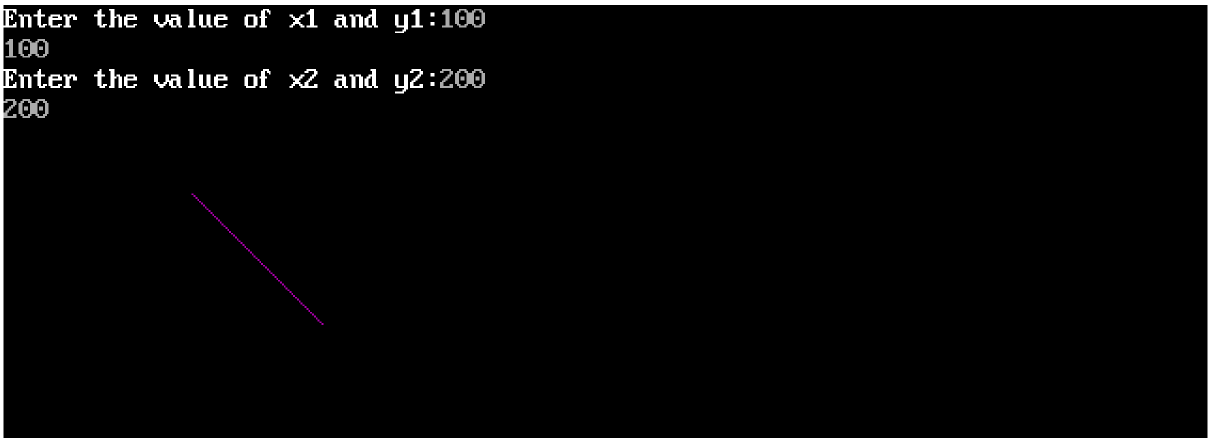
1) Practical 1

Aim - Write a Program in C for DDA Line Drawing Algorithm

```
#include <graphics.h>

#include <stdio.h>

#include <math.h>

#include <dos.h> int

main( )

{

float x,y,x1,y1,x2,y2,dx,dy,step; int

i,gd=DETECT,gm;

initgraph(&gd,&gm,"c:\\turboc3\\bgi");

printf("Enter the value of x1 and y1 : ");

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

the value of x2 and y2: ");

scanf("%f%f",&x2,&y2); dx=abs(x2-x1);

dy=abs(y2-y1);

 if(dx>=dy)

{step=dx;

else

step=dy;

dx=dx/step;

dy=dy/step;

x=x1;

 y=y1;

i=1;

 while(i<=step)
{
putpixel(x,y,5);
x=x+dx;
y=y+dy;
 i=i+1;
delay(100);
}
closegraph();
return 0;
}
```

Output :-



```
Enter the value of x1 and y1:100
100
Enter the value of x2 and y2:200
200
```

