**SSN College of Engineering**

**Department of Computer Science and Engineering**

## UCS1712 – GRAPHICS AND MULTIMEDIA LAB

**EX NO: 7 –** Cohen Sutherland Line Clipping Algorithm

Name: Satheesh Kumar G R

Register Number:185001136

Semester: VII

Batch: 2018-2022

**AIM**

To write a program in C++ using openGL to perform line clipping using Cohen Sutharland Line Clipping Algorithm.

**ALGORITHM:**

1. Read the endpoints (x1,y1),(x2,y2) and window co-ordnidates xwmin,xwmax,ywmin,ywmax.
2. Calculate region code for endpoints(x,y) as follows:
3. Calculate slope m as (y2-y1)/(x2-x1).
4. Region code of (x1,y1) is c1 and region code of (x2,y2) is c2.
5. If c1==0 and c2==0 : Line Completely inside
6. If c1&c2 == 0 : Line partially inside and partially outside.
   1. Find the window boundary where line intersects and calculate new endpoints.
7. If c1&c2 > 0:Line completely outside.

**CODE:**

#include<bits/stdc++.h>

#include<iostream>

#include<GL/glut.h>

using namespace std;

float a1,b1,a2,b2;

float xwmin,ywmin,xwmax,ywmax;

void myInit (void) {

    glClearColor(1.0,1.0,1.0,0.0);

    glColor3f(0.0f,0.0f,0.0f);

    glPointSize(4.0);

    glMatrixMode(GL\_PROJECTION);

    glLoadIdentity();

    gluOrtho2D(0.0,200.0,0.0,200.0);

}

int region\_code(float x,float y) {

    int code=0;

    if(y>ywmax) code=8;

    if(y<ywmin) code=4;

    if(x>xwmax) code+=2;

    if(x<xwmin) code+=1;

    return code;

}

void cohen\_sutherland() {

    int c1=region\_code(a1,b1);

    int c2=region\_code(a2,b2);

    float m=(b2-b1)/(a2-a1);

    if((c1|c2)>0) {

        if((c1&c2)>0)

            exit(0);

        float ai=a1,bi=b1;

        int c=c1;

        if(c==0) {

            ai=a2;

            bi=b2;

            c=c2;

        }

        float a,b;

        if((c&8)>0) {

            b=ywmax;

            a=ai+(1.0/m)\*(ywmax-bi);

        }

        else if((c&4)>0) {

            b=ywmin;

            a=ai+(1.0/m)\*(ywmin-bi);

        }

        else if((c&2)>0) {

            a=xwmax;

            b=bi+m\*(xwmax-ai);

        }

        else if((c&1)>0) {

            a=xwmin;

            b=bi+m\*(xwmin-ai);

        }

        if(c==c1) {

            a1=a;

            b1=b;

            c1=region\_code(a1,b1);

        }

        if(c==c2) {

            a2=a;

            b2=b;

            c2=region\_code(a2,b2);

        }

    }

}

void disp() {

    glColor3f(0.0f,0.0f,1.0f);

    glClear(GL\_COLOR\_BUFFER\_BIT);

    glBegin(GL\_LINES);

    glVertex2d(a1,b1);

    glVertex2d(a2,b2);

    glEnd();

    glColor3f(0.0f,0.0f,0.0f);

    glBegin(GL\_LINE\_LOOP);

    glVertex2d(xwmin,ywmin);

    glVertex2d(xwmin,ywmax);

    glVertex2d(xwmax,ywmax);

    glVertex2d(xwmax,ywmin);

    glEnd();

    glFlush();

