COMPILER DESIGN LAB

CS304

**Project Phase2: Syntax Analysis**

****

**M Shree Harsha Bhat**

211CS137

[mshreeharshabhat.211cs137@nitk.edu.in](mailto:mshreeharshabhat.211cs137@nitk.edu.in)

**Shreekara Rajendra**

211CS151

[shreekararajendra.211cs151@nitk.edu.in](mailto:shreekararajendra.211cs151@nitk.edu.in)

**Shyam Balaji**

211CS154

[shyambalaji.211cs154@nitk.edu.in](mailto:shyambalaji.211cs154@nitk.edu.in)

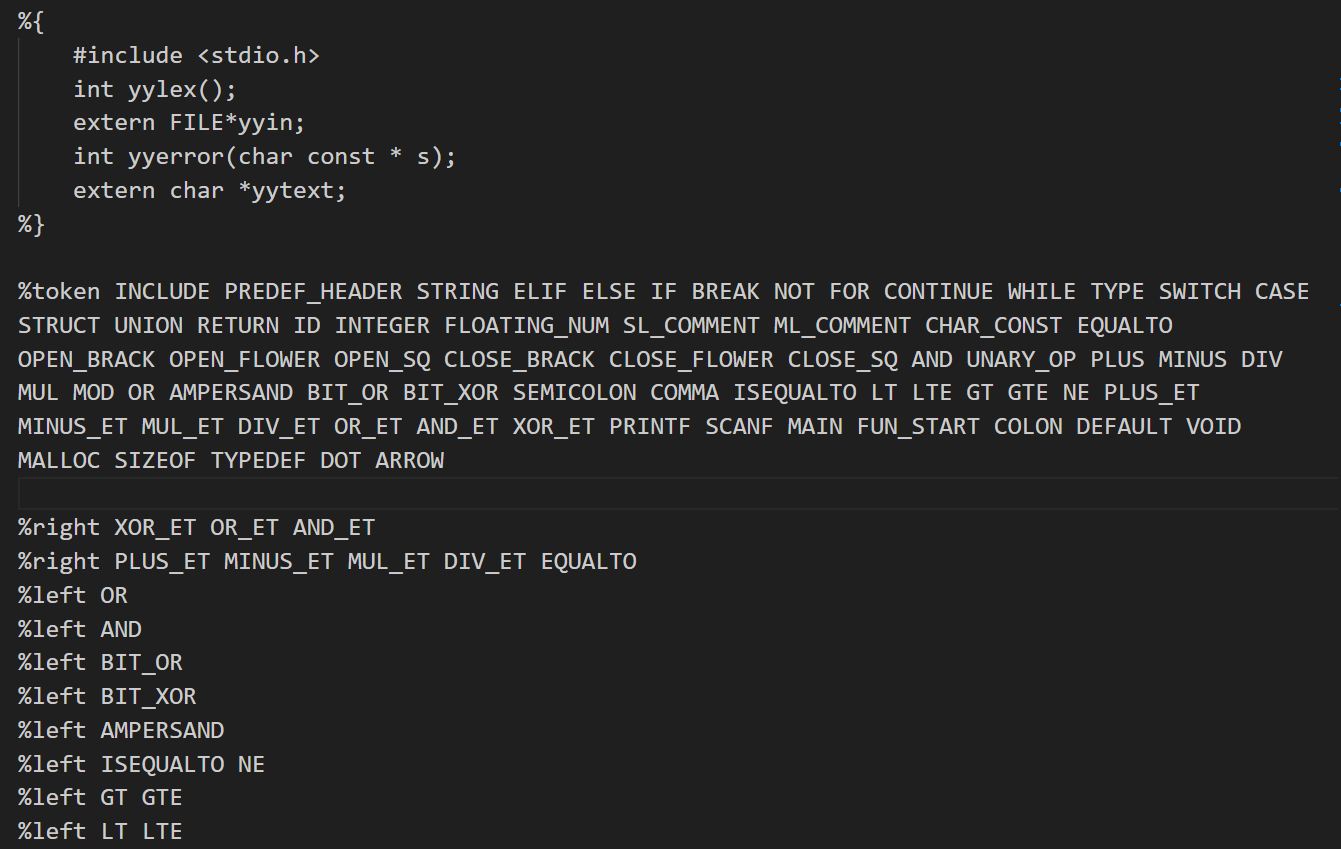
**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

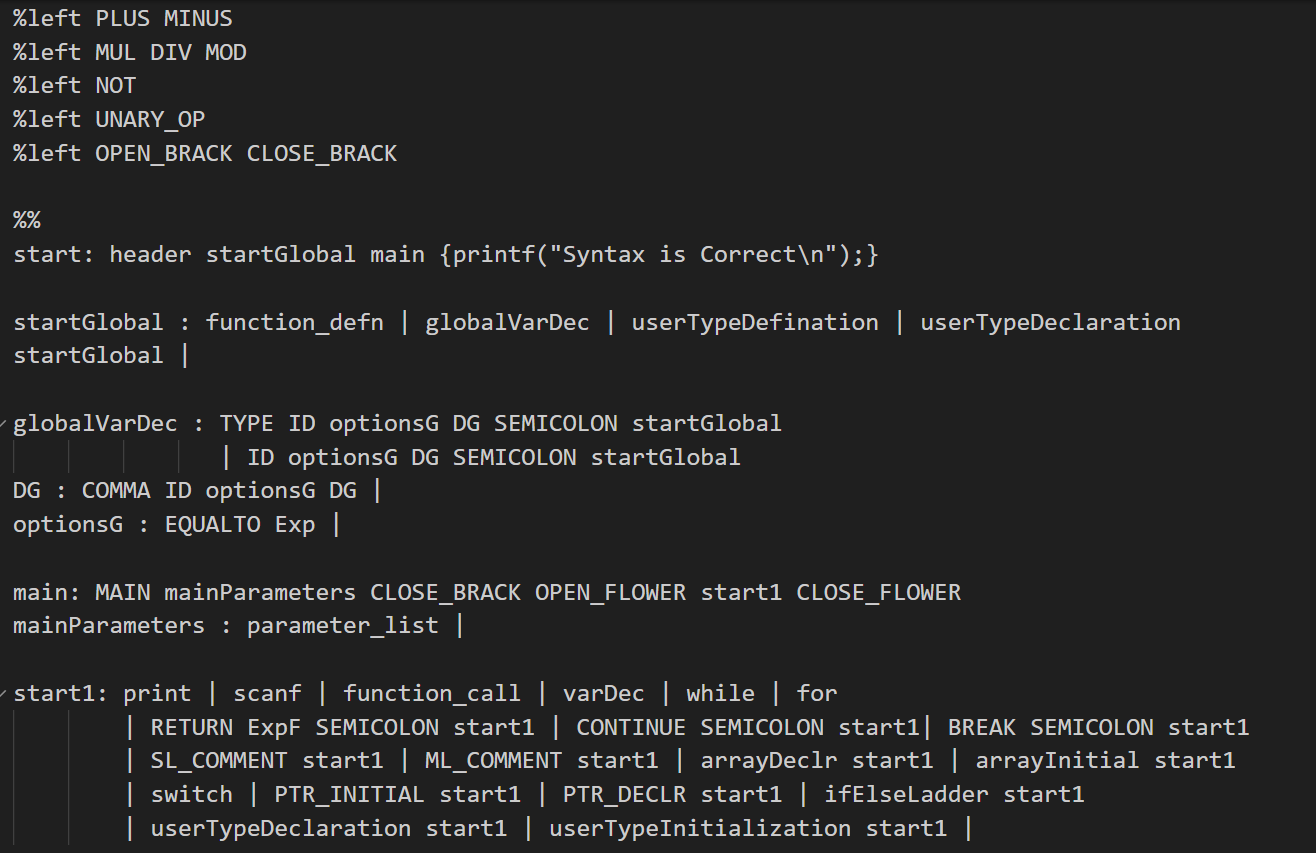
**NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA**

