**Assignment: Lab Report 02**

*Course title: Computer Graphics Lab*

*Course code: CSE-304*

*3rd Year 1st Semester Examination 2022*

**Date of Submission**: *4th June 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 | 370 | 202182 | Rubayed All Islam |
|  |  |  |  |

**Scan Conversion of a Circle( Mid Point Algorithm):**

*Code:*

#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,WHITE);

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

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

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

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

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

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

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

}

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

{

int x=0,y=r,d=1-r;

EightWaySymmetricPlot(xc,yc,x,y);

while(x<=y)

{

if(d<=0)

{

d=d+(2\*x)+3;

}

else

{

d=d+2\*(x-y)+5;

y=y-1;

}

x=x+1;

EightWaySymmetricPlot(xc,yc,x,y);

}

}

int main(void)

{

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

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

errorcode = graphresult();

if (errorcode != grOk)

{

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

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

getch();

exit(1);

}

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

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

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

scanf("%d",&r);

MidPointCircle(xc,yc,r);

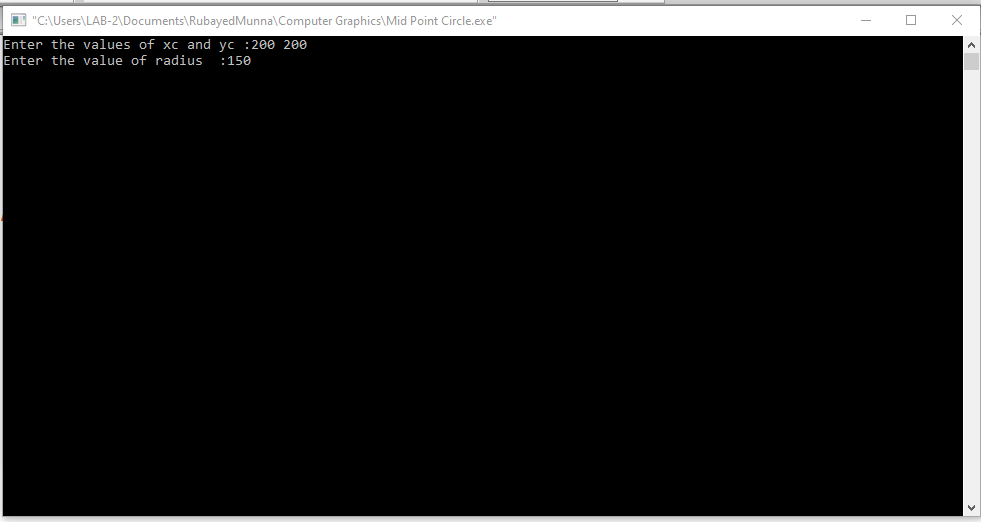
getch();

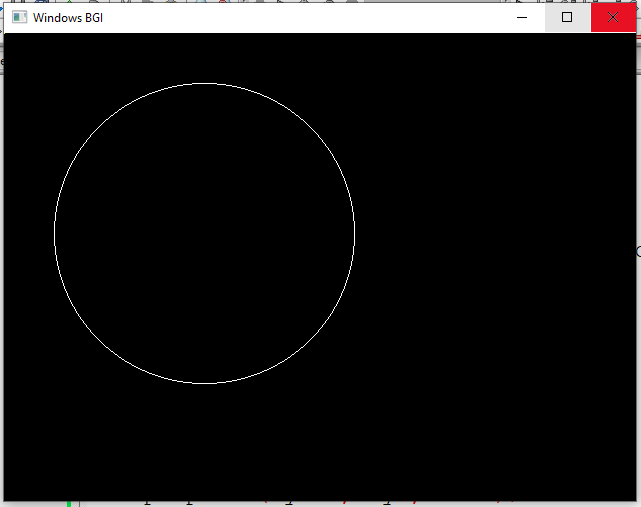
closegraph();

return 0;

}

*Output:*

**

**

**Scan Conversion of a Ellipse:**

*Code:*

#include <graphics.h>

#include <bits/stdc++.h>

using namespace std;

void EllipseScanConversion(int xc, int yc, int a, int b) {

int x = 0;

int y = b;

int decision = (b\*b) - (a\*a) \* b + (a\*a) / 4;

while ((2\*b\*b) \* x < (2\*a\*a) \* 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);

if (decision <= 0) {

x++;

decision += (2\*b\*b) \* x + (b\*b);

