SSN COLLEGE OF ENGINEERING, KALAVAKKAM

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

UCS1712-Graphics and Multimedia Lab

Programming Assignment 3

Bresenham's Line Drawing Algorithm in C++ using OpenGL

Name: Jayannthan P T *Dept: CSE 'A'* *Roll No.: 205001049*

To plot points that make up the line with endpoints (x0,y0) and (xn,yn) using Bresenham's line drawing

algorithm.

Case 1: +ve slope Left to Right line

Case 2: +ve slope Right to Left line

Case 3: -ve slope Left to Right line

Case 4: -ve slope Right to Left line

Each case has two subdivisions

(i) |m|<= 1 (ii) |m|>1

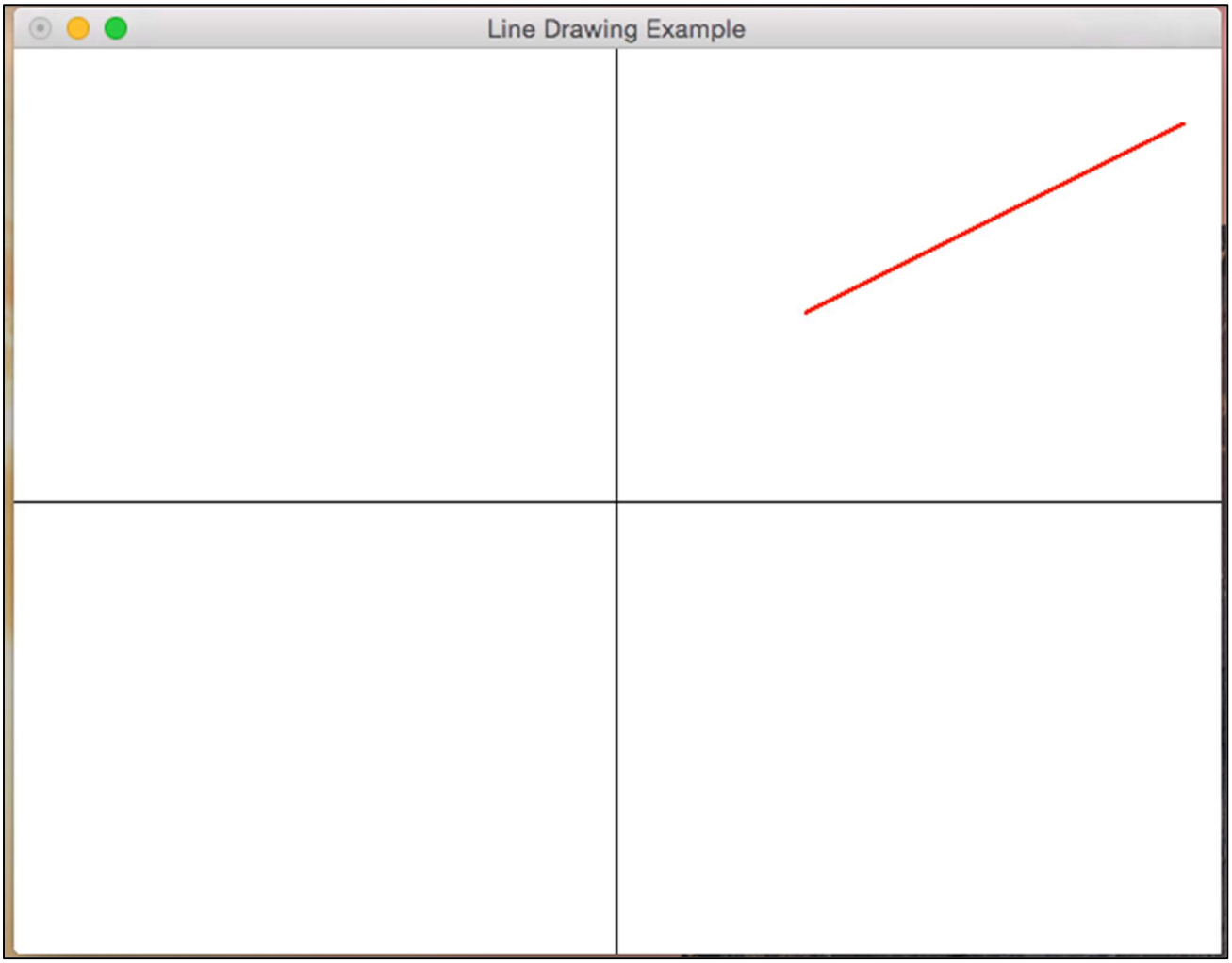
Note that all four cases of line drawing must be given as test cases.

