**YACC program for FOR LOOP STATEMENTS**

**Program:**  
  
// **Lex file: for.l**  
  
alpha [A-Za-z]  
digit [0-9]  
  
%%  
  
[\t \n]  
for             return FOR;  
{digit}+    return NUM;  
{alpha}({alpha}|{digit})\* return ID;  
"<="         return LE;  
">="         return GE;  
"=="         return EQ;  
"!="          return NE;  
"||"            return OR;  
"&&"         return AND;  
.                return yytext[0];  
  
%%  
  
  
// **Yacc file: for.y**  
  
%{  
#include <stdio.h>  
#include <stdlib.h>  
%}  
%token ID NUM FOR LE GE EQ NE OR AND  
%right "="  
%left OR AND  
%left '>' '<' LE GE EQ NE  
%left '+' '-'  
%left '\*' '/'  
%right UMINUS  
%left '!'  
  
%%  
     
S         : ST {printf("Input accepted\n"); exit(0);}  
ST       : FOR '(' E ';' E2 ';' E ')' DEF  
           ;  
DEF    : '{' BODY '}'  
           | E';'  
           | ST  
           |  
           ;  
BODY  : BODY BODY  
           | E ';'         
           | ST  
           |              
           ;  
         
E        : ID '=' E  
          | E '+' E  
          | E '-' E  
          | E '\*' E  
          | E '/' E  
          | E '<' E  
          | E '>' E  
          | E LE E  
          | E GE E  
          | E EQ E  
          | E NE E  
          | E OR E  
          | E AND E  
          | E '+' '+'  
          | E '-' '-'  
          | ID   
          | NUM  
          ;  
  
     
E2     : E'<'E  
         | E'>'E  
         | E LE E  
         | E GE E  
         | E EQ E  
         | E NE E  
         | E OR E  
         | E AND E  
         ;     
%%  
  
#include "lex.yy.c"  
main() {  
    printf("Enter the expression:\n");  
    yyparse();  
}       
       
  
  
**Output:**  
nn@linuxmint ~ $ lex for.l  
nn@linuxmint ~ $ yacc for.y  
conflicts: 25 shift/reduce, 4 reduce/reduce  
nn@linuxmint ~ $ gcc y.tab.c -ll -ly  
nn@linuxmint ~ $ ./a.out  
Enter the expression:  
for(i=0;i<n;i++)  
i=i+1;  
Input accepted  
nn@linuxmint ~ $