Practical - 7

AIM : Write a program to implement a calculator using Lex and YACC.

YACC file

```
%{
#include <stdio.h>
%}
%token NUMBER
%left '+' '-'
%left '*' '/' '%'
%left '(' ')'
%%
ArithmeticExpression:
   E { printf("\nResult=%d\n", $$); return 0; }
;
E:
   E '+' E { $$ = $1 + $3; }
 | E '-' E { $$ = $1 - $3; }
 | E '*' E { $$ = $1 * $3; }
 | E '/' E { $$ = $1 / $3; }
 | E '%' E { $$ = $1 % $3; }
 | '(' E ')' { $$ = $2; }
 | NUMBER { $$ = $1; }
;
%%

int main() {
   printf("Enter any expression here: ");
   yyparse();
   return 0;
}

void yyerror() {
    printf("\nEntered arithmetic expression is invalid\n\n");
}
```

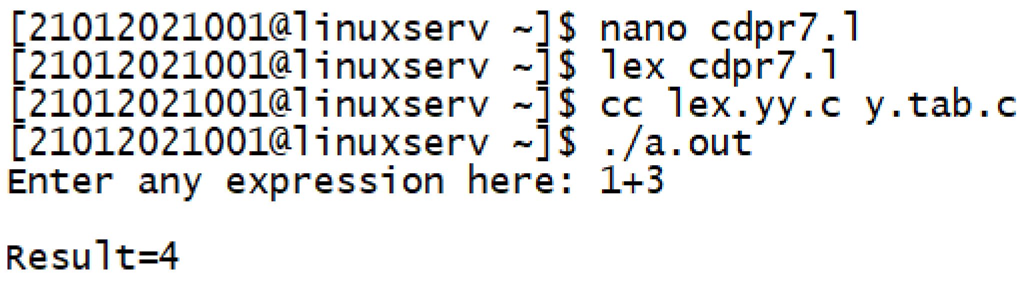
LEX file

```
%{
#include <stdio.h>
#include "y.tab.h"  // This includes the token definitions from the Yacc file.
extern int yylval;
%}

%%
[0-9]+          { yylval = atoi(yytext); return NUMBER; }
[ \t]          ;  // Ignore whitespace
[\n]           { return 0; }  // End of input
.              { return yytext[0]; }  // Return single character

%%

int yywrap() {
    return 1;  // Return 1 to indicate the end of the input
}
```



```
[21012021001@linuxserv ~]$ nano cdpr7.l
[21012021001@linuxserv ~]$ lex cdpr7.l
[21012021001@linuxserv ~]$ cc lex.yy.c y.tab.c
[21012021001@linuxserv ~]$ ./a.out
Enter any expression here: 1+3

Result=4
```