Exp 7e : YACC – FOR

LEX

```
1. Start
2. %% - rule section
       "for" return FOR
       [ \t\n]
       [0-9]+ return NUM
       [a-zA-Z][a-zA-Z0-9]*  return ID
       \"[^\"]*\"  return STRING
       "<"  return L
       ">"  return G
       "<="  return LE
       ">="  return GE
       "=="  return EE
       "!="  return NE
       "++"  return INC
       "--"  return DEC
       "||"  return OR
       "&&"  return AND
       .  return yytext[0]
3. yywrap() retturn 1
4. Stop
```

YACC

```
1. Start
2. %token FOR L G LE GE EE NE INC DEC OR AND ID NUM STRING
3. %% rule section
       for : FOR '(' asn ';' cond ';' asn ')' '{' stmt '}' { printf("valid for loop\n"); };
       asn : ID '=' E | ID INC | ID DEC ;
       cond : scond | scond AND cond | scond OR cond ;
       scond : nid | nid relop nid ;
       nid : ID | NUM ;
       relop : L | G | LE | GE | EE | NE ;
       stmt : ID '(' STRING other ')' ';' stmt | asn ';' stmt | ;
       other : ',' ID other | ',' '&' ID other | ;
       E : ID'='E | E'+'E | E'-'E | E'*'E | E'/'E | E INC | E DEC | nid | '(' nid ')' ;
4. yyerror() to handle error
5. in main() call yyparse()
```

for – Lex

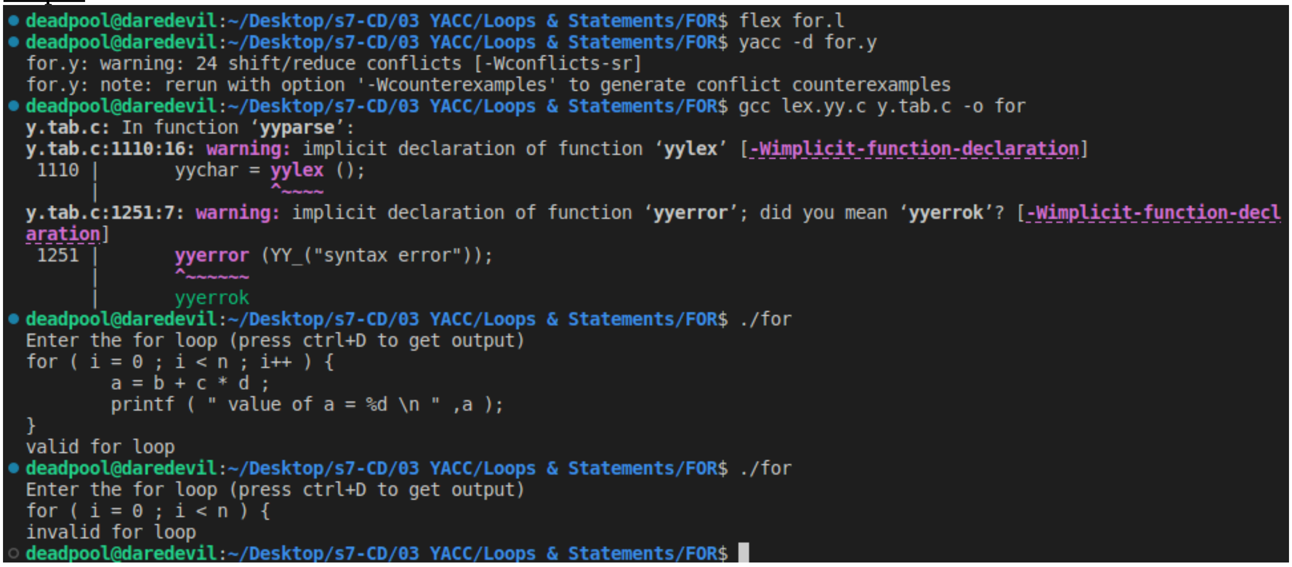
```
%{
  #include<stdio.h>
  #include "y.tab.h"
%}

%%
“for" { return FOR; }

[ \t\n]
[0-9]+ { return NUM; }
[a-zA-Z][a-zA-Z0-9]* { return ID; }
```

```
\"[^\"]*\" { return STRING; }

"<" { return L; }
">" { return G; }
"<=" { return LE; }
">=" { return GE; }
"==" { return EE; }
"!=" { return NE; }
"++" { return INC; }
"--" { return DEC; }
"||" { return OR; }
"&&" { return AND; }
. { return yytext[0]; }
%%

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

for – YACC

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

%token FOR L G LE GE EE NE INC DEC OR AND ID NUM STRING

%%
for : FOR '(' asn ';' cond ';' asn ')' '{' stmt '}' { printf("valid for loop\n"); };
asn : ID '=' E | ID INC | ID DEC ;

cond : scond | scond AND cond | scond OR cond ;
scond : nid | nid relop nid ;
nid : ID | NUM ;
relop : L | G | LE | GE | EE | NE ;
stmt : ID '(' STRING other ')' ';' stmt | asn ';' stmt | ;
other : ',' ID other | ',' '&' ID other | ;
E : ID'='E
| E'+'E
| E'-'E
| E'*'E
| E'/'E
| E INC
| E DEC
| nid
| '(' nid ')'
;
%%

int yyerror(){
   printf("invalid for loop\n");
   return 1;
}

int main(){
   printf("Enter for loop (press ctrl+D to get output)\n");
   yyparse();
   return 0;
}
```

<u>output</u>



```
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/Loops & Statements/FOR$ flex for.l
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/Loops & Statements/FOR$ yacc -d for.y
  for.y: warning: 24 shift/reduce conflicts [-Wconflicts-sr]
  for.y: note: rerun with option '-Wcounterexamples' to generate conflict counterexamples
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/Loops & Statements/FOR$ gcc lex.yy.c y.tab.c -o for
  y.tab.c: In function ‘yyparse’:
  y.tab.c:1110:16: warning: implicit declaration of function ‘yylex’ [-Wimplicit-function-declaration]
   1110 |       yychar = yylex ();
        |                ^~~~~
  y.tab.c:1251:7: warning: implicit declaration of function ‘yyerror’; did you mean ‘yyerrok’? [-Wimplicit-function-decl
  aration]
   1251 |       yyerror (YY_("syntax error"));
        |       ^~~~~~~
        |       yyerrok
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/Loops & Statements/FOR$ ./for
  Enter the for loop (press ctrl+D to get output)
  for ( i = 0 ; i < n ; i++ ) {
          a = b + c * d ;
          printf ( " value of a = %d \n " ,a );
  }
  valid for loop
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/Loops & Statements/FOR$ ./for
  Enter the for loop (press ctrl+D to get output)
  for ( i = 0 ; i < n ) {
  invalid for loop
○ deadpool@daredevil:~/Desktop/s7-CD/03 YACC/Loops & Statements/FOR$ █
```