**PROGRAM 2**

**Write a program to draw a line using DDA line drawing algorithm.**

#include<stdio.h>

#include<stdlib.h>

#include<graphics.h>

#define ROUND(a) ((int)(a+0.5))

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

{

float xsteps, ysteps, x=x1, y=y1;

int dx = x2-x1;

int dy = y2-y1;

int steps,k=1;

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

steps=abs(dx);

else steps=abs(dy);

xsteps= dx/(float)steps;

ysteps= dy/(float)steps;

putpixel(ROUND(x),ROUND(y),15);

while(k<=steps)

{

x+=xsteps;

y+=ysteps;

putpixel(ROUND(x), ROUND(y),15);

k++;

}

}

int main()

{

int x1, x2, y1, y2;

int gdriver = DETECT, gmode, errorcode;

initgraph(&gdriver, &gmode, "..\\");

errorcode = graphresult();

if (errorcode != grOk)

{

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

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

getch();

exit(1);

}

printf("Enter start point\n");

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

printf("Enter end point\n");

scanf("%d %d", &x2, &y2);

ddaline(x1, y1, x2, y2);

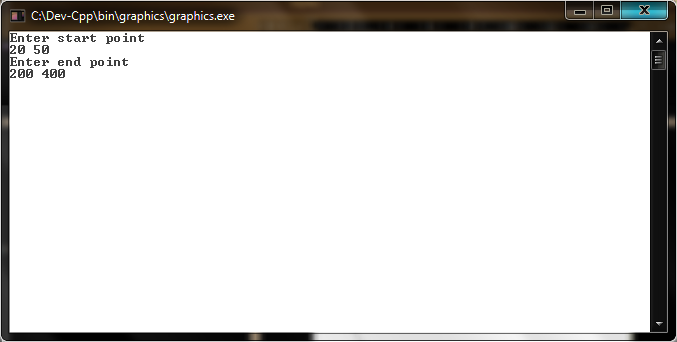
getch();

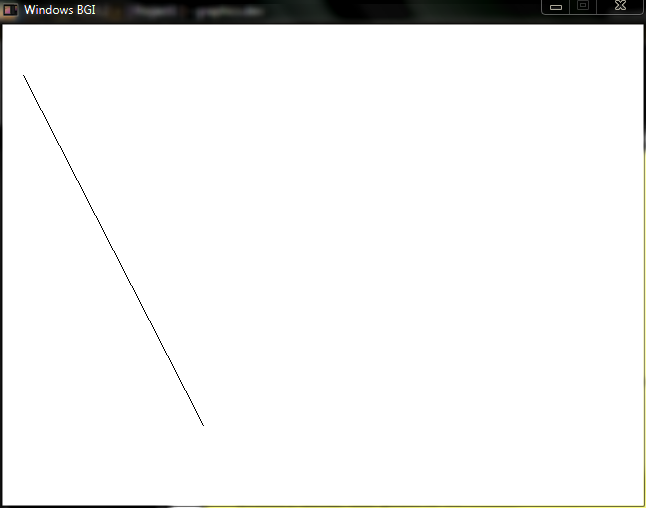
closegraph();

return 0;

}

**OUTPUT 2**





**PROGRAM 3**

**Write a program to draw a line using Bresenham line drawing algorithm (Mid-point line drawing algorithm).**

#include<stdio.h>

#include<stdlib.h>

#include<graphics.h>

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

{

int twody = 2\*abs(y2-y1);

int dx = abs(x2-x1);

int p = twody-dx;

int x, y, xend, ystep=1;

int twodydx= twody - 2\*dx;

if(x1>x2)

{

x=x2;

y=y2;

xend=x1;

if(y2>y1) ystep=-1;

}

else

{

x=x1;

y=y1;

xend=x2;

if(y1>y2) ystep=-1;

}

putpixel(x,y,15);

while(x<xend)

{

x++;

if(p<0)

{

p+=twody;

}

else

{

y+=ystep;

p+=twodydx;

}

putpixel(x,y,15);

}

}

int main()

