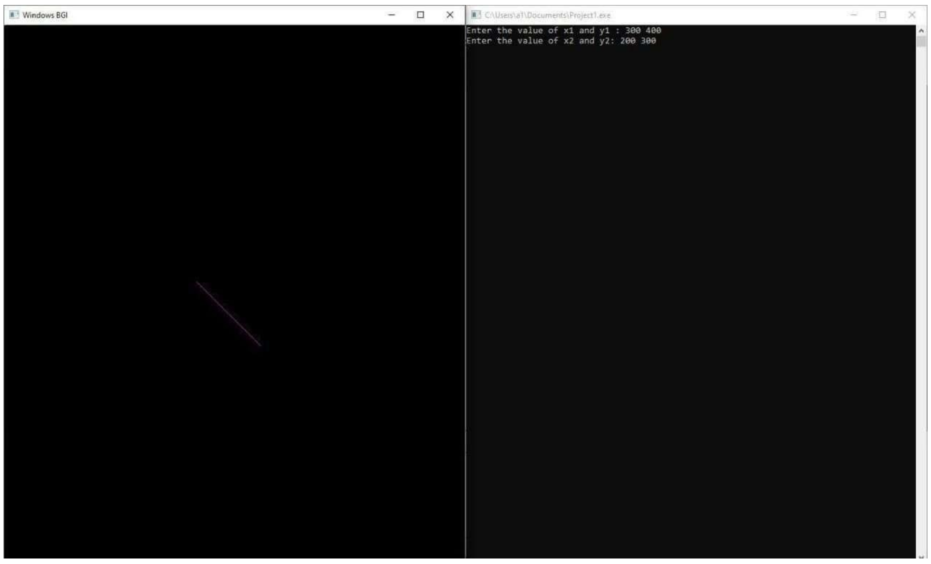
EXPERIMENT NO. 1

DDA Line Drawing Algorithm

Program Code:

```
#include<graphics.h>
#include<stdio.h> #include<math.h>int
main()
{
        int x,y,end,p,x1,x2,y1,y2,dx,dy;
        init window(800,800);
        printf("Enter the value of x1 : ");
        scanf("%d",&x1);
        printf("Enter the value of y1 : ");
        scanf("%d",&y1);
        printf("Enter the value of x2 : ");
        scanf("%d",&x2);
        printf("Enter the value of y2 : ");
        scanf("%d",&y2);
        dx=abs(x1-x2);
        dy=abs(y1-y2);
        p =2*dy-dx;
        if(x1>x2)
        {
                x=x2;y=y2;end=x1;
        }
        else
        {
                x = x1;y=y1;end=x2;
        }
        putpixel(x,y,WHITE);
        while(x<=end)
        {
                If(p<0)
        {
                x++;p=p+2*dy;
        }
        else
        {
                x++: y++;p=p+2*(dy-dx);
        }
        putpixel(x,y,WHITE);
        delay(100);
        }
        getch();
}
```

Output :-



```
Enter the value of x1 and y1 : 300 400
Enter the value of x2 and y2: 200 300
```