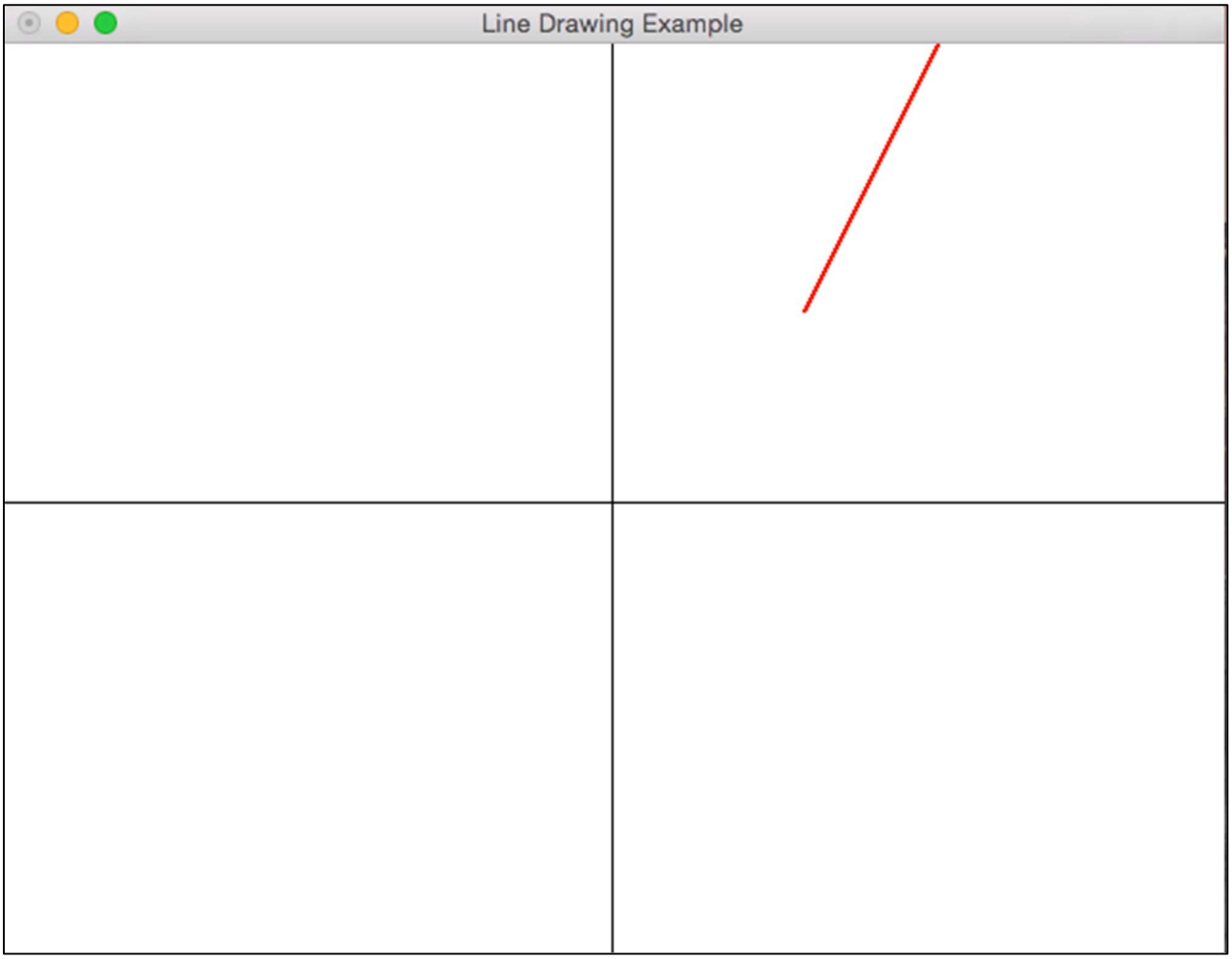
Source code:

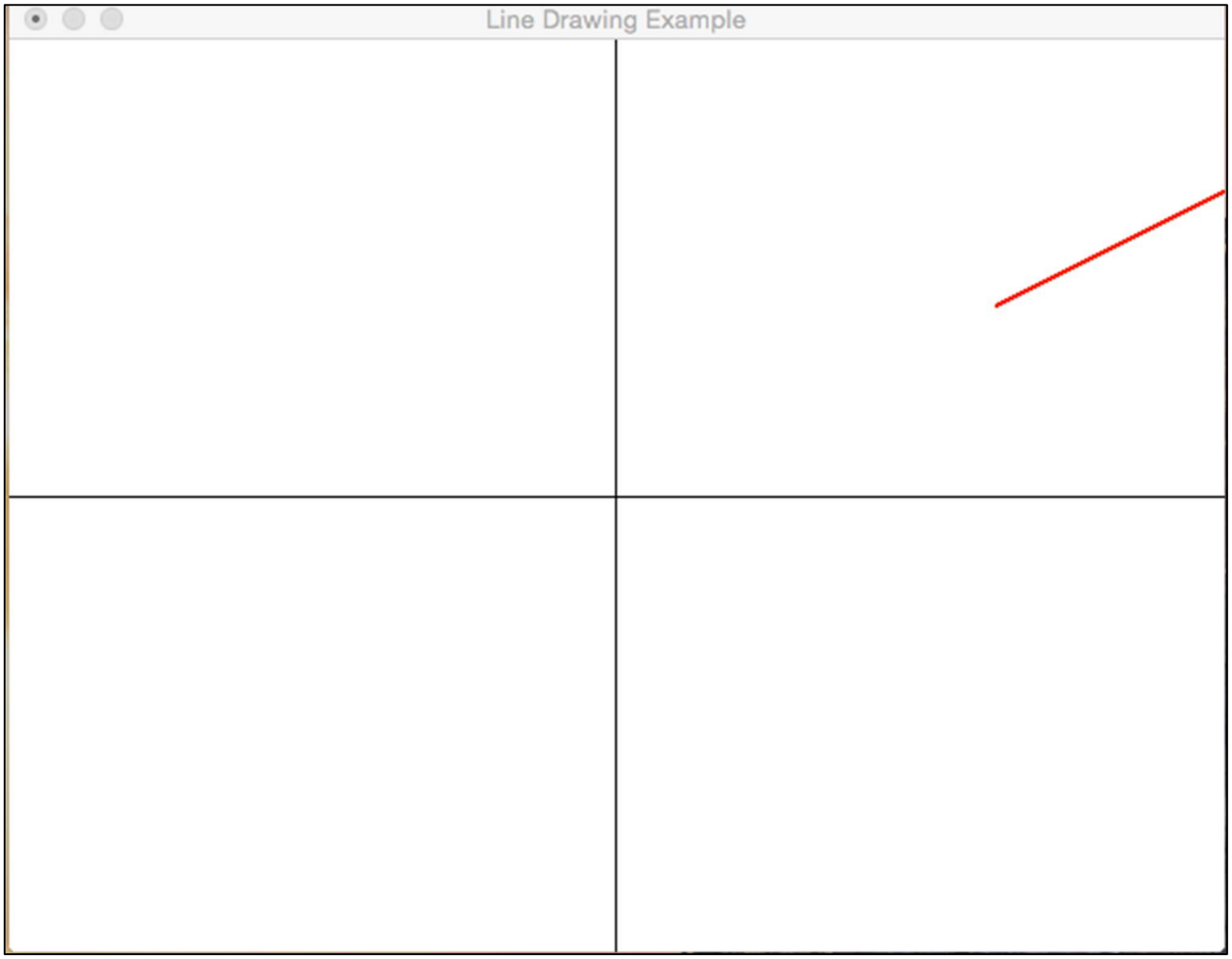
```
#include <iostream>
#include <GLUT/glut.h>
#include <cmath>
using namespace std;

int flag = 0;
int x_1 = 0, y_1 = 0, x_2 = 0, y_2 = 0;
string cnt = "YES";

void myInit()
{
    glClearColor(1.0, 1.0, 1.0, 0.0);
    glPointSize(2);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0.0, 640.0, 0.0, 480.0);
}

void drawLine()
{
```

