Assignment 01 Documentation

Name : Gio Jung
Student ID : 917160535
Class : Summer 2023, CSC - 413.02
Repositories Link : https://github.com/csc413-SFSU-Souza/csc413-p1-GioJung97

Introduction

Project Overview

This is a documentation regarding the project called "Calculator". The project requires fundamental skills and background of Java in order to run the program properly. The basic frame of the code of the project were given and there were several blank of the code to fill in. After editing the code, the program runs same as the normal calculator we think. When a user input an expression, the way the program evaluate the expression is exactly same as how we evaluate. Of course, if a user input invalid expression, it catches the expression and give an error message to a user. The great aspect of this program is that a user can input an expression all at once, it will evaluate and give an answer which is not like other simple normal calculator.

Technical Overview

This is a documentation regarding the project called "Calculator". The project requires the Data structure skill in order to make the code efficient and prevent the time complexity. The basic frame of the code of the project were given and there were several classes to work on. There are Operand, Operator, Evaluator, and EvaluatorUI classes. Basically, Operand and Operator classes check if the user typed valid expression. Evaluator class evaluates the expression correctly and lastly EvaluatorUI class builds the overall frame of the program which is the calculator based on Operand, Operator, and Evaluator classes. After the codes are completed, the program runs as a calculator. Whenever a user clicks the button of the calculator to type an input expression, the program stores the data of the expression as a String and evaluate the expression and display the result on the calculator.

Summary of Work Completed

I have completed the work to run the program properly. I tried to write the code as efficient as possible. Regarding the Operator classes, several child operator classes had to be created so that the code could read the operator of the input expression and evaluate the expression. Additionally, open and close parenthesis are also set as one of the operators so that while an input expression is being iterated, open and close parenthesis can be also read and stored. To check if the expression is valid, I used matches method to filter the invalid expression in each Operand and Operator class. Meanwhile, I used Hashmap and switch case statement for each operators to modify and maintain easier with extra code if there is. In the Evaluator class, I only let the open parenthesis be pushed in the operator stack, so whenever close parenthesis is being read, it evaluates the expression inside of the parenthesis and pop the open parenthesis which prevents one step of pushing close parenthesis in the operator stack. Of course, I added some extra conditions of if statements to filter the each expression. Lastly, in the EvaluatorUI class, I set the text field alignment to the Right. On the action performance part, I created String object to store the input expression and I used if statement to split the case in which a user clicks “C”, “CE”, “=”, and other buttons. Regarding the function of “CE”, I created new method which erase last digit of the string so whenever a user clicks “CE” button, the last character of the input expression is deleted.

Development Environment

Version of Java Used : openjdk 20.0.1

IDE used : IntelliJ

How to build or import my game in the IDE I used

IntelliJ is one of the IDE (Integrated Development Environment) program for Java and other programming languages. I personally think that it is very convenient for new users to get used to coding. However, when you try to build a new project, you can simply make a new project. Then it will automatically create several files such as "src" files or Java Libraries (accessible to use). In the "src", you can create new file such as "class", "Interface", or "enum" which most of the coding are taken a place. On the other hand, when you try to import my game in the IDE, you can use git command. You can open command or terminal and type "git clone https://github.com/csc413-SFSU-Souza/csc413-p1-GioJung97". Then the project file will automatically imported to your computer. Now, you can simply open the project by IntelliJ.

How to run my project

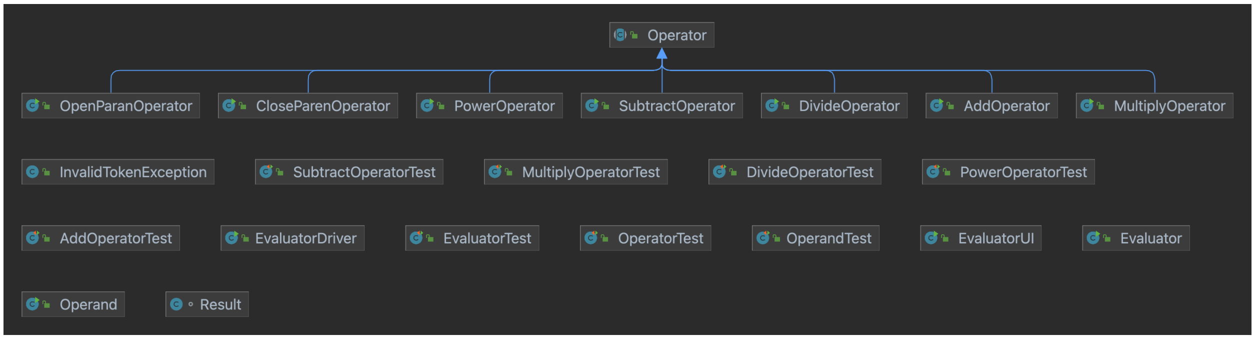
To run my project, you need to carefully choose which class you would like to compile. Optimally, if you want to run calculator, you need to choose "EvaluatorUI" class and run otherwise, the project will either run inside of IntelliJ, or won't work. After you run "EvaluatorUI", new file will be opened which is a calculator.

Assumptions Made when designing and implementing my project

While I was designing and implementing my project, I made several assumptions. Since my code doesn't work for negative numbers with input expression, I assumed the result for negative input numbers would been displayed as an error. Also, when I saw a given code in Evaluator class, what grabbed my attention was Stack. So, I assumed that the given code would push each operand and operator into each stack and pop after the calculation. Lastly, unlike other simple calculator (whenever you press operator buttons, the calculator displays nothing until you input the second operand), my code is available to use parenthesis, so I assumed that my calculator would allow a user to input all the expression in once and get the result.

Implementation Discussion

When I saw the given code at first, one thing I had on my mind was the child classes of Operator. Operator class is abstracted and there are five operators in this project, and two extra for each parenthesis. I could use interface instead, but with the given code, I designed each operators (+-*/^()) be inherited to Operator class since they are all part of operator class.



Project reflection

When I first saw the given code, I was overwhelmed because it was my first time making a project. So, I tried to approach the task incrementally and tackle it little by little, which helped me save time while working. Evaluator class was the main part which took a lot of time and concentration of this project. There might be a better way of putting open and close parenthesis, but I made them as child classes of Operator class to evaluate easier. On the EvaluatorUI class, one thing I wanted to fix is that while evaluating the input expression, when I tried to input wrong expression with multiple operators, the text field doesn't show the error message because of empty stack exception. The calculator works totally fine, but I just wanted to make it better, so if I gain more knowledge, I might get back to this project and play with UI part.

Project Conclusion and Results

Overall, this project made me to use the background of data structure which helped me to get into deeper level of Java. It was my first interaction with this size of project which might be the first step to be a software engineer though, I still need to learn a lot more than I think. As far as I know, this calculator program works properly which is the first goal on this project and after I have more experience, I will be able to consider more on efficiency of the code.