EXPERIMENT NO. 2

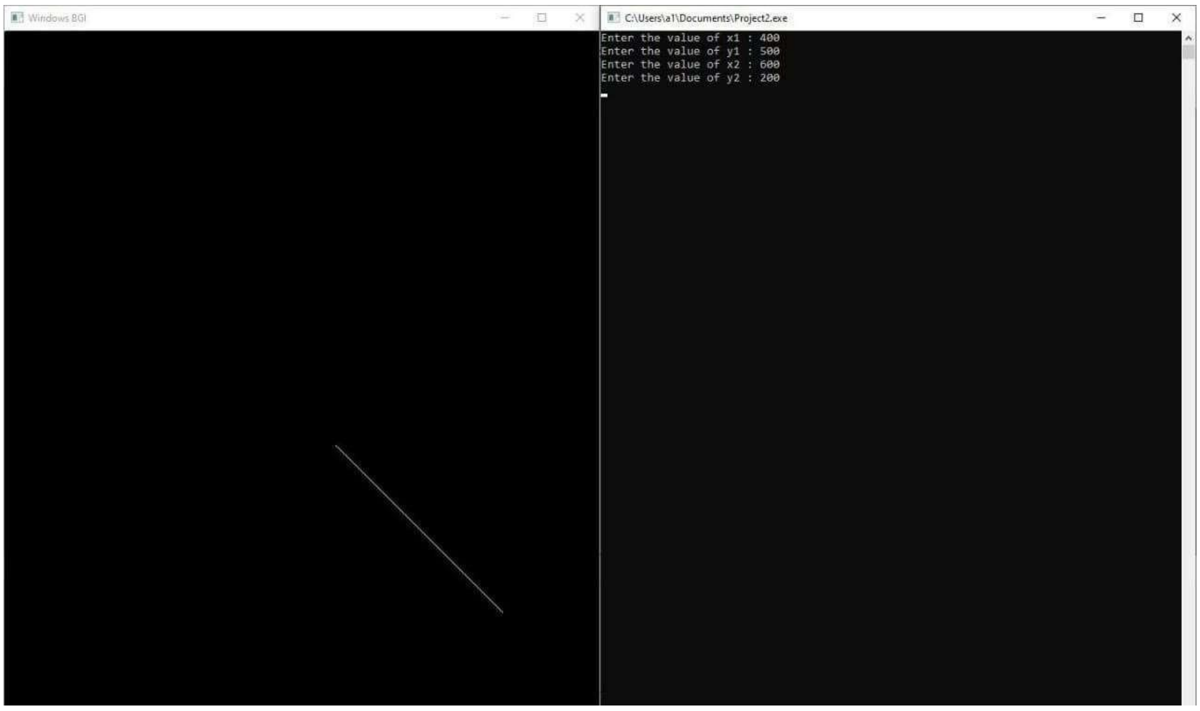
Bresenham Line Drawing Algorithm

Program Code:

```
#include<graphics.h>

#include<stdio.h> #include<math.h>int

main()

{

        int x,y,end,p,x1,x2,y1,y2,dx,dy;

        initwindow(800,800);

        printf("Enter the value of x1 : ");

        scanf("%d",&x1);

        printf("Enter the value of y1 : ");

        scanf("%d",&y1);

        printf("Enter the value of x2 : ");

        scanf("%d",&x2);

        printf("Enter the value of y2 : ");

        scanf("%d",&y2);

        dx=abs(x1-x2); dy=abs(y1-y2);  p = 2*dy-dx;

        if(x1>x2)

        {               x=x2;
                        y=y2;
                        end=x1;
        }
        else
        {               x=x1;
                        y=y1;
                        end=x2;
        }

        putpixel(x,y,WHITE)

        ;while(x<=end)

        {

                if(p<0)
                {
                        x++;
                        p=p+2*dy;

                }
                else
                {
                      x++;
                      y++;
                      p=p+2*(dy-dx);
                }
                putpixel(x,y,WHITE);

                delay(100);

        }
        getch();
```

}

Output :-



```
Enter the value of x1 : 400
Enter the value of y1 : 500
Enter the value of x2 : 600
Enter the value of y2 : 200
```

EXPERIMENT NO. 3

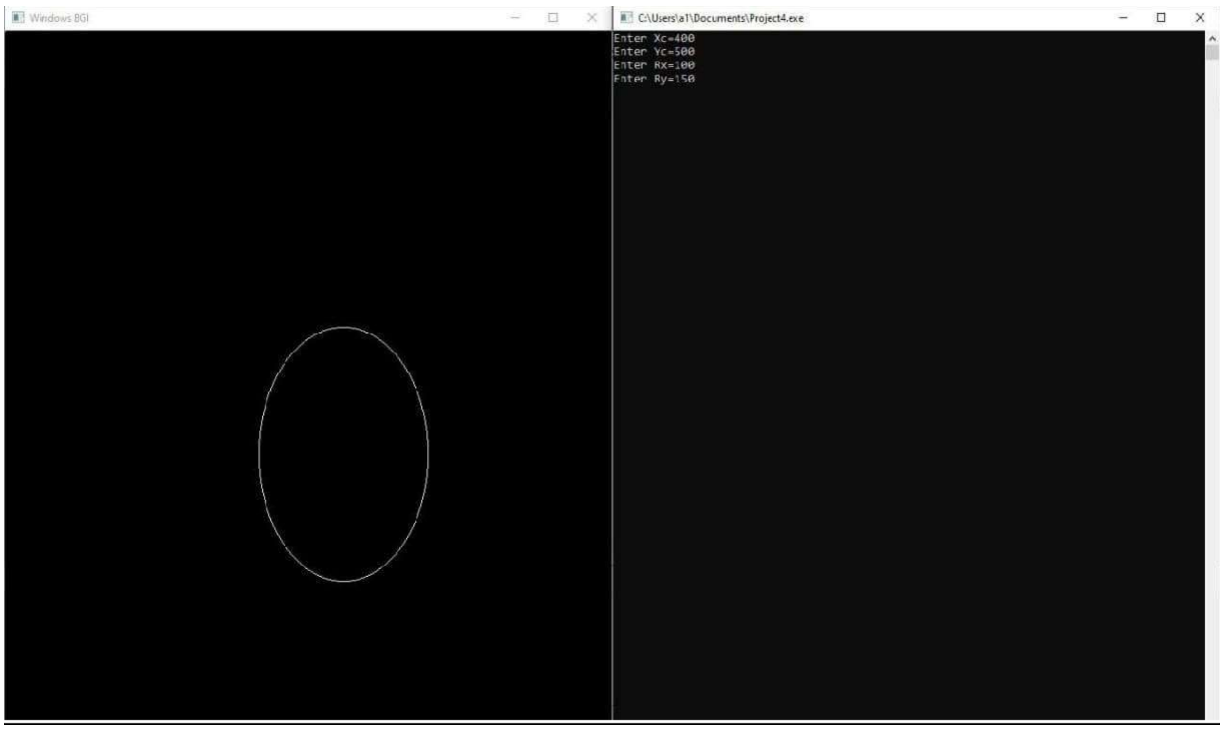
Midpoint Ellipse Algorithm

Program Code:

```
#include<stdio.h>

 #include<conio.h>

 #include<graphics.h>

 void ellipse(int xc,int yc,int rx,int ry)
 {
   int x, y, p;

   initwindow(800,800);

   x=0; y=ry;

   p=(ry*ry)-(rx*rx*ry)+((rx*rx)/4);
   while((2*x*ry*ry)<(2*y*rx*rx))
   {
        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(p<0)
        {
              x=x+1;

              p=p+(2*ry*ry*x)+(ry*ry);

        }
        else
        {
           x=x+1;y=y-1;

           p=p+(2*ry*ry*x+ry*ry)-(2*rx*rx*y);
        }
   }
   p=((float)x+0.5)*((float)x+0.5)*ry*ry+(y-1)*(y-1)*rx*rx-rx*rx*ry*ry;

          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(p>0)
        {
                y=y-1;
                p=p-(2*rx*rx*y)+(rx*rx);
        }
        else
        {
           y=y-1;x=x+1;

           p=p+(2*ry*ry*x)-(2*rx*rx*y)-(rx*rx);
        }

   }
   getch();
 }
  main()
 {
```

```
 int xc,yc,rx,ry;
 printf("Enter Xc=");
 scanf("%d",&xc);
printf("Enter Yc=");
scanf("%d",&yc);
printf("Enter Rx=");
scanf("%d",&rx);
printf("Enter Ry=");
scanf("%d",&ry);
ellipse(xc,yc,rx,ry);
getch();
}
```