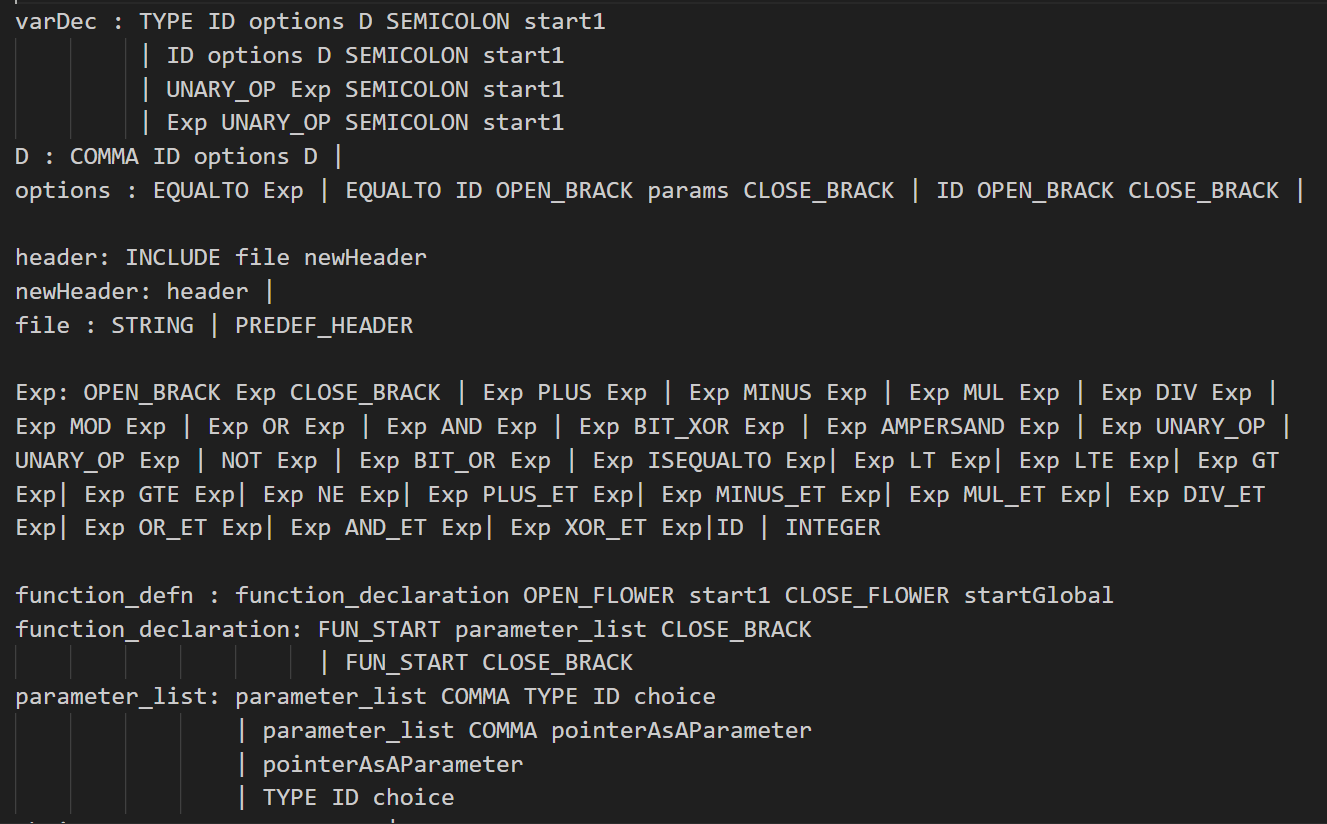
**SURATHKAL, MANGALORE – 57025**

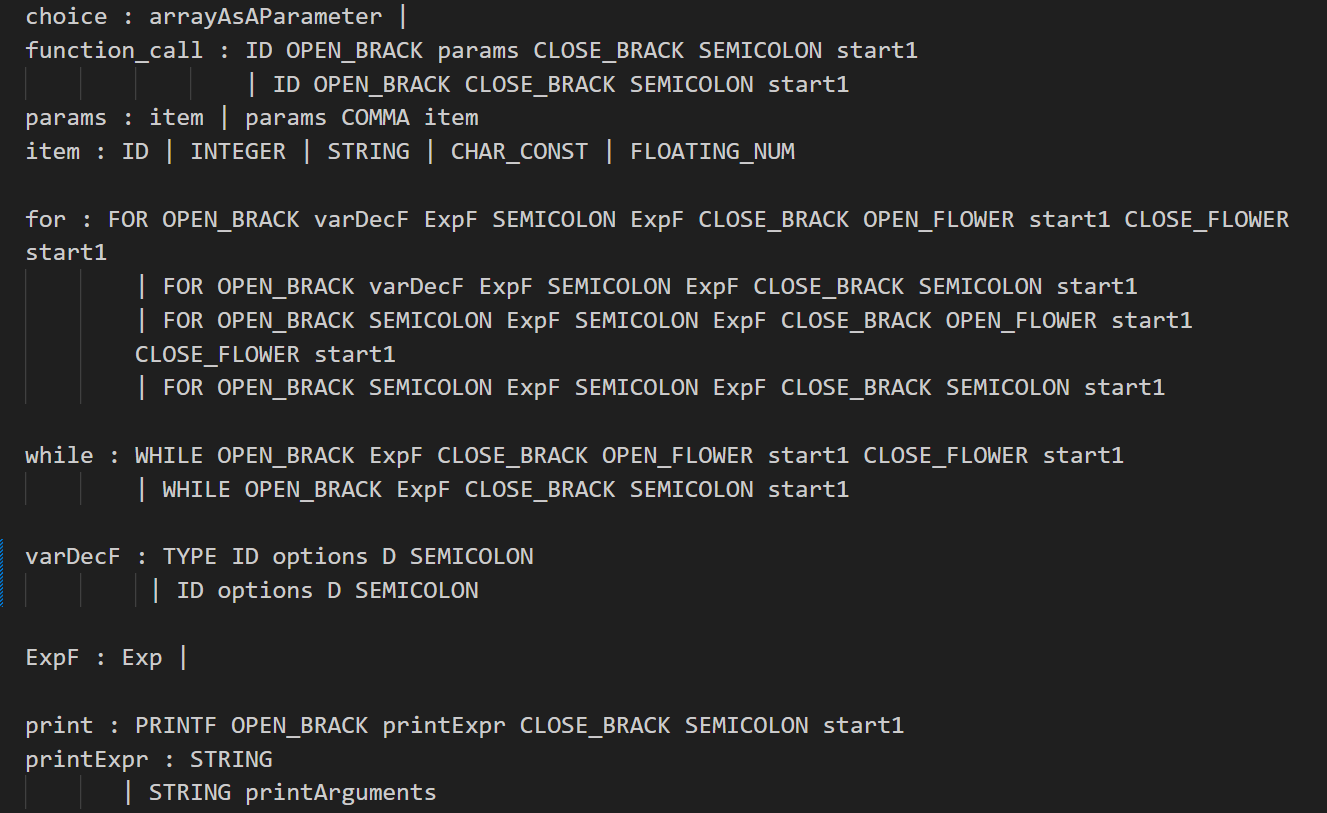
**CODE**

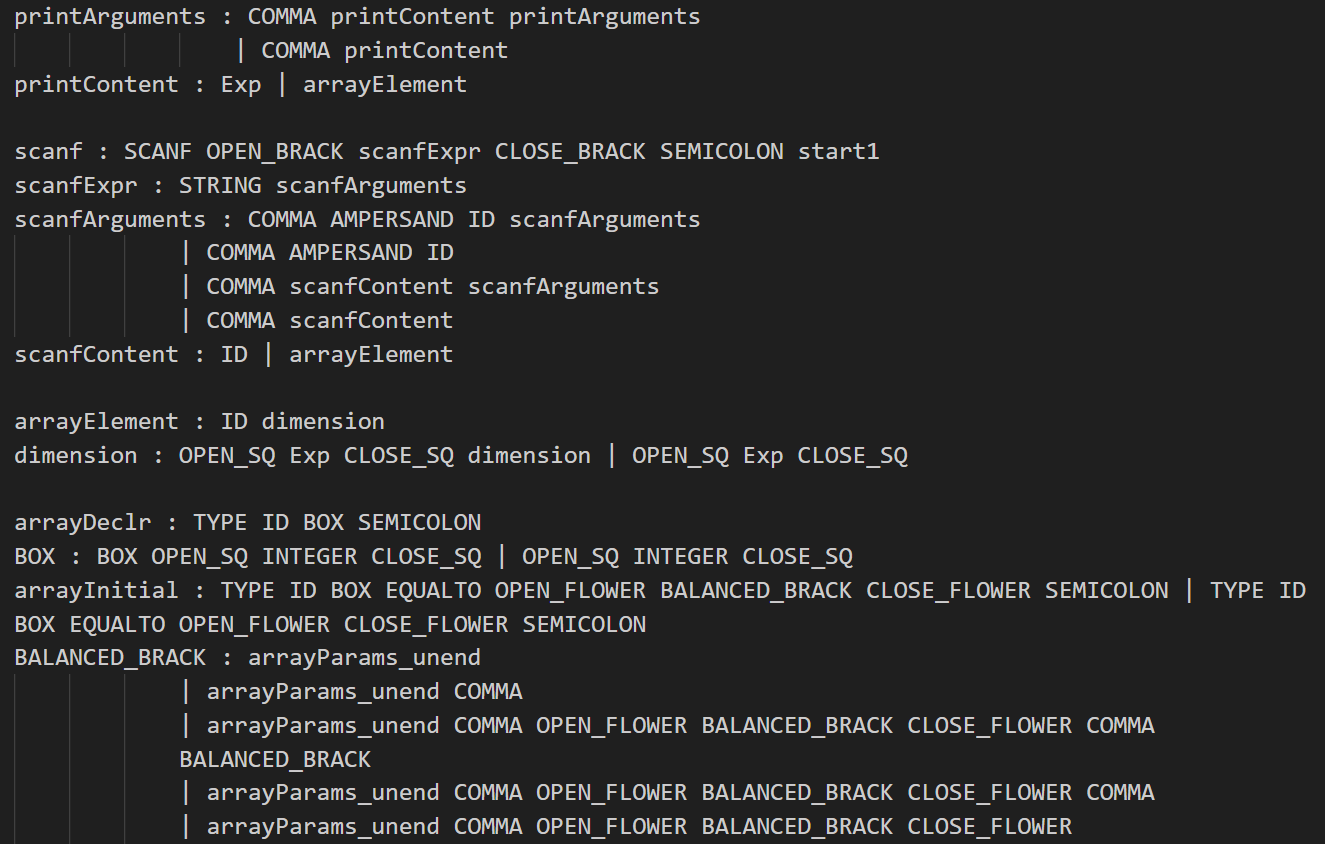
**1. YACC File (.y)**

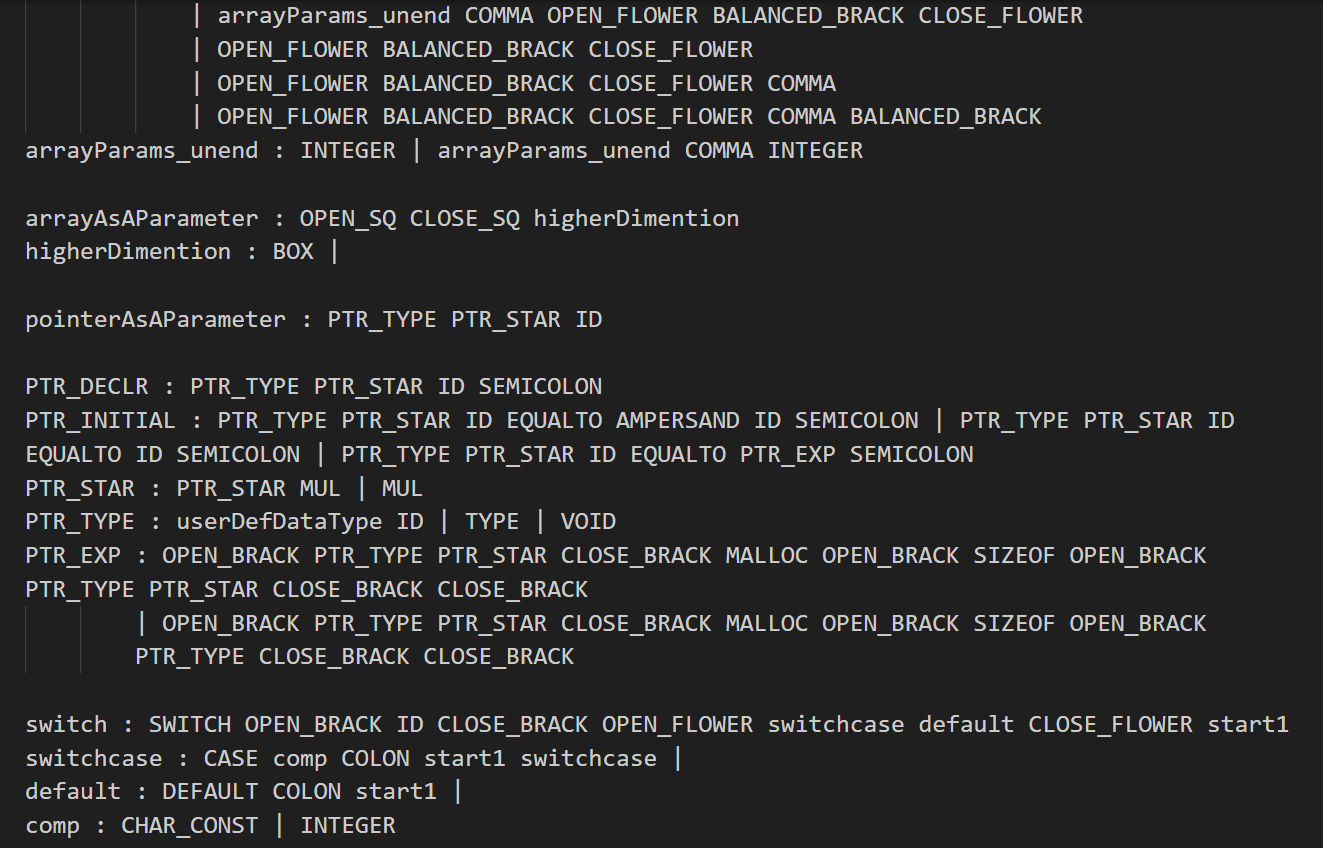


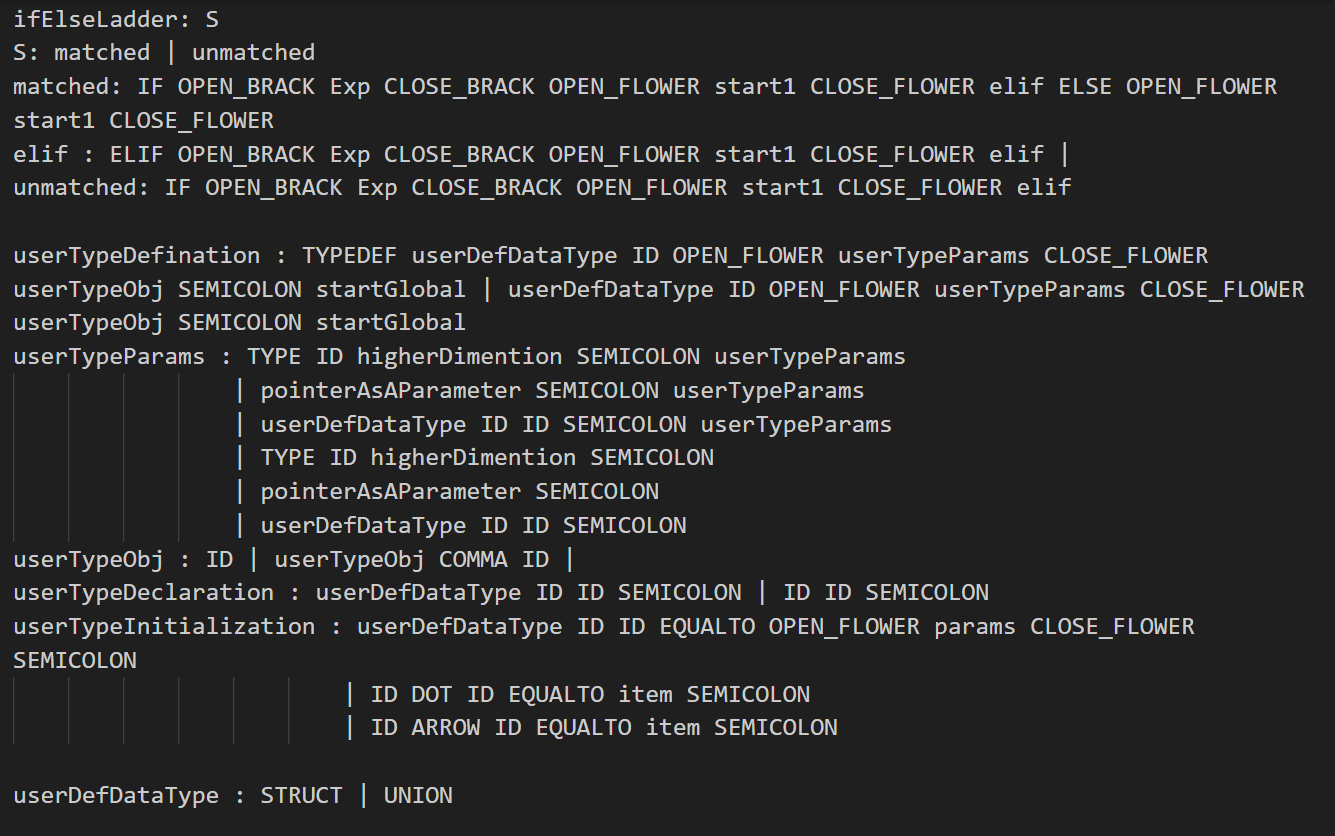


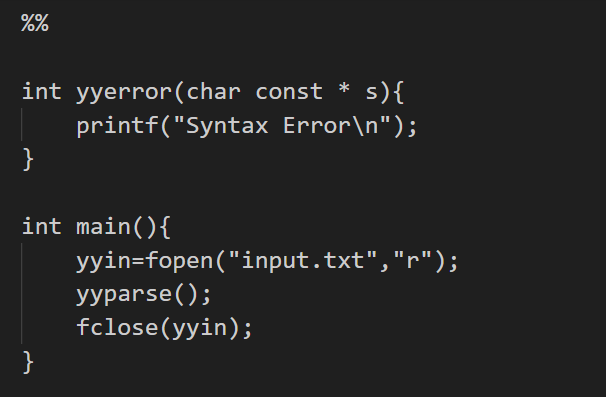












%{

#include <stdio.h>

int yylex();

extern FILE\*yyin;

int yyerror(char const \* s);

extern char \*yytext;

%}

%token INCLUDE PREDEF\_HEADER STRING ELIF ELSE IF BREAK NOT FOR CONTINUE WHILE TYPE SWITCH CASE STRUCT UNION RETURN ID INTEGER FLOATING\_NUM SL\_COMMENT ML\_COMMENT CHAR\_CONST EQUALTO OPEN\_BRACK OPEN\_FLOWER OPEN\_SQ CLOSE\_BRACK CLOSE\_FLOWER CLOSE\_SQ AND UNARY\_OP PLUS MINUS DIV MUL MOD OR AMPERSAND BIT\_OR BIT\_XOR SEMICOLON COMMA ISEQUALTO LT LTE GT GTE NE PLUS\_ET MINUS\_ET MUL\_ET DIV\_ET OR\_ET AND\_ET XOR\_ET PRINTF SCANF MAIN FUN\_START COLON DEFAULT VOID MALLOC SIZEOF TYPEDEF DOT ARROW

%right XOR\_ET OR\_ET AND\_ET

%right PLUS\_ET MINUS\_ET MUL\_ET DIV\_ET EQUALTO

%left OR

%left AND

%left BIT\_OR

%left BIT\_XOR

%left AMPERSAND

%left ISEQUALTO NE

%left GT GTE

%left LT LTE

%left PLUS MINUS

%left MUL DIV MOD

%left NOT

%left UNARY\_OP

%left OPEN\_BRACK CLOSE\_BRACK

%%

start: header startGlobal main {printf("Syntax is Correct\n");}

startGlobal : function\_defn | globalVarDec | userTypeDefination | userTypeDeclaration startGlobal |

globalVarDec : TYPE ID optionsG DG SEMICOLON startGlobal

| ID optionsG DG SEMICOLON startGlobal

DG : COMMA ID optionsG DG |

optionsG : EQUALTO Exp |

main: MAIN mainParameters CLOSE\_BRACK OPEN\_FLOWER start1 CLOSE\_FLOWER