{

int x1, x2, y1, y2;

int gdriver = DETECT, gmode, errorcode;

initgraph(&gdriver, &gmode, "..\\");

errorcode = graphresult();

if (errorcode != grOk)

{

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

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

getch();

exit(1);

}

printf("Enter start point\n");

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

printf("Enter end point\n");

scanf("%d %d", &x2, &y2);

bresline(x1, y1, x2, y2);

getch();

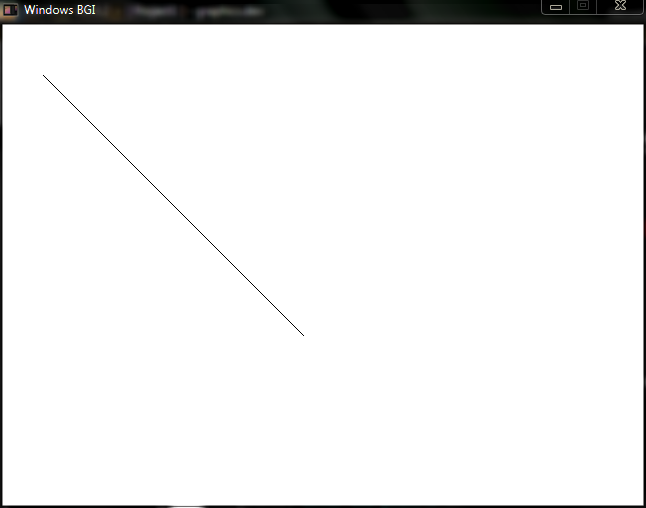
closegraph();

return 0;

}

**OUTPUT 3**





**PROGRAM 4**

**Write a program to draw a circle using mid-point circle algorithm.**

#include<stdio.h>

#include<stdlib.h>

#include<graphics.h>

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

{

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

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

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

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

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

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

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

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

}

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

{

int p = 1 - r;

int x= 0, y= r;

setpixel(xc,yc,x,y);

while(x<y)

{

x++;

if(p<0)

{

p+= 2\*x +1;

}

else

{

y--;

p+= 2\*(x-y) +1;

}

setpixel(xc,yc,x,y);

}

}

int main()

{

int xc, yc, r;

int gdriver = DETECT, gmode, errorcode;

initgraph(&gdriver, &gmode, "..\\");

errorcode = graphresult();

if (errorcode != grOk)

{

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

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

getch();

exit(1);

}

printf("Enter center of circle\n");

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

printf("Enter radius\n");

scanf("%d", &r);

midptcircle(xc,yc,r);

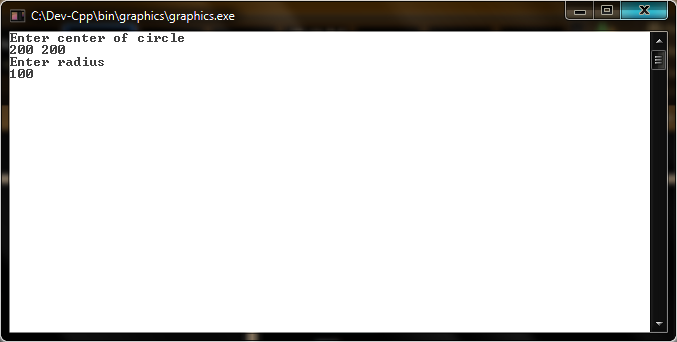
getch();

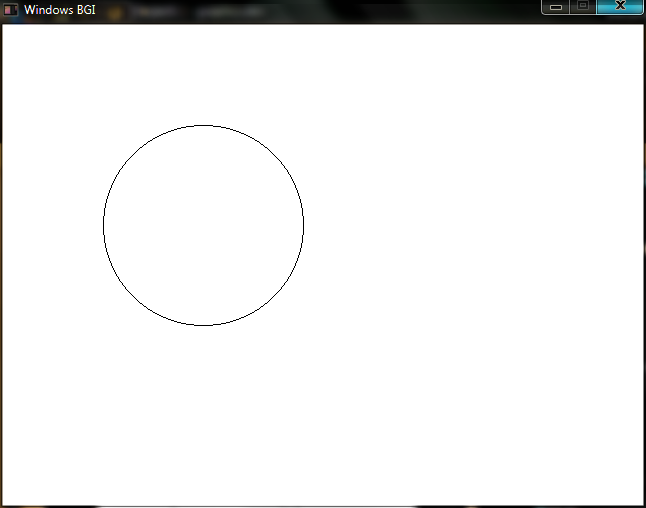
closegraph();

return 0;

