Exp 6 : YACC – VALID ARITHMETIC EXPRESSION

LEX

1. Start
2. Definitions Section - include an external declaration for \`yylval\`, which is often used to store the value associated with a token

3. Rules Section
    1. [0-9]+ : matches one or more digits and convert value of yylval to the integer representation of the matched text and returns the token type NUM
    2. [a-zA-Z][a-zA-Z0-9]* : matches an identifier and returns the token type ID
    3. [\t\n] : skips whitespaces
    4. . : matches any other character and returns it as a single character token

4. yywrap() indicate the end of input
5. Stop

YACC

1. Start
2. Token Declarations (%token) : NUM and ID
3. Operator Precedence Declarations (%left) : \`+\`, \`-\`, \`*\`, \`/\`, \`%\`
4. Grammar Rules Section

    E -> E+E | E-E | E*E |E/E { if($3 == 0){ yyerror() } } | E%E { if($3 == 0) { yyerror() }} | (E) | -NUM | NUM

5. yyerror() to handle
6. main() prompts the user to enter an arithmetic expression and calls yyparse() to parse it
7. Stop

Valid Arithmetic Expression (Lex)

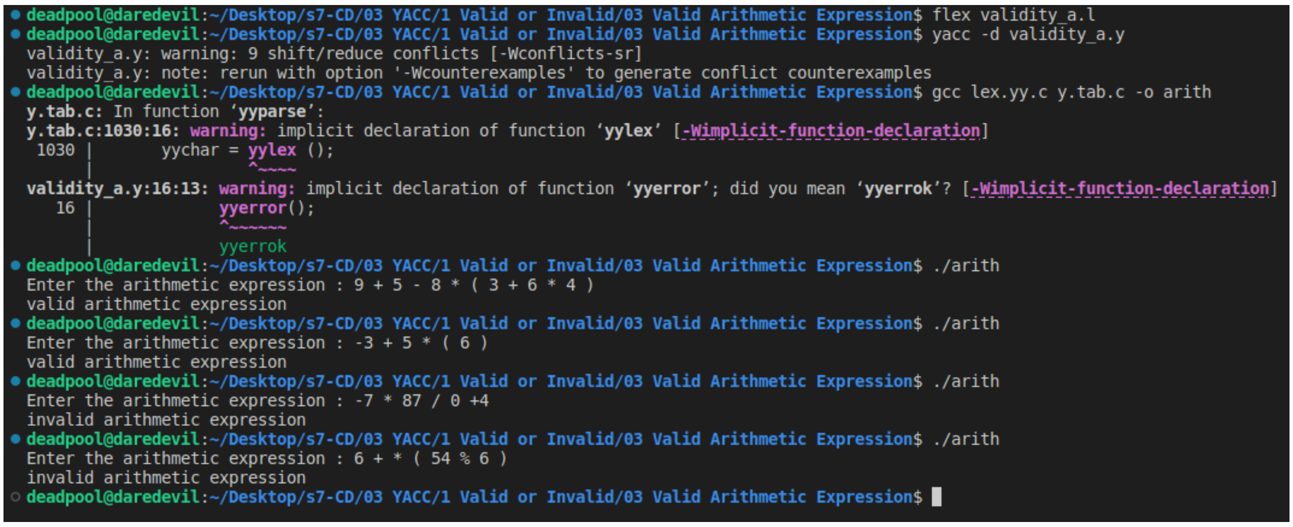
```
%{
   #include<stdio.h>
   #include "y.tab.h"
   extern int yylval;
%}


%%
[0-9]+ { yylval = atoi(yytext); return NUM; }
[a-zA-Z][a-zA-Z0-9]* { return ID; }
[ \t\n] {;}
. { return yytext[0]; }
%%
```

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

<u>Valid Arithmetic Expression (YACC)</u>

```
%{
   #include<stdio.h>
   int flag = 0;
%}

%token NUM ID
%left '+' '-'
%left '*' '/'
%left '(' ')'

%%
E:E'+'E
|E'-'E
|E'*'E
|E'/'E { if($3 == 0){
             yyerror();
         }  }
|E'%'E { if($3 == 0){
             yyerror();
         }  }
|'('E')'
|'-'NUM
|NUM
;
%%

int yyerror(){
   flag = 1;
   printf("invalid arithmetic expression\n");
   return 1;
}

int main(){
   printf("Enter the arithmetic expression : ");
   yyparse();
   if(flag == 0){
       printf("valid arithmetic expression\n");
   }
   return 0;
}
```

<u>output : if output is not printed then press ctrl+D</u>



```
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/1 Valid or Invalid/03 Valid Arithmetic Expression$ flex validity_a.l
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/1 Valid or Invalid/03 Valid Arithmetic Expression$ yacc -d validity_a.y
  validity_a.y: warning: 9 shift/reduce conflicts [-Wconflicts-sr]
  validity_a.y: note: rerun with option '-Wcounterexamples' to generate conflict counterexamples
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/1 Valid or Invalid/03 Valid Arithmetic Expression$ gcc lex.yy.c y.tab.c -o arith
  y.tab.c: In function ‘yyparse’:
  y.tab.c:1030:16: warning: implicit declaration of function ‘yylex’ [-Wimplicit-function-declaration]
   1030 |       yychar = yylex ();
        |                ^~~~~
  validity_a.y:16:13: warning: implicit declaration of function ‘yyerror’; did you mean ‘yyerrok’? [-Wimplicit-function-declaration]
     16 |             yyerror();
        |             ^~~~~~~
        |             yyerrok
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/1 Valid or Invalid/03 Valid Arithmetic Expression$ ./arith
  Enter the arithmetic expression : 9 + 5 - 8 * ( 3 + 6 * 4 )
  valid arithmetic expression
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/1 Valid or Invalid/03 Valid Arithmetic Expression$ ./arith
  Enter the arithmetic expression : -3 + 5 * ( 6 )
  valid arithmetic expression
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/1 Valid or Invalid/03 Valid Arithmetic Expression$ ./arith
  Enter the arithmetic expression : -7 * 87 / 0 +4
  invalid arithmetic expression
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/1 Valid or Invalid/03 Valid Arithmetic Expression$ ./arith
  Enter the arithmetic expression : 6 + * ( 54 % 6 )
  invalid arithmetic expression
○ deadpool@daredevil:~/Desktop/s7-CD/03 YACC/1 Valid or Invalid/03 Valid Arithmetic Expression$ █
```