NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR

Cachar, Assam

B.Tech. VIth Sem

Subject Code: CS-316

Subject Name: Compiler Design Lab

Submitted By:

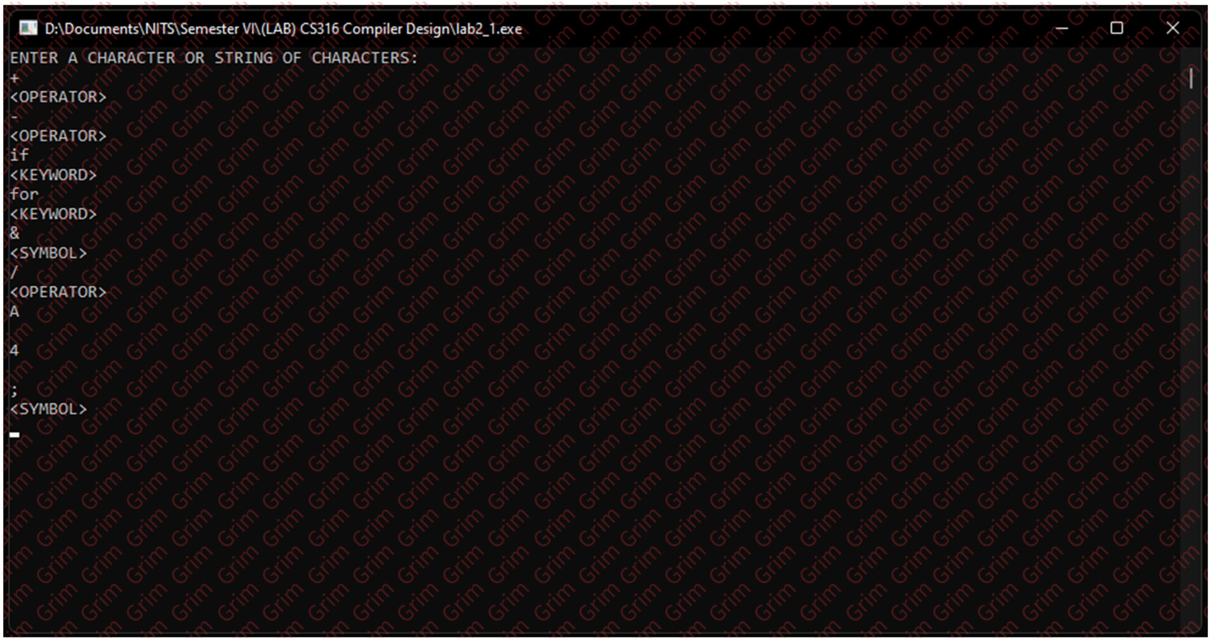
Name	: Subhojit Ghimire
Sch. Id.	: 1912160
Branch	: CSE – B

1. Write a lex program to identify keywords, symbols and operators.

➔ **CODE:**

```
%{
#include <stdio.h>
%}
%%
if|else|for|while|do|switch|int|char|float { printf ("keywords ");}
[a-zA-Z0-9]([a-zA-Z]|[0-9])* {}
"+"|"-"|"*"|"/"|"%" { printf ("operators ");}
. { printf ("symbols ");}
%%
yywrap(){
return 1;
}
main(){
printf ("write something \n");
yylex();
}
```

OUTPUT:


D:\Documents\NITS\Semester VI\(LAB) CS316 Compiler Design\lab2_1.exe

```
ENTER A CHARACTER OR STRING OF CHARACTERS:
+
<OPERATOR>
-
<OPERATOR>
if
<KEYWORD>
for
<KEYWORD>
&
<SYMBOL>
/
<OPERATOR>
A

4

;
<SYMBOL>
```

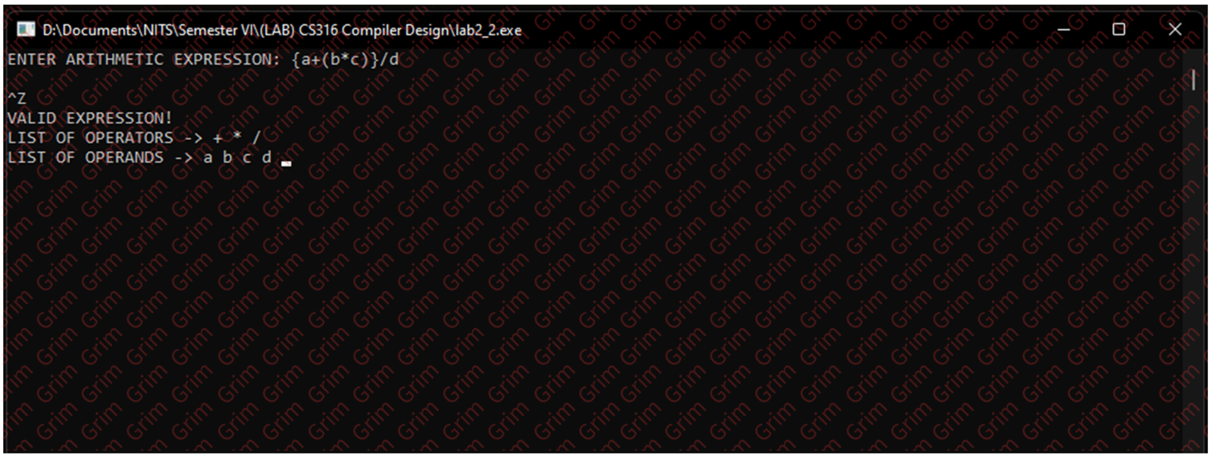
2. Write a lex program, which takes a C program as input, and display the list of identifiers and operators.

