**SAMPLE LEX PROGRAMS**

***Compilation Commands***

lex <filename.l> /\*lex.yy.c will get generated\*/

cc lex.yy.c

./a.out

**(or)**

lex <filename.l> /\* lex.yy.c will get generated\*/

cc lex.yy.c -o filename /\* saves the compiled file in the given filename \*/

**.**/filename

**1.Program to remove white spaces.**

%option main

non\_white [^ \t\n]\*

%%

{non\_white} ECHO;

. ;

[\n] ;

%%

**2. Program to check identifier or number**

%option main

%%

[a-z]([a-z]|[0-9])\* {printf("identifier");}

[0-9]\* {printf("number");}

.\* { printf("invalid");}

%%

**3. Program to remove comments from the code**

option noyywrap

%{

/\* idea here is to strip comments out of C code. \*/

/\* INITIAL is the default start state. Comment is our new \*/

/\* state where we remove comments. \*/

%}

%s COMMENT

%%

<INITIAL>"//".\* ;

<INITIAL>"/\*" BEGIN COMMENT;

<INITIAL>. ECHO;

<INITIAL>[\n] ECHO;

<COMMENT>"\*/" BEGIN INITIAL;

<COMMENT>. ;

<COMMENT>[\n] ;

%%

main() {

yylex();

}

**4. Program to count the number of characters, words and lines**

%option noyywrap

%{

int charCount = 0, wordCount = 0, lineCount = 0;

%}

word [^ \t\n]+

%%

{word} {wordCount++; charCount += yyleng; }

[\n] {charCount++; lineCount++; }

. {charCount++; }

%%

main() {

yylex();

printf("Characters: %d Words: %d Lines %d\n",charCount, wordCount, lineCount);

}

**5. Program to identify Tokens.**

%option main

/\* %{

int ID;

int LP;

int RP;

%} \*/

non-Identifier [^[a-z]]\*

%%

[a-z]\* {printf("\n ID ");}

[0-9]\* {printf("\n Number ");}

\( {printf("\n LP ");}

\) {printf("\n RP ");}

\= {printf("\n Assignment operator ");}

\+ {printf("\n Addition operator ");}

[ \t] {printf("\n delimiter ");}

[\n] {printf("\n newline delimiter ");}

non-Identifier ECHO;

%%