}

void clipline(unsigned char key,int x,int y) {

    if(key=='c') {

        cohen\_sutherland();

        glColor3f(0.0f,0.0f,1.0f);

        glClear(GL\_COLOR\_BUFFER\_BIT);

        glBegin(GL\_LINES);

        glVertex2d(a1,b1);

        glVertex2d(a2,b2);

        glEnd();

        glColor3f(0.0f,0.0f,0.0f);

        glBegin(GL\_LINE\_LOOP);

        glVertex2d(xwmin,ywmin);

        glVertex2d(xwmin,ywmax);

        glVertex2d(xwmax,ywmax);

        glVertex2d(xwmax,ywmin);

        glEnd();

        glFlush();

    }

}

int main(int argc,char\*\* argv) {

    cout<<"Lines"<<endl;

    cout<<"Line1 (a1,b1) : ";

    cin>>a1>>b1;

    cout<<"Line2 (a2,b2) : ";

    cin>>a2>>b2;

    cout<<"Window"<<endl;

    cout<<"XWmin : ";cin>>xwmin;

    cout<<"YWmin : ";cin>>ywmin;

    cout<<"XWmax : ";cin>>xwmax;

    cout<<"YWmax : ";cin>>ywmax;

    glutInit(&argc,argv);

    glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

    glutInitWindowSize(640, 480);

    glutInitWindowPosition(100, 150);

    glutCreateWindow("Cohen Sutherland Line clipping");

    glutDisplayFunc(disp);

    glutKeyboardFunc(clipline);

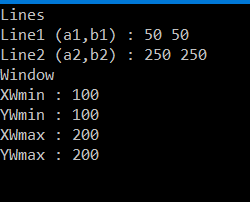
    myInit();

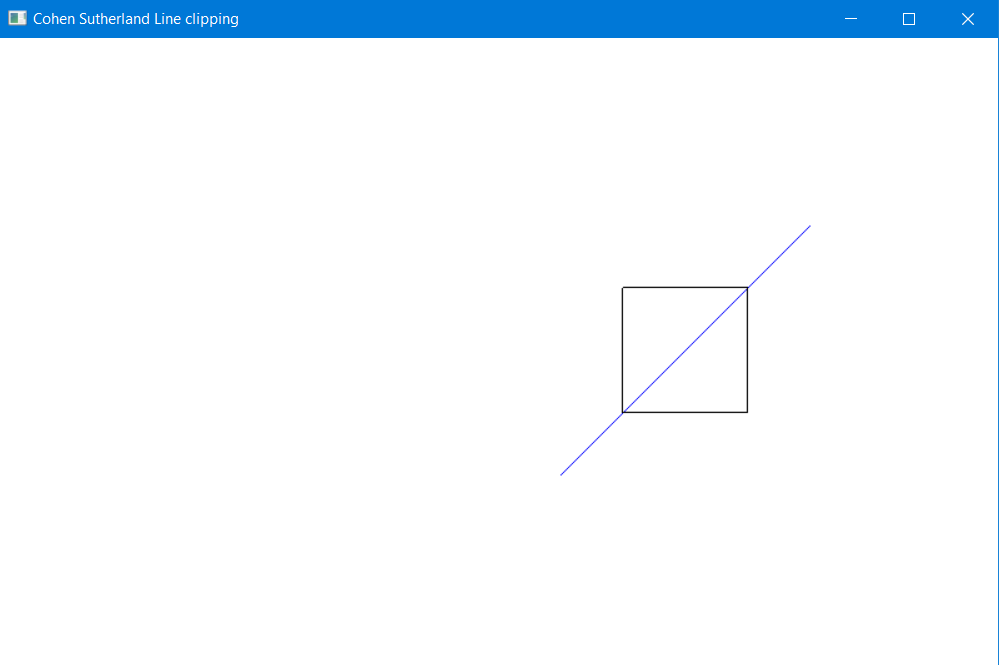
    glutMainLoop();