Output :-



```
Enter Xc=400
Enter Yc=500
Enter Rx=100
Enter Ry=150
```

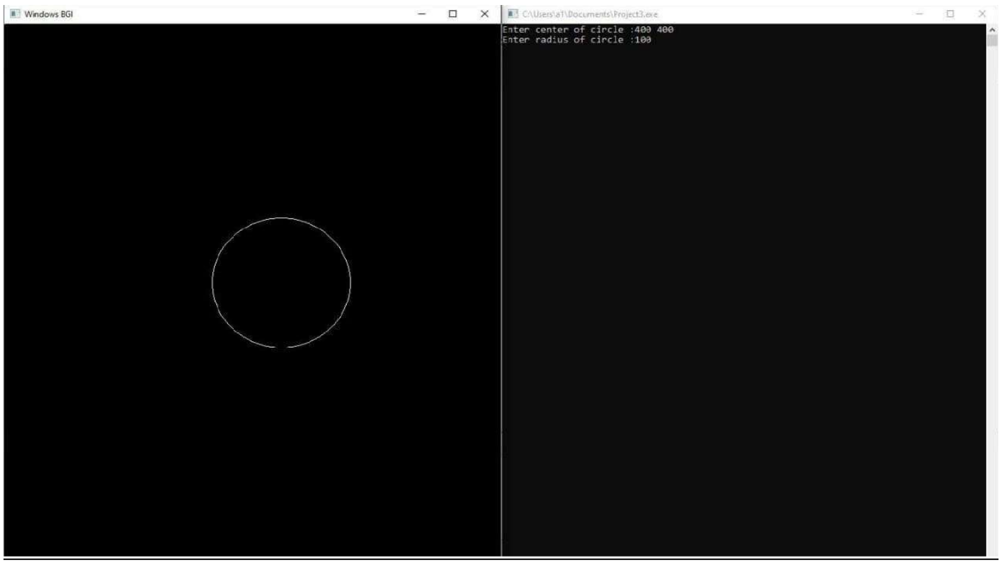
EXPERIMENT NO. 4

Midpoint Circle Algorithm

Program Code:

```
#include<graphics.h>
#include<stdio.h>
void pixel(int xc,int yc,int x,int y);
int main()
{
int xc,yc,r,x,y,p; initwindow(800,800);
printf("Enter center of circle :");
scanf("%d%d",&xc,&yc);
printf("Enter radius of circle :");
scanf("%d",&r);
x=0;
y=r;
p=1-r;
pixel(xc,yc,x,y);while(x<y)
{
        if(p<0)
{
        x++;p=p+2*x+1;
}
else
{
        x++;     y--;p=p+2*(x-y)+1;
}
pixel(xc,yc,x,y);
}
getch();
}
void pixel(int xc,int yc,int x,int 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);
}
```

Output :-



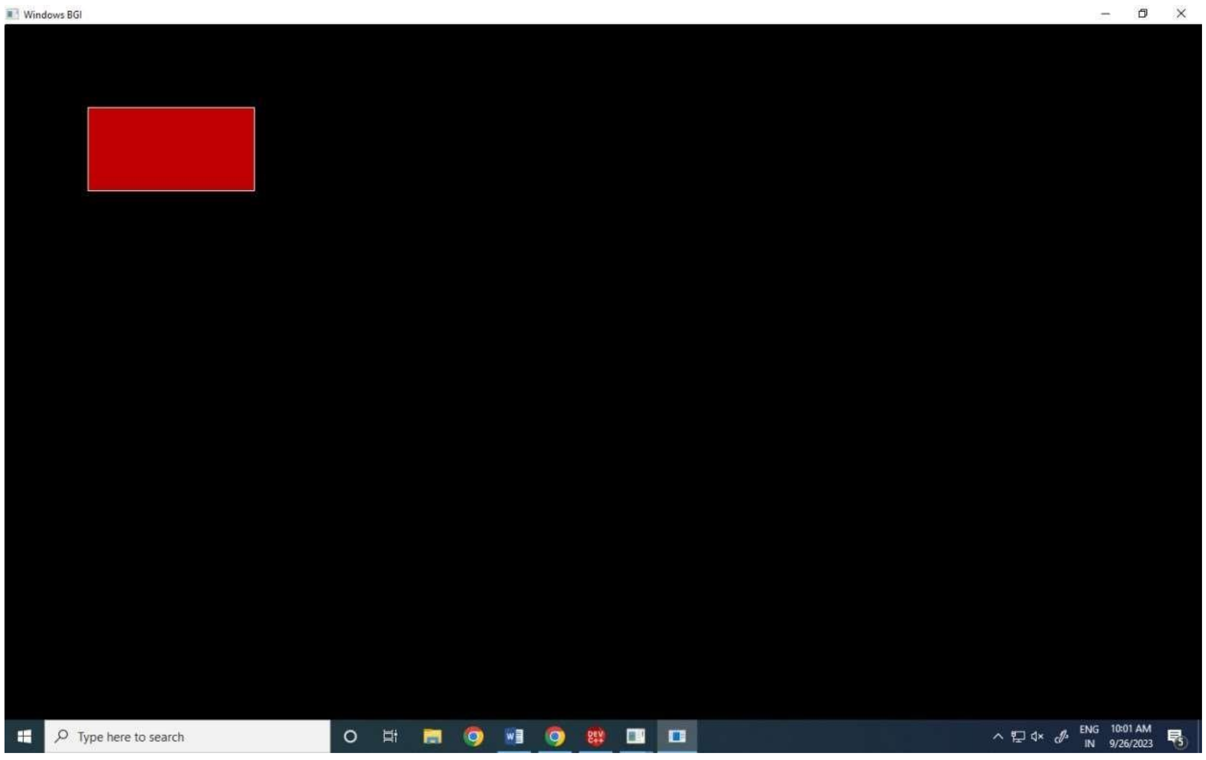
```
Enter center of circle :400 400
Enter radius of circle :100
```