---

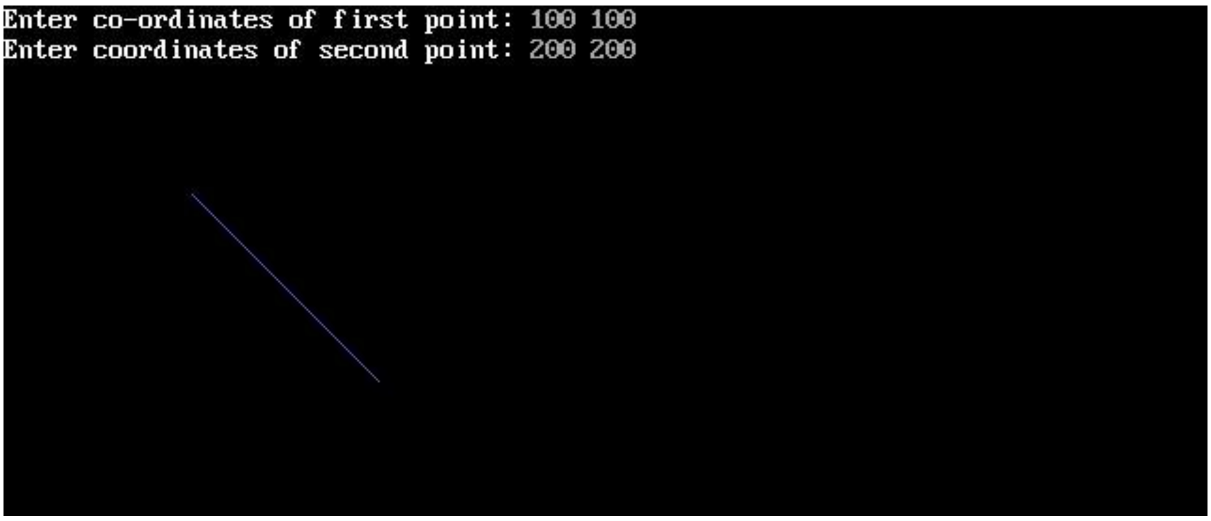
2) Practical 2

Aim - Write a Program in C to implement Bresenham's Line 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, "c:\\tc\\bgi");
printf("Enter co-ordinates of first point: ");
scanf("%d%d", &x0, &y0); printf("Enter coordinates of second point: ");
scanf("%d%d", &x1,
&y1); drawline(x0, y0, x1, y1); return 0;
}
```

Output :-



```
Enter co-ordinates of first point: 100 100
Enter coordinates of second point: 200 200
```

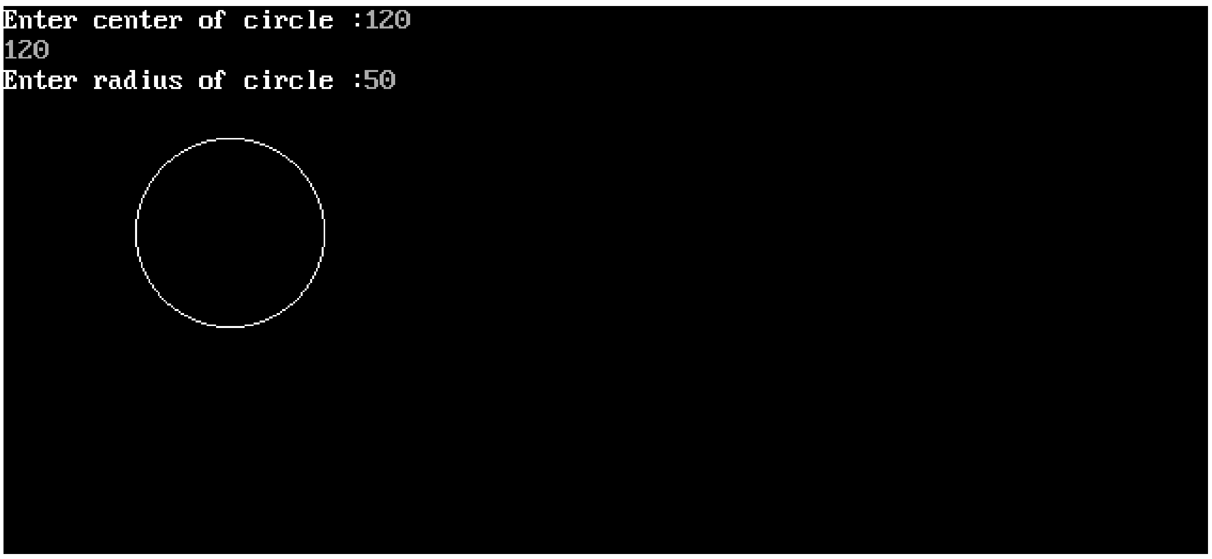
---

3) Practical 3

Aim - Write a Program in C to implement Midpoint Circle algorithm