mainParameters : parameter\_list |

start1: print | scanf | function\_call | varDec | while | for

| RETURN ExpF SEMICOLON start1 | CONTINUE SEMICOLON start1| BREAK SEMICOLON start1

| SL\_COMMENT start1 | ML\_COMMENT start1 | arrayDeclr start1 | arrayInitial start1

| switch | PTR\_INITIAL start1 | PTR\_DECLR start1 | ifElseLadder start1

| userTypeDeclaration start1 | userTypeInitialization start1 |

varDec : TYPE ID options D SEMICOLON start1

| ID options D SEMICOLON start1

| UNARY\_OP Exp SEMICOLON start1

| Exp UNARY\_OP SEMICOLON start1

D : COMMA ID options D |

options : EQUALTO Exp | EQUALTO ID OPEN\_BRACK params CLOSE\_BRACK | ID OPEN\_BRACK CLOSE\_BRACK |

header: INCLUDE file newHeader

newHeader: header |

file : STRING | PREDEF\_HEADER

Exp: OPEN\_BRACK Exp CLOSE\_BRACK | Exp PLUS Exp | Exp MINUS Exp | Exp MUL Exp | Exp DIV Exp | Exp MOD Exp | Exp OR Exp | Exp AND Exp | Exp BIT\_XOR Exp | Exp AMPERSAND Exp | Exp UNARY\_OP | UNARY\_OP Exp | NOT Exp | Exp BIT\_OR Exp | Exp ISEQUALTO Exp| Exp LT Exp| Exp LTE Exp| Exp GT Exp| Exp GTE Exp| Exp NE Exp| Exp PLUS\_ET Exp| Exp MINUS\_ET Exp| Exp MUL\_ET Exp| Exp DIV\_ET Exp| Exp OR\_ET Exp| Exp AND\_ET Exp| Exp XOR\_ET Exp|ID | INTEGER

function\_defn : function\_declaration OPEN\_FLOWER start1 CLOSE\_FLOWER startGlobal

function\_declaration: FUN\_START parameter\_list CLOSE\_BRACK

| FUN\_START CLOSE\_BRACK

parameter\_list: parameter\_list COMMA TYPE ID choice

| parameter\_list COMMA pointerAsAParameter

| pointerAsAParameter

| TYPE ID choice

choice : arrayAsAParameter |

function\_call : ID OPEN\_BRACK params CLOSE\_BRACK SEMICOLON start1

| ID OPEN\_BRACK CLOSE\_BRACK SEMICOLON start1

params : item | params COMMA item

item : ID | INTEGER | STRING | CHAR\_CONST | FLOATING\_NUM

for : FOR OPEN\_BRACK varDecF ExpF SEMICOLON ExpF CLOSE\_BRACK OPEN\_FLOWER start1 CLOSE\_FLOWER start1

| FOR OPEN\_BRACK varDecF ExpF SEMICOLON ExpF CLOSE\_BRACK SEMICOLON start1

| FOR OPEN\_BRACK SEMICOLON ExpF SEMICOLON ExpF CLOSE\_BRACK OPEN\_FLOWER start1 CLOSE\_FLOWER start1

| FOR OPEN\_BRACK SEMICOLON ExpF SEMICOLON ExpF CLOSE\_BRACK SEMICOLON start1

while : WHILE OPEN\_BRACK ExpF CLOSE\_BRACK OPEN\_FLOWER start1 CLOSE\_FLOWER start1

| WHILE OPEN\_BRACK ExpF CLOSE\_BRACK SEMICOLON start1

varDecF : TYPE ID options D SEMICOLON

| ID options D SEMICOLON

ExpF : Exp |

print : PRINTF OPEN\_BRACK printExpr CLOSE\_BRACK SEMICOLON start1

printExpr : STRING

| STRING printArguments

printArguments : COMMA printContent printArguments

| COMMA printContent

printContent : Exp | arrayElement

scanf : SCANF OPEN\_BRACK scanfExpr CLOSE\_BRACK SEMICOLON start1

scanfExpr : STRING scanfArguments

scanfArguments : COMMA AMPERSAND ID scanfArguments

| COMMA AMPERSAND ID

| COMMA scanfContent scanfArguments

| COMMA scanfContent

scanfContent : ID | arrayElement

arrayElement : ID dimension

dimension : OPEN\_SQ Exp CLOSE\_SQ dimension | OPEN\_SQ Exp CLOSE\_SQ

arrayDeclr : TYPE ID BOX SEMICOLON

BOX : BOX OPEN\_SQ INTEGER CLOSE\_SQ | OPEN\_SQ INTEGER CLOSE\_SQ

arrayInitial : TYPE ID BOX EQUALTO OPEN\_FLOWER BALANCED\_BRACK CLOSE\_FLOWER SEMICOLON | TYPE ID BOX EQUALTO OPEN\_FLOWER CLOSE\_FLOWER SEMICOLON