EXPERIMENT NO. 5

Boundary Filling Algorithm :

```
#include <stdio.h>
#include <graphics.h>
void boundaryFill(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);
          boundaryFill(x + 1, y, fill_color, boundary_color);
          boundaryFill(x - 1, y, fill_color, boundary_color);
          boundaryFill(x, y + 1, fill_color, boundary_color);
          boundaryFill(x, y - 1, fill_color, boundary_color);
    }
}
int main() {
int gd = DETECT, gm;
 initgraph(&gd, &gm, "C:\\Turboc3\\BGI");rectangle(100,
100, 300, 200);
 int fill_color = RED;
int boundary_color = WHITE;
boundaryFill(200, 150, fill_color, boundary_color);
delay(5000); // Delay to see the result closegraph(); //
Close the graphics window
return 0;
}
```

Output :



EXPERIMENT NO. 6

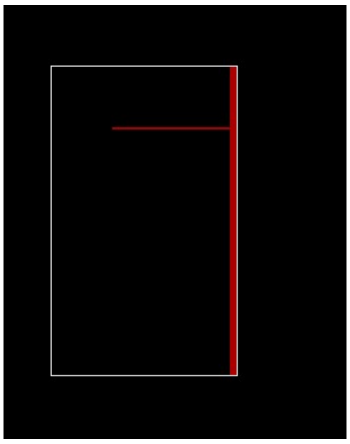
Draw and fill polygon :

Program Code :-

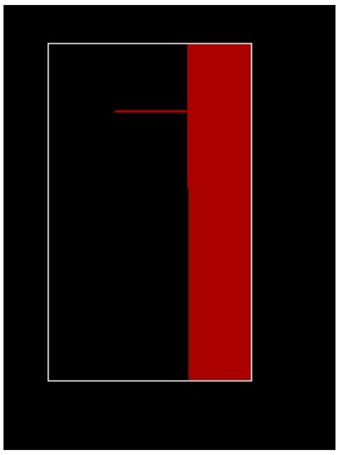
```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<dos.h>
void fill_right(x,y)
int x , y ;
{
if((getpixel(x,y) != WHITE)&&(getpixel(x,y) != RED))
{
putpixel(x,y,RED);fill_right(++x,y);
x = x - 1 ;
fill_right(x,y-1);
fill_right(x,y+1);
}
delay(1);
}
void fill_left(x,y)int x , y ;
{
if((getpixel(x,y) != WHITE)&&(getpixel(x,y) != RED))
{
putpixel(x,y,RED);

fill_left(--x,y); x
= x + 1 ;
fill_left(x,y-1);
fill_left(x,y+1);
}

delay(1);

}

void main()

{

int x,y,n,i;

int gd=DETECT,gm;
clrscr();
initgraph(&gd,&gm,"c:\\tc\\bgi");

/*- draw object -*/ line
(50,50,200,50);
line  (200,50,200,300);

line  (200,300,50,300);

line (50,300,50,50);

/*- set seed point -*/ x =
100; y = 100;
fill_right(x,y);
fill_left(x-1,y);
getch();
}
```