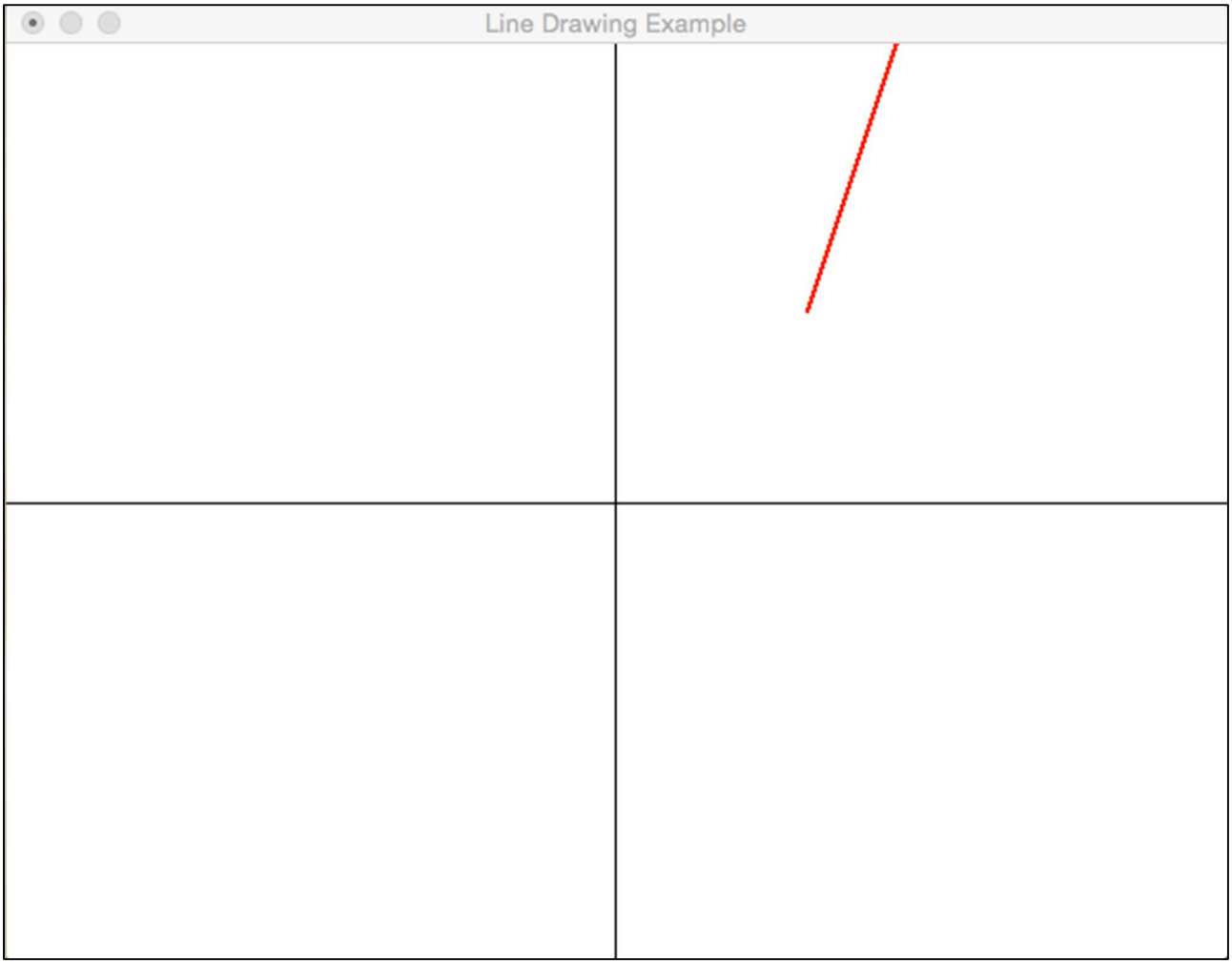
```
    x_1 += 320;
    y_1 += 240;
    x_2 += 320;
    y_2 += 420;
    glColor3f(1.0f, 0.0f, 0.0f);
    float dx = x_2 - x_1;
    float dy = y_2 - y_1;
    int p = 2 * dy - dx;
    glBegin(GL_POINTS); // Begin drawing pointswhile (x_1 < x_2)
    {
        glVertex2f(x_1, y_1);
        x_1++;
        if (x_1 >= 0)
        {
            p -= 2 * dx;
            y_1++;
        }
        p += 2 * dx;
    }
    glEnd();
    // End drawing points
}

void drawAxes()
{
    glBegin(GL_LINES);
    // Draw X-axis
    glColor3f(0.0f, 0.0f, 0.0f); // Set color to black
    glVertex2f(0, 240);          // X-axis starting point
    glVertex2f(640, 240);        // X-axis ending point
    // Draw Y-axis
    glVertex2f(320, 0);   // Y-axis starting point
    glVertex2f(320, 480); // Y-axis ending point
    glEnd();              // End drawing lines
}

void myDisplay()
{
    glClear(GL_COLOR_BUFFER_BIT);
    if (flag == 0)
    {
        flag = 1;
        x_1 = 0, y_1 = 0, x_2 = 0, y_2 = 0;
        cout << "Point 1 : ";
        cin >> x_1 >> y_1;
        cout << "Point 2 : ";
        cin >> x_2 >> y_2;
    }
    drawAxes();
    drawLine();
    glFlush();
    cout << "Want to continue (YES/NO) : ";
    cin >> cnt;
    if (cnt == "NO")
```