BALANCED\_BRACK : arrayParams\_unend

| arrayParams\_unend COMMA

| arrayParams\_unend COMMA OPEN\_FLOWER BALANCED\_BRACK CLOSE\_FLOWER COMMA BALANCED\_BRACK

| arrayParams\_unend COMMA OPEN\_FLOWER BALANCED\_BRACK CLOSE\_FLOWER COMMA

| arrayParams\_unend COMMA OPEN\_FLOWER BALANCED\_BRACK CLOSE\_FLOWER

| OPEN\_FLOWER BALANCED\_BRACK CLOSE\_FLOWER

| OPEN\_FLOWER BALANCED\_BRACK CLOSE\_FLOWER COMMA

| OPEN\_FLOWER BALANCED\_BRACK CLOSE\_FLOWER COMMA BALANCED\_BRACK

arrayParams\_unend : INTEGER | arrayParams\_unend COMMA INTEGER

arrayAsAParameter : OPEN\_SQ CLOSE\_SQ higherDimention

higherDimention : BOX |

pointerAsAParameter : PTR\_TYPE PTR\_STAR ID

PTR\_DECLR : PTR\_TYPE PTR\_STAR ID SEMICOLON

PTR\_INITIAL : PTR\_TYPE PTR\_STAR ID EQUALTO AMPERSAND ID SEMICOLON | PTR\_TYPE PTR\_STAR ID EQUALTO ID SEMICOLON | PTR\_TYPE PTR\_STAR ID EQUALTO PTR\_EXP SEMICOLON

PTR\_STAR : PTR\_STAR MUL | MUL

PTR\_TYPE : userDefDataType ID | TYPE | VOID

PTR\_EXP : OPEN\_BRACK PTR\_TYPE PTR\_STAR CLOSE\_BRACK MALLOC OPEN\_BRACK SIZEOF OPEN\_BRACK PTR\_TYPE PTR\_STAR CLOSE\_BRACK CLOSE\_BRACK

| OPEN\_BRACK PTR\_TYPE PTR\_STAR CLOSE\_BRACK MALLOC OPEN\_BRACK SIZEOF OPEN\_BRACK PTR\_TYPE CLOSE\_BRACK CLOSE\_BRACK

switch : SWITCH OPEN\_BRACK ID CLOSE\_BRACK OPEN\_FLOWER switchcase default CLOSE\_FLOWER start1

switchcase : CASE comp COLON start1 switchcase |

default : DEFAULT COLON start1 |

comp : CHAR\_CONST | INTEGER

ifElseLadder: S

S: matched | unmatched

matched: IF OPEN\_BRACK Exp CLOSE\_BRACK OPEN\_FLOWER start1 CLOSE\_FLOWER elif ELSE OPEN\_FLOWER start1 CLOSE\_FLOWER

elif : ELIF OPEN\_BRACK Exp CLOSE\_BRACK OPEN\_FLOWER start1 CLOSE\_FLOWER elif |

unmatched: IF OPEN\_BRACK Exp CLOSE\_BRACK OPEN\_FLOWER start1 CLOSE\_FLOWER elif

userTypeDefination : TYPEDEF userDefDataType ID OPEN\_FLOWER userTypeParams CLOSE\_FLOWER userTypeObj SEMICOLON startGlobal | userDefDataType ID OPEN\_FLOWER userTypeParams CLOSE\_FLOWER userTypeObj SEMICOLON startGlobal

userTypeParams : TYPE ID higherDimention SEMICOLON userTypeParams

| pointerAsAParameter SEMICOLON userTypeParams

| userDefDataType ID ID SEMICOLON userTypeParams

| TYPE ID higherDimention SEMICOLON

| pointerAsAParameter SEMICOLON

| userDefDataType ID ID SEMICOLON

userTypeObj : ID | userTypeObj COMMA ID |

userTypeDeclaration : userDefDataType ID ID SEMICOLON | ID ID SEMICOLON

userTypeInitialization : userDefDataType ID ID EQUALTO OPEN\_FLOWER params CLOSE\_FLOWER SEMICOLON

| ID DOT ID EQUALTO item SEMICOLON

| ID ARROW ID EQUALTO item SEMICOLON

userDefDataType : STRUCT | UNION

%%

int yyerror(char const \* s){

printf("Syntax Error\n");

}

int main(){

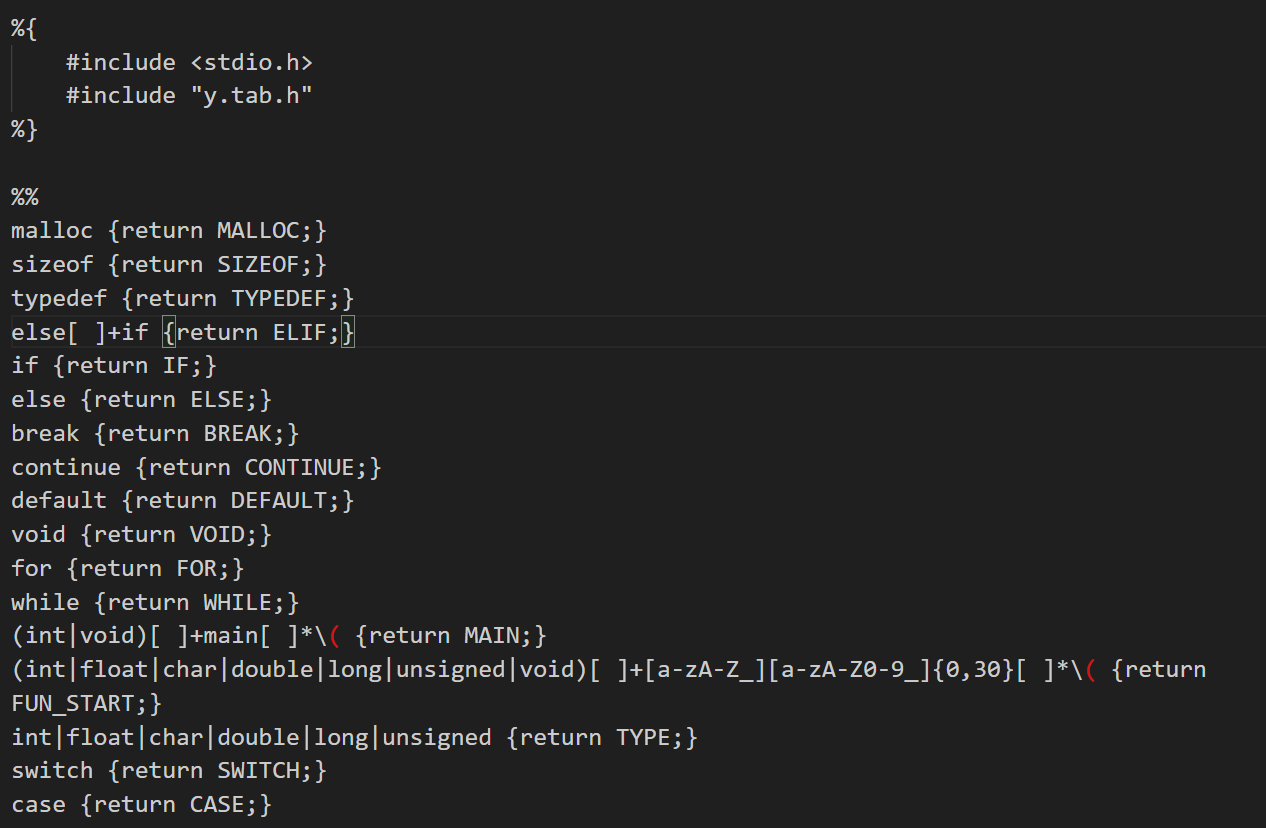
yyin=fopen("input.txt","r");

