Lex and Yacc for simple Calculator

Lex:

```
/  Lex rules for simple
calculator %{
/  including required libraries and
files #include<stdlib.h>
#include
"y.tab.h" extern
int yylval; %}
  /       representations of the symbols
%%
[0-9]+ {yylval=atoi(yytext); return
NUMBER;} ">=" return GE ;
"<=" return LE ;
"!=" return NE ;
"==" return
EQ ; [\n] return
0 ; [\t];
. return yytext[0];
%%
```

Yacc:

```
/  Yacc grammar for simple
calculator %{
#include<stdio.h>
%}
%token NAME
NUMBER %left GE LE NE
EQ '<' '>' '%' %left '-' '+'
%left '*'
'/' %nonassoc
UMINUS
%%
statement:NAME '=' exp
        |exp {printf("=%d\n",$1);}
        ;
exp:NUMBER { $$ = $1;}
        |exp '+' exp { $$ = $1 + $3 ;}
        |exp '-' exp { $$ = $1 - $3 ;}
        |exp '*' exp { $$ = $1 * $3 ;}
        |exp '/' exp { $$ = $1 / $3 ;}
        |exp '<' exp { $$ = $1 < $3 ;}
        |exp '>' exp { $$ = $1 > $3 ;}
        |exp '%' exp { $$ = $1 % $3 ;}
        |exp GE exp { $$ = $1 >= $3 ;}
        |exp LE exp { $$ = $1 <= $3 ;}
        |exp NE exp { $$ = $1 != $3 ;}
        |exp EQ exp { $$ = $1 == $3 ;}
        |'-' exp %prec UMINUS { $$ = -
        $2 ;} |'(' exp ')' { $$ = $2;}
%%
int main()
```

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

OUTPUT: