SCAN LINE ALGORITHM

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

// Define polygon vertices
float x1 = 250.0, y1 = 200.0;
float x2 = 150.0, y2 = 300.0;
float x3 = 250.0, y3 = 400.0;
float x4 = 350.0, y4 = 300.0;

// Draw a pixel at (x, y)
void draw_pixel(int x, int y) {
   glBegin(GL_POINTS);
   glVertex2i(x, y);
   glEnd();
}

// Detect and update edge boundaries
void edgedetect(float x1, float y1, float x2, float y2, int le[], int re[]) {
   if (y1 > y2) {
      // Swap points to ensure y1 <= y2
      float temp = x1; x1 = x2; x2 = temp;
      temp = y1; y1 = y2; y2 = temp;
   }

   float slope = (y1 == y2) ? 0 : (x2 - x1) / (y2 - y1); // Horizontal or sloped edge
   float x = x1;

   for (int y = (int)y1; y <= (int)y2; y++) {
```

```
      if (x < le[y]) le[y] = (int)x; // Update left edge
      if (x > re[y]) re[y] = (int)x; // Update right edge
      x += slope;
    }
}

// Fill the polygon using the scan line algorithm
void scanfill() {
    int le[500], re[500]; // Left and right edge arrays

    // Initialize edge arrays
    for (int i = 0; i < 500; i++) {
        le[i] = 500;
        re[i] = 0;
    }

    // Detect edges for the polygon
    edgedetect(x1, y1, x2, y2, le, re);
    edgedetect(x2, y2, x3, y3, le, re);
    edgedetect(x3, y3, x4, y4, le, re);
    edgedetect(x4, y4, x1, y1, le, re);

    // Fill pixels between the edges for each scan line
    for (int y = 0; y < 500; y++) {
        if (le[y] <= re[y]) {
            for (int x = le[y]; x < re[y]; x++) {
                draw_pixel(x, y);
            }
        }
```

```
    }
}

// Display the filled polygon
void display() {
    glClear(GL_COLOR_BUFFER_BIT);

    // Draw polygon outline
    glColor3f(0.0, 0.0, 1.0); // Blue outline
    glBegin(GL_LINE_LOOP);
    glVertex2f(x1, y1);
    glVertex2f(x2, y2);
    glVertex2f(x3, y3);
    glVertex2f(x4, y4);
    glEnd();

    // Fill the polygon
    glColor3f(0.0, 1.0, 1.0); // Cyan fill color
    scanfill();

    glFlush();
}

// Initialize OpenGL
void init() {
    glClearColor(1.0, 1.0, 1.0, 1.0); // White background
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0.0, 500.0, 0.0, 500.0); // 2D orthographic projection
```

```
}

// Main function
int main(int argc, char **argv) {
   glutInit(&argc, argv);
   glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
   glutInitWindowSize(500, 500);
   glutCreateWindow("Scan Line Polygon Fill");
   glutDisplayFunc(display);
   init();
   glutMainLoop();
   return 0;
}
```