Output :-

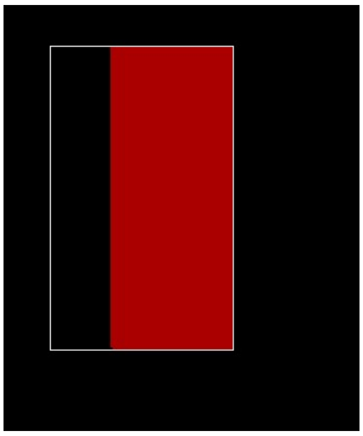
T1



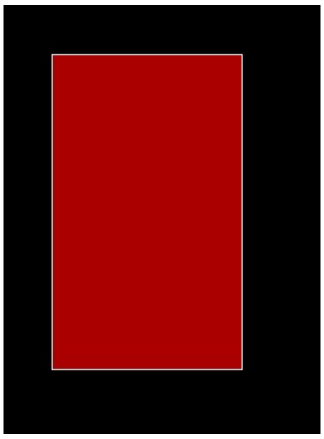
T2



T3



T4



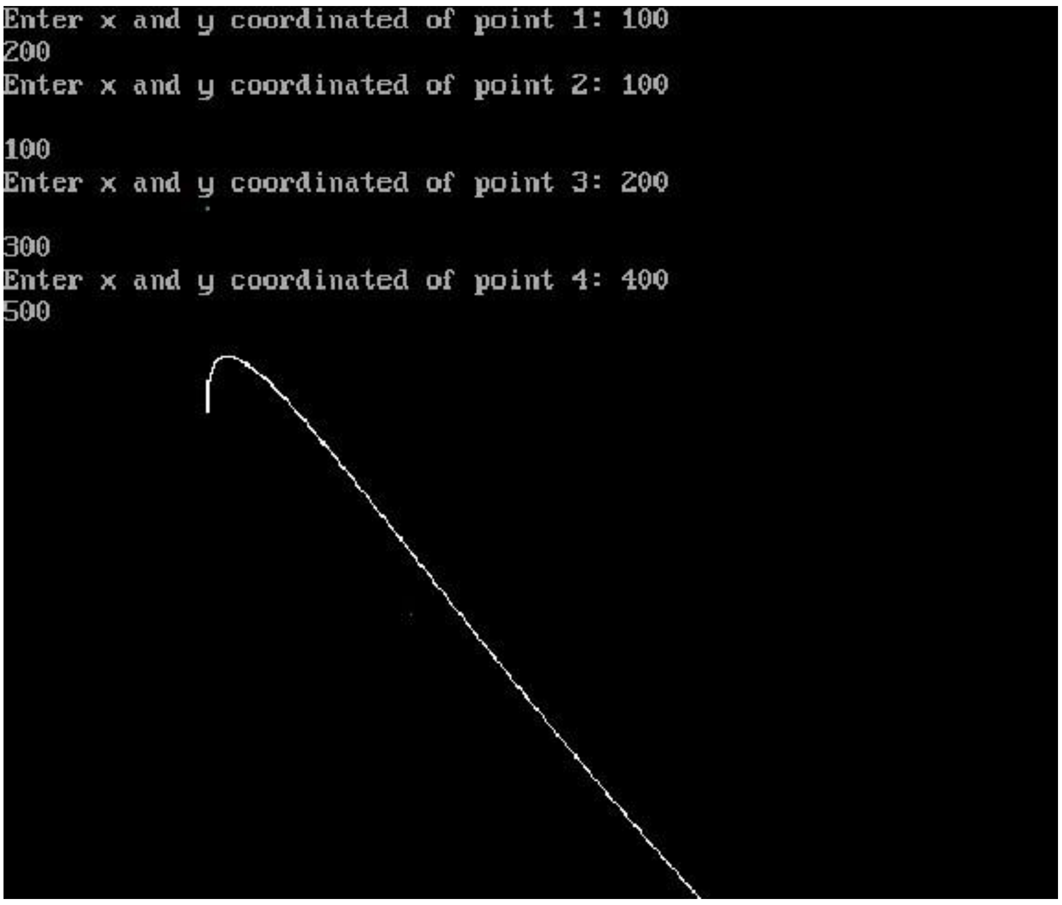
EXPERIMENT NO. 7

Bezier Curve :

Input:

```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
#include<math.h> void
main(){
    int x[4],y[4],i; double
    puty,putx,t;int
    gd=DETECT,gm;
    initgraph(&gd,&gm,"c:\\tc\\bgi");for ( i
    = 0; i < 4; i++)
    {
        printf("Enter x and y coordinated of point %d: ",i+1);scanf("%d%d",&x[i],&y[i]);
        putpixel(x[i],y[i],3);
    }
     for(t=0.0;t<=1.0;t=t+0.001){
        putx=pow(1-t,3)*x[0]+ 3*t*pow(1-t,2)*x[1]+ 3*t*t*pow(1-t,1)*x[2]+ pow(t,3)*x[3];
        puty=pow(1-t,3)*y[0]+ 3*t*pow(1-t,2)*y[1]+ 3*t*t*pow(1-t,1)*y[2]+ pow(t,3)*y[3];
        putpixel(putx,puty,WHITE);
    }
    getch(); closegraph();
}
```

Output :-



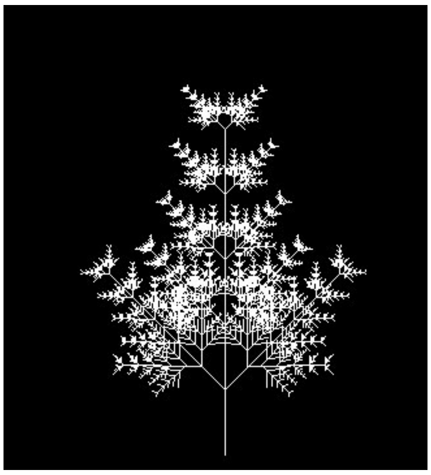
```
Enter x and y coordinated of point 1: 100
200
Enter x and y coordinated of point 2: 100

100
Enter x and y coordinated of point 3: 200

300
Enter x and y coordinated of point 4: 400
500
```

