub Repository: The project will be documented through a GitHub repository containing all relevant information regarding the project.

Software Requirements Specifications (Document 2): This document houses the functional and non-functional requirements for the project. It will be stored in the GitHub repository.

Software Architecture Document (Document 3): This document goes over how the program will be created and organized, and how different parts of it will interact with one another.

User Manual (Document 4): This document will go over the steps on how to get started with the program, what it can and cannot do, and give examples on how to correctly use the Arithmetic Expression Evaluator in C++.

Test Cases Document (Document 5): This document houses the multitude of test cases that were created to attempt to find and fix any bugs or problems with the program.

Iteration Plans: The plan is to divide the project into two major goals: parsing the order of operations and parsing the arithmetic of the equation. We will iteratively implement the following operators: +, -, *, /, %, and ^.

Vision: Our vision for this project is to create an Arithmetic Expression Evaluator using C++ to the specifications in this document.

Glossary:

- EECS: Electrical Engineering and Computer Science
- KU: The University of Kansas
- UML: Unified Modeling Language
- SRS: Software Requirements Specifications
- SAD: Software Architecture Document

1.5 Overview

This *Software Development Plan* contains the following information:

Project Overview — The purpose of this project is to create a C++ program that can parse and evaluate arithmetic expressions containing operators as well as numeric constants. A project plan, requirements document, test cases, and design document will be formulated along with the final program.

Project Organization — The project team is organized into a variety of roles, including management, quality assurance, secretarial/recording, configuration management, and scheduling.

Management Process — The cost of the project is expected to be zero. This project will be completed over the course of the fall 2023 semester and will go through the following phases:

- Planning
- Design
- Implementation
- Testing
- Finalization

Applicable Plans and Guidelines — Our team will utilize various tools, including UML, GitHub, and the EDUcation document template, throughout the development process. We will also take advantage of the resources provided by the KU EECS department and our personal computers. We will follow an iterative design model.

2. Project Overview

2.1 Project Purpose, Scope, and Objectives

The tangible purpose of this project is to create a C++ program that can parse and evaluate arithmetic expressions containing operators +, -, *, /, %, and ^ as well as numeric constants. The program will be able to handle parenthesis and accurately determine the correct precedence of operations to produce an accurate result when prompted with user input. The program should also be expected to handle user input errors.

An adjacent objective of this project is formulating a professional development process which produces a finalized project plan, requirements document, thorough test cases, and design document in addition to the final program.

2.2 Assumptions and Constraints

It is assumed that the project team will remain a 5-person project that will be completed iteratively and be submitted for final review by the end of the fall 2023 semester. Our equipment will be composed of our personal computers and the lab machines available via the KU EECS department. Our personal constraints will be determined by our individual schedules over the next semester.

2.3 Project Deliverables

The first deliverable is this complete and thorough Software Development Plan. This Software Development Plan should lay out the team organization, the process for completing the software, and referencing any external sources that contributed to or were involved in the project's completion. Target Date: 09/24/23

A complete documentation for the Project Requirements. This should establish what the Expression Evaluator is expected to be capable of, how it should handle exceptions, what use cases are expected and how to process them, and what non-functional requirements need to be met by the finished product. Target Date: 10/17/23

Documentation of the Project Architecture and Design. This should define how the project is to be put together based on the established requirements. Target Date: 11/04/23

Documentation on Project Test Cases, verifying that the complete software meets all requirements documented in the Projects Requirements document Target Date: 12/02/23

A complete User Manual that informs any given user how to operate the completed Arithmetic Expression Evaluator. This should also provide examples of how to use the software. Target Date: 12/05/23

Fully implemented project, including revisions for the project management plan, requirements documents, design documents, test cases, C++ source code, and the user manual. Target Date: 12/07/23

2.4 Evolution of the Software Development Plan

The *Software Development Plan* must be updated and revised each session to incorporate the goal assigned during each meeting.

The Software Development Plan will be revised before each iteration phase starts.

Date	Version	Criteria
9/15/2023	1.0	The Software Development Plan 1.0v will include the full "Introduction," "Project Overview," and "Project Organization" sections to completion, with each section providing a full description of each statement with references (if required)
9/19/2023	1.1	The Software Development Plan 1.1v will include the full revised plan that includes references and project sources
9/22/2023	1.2	Reviewed the Software Development Plan and made some edits to previously completed sections.