}

**OUTPUT 4**





**PROGRAM 5**

**Write a program to draw ellipse using mid-point ellipse algorithm.**

#include<stdio.h>

#include<stdlib.h>

#include<graphics.h>

#define ROUND(a) ((int)(a+0.5))

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

{

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

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

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

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

}

void midptellipse(int xc, int yc, int rx, int ry)

{

int rx2 = rx\*rx, ry2 = ry\*ry;

int tworx2 = 2\*rx2, twory2=2\*ry2;

int x=0, y=ry;

int p= ROUND(ry2 + 0.25\*rx2 - rx2\*ry);

int px= 0;

int py = tworx2\*ry;

setpixel(xc,yc,x,y);

while(px < py)

{

x++;

px += twory2;

if(p<0)

p+= px + ry2;

else

{

y--;

py -= tworx2;

p+= px+ry2-py;

}

setpixel(xc,yc,x,y);

}

p = ROUND(ry2\*(x+0.5)\*(x+0.5) + rx2\*(y-1)\*(y-1) - rx2\*ry2);

while (y>0)

{

y--;

py -= tworx2;

if(p>0)

{

p+= rx2-py;

}

else

{

x++;

px+=twory2;

p+=px-py+rx2;

}

setpixel(xc,yc,x,y);

}

}

int main()

{

int xc, yc, rx, ry;

int gdriver = DETECT, gmode, errorcode;

initgraph(&gdriver, &gmode, "..\\");

errorcode = graphresult();

if (errorcode != grOk)

{

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

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

getch();

exit(1);

}

printf("Enter center of ellipse\n");

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

printf("Enter x and y radius\n");

scanf("%d %d", &rx, &ry);

midptellipse(xc,yc,rx,ry);

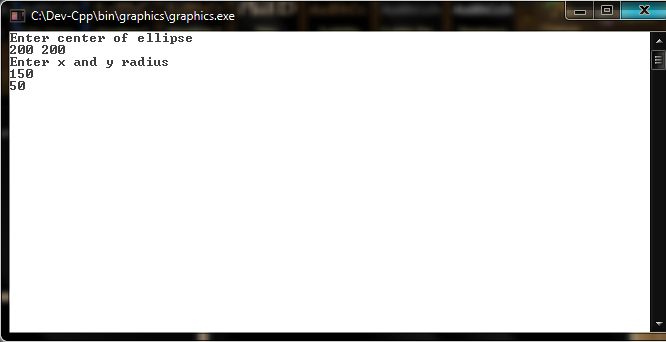
getch();

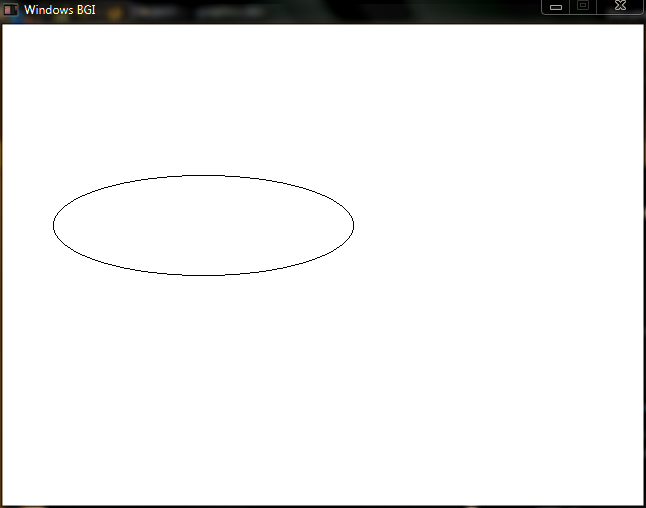
closegraph();

return 0;

