Assignment 1: Expression Evaluator and Calculator GUI

CSC 413 Summer 2019
Student name: Tsun Ming Lee
ID: 918541473

Github Link:
https://github.com/csc413-02-summer2019/csc413-p1-hkmatthew711

1. Introduction

1.1 Project Overview

This project is a simple calculator that can display basic mathematical expressions and perform simple algorithm works with addition, subtraction, multiplication and division.

1.2 Technical Overview

This project uses stack to store up the operators and operands. The single public method in Evaluator class takes a single String parameter to represent a mathematical expression, and it evaluates the expression then returns the result.

1.3 Summary of Work Completed

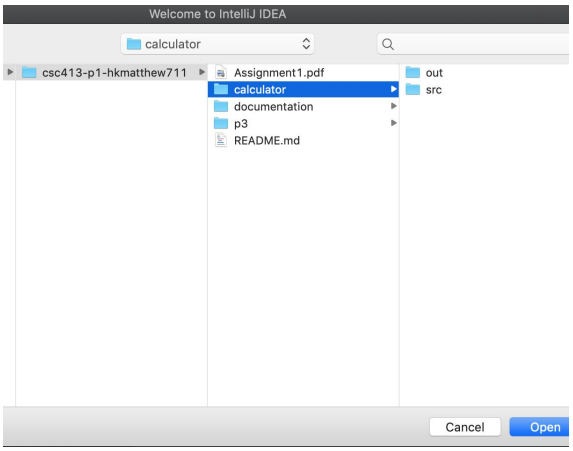
In this project, we learned how to use hash map, implement various methods in Evaluator class and abstract Operator class, and set up the UI for this project. Eventually, this project performs just like a calculator we use everyday with simple arithmetic.

2. Development Environment

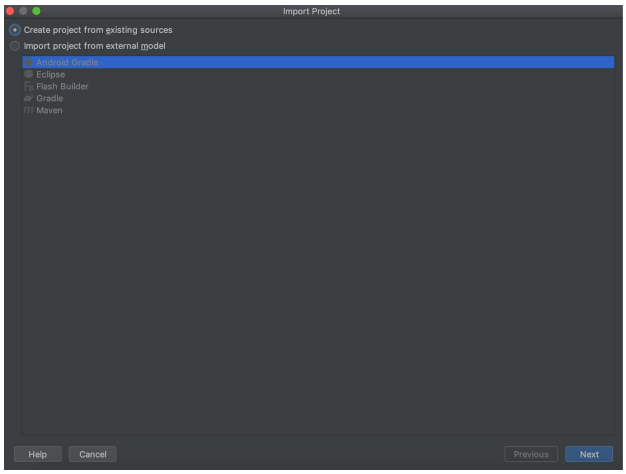
For this project, I use IntelliJ IDEA CE to compile it, java version "2019.1.3".

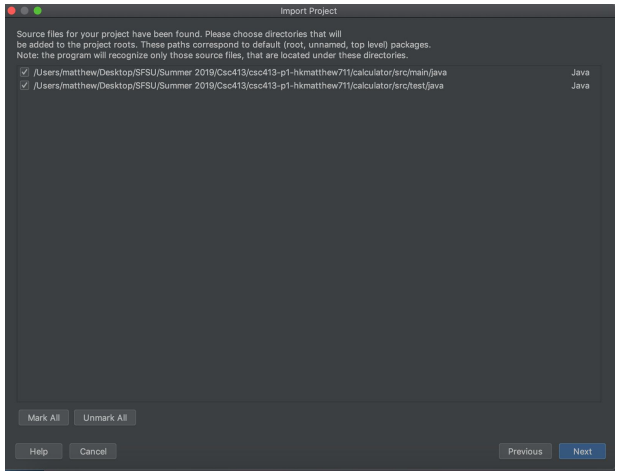
3. How to Build/Import your Project

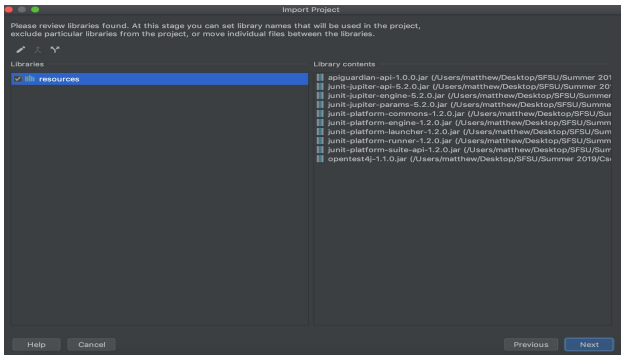
The environment for the project is IntelliJ. First, open IntelliJ. Then click the Import Project on the welcome screen. Find the project location and click open.