3. Project Organization

3.1 Organizational Structure

The project team is composed of five individuals, each with a role set for them in the project's scope. Throughout the project, these roles may change, but at any one point in time, the project will have these roles:

- Project Manager: The individual in charge of managing the work being done on the software.
- Recording/Secretarial Engineer: The individual who takes notes of the meetings and alerts the team when new project requirements have been given.
- Quality Assurance Engineer: The individual in charge of assuring that quality work is being done and all requirements are met.
- Configuration Management Engineer: The individual in charge of configuring the software to meet the necessary goals.
- Scheduling Engineer: The individual in charge of scheduling meeting times and making sure that each member can attend group meetings.

3.2 External Interfaces

N/A

3.3 Roles and Responsibilities

Person	Unified Process for EDUcation Role
Benjamin Kozlowski	Project Manager
Jacob Leehy	Quality Assurance Engineer
Nicholas Hausler	Scheduling Engineer

Steve Gan	Configuration Management Engineer
MJ McGee	Recording/Secretarial Engineer

Project Manager: Benjamin Kozlowski

Responsible For:

- Overseeing/leading team meetings.
- Monitoring work done by team members
- Manage potential shortcomings
- Fill in for team members unable to complete their duties (i.e., sickness, emergency)
- Contacting professor/TA with any questions
- Working on project deliverables

Quality Assurance Engineer: Jacob Leehy

Responsible For:

- Validating published material
- Reporting any issues with deliverables
- Review work and test products
- Organize solutions for discovered issues
- Working on project deliverables

Scheduling Engineer: Nicholas Hausler

Responsible For:

- Asking for schedules from teammates
- Scheduling meetings on possible dates
- Working around schedules to ensure everyone has a chance to attend meetings
- Working on project deliverables

Configuration Management Engineer: Steve Gan

Responsible For:

- Configuring the software to meet the necessary goals
- Approving/Denying change requests to deal with bugs
- Managing the GitHub repository
- Working on project deliverables

Recoding/Secretarial Engineer: MJ McGee

Responsible For:

- Taking notes during team meetings
- Keeping track of meeting attendance
- Alerts the team of new project requirements
- Working on project deliverables

Benjamin Kozlowski
Contact Info: bkozlowski@ku.edu (559-590-1500)
Major: Computer Science
Relevant Completed Courses: EECS 168, 268, 140
Relevant Currently Enrolled Courses: EECS 348, 210

Programming Knowledge: Python, JavaScript
Basic Knowledge: C, C++
Availability:
Monday: 2:00 pm -8:00 pm
Tuesday: 6:00 pm - 8:00 pm
Wednesday: 4:00 pm - 8:00 pm
Thursday: 6:00 pm - 8:00 pm
Friday: 2:00 pm - 8:00 pm

Jacob Leehy
Contact Info: jacob.leehy@ku.edu (816-839-8847)
Major: Computer Science (Sophomore 2026)
Relevant Completed Courses: EECS 168, 268, 140
Relevant Currently Enrolled Courses: EECS 348, 210, 388, 468
Programming Knowledge: Python, JavaScript, HTML, Java
Basic Knowledge: C, C++, UML
Availability:
Monday: 11:00 am - 8:00 pm
Tuesday: 5:00 pm - 9:00 pm
Wednesday: 11:00 am - 9:00 pm
Thursday: 7:00 pm - 9:00 pm
Friday: 11:00 am - 9:00 pm

MJ McGee
Contact Info: mj.mcgee@ku.edu (515-782-5535)
Major: Computer Science
Relevant Completed Courses: EECS 168, EECS 268, EECS 140
Relevant Currently Enrolled Courses: EECS 348, EECS 210
Programming Knowledge: Python, C#
Basic Knowledge: C, C++, Java
Availability:
Monday: 8:00am – 11:00am
Tuesday: 8:00am – 12:00pm
Wednesday: 6:00pm – 8:00pm
Thursday: 8:00am – 12:00pm
Friday: 8:00am – 11:00am

Steve Gan
Contact Info: steve.gan0710@gmail.com (913-226-3989)
Major: Computer Science
Relevant Completed Courses: EECS 168, 268, 140, 330, 210, MATH 290, 127
Relevant Currently Enrolled Courses: EECS 348, 510, 461, 468
Programming Knowledge: Python, JavaScript, Java
Basic Knowledge: C, C++, React.js
Availability:
Monday: All Day Online
Tuesday: 9:00 am - 12:00 pm, after 4:00 pm
Wednesday: After 2:30 pm
Thursday: 9:00 am - 12:00 pm, after 4:00 pm
Friday: All Day Online

Nicholas Hausler
Contact Info: n704h227@ku.edu
Major: Computer Science
Relevant Completed Courses: EECS 168, 268, 140
Relevant Currently Enrolled Courses: EECS 348, 468, 330, 388
Programming Knowledge: Python, JavaScript, HTML, Git
Basic Knowledge: C, C++
Availability:

Monday: 12:30 pm - 10:00 pm
Tuesday: 4:00 pm - 10:00 pm
Wednesday: 3:00 pm - 10:00 pm
Thursday: 4:00 pm - 10:00 pm
Friday: 12:30 pm - 10:00 pm

4. Management Process

4.1 Project Estimates

N/A

4.2 Project Plan

Our project resources are composed of the five members of our group and the time we'll be able to dedicate to the project's implementation. The rough schedule for our project will be as follows:

- September 5: Project part 1 assigned: Project management plan.
  - Artifacts to be delivered: Project Management Plan
- September 19: Project part 2 assigned: Project requirements.
  - Artifacts to be delivered: Project requirement documentation
- October 24: Project part 3 assigned: Project architecture and design.
  - Artifacts to be delivered: Project architecture and design documentation
- October 31: Project part 4 assigned: Project implementation.
  - Artifacts to be delivered: Project implementation documentation
- November 14: Project part 5 assigned: Project test cases.
  - Artifacts to be delivered: Project test case catalog
- November 28: Project part 6 assigned: Project user manual.
  - Artifacts to be delivered: Project user manual
- December 5: Final submission made.
  - Artifacts to be delivered: Final GitHub repository

4.2.1 Phase Plan

N/A

4.2.2 Iteration Objectives

Iteration 1: Implement a stack to be used in parsing operations.	October 31st
Iteration 2: Addition operator and integer math.	November 2nd
Iteration 3: Subtraction and multiplication operands.	November 4th
Iteration 4: Division and modulo operands.	November 6th
Iteration 5: Exponentiation operator and float math.	November 8th
Iteration 6: Parenthesis handling and numeric constants.	November 10th
Iteration 7: Final team verification of software.	November 12th
Iteration 8: Testing begins. Any corrections will be made after this date.	November 14th

4.2.3 Releases

Version	**Date**	**Description**
0.0	9/15/2023	Project Creation

4.2.4 Project Schedule

Milestone	**Target Date**
Project Management Plan	Sept. 24
Project Requirements	Oct. 24
Project Architecture and Design	Oct. 31
Project Implementation	Nov. 14
Project Test Cases	Nov. 28
Project User Manual	Dec. 5th
Final Submission	Dec. 5th

4.2.5 Project Resourcing

N/A

4.3 Project Monitoring and Control

Project Monitoring and Control will be composed of three parts: Quality Control, Risk Management, and Configuration Management. These can be summarized as ensuring overall quality and maintaining processes to address issues that may arise throughout the project. We will ensure that these processes, as described below, will be followed to mitigate risk and manage complexity.

4.4 Requirements Management

N/A

4.5 Quality Control

Any defects or bugs will be recorded and tracked through change requests and brought up during team meetings.

Deliverables for each milestone will go through the appropriate review process by the quality assurance engineer. This process will ensure that each deliverable is consistent with the original Project Description and is free of any defects.

Deliverables will also be assessed by the team as a whole prior to submission to ensure that all required components are present and functional.

4.6 Reporting and Measurement

N/A

4.7 Risk Management

The main risk involved with this project is the case that a team member either drops the class or simply stops working with the team, refusing to communicate, or purposefully sabotaging the team's work. In this case, the issue will be documented, and steps will be taken with the help of TA's or the professor to rectify the problem.

4.8 Configuration Management

Project documents will be stored in a GitHub repository, including this planning document, test cases, and source code. Changes made to this document will be uploaded in a .pdf format to GitHub.

Potential changes to source code or any other deliverables will be reviewed by the configuration management engineer and the rest of the team members. Once cleared, the changes will be uploaded.

5. Annexes

The project will follow the UPEDU process.

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