|  |  |  |
| --- | --- | --- |
| 1. LEX PROGRAM FOR NO.OF  CHARS,LINES,WORDS | 2. LEX PROGRAM FOR ALL CONSTANTS | 3. LEX PROGRAM FOR MACROS  AND HEADER FILES |
| %{  int i =0,l=0,c=0;  %}  %%  [\n] {l++;}  [ ] {i++;}  [a-zA-Z0-9] {c++;}  %%  int yywrap(){} int main()  {  printf("enter the string: "); yylex();  printf("no of lines:%d\n",l); printf("no of words is:%d",i+l); printf("no of characters:%d",c);  } | %{  int cons = 0;  %}  digit [0-9]  %%  {digit}+"."{digit}+ { cons++; printf("%s is a floating-point constant\n", yytext); }  {digit}+ { cons++; printf("%s is an integer constant\n", yytext); }  .|\n { }  %%  int yywrap() {  }  int main() {  printf("Enter the code:"); yylex();  printf("Number of Constants: %d\n", cons); return 0;  } | %{  int nmacro, nheader;  %}  %%  "#define" {nmacro++;} "#include" {nheader++;}  .|\n { }  %%  int yywrap()  {  return 1;  }  int main()  {  printf("enter the string:\n"); yylex();  printf("Number of macros defined = %d \n Number of header files included = %d\n",nmacro,nheader);  } |
| 4. LEX PROGRAM FOR HTML | 5.LEX PROGRAM FOR ADD LINE NUMBER | 6.LEX PROGRAM FOR COUNT  COMMENT LINES |
| %{  #include <stdio.h>  %}  %%  \<[^>]\*\> fprintf(yyout,"%s\n",yytext);  .|\n;  %%  int yywrap()  {  return 1;  }  int main()  {  yyin=fopen("sample1.html","r"); yyout=fopen("output.txt","w"); yylex();  return 0;  } | %{  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();  } | %{  #include<stdio.h> int n=0;  %}  %%  "/"[a-zA-Z0-9 \n\t]+"/" {n++;}  "//"[a-zA-Z0-9 \n\t]+"//" {n++;}  %%  int yywrap()  {}  int main()  {  printf("enter:"); yylex();  printf("no of comment lines:%d",n);  } |

|  |  |  |
| --- | --- | --- |
| 7. LEX PROGRAM CAPITAL WORDS FROM THE GIVEN INPUT | 8.LEX PROGRAM FOR EMAIL VALID OR NOT | 9.LEX PROGRAM FOR CAPITAL  WORDS or SUBSTRING abc to  ABC |
| %{  %}  %%  [A-Z]+ {printf("%s\n", yytext);}  .|\n {}  %%  int yywrap(){} int main()  {  printf("Enter a letter"); yylex();  } | %{  %}  %%  [a-z.0-9]+@[a-z]+(.com|.in) {printf("\n valid\n");}  .+ {printf("\n Invalid\n");}  %%  int yywrap()  {}  int main()  {  printf("\nEnter : "); yylex();  } | %{  #include <ctype.h>  %}  %%  [a-z] { printf("%c", toupper(yytext[0])); }  .|\n { printf("%s", yytext); }  %%  int yywrap()  {  }  int main() { yylex(); return 0;  } |
| 10.LEX PROGRAM FOR MOBILE NUMBER VALID OR NOT | 11.LEX PROGRAM separate the tokens in the given C program and display with appropriate caption. | 12. & 13. LEX PROGRAM FOR COUNT VOWELS AND CONSONANTS |
| %{  %}  %%  [0-9][0-9]{9} {printf("\n mobile number valid\n");}  .+ {printf("\n mobile number invalid\n");}  %%  int yywap()  {}  int main()  {  printf("\n enter the mobile number:"); yylex();  } | %{  #include<stdio.h>  %}  %%  bool|int|float|main|printf|int|char|float|double|void|if|while|for|do|main|return|else|elseif  {printf("\n%s is a Keyword",yytext);}  [-,+]?[0-9]+ {printf("\n%s is a numbers",yytext);} [,.;]+ {printf("\n%s is a Punctuation Chars",yytext);}  [a-zA-Z\_][a-zA-Z0-9\_]\* { printf("Identifier: %s\n", yytext); }  ["a-zA-Z"]+ {printf("\n%s is a string",yytext);}  [!%^&-+\*()]+ {printf("\n%s is a mathematical operator",yytext);}  %%  int yywrap()  {  }  int main()  {  yylex();  } | %{  int vcount=0; int ccount=0;  %}  %%  [aeiouAEIOU] {vcount++;} [a-z,A-Z] {ccount++;}  %%  int yywrap(){} int main()  {  printf("enter the string with vowels and consonants:"); yylex();  printf("\n no of vowels ::%d \n",vcount); printf("\n no of consonants ::%d \n",ccount);  } |
| 14. LEX PROGRAM FOR KEYWORDS AND IDENTIFIERS | 15. LEX PROGRAM FOR Number AND IDENTIFIERS LIST | 16.LEX PROGRAM FOR COUNT OF POSITIVE NUMBER AND NEGATIVE NUMBER |

