LEX PROGRAM FOR CAPITAL WORDS

```
%{
        #include<stdio.h>
%}

%%
[A-Z]+[\t\n ] { printf("%s",yytext); }
. ;
%%
Int yywrap(){}
int main( )
{
        printf("Enter the input string:\n");
        yylex();
}
```

LEX PROGRAM FOR EMAIL VALID OR NOT

```
%{
%}

%%

[a-z.0-9_]+@[a-z]+".com"|".in"  { printf("it is valid");}
.+  { printf("it is not valid");}
%%
int yywrap(){}

int main()
{
printf("enter the mail:");
yylex();
}
```

LEX PROGRAM FOR MOBILE NUMBER VALID OR NOT

```
%{
%}

%%

[6-9][0-9]{9} {printf("\n mobile number valid\n");}
.+ {printf("\n mobile number invalid\n");}

%%
int yywrap(void){}

int main()
{
printf("\n enter the mobile number:");
yylex();
printf("\n");
return 0;
}
```

LEX PROGRAM FOR COUNT COMMENT LINES

```
%{
#include<stdio.h>
int nc=0;
%}

%%
"/*"[a-zA-Z0-9\n\t ]*"*/"  {nc++;}
"//"[a-zA-Z0-9\t ]*"\n"   {nc++;}
%%
int yywrap( ){}

int main(int argc ,char* argv[])
{
        yyin=fopen(argv[1],"r");
```

LEX PROGRAM FOR COUNT OF POSITIVE NUMBER AND NEGATIVE NUMBER

```
%{
int positive_no = 0, negative_no = 0;
%}

%%
^[-][0-9]+ {negative_no++;
                printf("negative number = %s\n",yytext);}

[0-9]+ {positive_no++;
                printf("positive number = %s\n",yytext);}

%%

int yywrap(){}
```

LEX PROGRAM FOR HTML

```
%{
#include<stdio.h>
%}

%%
\<[^>]*\> fprintf(yyout,"%s\n",yytext);
.|\n;

%%

int yywrap()
{
return 1;
}
```

```
        yyout=fopen("output.c","w");
        yylex( );
        printf("The number of comment lines=%d\n",nc);

}
```

```
int main()

{

yylex();
printf ("number of positive numbers = %d,"
                "number of negative numbers = %d\n",
                        positive_no, negative_no);

return 0;
}
```

```
int main()
{
yyin=fopen("sample.html","r");
yyout=fopen("output.txt","w");
yylex();
return 0;
}
```

LEX PROGRAM FOR IDENTIFIER OR NOT

```
%{
#include<stdio.h>
%}

%%

[a-zA-Z][a-zA-Z0-9]+ { printf("\n%s is IDENTIFIER", yytext);}
.+ { printf("\n%s is NOT AN IDENTIFIER",yytext);}

%%
int yywrap(){}
int main()
{
        while( yylex());
}
```

LEX PROGRAM FOR COUNT VOWELS AND CONSONENTS

```
%{
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 consonents are:%d\n",const_count);
return 0;
}
```

LEX PROGRAM FOR ADD LINE NUMBER

```
%{
#include<stdio.h>
int ln=0;
%}

%%

.* {ln++; fprintf(yyout,"\n%d:%s",ln,yytext);}
%%
int yywrap(){}

int main()
{
yyin=fopen("simple.txt","r");
yyout=fopen("out.txt","w");
yylex();
return 0;
}
```

LEX PROGRAM FOR COMMENT OR NOT

```
%{
#include<stdio.h>
%}
%%

[/]{2}.* { printf("\n%s is COMMENT", yytext);}
.+ { printf("\n %s is NOT A COMMENT",yytext);}
%%
```

LEX PROGRAM FOR DIGIT OR NOT

```
%{
#include<stdio.h>
%}
%%
[0-9]+|[0-9]*\.[0-9]+ { printf("\n%s is DIGIT", yytext);}
.+ { printf("\n%s is NOT A DIGIT",yytext);}
```

LEX PROGRAM FOR MACROS AND HEADER FILES

```
%{
int nmacro, nheader;
%}
%%
^#define { nmacro++; }
^#include { nheader++; }
%%
int yywrap(void) {
```

```
int yywrap(){}

int main()
{
        while( yylex());
}
```

```
%%
int yywrap(){}
int main()
{
while( yylex());
}
```

```
return 1;
}
int main() {
yylex();
printf("Number of macros defined = %d\n", nmacro);
printf("Number of header files included = %d\n", nheader);
}
```

