Lab 6: Implement a desk calculator using LEX and YACC.

Fourth.l

```
%{
 # include<stdio.h>
 # include<stdlib.h>
 # include "y.tab.h"
%}
num ([0-9]+)(\.[0-9]+)?([eE][-+]?[0-9]+)?
%%
"sin"   {return SIN;}
"cos"   {return COS;}
"tan"   {return TAN;}
[A-Za-z][A-Za-z0-9]*    {yylval.p=install_id(yytext);return id;}
{num}  {yylval.v=atof(yytext); return  NUM;}
[\n]    {return yytext[0];}
.   {return yytext[0];}
%%
int yywrap()
{
 return 1;
}
```

d1.y

```
%{
# include<math.h>
# include<stdio.h>
struct symtab
{
char *name;
double val;
}SYM[20];
void disp();
struct symtab *install_id(char *s);
%}
%union{
  double v;
  struct symtab *p;
}
%token SIN COS TAN
%token<v> NUM
%token<p> id
%right '='
%left '+' '-'
%left '*' '/'
%nonassoc UMINUS
%type<v> E
%%
lines: lines S
    |S
```

```
            ;
S: id '=' E '\n'      {$1->val=$3;}
 |E '\n'   {printf("ans = %lf\n",$1);}
E: E'+'E              {$$ = $1 + $3;}
    | E'-'E           {$$=$1-$3;}
    |E'*'E            {$$=$1*$3;}
    |E'/'E {$$=$1/$3;}
  | '-' E   %prec UMINUS {$$ = -$2;}
 |'('E')'   {$$ = $2;}
 |NUM
 |id        {$$=$1->val;}
 |SIN'('E')'          {$$=sin(($3*3.14)/180);}
 |COS'('E')'        {$$=cos(($3*3.14)/180);}
 |TAN'('E')'    {$$=tan(($3*3.14)/180);}
;
%%
main()
{
yyparse();
disp();
}
int yyerror()
{
return 1;
}

void disp()
{
struct symtab *k;
for(k=SYM;k<&SYM[20];k++)
{
 if(k->name)
 {
 printf("%s\t%f\n",k->name,k->val);
 }  }  }
struct symtab * install_id(char *s)
{
struct symtab *k;
for(k=SYM;k<&SYM[20];k++)
{
 if(k->name&&!strcmp(k->name,s))
 return k;
 else
 if(!k->name)
 {
 k->name=strdup(s);
 return k;
 }
}
}
```

OUTPUT:

```
-> ./a.out
tan(45)
ans = 0.999204
sin(90)
ans = 1.000000
cos(60)
ans = 0.500460
12+12
ans = 24.000000
12*12
ans = 144.000000
1212
ans = 1212.000000
12/12
ans = 1.000000
->
```