}

**OUTPUT 5**





**PROGRAM 6**

**Write a program to draw a thick line.**

#include<stdio.h>

#include<stdlib.h>

#include<graphics.h>

#define ROUND(a) ((int)(a+0.5))

void setpixel(int x, int y, int X, int Y)

{

putpixel(x,y,15);

putpixel(x+X,y+Y,15);

putpixel(x-X,y-Y,15);

}

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

{

float xsteps, ysteps, x=x1, y=y1;

int X=0, Y=0;

int dx = x2-x1;

int dy = y2-y1;

int steps,k=1;

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

{

steps=abs(dx);

Y=1;

}

else

{

steps=abs(dy);

X=1;

}

xsteps= dx/(float)steps;

ysteps= dy/(float)steps;

setpixel(ROUND(x),ROUND(y),X,Y);

while(k<=steps)

{

x+=xsteps;

y+=ysteps;

setpixel(ROUND(x), ROUND(y),X,Y);

k++;

}

}

int main()

{

int x1, x2, y1, y2;

int gdriver = DETECT, gmode, errorcode;

initgraph(&gdriver, &gmode, "..\\");

errorcode = graphresult();

if (errorcode != grOk)

{

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

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

getch();

exit(1);

}

printf("Enter start point\n");

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

printf("Enter end point\n");

scanf("%d %d", &x2, &y2);

thickline(x1, y1, x2, y2);

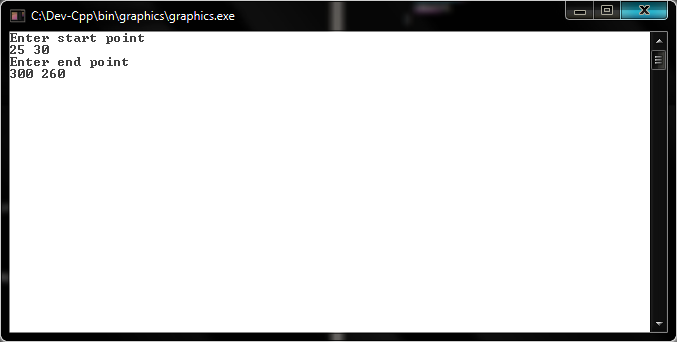
getch();

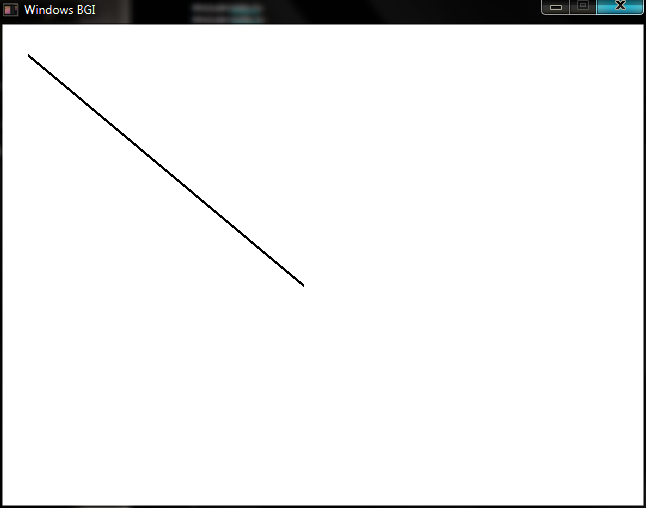
closegraph();

return 0;

}

**OUTPUT 6**





**PROGRAM 7**

**Write a program to draw a solid circle.**

#include<stdio.h>

#include<graphics.h>

void setpixel(int xCenter,int yCenter,int x,int y)

{

for(int i=x;i>=(-x);i--)

{

putpixel(xCenter+i,yCenter+y,15);

putpixel(xCenter+i,yCenter-y,15);

}

for(int i=y;i>=(-y);i--)

{

putpixel(xCenter+i,yCenter+x,15);

putpixel(xCenter+i,yCenter-x,15);