EXPERIMENT NO. 8

Fractal Generation Program

```
#include<stdio.h>
#include<math.h>
#include<graphics.h>
int a;
int drawfern(int x, int y, int l, int arg, int n)
{
 int x1, y1, i;
int l1, xpt, ypt;
 if(n>0 && !kbhit())
 {
 x1=(int)(x-l*sin(arg*3.14/180));
y1=(int)(y-l*cos(arg*3.14/180));
line(x,y,x1,y1);
l1=(int)(l/5);
for(i=1;i<6;i++)
 {
 xpt=(int)(x-i*l1*sin(arg*3.14/180));
  ypt=(int)(y-i*l1*cos(arg*3.14/180));
 drawfern(xpt,ypt,(int)(l/(i+1)),arg+a,n-1);
 drawfern(xpt,ypt,(int)(l/(i+1)),arg-a,n-1);
 delay(2);
 }
 }
}
 int main()
{
            initwindow(800,800);       int x, y,l;
            x=getmaxx()/2;
            y=getmaxy()/2;l=250;
            a=45;
                    setcolor(WHITE);
                  drawfern(x,y,l,0,5);
            getch();
}
}
```

Output:

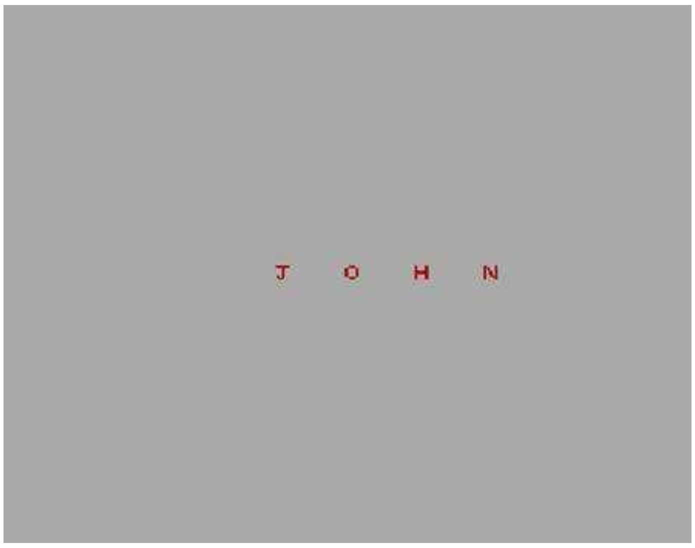


EXPERIMENT NO. 9

Program Code :-

```
#include <stdio.h>
#include <conio.h>
#include <graphics.h>void
main()
{
int i,j,k,x,y;
int gd=DETECT,gm;//DETECT is macro defined in graphics.h
int ch1[][10]={
{1,1,1,1,1,1,1,1,1,1},
{1,1,1,1,1,1,1,1,1,1},
{0,0,0,0,1,1,0,0,0,0},
{0,0,0,0,1,1,0,0,0,0},
{0,0,0,0,1,1,0,0,0,0},
{0,0,0,0,1,1,0,0,0,0},
{0,0,0,0,1,1,0,0,0,0},
{0,1,1,0,1,1,0,0,0,0},
{0,1,1,0,1,1,0,0,0,0},
{0,0,1,1,1,0,0,0,0,0}};
int ch2[][10]={
{0,0,0,1,1,1,1,0,0,0},
{0,0,1,1,1,1,1,1,0,0},
{1,1,0,0,0,0,0,0,1,1},
{1,1,0,0,0,0,0,0,1,1},
{1,1,0,0,0,0,0,0,1,1},
{1,1,0,0,0,0,0,0,1,1},
{1,1,0,0,0,0,0,0,1,1},
{1,1,0,0,0,0,0,0,1,1},
{0,0,1,1,1,1,1,1,0,0},
{0,0,0,1,1,1,1,0,0,0}};
int ch3[][10]={ {1,1,0,0,0,0,0,0,1,1},
{1,1,0,0,0,0,0,0,1,1},
{1,1,0,0,0,0,0,0,1,1},
{1,1,0,0,0,0,0,0,1,1},
{1,1,1,1,1,1,1,1,1,1},
{1,1,1,1,1,1,1,1,1,1},
{1,1,0,0,0,0,0,0,1,1},
{1,1,0,0,0,0,0,0,1,1},
{1,1,0,0,0,0,0,0,1,1},
{1,1,0,0,0,0,0,0,1,1}};
int ch4[][10]={ {1,1,0,0,0,0,0,0,1,1},
{1,1,1,1,0,0,0,0,1,1},
{1,1,0,1,1,0,0,0,1,1},
{1,1,0,1,1,0,0,0,1,1},
{1,1,0,0,1,1,0,0,1,1},
{1,1,0,0,1,1,0,0,1,1},
{1,1,0,0,0,1,1,0,1,1},
{1,1,0,0,0,1,1,0,1,1},
{1,1,0,0,0,0,1,1,1,1},
{1,1,0,0,0,0,0,0,1,1}};
initgraph(&gd,&gm,"c:\\tc\\bgi");//initialize graphic mode setbkcolor(LIGHTGRAY);
for(k=0;k<4;k++)
{
for(i=0;i<10;i++)
{
for(j=0;j<10;j++)
{
if(k==0)
{
if(ch1[i][j]==1) putpixel(j+250,i+230,RED);
}
if(k==1)
{
if(ch2[i][j]==1) putpixel(j+300,i+230,RED);
}
if(k==2)
{
if(ch3[i][j]==1) putpixel(j+350,i+230,RED);
}
if(k==3)
{
if(ch4[i][j]==1) putpixel(j+400,i+230,RED);
}
}
delay(200);
}
}
getch();
closegraph();
}
```

