# Practical No: 7

Name : Patel Savankumar P.
Enroll No : 19BCE519
Subject : Compiler Construction

**AIM: To implement grammar rules for control statements, and Loop control.**

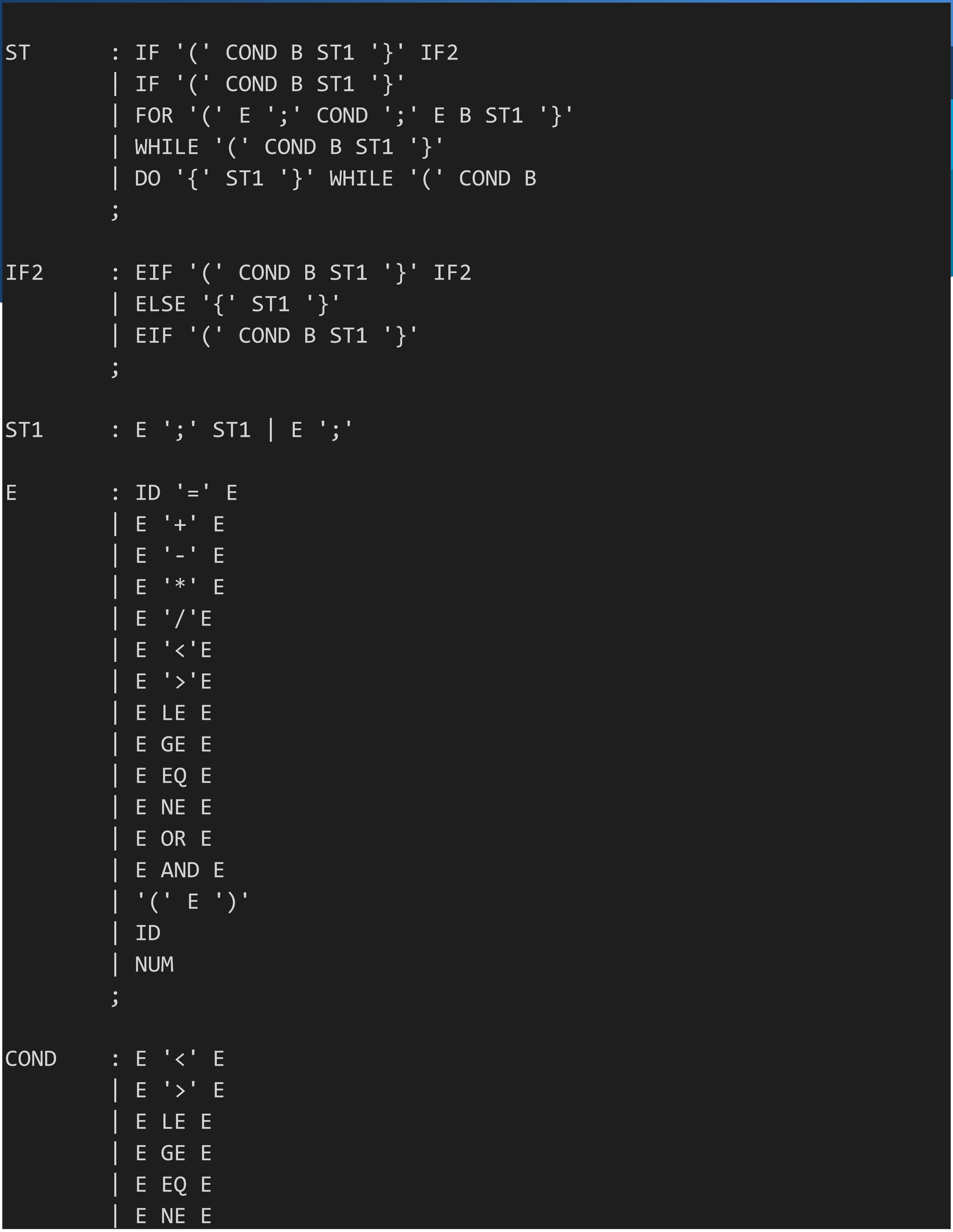
**File 1: practical7.l**

```
%{
#include "y.tab.h"
%}

alpha [A-Za-z]
digit [0-9]

%%
[\t]    ;
[\n]    ;
"exit"    return 0;
if      return IF;
else    return ELSE;
while   return WHILE;
do      return DO;
for     return FOR;
"else if"    return EIF;
[0-9]+    return NUM;
```

```
{alpha}({alpha}|{digit})*    return ID;
"<="    return LE;
">="    return GE;
"=="    return EQ;
"!="    return NE;
"||"    return OR;
"&&"    return AND;
"){"    return B;
") {"   return B;
")\n{"   return B;
");"    return B;
";}"    return B;
" "     ;
.       return yytext[0];
%%

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

## File 2: practical7.y

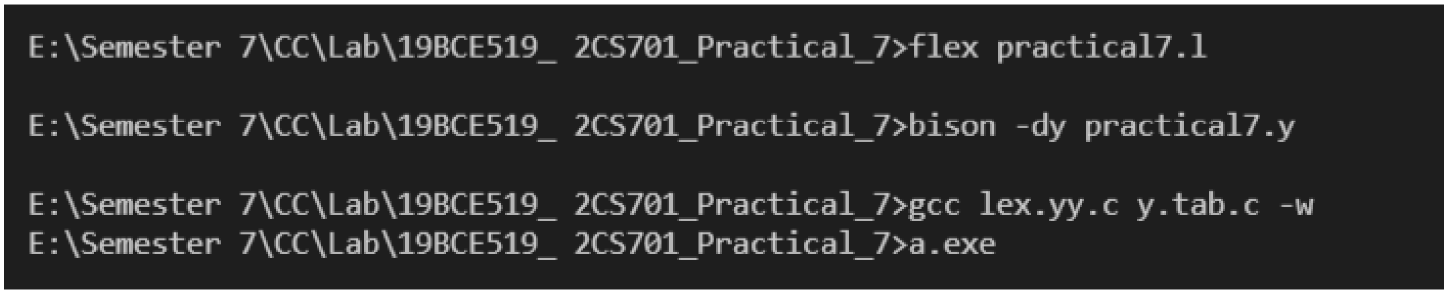
```
%{
#include <stdio.h>
#include <stdlib.h>
%}

