Final Project Requirements & Guidelines

CSC 413, Fall 2023

Your final project demonstrates the software development skills you have learned this semester. For the last several weeks we have discussed project ideas and ways to start the planning process. You may choose your own project goals, as long as the program is appropriate for a classroom setting and meets the requirements below.

Prompts

If you are still unsure about what project you would like to submit, consider starting with one of the following prompts. You may change the prompt as you like to make the project more interesting to you if desired.

1. Adventure game, using a CLI (text) interface or a GUI (Swing) interface, in which the player must solve a series of puzzles or challenges to reach a destination.
2. Physics simulation which generates data files (e.g. CSV) describing how one or more physical processes develop over time. Homework Project 2 had you write a simple version of this program -- simulating a billiard ball in free-fall.
3. Stock exchange simulation, in which the user can check the price of simulated companies' shares and buy or sell them to earn points over time.
4. Note-taking application which lets the user organize information in various ways, prioritize items on their to-do list, etc. This project is best suited for a Swing GUI.

Submission

Your submission must include the following items. Requirements for each are described below. It is OK for your UI or class diagram to change over time. Your source code does not need to exactly match your planning documents.

1. Plan for user interface (PDF)
2. Planned class diagram (PDF)
3. Java source code
4. Brief summary of your development process (PDF)

NOTE: If any third-party libraries or resources are required to run your program, DO NOT include the library code with your submission, just make note of the requirement.

1. Plan for User Interface

Your final project submission must include a **PDF** showing how you expect a user to interact with your program.

If you are working on a project with a GUI, draw a diagram showing the planned layout of the UI for each window. Describe the function of each button, etc. the user will interact with. Also describe any “pop-up” or modal windows the program may ask the user to respond to.

If you are working on a Console / Command Line project (using text to interact with the user), describe each menu or set of options available to the user. Include a brief description of important use cases and how a user would interact with the menus to perform each task. For example, see the use/test cases in our Homework Projects.

2. Planned Class Diagram

Your final project must include AT LEAST 10 Classes and/or Interfaces you have created. Re-use of example code from the course is allowed, but will not be counted toward this requirement. Third-party code such as libraries will also not be counted.

Your class diagram may change over time and deviate from this initial plan. As you write the code, you may need to add new classes or remove others. However, both this plan and your final source code must meet the above minimum size requirement.

To show that you have considered your code architecture before writing the code, draw a Class Diagram describing the classes/interfaces you plan to implement.

For each class or interface, include:

- Brief description of the class/interface’s purpose in the program. Why do you need it?
- Name, data type, and access level of member variables (if any)
- Function signatures of public methods.

Show composition, inheritance relationships, and the implementation of interfaces by drawing arrows between the classes as we have done in class.

You may produce your diagram digitally, on paper, on a whiteboard, etc. but please submit your class diagram in **PDF** form.

Template for a Class Description

Name

Description of the class and its purpose.

Variables	Methods
- myPrivateVariable: datatype	+ methodName(argument1: datatype) -> returntype
+ myPublicVariable: datatype	

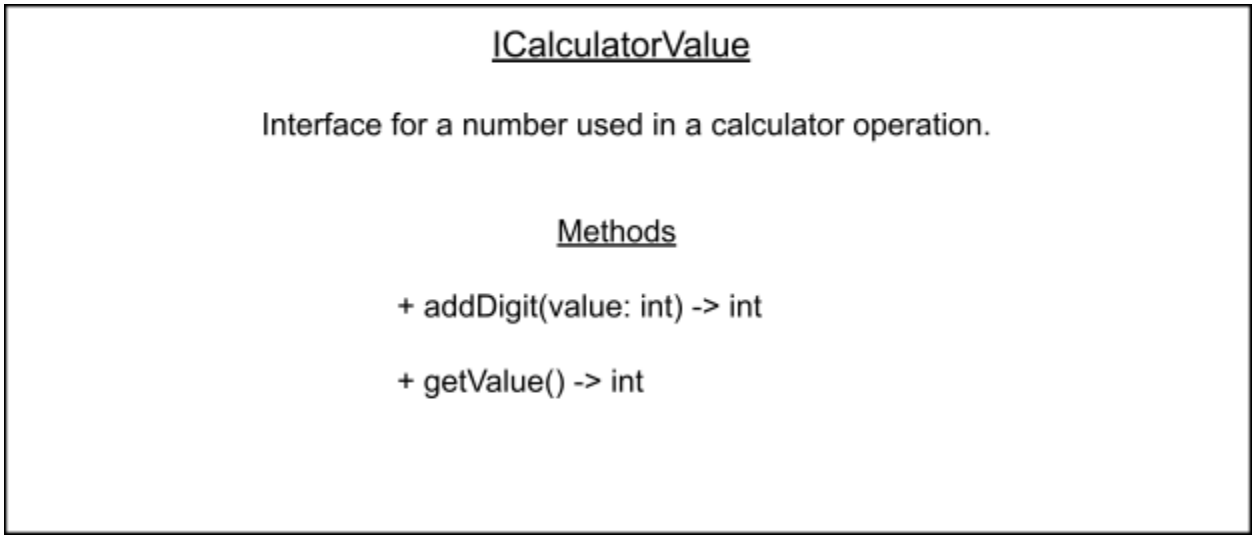
Example Class Description

Number

Stores a numerical value used in a calculator operation.
Values are constructed sequentially by pressing individual numbers on the keypad. This class stores the number being entered.

Variables	Methods
- value: int	+ Number()
	+ Number(initialValue: int)
	+ addDigit(digit: int) -> void
	+ getValue() -> int

Example Interface Description



3. Java Source Code

Your final project must include AT LEAST 10 Classes and/or Interfaces you have created. Re-use of example code from the course is allowed, but will not be counted toward this requirement. Third-party code such as libraries will also not be counted.

The code you submit will be graded on both its structure (the class diagram I can draw by reading your code) and its quality (how the code is formatted).

Your code should apply the design principles we have learned in class, even if your code does not follow your initial plan exactly. This includes the proper use of abstraction, encapsulation, composition, inheritance, and design patterns as needed in your program.

The code should also be well-formatted and easy to read. Code quality will be graded based on:

- Naming conventions of variables, classes, methods, etc.
- Use of whitespace (blank lines) to organize blocks of related code
- Use of comments to describe blocks of code
- Use of documentation comments to describe methods/classes
- Use of intermediate variables where appropriate
- Length of methods (how many lines of code per method)

4. Brief Summary of Development Process

When you are done writing your code, please write a few paragraphs in a **PDF** summarizing how the project went. Reflect on how your plans influenced your coding, how much your design changed during implementation, and what you could improve if you were to continue working on this project.

If your project's User Interface is significantly different from your UI plan, please also include instructions for the program's use.