```
#include<graphics.h>
#include<stdio.h>
#include<conio.h>
void pixel(int xc,int yc,int x,int y);
int main()
{
int gd,gm,xc,yc,r,x,y,p;
detectgraph(&gd,&gm);
initgraph(&gd,&gm,"C://Tc//BGI");
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();
closegraph();
return 0;
}
void pixel(int xc,int yc,int x,int y)
{
putpixel(xc+x,yc+y,WHITE);
putpixel(xc+x,ycy,WHITE);
putpixel(xc-x,yc+y,WHITE);
putpixel(xcx,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 :120
120
Enter radius of circle :50
```

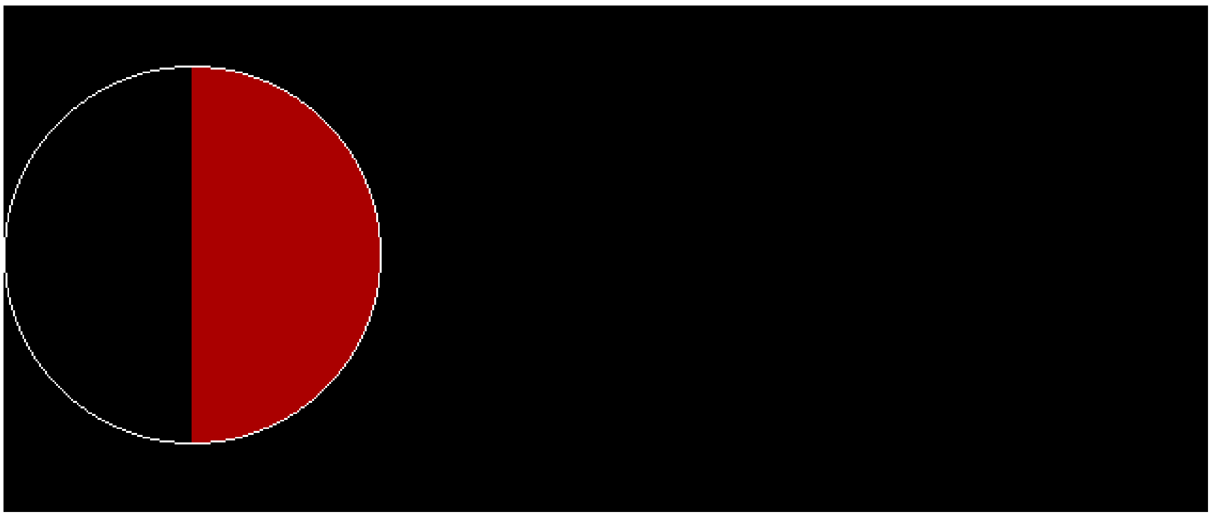
4) Practical 5

Aim - Write a Program in C to implement Area Filling Algorithm: Boundary Fill and Flood Fill

```
#include<stdio.h>
#include<graphics.h>
#include<dos.h>
void boundaryfill(int x,int y,int f_color,int b_color)
{
```