}

}

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

{

int p = 1 - r;

int x= 0, y= r;

setpixel(xc,yc,x,y);

while(x<y)

{

x++;

if(p<0)

{

p+= 2\*x +1;

}

else

{

y--;

p+= 2\*(x-y) +1;

}

setpixel(xc,yc,x,y);

}

}

int main()

{

int xc, yc, r;

int gdriver = DETECT, gmode, errorcode;

initgraph(&gdriver, &gmode, "..\\");

errorcode = graphresult();

if (errorcode != grOk)

{

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

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

getch();

exit(1);

}

printf("Enter center of circle\n");

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

printf("Enter radius\n");

scanf("%d", &r);

midptcircle(xc,yc,r);

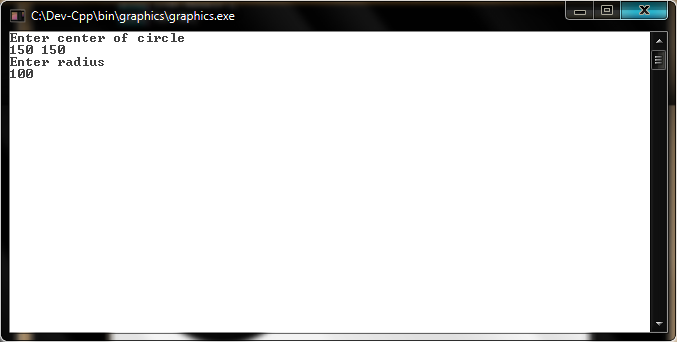
getch();

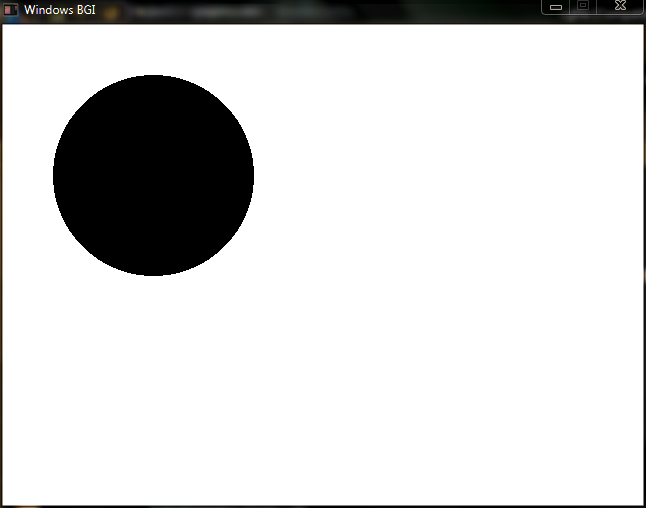
closegraph();

return 0;

}

**OUTPUT 7**





**PROGRAM 8**

**Write a program to draw a solid ellipse.**

#include<stdio.h>

#include<graphics.h>

#define ROUND(a) ((int)(a+0.5))

void setpixel(int xCenter, int yCenter, int x, int y)

{

for(int i=x;i>=(-x);i--)

{

putpixel(xCenter+i, yCenter+y,15);

putpixel(xCenter+i, yCenter-y,15);

}

}

void midptellipse(int xc, int yc, int rx, int ry)

{

int rx2 = rx\*rx, ry2 = ry\*ry;

int tworx2 = 2\*rx2, twory2=2\*ry2;

int x=0, y=ry;

int p= ROUND(ry2 + 0.25\*rx2 - rx2\*ry);

int px= 0;

int py = tworx2\*ry;

setpixel(xc,yc,x,y);

while(px < py)

{

x++;

px += twory2;

if(p<0)

p+= px + ry2;

else

{

y--;

py -= tworx2;

p+= px+ry2-py;

}

setpixel(xc,yc,x,y);

}

p = ROUND(ry2\*(x+0.5)\*(x+0.5) + rx2\*(y-1)\*(y-1) - rx2\*ry2);

while (y>0)

{

y--;

py -= tworx2;

if(p>0)

{

p+= rx2-py;

