**NAME:- SURWADE TRISHARAN RAJESH**

**ROLL NO:- 47**

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

1. **Line drawing algorithm (DDA and Bresenham’s Line Algorithm)**

**A.DDA Line Algorithm**

#include< graphics.h>

#include<stdio.h>

#include<iostream.h>

#include<math.h>

#include<conio.h>

#include<dos.h>

int maxx, maxy;

class Line

{

float x1, x2, y1, y2, dx, dy, length, x, y;

int i, gd, gm;

public:

void getData();

void draw();

};

void Line :: getData()

{

cout<<":\n Enter the x coordinate of first point :";

cin>>x1;

cout<<"\n Enter the y coordinate of first point :";

cin>>y1;

cout<<":\n Enter the x coordinate of second point :";

cin>>x2;

cout<<":\n Enter the y coordinate of second point :";

cin>>y2;

}

void Line :: draw()

{

dx = abs(x2-x1);

dy = abs(y2-y1);

if(dx>=dy)

length = dx;

else

length =dy;

dx = (x2-x1)/length;

dy = (y2-y1)/length;

x = x1+0.5;

y = y1+0.5;

i=1;

while (i<=length)

{

//Rounding

putpixel(320+x, 240-y,15);

x = x + dx;

y = y + dy;

i++;

delay(100);

}

}

int main() {

int maxx, maxy;

int gd = DETECT, gm;

initgraph(&gd, &gm, "C:\\Turboc3\\BGI"); // Specify the correct path to BGI

maxx = getmaxx(); // Get the max x coordinate of the screen

maxy = getmaxy(); // Get the max y coordinate of the screen

// Draw the axis lines (x and y axes)

line(320, 0, 320, maxy); // y-axis

line(0, 240, maxx, 240); // x-axis

cout << "\nmaxx = " << maxx << " maxy = " << maxy << endl;

// Create an object of the Line class

Line l;

l.getData(); // Get the coordinates of the points

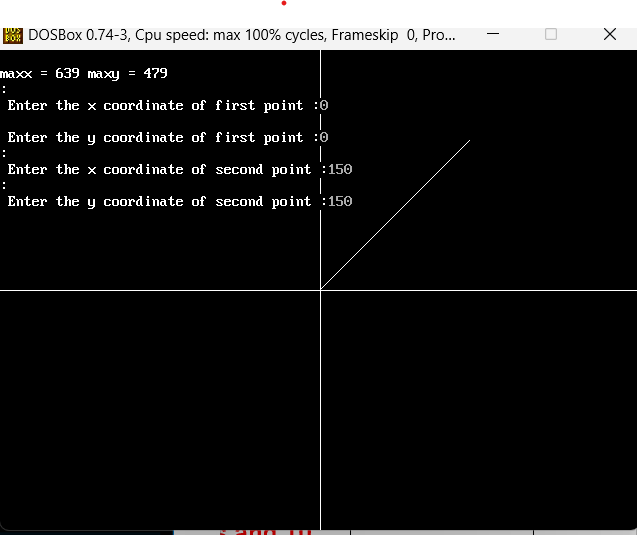
l.draw(); // Draw the line using the DDA algorithm

getch(); // Wait for user input

closegraph(); // Close the graphics mode before exiting

return 0;

}

**OUTPUT:** 

**B. Bresenham’s Line Algorithm**

#include <graphics.h>

#include <stdio.h>

#include <iostream.h>

#include <math.h>

#include <conio.h>

#include <dos.h>

int maxx, maxy;

class Line {

private:

int x1, y1, x2, y2;

int dx, dy, p, x, y;

int gd, gm;

public:

/\* Line() : gd(DETECT), gm(DETECT) {

initgraph(&gd, &gm, "C:\\TurboC3\\BGI"); // Update path to the BGI folder

}\*/

// Function to get the coordinates of two points from the user

void getData() {

cout << "\nEnter the x coordinate of first point: ";

cin >> x1;

cout << "\nEnter the y coordinate of first point: ";

cin >> y1;

cout << "\nEnter the x coordinate of second point: ";

cin >> x2;

cout << "\nEnter the y coordinate of second point: ";

cin >> y2;

}

// Function to implement Bresenham's Line Drawing Algorithm

void draw() {

dx = x2 - x1;

dy = y2 - y1;

// Bresenham's algorithm setup

p = 2 \* dy - dx;

x = x1;

y = y1;

// Draw the starting point

putpixel(320 + x, 240 - y, 15); // Rounding offset applied to center (320, 240)

// Bresenham's Line Drawing Algorithm

while (x < x2) {

if (p < 0) {

p = p + 2 \* dy; // Increment p and only update x

} else {

p = p + 2 \* (dy - dx); // Increment p and both x and y

y = y + 1; // Move to the next row (upwards)

}

x = x + 1; // Move to the next column

putpixel(320 + x, 240 - y, 15); // Plot the pixel with the rounding offset applied

delay(100);

}

}

// Destructor to close the graphics window

~Line() {

closegraph(); // Close the graphics mode before exiting

}

};

int main() {

int gd = DETECT, gm;

initgraph(&gd, &gm, "C:\\TurboC3\\BGI"); // Initialize graphics mode, update path if needed

maxx = getmaxx(); // Get the maximum x-coordinate of the screen

maxy = getmaxy(); // Get the maximum y-coordinate of the screen

// Draw the axis lines (x and y axes)

line(320, 0, 320, maxy); // y-axis

line(0, 240, maxx, 240); // x-axis

cout << "\nmaxx = " << maxx << " maxy = " << maxy << endl;

// Create an object of the Line class

Line l;

l.getData(); // Get the coordinates of the points

l.draw(); // Draw the line using Bresenham's algorithm

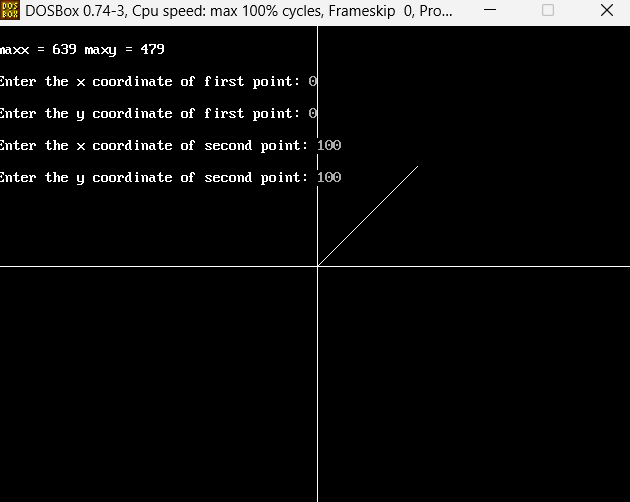
getch(); // Wait for a key press

closegraph(); // Close the graphics mode before exiting

return 0;

}

**OUTPUT:**

****