**PRACTICAL 09**

**AIM - Parser for ''FOR'' loop statements.**

**CODE-**

**LEX;**

**%{**

**#include "y.tab.h"**

**#include <stdlib.h>**

**#include <string.h>**

**%}**

**%%**

**"for" { return FOR; }**

**"(" { return LPAREN; }**

**")" { return RPAREN; }**

**";" { return SEMICOLON; }**

**"{" { return LBRACE; }**

**"}" { return RBRACE; }**

**"++" { return INC; }**

**"--" { return DEC; }**

**"<=" { return LE; }**

**">=" { return GE; }**

**"==" { return EQ; }**

**"!=" { return NE; }**

**"<" { return LT; }**

**">" { return GT; }**

**"=" { return ASSIGN; }**

**"+" { return PLUS; }**

**"-" { return MINUS; }**

**"\*" { return MULT; }**

**"/" { return DIV; }**

**[a-zA-Z\_][a-zA-Z0-9\_]\* { yylval.str = strdup(yytext); return ID; }**

**[0-9]+ { yylval.num = atoi(yytext); return NUMBER; }**

**[ \t\n]+ ; // skip whitespace**

**. { return yytext[0]; }**

**%%**

**int yywrap() {**

**return 1;**

**}**

**YACC**

**%{**

**#include <stdio.h>**

**#include <stdlib.h>**

**int yylex(void);**

**void yyerror(const char \*s);**

**%}**

**%union {**

**int num;**

**char \*str;**

**}**

**%token <num> NUMBER**

**%token <str> ID**

**%token FOR LPAREN RPAREN SEMICOLON ASSIGN LT GT LE GE EQ NE**

**%token PLUS MINUS MULT DIV**

**%token INC DEC**

**%token