```
#include<windows.h>
#include<bits/stdc++.h>
#include<GL/glut.h>

using namespace std;

int width = 1000, height = 500, window;

static int cw_coordinates[4];
vector<pair<int, int> > lines;

bool first_time = true, clipping_window_made = false, lines_drawn = false;
int min_x, max_x, min_y, max_y;

int Centre = 0, Left = 1, Right = 2, Bottom = 4, Top = 8;
int temp = max_x;

int return_position(int x, int y)
{
    int position = Centre;

    if (x < min_x)
        position |= Left;
    else if (x > max_x)
        position |= Right;
    if (y < min_y)
        position |= Bottom;
    else if (y > max_y)
        position |= Top;

    return position;
}

void clipping_algorithm(int i)
{
    int x , y, x1, y1, x2, y2;
    float m;
    x1 = lines[i].first;
    y1 = lines[i].second;
    x2 = lines[i+1].first;
    y2 = lines[i+1].second;
    m = (y2 - y1)/(x2 - x1);

    int pos1 = return_position(x1, y1);
    int pos2 = return_position(x2, y2);

    while(1)
    {
        if( pos1 == 0 && pos2 == 0)
            break;
```

```
    else if(pos1 & pos2)
        break;
    else
    {
        int pos_output;

        if (pos1 != 0)
            pos_output = pos1;
        else
            pos_output = pos2;
        if (pos_output & Top)
        {
            x = x1 + (x2 - x1) * (max_y - y1) / (y2 - y1);
            y = max_y;
        }
        else if (pos_output & Bottom)
        {
            x = x1 + (x2 - x1) * (min_y - y1) / (y2 - y1);
            y = min_y;
        }
        else if (pos_output & Right)
        {
            y = y1 + (y2 - y1) * (max_x - x1) / (x2 - x1);
            x = max_x;
        }
        else if (pos_output & Left)
        {
            y = y1 + (y2 - y1) * (min_x - x1) / (x2 - x1);
            x = min_x;
        }
        if (pos_output == pos1)
        {
            x1 = x;
            y1 = y;
            pos1 = return_position(x1, y1);
        }
        else
        {
            x2 = x;
            y2 = y;
            pos2 = return_position(x2, y2);
        }
    }
}

lines[i].first = x1;
lines[i].second = y1;
lines[i+1].first = x2;
lines[i+1].second = y2;
```

```
    if(pos1 & pos2)
    {
        lines[i].first = temp;
        lines[i].second = temp;
        lines[i+1].first = ++temp;
        lines[i+1].second = ++temp;
    }
}

void start_clipping()
{
    min_x = cw_coordinates[0];
    max_x = cw_coordinates[2];

    min_y = cw_coordinates[3];
    max_y = cw_coordinates[1];

    for(int i = 0; i < lines.size(); i += 2)
        clipping_algorithm(i);

    glutPostRedisplay();
}

void menu_func(int n)
{
    if(n == -1)
        glutDestroyWindow(window);
    else if(n == 1)
        start_clipping();
}

void createmenu()
{
    glutCreateMenu(menu_func);
        glutAddMenuEntry("Clip", 1);
        glutAddMenuEntry("Exit", -1);
    glutAttachMenu(GLUT_LEFT_BUTTON);
}

void mouse(int button,int state,int x,int y)
{
    if(button == GLUT_LEFT_BUTTON && state == GLUT_UP && first_time == false)
        clipping_window_made = true;
    if(clipping_window_made && !lines_drawn && button == GLUT_LEFT_BUTTON && state
== GLUT_DOWN)
            lines.push_back({x, height - y});
    if(clipping_window_made && button == GLUT_RIGHT_BUTTON && state == GLUT_DOWN &&
!lines_drawn)
    {
        lines_drawn = true;
```

```
        createmenu();
    }

    glutPostRedisplay();

}

void motion(int x, int y)
{
    if(!clipping_window_made && first_time)
    {
        cw_coordinates[0] = x;
        cw_coordinates[1] = height - y;
        cw_coordinates[2] = x;
        cw_coordinates[3] = height - y;
        first_time = false;
    }
    else if(!clipping_window_made && !first_time)
    {
        cw_coordinates[2] = x;
        cw_coordinates[3] = height - y;
    }
    glutPostRedisplay();
}

void keyboard(unsigned char key, int x, int y)
{
    if(key == 'c' && lines_drawn)
        start_clipping();
}

void display()
{
    glClear(GL_COLOR_BUFFER_BIT);

    // Drawing the Clipping Window
    glColor3f(0.0f, 1.0f, 1.0f);
    glBegin(GL_LINE_LOOP);
        glVertex2i(cw_coordinates[0], cw_coordinates[1]);
        glVertex2i(cw_coordinates[0], cw_coordinates[3]);
        glVertex2i(cw_coordinates[2], cw_coordinates[3]);
        glVertex2i(cw_coordinates[2], cw_coordinates[1]);
    glEnd();

    // Drawing various lines
    if(clipping_window_made)
    {
        glColor3f(1.0f, 1.0f, 0.0f);
        glBegin(GL_LINES);
            for(int i=0; (i + 1)<lines.size(); i+=2)
```

```
                {
                    glVertex2i(lines[i].first, lines[i].second);
                    glVertex2i(lines[i+1].first, lines[i+1].second);
                }
                glEnd();
    }
    glFlush();
}

void initGL()
{
        glClearColor(0.0f,0.0f,0.0f,0.0f);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0, width, 0, height);
    glFlush();
}

int main(int argc,char **argv)
{
    glutInit(&argc,argv);
    glutInitWindowSize(width, height);
    glutInitWindowPosition(0, 0);
    window = glutCreateWindow("Cohen-Sutherland Line Clipping Algorithm");

    glutDisplayFunc(display);
    glutMotionFunc(motion);
    glutMouseFunc(mouse);
    glutKeyboardFunc(keyboard);
    initGL();
    glutMainLoop();

    return 0;

}
```