**Lab Report-01**

**Title: Scan Conversion of Point, Line and Circle**

*Course Title: Computer Graphics Laboratory*

*Course code: CSE-304*

*3rd Year 1st Semester Examination 2022*

**Date of Submission**: 28-05-2023

****

**Submitted to-**

***Dr. Mohammad Shorif Uddin***

Professor

***Dr. Morium Akter***

Associate Professor

*Department of Computer Science and Engineering*

*Jahangirnagar University*

*Savar, Dhaka-1342*

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl** | Class Roll | Exam Roll | Name |
| 01 | 358 | 202170 | Jannati Tajrimin Mitu |

**Scan Conversion Of a Point:**

#include <stdio.h>

void scanConvertPoint(int x, int y, char canvas[][20], int width, int height) {

canvas[y][x] = '\*';

}

void drawCanvas(char canvas[][20], int width, int height) {

for (int i = 0; i < height; i++) {

for (int j = 0; j < width; j++) {

printf("%c ", canvas[i][j]);

}

printf("\n");

}

}

int main() {

// Define the size of the canvas

const int width = 20;

const int height = 10;

// Create a 2D canvas filled with spaces

char canvas[height][width];

for (int i = 0; i < height; i++) {

for (int j = 0; j < width; j++) {

canvas[i][j] = ' ';

}

}

// Define the point to scan convert

int x = 7;

int y = 4;

// Scan convert the point

scanConvertPoint(x, y, canvas, width, height);

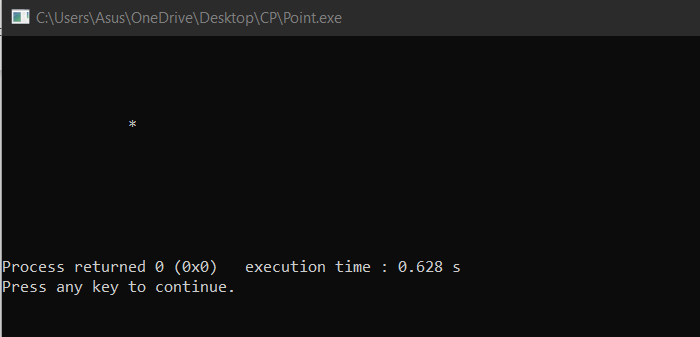
// Draw the canvas

drawCanvas(canvas, width, height);

return 0;

}

**Output:**



**Scan Conversion of a line using DDA Algorithm:**

#include<graphics.h>

#include<conio.h>

#include<stdio.h>

int main()

{

int gd = DETECT ,gm, i;

float x, y,dx,dy,steps;

int x0, x1, y0, y1;

initgraph(&gd, &gm, "C:\\TC\\BGI");

setbkcolor(WHITE);

x0 = 100 , y0 = 200, x1 = 500, y1 = 300;

dx = (float)(x1 - x0);

dy = (float)(y1 - y0);

if(dx>=dy) {

steps = dx;

}

else {

steps = dy;

}

dx = dx/steps;

dy = dy/steps;

x = x0; y = y0;

i = 1;

while(i<= steps) {

putpixel(x, y, RED);

x += dx;

y += dy; i=i+1;

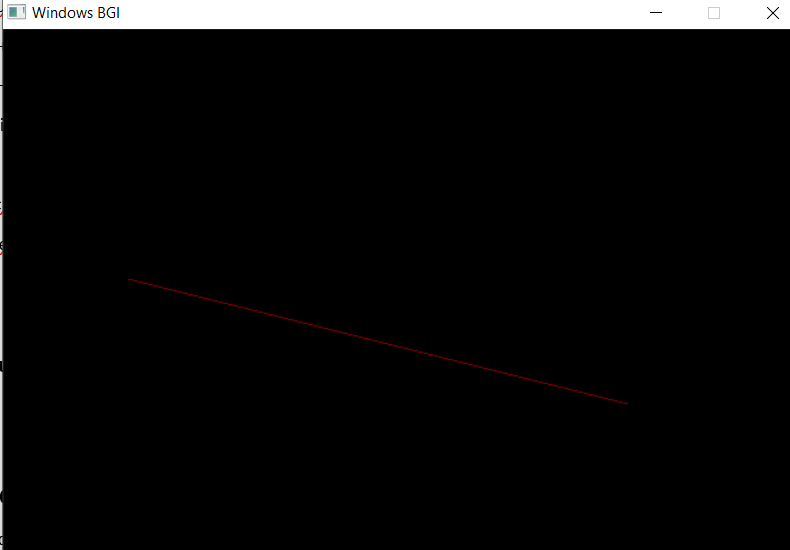
}

getch();

closegraph();