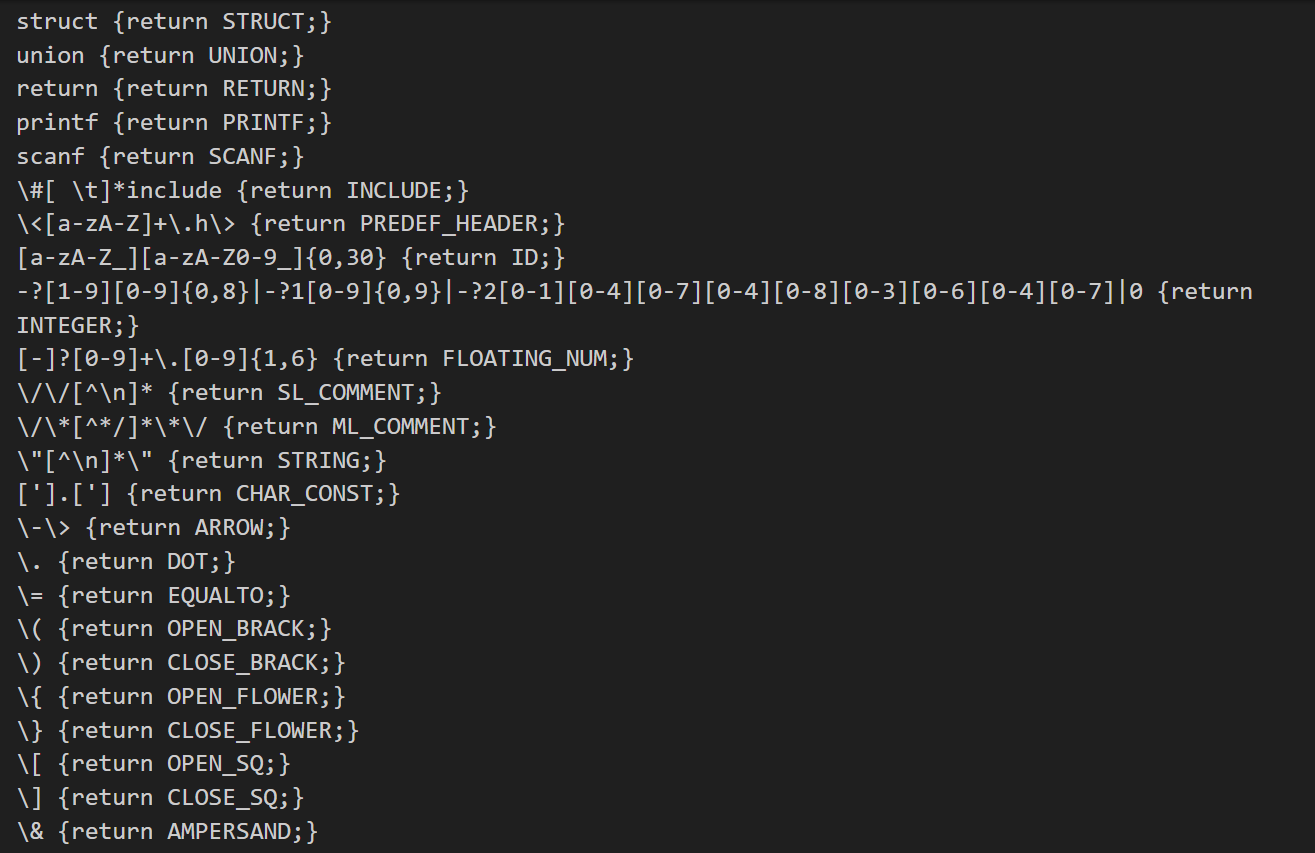
yyparse();

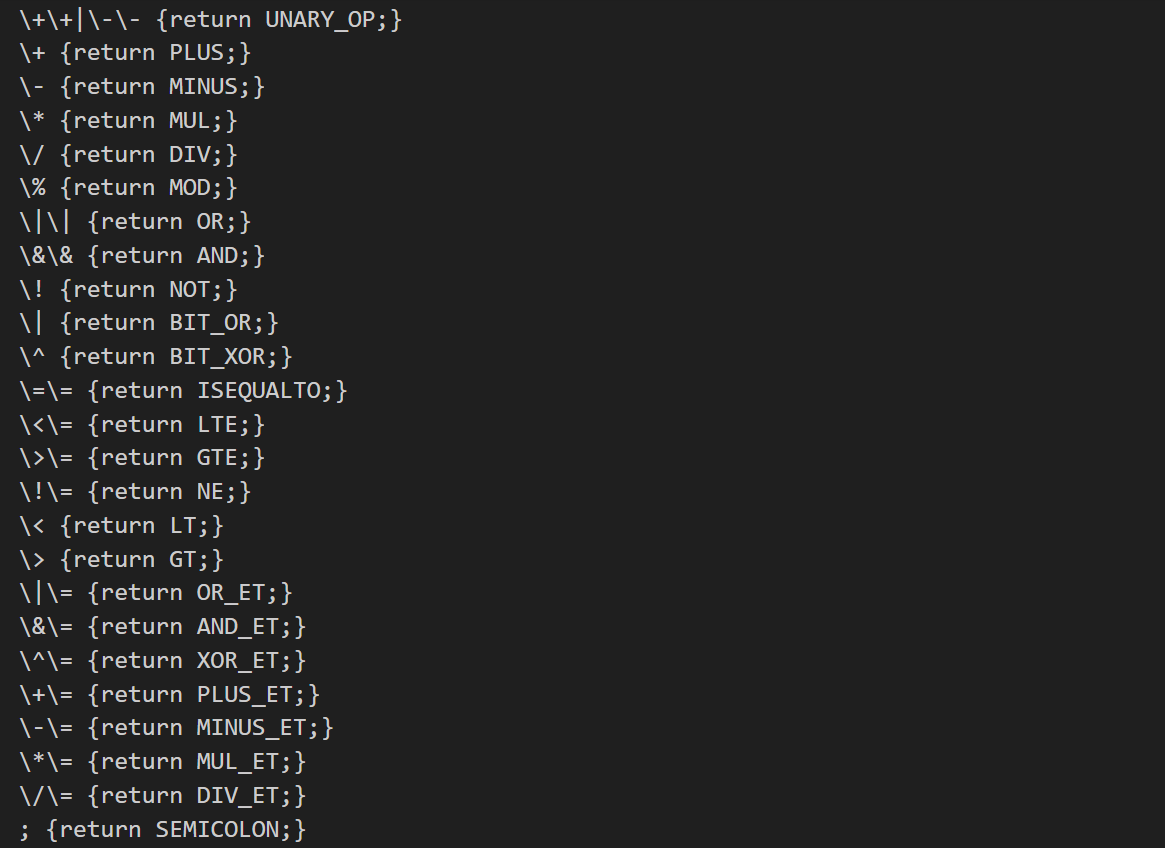
fclose(yyin);

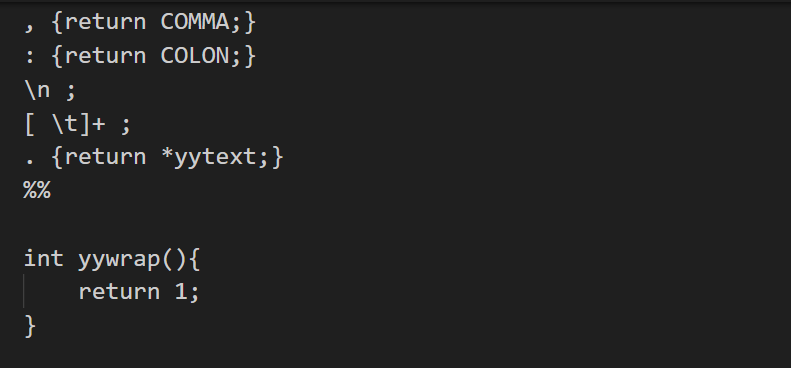
}

**LEX File (.l)**

****

****

****

****

%{

#include <stdio.h>

#include "y.tab.h"

%}

%%

malloc {return MALLOC;}

sizeof {return SIZEOF;}

typedef {return TYPEDEF;}

else[ ]+if {return ELIF;}

if {return IF;}

else {return ELSE;}

break {return BREAK;}

continue {return CONTINUE;}

default {return DEFAULT;}

void {return VOID;}

for {return FOR;}

while {return WHILE;}

(int|void)[ ]+main[ ]\*\( {return MAIN;}

(int|float|char|double|long|unsigned|void)[ ]+[a-zA-Z\_][a-zA-Z0-9\_]{0,30}[ ]\*\( {return FUN\_START;}

int|float|char|double|long|unsigned {return TYPE;}

switch {return SWITCH;}

case {return CASE;}

struct {return STRUCT;}

union {return UNION;}

return {return RETURN;}

printf {return PRINTF;}

scanf {return SCANF;}

\#[ \t]\*include {return INCLUDE;}

\<[a-zA-Z]+\.h\> {return PREDEF\_HEADER;}

[a-zA-Z\_][a-zA-Z0-9\_]{0,30} {return ID;}

-?[1-9][0-9]{0,8}|-?1[0-9]{0,9}|-?2[0-1][0-4][0-7][0-4][0-8][0-3][0-6][0-4][0-7]|0 {return INTEGER;}

