#include <graphics.h>

#include <iostream>

using namespace std;

class MidpointCircle {

public:

// Function to draw a single circle using the Midpoint Circle Algorithm

void midPointCircleDraw(int x\_centre, int y\_centre, int r, int color) {

int x = r, y = 0;

int P = 1 - r; // Initial decision parameter

// Set the drawing color

setcolor(color);

// Plot the initial points

plotCirclePoints(x\_centre, y\_centre, x, y);

while (x > y) {

y++;

// Decision parameter check

if (P <= 0) {

P = P + 2 \* y + 1; // Move east

} else {

x--;

P = P + 2 \* y - 2 \* x + 1; // Move southeast

}

// Plot the circle points

if (x < y) break; // Exit when x and y cross

plotCirclePoints(x\_centre, y\_centre, x, y);

if (x != y) {

plotCirclePoints(x\_centre, y\_centre, y, x); // Symmetry

}

delay(10); // Slow down the drawing to visualize it

}

}

private:

// Helper function to plot points on the circle

void plotCirclePoints(int x\_centre, int y\_centre, int x, int y) {

// Plot all 8 symmetric points of the circle

putpixel(x\_centre + x, y\_centre + y, WHITE);

putpixel(x\_centre - x, y\_centre + y, WHITE);

putpixel(x\_centre + x, y\_centre - y, WHITE);

putpixel(x\_centre - x, y\_centre - y, WHITE);

putpixel(x\_centre + y, y\_centre + x, WHITE);

putpixel(x\_centre - y, y\_centre + x, WHITE);

putpixel(x\_centre + y, y\_centre - x, WHITE);

putpixel(x\_centre - y, y\_centre - x, WHITE);

}

};

int main() {

int gd = DETECT, gm;

initgraph(&gd, &gm, NULL); // Initialize the graphics mode

int x, y, start\_radius, num\_circles;

cout << "Enter the center coordinates (x, y): ";

cin >> x >> y;

cout << "Enter the starting radius of the first circle: ";

cin >> start\_radius;

cout << "Enter the number of concentric circles to draw: ";

cin >> num\_circles;

MidpointCircle ob;

// Draw concentric circles with increasing radius

for (int i = 0; i < num\_circles; i++) {

ob.midPointCircleDraw(x, y, start\_radius + i \* 20, WHITE); // Increment radius by 20

}

getch(); // Wait for user input to close the graphics window

closegraph(); // Close the graphics window

return 0;

}