LBRACE RBRACE**

**%left PLUS MINUS**

**%left MULT DIV**

**%%**

**program:**

**/\* empty \*/**

**| program for\_stmt**

**;**

**for\_stmt:**

**FOR LPAREN init SEMICOLON condition SEMICOLON increment RPAREN body**

**{**

**printf("Valid FOR loop statement parsed.\n");**

**}**

**;**

**init:**

**ID ASSIGN expr**

**;**

**condition:**

**expr LT expr**

**| expr GT expr**

**| expr LE expr**

**| expr GE expr**

**| expr EQ expr**

**| expr NE expr**

**;**

**increment:**

**ID ASSIGN expr**

**| ID INC**

**| ID DEC**

**| INC ID**

**| DEC ID**

**;**

**body:**

**statement**

**| LBRACE statements RBRACE**

**;**

**statements:**

**/\* empty \*/**

**| statements statement**

**;**

**statement:**

**expr SEMICOLON**

**| for\_stmt**

**;**

**expr:**

**NUMBER**

**| ID**

**| expr PLUS expr**

**| expr MINUS expr**

**| expr MULT expr**

**| expr DIV expr**

**| ID INC**

**| ID DEC**

**| INC ID**

**| DEC ID**

**;**

**%%**

**void yyerror(const char \*s) {**

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

**}**

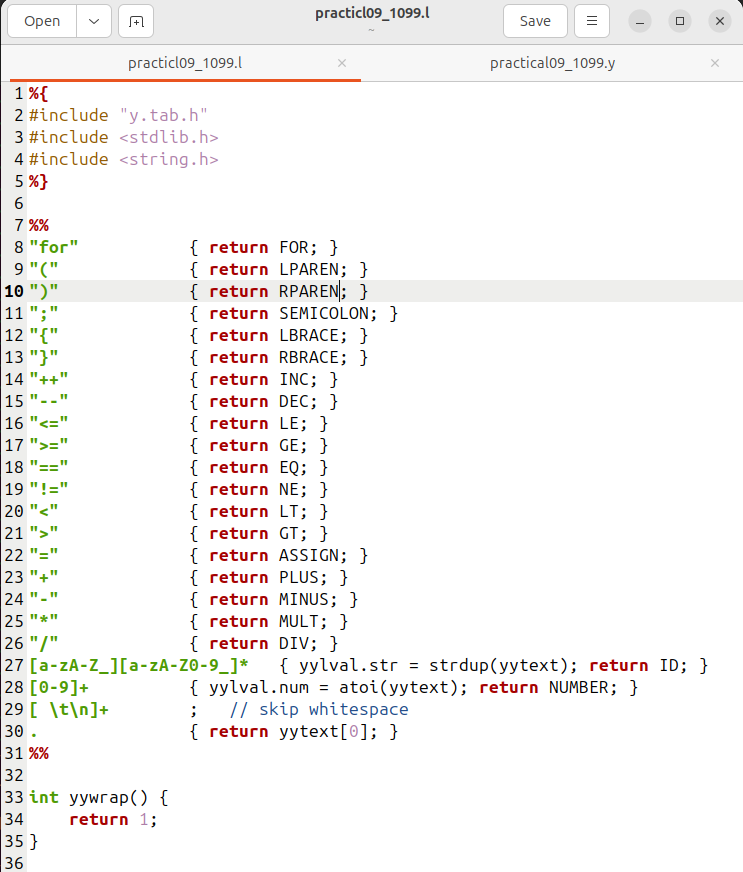
**int main() {**

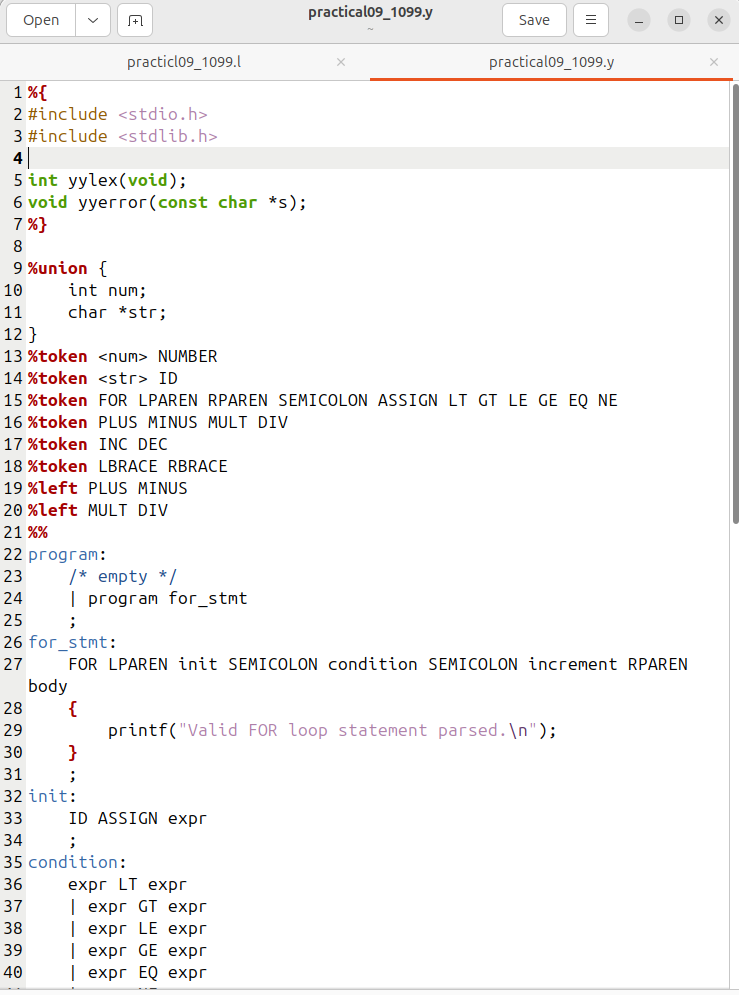
**printf("Enter a FOR loop statement:\n");**

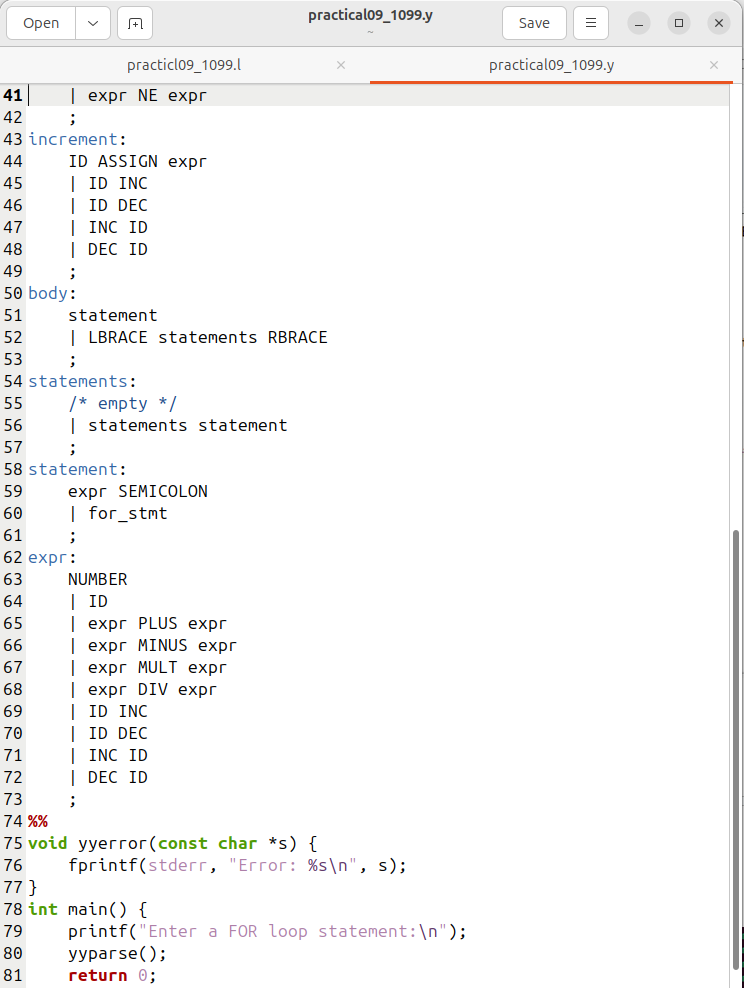
**yyparse();**

**return 0;**

**}**



****

****

