**Ex No: 5**
**Date:**

## RECOGNIZE AN ARITHMETIC EXPRESSION USING LEX AND YACC

### AIM:

To check whether the arithmetic expression using lex and yacc tool.

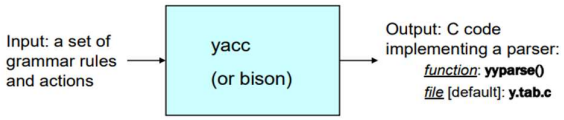
### ALGORITHM:

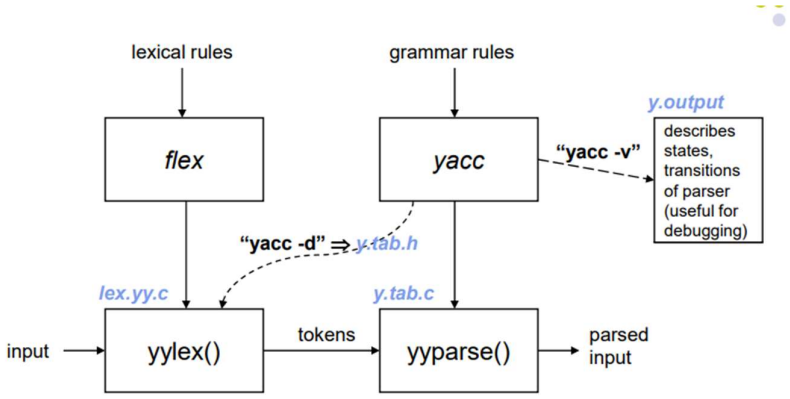
- Using the flex tool, create lex and yacc files.
- In the C include section define the header files required.
- In the rules section define the REGEX expressions along with proper definitions.
- In the user defined section define yywrap() function.
- Declare the yacc file inside it in the C definitions section declare the header files required along with an integer variable valid with value assigned as 1.
- In the Yacc declarations declare the format token num id op.
- In the grammar rules section if the starting string is followed by assigning operator or identifier or number or operator followed by a number or open parenthesis followed by an identifier. The x could be an operator followed by an identifier or operator or no operator then declare that as valid expressions by making the valid stay in 1 itself.
- In the user definition section if the valid is 0 print as Invalid expression in yyerror() and define the main function.

### LEX AND YACC WORKING :

Parser generator:

- Takes a specification for a context-free grammar.
- Produces code for a parser.






**PROGRAM:**

**validexp.l:**

```
%{
#include "y.tab.h"
%}

%%
[a-zA-Z_][a-zA-Z_0-9]*  { return id; }
[0-9]+                  { return num; }
[+/*]                   { return op; }
.                       { return yytext[0]; }
\n                      { return 0; }
%%

int yywrap() {
   return 1;
}
```

**validexp.y:**

```
 %{
   #include<stdio.h>
%}
%token NUMBER
%token VARIABLE

%left '+' '-'
%left '*' '/' '%'
%left '(' ')'
%%
```

```
S: VARIABLE'='E {
      printf("\nEntered arithmetic expression is Valid\n\n");
      return 0;
    }
E:E'+'E
 |E'-'E
 |E'*'E
 |E'/'E
 |E'%'E
 |'('E')'
 | NUMBER
 | VARIABLE
;
%%
void main()
{
  printf("\nEnter Any Arithmetic Expression which can have operations
Addition, Subtraction, Multiplication, Divison, Modulus and Round
brackets:\n");
  yyparse();
}
void yyerror()
{
  printf("\nEntered arithmetic expression is Invalid\n\n");
}
```

**OUTPUT:**

```
root@thamizh-HP-Pavilion-Laptop-15-eg2xxx:/home/thamizh/Desktop/CD_Record/exp6# lex 275_exp6.l
root@thamizh-HP-Pavilion-Laptop-15-eg2xxx:/home/thamizh/Desktop/CD_Record/exp6# yacc -d 275_exp6.y
root@thamizh-HP-Pavilion-Laptop-15-eg2xxx:/home/thamizh/Desktop/CD_Record/exp6# cc lex.yy.c y.tab.c
y.tab.c: In function ‘yyparse’:
y.tab.c:1031:16: warning: implicit declaration of function ‘yylex’ [-Wimplicit-function-declaration]
 1031 |       yychar = yylex ();
      |                ^~~~~
y.tab.c:1166:7: warning: implicit declaration of function ‘yyerror’; did you mean ‘yyerrok’? [-Wimplicit-function-declaration]
 1166 |       yyerror (YY_("syntax error"));
      |       ^~~~~~~
      |       yyerrok
root@thamizh-HP-Pavilion-Laptop-15-eg2xxx:/home/thamizh/Desktop/CD_Record/exp6# ./a.out

Enter the expression:
s=75;

Valid expression!
```

**RESULT:**