**NAME:- SURWADE TRISHARAN RAJESH**

**ROLL NO:- 47**

**CLASS -MSC(CS)-IInd Year**

**4. Polygon filling algorithm**

#include <graphics.h>

#include <conio.h>

#include <iostream.h>

#include <dos.h>

class Polygon {

private:

int n; // Number of edges (vertices)

int a[20][2]; // Coordinates of the vertices

float s[20]; // Slopes of the edges

int xi[20]; // X-intersections for the scanline algorithm

public:

// Constructor to initialize the polygon

Polygon(int edges) : n(edges) {

int i; // Declare 'i' outside loop to avoid multiple declarations

// Input the coordinates for the vertices

cout << "Enter the coordinates (X Y) of the polygon vertices:\n";

for (i = 0; i < n; i++) {

cout << "X" << i << " Y" << i << ": ";

cin >> a[i][0] >> a[i][1];

}

// Close the polygon by connecting the last vertex to the first

a[n][0] = a[0][0];

a[n][1] = a[0][1];

}

// Method to calculate slopes of the edges

void calculateSlopes() {

int i; // Reuse 'i' for each loop

for (i = 0; i < n; i++) {

int dy = a[i + 1][1] - a[i][1];

int dx = a[i + 1][0] - a[i][0];

if (dy == 0)

s[i] = 1.0; // Horizontal line

else if (dx == 0)

s[i] = 0.0; // Vertical line

else

s[i] = (float)dx / dy; // General slope

}

}

// Method to draw the polygon edges

void drawEdges() {

int i; // Reuse 'i' for each loop

for (i = 0; i < n; i++) {

line(a[i][0], a[i][1], a[i + 1][0], a[i + 1][1]);

}

}

// Method to find intersections and fill the polygon using scanline algorithm

void fillPolygon() {

int y, i, j, k, temp; // Declare all loop variables outside the loops

for (y = 0; y < 480; y++) {

k = 0;

// Find intersections of the edges with the scanline

for (i = 0; i < n; i++) {

if (((a[i][1] <= y) && (a[i + 1][1] > y)) ||

((a[i][1] > y) && (a[i + 1][1] <= y))) {

xi[k] = (int)(a[i][0] + s[i] \* (y - a[i][1]));

k++;

}

}

// Sort the intersections

for (j = 0; j < k - 1; j++) {

for (i = 0; i < k - 1; i++) {

if (xi[i] > xi[i + 1]) {

temp = xi[i];

xi[i] = xi[i + 1];

xi[i + 1] = temp;

}

}

}

// Directly fill between pairs of intersections

setcolor(3); // Set color for filling

for (i = 0; i < k; i += 2) {

line(xi[i], y, xi[i + 1] + 1, y);

delay(100); // Add a delay of 10 milliseconds to visualize filling process

}

}

}

// Method to display the polygon

void draw() {

int gd = DETECT, gm;

initgraph(&gd, &gm, "C:\\Turboc3\\bgi");

// Draw the polygon edges

drawEdges();

// Calculate the slopes for each edge

calculateSlopes();

// Fill the polygon using the scanline algorithm

fillPolygon();

getch(); // Wait for user input before closing

closegraph();

}

};

int main() {

int n;

cout << "Enter the number of edges (vertices) of the polygon: ";

cin >> n;

// Create a Polygon object and draw it

Polygon polygon(n);

polygon.draw();

return 0;

}

**OUTPUT:**



