Date _____

Expt. No. _____ Page No. _____

Program 6

Write a program to implement the Cohen-Sutherland line clipping algorithm. Make provision to specify the input for multiple lines, window for clipping and viewport for display the clipped image.

```
#include<stdio.h>
#include<stdlib.h>
#include<gl/glut.h>

#define outcode int
#define true 1
#define false 0

double xmin, ymin, xmax, ymax;
double xvmin, yvmin, xvmax, yvmax;

const int RIGHT = 4;
const int LEFT = 8;
const int TOP = 1;
const int BOTTOM = 2;

int n;
struct line_segment{
        int x1, y1, x2, y2;
};
struct line_segment ls[10];
```

Teacher's Signature _____

```
Outcode computeOutCode(double x, double y)
{
    outcode code=0;
    if (y>ymax)
        code |= TOP;
    else if (y<ymin)
        code |= BOTTOM;
    else if (x>xmax)
        code |= RIGHT;
    else if (x<xmin)
        code |= LEFT;
    return code;
}

void cohenSutherland(double x0, double y0, double x1, double y1)
{
    Outcode outcode0, outcode1, outcodeOut;
    bool accept=false, done=false;

    outcode0 = computeOutCode(x0, y0);
    outcode1 = computeOutCode(x1, y1);

    do
    {
        if (!(outcode0 | outcode1))
        {
            accept = true;
            done = true;
        }
```

Date _____

Expt. No. _____ Page No. _____

```
else if (outcode0 & outcode1)
        done = true;
else
    {
        double x,y;
        outcodeout = outcode0 ? outcode0 : outcode1;
        if (outcodeout & TOP)
          {
            x = x0+(x1-x0) *(ymax-y0)/(y1-y0);
            y = ymax;
          }
        else if (outcodeout & BOTTOM)
        {
            x = x0 + (x1-x0)* (ymin-y0)/(y1-y0)
            y= ymin;
        }
        else if (outcodeout & RIGHT)
        {
            y = y0 +(y1-y0) * (xmax-x0)/(x1-x0);
            x = xmax;
        }
        else
        {
            y = y0 + (y1-y0) * (xmin-x0)/(x1-x0);
            x = xmin;
        }
```

Teacher's Signature _____

```
if (outcodeout == outcode0)
    {
        x0=x;
        y0=y;
        outcode0 = computeoutcode(x0,y0);
    }
else
    {
        x1=x;
        y1=y;
        outcode1 = computeoutcode(x1,y1);
    }
}while(!done);
if(accept)
    {
        double sx=(xumax-xumin)/(xmax-xmin);
        double sy=(yumax-yumin)/(ymax-ymin);
        double vx0 = xumin+(x0-xmin)*sx;
        double vy0 = yumin+(y0-ymin)*sy;
        double vx1 = xumin+(x1-xmin)*sx;
        double vy1 = yumin+(y1-ymin)*sy;
            glcolor3f(1,0,0);
            glBegin(GL_LINE_LOOP);
            glcolor3f(0,0,1)
            glBegin(GL_LINES)
            glEnd();
        }
}
```

Date _____

Expt. No. _____ Page No. _____

```
void display()
{
    glClear (GL_COLOR_BUFFER_BIT);
    glColor3f(0,0,1);
    glBegin(GL_LINE_LOOP);
    glVertex2f (xmin,ymin);
    glVertex2f (xmin,ymin);
    glVertex2f (xmax,ymax);
    glVertex2f (xmin,ymax);
    glEnd();
    for(int i=0;i<n;i++)
        {
            glBegin (GL_LINES);
            glVertex2d (ls[i].x1, ls[i].y1);
            glVertex2d (ls[i].x2, ls[i].y2);
            glEnd();
        }
    for(int i=0;i<n;i++)
            cohensutha(ls[i].x1, ls[i].y1, ls[i].x2, ls[i].y2);

    glFlush();
}

void myInit()
{
    glClearColor(1.1,1,1);
    glColor3f (1,0,0);
    glPointSize(1.0);
```

Teacher's Signature _____

```
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0,500,0,500);
}

void main(int argc, char** argv)
{
    printf("Enter window coordinates (xmin,ymin,xmax,ymax):\n");
    scanf_s("%f %f %f %f", &xmin, &ymin, &xmax, &ymax);
    printf("Enter viewport coordinates (xvmin,yvmin, xvmax yvmax):\n");
    scanf_s("%f %f %f %f", &xvmin, &yvmin, &xvmax, &yvmax);
    printf("Enter no. of lines:\n");
    scanf_s("%d", &n);
    for(int i=0; i<n; i++)
    {
        printf("Enter line endpoint (x1,y1,x2,y2):\n");
        scanf_s("%d %d %d %d", &ls[i].x1, &ls[i].y1, &ls[i].x2,
                                                    &ls[i].y2);
    }
    glutInit(&argc,argv);
    glutCreateWindow("Clip");
    myinit();
    glutDisplayFunc(display);
    glutMainLoop();
}
```

OUTPUT:-

```
Enter window coordinates (xmin ymin xmax ymax):
100 100 250 250
Enter viewport coordinates (xvmin yvmin xvmax yvmax) :
300 300 400 400
Enter no. of lines:
4
Enter line endpoints (x1 y1 x1 y2):
120 140 200 170
Enter line endpoints (x1 y1 x2 y2):
40 130 200 210
Enter line endpoints (x1 y1 x2 y2):
30 125 125 350
Enter line endpoints (x1 y1 x2 y2):
160 190 190 110
```