Universal College of Engineering, Kaman
Department of Computer Engineering
Subject: Computer Graphics

Experiment No: 1

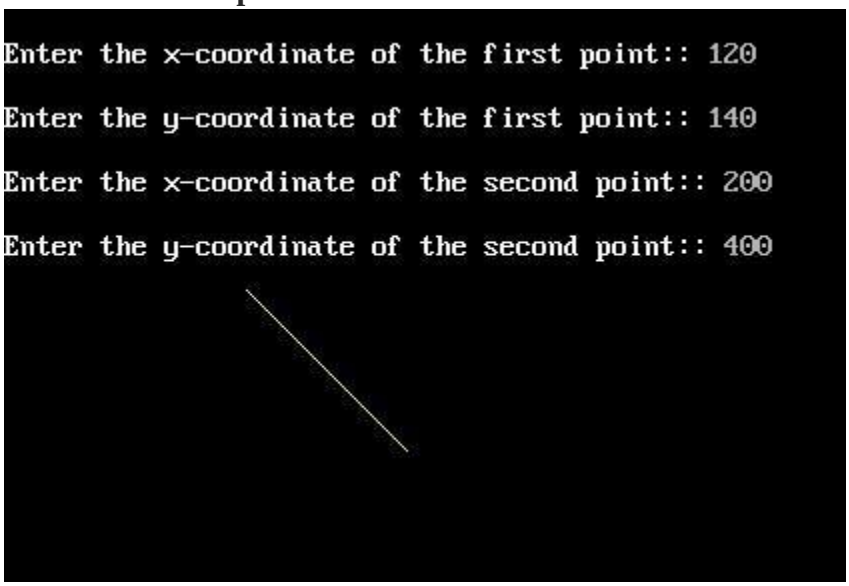
Roll No:141	**Name:** TANAY PATIL	**Div: D**	**Batch: D3**

Aim: To implement DDA line drawing algorithm in C.

Program:

```
#include<stdio.h>
#include<graphics.h
> #include<conio.h>
void main()
{ int x,y,x1,y1,x2,y2,p,dx,dy; int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
printf("\nEnter the x-coordinate of the first point:
"); scanf("&d",&x1); printf("\nEnter the y-
coordinate of the first point: "); scanf("&d",&y1);
printf("\nEnter the x-coordinate of the second
point: "); scanf("&d",&x2); printf("\nEnter the y-
coordinate of the second point: ");
scanf("&d",&y2); x=x1; y=y1; dx=x2-x1; dy=y2-y1;
p=(2*dy-dx); while(x<=x2) { if(p<0)
{ x=x+1;
p=p+2*dy
;
}
else
{ x=x+1; y=y+1;
p=p+(2*dy)-
(2*dx);
}
putpixel(x,y,YELLOW)
;
}
getch();
closegraph();}
```

Screenshot of output



Conclusion:

1. We encoded program for drawing DDA line using different coordinates.
2. DDA algorithm is faster method for calculating a pixel position for a direct use.

Universal College of Engineering, Kaman
Department of Computer Engineering
Subject: Computer Graphics

Experiment No: 2

RollNo:141	Name: TANAY PATIL	Div: D	Batch: D3

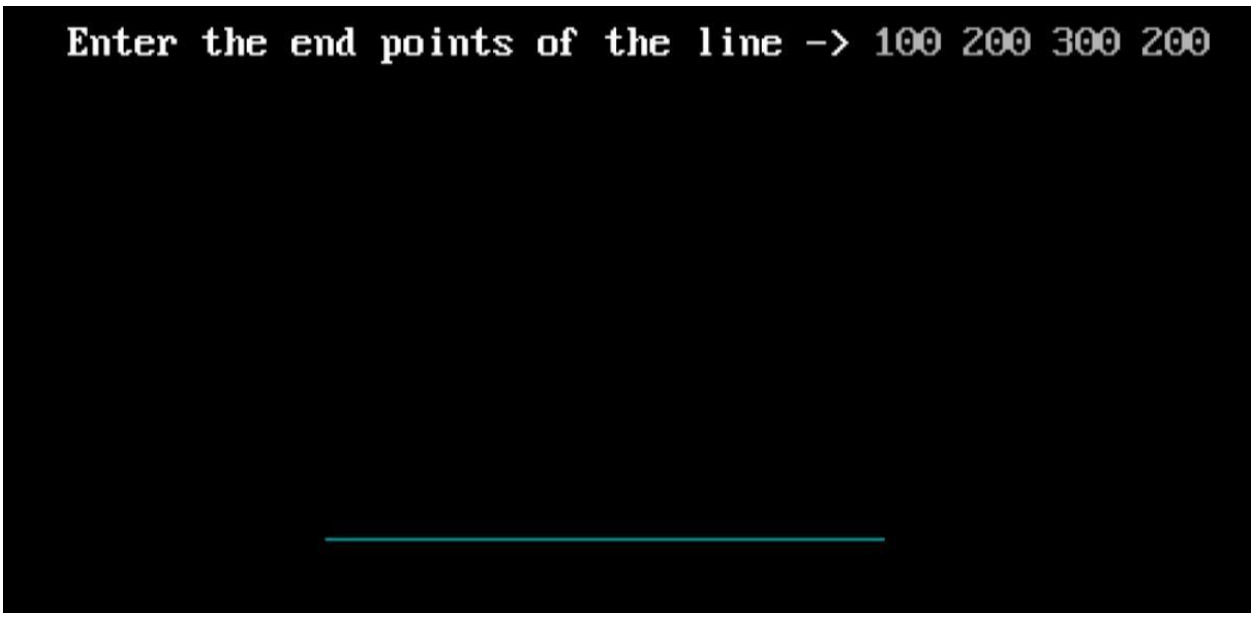
Aim: To implement Bresenham's Line algorithm.