Output :-



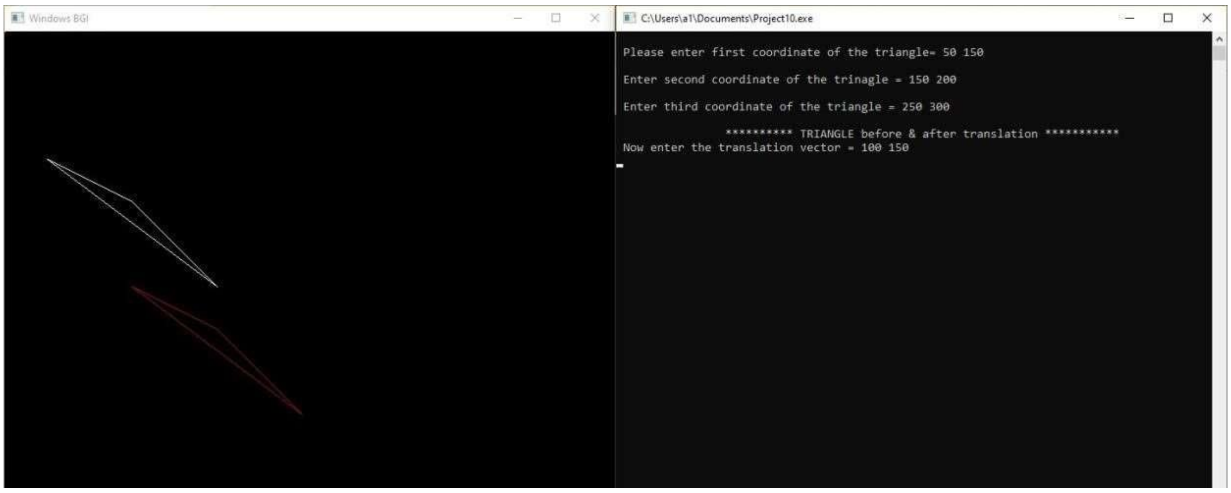
```
J   O   H   N
```

EXPERIMENT NO. 10

2D Transformations : Translation

```
#include<conio.h>
#include<graphics.h>
#include<stdio.h> int
main()
{
int gd=DETECT,gm;
int x,y,x1,y1,x2,y2,tx,ty;
initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
printf("\n Please enter first coordinate of the triangle= ");scanf("%d
%d", &x,&y);
printf("\n Enter second coordinate of the trinagle = ");
scanf("%d %d",&x1,&y1);
printf("\n Enter third coordinate of the triangle = ");
scanf("%d %d",&x2,&y2);
printf("\n\t\t**** TRIANGLE before & after translation *****");
line(x,y,x1,y1);
line(x1,y1,x2,y2);
line(x2,y2,x,y);
printf("\n Now enter the translation vector = ");scanf("%d
%d",&tx,&ty);
setcolor(RED);
line(x+tx,y+ty,x1+tx,y1+ty);
line(x1+tx,y1+ty,x2+tx,y2+ty);
line(x2+tx,y2+ty,x+tx,y+ty);
getch();
closegraph();
}
```

Output :

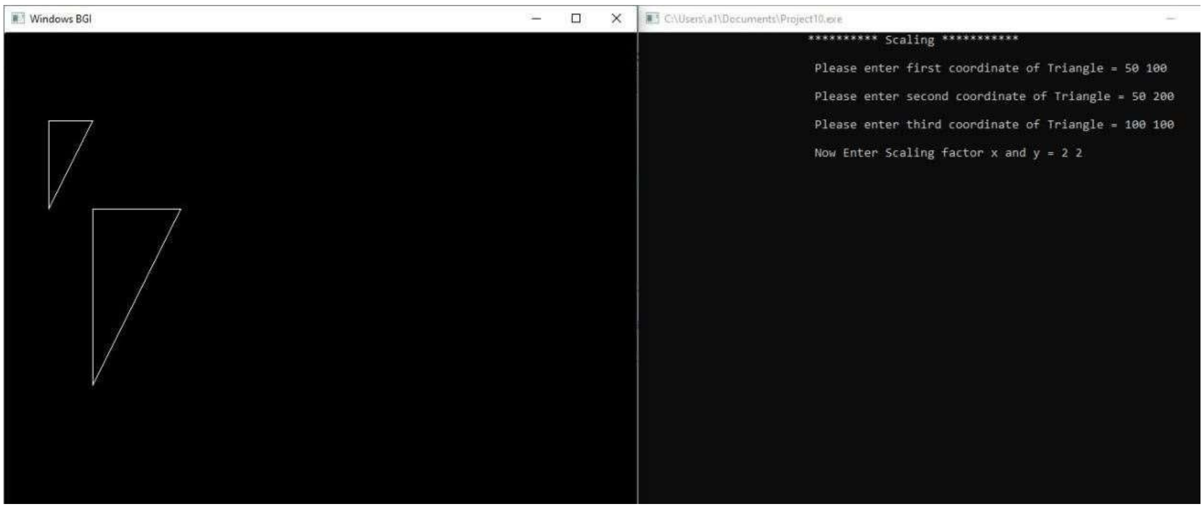


```
Please enter first coordinate of the triangle= 50 150

Enter second coordinate of the trinagle = 150 200

Enter third coordinate of the triangle = 250 300

               ********** TRIANGLE before & after translation ***********
Now enter the translation vector = 100 150
```