➔ **CODE:**

```
%{
#include <stdio.h>
#include <string.h>
int numtor = 0, numand = 0, flag = 1, top = -1;
int l = 0, j = 0;
char operands[10][10], operators[10][10], stack[100];
%}

%%
"(" { top++; stack[top] = '(';}
"{" { top++; stack[top] = '{';}
"[" { top++; stack[top] = '[';}
")" {
if (stack[top] != '(')
flag = 0;
else if (numand > 0 && ((numand - numtor) != 1))
flag = 0;
else
top--;
}
"}" {
if (stack[top] != '{')
flag = 0;
else if (numand > 0 && ((numand - numtor) != 1))
flag = 0;
else
top--;
}
"]" {
if (stack[top] != '[')
flag = 0;
else if (numand > 0 && ((numand - numtor) != 1))
flag = 0;
else
top--;
}
"+"|"-"|"*"|"/" {
numtor++;
strcpy (operators[l], yytext);
l++;
}
[0-9]+|[a-zA-Z][a-zA-Z0-9_]* {
numand++;
strcpy (operands[j], yytext);
j++;
}
```

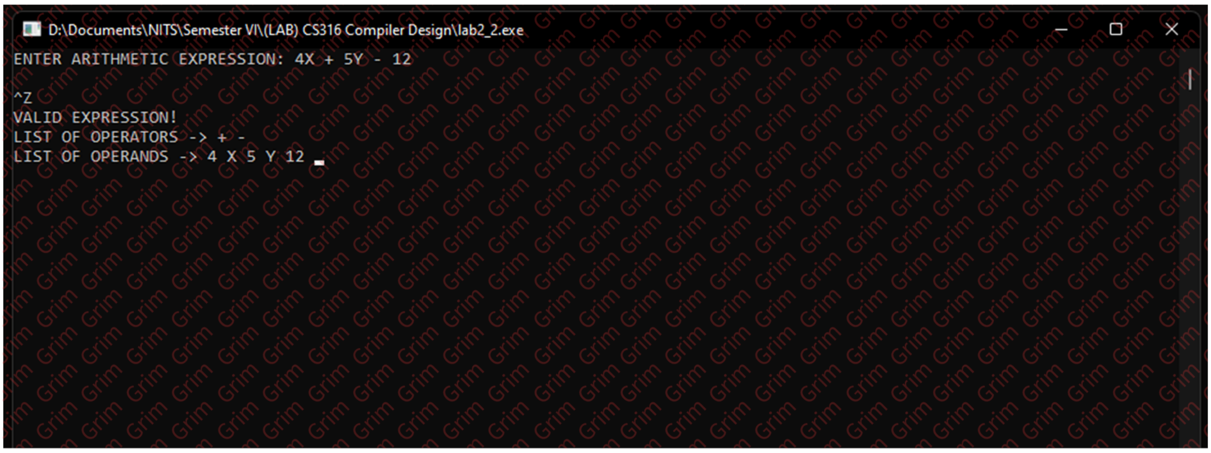
```
    " "|"\n" {}
    %%

    int yywrap(){
    return 1;
    }

    main(){
    printf ("write arithmetic expression\n ");
    yylex ();
    if (flag == 1 && top == -1 && (numand - numtor == 1)) {
    printf ("it's a valid expression\n");
    int i;
    printf ("the operators are:\n ");
    for (i = 0; i < l; ++i)
    printf ("%s\n", operators[i]);
    printf ("\nthe operands are:\n");
    for (i = 0; i < j; ++i)
    printf ("%s\n", operands[i]);
    }
    else
    printf ("it's not a valid expression\n");
    int a;
    scanf("%d", &a);
}
```

OUTPUT:



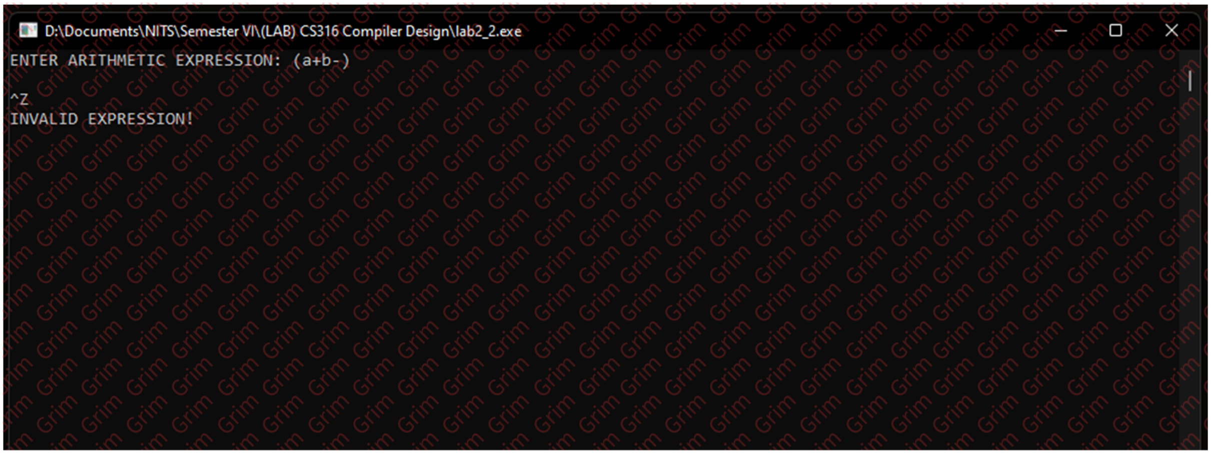
D:\Documents\NITS\Semester VI\(LAB) CS316 Compiler Design\lab2_2.exe

```
ENTER ARITHMETIC EXPRESSION: {a+(b*c)}/d

^Z
VALID EXPRESSION!
LIST OF OPERATORS -> + * /
LIST OF OPERANDS -> a b c d
```



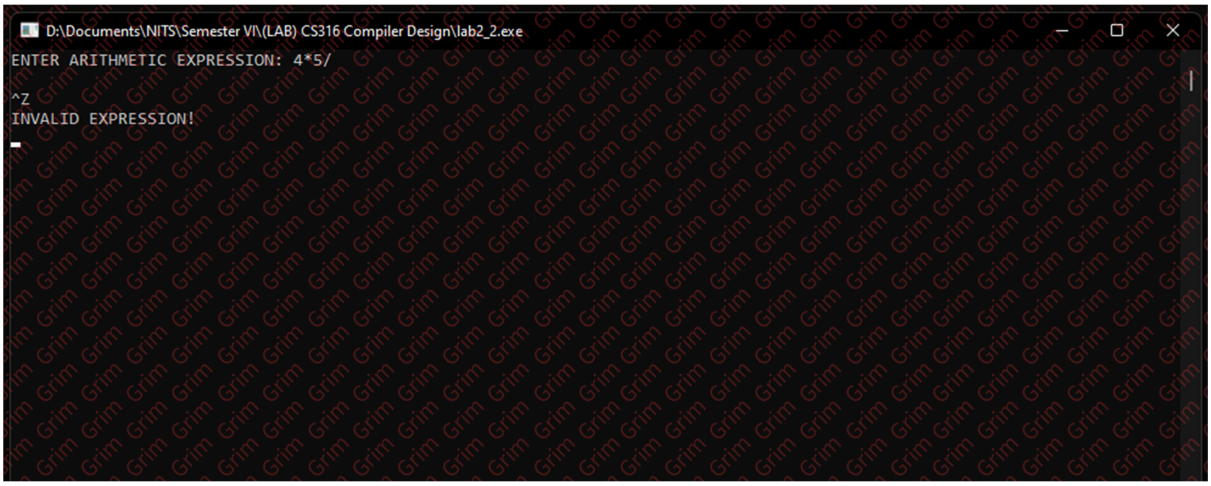
D:\Documents\NITS\Semester VI\(LAB) CS316 Compiler Design\lab2_2.exe

```
ENTER ARITHMETIC EXPRESSION: 4X + 5Y - 12

^Z
VALID EXPRESSION!
LIST OF OPERATORS -> + -
LIST OF OPERANDS -> 4 X 5 Y 12
```



D:\Documents\NITS\Semester VI\(LAB) CS316 Compiler Design\lab2_2.exe

```
ENTER ARITHMETIC EXPRESSION: (a+b-)

^Z
INVALID EXPRESSION!
```



D:\Documents\NITS\Semester VI\(LAB) CS316 Compiler Design\lab2_2.exe

```
ENTER ARITHMETIC EXPRESSION: 4*5/

^Z
INVALID EXPRESSION!
```

NOTE: For linux users, the EOF trigger is Ctrl+D
For Windows users, the EOF trigger is Ctrl+Z+Return/Enter.
(when to use EOF trigger? After you write your expression in q2 program, the program pauses, i.e., valid/invalid statements are not executed instantly. Trigger EOF to execute statements after yylex())