LEX PROGRAM FOR KEYWORDS AND IDENTIFIERS

```
%{
#include<stdio.h>
%}

%%

if|else|while|int|switch|for|char { printf("its a keyword");}
[a-zA-Z0-9]+ { printf("\n%s is IDENTIFIER", yytext);}

%%
int yywrap( ){}
int main()
{
        while( yylex());
}
```

LEX PROGRAM FOR BASIC MATHEMATICAL OPERATIONS

```
%{
#include<stdio.h>
%}
%%

"="|"+"|"-"|"/"|"*" { printf("valid");}
.+  {printf("invalid");}
%%
int yywrap(){}
int main()
{
printf("enter the input:");
yylex();
return 0;
}
```

LEX PROGRAM FOR DOB VALID OR NOT

```
%{
#include<stdio.h>
%}

%%

[0-9][0-9]\/[0-1][0-9]\/[1-2][0-9]{3} { printf("valid");}
.+ { printf("invalid");}
%%

int yywrap(){}

int main()
{
yylex();
}
```

LEX PROGRAM FOR URL VALID OR NOT

LEX PROGRAM FOR COUNT NO OF TOKENS

```
%{
int n = 0 ;
%}
%%
"while"|"if"|"else"|"int"|"float"  {n++;printf("\t keywords : %s", yytext);}
[a-zA-Z_][a-zA-Z0-9_]* {n++;printf("\t identifier : %s", yytext);}
"<="|"=="|"="|"++"|"-"|"*"|"+" {n++;printf("\t operator : %s", yytext);}
[(){}|, ;]    {n++;printf("\t separator : %s", yytext);}
[0-9]*"."[0-9]+ {n++;printf("\t float : %s", yytext);}
[0-9]+ {n++;printf("\t integer : %s", yytext);}
```

LEX PROGRAM FOR SUBSTRING CONVERT abc to ABC

```
%{
%}
%%

[a-z] {printf("%c",yytext[0]-32);}
. {}

%%
int yywrap(void){}

int main()
```

	. ;
	%%
	int yywrap(){
	return 1;
	}
	int main()
	{
	  yylex();
	  printf("\n total no. of token = %d\n", n);
	
	}
		{
		printf("\nenter the string : ");
		yylex();
		}

LEX PROGRAM FOR NO.OF CHARS,LINES,WORDS	LEX PROGRAM FOR ALL CONSTANTS	LEX PROGRAM TO COUNT WORDS

%{
int nlines,nwords,nchars;
%}
 	
%%
\n {
  nchars++;nlines++;
  }
 	
[^ \n\t]+ {nwords++, nchars=nchars+yyleng;}
. {nchars++;}
 	
%%
 	
int yywrap(void) {}
int main()
{
 yylex();
 printf("Lines =
 %d\nChars=%d\nWords=%d",nlines,nchars,nwords);
 	
 return 0;
}
	%{
	%}
	
	%%
	<INITIAL>[0-9]+ {printf("Integer\n");}
	<INITIAL>[0-9]+[.][0-9]+ {printf("Float\n");}
	<INITIAL>[A-Za-z0-9_]* {printf("Identifier\n");}
	<INITIAL>[^\n] {printf("Invalid\n");}
	
	%%
	
	int yywrap(){}
	
	int main()
	{
	printf("Enter String\n");
	yylex();
	return 0;
	}
		%{
		#include<stdio.h>
		#include<string.h>
		int i = 0;
		%}
		
		%%
		([a-zA-Z0-9])* {i++;}
		
		"\n" {printf("%d\n", i); i = 0;}
		%%
		
		int yywrap(){}
		
		int main()
		{
		  printf("Enter the Sentence :");
		  yylex();
		
		  return 0;
		}

LEX PROGRAM TO COUNT THE FREQUENCY OF THE CODE	Lex code to find the length of the longest word	**Lex code to replace a word with another word in a file**

```
%{
   int icount=0,factcount=0;
%}

%%

fact    factcount++;
i       icount++;
(.|\n)    ;

%%
main()
{
   yylex();
   printf("Count of \"fact\"= %d \nCount of letter 'i' =
 %d\n",factcount,icount);
   return 0;
}
```

```
%{
 int counter = 0;
 %}
%%
 [a - zA - Z] +
 {
  if (yyleng > counter)
{
   counter = yyleng;
  }
}
%%
int main() {
 yylex();
 printf("largest%d",counter);
 printf("\n");
}
```

```
%{
#include <stdio.h>
%}

%%
"greeks"   { printf("ReplacementWord "); }
.|\n            { putchar(yytext[0]); }
%%
int yywrap(){
return 1;
}
int main() {
   yylex();
   return 0;
}
```