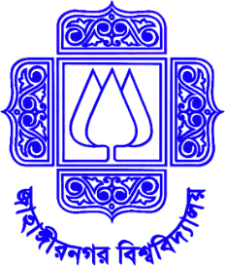
**Lab Report 01**

*Course title: Computer Graphics Laboratory*

*Course code: CSE-304*

*3rd Year 1st Semester 2022*

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



###### **Submitted to-**

###### Dr. Mohammad Shorif Uddin

###### Professor

*Department of Computer Science and Engineering*

*Jahangirnagar University*

*Savar, Dhaka-1342*

*And*

*Dr. Morium Akter*

###### Associate Professor

*Department of Computer Science and Engineering*

*Jahangirnagar University*

*Savar, Dhaka-1342*

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl** | Class Roll | Exam Roll | Name |
| 01 | 349 | 202161 | Subarna Saha |

**Experiment No.: 1**

**Experiment Name:** Scan conversion of a point.

**Source code in C:**

#include<stdio.h>

#include<math.h>

#include <graphics.h>

int main()

{

int x, y;

float x1, y1;

printf("Enter the co-ordinate of the point: ");

scanf("%f %f", &x1, &y1);

x=floor(x1);

y=floor(y1);

int color = WHITE;

int gd = DETECT, gm;

initgraph(&gd, &gm, (char\*)"");

putpixel(x, y, color);

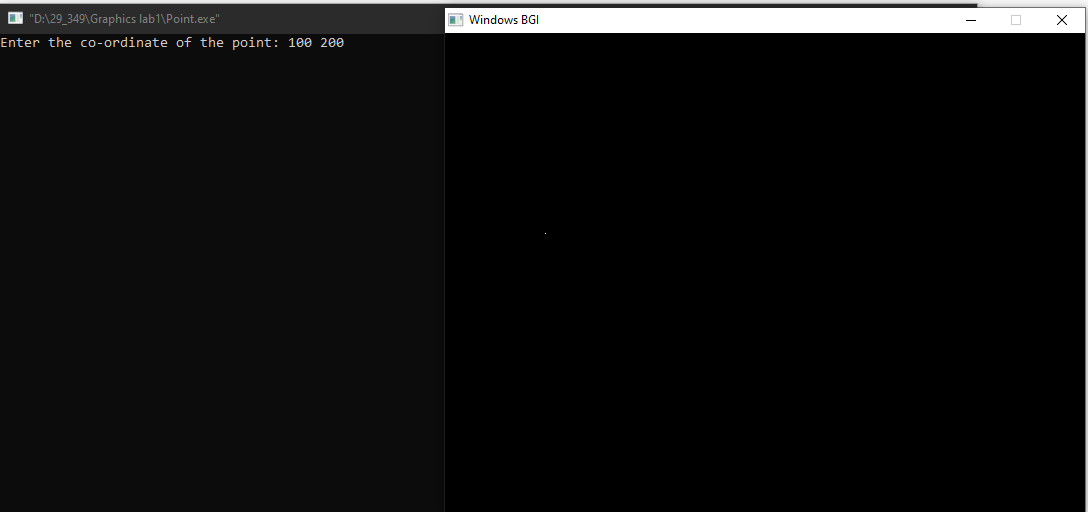
delay(5000);

closegraph();

return 0;

}

**Input and Output:**

****

**Experiment No.: 2**

**Experiment Name:** Scan conversion of a line using DDA algorithm.

**Source code in C:**

#include<stdio.h>

#include<math.h>

#include <graphics.h>

void drawLineDDA(int x1, int y1, int x2, int y2)

{

int dx = x2 - x1;

int dy = y2 - y1;

int steps;

if(abs(dx) > abs(dy))

{

steps=abs(dx);

}

else

{

steps=abs(dy);

}

float xIncrement = dx/(float)steps;

float yIncrement = dy/(float)steps;

float x = x1;

float y = y1;

for (int i=0; i<=steps; i++)

{

putpixel(round(x), round(y), WHITE);

x+=xIncrement;

y+=yIncrement;

}

}

int main()

{

float x,y;

int x1,y1,x2,y2;

printf("Enter the co-ordinate of the first point of the line: ");

scanf("%f %f", &x, &y);

x1=x, y1=y;

printf("Enter the co-ordinate of the second point of the line: ");

scanf("%f %f", &x, &y);

x2=x;y2=y;

int gd = DETECT, gm;

initgraph(&gd, &gm, (char\*)"");

drawLineDDA(x1, y1, x2, y2);

delay(10000);

closegraph();

return 0;

}

**Input and Output:**

****

**Experiment No.: 3**

**Experiment Name:** Scan conversion of a line using Bresenham’s algorithm.

**Source code in C:**

#include<stdio.h>

#include<math.h>

#include <graphics.h>

void drawLineBresenham(int x1, int y1, int x2, int y2)

{

int dx = abs(x2 - x1);

int dy = abs(y2 - y1);

int x, y;

int p;

int xStep, yStep;

if(x1<x2)

{

xStep=1;

}

else

{

xStep=-1;

}

if(y1<y2)

{

yStep=1;

}

else

{

yStep=-1;

}

if(dx >= dy)

{

p = 2\*dy-dx;

y = y1;

for (x=x1; x!=x2; x+=xStep)

{

putpixel(x, y, WHITE);

if (p >= 0)

{

y += yStep;

p -= 2 \* dx;

}

p+=2\*dy;

}

}

else

{

p = 2\*dx-dy;

x = x1;

for (y=y1; y!=y2; y+=yStep)

