LAB-5

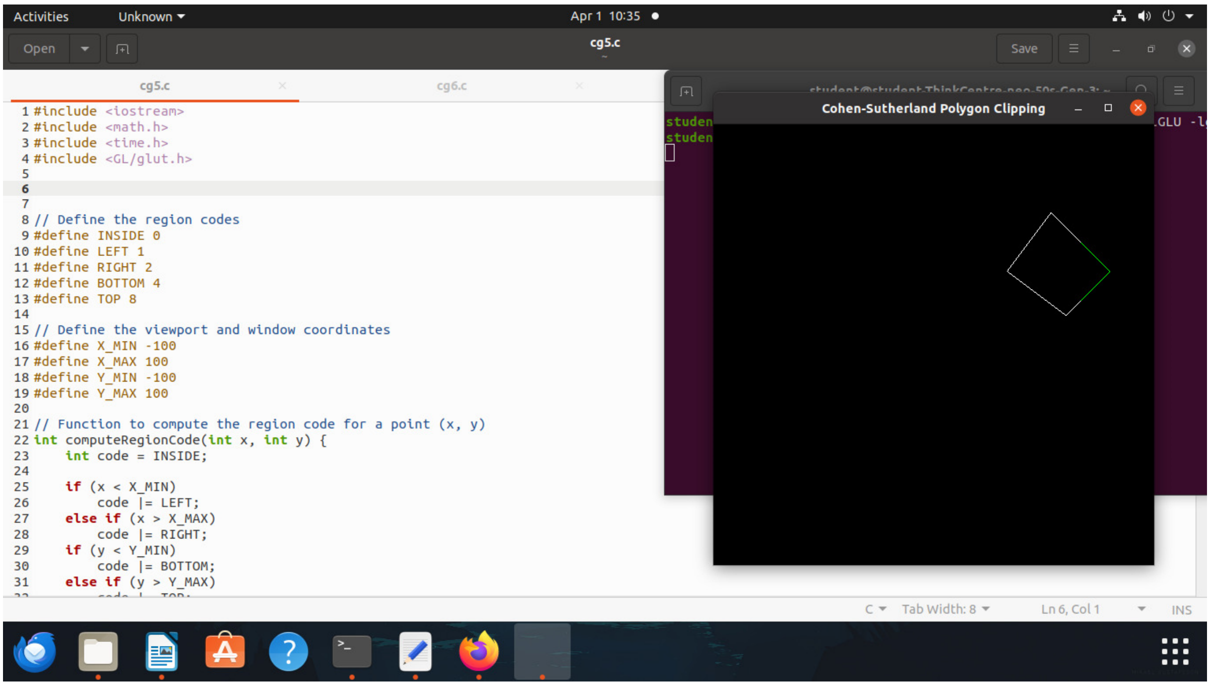
Implement Cohen Sutherland polygon clipping method to clip the polygon with respect the viewport and window. Use mouse click, keyboard interface.

Source Code :

```
#include <GL/glut.h>
#include <stdio.h>

#define INSIDE 0
#define LEFT 1
#define RIGHT 2
#define BOTTOM 4
#define TOP 8

#define X_MIN -100
#define X_MAX 100
#define Y_MIN -100
#define Y_MAX 100

int computeRegionCode(int x, int y) {
    int code = INSIDE;

    if (x < X_MIN)
        code |= LEFT;
    else if (x > X_MAX)
        code |= RIGHT;
    if (y < Y_MIN)
        code |= BOTTOM;
    else if (y > Y_MAX)
```

```
        code |= TOP;

    return code;
}

void clipLine(int x0, int y0, int x1, int y1) {
    int code0 = computeRegionCode(x0, y0);
    int code1 = computeRegionCode(x1, y1);
    int accept = 0;

    while (1) {
        if (!(code0 | code1)) { // Both endpoints are inside
            accept = 1;
            break;
        } else if (code0 & code1) { // Both endpoints are outside
            break;
        } else { // Some part of the line is inside
            int x, y;

            int code = code0 ? code0 : code1;

            if (code & TOP) { // Point is above the window
                x = x0 + (x1 - x0) * (Y_MAX - y0) / (y1 - y0);
                y = Y_MAX;
            } else if (code & BOTTOM) { // Point is below the window
                x = x0 + (x1 - x0) * (Y_MIN - y0) / (y1 - y0);
                y = Y_MIN;
            } else if (code & RIGHT) { // Point is to the right of the window
                y = y0 + (y1 - y0) * (X_MAX - x0) / (x1 - x0);
                x = X_MAX;
            } else if (code & LEFT) { // Point is to the left of the window
                y = y0 + (y1 - y0) * (X_MIN - x0) / (x1 - x0);
                x = X_MIN;
            }

            if (code == code0) {
                x0 = x;
                y0 = y;
                code0 = computeRegionCode(x0, y0);
            } else {
                x1 = x;
                y1 = y;
                code1 = computeRegionCode(x1, y1);
            }
        }
    }

    if (accept) {
        glColor3f(1.0, 1.0, 1.0); // Set color to white
        glBegin(GL_LINES);
        glVertex2i(x0, y0);
        glVertex2i(x1, y1);
        glEnd();
    }
}

void display() {
    glClear(GL_COLOR_BUFFER_BIT);

    glColor3f(0.0, 1.0, 0.0); // Set color to green
    glBegin(GL_LINE_LOOP);
    glVertex2i(50, 50);
    glVertex2i(80, 90);
    glVertex2i(120, 50);
    glVertex2i(90, 20);
    glEnd();

    clipLine(50, 50, 80, 90);
    clipLine(80, 90, 120, 50);
    clipLine(120, 50, 90, 20);
    clipLine(90, 20, 50, 50);

    glFlush();
}

void init() {
    glClearColor(0.0, 0.0, 0.0, 1.0); // Set clear color to black
    gluOrtho2D(-150, 150, -150, 150); // Set orthographic projection
}
int main(int argc, char **argv) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(500, 500); // Set window size
    glutInitWindowPosition(100, 100); // Set window position
    glutCreateWindow("Cohen-Sutherland Polygon Clipping");
    init();
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}
```

Output :



```
#include <iostream>
#include <math.h>
#include <time.h>
#include <GL/glut.h>



// Define the region codes
#define INSIDE 0
#define LEFT 1
#define RIGHT 2
#define BOTTOM 4
#define TOP 8

// Define the viewport and window coordinates
#define X_MIN -100
#define X_MAX 100
#define Y_MIN -100
#define Y_MAX 100

// Function to compute the region code for a point (x, y)
int computeRegionCode(int x, int y) {
    int code = INSIDE;

    if (x < X_MIN)
        code |= LEFT;
    else if (x > X_MAX)
        code |= RIGHT;
    if (y < Y_MIN)
        code |= BOTTOM;
    else if (y > Y_MAX)
```