**CPT-287 Team Project 2A**

**Infix Expression Parser**

**-Team Members-**

Tu Nguyen

Kyle Molitor

Joel Estes

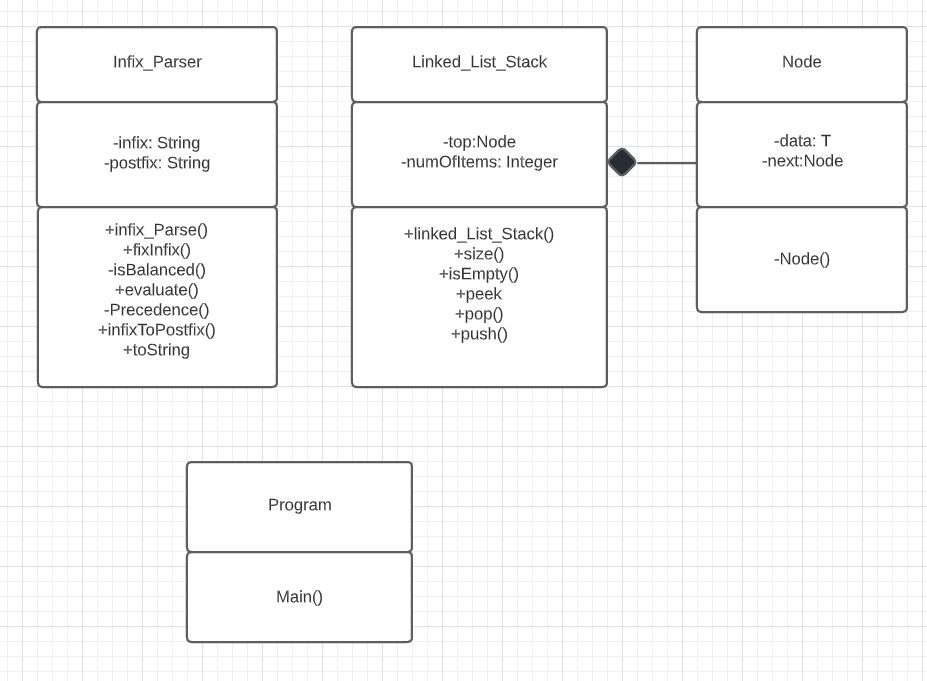
Josh Vander Veld

**-System Design-**

*Data Structures*

This system implements a Linked list Stack and Built in Java stack data structure. The Linked List Stack implements a Linked list of nodes used to store data as well as functions for checking and modifying the stack. The Infix Parser is used to parse any given expression read in from an input file. Infix parser contains functions for parsing the input expression such as, isBalanced, evaluate, toString ext. The program class contains all the driver code to read in an input file and output the result.

*UML Class Diagram*

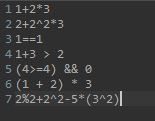


**-Test Cases-**

***Test Case 1:***

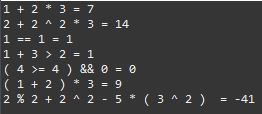
*Input File*

Input file contains expressions.

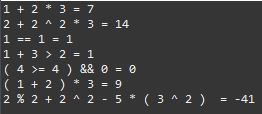
**

*Output Console*

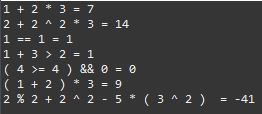
Program successfully reads and gives correct answer.



Expected output:



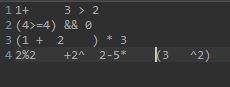
Actual output:



***Test Case 2:***

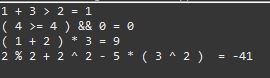
*User Input*

Input file expressions contain inconsistent spaces.

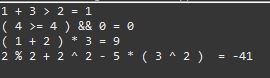


*Output Console*

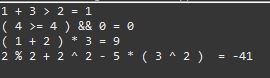
Program successfully reads and corrects spacing.



Expected output:



Actual output:



**-Team Contributions-**

**Tu**: Contributed to the Program class for reading an input file and outputting results to console. Contributed to Infix\_Parser class. Responsible for writing the constructors and the following methods, fixInFix and toString. Modified IsBalanced method, evaluate method, and infixToPostfix method.

**Kyle**: Contributed to Program class for reading input from a text file. Contributed to the Infix\_Parser class and modified the precedence function to support all operators needed in the assignment. Responsible for creating and writing this report as well as contributing to both test cases. Created and maintained GitHub project repository.

**Joel**: Created UML Class diagram.

**Josh**: Contributed to test cases. Helped clean up code in Infix Parser class.

**-Future System Improvements-**

This system could be improved to support more operators and cases. The infix parser could contain more methods and more support for all variations of expressions such as converting to and from infix, postfix, prefix expressions.

With this system we evaluate the expression by changing it from infix to postfix, we could make it more efficient by using a binary search tree to store values. We could also add functionality for more advanced expressions with non-number values such as PI, sine, cosine, tangent. This program could be turned into a website that would allow you to enter an expression and choose if you want to covert and solve the expression.