{

putpixel(x, y, WHITE);

if (p >= 0)

{

x+=xStep;

p-=2 \* dy;

}

p+=2\*dx;

}

}

}

int main()

{

int x1, y1, x2, y2;

float x, y;

printf("Enter the coordinates of the first point: ");

scanf("%f %f", &x, &y);

x1=x, y1=y;

printf("Enter the coordinates of the second point: ");

scanf("%f %f", &x, &y);

x2=x, y2=y;

int gd = DETECT, gm;

initgraph(&gd, &gm, (char\*)"");

drawLineBresenham(x1, y1, x2, y2);

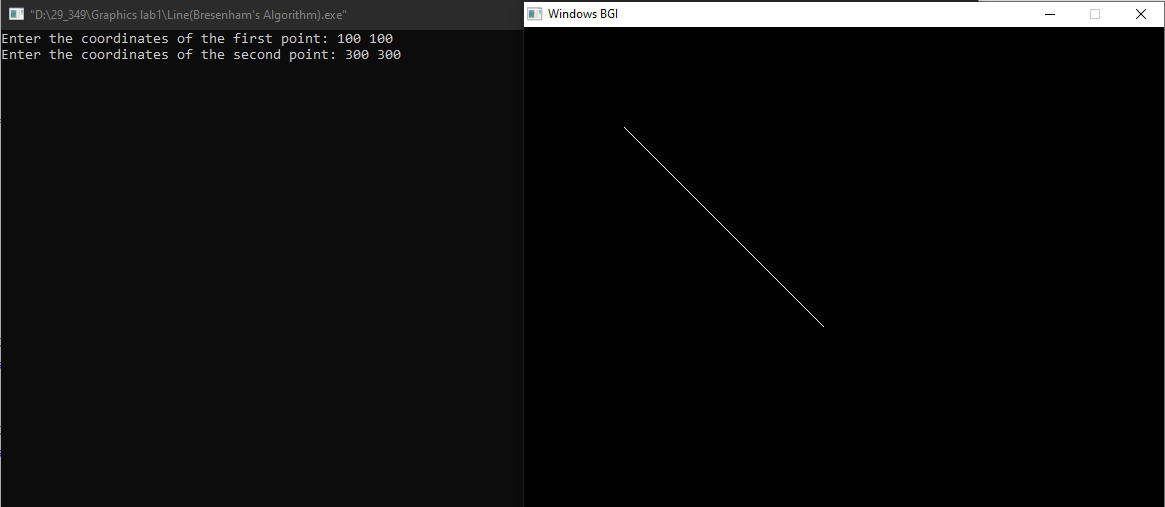
delay(10000);

closegraph();

return 0;

}

**Input and Output:**



**Experiment No.: 4**

**Experiment Name:** Scan conversion of a circle using Bresenham’s algorithm.

**Source code in C:**

#include<stdio.h>

#include<math.h>

#include <graphics.h>

void drawCircleBresenham(int xc, int yc, int radius)

{

int x = 0;

int y = radius;

int p = 3- 2\*radius;

while (x<=y)

{

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

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

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

putpixel(xc - x, yc - y, WHITE);

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

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

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

putpixel(xc - y, yc - x, WHITE);

if (p<0)

{

p += 4\*x+6;

}

else

{

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

y--;

}

x++;

}

}

int main()

{

int xc, yc, radius;

printf("Enter the center coordinates: ");

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

printf("Enter the radius: ");

scanf("%d", &radius);

int gd = DETECT, gm;

initgraph(&gd, &gm, (char\*)"");

drawCircleBresenham(xc, yc, radius);

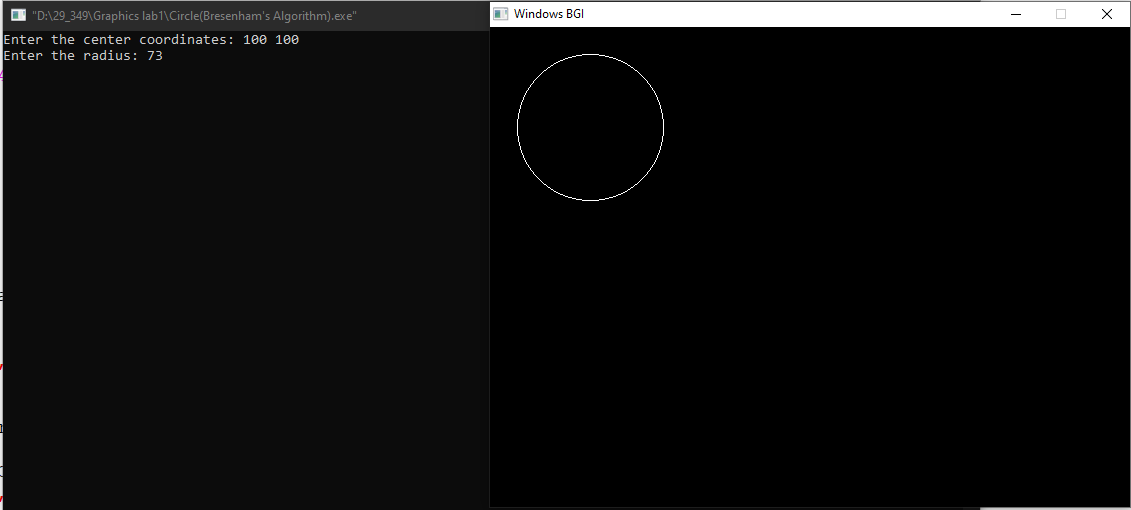
delay(5000);

closegraph();

return 0;

}

**Input and Output:**

****