```
if(getpixel(x,y)!=b_color&&getpixel(x,y)!=f_color)
{
putpixel(x,y,f_color);
boundaryfill(x+1,y,f_color,b_color);
boundaryfill(x,y+1,f_color,b_color);
boundaryfill(x,y,f_color,b_color);
boundaryfill(x,y-1,f_color,b_color);
}
}
//getpixel(x,y) gives the color of specified pixel
int main()
{
int gm,gd=DETECT,radius;
int x,y;
printf("Enter x and y positions for circle\n");
scanf("%d%d",&x,&y);
printf("Enter radius of circle\n");
scanf("%d",&radius);
initgraph(&gd,&gm,"c:\\tc\\bgi");
circle(x,y,radius);
boundaryfill(x,y,4,15);
delay(5000);
closegraph();
return 0;
}
```

Output :-

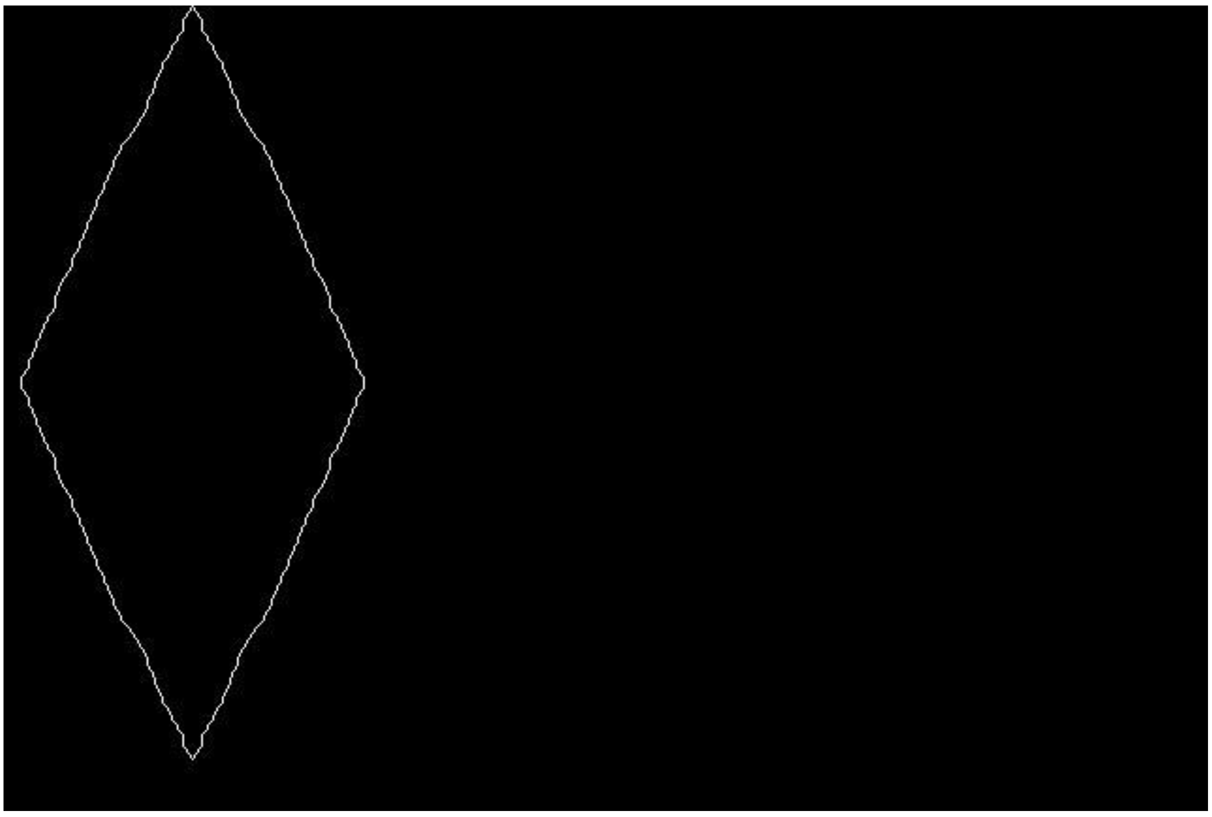


5) Practical 4

Aim - Write a Program in C to implement Midpoint Ellipse 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, "c:\\tc\\bgi");
printf("Enter co-ordinates of first point: ");
scanf("%d%d", &x0, &y0); printf("Enter coordinates of second point: ");
scanf("%d%d", &x1,
&y1); drawline(x0, y0, x1, y1); return 0;
}
int 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 :-

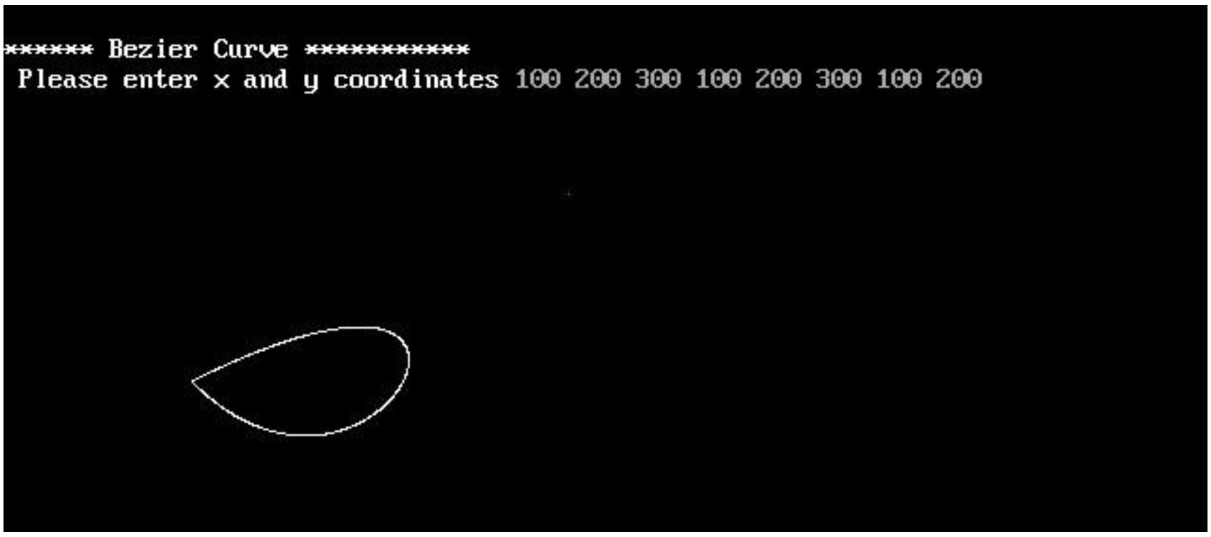


6) Practical 9

Aim - Write a Program in C to implement Bezier Curve