|  |  |  |
| --- | --- | --- |
| %{  #include<stdio.h>  %}  %%  if|else|while|int|switch|for|char { printf("\n%s is a KEYWORD", yytext);}  [a-zA-Z0-9]+ { printf("\n%s is IDENTIFIER", yytext);}  %%  int yywrap( ){} int main()  {  yylex();  } | %{  #include <stdio.h> #include <stdlib.h> #include <string.h>  char alphabetList[1000] = ""; char numberList[1000] = "";  %}  %%  [0-9]+ { strcat(numberList, yytext); }  [a-zA-Z]+ { strcat(alphabetList, yytext); }  . { printf("Invalid input: %s\n", yytext); }  %%  int yywrap() {  printf("Alphabets: %s\n", alphabetList); printf("Numbers: %s\n", numberList); return 1;  }  int main() {  char input[100]; printf("Enter the input: ");  fgets(input, sizeof(input), stdin); yy\_scan\_string(input);  yylex(); return 0;  } | %{  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(){} int main()  {  yylex();  printf("number of posive integers=%d," "number of negativenumbers=%d\n",  positive\_no,negative\_no);  return 0;  } |
| 17.LEX PROGRAM FOR URL VALID OR NOT | 18.LEX PROGRAM FOR DOB VALID OR NOT | 19.LEX PROGRAM FOR DIGIT OR  NOT |
| %{  %}  %%  [http://]+[[www.]](http://www/)+[a-z]+".com" {printf("\n valid url\n");}  .+ {printf("\n invalid url\n");}  %%  int yywrap()  {}  int main()  {  printf("\n enter the url:"); yylex();  } | %{  %}  %%  [0-9][0-9]\/[0-1][0-9]\/[1-2][0-9]{3} { printf("valid");}  .+ { printf("invalid");}  %%  int yywrap(){} int main()  {  yylex();  } | %{  #include<stdio.h>  %}  %%  [0-9]+|[0-9]\*\.[0-9]+ { printf("\n%s is DIGIT", yytext);}  .+ { printf("\n%s is NOT A DIGIT",yytext);}  %%  int yywrap(){} int main()  {  yylex();  } |

|  |  |  |
| --- | --- | --- |
| 20.LEX PROGRAM FOR BASIC MATHEMATICAL OPERATIONS | 21. lex code to find the length of the longest word | 22.LEX code to count the frequency of the given word in a file |
| %{  #include<stdio.h> float op1=6,op2=7;  %}  %%  "+" {printf("sum =%lf",op1+op2);}  "-" {printf("diff=%lf",op1-op2);}  "\*" {printf("mul=%lf",op1\*op2);}  "/" {printf("div=%lf",op1/op2);}  . {printf("enter proper operator.");}  %%  int yywrap(){} int main()  {  printf("enter number 1");  printf("enter number 2"); printf("Enter the Operator::"); yylex();  } | /\*lex code to find the length of the longest word\*/  % {  int counter = 0; %  }  %  % [a - zA - Z] + {  if (yyleng > counter) {  counter = yyleng;  }  } %  %  main() {  yylex();  printf("largest: %d", counter); printf("\n");  } | %{  #include<stdio.h> #include<string.h> char word [] = "geeks"; int count = 0;  %}  %%  [a-zA-Z]+ { if(strcmp(yytext, word)==0) count++; }  . ;  %%  int yywrap()  {  return 1;  }  int main()  {  extern FILE \*yyin, \*yyout; yyin=fopen("input.txt", "r"); yylex();  printf("%d", count);  } |
| 23.LEX code to replace a word with another taking input from file | 25. LEX program to recognize a word and relational operator. | 26.Write a [LEX program to accept](https://copyprogramming.com/howto/what-s-the-difference-between-flex-start-and-baseline) [string starting with vowel](https://copyprogramming.com/howto/what-s-the-difference-between-flex-start-and-baseline). |

|  |  |  |
| --- | --- | --- |
| %{  #include<stdio.h> #include<string.h>  char replace\_with [] = "Best"; char replace [] ="A";  %}  %%  [a-zA-Z]+ { if(strcmp(yytext, replace)==0)  fprintf(yyout, "%s", replace\_with); else  fprintf(yyout, "%s", yytext);}  . fprintf(yyout, "%s", yytext);  %%  int yywrap()  {  return 1;  }  int main()  {  extern FILE \*yyin, \*yyout; yyin=fopen("input.txt", "r"); yyout=fopen("output.txt", "w");  yylex();  } |  | % {  int flag = 0;  % }  %%  [aeiouAEIOU].[a-zA-Z0-9.]+ flag=1; [a-zA-Z0-9]+  %%  main()  {  yylex();  if (flag == 1)  printf("Accepted");  else  printf("Not Accepted");  } |