%token ID NUM IF LE GE EQ NE OR AND ELSE B WHILE DO FOR EIF
%right '='
%left AND OR
%left '<' '>' LE GE EQ NE
%left '+' '-'
%left '*' '/'
%left '!'
%%

S       : ST {printf("\nSyntax is Valid");exit(0);};
```



```
ST       : IF '(' COND B ST1 '}' IF2
         | IF '(' COND B ST1 '}'
         | FOR '(' E ';' COND ';' E B ST1 '}'
         | WHILE '(' COND B ST1 '}'
         | DO '{' ST1 '}' WHILE '(' COND B
         ;

IF2      : EIF '(' COND B ST1 '}' IF2
         | ELSE '{' ST1 '}'
         | EIF '(' COND B ST1 '}'
         ;

ST1      : E ';' ST1 | E ';'

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 ')'
         | ID
         | NUM
         ;

COND     : E '<' E
         | E '>' E
         | E LE E
         | E GE E
         | E EQ E
         | E NE E
```

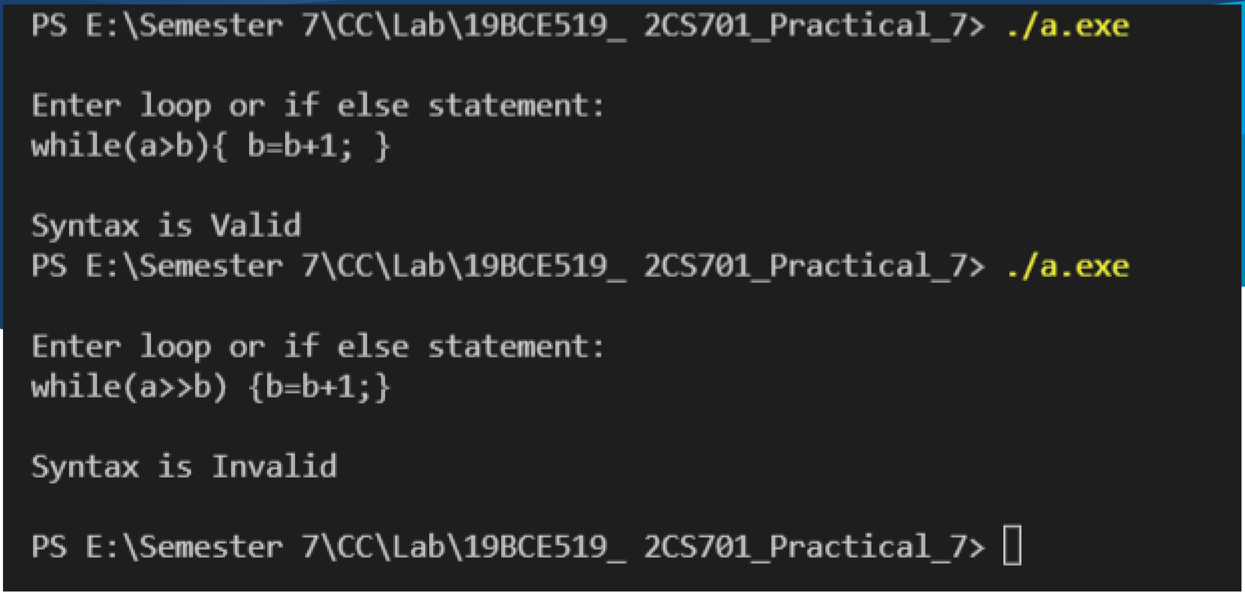
```
        | E OR E
        | E AND E
        | ID
        | NUM
        ;
%%

void main()
{
printf("\nEnter loop or if else statement:\n");

yyparse();
}

void yyerror()
{
printf("\nSyntax is Invalid\n\n");
}
```

## Execution Sequence:



```
E:\Semester 7\CC\Lab\19BCE519_ 2CS701_Practical_7>flex practical7.l

E:\Semester 7\CC\Lab\19BCE519_ 2CS701_Practical_7>bison -dy practical7.y

E:\Semester 7\CC\Lab\19BCE519_ 2CS701_Practical_7>gcc lex.yy.c y.tab.c -w
E:\Semester 7\CC\Lab\19BCE519_ 2CS701_Practical_7>a.exe
```

## Output:



```
PS E:\Semester 7\CC\Lab\19BCE519_ 2CS701_Practical_7> ./a.exe

Enter loop or if else statement:
while(a>b){ b=b+1; }

Syntax is Valid
PS E:\Semester 7\CC\Lab\19BCE519_ 2CS701_Practical_7> ./a.exe

Enter loop or if else statement:
while(a>>b) {b=b+1;}

Syntax is Invalid

PS E:\Semester 7\CC\Lab\19BCE519_ 2CS701_Practical_7>
```

## Conclusion:

From this practical I learned how to write Yacc and Lex code to check for conditional and loop constructs.