**CPT-281 Team Project 3A: Binary Tree Infix Expression Parser**

Contributors: Athul Jaishankar, Timothy Huffman, Kathleen Dunn, Tyler Blackmore

Project Summary:

This project is an Binary tree Infix expression parser system that helps parse an infix expression that supports arithmetic and logical operators with specified precedencies. The system utilizes binary trees and stacks for efficient management of expression data.

Technical Requirements:

▪ The Binary Tree Infix expression parser system will support:

Operator Precedence Example

1) Power ( ‘^’ ) 7 2 ^ 8

2) Arithmetic ( ‘\*’, ‘/’, ‘%’ ) 6 6 \* 2

3) Arithmetic ( ‘+’, ‘-’ ) 5 6 - 2

4) Comparison ( ‘>’, ‘>=’, ‘<’, ‘<=’ ) 4 6 > 5

5) Equality Comparison ( ‘==’, ‘!=’ ) 3 6 != 5

6) Logical And ( ‘&&’ ) 2 6 > 5 && 4 > 5

7) Logical Or ( ‘| |’ ) 1 1 | | 0

▪ The infix expression parser is flexible with the given expressions. The user don’t need to worry about writing the spaces between operands and operators

▪ The file that keeps track of the infix expression is a plain text file. An original file input format is made based on this example:

((2 + 3) \* 4) - (5 \* (6 - 7))

(1 | | (0 && 1)) && (1^ ( 1 && 0 ))

(( 2 \*3) ^ 2 ) + ( 4\* 5) % 3

In the example above, each line stores a valid infix expression with appropriate suitable operators and operands.

**System Design:**

**Data Structures:**

**UML:**

A diagram of a computer program

Description automatically generated with medium confidence

**Test Cases:**

**Team Member Contributions:**

**Future Improvements:**