Compiler Design - Comprehensive Parser for Mathematical Expressions: From Input to 3-Address Code

-By Aditya Hegde

Problem Statement:

The project's goal is to create a robust parser for standard calculator numerical expressions, covering each major step of the compiler's process, from input to generating 3-address code. It aims to enable clarity and ease of expression evaluation.

Applications:

The developed parser has versatile applications in standard calculators across various devices that support 3-address code. It provides a foundation for creating calculator software capable of processing mathematical expressions efficiently.

Results:

The project delivers a well-documented script with the following features:

Input Preprocessing: The script efficiently cleans input expressions by removing whitespaces.

Tokenization and Classification: It tokenizes and classifies each character into variables, operators, or brackets, facilitating structured parsing.

Mathematical Validity Check: The script validates expressions for mathematical correctness, ensuring they meet mathematical rules and formatting standards.

Conversion to Postfix Notation: Expressions are converted to postfix notation, simplifying evaluation.

Abstract Syntax Tree Generation: An Abstract Syntax Tree (AST) is generated, providing a hierarchical representation of the expression's structure.

Visualization of the AST: The AST is presented in a readable format, aiding understanding and debugging.

3-Address Code Generation: The script generates 3-address code, an intermediate representation of the expression that can be further compiled to machine-level language.

Future Scope:

The project holds potential for expansion, with future enhancements such as:

Scientific Function Integration: Including functions like logarithms and trigonometry for advanced mathematical computations.

Extended Operator Support: Incorporating more operators to broaden the scope of calculations.

Boolean Expressions: Extending capabilities to handle boolean expressions for logical operations.

Programmable Calculation: Enabling users to create and execute custom calculations.

User Interface Development: Building a user-friendly interface for easy interaction.

The project's comprehensive capabilities provide a strong foundation for calculator software development and mathematical expression evaluation while allowing room for future growth and refinement.