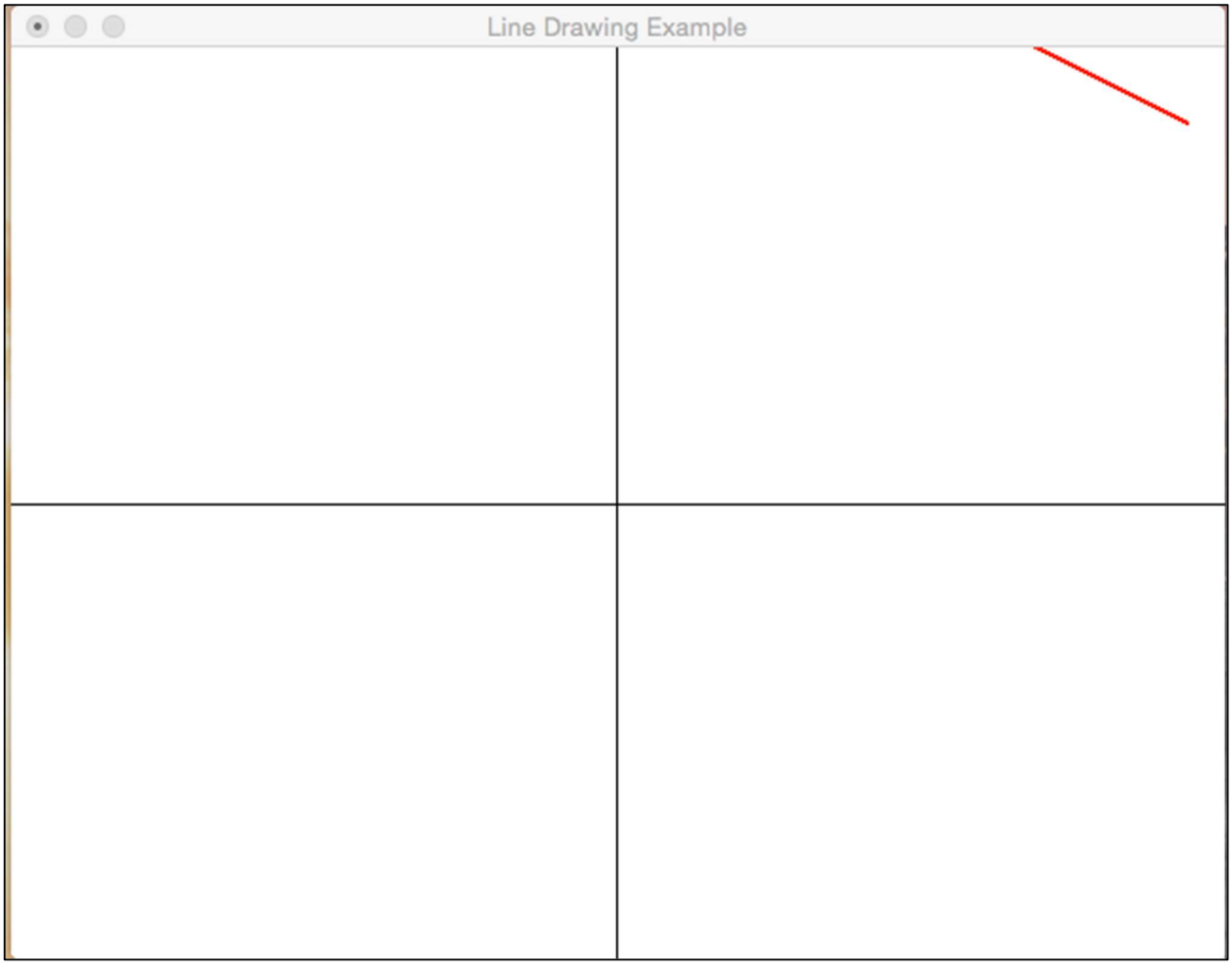
```
    {
        cout << "Exiting...\n";
        exit(0);
    }
    flag = 0;             // Reset flag for the next line
    glutPostRedisplay(); // Continue updating the display
}

int main(int argc, char **argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(640, 480);
    glutCreateWindow("Line Drawing Example");
    myInit();
    glutDisplayFunc(myDisplay);
    glutMainLoop();
    return 0;
}
```

