**Team Project 2: Infix Expression Parser**

Rawan Alhachami, Ollie Peel

CPT-287-R82 Intro to Data Struct. With Java

Dr. Wang

July 12, 2024

**System Design**

Our system starts in the main() method of the Driver class which uses the ExpressionParser.evalExpression() method to parse and evaluate each expression from an input file. The evalExpression() method uses the evaluate() and precedence() methods in the same class which evaluate the expression from stacks based on the operator and assign precedence values to the operators.

The data structures our system used were stacks and an array list. The array list was used in the Driver class to store each expression from the input file. Stacks were used in the evalExpression() and evaluate() methods of the ExpressionParser class. The evalExpression() method used stacks to separately store the integers and operators of the expression and the evaluate() method evaluated an expression made of an operator and two integers from the same stacks previously mentioned.

**UML Diagram**

**A diagram of a stacking process

Description automatically generated with medium confidence**

**Test Cases**

**Case 1:**

**Input: 1 + 3 > 2**

**Output: 1**

**Case 2:**

**Input: (1+2)\*3**

**Output: 9**

**Contributions**

Rawan Alhachami: Finished the evalExpression method and edited line appearance in precedence() method in the ExpressionParser class, added code to Driver class to handle exceptions for invalid expressions and errors, edited BufferedReader method a little bit for clarity. Worked on test cases and improvements of project report.

Ollie Peel: Created and added code to the Driver class which reads expressions from an input file. Created and added code to the evaluate() and precedence() methods of the ExpressionParser class as well as started code for the evalExpression() method in the same class. Worked on the UML diagram and system design sections of the project report.

**Improvements**

In the future, the parser section of the program could possibly be redesigned to handle different data types like variables.

The error messages could also be more specific or detailed so that the user can know exactly what was incorrect and why.

There could also be a feature added for the user to be able to access previous calculations.