}

else

{

x++;

px+=twory2;

p+=px-py+rx2;

}

setpixel(xc,yc,x,y);

}

}

int main()

{

int xc, yc, rx, ry;

int gdriver = DETECT, gmode, errorcode;

initgraph(&gdriver, &gmode, "..\\");

errorcode = graphresult();

if (errorcode != grOk)

{

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

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

getch();

exit(1);

}

printf("Enter center of ellipse\n");

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

printf("Enter x and y radius\n");

scanf("%d %d", &rx, &ry);

midptellipse(xc,yc,rx,ry);

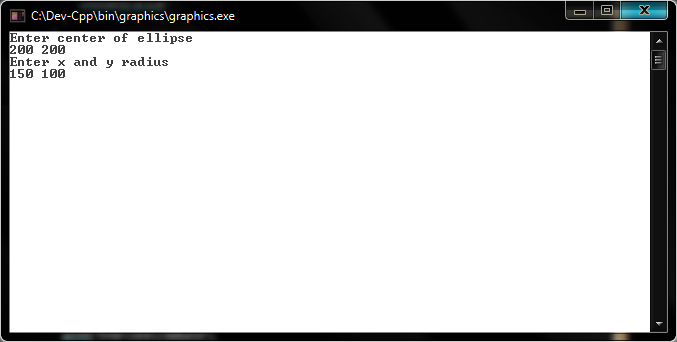
getch();

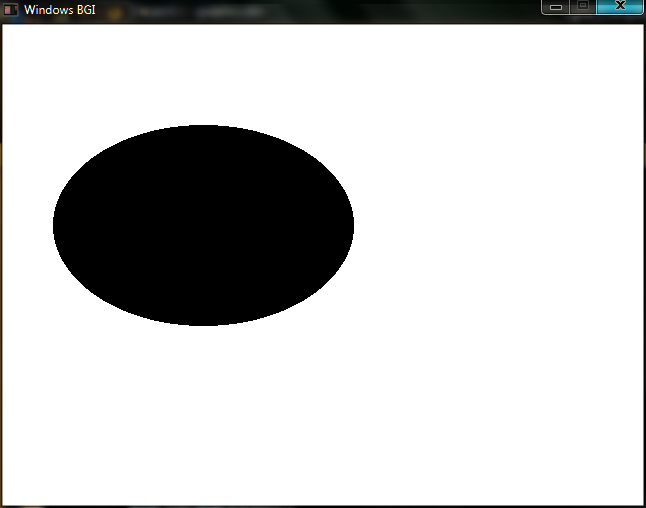
closegraph();

return 0;

}

**OUTPUT 8**





**PROGRAM 9**

**Write a program to draw a polygon.**

#include<stdio.h>

#include<stdlib.h>

#include<graphics.h>

#define ROUND(a) ((int)(a+0.5))

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

{

float xsteps, ysteps, x=x1, y=y1;

int dx = x2-x1;

int dy = y2-y1;

int steps,k=1;

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

steps=abs(dx);

else steps=abs(dy);

xsteps= dx/(float)steps;

ysteps= dy/(float)steps;

putpixel(ROUND(x),ROUND(y),15);

while(k<=steps)

{

x+=xsteps;

y+=ysteps;

putpixel(ROUND(x), ROUND(y),15);

k++;

}

}

void quadilateral(int x[4], int y[4])

{

int i=0;

while(i<4)

{

if(i==3)

{

ddaline(x[i],y[i],x[0],y[0]);

return;

}

ddaline (x[i],y[i],x[i+1],y[i+1]);

i++;

}

}

int main()

{

int x[4],y[4],i=0;

int gdriver = DETECT, gmode, errorcode;

initgraph(&gdriver, &gmode, "..\\");

errorcode = graphresult();

if (errorcode != grOk)

{

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

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

getch();

exit(1);

}

printf("Enter quadilateral points in sequence\n");

while(i<4)

{

scanf("%d %d", &x[i], &y[i]);

i++;

}

quadilateral(x,y);

getch();

closegraph();

return 0;

}

**OUTPUT 9**