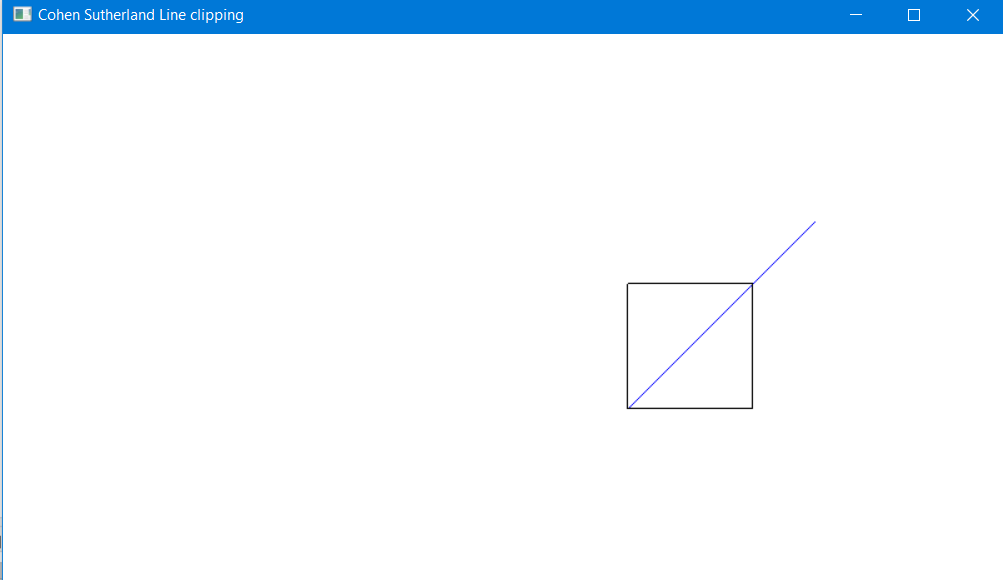
    return 0;

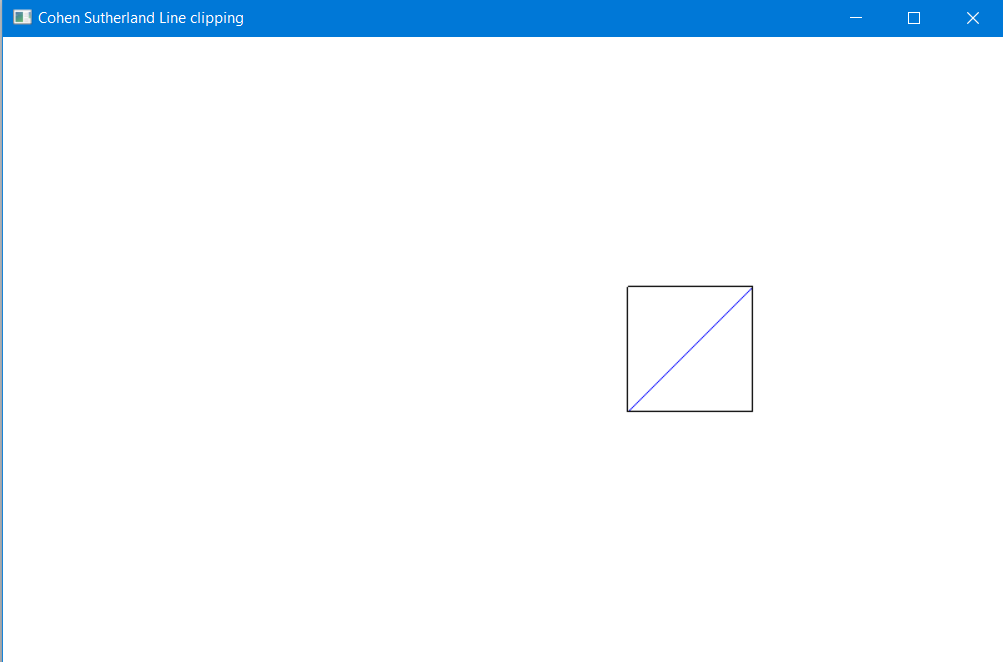
}

OUTPUT:









RESULT:

Thus line clipping is performed using Cohen Sutharland Line Clipping Algorithm.