CSC 413 Project Documentation

Summer 2020

Tin Thu Zar Aye

920615641

CSC413-02

<https://github.com/csc413-02-SU2020/csc413-p1-TinThu>

Table of Contents

[1 Introduction 3](#_Toc522827688)

[1.1 Project Overview 3](#_Toc522827689)

[1.2 Technical Overview 3](#_Toc522827690)

[1.3 Summary of Work Completed 3](#_Toc522827691)

[2 Development Environment 3](#_Toc522827692)

[3 How to Build/Import your Project 3](#_Toc522827693)

[4 How to Run your Project](#_Toc522827694) 4

[5 Assumption Made 4](#_Toc522827695)

[6 Implementation Discussion 5](#_Toc522827696)

[6.1 Class Diagram 5](#_Toc522827697)

[7 Project Reflection 6](#_Toc522827698)

[8 Project Conclusion/Results 6](#_Toc522827699)

# Introduction

## Project Overview

* This project is called “Expression Evaluation and Calculator GUI”. Basically, we created the simple calculator which can handle the sign “+,-,\*,/,^ and () and also we can do the simple calculation. There are also including the clear button and backspace button in the calculator. We have to create to be handle the number and operators which are “+,-,\*,/,^,() sign as their precedence order to get the correct answer. We also created the manual calculator to be able to calculate by clicking each number and operator to get answer.

## Technical Overview

* In this project, we have to implement the Operand class, Operator class, Evaluator class and EvaluatorUI class. In the Operand class, we have to check the operand should be a number even though the user input as a String. Operand class and Operator classes are related when we do the calculation for the 2 Operands. In the Operator class, there is 2 abstract functions which are “priority function” and “execute function”. For the abstract “priority function”, we have to create the sub-classes to handle the priority of the operator such as “+,-,\*,/,^ and () , for example, + sign and – sign are priority 1, \* sing and / sign are priority 2 and so on. For the abstract “execute function”, we also need to create the sub-class to make all the calculation with 2operands.
* When we move to the Evaluator class, we have to create the 2 stacks, one for the Operand class and another one is for the Operator class. At first, we need to scan the operand and push to the operandStack then check the operator. If we see the openParenthesis we need to push to the operatorStack and if we see the closeParenthesis, we need to do the calculation until we see the openParenthesis. if we don’t see both openParenthesis and closeParenthesis, we just need to process an Operator.
* After the Evaluator class completely worked, we need to implement the EvaluatorUI class by using the Evaluator class. In the Evaluator class, we have to create the calculator to calculate the manually by using the evaluator class. When we press the “=” sign, the correct answer should be pop out on the calculator screen.

## Summary of Work Completed

* There are 4 main classes which are Operand class, Operator class, Evaluator class and EvaluatorUI class. There are also including the test classes which we can test after each class is completely implement. We also need to complete step by step, for example, first, we have to implement the Operand class; second the Operator class; third, the Evaluator class and then finally we have to implement the EvaluatorUI class by reusing the Evaluator class.

# Development Environment

* I am using the IntelliJ IDE version – 2019.3.5. Build #IC-193.7288.26

# How to Build/Import your Project

* I am importing the project. I copied the link from the GitHub repository and clone in my computer desktop then import to the intelliJ.

# How to Run your Project

* My code can handle the +,-,\*,/,^ and (). For example if you run “2+3” or “(2+3)”, there is no error for the () too.

# Assumption Made

* At first, I tried to implement the Operand class to be able to run the OperandTest.
* Second, I did implement the Operator class which include 2 abstract classes to make the sub-classes to be able to run the OperatorTest.
* Third, I needed to implement the Evaluator class by using both Operand class and Operator classes.
* Finally, I need to reuse the Evaluator class, I have to complete the “actionPerformed()” class where is from the EveluatorUI class.

# Implementation Discussion

## Class Diagram

|  |
| --- |
| **EvaluatorUI** |

|  |
| --- |
| **Evaluator** |

|  |
| --- |
| **Sub-classes for the abstract classes** |
| -AddOperator |
| -SubtractOperator |
| -MultiplyOperator |
| -DivideOperator |
| -PowerOperator |
| -LeftParenthesisOperator |
| -RightParenthesisOperator |

|  |
| --- |
| **Operand** |
| -public Operand(string token); |
| -public Operand(int value); |
| -public int getValue(); |
| -public static boolen check(String token); |

|  |
| --- |
| **Operator Class** |
| -public abstract int priority(); |
| -public abstract Operand execute(Operand operandOne, Operand operandTwo); |
| -public static Operator getOperator(String token); |
| -puclic static boolean check(String token); |

# Project Reflection

* After I watched the video for the introduction of the project, I need to read the whole assignment question and took time to understand fully the requirement. As the video said, I did the implementation for the Operand class first and then move to the Operator class. I had to figure it out how to create the sub-class for the 2 abstract classes from the Operator class. I think the Evaluator class is the most difficult part for this project. I needed basically 3 days to be done just for the Evaluator class. Slack is also helping me a lot to have done the project.

# Project Conclusion/Results

* To conclude, I think my project is perfectly working. As a result, I got the correct answer for the “+,-,\*,/,^,()”.