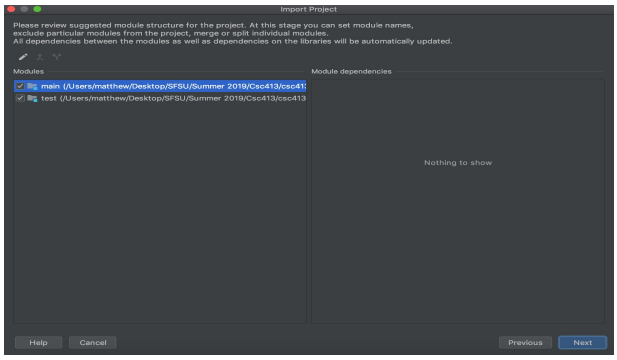


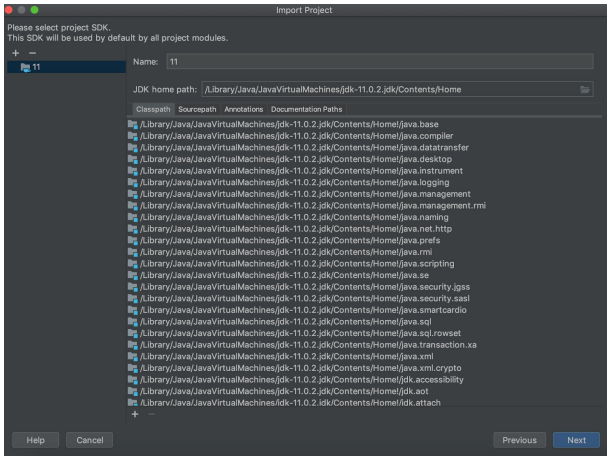
Then following the screen and keep click on Next, until Finish.

