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

int points[4][2];
int clicks = 0;
bool curveDrawn = false;

// Function to draw a single point
void drawPoint(GLfloat x, GLfloat y)
{
    glColor3f(0.0, 1.0, 0.0);
    glPointSize(5.0f);
    glBegin(GL_POINTS);
    glVertex2f(x, y);
    glEnd();
    glFlush();
}

// Function to draw a line between two points
void drawLine(GLfloat x1, GLfloat y1, GLfloat x2, GLfloat y2)
{
    glColor3f(0.0, 1.0, 0.0);
    glBegin(GL_LINES);
    glVertex2f(x1, y1);
    glVertex2f(x2, y2);
    glEnd();
    glFlush();
}

// Recursive function for midpoint subdivision to draw Bezier curve
void midPointSubDivision(GLfloat x1, GLfloat y1, GLfloat x2, GLfloat y2,
                         GLfloat x3, GLfloat y3, GLfloat x4, GLfloat y4)
{
    GLfloat xAB = (x2 + x1) / 2;
    GLfloat yAB = (y2 + y1) / 2;
    GLfloat xBC = (x3 + x2) / 2;
    GLfloat yBC = (y3 + y2) / 2;
    GLfloat xCD = (x4 + x3) / 2;
    GLfloat yCD = (y4 + y3) / 2;

    GLfloat xABC = (xAB + xBC) / 2;
    GLfloat yABC = (yAB + yBC) / 2;
    GLfloat xBCD = (xBC + xCD) / 2;
    GLfloat yBCD = (yBC + yCD) / 2;

    GLfloat xABCD = (xABC + xBCD) / 2;
    GLfloat yABCD = (yABC + yBCD) / 2;
    // If the segment is long enough, subdivide further
    if (((x2 - x1) * (x2 - x1) + (y2 - y1) * (y2 - y1)) > 4 ||
        ((x3 - x2) * (x3 - x2) + (y3 - y2) * (y3 - y2)) > 4 ||
        ((x4 - x3) * (x4 - x3) + (y4 - y3) * (y4 - y3)) > 4)
    {
        midPointSubDivision(x1, y1, xAB, yAB, xABC, yABC, xABCD, yABCD);
        midPointSubDivision(xABCD, yABCD, xBCD, yBCD, xCD, yCD, x4, y4);
    }
    else
    {
        drawLine(x1, y1, xABCD, yABCD);
        drawLine(xABCD, yABCD, x4, y4);
    }
}

// Initialization
void init()
{
    glClearColor(0.0, 0.0, 0.0, 1.0); // Black background
    glClear(GL_COLOR_BUFFER_BIT);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0, 600, 0, 600); // 2D orthographic projection
}

// Display callback
void display()
{
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(0.0, 1.0, 0.0);
    if (curveDrawn)
    {
        for (int i = 0; i < 4; ++i)
        {
            drawPoint(points[i][0], points[i][1]);
        }
    }
    glFlush();
}

// Mouse callback
void mouse(int btn, int state, int x, int y)
{
    if (state == GLUT_DOWN && btn == GLUT_LEFT_BUTTON)
    {
        if (clicks < 4)
        {
            points[clicks][0] = x;
            points[clicks][1] = 600 - y; // Flip y for OpenGL coordinate system
            drawPoint(x, 600 - y);
            clicks++;
        }

        if (clicks == 4)
        {
            curveDrawn = true;
            // Draw control polygon (optional)
            drawLine(points[0][0], points[0][1], points[1][0], points[1][1]);
            drawLine(points[1][0], points[1][1], points[2][0], points[2][1]);
            drawLine(points[2][0], points[2][1], points[3][0], points[3][1]);

            // Draw Bezier curve
            midPointSubDivision(points[0][0], points[0][1],
                                points[1][0], points[1][1],
                                points[2][0], points[2][1],
                                points[3][0], points[3][1]);
        }
    }
}

int main(int argc, char **argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowPosition(100, 100);
    glutInitWindowSize(600, 600);
    glutCreateWindow("Bezier Curve - Midpoint Subdivision");
    init();
    glutDisplayFunc(display);
    glutMouseFunc(mouse);
    glutMainLoop();
    return 0;
}
```