**Mohammad Khan**

**Project 5**

**Design Document:**

The Infix Calculator is a program that evaluates arithmetic expressions in infix notation. It takes an infix expression as input, converts it to postfix notation, evaluates the postfix expression, and returns the result.

The program consists of a single class, **InfixCalculator**, which has private member functions for checking if the input expression is well-formed and has balanced brackets, and for converting an infix expression to postfix notation and evaluating a postfix expression. The public interface of the class consists of a constructor, a method for setting the input expression, and a method for evaluating the expression.

The **set\_expression()** method takes a string argument and checks if it is a well-formed infix expression with balanced brackets. If the input is valid, it sets the **infix\_expression** private member variable to the input expression and returns true. If the input is invalid, it returns false.

The **evaluate()** method calls the **infix\_to\_postfix()** method to convert the **infix\_expression** to postfix notation, and then calls the **evaluate\_postfix()** method to evaluate the postfix expression and return the result.

The **infix\_to\_postfix()** method uses a stack to convert an infix expression to postfix notation. It iterates through the input expression and pushes operands onto the output string and operators onto a stack. When a closing parenthesis is encountered, it pops operators off the stack and appends them to the output string until an opening parenthesis is encountered. When all the input characters have been processed, it pops any remaining operators off the stack and appends them to the output string.

The **evaluate\_postfix()** method evaluates a postfix expression using a stack. It iterates through the input string and pushes operands onto the stack and performs the corresponding operation when an operator is encountered.

**Test Data:**

Test Case 1: Input Expression: "2+3\*4" Expected Output: 14 Explanation: The expression should be evaluated as 2 + (3 \* 4) = 14.

Test Case 2: Input Expression: "((4+2)\*3)/2-1" Expected Output: 8 Explanation: The expression should be evaluated as (((4 + 2) \* 3) / 2) - 1 = 8.

Test Case 3: Input Expression: "5\*(2+3" Expected Output: Invalid expression! Please enter a valid infix expression. Explanation: The expression is not well-formed because it has an unbalanced parenthesis.

Test Case 4: Input Expression: "5^2" Expected Output: Invalid expression! Please enter a valid infix expression. Explanation: The expression is not well-formed because it contains an invalid character.

**User Document for Infix Calculator Program**

Introduction

This document provides information on how to use the Infix Calculator program. This program takes a user-provided infix expression, evaluates it, and returns the result. To run the program type g++ infix\_calculator.cpp -std=c++11.

Usage

Run the program.

Enter an infix expression when prompted.

Press "Enter" to submit the expression.

The program will evaluate the expression and return the result.

Repeat steps 2-4 to evaluate more expressions.

Type "exit" to quit the program.

Example:

Enter an infix expression (or type 'exit' to quit): 2+3\*4 The result of the expression is: 14. Enter an infix expression (or type 'exit' to quit): (8-2)\*(6+4)/2 The result of the expression is: 30 Enter an infix expression (or type 'exit' to quit): exit

Error Messages

If the user enters an invalid expression, the program will display the following error message:

Invalid expression! Please enter a valid infix expression.

Conclusion

The Infix Calculator program provides a simple and efficient way to evaluate arithmetic expressions. With this program, users can quickly evaluate complex expressions without the need for pen and paper.