```
#include<graphics.h>
#include<math.h>
#include<conio.h>
#include<stdio.h>
int main()
{
 int x[4],y[4],i;
double put_x,put_y,t;
int gr=DETECT,gm;
initgraph(&gr,&gm,"C:\\TURBOC3\\BGI");
printf("\n*** Bezier Curve ****");
printf("\n Please enter x and y coordinates ");
for(i=0;i<4;i++)
{
scanf("%d%d",&x[i],&y[i]);
putpixel(x[i],y[i],3);
}
for(t=0.0;t<=1.0;t=t+0.001)
{
put_x = pow(1-t,3)x[0] + 3*t*pow(1-t,2)*x[1] + 3*t*t(1-t)*x[2] + pow(t,3)*x[3];
// Formula to draw curve
put_y = pow(1-t,3)y[0] + 3*t*pow(1-t,2)*y[1] + 3*t*t(1-t)*y[2] +
pow(t,3)*y[3]; putpixel(put_x,put_y, WHITE);
}
getch();
closegraph();
}
```

Output :-



```
******* Bezier Curve ************
 Please enter x and y coordinates 100 200 300 100 200 300 100 200
```

---

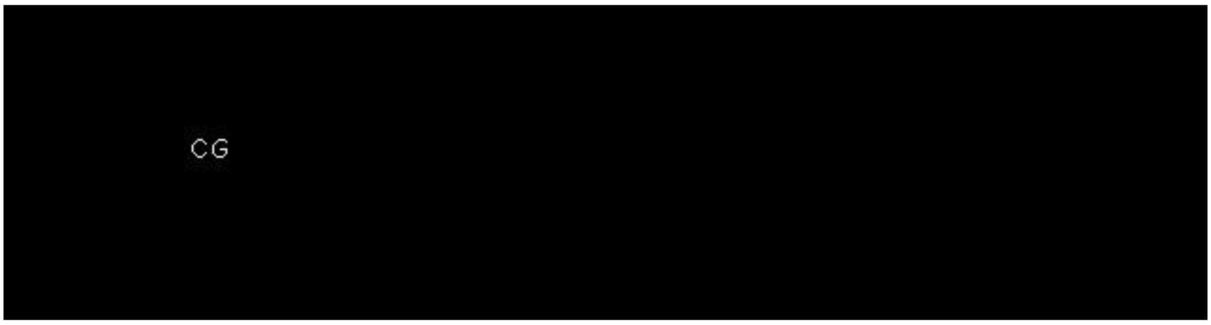
7) Practical 10

Aim - Write a Program in C for Character Generation

