Lex program to count the number of vowels and consonants:	**Lex program to identify the tokens:**	**Lex program to count the no. of words,lines and characters:**

%{
int vow_count=0;
int const_count =0;
%}
%%
[aeiouAEIOU] {vow_count++;}
[a-zA-Z] {const_count++;}
%%
int yywrap(){}
int main()
{
printf("Enter the string of vowels and consonents:");
yylex();
printf("Number of vowels are: %d\n", vow_count);
printf("Number of consonants are: %d\n", const_count);
return 0;
}
	digit [0-9]
	letter [A-Za-z]
	%{
	int count_id,count_key;
	%}
	%%
	(stdio.h|conio.h) { printf("%s is a standard library\n",yytext); }
	(include|void|main|printf|int) { printf("%s is a keyword\n",yytext); count_key++; }
	{letter}({letter}|{digit})* { printf("%s is a identifier\n", yytext); count_id++; }
	{digit}+ { printf("%s is a number\n", yytext); }
	\"(\\.|[^"\\])*\" { printf("%s is a string literal\n", yytext); }
	.|\n { }
	%%
	int yywrap(void) {
	return 1;
	}
	int main(int argc, char *argv[]) {
	yyin = fopen(argv[1], "r");
	yylex();
	printf("number of identifiers = %d\n", count_id);
	printf("number of keywords = %d\n", count_key);
	fclose(yyin);
	}
		%{
		int nchar, nword, nline;
		%}
		%%
		\n { nline++; nchar++; }
		[^ \t\n]+ { nword++, nchar += yyleng; }
		. { nchar++; }
		%%
		int yywrap(void) {
		return 1;
		}
		int main(int argc, char *argv[]) {
		yyin = fopen(argv[1], "r");
		yylex();
		printf("Number of characters = %d\n", nchar);
		printf("Number of words = %d\n", nword);
		printf("Number of lines = %d\n", nline);
		fclose(yyin);
		}

Lex program to count the no. of macros:	**Lex program to print all the HTML tags:**	**Lex program to add line number before each line:**

%{
int nmacro, nheader;
%}
%%
^#define { nmacro++; }
^#include { nheader++; }
.|\n { }
%%
int yywrap(void) {
return 1;
}
int main(int argc, char *argv[]) {
yyin = fopen(argv[1], "r");
yylex();
printf("Number of macros defined = %d\n", nmacro);
printf("Number of header files included = %d\n",
nheader);
fclose(yyin);
}
	%{
	int tags;
	%}
	%%
	"<"[^>]*> { tags++; printf("%s \n", yytext); }
	.|\n { }
	%%
	int yywrap(void) {
	return 1; }
	int main(void)
	{
	FILE *f;
	char file[10];
	printf("Enter File Name : ");
	scanf("%s",file);
	f = fopen(file,"r");
	yyin = f;
	yylex();
	printf("\n Number of html tags: %d",tags);
	fclose(yyin);
	}
		%{
		int yylineno;
		%}
		%%
		^(.*)\n printf("%4d\t%s", ++yylineno, yytext);
		%%
		int yywrap(void) {
		return 1;
		}
		int main(int argc, char *argv[]) {
		yyin = fopen(argv[1], "r");
		yylex();
		fclose(yyin);
		}

Lex program to count the no. of comments:

```
%{
int com=0;
%}
%s COMMENT
%%
"/*" {BEGIN COMMENT;}
<COMMENT>"*/" {BEGIN 0; com++;}
<COMMENT>\n {com++;}
<COMMENT>. {;}
\/\/.* {; com++;}
.|\n {fprintf(yyout,"%s",yytext);}
%%
void main(int argc, char *argv[])
{
if(argc!=3)
{
printf("usage : ./a.exe input.c output.c\n");
exit(0);
}
yyin=fopen(argv[1],"r");
yyout=fopen(argv[2],"w");
yylex();
printf("\n number of comments are = %d\n",com);
}
int yywrap()
{
return 1;
}
```

Lex program to construct simple calculator:

```
%{
#undef yywrap
#define yywrap() 1
int f1=0,f2=0;
char oper;
float op1=0,op2=0,ans=0;
void eval();
%}
DIGIT [0-9]
NUM {DIGIT}+(\.{DIGIT}+)?
OP [*/+-]
%%
{NUM} {
if(f1==0)
        {
                op1=atof(yytext);
                f1=1;
        }

        else if(f2==-1)
        {
                op2=atof(yytext);
                f2=1;
        }

        if((f1==1) && (f2==1))
        {
                eval();
                f1=0;
                f2=0;
        }
}
{OP} {
        oper=(char) *yytext;
        f2=-1;
}
[\n] {
        if(f1==1 && f2==1)
        {
                eval;
                f1=0;
                f2=0;
        }
}
%%
int main()
{       yylex();
}
void eval()
{
        switch(oper)
        {
                case '+':
                        ans=op1+op2;
                        break;
                case '-':
                        ans=op1-op2;
                        break;

                case '*':
                        ans=op1*op2;
                        break;
                case '/':
                        if(op2==0)
                        {       printf("ERROR");
                                return;
                        }
                        else
                        {       ans=op1/op2;
                        }
                        break;
                default:
                        printf("operation not available");
                        break;
        }printf("The answer is = %lf",ans);}
```