Date: 07/08/2024

Experiment 2.4

AIM

To implement a calculator using LEX and YACC

ALGORITHM

1. Start
2. Create a lex file with following lexical rules:
    1. Include y.tab.h generated by YACC program.
    2. If input is a sequence of digits:
        1. Copy value to yylval.
        2. Return token IDENTIFIER.
    3. If input is an arithmetic operator or a newline character, return the first character in the input.
3. Create YACC to parse input as follows:
    1. The input consists of multiple lines.
    2. Each line contains an expression followed by a newline character.
    3. Each expression can be expanded as one of the following:
        1. expression + term

            In this case, the value of resultant expression is the sum of values of the first expression and the term.

        2. expression – term

            In this case, the value of resultant expression is the difference of values of the first expression and the term.

        3. term

            In this case, the value of the expression is that of the term.
    4. Each term can be expanded as one of the following:
        1. term * factor

            In this case, the value of resultant term is the product of values of the first term and the factor.

2. term / factor

   In this case, the value of resultant term is the difference of values of the first term and the factor.

3. factor

   In this case, the value of the term is that of the factor.

5. A factor can be expanded as follows:
   1. A factor can be an IDENTIFIER token. In this case, the value of the factor is the numerical value of the identifier.
   2. (expression)

      In this case, the value of the factor is that of the expression within the brackets.
6. In user code section:
   1. Define error handling.
   2. Define main function to call yyparse().

4. Use lex command to generate C program.
5. Use yacc to create y.tab,h and y.tab.c
6. Compile and run y.tab.c along with lex program
7. Stop

RESULT

Successfully implemented a calculator with YACC and LEX.