Develop a program to clip a line using Cohen Sutherland algorithm.

```
#include <GL/glut.h>
#include <iostream>
using namespace std;

const int INSIDE = 0;
const int LEFT = 1;
const int RIGHT = 2;
const int BOTTOM = 4;
const int TOP = 8;

int xMin = 50, yMin = 50, xMax = 300, yMax = 300;

struct Line {
   int x1, y1, x2, y2;
};

const int MAX_LINES = 10;
Line lines[MAX_LINES];
int lineCount = 0;

int getCode(int x, int y) {
   int code = INSIDE;
   if (x < xMin) code |= LEFT;
   else if (x > xMax) code |= RIGHT;
```

```
    if (y < yMin) code |= BOTTOM;
    else if (y > yMax) code |= TOP;
    return code;
}

void clipLine(int &x1, int &y1, int &x2, int &y2) {
    int code1 = getCode(x1, y1);
    int code2 = getCode(x2, y2);
    bool done = false;
    while (true) {
        if ((code1 == 0) && (code2 == 0)) {
            done = true;
            break;
        } else if (code1 & code2) {
            return;
        } else {
            int outsideCode, x, y;
            if (code1 != 0) outsideCode = code1;
            else outsideCode = code2;

            if (outsideCode & TOP) {
                x = x1 + (x2 - x1) * (yMax - y1) / (y2 - y1);
                y = yMax;
            }
            else if (outsideCode & BOTTOM) {
                x = x1 + (x2 - x1) * (yMin - y1) / (y2 - y1);
```

```
            y = yMin;
        } else if (outsideCode & RIGHT) {
            y = y1 + (y2 - y1) * (xMax - x1) / (x2 - x1);
            x = xMax;
        }
        else if (outsideCode & LEFT) {
            y = y1 + (y2 - y1) * (xMin - x1) / (x2 - x1);
            x = xMin;
        }

        if (outsideCode == code1) {
            x1 = x;
            y1 = y;
            code1 = getCode(x1, y1);
        } else {
            x2 = x;
            y2 = y;
            code2 = getCode(x2, y2);
        }
      }
   }
}

void drawClippingWindow() {
   glColor3f(1.0, 0.0, 0.0);
   glBegin(GL_LINE_LOOP);
```

```
    glVertex2i(xMin, yMin);
    glVertex2i(xMax, yMin);
    glVertex2i(xMax, yMax);
    glVertex2i(xMin, yMax);
    glEnd();
}

void drawLine(int x1, int y1, int x2, int y2) {
    glBegin(GL_LINES);
    glVertex2i(x1, y1);
    glVertex2i(x2, y2);
    glEnd();
}

void display() {
    glClear(GL_COLOR_BUFFER_BIT);
    drawClippingWindow();

    for (int i = 0; i < lineCount; i++) {
        glColor3f(0.0, 1.0, 0.0);
        drawLine(lines[i].x1, lines[i].y1, lines[i].x2, lines[i].y2);

        int clippedX1 = lines[i].x1, clippedY1 = lines[i].y1;
        int clippedX2 = lines[i].x2, clippedY2 = lines[i].y2;

        clipLine(clippedX1, clippedY1, clippedX2, clippedY2);
```

```
        glColor3f(0.0, 0.0, 1.0);
        drawLine(clippedX1, clippedY1, clippedX2, clippedY2);
    }

    glFlush();
}

void init() {
    glClearColor(0.0, 0.0, 0.0, 1.0);
    glColor3f(0.0, 0.0, 0.0);
    glMatrixMode(GL_PROJECTION);
    glOrtho(0.0, 400.0, 0.0, 400.0, -1.0, 1.0);
}

int main(int argc, char** argv) {
    cout << "Enter the number of lines (max " << MAX_LINES << "): ";
    cin >> lineCount;

    if (lineCount > MAX_LINES) {
        cout << "Exceeds maximum number of lines (" << MAX_LINES << ").
Exiting.\n";
        return 1;
    }

    for (int i = 0; i < lineCount; i++) {
        cout << "Enter line " << i + 1 << " coordinates (x1, y1, x2, y2): ";
```

```
        cin >> lines[i].x1 >> lines[i].y1 >> lines[i].x2 >> lines[i].y2;
    }

    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(500, 500);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("Cohen-Sutherland Line Clipping Algorithm");
    init();
    glutDisplayFunc(display);
    glutMainLoop();

    return 0;
}
```