**Team Project 2A**

**Infix Expression Parser**

Andrew Lange, Mason Johnson, Matt Hays, Staven Valet

Design Explanation

In the Main method, create a buffered reader using a FileReader to open “test.txt”.

Reads the file line by line creating an object of InfixExpression that parses the stored line, then output the results of the expression.

It will throw a “FileNotFoundException” if the file is not located and an IOException if there is an error while handling the file.

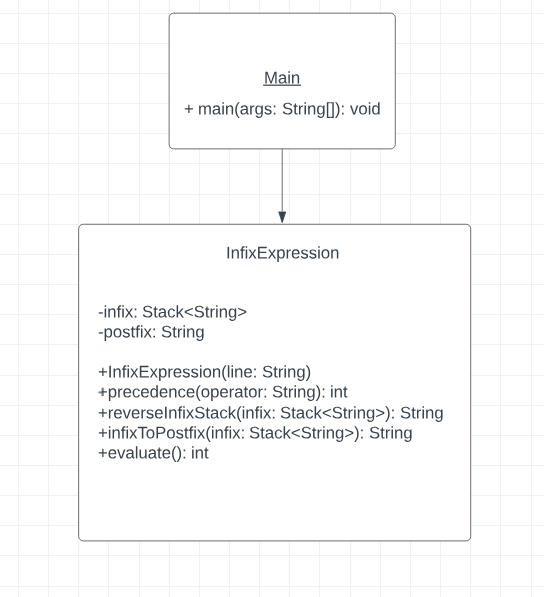
In the InfixExpression class the constructor gets passed the expression store as string variable named “line” and parses it into a stack of strings, tokenizing each operand and operator in the initial expression. Inside the constructor it calls the reverseInfixStack() method to make the beginning of the expression appear on top of the stack for easier processing.

Afterwards, it calls infixToPostfix() and converts the infix stack to a postfix expression stored as a string. Inside infixToPostfx() it arranges the operators in corresponding order based on precedence determined by the precedence() method.

After the expression has been converted to postfix the evaluate() method is called to return the result of the parsed expression.

Arithmetic expressions return the value that was calculated and Boolean and logical operator expressions return 0 (False) or 1 (True).

UML class diagram



Test Cases

Line 1: 20 % 2 <= 4 + 4

Line 2: 23 + 4 \* 8

Result 1: 1 (True)

Result 2: 55

Contributions

Andrew Lange – InfixExpression Cleanup, UML Class Diagram

Mason Johnson –

Matt Hays – Designing and implementing Main method and InfixExpression class, and project outline documentation.

Steven Valet –