Practical - 9

AIM: Write a program to validate the IF…. ELSE statement using Lex and YACC.

YACC CODE

%{

#include <stdio.h>

#include <stdlib.h>

int yylex(void);

void yyerror(const char *s);

%}

%token ID NUM IF OPBR CLBR LE GE EQ NE OR AND ELSE

%right '='

%left AND OR

%left '<' '>' LE GE EQ NE

%left '+' '-'

%left '*' '/'

%right UMINUS

%left '!'

%%

S   : ST   { printf("Input accepted.\n"); exit(0); }

    ;

ST   : IF '(' E2 ')' OPBR ST1 ';' CLBR ELSE ST1 ';'

    | IF '(' E2 ')' OPBR ST1 ';' CLBR

    ;

ST1 : ST

| 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

| 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

    | ID

    | NUM

    ;

%%

void yyerror(const char *str) {

    fprintf(stderr, "Syntax Error: %s\n", str);

}

int yywrap(void) {

    return 1;

}


int main(void) {

    printf("Enter the expression: ");

    yyparse();

    return 0;

}
```

LEX CODE

```
%{

#include <stdio.h>

#include "y.tab.h"

%}


alpha [A-Za-z]

digit [0-9]
```

%%

[\t\n]   ; /* Skip whitespace */

"if"   { return IF; }

"else"   { return ELSE; }

"{"   { return OPBR; }

"}"   { return CLBR; }

{digit}+   { return NUM; }

{alpha}({alpha}|{digit})*   { return ID; }

"<="   { return LE; }

">="   { return GE; }

"=="   { return EQ; }

"!="   { return NE; }

"||"   { return OR; }

"&&"   { return AND; }

.   { return yytext[0]; }

%%

```
[21012021001@linuxserv ~]$ nano cdpr9.y
[21012021001@linuxserv ~]$ nano cdpr9.l
[21012021001@linuxserv ~]$ yacc -d cdpr9.y
[21012021001@linuxserv ~]$ lex cdpr9.l
[21012021001@linuxserv ~]$ cc lex.yy.c y.tab.c
[21012021001@linuxserv ~]$ ./a.out
Enter the expression: if ( a < 7 ) { x = 1 ; } else x = 0 ;
Input accepted.
[21012021001@linuxserv ~]$ if a<b x
>
>
[21012021001@linuxserv ~]$ if (a<b)
> if (a>4) {x = 3}
-bash: syntax error near unexpected token `{x'
```