# PRACTICAL-1

**Objective:** Write a Lex or program to identify the tokens.

**Code:**

```
%{
        #include<stdio.h>
%}
%%
int|float|char|if|else|for|while {printf("%s --->Keyword\n",yytext);}
[a-zA-Z]([a-zA-Z0-9])*  { printf("%s is  Identifier\n",yytext); }
([0-9]|[0-9]"."[0-9])+ {printf("%s is Number\n",yytext); }
"+"|"-"|"*"|"/"|"=" {printf("%s is Operator\n",yytext); }
[ \t\n]+ /* Eating up space */
%%
int yywrap()
{
return 1;
}
int main()
{
        printf("Enter some expressions :\n");
        yylex();
        return 0;
}
```

**Output:**

```
C:\Users\v4984\Desktop\c pr

Enter some expressions :
a+b=c
a is  Identifier
+ is Operator
b is  Identifier
= is Operator
c is  Identifier
```

# PRACTICAL-2

**Objective: Write a Lex or program to implement Simple Calculator.**

**Code:**

```
%option noyywrap
%{
        #include<stdio.h>
        int op= 0,i;
        float a,b;
%}
dig [0-9]+|([0-9]*)"."([0-9]+)
add "+"
sub "-"
mul "*"
div "/"
pow "^"
ln \n
%%
{dig} {digi();}
{add} {op=1;}
{sub} {op=2;}
{mul} {op=3;}
{div} {op=4;}
{pow} {op=5;}
{ln} {printf("\n The Answer :%f\n\n",a);}
%%
digi()
{
if(op==0)
a=atof(yytext);
else{
b=atof(yytext);
switch(op)
```

```
{
case 1:
        a=a+b;
        break;
case 2:
        a=a-b;
        break;
case 3:
        a=a*b;
        break;
case 4:
        a=a/b;
        break;
case 5:
        for(i=a; b>1;b--)
        a=a*i;
        break;
default:
break;
}
op=0;
}
}
main(int argv, char *argc[])
{
yylex();
}
```

**Output:**

```
C:\Users\v4984\Desktop\c pr
10+5
 The Answer :15.000000
10-5
 The Answer :5.000000
10/5
 The Answer :2.000000
10*2
 The Answer :20.000000
10^2
 The Answer :100.000000
```

# PRACTICAL-3

**Objective:** Write a lex program to check the given number is even or not.

**Code:**

```
%{
#include <stdio.h>
#include <stdlib.h>
%}
%%
[0-9]+  {
          int num = atoi(yytext);
          if (num % 2 == 0)
             printf("%s is Even\n", yytext);
          else
             printf("%s is Odd\n", yytext);
       }
%%
int main() {
   printf("Enter a number: ");
   yylex();
   return 0;
}
int yywrap() {
   return 1;
}
```

**Output:**

C:\Users\v4984\Desktop\c pr

```
Enter a number: 11
11 is Odd
10
10 is Even
```

# PRACTICAL-4

**Objective:** Write a lex program to check whether the given operator is relational operator or not.

**Code:**

```
%{
#include <stdio.h>
%}
%%
"==" { printf("%s is a relational operator\n", yytext); }
"!=" { printf("%s is a relational operator\n", yytext); }
">=" { printf("%s is a relational operator\n", yytext); }
"<=" { printf("%s is a relational operator\n", yytext); }
">" { printf("%s is a relational operator\n", yytext); }
"<" { printf("%s is a relational operator\n", yytext); }
.  { printf("%s is NOT a relational operator\n", yytext); }
\n  { /* Ignore new lines */ }
%%

int main() {
   printf("Enter an operator: ");
   yylex();
   return 0;
}
int yywrap() {
   return 1;
```

**Output:**

C:\Users\v4984\Desktop\c pr

```
Enter an operator: >=
>= is a relational operator
==
== is a relational operator
<=
<= is a relational operator
=
= is NOT a relational operator
```

# PRACTICAL-5

**Objective:** Write a program to count number of word.

**Code:**

```
%{
#include <stdio.h>
int word_count = 0;
%}
%%
[a-zA-Z0-9]*  { word_count++;}
"\n" {printf("%d",word_count); word_count=0;}
%%
int yywrap()
{
return 1;
}
int main() {
    printf("Enter text (Ctrl+D to end input):\n");
    yylex();
   return 0;
}
```

**Output:**