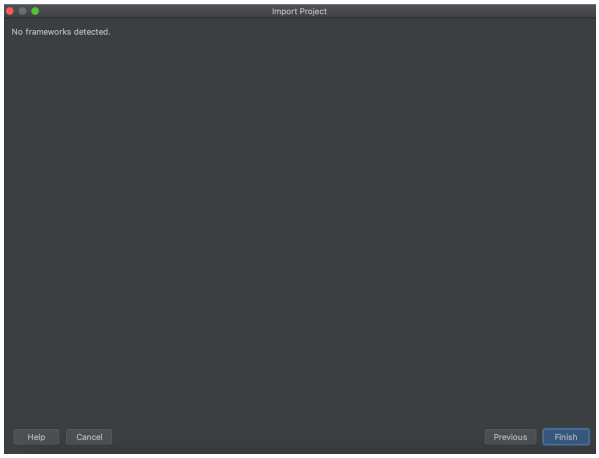






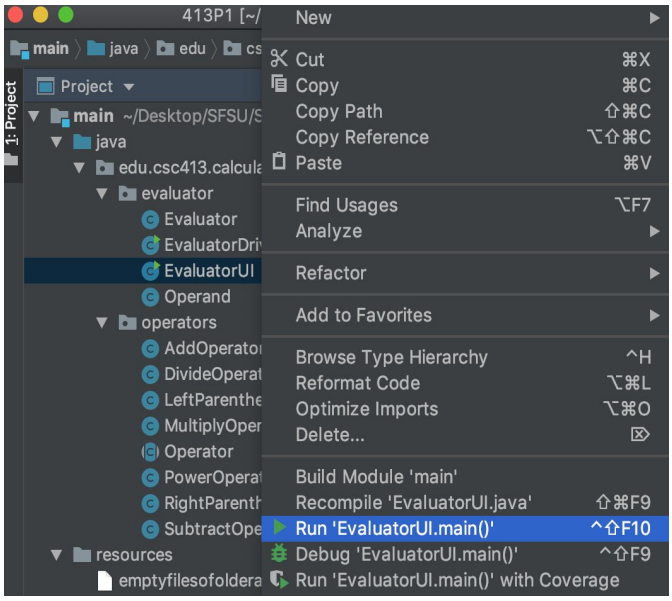


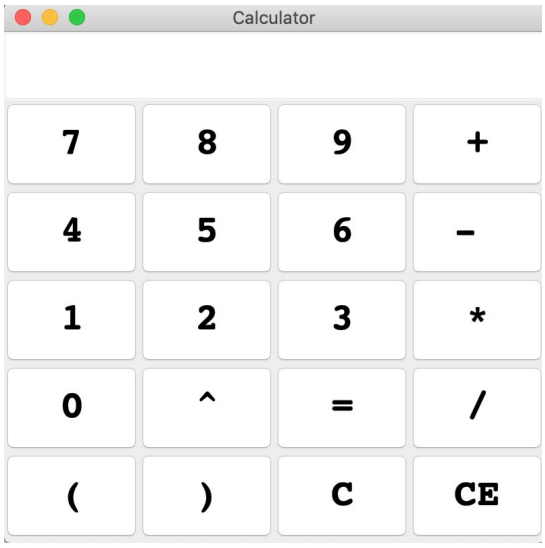




4. How to Run your Project

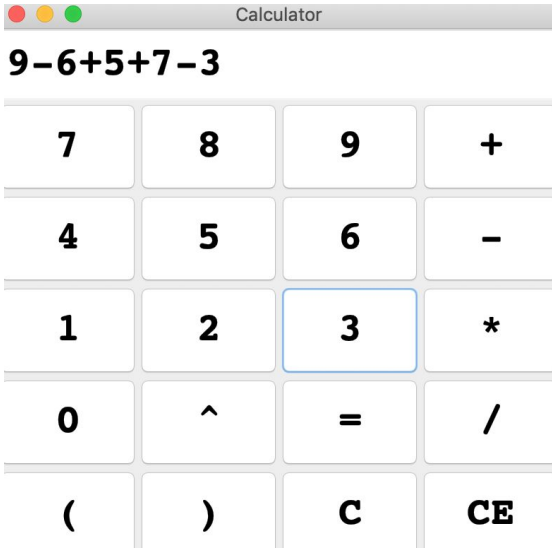
After importing project, it comes to the screen that shows all the project files on the left side, and double click on the file named EvaluatorUI . Then, it shows all the coding done on the file. Right click the EvaluatorUI and choose Run ‘EvaluatorUI.main()’ to run this project. Then the calculator will pop up.



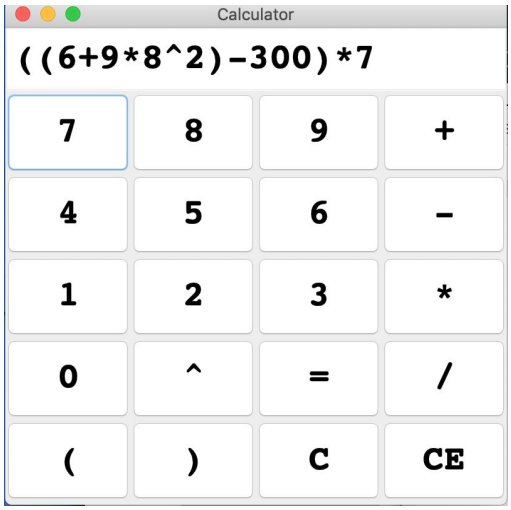


Then we can do some calculations:

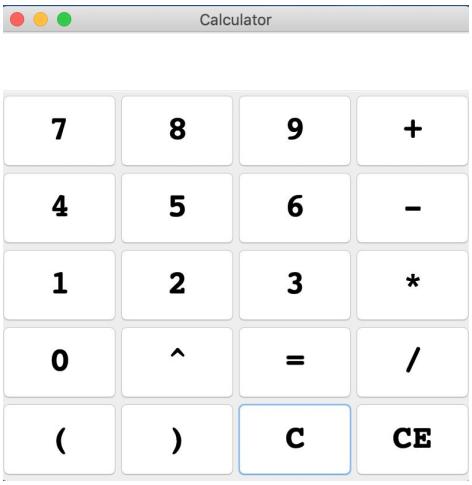
9 - 6 + 5 + 7 - 3 = 12



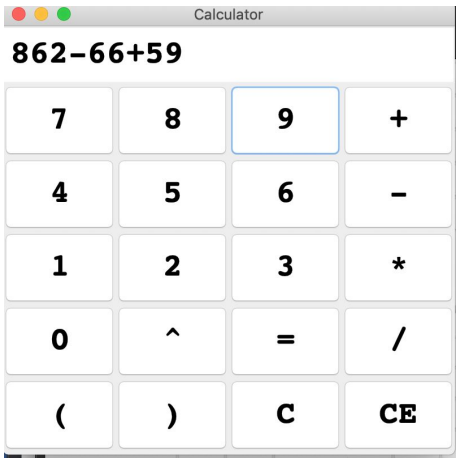
((6 + 9 * 8^2) - 300) * 7 = 1974

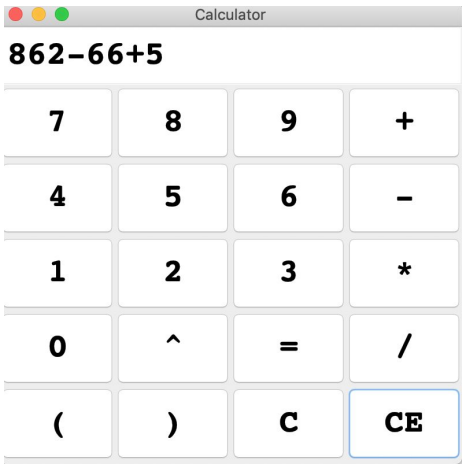


Click button C to clear all the things



Click button CE to clear the last input





5. Assumptions Made

-- In this project, I assume that all the input should be integers and can not be negative numbers. This project can handle addition, subtraction, multiplication, division, and exponents operators.

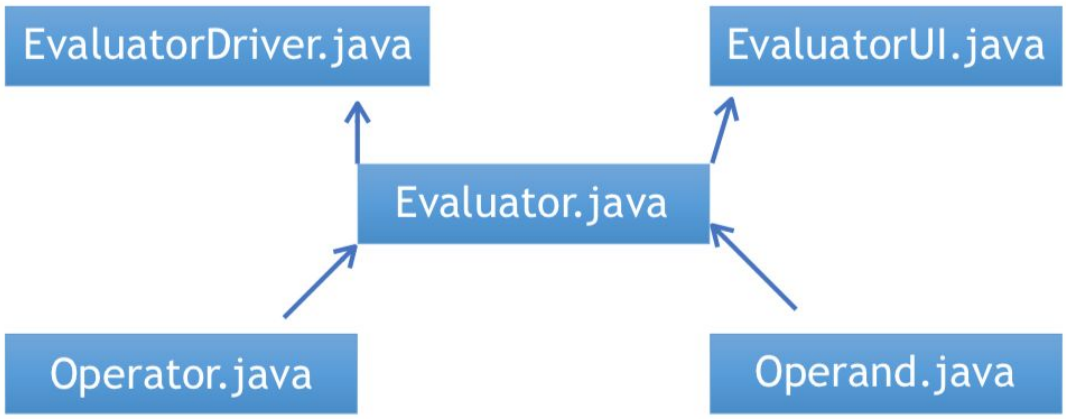
-- I first started with a simple algorithm and then gradually tested the complex algorithm like “(”, “)” algorithm.

6. Implementation Discussion

6.1 Class Diagram

There are 5 files in this project: Operand.java, Operator.java, Evaluator.java, EvaluatorUI.java, and EvaluatorDriver.java

I needed to implement the first 4 files, except EvaluatorDriver, which is just a test file. Here is a diagram indicating the relationship among the five files:



7. Project Reflection

When I started to work in this project, there were many java-related concept that I almost forgot since I had left working with Java after a semester of C courses. I picked myself up by studying online and asking other classmates for coding suggestions. It also reminded me that there are so much to learn in Java, and I am still learning to become better. As I was working along, I encountered some errors. One of the errors that I found difficult to solve was reading the parentheses. Whenever I input an expression with parentheses, it always flagged error messages, such as empty stacks

and invalid token. I managed to solve this issue by adding the parentheses symbols in the delimiter string.

8. Project Conclusion/Results

In conclusion, the project performs fine with basic algorithm with addition, subtraction, multiplication, and division. However, it does not handle input of negative integer. If project evaluates an expression with negative integers, it gives out a bunch of error messages because the project does not support changing sign of an integer. Moreover, the project does not handle fraction because the result always displays the whole number even though it is followed by some decimals.