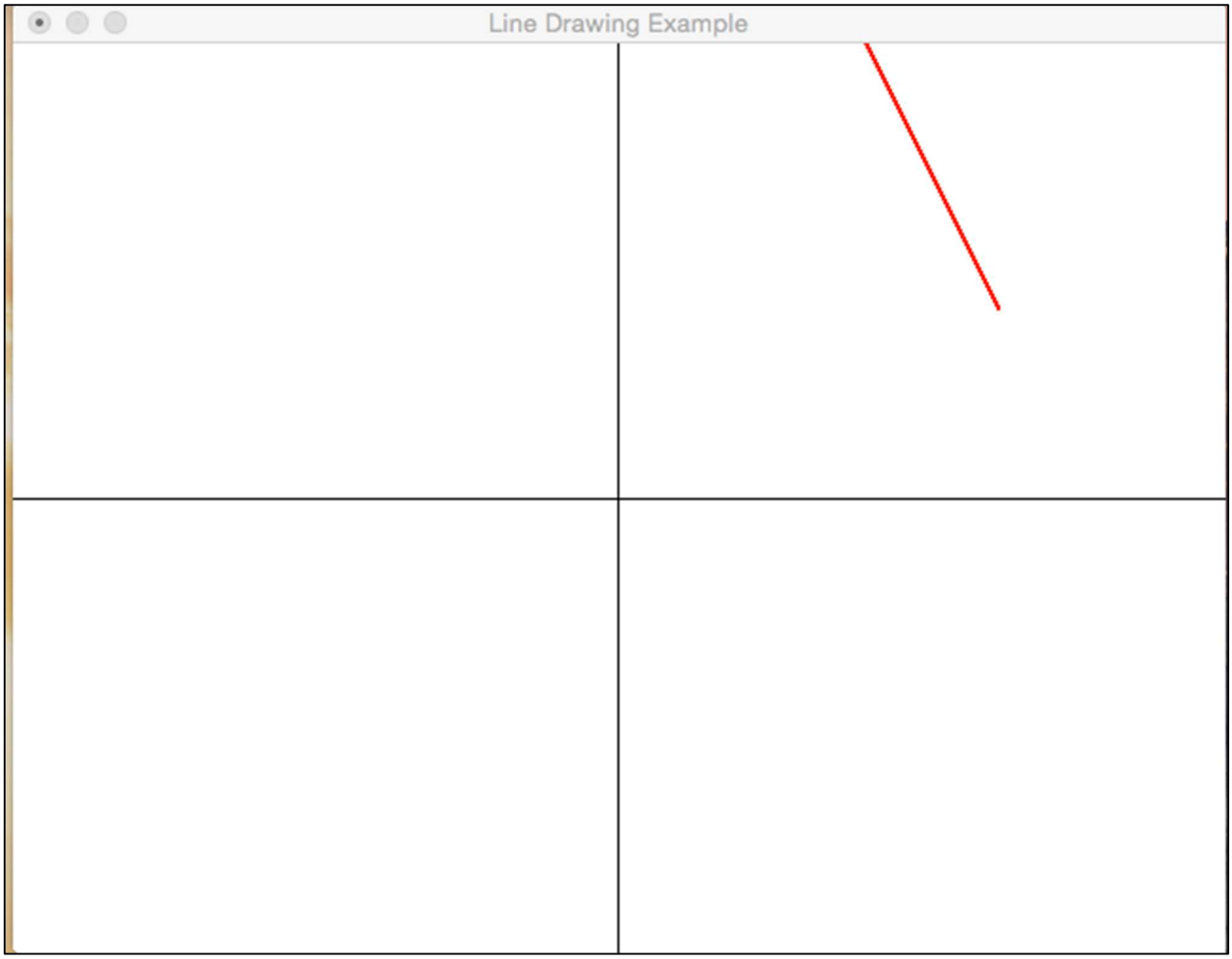

Line Drawing Example

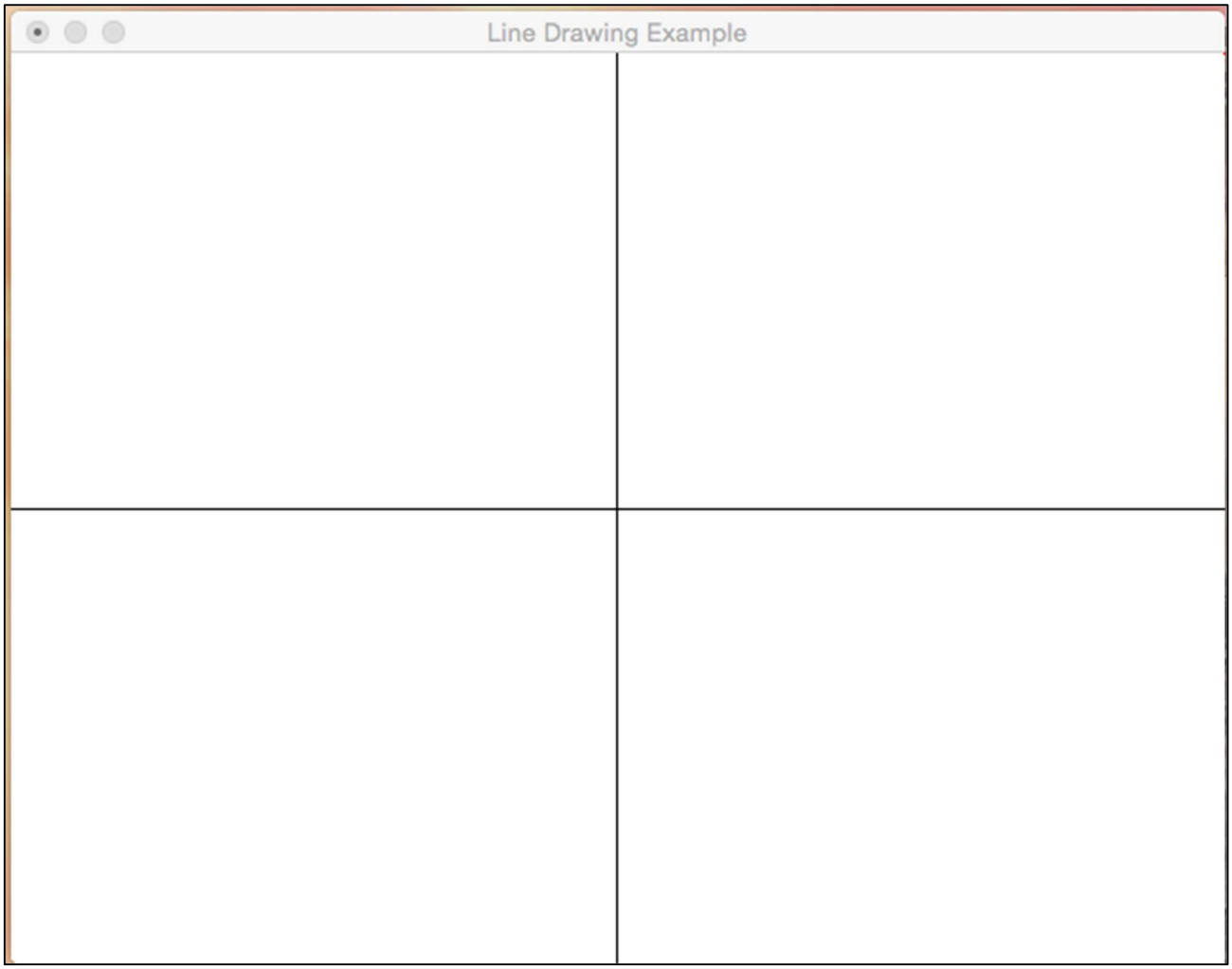

Line Drawing Example

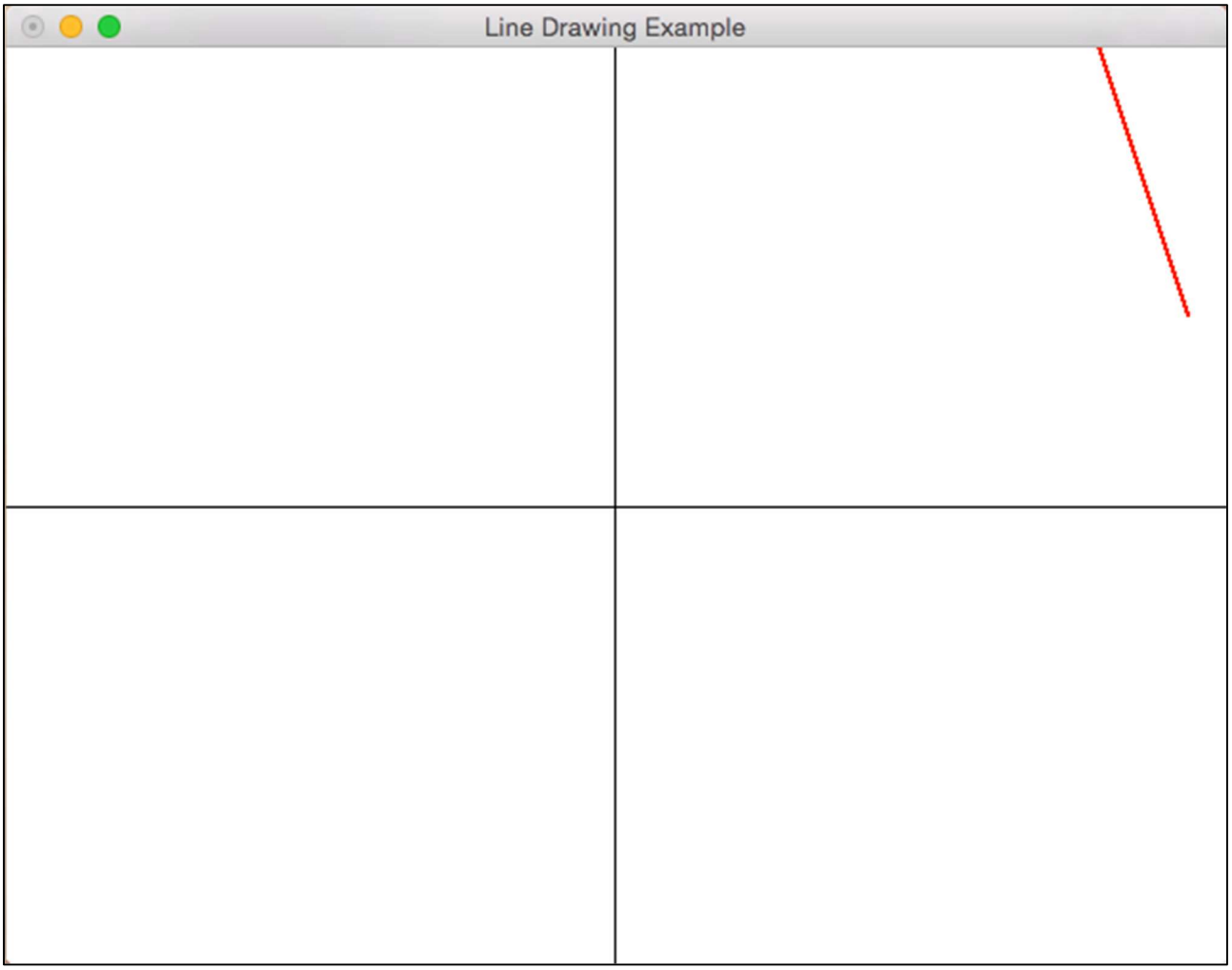

Line Drawing Example


Line Drawing Example


Line Drawing Example


Line Drawing Example


Line Drawing Example




```
Point 1 : 100 100
Point 2 : 300 200
Want to continue (YES/NO) : YES
Sep  4 10:31:18 ssn-macs-Mac-mini.local DDA
update.
Point 1 : 100 100
Point 2 : 200 300
Want to continue (YES/NO) : YES
Sep  4 10:33:14 ssn-macs-Mac-mini.local DDA
update.
Point 1 : 400 200
Point 2 : 200 100
Want to continue (YES/NO) : YES
Point 1 : 200 400
Point 2 : 100 100
Want to continue (YES/NO) : YES
Sep  4 10:36:37 ssn-macs-Mac-mini.local DDA
update.
Point 1 : 100 300
Point 2 : 300 200
Want to continue (YES/NO) : YES
Sep  4 10:37:10 ssn-macs-Mac-mini.local DDA
update.
Point 1 : 100 300
Point 2 : 200 100
Want to continue (YES/NO) : YES
Sep  4 10:38:16 ssn-macs-Mac-mini.local DDA
update.
Point 1 : 400 200 200 300
Point 2 : Want to continue (YES/NO) : YES
```

```
Point 1 : 400 200 200 300
Point 2 : Want to continue (YES/NO) : YES
Sep  4 10:38:46 ssn-macs-Mac-mini.local DDA[8
update.
Point 1 : 200 400
Point 2 : 300 100
Want to continue (YES/NO) : NO
```