MANICK VISHAL C (220701158)

EXP NO: 07 **DATE:**

RECOGNIZE A VALID CONTROL STRUCTURES SYNTAX OF C LANGUAGE (FOR LOOP, WHILE LOOP, IF-ELSE, IF-ELSE-IF, SWITCH CASE, ETC.,)

AIM:

To design and implement a LEX and YACC program that recognizes the syntax of common control structures in C programming, including:

For loop

- While loop
- If-else
- If-else-if
- Switch-case

ALGORITHM:

LEX (Lexical Analyzer)

1. Start
2. Define token patterns for:
    - Keywords (e.g., if, else, for, while, switch, case)
    - Identifiers (variable names)
    - Operators (arithmetic and relational)
    - Parentheses (), {}, etc.)
    - Semicolon (;)
3. Pass recognized tokens to YACC for syntax validation.
4. End

YACC (Syntax Analyzer)

1. Start
2. Define grammar rules for:
    - For loop: for(initialization; condition; increment) { ... }
    - While loop: while(condition) { ... }
    - If-else: if(condition) { ... } else { ... }
    - If-else-if: if(condition) { ... } else if(condition) { ... } else { ... }
    - Switch-case: switch(expression) { case value: ... default: ... }
3. Parse the input expression and validate the syntax of the control structures.
4. Print appropriate messages for valid or invalid control structure syntax.
5. End

MANICK VISHAL C (220701158)

PROGRAM:

Control.l

```
%{
#include "control.tab.h"
#include <string.h>
#include <stdlib.h>
%}

%%

"if"        { return IF; }
"else"      { return ELSE; }
"while"     { return WHILE; }
"for"       { return FOR; }

"<="        { return LE; }
">="        { return GE; }
"=="        { return EQ; }
"!="        { return NE; }

[a-zA-Z_][a-zA-Z0-9_]*    { yylval.str = strdup(yytext); return ID; }
[0-9]+                    { yylval.str = strdup(yytext); return NUM; }

"="         { return '='; }
"+"         { return '+'; }
"-"         { return '-'; }
"*"         { return '*'; }
"/"         { return '/'; }
"<"         { return '<'; }
">"         { return '>'; }

"("         { return '('; }
")"         { return ')'; }
"{"         { return '{'; }
"}"         { return '}'; }
";"         { return ';'; }

[ \t\n]+    { /* ignore whitespace */ }
.           { return yytext[0]; }
%%

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

MANICK VISHAL C (220701158)

control.y

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

void yyerror(const char* s);
int yylex();
extern FILE* yyin;
%}

%union {
   char* str;
}

%token <str> ID NUM
%token IF ELSE WHILE FOR
%token LE GE EQ NE

%left '<' '>' LE GE EQ NE
%left '+' '-'
%left '*' '/'

%%

program:

     stmt_list
   ;

stmt_list:
     stmt
   | stmt_list stmt
   ;

stmt:

     expr ';'                              { /* Expression statement - skip message */ }
   | IF '(' expr ')' stmt                    { printf("IF condition works\n"); }
   | IF '(' expr ')' stmt ELSE stmt            { printf("IF-ELSE condition works\n"); }
   | WHILE '(' expr ')' stmt                  { printf("WHILE loop works\n"); }
   | FOR '(' expr ';' expr ';' expr ')' stmt  { printf("FOR loop works\n"); }
   | '{' stmt_list '}'                       { /* Compound statement - no message needed */ }
   ;
```

```
expr:

     ID '=' expr
   | expr '+' expr
   | expr '-' expr
   | expr '*' expr
   | expr '/' expr
   | expr '<' expr
   | expr '>' expr
   | expr LE expr
   | expr GE expr
   | expr EQ expr
   | expr NE expr
   | ID
   | NUM
   ;

%%
void yyerror(const char* s) {
   printf("Syntax Error: %s\n", s);
}

int main() {
   printf("Enter a C control structure (Ctrl+Z to stop):\n");
   yyin = stdin;  // set to standard input
   yyparse();
   return 0;
}
```

OUTPUT :

```
yacc -d control_structures.y
lex control_structures.l
gcc lex.yy.c y.tab.c -o control_validator
./control_validator
if (a > b) {
 // statements
} else {
// statements
}
for (int i = 0 i < 10; i++) {
// statements
}
```

RESULT:

Thus the above program to recognize a valid control structures syntax of c language (for loop, while loop, if-else, if-else-if, switch case has been implemented and executed successfully with LEX and YACC.