2D Transformation : Scaling

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
int main(){
int x,y,x1,y1,x2,y2;
int scl_fctr_x,scl_fctr_y;int
gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
printf("\t\t\t**** Scaling *****\n");
printf("\n\t\t\t Please enter first coordinate of Triangle = ");scanf("%d
%d",&x,&y);
printf("\n\t\t\t Please enter second coordinate of Triangle = ");
 scanf("%d %d",&x1,&y1);
printf("\n\t\t\t Please enter third coordinate of Triangle = ");scanf("%d
%d",&x2,&y2);
line(x,y,x1,y1);
line(x1,y1,x2,y2);
line(x2,y2,x,y);
printf("\n\t\t\t Now Enter Scaling factor x and y = ");
 scanf("%d %d",&scl_fctr_x,&scl_fctr_y);
x = x* scl_fctr_x;
 x1 = x1* scl_fctr_x;
x2 = x2* scl_fctr_x;
y  = y*  scl_fctr_y;
y1 = y1* scl_fctr_y;
 y2= y2 * scl_fctr_y ;
line(x,y,x1,y1);
 line(x1,y1,x2,y2);
 line(x2,y2,x,y);
getch();
closegraph();
}
```

Output :-

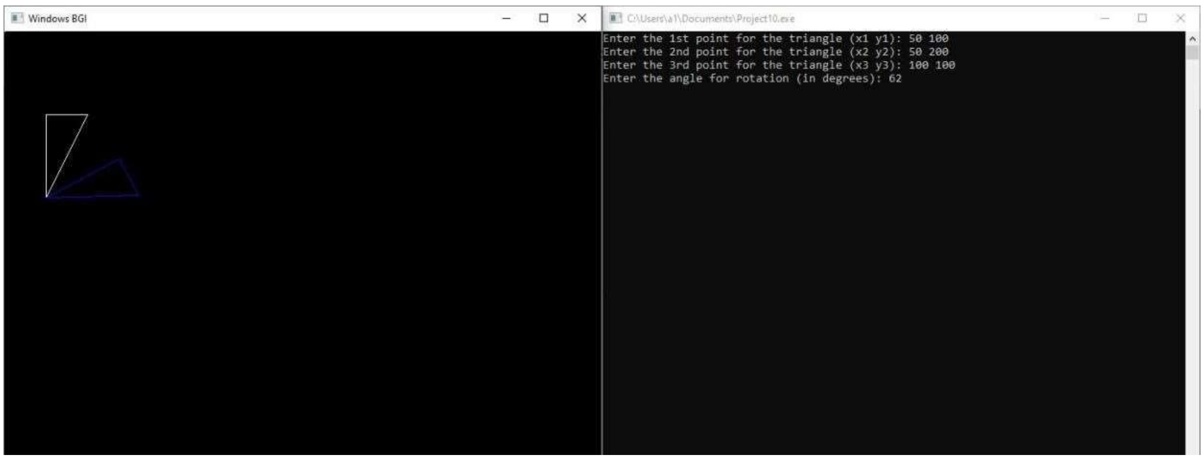


```
********** Scaling ***********

 Please enter first coordinate of Triangle = 50 100

 Please enter second coordinate of Triangle = 50 200

 Please enter third coordinate of Triangle = 100 100

 Now Enter Scaling factor x and y = 2 2
```

2D Transformation: Rotation

```
#include <stdio.h>
#include <conio.h>
#include <graphics.h>
 #include <math.h>
void DrawTriangle(int x1, int y1, int x2, int y2, int x3, int y3);
void RotateTriangle(int x1, int y1, int x2, int y2, int x3, int y3, float angle);int
main()
{
int gd = DETECT, gm;
int x1, y1, x2, y2, x3, y3;
float angle;
initgraph(&gd, &gm, "");
printf("Enter the 1st point for the triangle (x1 y1): ");
scanf("%d%d", &x1, &y1);
printf("Enter the 2nd point for the triangle (x2 y2): ");
scanf("%d%d", &x2, &y2);
printf("Enter the 3rd point for the triangle (x3 y3): ");
scanf("%d%d", &x3, &y3);
DrawTriangle(x1, y1, x2, y2, x3, y3);
printf("Enter the angle for rotation (in degrees): ");
scanf("%f", &angle);
RotateTriangle(x1, y1, x2, y2, x3, y3, angle);
getch();
closegraph();
return 0;
}
void DrawTriangle(int x1, int y1, int x2, int y2, int x3, int y3)
{
   line(x1, y1, x2, y2);
   line(x2, y2, x3, y3);
   line(x3, y3, x1, y1);
}
void RotateTriangle(int x1, int y1, int x2, int y2, int x3, int y3, float angle)
{
int p = x2, q = y2;
float radianAngle = (angle * 3.14) / 180.0;
int a1 = p + (x1 - p) * cos(radianAngle) - (y1 - q) * sin(radianAngle);
int b1 = q + (x1 - p) * sin(radianAngle) + (y1 - q) * cos(radianAngle);
int a2 = p + (x2 - p) * cos(radianAngle) - (y2 - q) * sin(radianAngle);
int b2 = q + (x2 - p) * sin(radianAngle) + (y2 - q) * cos(radianAngle);
int a3 = p + (x3 - p) * cos(radianAngle) - (y3 - q) * sin(radianAngle);
int b3 = q + (x3 - p) * sin(radianAngle) + (y3 - q) * cos(radianAngle);
setcolor(1);
DrawTriangle(a1, b1, a2, b2, a3, b3);
}
```

Output :



```
Enter the 1st point for the triangle (x1 y1): 50 100
Enter the 2nd point for the triangle (x2 y2): 50 200
Enter the 3rd point for the triangle (x3 y3): 100 100
Enter the angle for rotation (in degrees): 62
```