# *CSC 413 Project Documentation*

# *Spring 2019*

***Student name: Randy Chen***

***Student ID: 922525848***

***CSC413.01***

***GitHub Repository Link:***
[https://github.com/csc413-SFSU-FA2024/calculator-Randychen10](https://github.com/csc413-SFSU-FA2024/calculator-Randychen10)

# Table of Contents

# 1 Introduction

## 1.1 Project Overview

In this project, we are creating a calculator using the provided files in the github repo. This calculator is supposed to be similar to a virtual calculator you would find on the internet. The objective is to pass all test cases, and code in the functions for the GUI version of the calculator.

## 1.2 Technical Overview

This program is written in Java and is split up into two different folders. The first being the Evaluator folder. In this folder we are given four classes, the Evaluator, EvaluatorUI, an InvalidTokenException, and Operand. The second folder contains the operator class, this class is an abstract class. In this folder I was required to create sub classes for the different types of operators in this assignment and set a priority to these operators.

Using a hashmap in the Operator class, we are able to route keys to their subclasses to perform the operation. Each subclass contains two abstract methods that came from the parent operator class.

Inside the Evaluator class, operands and operators are each stored in a stack. We have to determine if the stack is empty as well as checking for parenthesis. After that, we take the problem provided by the user, and run it through the Operator class to check for their priority. Finally, we return the value back to the user.

## 1.3 Summary of Work Completed

In the main Operator class, we utilize a hashmap to be able to route keys to their subclasses. Each subclass contains methods from the main Operator parent class to correct perform their operation. The methods from the parent class are abstract methods the gives the operators priority and the ability to execute. I fixed the error in Evaluator which was creating an object linking to the hashmap created in the Operator class. EvaluatorUI was missing code to give it function, so when we would start it, nothing would happen when the buttons were pressed.

# 2 Development Environment

- The version of the calculator is Java 22

- IntelliJ IDEA Ultimate was my IDE of choice

# 3 How to Build/Import your Project

1. Clone the repo from github
2. Open the cloned folder using IntelliJ
3. Open the calculator folder and look for the Evaluator sub folder
4. Open either Evaluator or EvaluatorUI and press the green play button

# 4 How to Run your Project

- (Assuming you already cloned the folder) open command prompt or powershell
- cd into the location of your project
- use the provided commands in the github repo

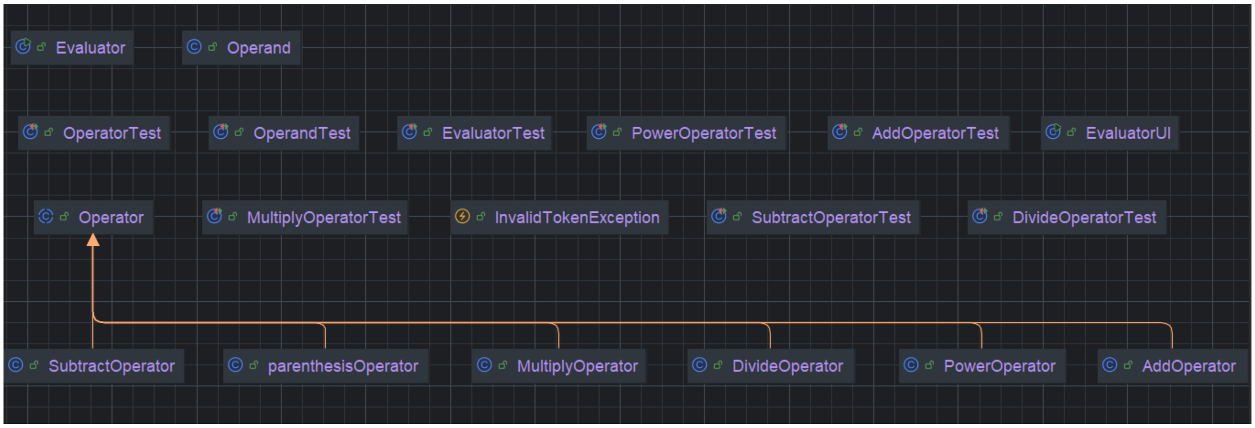
# 5 Assumption Made

- I assumed this project was going to be a difficult lengthy process, but I was extremely wrong and it turns out I already know most of the required knowledge.

# 6 Implementation Discussion

- Since most of the code was already provided, we just had to implement the rest of the code using the four pillars of object oriented programming. We mainly used abstraction and inheritance for this project. With the use of abstraction, we were able to inherit methods for all of the operator subclasses.

## 6.1 Class Diagram



# 7 Project Reflection

- At the start of this project, I was so lost and did not know where to start. I was convinced that I was going to drop the class within the first week, but after attending the first week of lecture, I was able to obtain a general understanding of what I need to do and where to start. After that, I reviewed old notes from CSC220 and refreshed my mind on Java. Overall, I think this project was very useful as a refresher even though I struggled. This project helped me prepare for the project two.

## 8 Project Conclusion/Results

- At the end, I was able to pass all test cases and I was able to provide the GUI with full functionality.