**Write a program to rotate an object by 90 degree in clockwise direction**

#include<graphics.h>

#include<stdio.h>

#include<conio.h>

#include<math.h>

int main()

{

int gd=DETECT,gm;

int pivot\_x,pivot\_y,x,y;

double degree,radian;

int rotated\_point\_x,rotated\_point\_y;

initgraph(&gd,&gm,"C://TURBOC3//BGI");

cleardevice();

printf("\t\t\*\*\*\*\*\*\*\*\*\*\* ROTATION \*\*\*\*\*\*\*\*\*\*\* \n");

printf("\n Enter an initial coordinates of the line = ");

scanf("%d %d",&pivot\_x,&pivot\_y);

printf("\n Enter a final coordinates of the line = ");

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

line(pivot\_x,pivot\_y,x,y);

printf("\n\n Now, Enter a degree = ");

scanf("%lf",&degree);

radian=degree\*0.01745;

rotated\_point\_x=(int)(pivot\_x +((x-pivot\_x)\*cos(radian)-(y-pivot\_y)\*sin(radian)));

rotated\_point\_y=(int)(pivot\_y +((x-pivot\_x)\*sin(radian)+(y-pivot\_y)\*cos(radian)));

setcolor(RED);

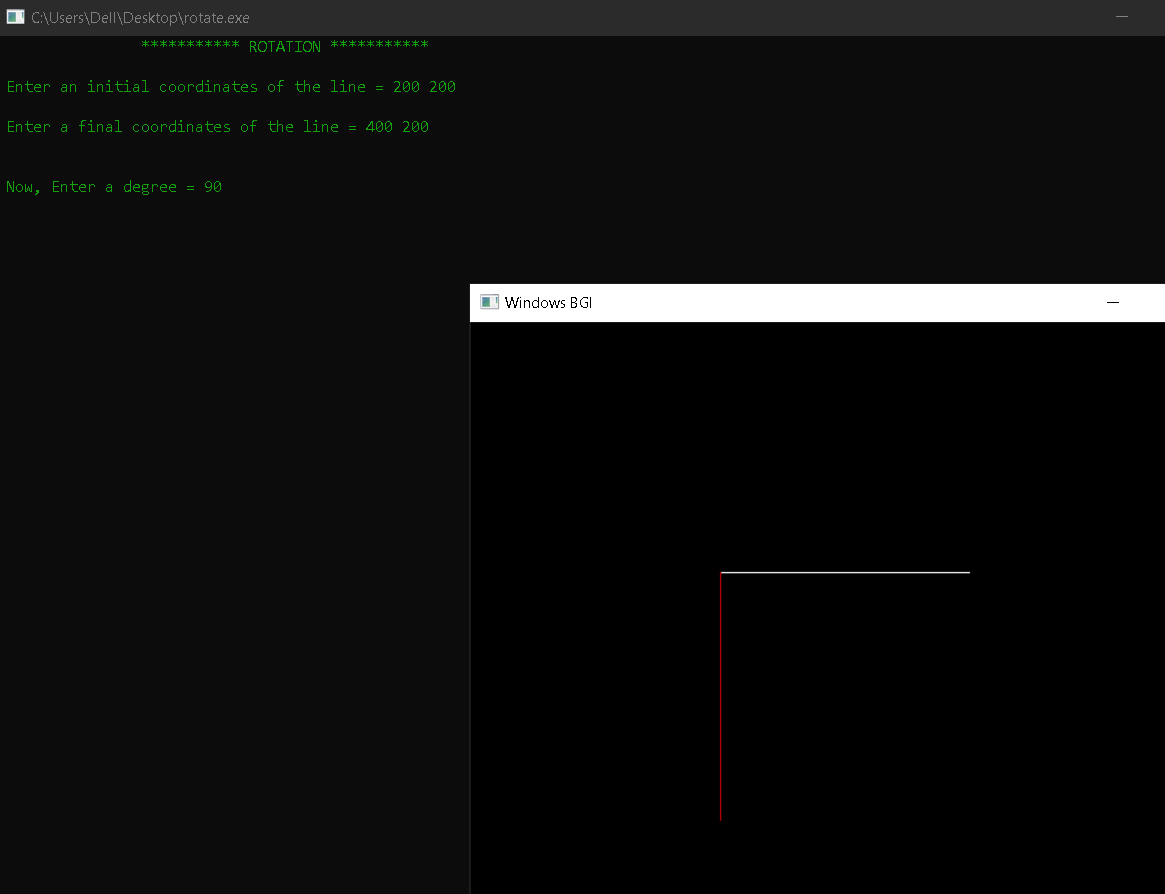
line(pivot\_x,pivot\_y,rotated\_point\_x,rotated\_point\_y);

getch();

closegraph();

return 0;

}



**Write a program to draw a line using Bresenham’s line generation 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,NULL);

printf("Enter co-ordinates of first point: ");

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

printf("Enter co-ordinates of second point: ");

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

drawline(x0, y0, x1, y1);

getch();

closegraph();

return 0;

}

Text

Description automatically generated

A picture containing shape

Description automatically generated

**Write a program to implement Boundary-fill algorithm.**

#include <graphics.h>

void boundaryFill4(int x, int y, int fill\_color,int boundary\_color)

{

if(getpixel(x, y) != boundary\_color &&

getpixel(x, y) != fill\_color)

{

putpixel(x, y, fill\_color);

boundaryFill4(x + 1, y, fill\_color, boundary\_color);

boundaryFill4(x, y + 1, fill\_color, boundary\_color);

boundaryFill4(x - 1, y, fill\_color, boundary\_color);

boundaryFill4(x, y - 1, fill\_color, boundary\_color);

}

}

int main()

{

int gd = DETECT, gm;

initgraph(&gd, &gm, "");

int x = 250, y = 200, radius = 50;

circle(x, y, radius);

boundaryFill4(x, y, 6, 15);

delay(10000);

getch();

closegraph();

return 0;

}

Icon

Description automatically generated with low confidenceText

Description automatically generated