```
#include<stdio.h>
#include<conio.h>
```

```
#include<graphics.h>
int main()
{
int gd=DETECT,gm,i,j;
int a[20][20]={{0,0,0,1,1,1,0,0,0,0,0,0,0,0,1,1,1,1,0,0},
{0,0,1,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1,0},
{0,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,1},
{1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0},
{1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0},
{1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,1,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0},
{0,1,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,1,0},
{0,0,1,0,0,0,1,0,0,0,0,0,1,0,0,0,0,1,0,0},
{0,0,0,1,1,1,0,0,0,0,0,0,0,1,1,1,1,0,0,0}};
initgraph(&gd,&gm,"c:\\tc\\bgi");
for(i=0;i<19;i++)
{
for(j=0;j<19;j++)
{
if(a[i][j]==1)
putpixel(100+j,200+i,WHITE);
}
}
getch();
return 0;
}
```

Output :-



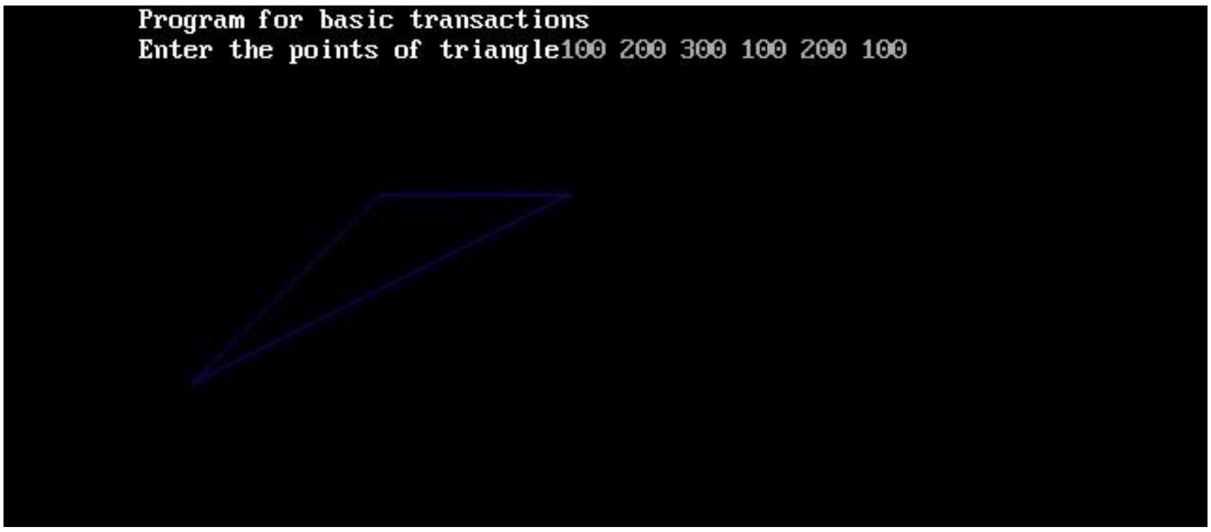
8) Practical 6

Write a Program in C to implement 2D transformations:

1. Translation 2. Rotation 3. Scaling

```
#include<graphics.h>
#include<stdlib.h>
#include<stdio.h>
#include<conio.h>
#include<math.h>
int main()
{
int gm;
int gd=DETECT;
int x1,x2,x3,y1,y2,y3,nx1,nx2,nx3,ny1,ny2,ny3,c;
int sx,sy,xt,yt,r;
float t;
initgraph(&gd,&gm,"c:\\tc\\bgi");
printf("\t Program for basic transactions");
printf("\n\t Enter the points of triangle");
setcolor(1);
scanf("%d%d%d%d%d%d",&x1,&y1,&x2,&y2,&x3,&y3);
line(x1,y1,x2,y2);
line(x2,y2,x3,y3);
line(x3,y3,x1,y1);
getch();
printf("\n 1.Transaction\n 2.Rotation\n 3.Scalling\n 4.exit");
printf("Enter your choice:");
scanf("%d",&c);
switch(c)
{
case 1:
printf("\n Enter the translation factor");
scanf("%d%d",&xt,&yt);
nx1=x1+xt;
ny1=y1+yt;
nx2=x2+xt;
ny2=y2+yt;
nx3=x3+xt;
ny3=y3+yt;
line(nx1,ny1,nx2,ny2);
line(nx2,ny2,nx3,ny3);
line(nx3,ny3,nx1,ny1);
break;

case 2:
printf("\n Enter the angle of rotation");
scanf("%d",&r);
t=3.14*r/180; nx1=abs(x1*cos(t)-
y1*sin(t));
ny1=abs(x1*sin(t)+y1*cos(t));
nx2=abs(x2*cos(t)-y2*sin(t));
ny2=abs(x2*sin(t)+y2*cos(t));
nx3=abs(x3*cos(t)-y3*sin(t));
ny3=abs(x3*sin(t)+y3*cos(t));
line(nx1,ny1,nx2,ny2);
line(nx2,ny2,nx3,ny3);
line(nx3,ny3,nx1,ny1);
getch();
break;
case 3:
printf("\n Enter the scalling factor");
scanf("%d%d",&sx,&sy);
nx1=x1*sx; ny1=y2*sy;
nx2=x2*sx; ny2=y2*sy;
nx3=x3*sx; ny3=y3*sy;
line(nx1,ny1,nx2,ny2);
line(nx2,ny2,nx3,ny3);
line(nx3,ny3,nx1,ny1);
getch();
break;
case 4:
break;
default:
printf("Enter the correct choice");
}
closegraph();
return 0;
}
```

Output :-



```
Program for basic transactions
Enter the points of triangle100 200 300 100 200 100
```