Computer Science Department
San Francisco State University
CSC 413
Spring 2018

Documentation
Assignment 1 - Expression Evaluator and Calculator GUI

Student Name: Junjie Lu

GitHub Repository: https://github.com/csc413-03-sp18/csc413-p1-Jangey

Introduction and Overview:

This is a program come with two functions:
1. An object that evaluates mathematical expressions

- The calculate will be evaluates the mathematical expressions and show users the final answer for the mathematical expressions.

The Example for evaluates mathematical expression:

Addition Test:

- 1+2*3
- ((1+2) *3)
- (1+2) *3
- 12+34-45*56/3
- (12+34-45) *56/3
- 3*4+6-32/5

Basic Test:

- 2+3*4
- (2+3) *4

Hard Test:

- 2+3-5*((2-3)*2-5*2+3*(2-3-5-5*6)+4/2)*2-9

Parentheses Test:

- ((1+2)*3)
- (1+2)*3

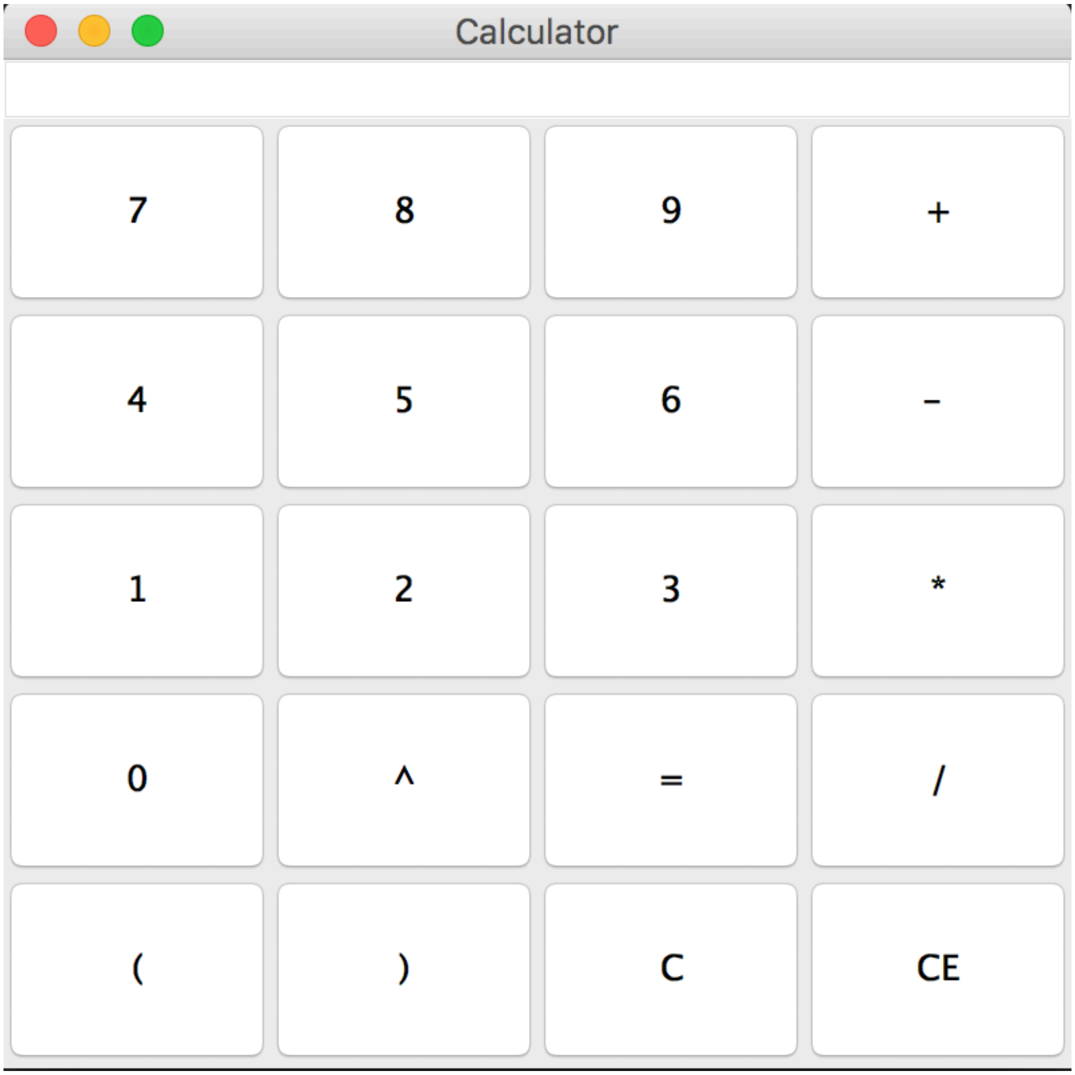
2. A calculator GUI

This is the GUI for the calculator with the Input/Output text field and 20 buttons.

20 Buttons:

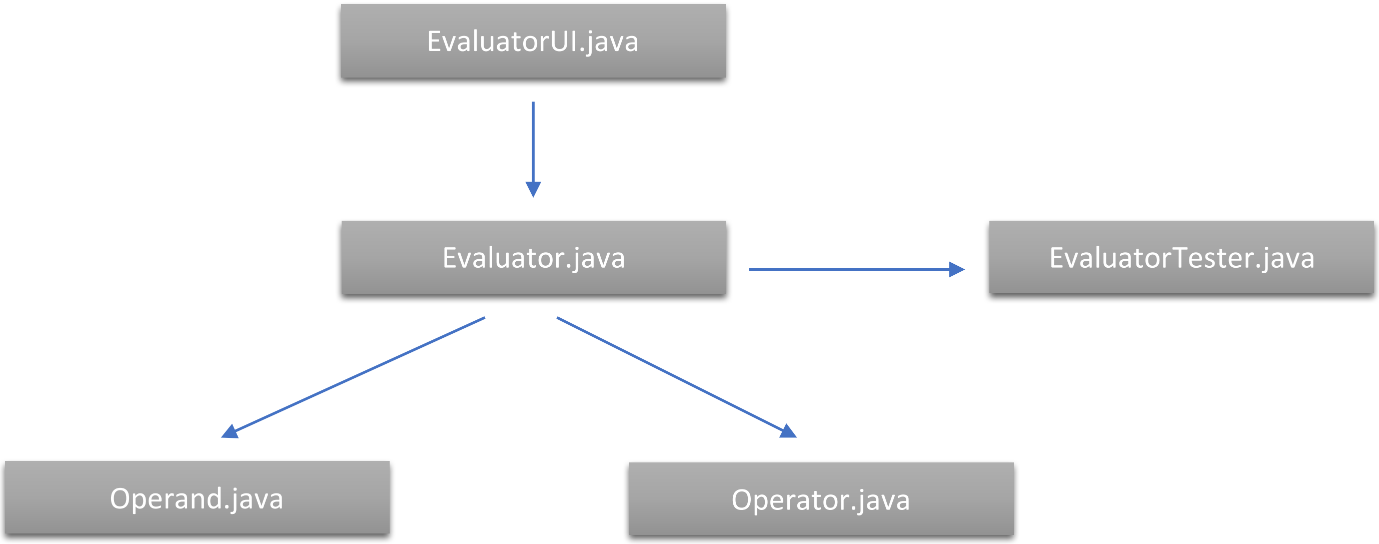
“0 – 9” Numbers Operator
“-” Subtraction Operator
“/” Division Operator
“(” Open Parenthesis
“=” Equal Operator
“CE” Clear Current Input Operator

“+” Addition Operator
“*” Multiply Operator
“^” Power Operator
“)” Close Parenthesis
“C” Clear All Operator

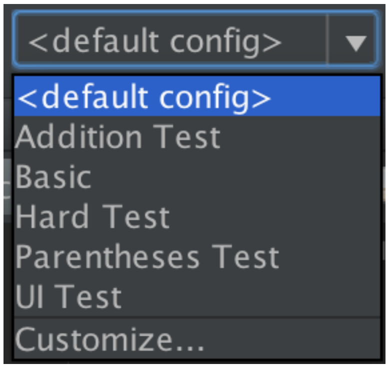


The documents inside the program:

The Program is using JAVA language, I'm using NETBEAN as an IDE.
The Program including Evaluator.java, EvaluatorTester.java, EvaluatorUI.java, Operand.java, Operator.java.
The class diagrams hierarchy for this program is show as below:



When users using NetBeans, on the top bar, the users can choose the test to run the program. For Addition test, basic, Hard Test and Parentheses Test are evaluating mathematics expressions. And UI test is for running the Calculator GUI program. Users can choose the test they want to test the program to compile and execute the program.



Algorithm:

The program using HashMap to order the priority for each operator inside the expression, the lower number have prime priority. For the mathematic expression algorithm, the priority is show as the table below:

Operator	**Priority**
(,)	-1
#	0
!	1
+, -	2
***, /**	3
^	4

Assumptions:

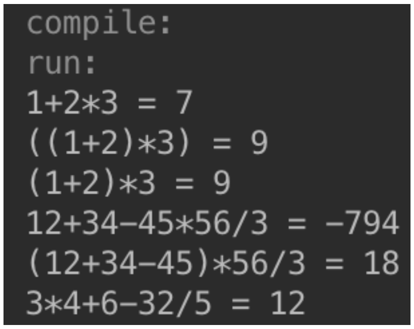
In this program, I start from the simple addition and subtraction operator, and using try to run my program with the basic test and addition test. After my code passed the basic test, I started work on multiply operator and division operator and then work on the parentheses operators. Last, I test my code for each test and make sure the code is working perfect for all of tests.

Result:

For the expression evaluator, the program will be show the completely expression with the final answer. Also, the program can be calculator multiply expression at the same time.

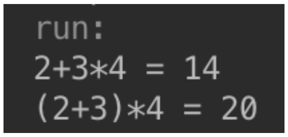
Here is the result for the expression evaluator test:

Addition Test:



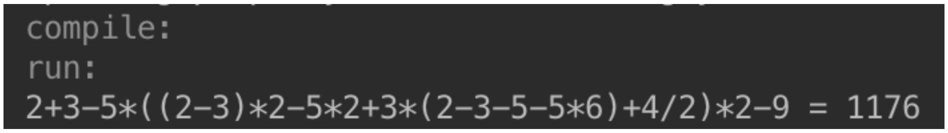
```
compile:
run:
1+2*3 = 7
((1+2)*3) = 9
(1+2)*3 = 9
12+34-45*56/3 = -794
(12+34-45)*56/3 = 18
3*4+6-32/5 = 12
```

Basic Test:



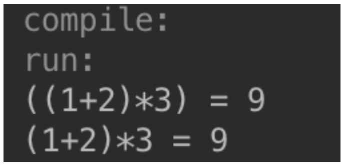
```
run:
2+3*4 = 14
(2+3)*4 = 20
```

Hard Test:



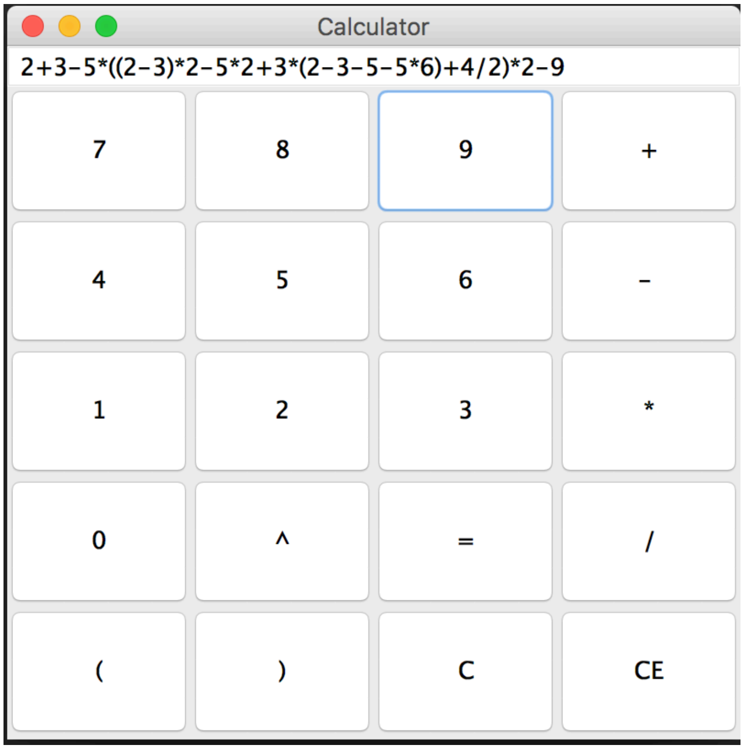
```
compile:
run:
2+3-5*((2-3)*2-5*2+3*(2-3-5-5*6)+4/2)*2-9 = 1176
```

Parentheses Test:



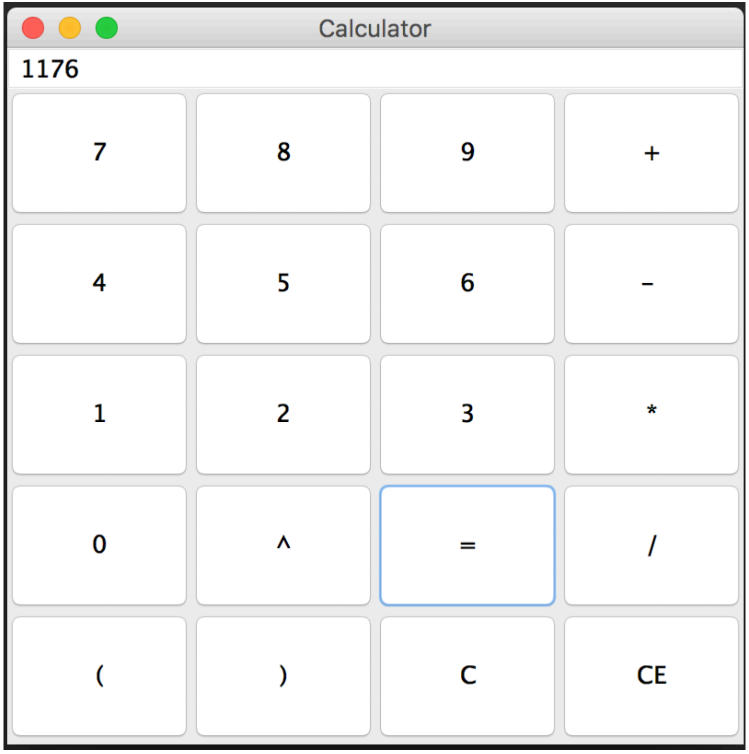
```
compile:
run:
((1+2)*3) = 9
(1+2)*3 = 9
```

For the Calculator GUI test, I use the hard test for example input, and it is a valid expression equation:


Calculator

2+3−5*((2−3)*2−5*2+3*(2−3−5−5*6)+4/2)*2−9

After I click "=" button, the calculator will show the result for the input expression:


Calculator

1176

Conclusions:

Expression Evaluator:
The program is using JAVA HashMap to get the priority for the expression. I had review how is the HashMap working and priority ordering. The hard part is the algorithm for the expression, because it has to think multiple way for the expression, for example if the expression has more than one parentheses. And think about what is the rest operator or number left inside the expression. I have to test multiple times to test the program is working for each situation.

Calculator GUI:
I learn from this GUI is how to make each button working but also it will store the total input together before the expression execute. The hard part for this program is the two new buttons “C” and “CE”, because those two buttons do two different things. “C” is clear everything inside the text field, but the “CE” only clear last digital inputs. Therefore, I have to create two variables for “expression” and “new input”. It is the way to make sure the “CE” function is working only for the “new input”.