} else {

x++;

y--;

decision += (2\*b\*b) \* x - (2\*a\*a) \* y + (b\*b);

}

}

decision = (b\*b) \* (x + 0.5) \* (x + 0.5) + (a\*a) \* (y - 1) \* (y - 1) - (a\*a) \* (b\*b);

while (y >= 0) {

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

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

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

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

if (decision > 0) {

y--;

decision += (a\*a) - (2\*a\*a) \* y;

} else {

y--;

x++;

decision += (2\*b\*b) \* x - (2\*a\*a) \* y + (a\*a);

}

}

}

int main() {

int gdriver = DETECT, gmode, errorcode;

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

errorcode = graphresult();

if (errorcode != grOk)

{

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

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

getch();

exit(1);

}

int xc, yc, a, b;

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

cin >> xc >> yc;

cout << "Enter semi-major axis (a): ";

cin >> a;

cout << "Enter semi-minor axis (b): ";

cin >> b;

EllipseScanConversion(xc, yc, a, b);

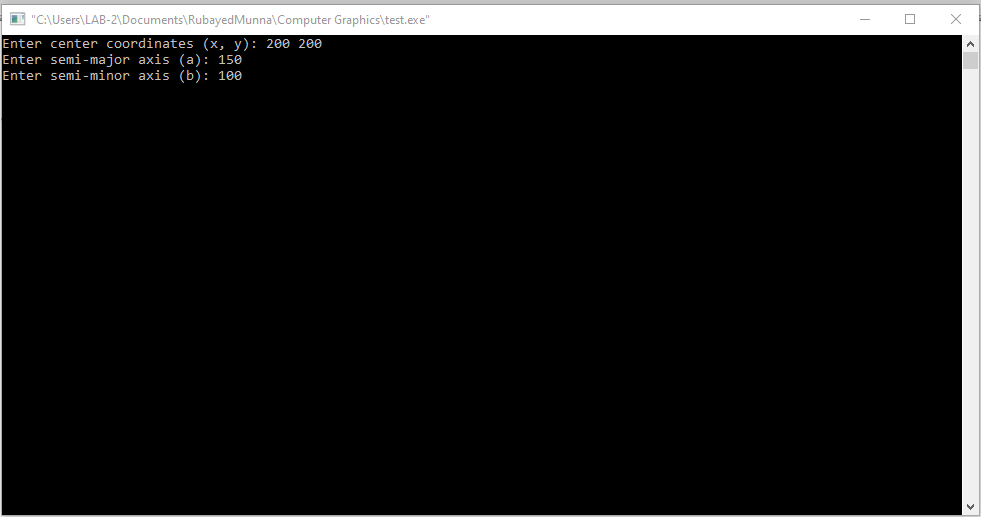
getch();

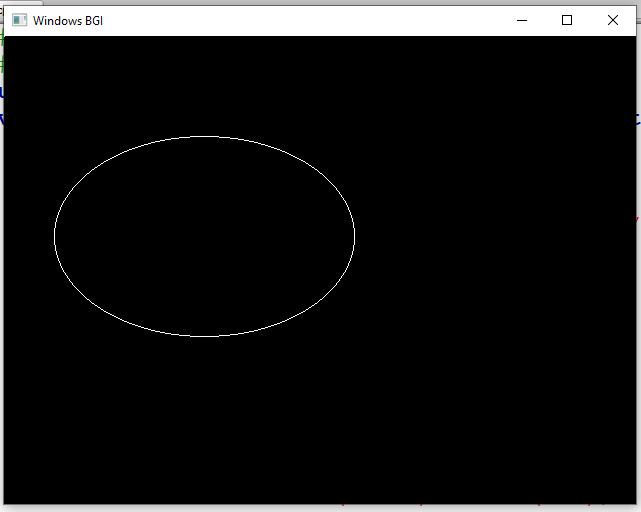
closegraph();

return 0;

}

*Output:*

**

**