Date : 22/03/2022

CSPC62 : COMPILER DESIGN
LAB-5

Roll no. : **106119100**

Name : **Rajneesh Pandey**

Section : **CSE-B**

Construct Syntax tree and generate intermediate code for assignment statement and expressions in C.

Code:

lexer.l

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

%%
[\t ] ;
[0-9]+ {yylval.symbol = (char)(yytext[0]);return NUMBER;}
[a-z] {yylval.symbol = (char)(yytext[0]);return LETTER;}
">=" {return GE;}
"<=" {return LE;}
"==" {return EQ;}
"!=" {return NE;}
"&&" {return AND;}
"||" {return OR;}
. {return yytext[0];}
\n {return 0;}
%%

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

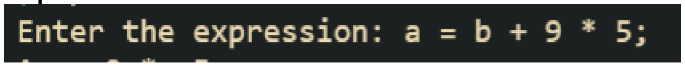
parser.y

```
%{
    #include<string.h>
    #include<stdio.h>
    int yylex(void);
    int yyerror(const char *s);
    char addtotable(char,char,char[]);
    int index1=0;
    char temp = 'A'-1;
    struct expr{
    char operand1;
    char operand2;
    char operator[2];
    char result;
};
%}

%union{
    char symbol;
}


    %left OR
    %left AND
    %left GE LE NE EQ
    %left '<' '>'
    %left '+' '-'
    %left '/' '*'
    %token GE NE LE EQ AND OR
    %token <symbol> LETTER NUMBER
    %type <symbol> exp
    %start statement
%%
```

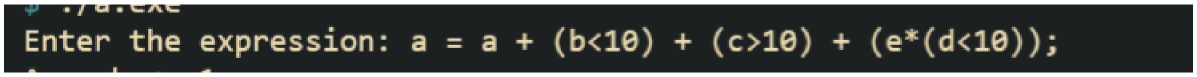
```
statement: LETTER '=' exp ';' {addtotable((char)$1,(char)$3, "=");};
exp: exp '+' exp {$$ = addtotable((char)$1,(char)$3,"+");}
|exp '-' exp {$$ = addtotable((char)$1,(char)$3,"-");}
|exp '/' exp {$$ = addtotable((char)$1,(char)$3,"/");}
|exp '*' exp {$$ = addtotable((char)$1,(char)$3,"*");}
|exp '<' exp {$$ = addtotable((char)$1,(char)$3,"<");}
|exp '>' exp {$$ = addtotable((char)$1,(char)$3,">");}
|exp AND exp {$$ = addtotable((char)$1, (char)$3, "&&");}
|exp OR exp {$$ = addtotable((char)$1, (char)$3, "||");}
|exp GE exp {$$ = addtotable((char)$1, (char)$3, ">=");}
|exp LE exp {$$ = addtotable((char)$1, (char)$3, "<=");}
|exp NE exp {$$ = addtotable((char)$1, (char)$3, "!=");}
|exp EQ exp {$$ = addtotable((char)$1, (char)$3, "==");}
|'(' exp ')' {$$ = (char)$2;}
|NUMBER {$$ = (char)$1;}
|LETTER {$$ = (char)$1;}
;
%%
struct expr arr[20];

int yyerror(const char *s){
    printf("%s",s);
}

char addtotable(char a, char b, char o[]){
    temp++;
    arr[index1].operand1 = a;
    arr[index1].operand2 = b;
    strcpy(arr[index1].operator, o);
    arr[index1].result=temp;
    index1++;
    return temp;
}
```

```c
void threeAdd(){
    int i=0;
    while(i<index1){
        printf("%c := ",arr[i].result);
        printf("%c ",arr[i].operand1);
        printf("%c%c ",arr[i].operator[0], arr[i].operator[1]);
        printf("%c",arr[i].operand2);
        i++;
        printf("\n");
    }
}
int main(){
    printf("Enter the expression: ");
    yyparse();
    threeAdd();
    printf("\n");
    return 0;
}
```

Input:

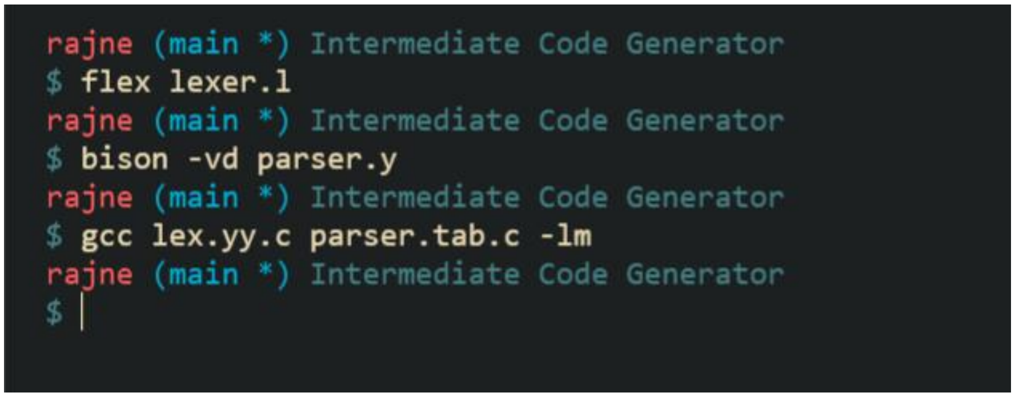


```
Enter the expression: a = b + 9 * 5;
```



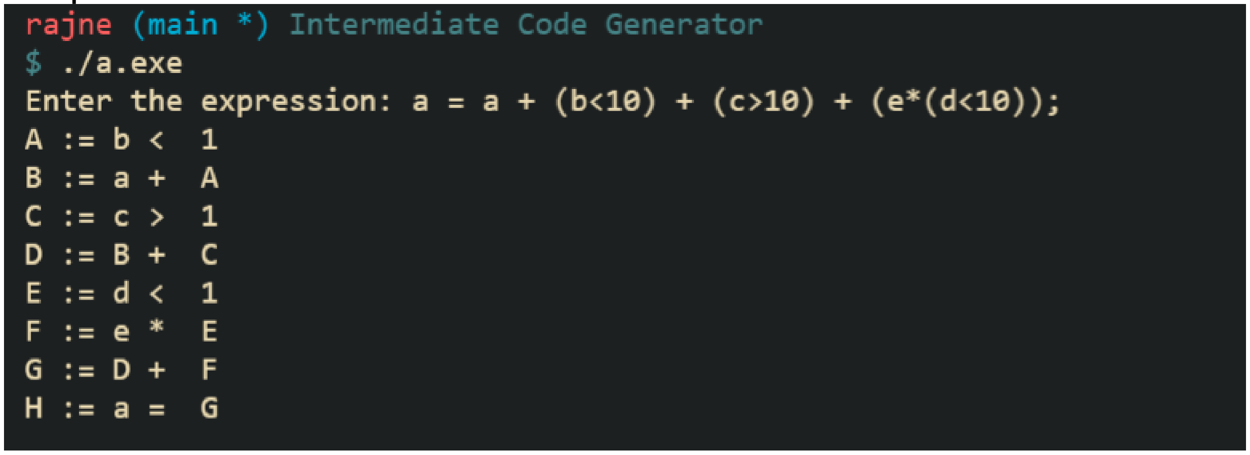
```
Enter the expression: a = a + (b<10) + (c>10) + (e*(d<10));
```

Run :

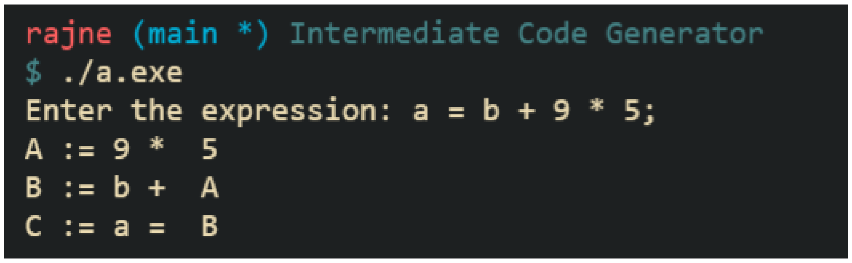


```
rajne (main *) Intermediate Code Generator
$ flex lexer.l
rajne (main *) Intermediate Code Generator
$ bison -vd parser.y
rajne (main *) Intermediate Code Generator
$ gcc lex.yy.c parser.tab.c -lm
rajne (main *) Intermediate Code Generator
$ |
```

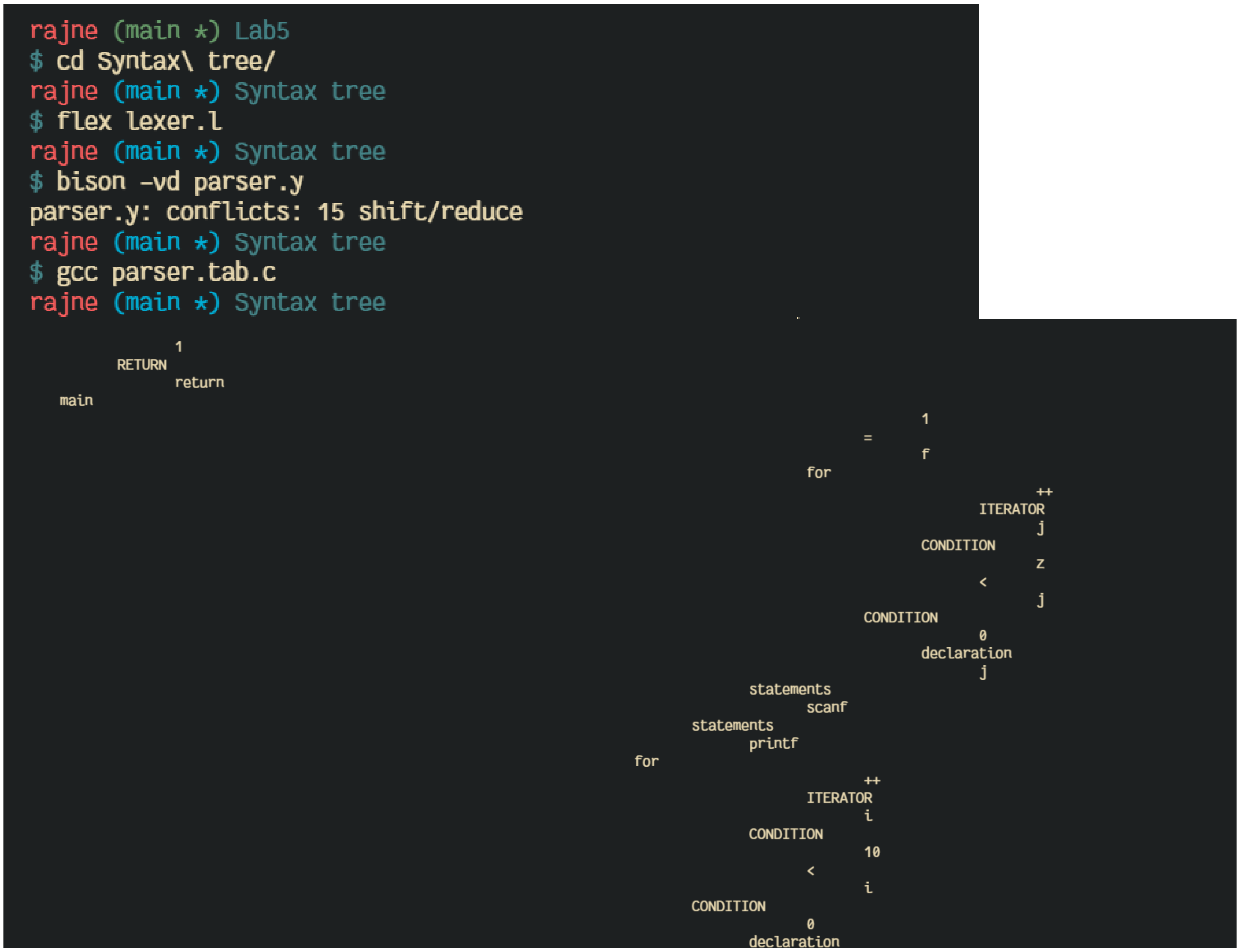
Output :



```
rajne (main *) Intermediate Code Generator
$ ./a.exe
Enter the expression: a = a + (b<10) + (c>10) + (e*(d<10));
A := b <  1
B := a +  A
C := c >  1
D := B +  C
E := d <  1
F := e *  E
G := D +  F
H := a =  G
```

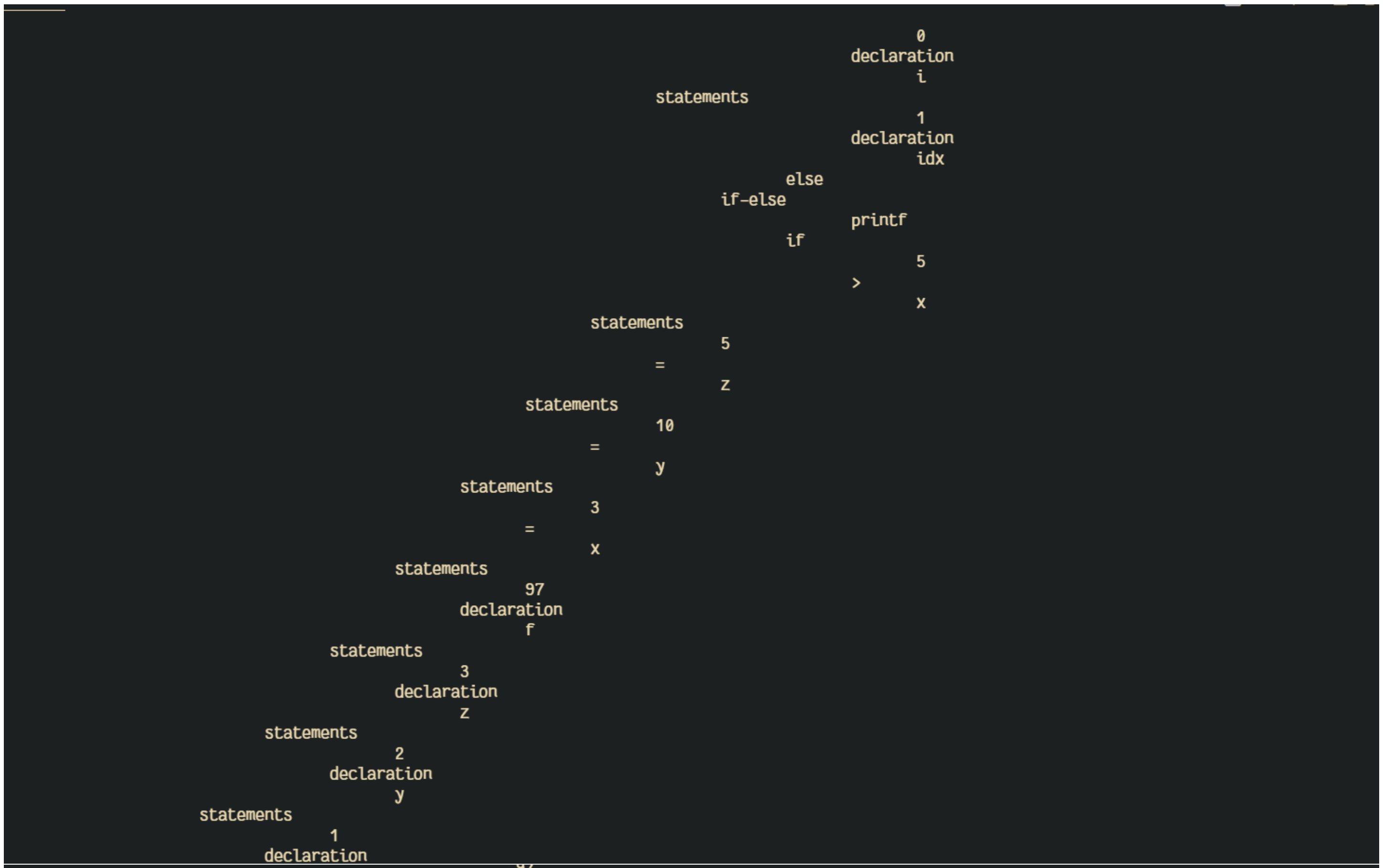


```
rajne (main *) Intermediate Code Generator
$ ./a.exe
Enter the expression: a = b + 9 * 5;
A := 9 *  5
B := b +  A
C := a =  B
```

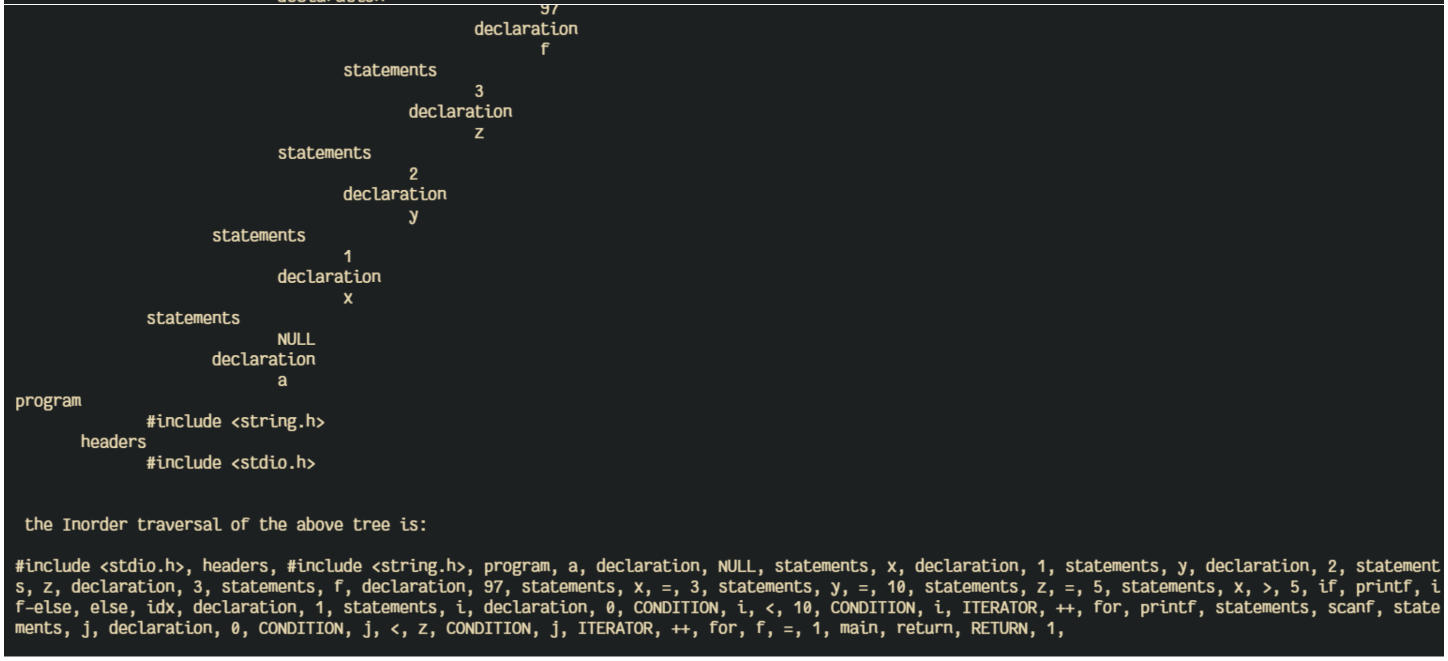


```
rajne (main *) Lab5
$ cd Syntax\ tree/
rajne (main *) Syntax tree
$ flex lexer.l
rajne (main *) Syntax tree
$ bison -vd parser.y
parser.y: conflicts: 15 shift/reduce
rajne (main *) Syntax tree
$ gcc parser.tab.c
rajne (main *) Syntax tree
```

```
            1
        RETURN
            return
    main
                                                                1
                                                            =
                                                                f
                                                        for
                                                                        ++
                                                                    ITERATOR
                                                                        j
                                                                CONDITION
                                                                        z
                                                                    <
                                                                        j
                                                            CONDITION
                                                                    0
                                                                declaration
                                                                    j
                                                    statements
                                                        scanf
                                                statements
                                                    printf
                                            for
                                                            ++
                                                        ITERATOR
                                                            i
                                                    CONDITION
                                                            10
                                                        <
                                                            i
                                                CONDITION
                                                        0
                                                    declaration
```



```
                                                                                                                0
                                                                                                        declaration
                                                                                                                i
                                                                                statements
                                                                                                                1
                                                                                                        declaration
                                                                                                                idx
                                                                                                mispar
                                                                                        if-else
                                                                                                        printf
                                                                                                if
                                                                                                                5
                                                                                                        >
                                                                                                                x
                                                                        statements
                                                                                        5
                                                                                =
                                                                                        z
                                                                statements
                                                                                10
                                                                        =
                                                                                y
                                                        statements
                                                                        3
                                                                =
                                                                        x
                                                statements
                                                                97
                                                        declaration
                                                                f
                                        statements
                                                        3
                                                declaration
                                                        z
                                statements
                                                2
                                        declaration
                                                y
                        statements
                                        1
                                declaration
```



```
                                                                97
                                                        declaration
                                                                f
                                        statements
                                                        3
                                                declaration
                                                        z
                                statements
                                                2
                                        declaration
                                                y
                        statements
                                        1
                                declaration
                                        x
                statements
                                NULL
                        declaration
                                a
program
                #include <string.h>
        headers
                #include <stdio.h>


 the Inorder traversal of the above tree is:

#include <stdio.h>, headers, #include <string.h>, program, a, declaration, NULL, statements, x, declaration, 1, statements, y, declaration, 2, statement
s, z, declaration, 3, statements, f, declaration, 97, statements, x, =, 3, statements, y, =, 10, statements, z, =, 5, statements, x, >, 5, if, printf, i
f-else, else, idx, declaration, 1, statements, i, declaration, 0, CONDITION, i, <, 10, CONDITION, i, ITERATOR, ++, for, printf, statements, scanf, state
ments, j, declaration, 0, CONDITION, j, <, z, CONDITION, j, ITERATOR, ++, for, f, =, 1, main, return, RETURN, 1,
```