}

**Output:**



**Scan Conversion of a line using Bresenham’s Algorithm:**

#include<stdio.h>

#include<graphics.h>

void drawline(int x0, int y0, int x1, int y1)

{

int dx, dy, p, x, y;

dx=x1-x0;

dy=y1-y0;

x=x0;

y=y0;

p=2\*dy-dx;

while(x<x1)

{

if(p>=0)

{

putpixel(x,y,7);

y=y+1;

p=p+2\*dy-2\*dx;

}

else

{

putpixel(x,y,7);

p=p+2\*dy;}

x=x+1;

}

}

int main()

{

int gdriver=DETECT, gmode, error, x0, y0, x1, y1;

initgraph(&gdriver, &gmode, "c:\\turboc3\\bgi");

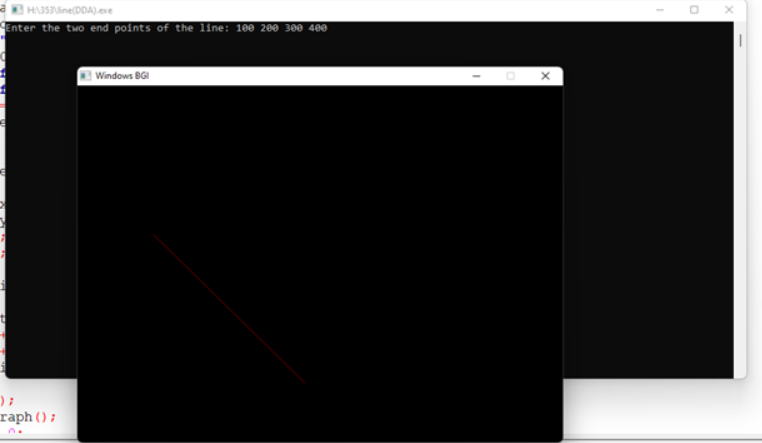
printf("Enter the end points of the line: ");

scanf("%d%d%d%d", &x0, &y0,&x1,&y1);

drawline(x0, y0, x1, y1);

return 0;

}

**Output:**

**Scan Conversion of a Circle using Bresenham’s Algorithm:**

#include <graphics.h>

#include <stdlib.h>

#include <stdio.h>

#include <conio.h>

#include <math.h>

void EightWaySymmetricPlot(int xc,int yc,int x,int y)

{

putpixel(x+xc,y+yc,RED);

putpixel(x+xc,-y+yc,YELLOW);

putpixel(-x+xc,-y+yc,GREEN);

putpixel(-x+xc,y+yc,YELLOW);

putpixel(y+xc,x+yc,12);

putpixel(y+xc,-x+yc,14);

putpixel(-y+xc,-x+yc,15);

putpixel(-y+xc,x+yc,6);

}

void BresenhamCircle(int xc,int yc,int r)

{

int x=0,y=r,d=3-(2\*r);

EightWaySymmetricPlot(xc,yc,x,y);

while(x<=y)

{

if(d<=0)

{

d=d+(4\*x)+6;

}

else

{

d=d+(4\*x)-(4\*y)+10;

y=y-1;

}

x=x+1;

EightWaySymmetricPlot(xc,yc,x,y);

}

}

int main(void)

{

/\* request auto detection \*/

int xc,yc,r,gdriver = DETECT, gmode, errorcode;

/\* initialize graphics and local variables \*/

initgraph(&gdriver, &gmode, "C:\\TURBOC3\\BGI");

/\* read result of initialization \*/

errorcode = graphresult();

if (errorcode != grOk) /\* an error occurred \*/

{

printf("Graphics error: %s\n", grapherrormsg(errorcode));

printf("Press any key to halt:");

getch();

exit(1); /\* terminate with an error code \*/

}

printf("Enter the values of xc and yc :");

scanf("%d%d",&xc,&yc);

printf("Enter the value of radius :");

scanf("%d",&r);

BresenhamCircle(xc,yc,r);

getch();

closegraph();

return 0;

}

**Output:**

