calc.l

```
%{
#include<stdio.h>
#include<stdlib.h>
#include <math.h>
%}

DIGIT [0-9]+\.?|[0-9]*\.[0-9]+


%%

{DIGIT}    {yylval=atof(yytext);return NUM;}
"exit"   {return exit_command;}
cos|COS             {return COS;}
sin|SIN       {return SIN;}
tan|TAN     {return TAN;}
log|LOG     {return LOG;}
sqrt|SQRT   {return SQRT;}


\n|.     {return yytext[0];}
```

calc.y

```
%{
   #include<ctype.h>
   #include<stdio.h>
   #include <math.h>
   #include<stdlib.h>
   #define YYSTYPE double

   int yylex(void);
   void yyerror(char *s);
%}


%token NUM
```

```
%token COS SIN TAN LOG
%token SQRT
%token BOOL


%left '+' '-'
%left '*' '/' '%'
%left '&' '|' '!' 'x'

%token exit_command
%token echo


%%

S       : S E '\n' { printf("Answer: %g \nEnter next expression:\n", $2); }
        | S '\n'
        |
        | error '\n' { yyerror("Error: Enter once more...\n" );yyerrok; }

        ;
E       : E '+' E    { $$ = $1 + $3; }
        | E'-'E    { $$=$1-$3; }
        | E'*'E    { $$=$1*$3; }

         |E'/'E    { $$=$1/$3; }
         |E'&'E              { $$=(int)$1&(int)$3; }
         |E'|'E              { $$=(int)$1|(int)$3; }


        | '('E')'    { $$=$2; }

        | NUM
        | COS'('E')' {$$=cos($3);}
        | SIN'('E')' {$$=sin($3);}
        | TAN'('E')' {$$=tan($3);}
        | LOG'('E')' {$$=log($3);}
       | SQRT'('E')' {$$=sqrt($3);}
       | exit_command          {exit(EXIT_SUCCESS);}
```

```
        ;


%%

#include "lex.yy.c"

int main()
{
   printf("Enter the expression: ");
   yyparse();
}
```

addr.l

```
%{
#include<stdio.h>
#include<stdlib.h>
#include"y.tab.h"
int yywrap(void);
%}

%%
[0-9]+ {yylval=atoi(yytext);return STR;}
[+-/*\n] return *yytext;
\
[ \t] {}
. {printf("invalid string");}
%%

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

addr.y

```
%{

#include "lex.yy.c"
#include<stdio.h>
#include<stdlib.h>
int yylex(void);
void yyerror(char *s);
int i=0;
char p='p';
%}


//%left '-"+'
%left '/"+'
%left '*"-'

%token STR

%%
prog : prog expr '\n' {printf("%c = %c\n",p,p-1);printf("Result=%d",$2);}
|
;
expr : STR
| expr '+' expr {if(i==0){ printf("%c = %d %c %d\n",p,$1,'+',$3); p++;i++;$$=$1+
$3;}
                else{ printf("%c = %c %c %d\n",p,p-1,'+',$3);p++;$$=$1+$3;}}
| expr '-' expr {if(i==0){ printf("%c = %d %c %d\n",p,$1,'-',$3); p++;i++;$$=$1-
$3;}
                else{ printf("%c = %c %c %d\n",p,p-1,'-',$3);p++;$$=$1-$3;}}
| expr '*' expr {if(i==0){ printf("%c = %d %c %d\n",p,$1,'*',$3); p++;i++;$
$=$1*$3;}
                else{ printf("%c = %c %c %d\n",p,p-1,'*',$3);p++;$$=$1*$3;}}
| expr '/' expr {if(i==0){ printf("%c = %d %c %d\n",p,$1,'/',$3); p++;i++;$
$=$1/$3;}
                else{ printf("%c = %c %c %d\n",p,p-1,'/',$3);p++;$$=$1/$3;}}

;
```

```
%%

void yyerror(char *s)
{
printf("%s\n",s);
}

int main()
{
printf("--Three address code generation --\n");
printf("Enter the expression:\n");
yyparse();
return 0;
}
```