[-]?[0-9]+\.[0-9]{1,6} {return FLOATING\_NUM;}

\/\/[^\n]\* {return SL\_COMMENT;}

\/\\*[^\*/]\*\\*\/ {return ML\_COMMENT;}

\"[^\n]\*\" {return STRING;}

['].['] {return CHAR\_CONST;}

\-\> {return ARROW;}

\. {return DOT;}

\= {return EQUALTO;}

\( {return OPEN\_BRACK;}

\) {return CLOSE\_BRACK;}

\{ {return OPEN\_FLOWER;}

\} {return CLOSE\_FLOWER;}

\[ {return OPEN\_SQ;}

\] {return CLOSE\_SQ;}

\& {return AMPERSAND;}

\+\+|\-\- {return UNARY\_OP;}

\+ {return PLUS;}

\- {return MINUS;}

\\* {return MUL;}

\/ {return DIV;}

\% {return MOD;}

\|\| {return OR;}

\&\& {return AND;}

\! {return NOT;}

\| {return BIT\_OR;}

\^ {return BIT\_XOR;}

\=\= {return ISEQUALTO;}

\<\= {return LTE;}

\>\= {return GTE;}

\!\= {return NE;}

\< {return LT;}

\> {return GT;}

\|\= {return OR\_ET;}

\&\= {return AND\_ET;}

\^\= {return XOR\_ET;}

\+\= {return PLUS\_ET;}

\-\= {return MINUS\_ET;}

\\*\= {return MUL\_ET;}

\/\= {return DIV\_ET;}

; {return SEMICOLON;}

, {return COMMA;}

: {return COLON;}

\n ;

[ \t]+ ;

. {return \*yytext;}

%%

int yywrap(){

return 1;

}

**Various Structures of C Programming Language Implemented for Syntax Checking**

1. Syntax for Header File

2. Main Function

3. Function Definition + Declaration

4. Function call

5. Printf and Scanf

6. Variable Declaration

7. Array Declaration

8. if else ladder

9. for loop

10. while loop

11. Expressions

12. Comments

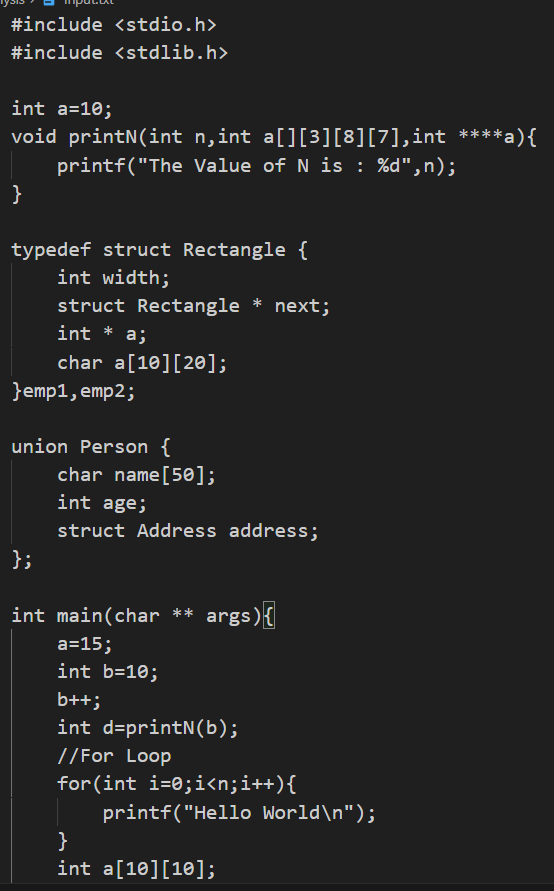
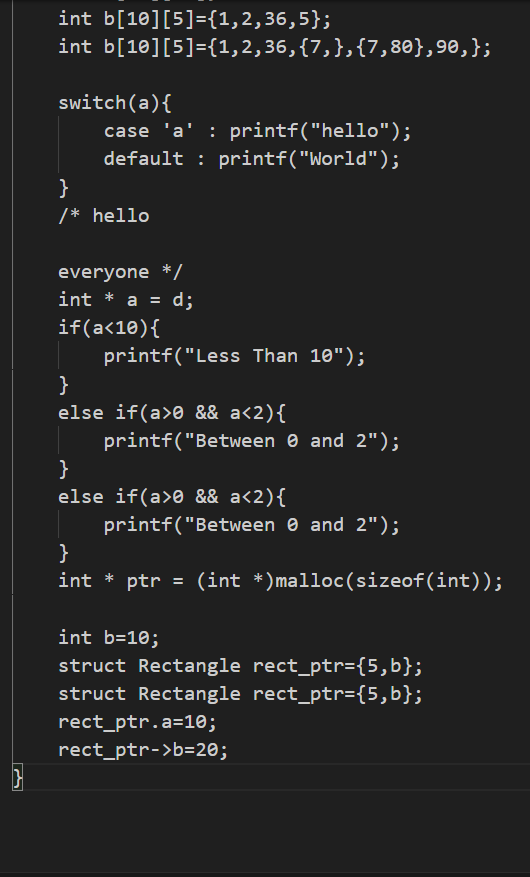
13. Switch case

