

## **Assignment No 1**



**Name:**

**M.Zain Ul Abideen**

**Reg No:**

**FA22-BCS-090**

**Subject:**

**Compiler Construction**

**Submitted to:**

**Mr. Nasir Mehdi**

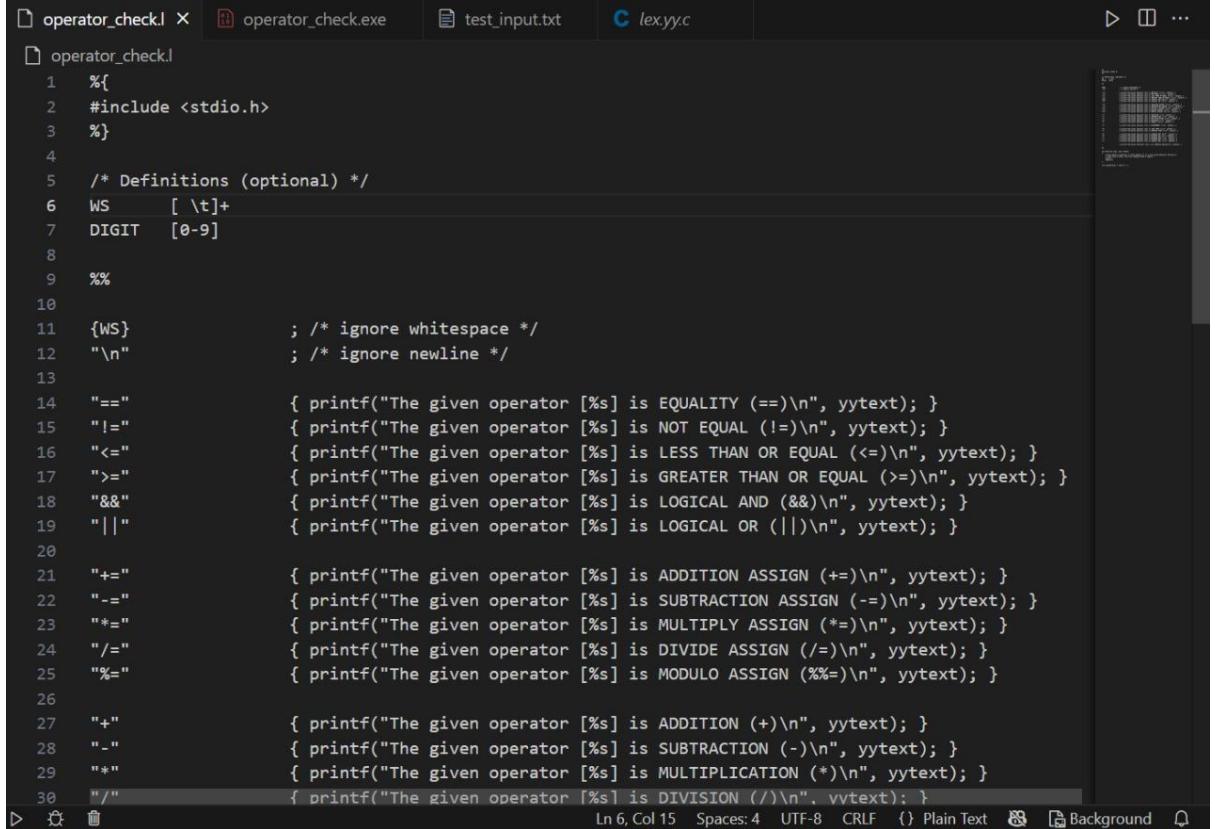
## Write a lex file to check whether a user enter a VALID operator or INVALID not.

This program is written in Lex (Flex) to check whether a user enters a valid operator (like +, -, \*, /, &&, ||, etc.) or an invalid operator.

If the entered symbol matches a defined operator, the program displays its meaning; otherwise, it prints that the operator is invalid.

### How the Program Works

- The user enters an operator, and the program checks it against a list of valid operators using Lex rules.
- If the input matches, it prints the operator's name; otherwise, it shows "INVALID operator," and then the program ends.



```
%{  
#include <stdio.h>  
%}  
  
/* Definitions (optional) */  
WS      [ \t]+  
DIGIT   [0-9]  
  
%%  
  
{\WS}          ; /* ignore whitespace */  
\n          ; /* ignore newline */  
  
==          { printf("The given operator [%s] is EQUALITY (==)\n", yytext); }  
!=          { printf("The given operator [%s] is NOT EQUAL (!=)\n", yytext); }  
<=          { printf("The given operator [%s] is LESS THAN OR EQUAL (<=)\n", yytext); }  
>=          { printf("The given operator [%s] is GREATER THAN OR EQUAL (>=)\n", yytext); }  
&&          { printf("The given operator [%s] is LOGICAL AND (&&)\n", yytext); }  
||          { printf("The given operator [%s] is LOGICAL OR (||)\n", yytext); }  
  
+=          { printf("The given operator [%s] is ADDITION ASSIGN (+=)\n", yytext); }  
-=          { printf("The given operator [%s] is SUBTRACTION ASSIGN (-=)\n", yytext); }  
*=          { printf("The given operator [%s] is MULTIPLY ASSIGN (*=)\n", yytext); }  
/=          { printf("The given operator [%s] is DIVIDE ASSIGN (/=)\n", yytext); }  
%="          { printf("The given operator [%s] is MODULO ASSIGN (%=)\n", yytext); }  
  
+          { printf("The given operator [%s] is ADDITION (+)\n", yytext); }  
-          { printf("The given operator [%s] is SUBTRACTION (-)\n", yytext); }  
*          { printf("The given operator [%s] is MULTIPLICATION (*)\n", yytext); }  
/          { printf("The given operator [%s] is DIVISION (/)\n", yytext); }
```

The screenshot shows a terminal window within a code editor interface. The terminal tab is selected at the top. The window title is "powershell". The terminal content displays the execution of a PowerShell script named "operator\_check.exe". The script prompts the user to enter an operator and then prints its meaning. It also demonstrates reading from a file named "test\_input.txt" and piping it to the script.

```
PS D:\Compiler Construction Lab\Theory Assignment 1> .\operator_check.exe
e
Enter an operator to check whether it is in the valid operators OR Not
PS D:\Compiler Construction Lab\Theory Assignment 1> echo "==" | .\operator_check.exe
Enter an operator to check whether it is in the valid operators OR Not
The given operator [==] is EQUALITY (==)
PS D:\Compiler Construction Lab\Theory Assignment 1> Get-Content test_input.txt | .\operator_check.exe
Enter an operator to check whether it is in the valid operators OR Not
The given operator [==] is EQUALITY (==)
The given operator [!=] is NOT EQUAL (!=)
The given operator [<=] is LESS THAN OR EQUAL (<=)
The given operator [>=] is GREATER THAN OR EQUAL (>=)
The given operator [&&] is LOGICAL AND (&&)
The given operator [||] is LOGICAL OR (||)
The given operator [+=] is ADDITION ASSIGN (+=)
The given operator [+] is ADDITION (+)
The given operator [-] is SUBTRACTION (-)
```