**9.longest word**

%{

#include <stdio.h>

#include <string.h>

int max\_length = 0;

int current\_length = 0;

void update\_max\_length() {

if (current\_length > max\_length) {

max\_length = current\_length;

}

current\_length = 0;

}

%}

%%

[a-zA-Z]+ {

current\_length = yyleng;

update\_max\_length();

}

.|\n {

update\_max\_length();

}

%%

int main() {

printf("Enter the text (end with CTRL+D on Unix or CTRL+Z on Windows):\n");

yylex();

update\_max\_length();

printf("The length of the longest word is: %d\n", max\_length);

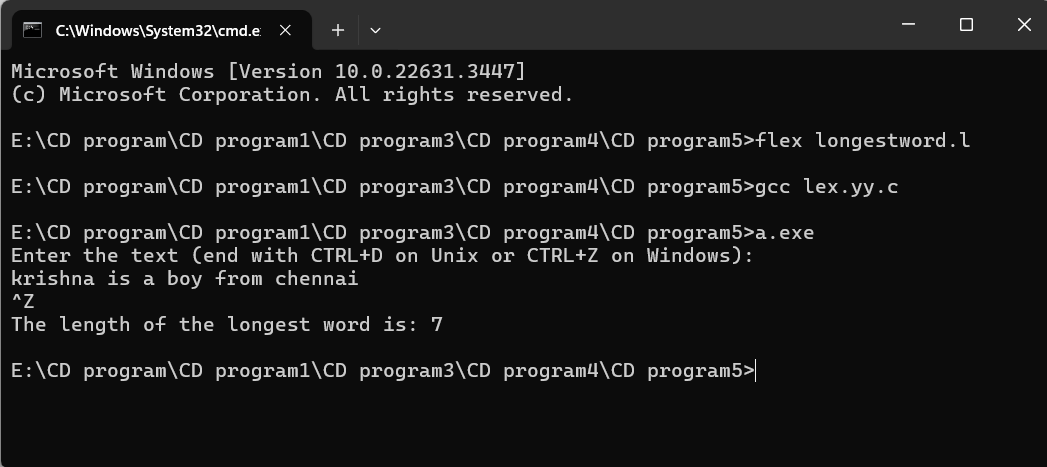
return 0;

}

int yywrap() {

return 1;

}



**10.url valid**

%%

((http)|(ftp))s?:\/\/[a-zA-Z0-9](.[a-z])+(.[a-zA-Z0-9+=?]\*)\* {printf("\nURL Valid\n");}

.+ {printf("\nURL Invalid\n");}

%%

void main()

{

printf("\nEnter URL : ");

yylex();

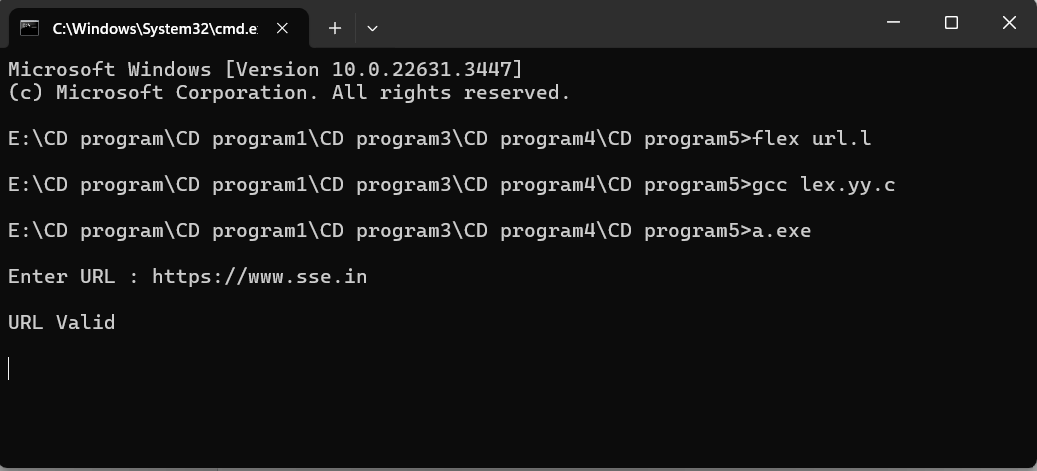
printf("\n");

}

int yywrap()

{

}



**11.dob valid**

%%

((0[1-9])|([1-2][0-9])|(3[0-1]))\/((0[1-9])|(1[0-2]))\/(19[0-9]{2}|2[0-9]{3}) printf("Valid DoB");

.\* printf("Invalid DoB");

%%

int main()

{

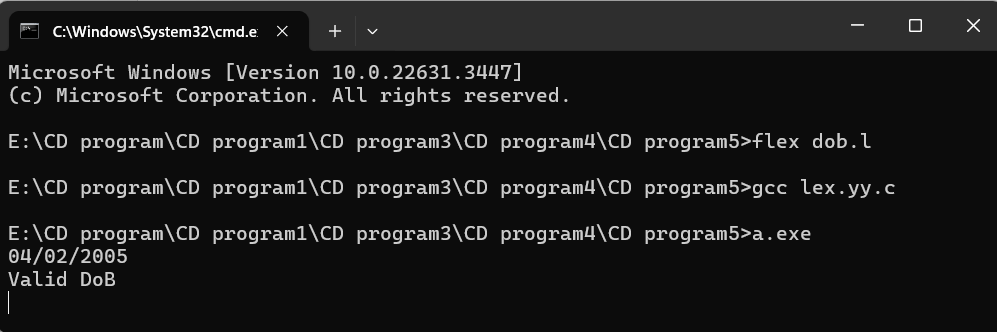
yylex();

return 0;

}

int yywrap()

{}



**12.word and relational operator**

%{

#include <stdio.h>

%}

%%

[a-zA-Z]+ {

printf("Word: %s\n", yytext);

}

"==" {

printf("Relational operator: %s\n", yytext);

}

"!=" {

printf("Relational operator: %s\n", yytext);

}

"<=" {

printf("Relational operator: %s\n", yytext);

}

">=" {

printf("Relational operator: %s\n", yytext);

}

"<" {

printf("Relational operator: %s\n", yytext);

}

">" {

printf("Relational operator: %s\n", yytext);

}

[ \t\n]+ ; // Ignore whitespace

. {

printf("Unknown character: %s\n", yytext);

}

%%

int main() {

printf("Enter words and relational operators (end with CTRL+D on Unix or CTRL+Z on Windows):\n");

yylex();

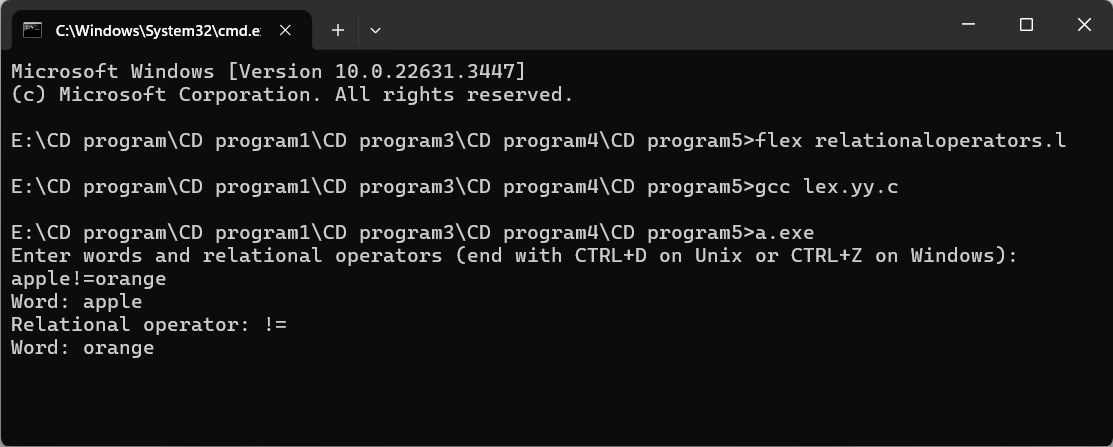
return 0;

}

int yywrap() {

return 1;

}



**13.replace word**

%{

#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;

}

/\* code section \*/

int main()

{

        extern FILE \*yyin, \*yyout;

        /\* open the input file

           in read mode \*/

    yyin=fopen("input.txt", "r");

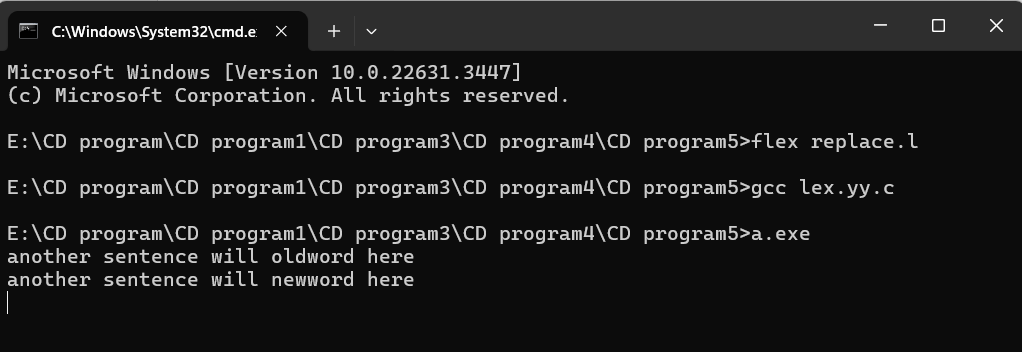
        /\* open the output file

           in write mode \*/

    yyout=fopen("output.txt", "w");

        yylex();

}

****

**14.mathematical operation**

%{

#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:

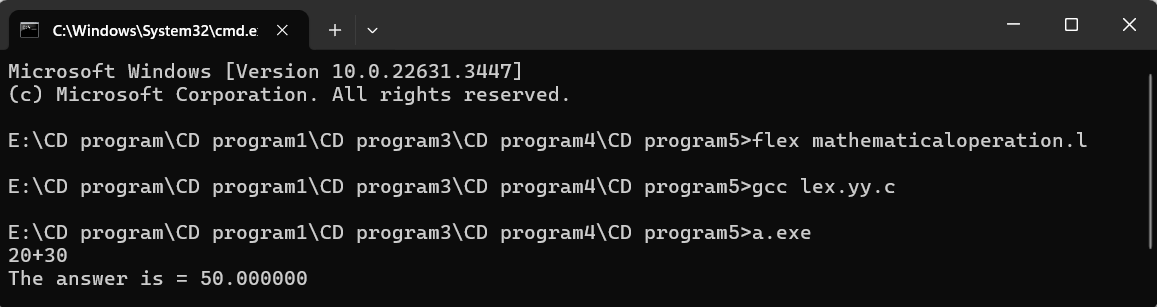
printf("operation not available");

break;

}

printf("The answer is = %lf",ans);

}

****

**15.email valid**

%{

int flag=0;

%}

%%

[a-z . 0-9]+@[a-z]+".com"|".in" { flag=1; }

%%

int main()

{

yylex();

if(flag==1)

printf("Accepted");

else

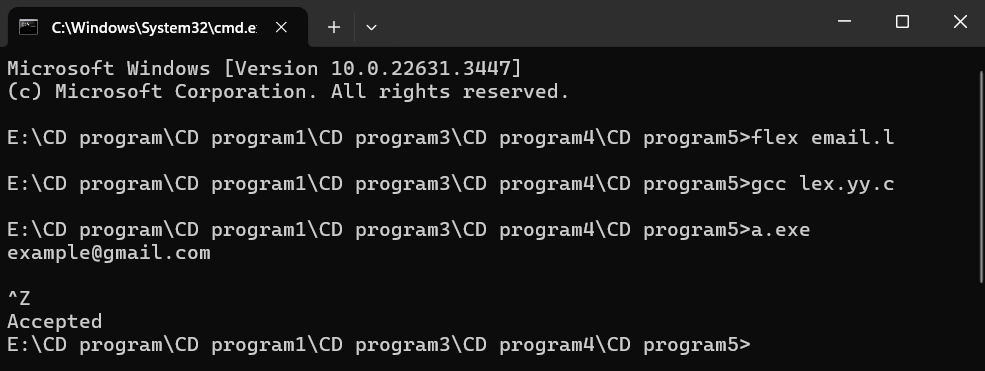
printf("Not Accepted");

}

int yywrap()

{ return 1;

}

****

**16.substring**

%{

#include <stdio.h>

%}

%%

[a-zA-Z ]\* {

int i;

for (i = 0; i <= yyleng - 3; i++) {

if (yytext[i] == 'a' && yytext[i+1] == 'b' && yytext[i+2] == 'c') {

yytext[i] = 'A';

yytext[i+1] = 'B';

yytext[i+2] = 'C';

}

}

printf("%s", yytext);

}

[\t]+ ; // ignore tabs

\n { printf("\n"); } // print newline

. { ECHO; } // echo unrecognized characters

%%

int main() {

yylex();

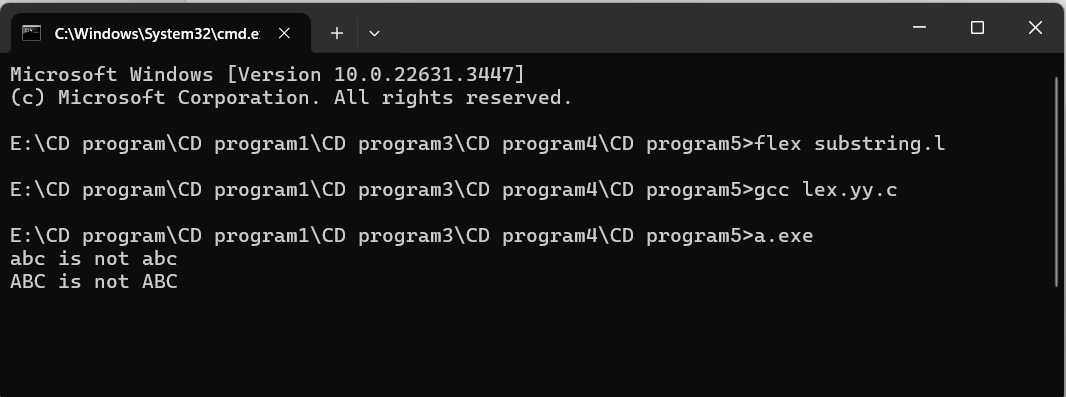
return 0;

}

int yywrap() {

return 1;

}

****