14. Pointer

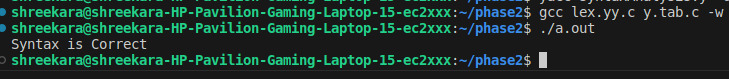
15. Structures and Unions

**INPUT TEST CASES**

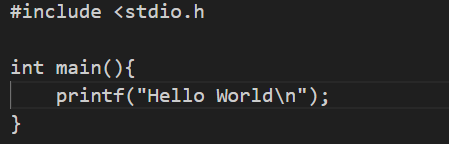
Test Case 1: ALL CORRECT

**OUTPUT**

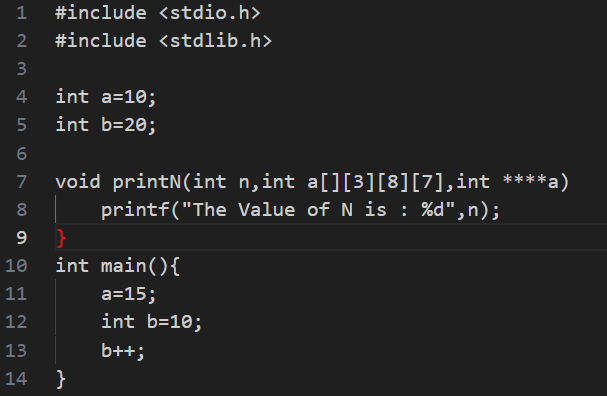


TEST CASE 2: ERROR IN Header File Syntax



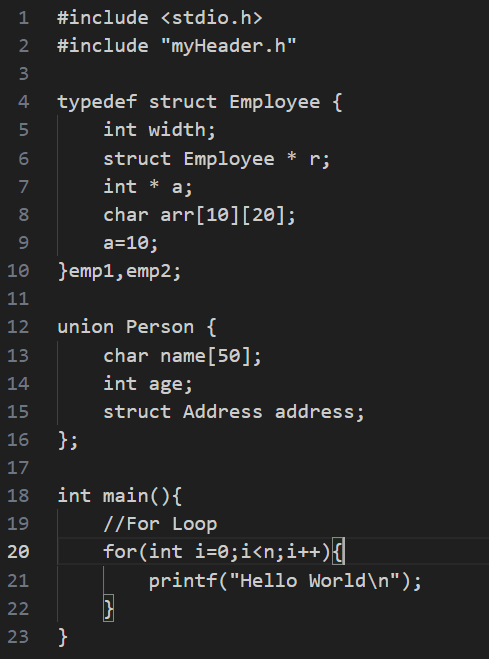
**OUTPUT**

TEST CASE 3: ERROR IN Function Definition Syntax



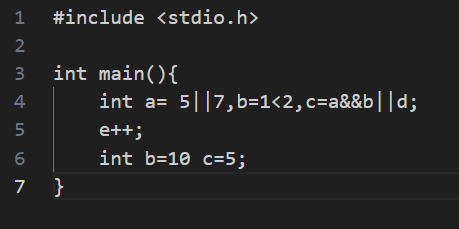
**OUTPUT**

TEST CASE 4: ERROR IN Struct Syntax



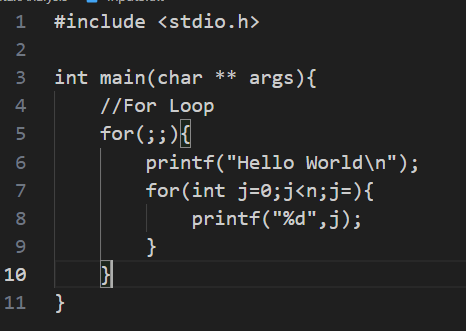
**OUTPUT**

TEST CASE 5: ERROR IN Expression Syntax



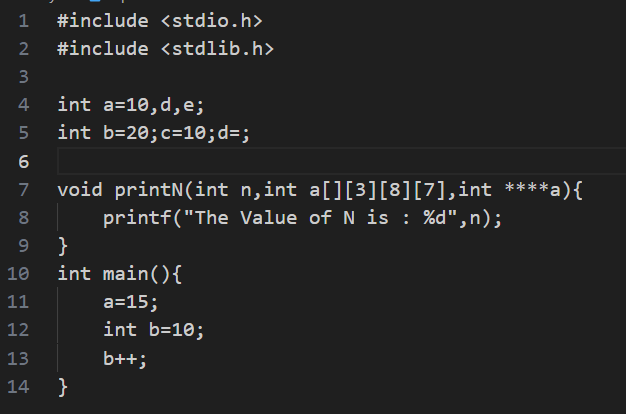
**OUTPUT**

TEST CASE 6: ERROR IN FOR Loop Syntax



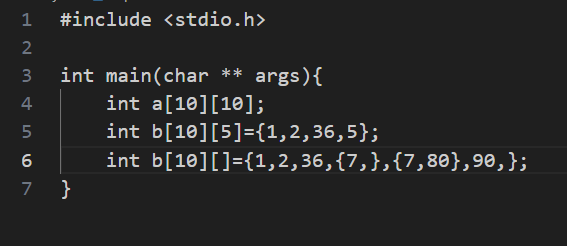
**OUTPUT**

TEST CASE 7: ERROR IN Variable Declaration Syntax



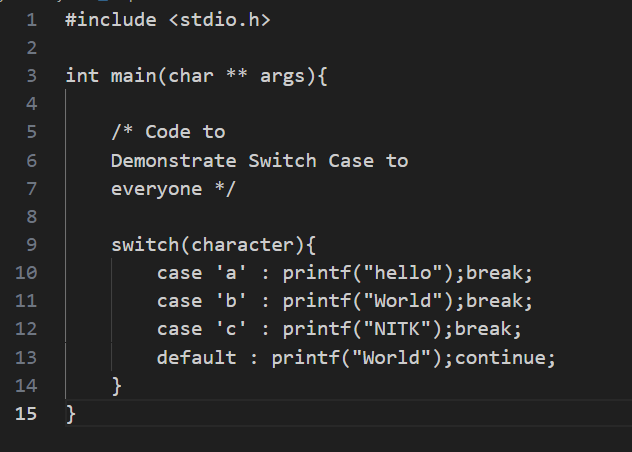
**OUTPUT**

TEST CASE 8: ERROR IN Array Declaration Syntax



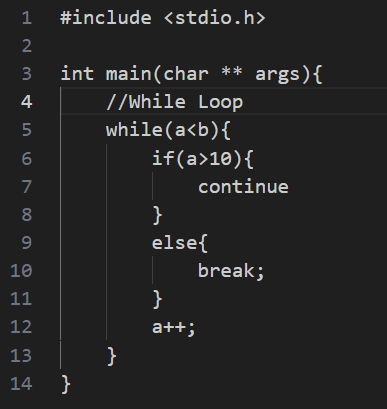
**OUTPUT**

TEST CASE 9: ERROR IN Switch Case Syntax



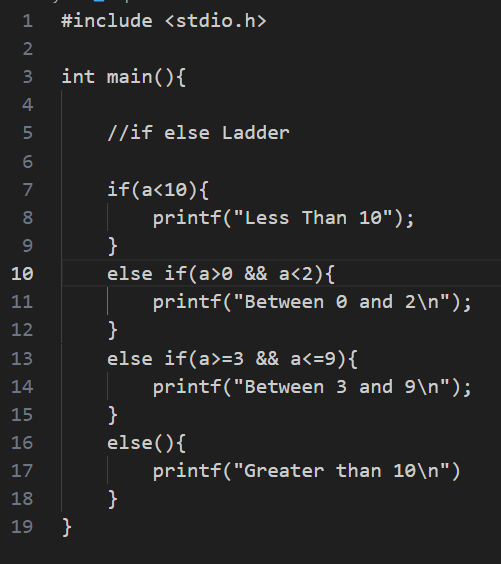
**OUTPUT**

TEST CASE 10: ERROR IN While Loop Syntax



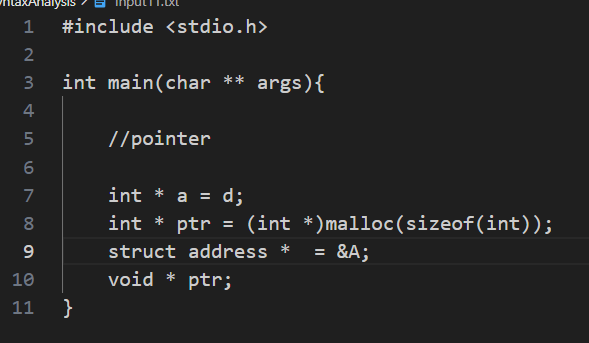
**OUTPUT**

TEST CASE 11: ERROR IN If Else Ladder Syntax



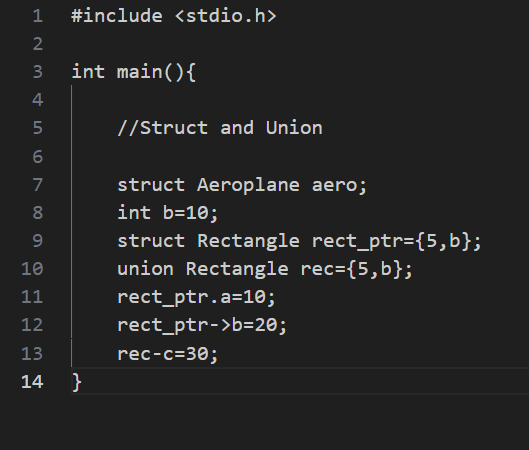
**OUTPUT**

TEST CASE 12: ERROR IN Pointer Declaration Syntax

****

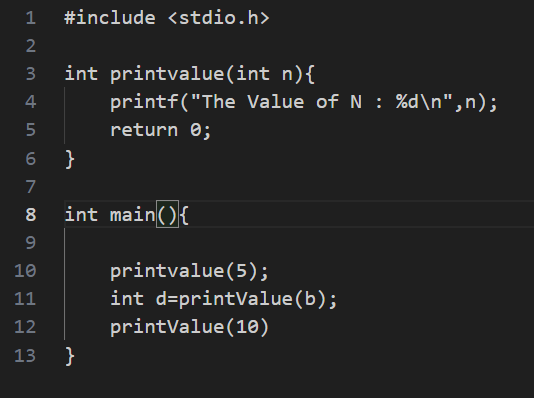
**OUTPUT**

TEST CASE 13: ERROR IN Structure and Union Initialization and Declaration Syntax



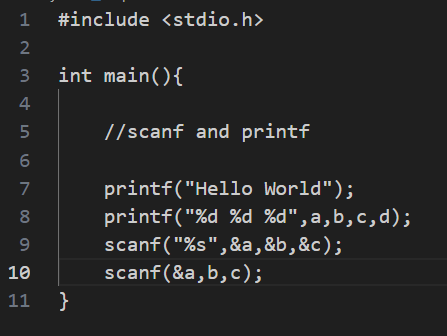
**OUTPUT**

TEST CASE 14: ERROR IN function call statements Syntax



**OUTPUT**

TEST CASE 15: ERROR IN Printf and Scanf Statements Syntax

****

**OUTPUT**