Exp 7g : YACC – DO WHILE

LEX

```
1. Start
2. %% - rule section
       “do” return DO
       "while" return WHILE
       [ \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 DO WHILE L G LE GE EE NE INC DEC OR AND ID NUM STRING
3. %% rule section
       S : do while { print "valid do-while loop" };
       do : DO '{' stmt '}' ;
       while : WHILE '(' cond ')' ';' ;
       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 | E ';' 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()
```

do while – Lex

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

%%
“do” { return DO; }
"while" { return WHILE; }
```

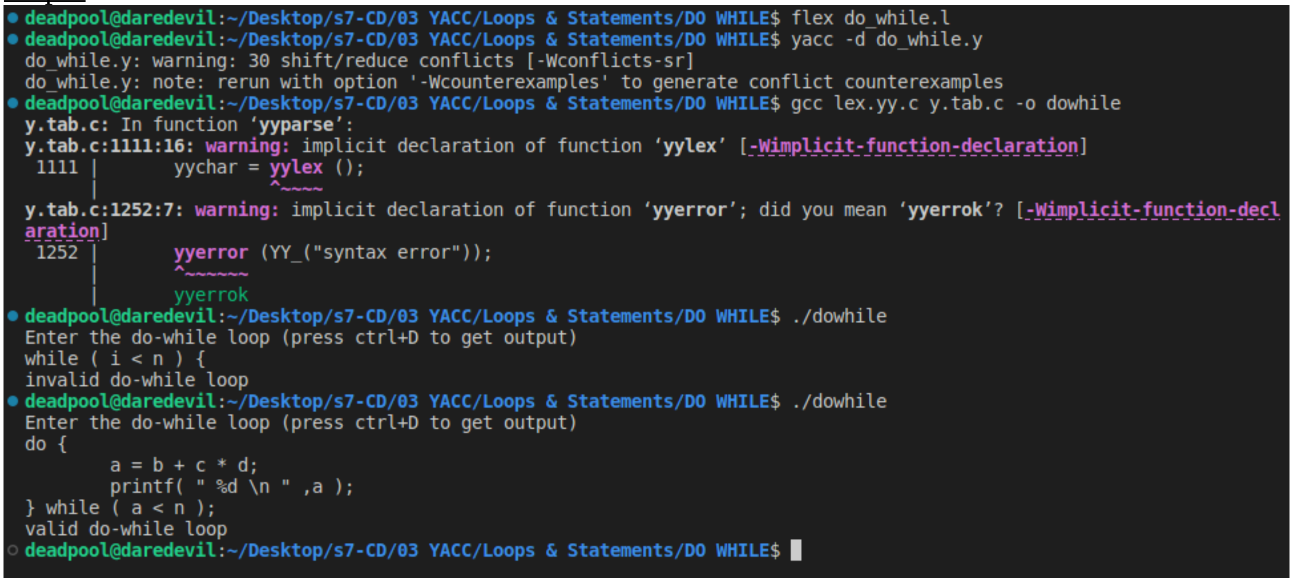
```
[ \t\n]
[0-9]+ { return NUM; }
[a-zA-Z][a-zA-Z0-9]* { return ID; }
\"[^\"]*\" { return STRING; }

"<" { 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;
}
```

<u>do while – YACC</u>

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

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

%%
S : do while { printf("valid do-while loop\n"); };
do : DO '{' stmt '}' ;
while : WHILE '(' cond ')' ';' ;

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 | E ';' 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 do-while loop\n");
    return 1;
}

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

<u>output</u>



```
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/Loops & Statements/DO WHILE$ flex do_while.l
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/Loops & Statements/DO WHILE$ yacc -d do_while.y
  do_while.y: warning: 30 shift/reduce conflicts [-Wconflicts-sr]
  do_while.y: note: rerun with option '-Wcounterexamples' to generate conflict counterexamples
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/Loops & Statements/DO WHILE$ gcc lex.yy.c y.tab.c -o dowhile
  y.tab.c: In function 'yyparse':
  y.tab.c:1111:16: warning: implicit declaration of function 'yylex' [-Wimplicit-function-declaration]
   1111 |       yychar = yylex ();
        |                ^~~~~
  y.tab.c:1252:7: warning: implicit declaration of function 'yyerror'; did you mean 'yyerrok'? [-Wimplicit-function-decl
  aration]
   1252 |       yyerror (YY_("syntax error"));
        |       ^~~~~~~
        |       yyerrok
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/Loops & Statements/DO WHILE$ ./dowhile
  Enter the do-while loop (press ctrl+D to get output)
  while ( i < n ) {
  invalid do-while loop
● deadpool@daredevil:~/Desktop/s7-CD/03 YACC/Loops & Statements/DO WHILE$ ./dowhile
  Enter the do-while loop (press ctrl+D to get output)
  do {
          a = b + c * d;
          printf( " %d \n " ,a );
  } while ( a < n );
  valid do-while loop
○ deadpool@daredevil:~/Desktop/s7-CD/03 YACC/Loops & Statements/DO WHILE$ █
```