INSTITUTE OF ENGINEERING & MANAGEMENT

NAME OF THE STUDENT : Abhishek Anand

SEMESTER : 6th

ROLL NO. : 57

ASSIGNMENT OF DEPARTMENT : Compiler Design Lab

DATE OF EXPERIMENT / PROJECT : 12/04/2022

DATE OF SUBMISSION :

TITLE : C program to simulate lexical analyzer for validatir operator

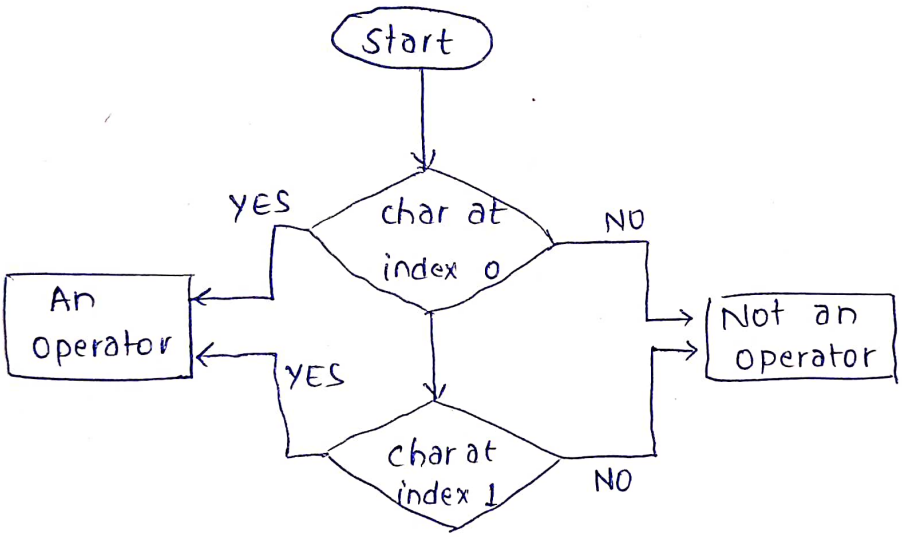
OBJECT : To write a C program to simulate lexical analyzer for validating operators.

Objective- To Write a c program to simulate lexical analyzer for validating operators

Resource: Online GDB, gcc/g++

Program logic & Procedure:

- First read the given input.
- If the given input matches with any of the operator symbol then it will display in terms of words of the particular symbol.
- Else it will print not an operator.



Program :-

```
# include <stdio.h>
# include <conio.h>
int main(){
        char str[5];
        printf("\n Enter any operator :");
        scanf("%s", str);
        switch(str[0]){
                case '>':
                        if(str[1] == '=')
```

Name: Abhishek Anand
Section: A

Roll No.: 57
Year: 3rd

```
    printf (" \n Greater than or equal to ");
    else
       printf (" \n Greater than");
       break;
case '<' :
       if (str[1] == '=')
      printf (" \Less than or equal to");
       else
       printf ("\n Less than");
       break;
case '=' :
       if ( str[1] == '=')
       printf ("\n Equal to");
      else
         printf (" \n Assignment");
       break;
case '!' :  if (str[1] == '=')
            printf ("\n Not equal");
            else
              printf ("\n Bit not");
            break;
   case '|' :
              if (str[1] == '|')
              printf ("\n Logical or");
              else
                 printf ("\n Bitwise OR");
               break;
    Case '+' :
              printf (" \n Addition");
              break;
      case '-';
              printf ("\n Subtraction");
              break;
```

```
case '*':
        printf ("\n Multiplication");
        break;
case '/' ;
        printf ("\n Division");
        break ;
case '%' :
        printf ("\n Modulus");
      break;
default :
        printf ("\n Not an operator");
    }
  return 0 ;
}
```

Output :

Enter any operator : *
=> Multiplication

Enter any operator : ==
=> Equal to

Enter any operator : !=
=> Not Equal to

Discussion :

For the above mentioned problem statement, I Used the switch case method to solve the problem Where I firstly taken an input string then I given the switch case, so that if there will be any matching operator found at 0th & 1st index then it will return the operator in terms of words of that particular symbol or If there is no operator found. It will print Not an operator.