Program:

```
#include <stdio.h>
#include <conio.h>
#include <graphics.h> int gd
= DETECT, gm; void
main(){ int x1, y1, x2, y2,
delx, dely, p; clrscr();
initgraph(&gd, &gm, ""); printf("\n Enter
the end points of the line -> "); scanf("%d %d
%d %d", &x1, &y1, &x2, &y2);
putpixel(x1, y1, 3);
delx = x2 - x1; dely
= y2 - y1; p = (2 *
dely) - delx;
while(delx - 1){
if(p < 0){ x1++;
putpixel(x1, y1, 3);
p += (2 * dely);
}
else{
x1++; y1++;
putpixel(x1, y1,
3); p += (2 * dely)
- (2 * delx);
}
delx--;
```

}getch(
); }

Screenshot of output



Conclusion:

1. We encoded program for drawing Bresenhams line using different coordinate.

Universal College of Engineering, Kaman
Department of Computer Engineering
Subject: Computer Graphics

Experiment No: 3

Roll No: 141	Name:TANAY PATIL	Div: D	Batch: D3

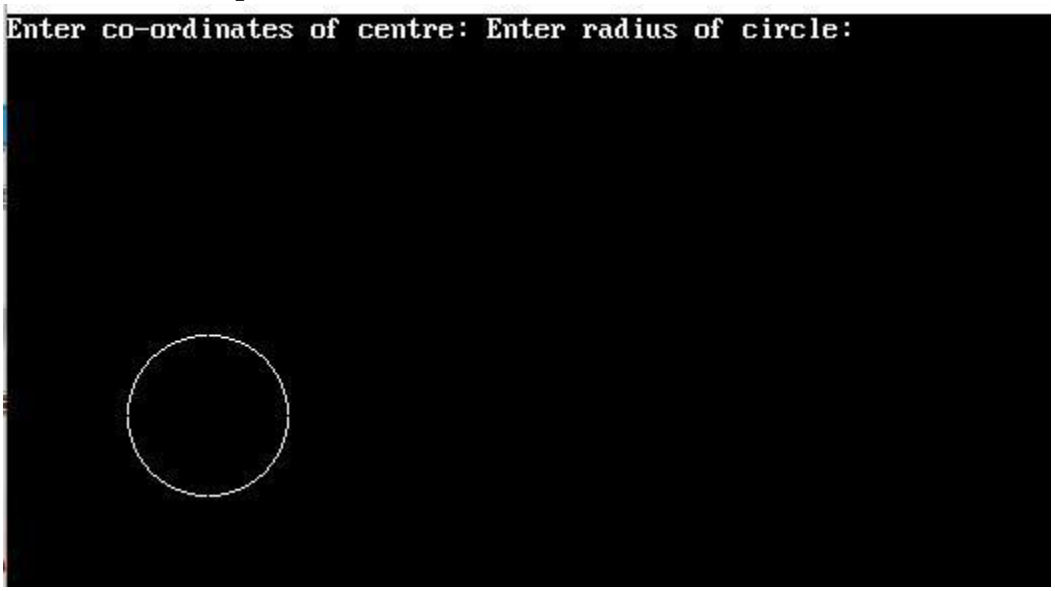
Aim: To implement midpoint Circle algorithm.

Program:

```
#include <stdio.h>
#include <graphics.h> #include <conio.h> void main() { int
xc,yc,x,y,p,r; int gdriver = DETECT, gmode;
initgraph(&gdriver,&gmode,"C:\\TurboC++\\Disk\\TURBOC3\\BGI");
printf("Enter co-ordinates of centre: "); xc=100,yc=200;
printf("Enter radius of circle: "); r=40;
x = 0; y
= r; p =
1-r;
do{
if(p<0){
x=x+1;
p = p+2*x +1;
} else{ x=x+1;
y=y-1; p +=
2*x-2*y +1;
}
putpixel(xc+x,yc+y,WHITE);
putpixel(xc+y,yc+x,WHITE); putpixel(xc+x,yc-
y,WHITE); putpixel(xc+y,yc-x,WHITE);
putpixel(xc-x,yc-y,WHITE); putpixel(xc-y,yc-
x,WHITE); putpixel(xc-x,yc+y,WHITE);
putpixel(xc-y,yc+x,WHITE);
}
while(x<=y);
```

```
getch();
closegraph();
}
```

Screenshot of output



Conclusion:

1. We encoded program for Midpoint Circle using its algorithm.

Universal College of Engineering, Kaman
Department of Computer Engineering
Subject: Computer Graphics

Experiment No: 4

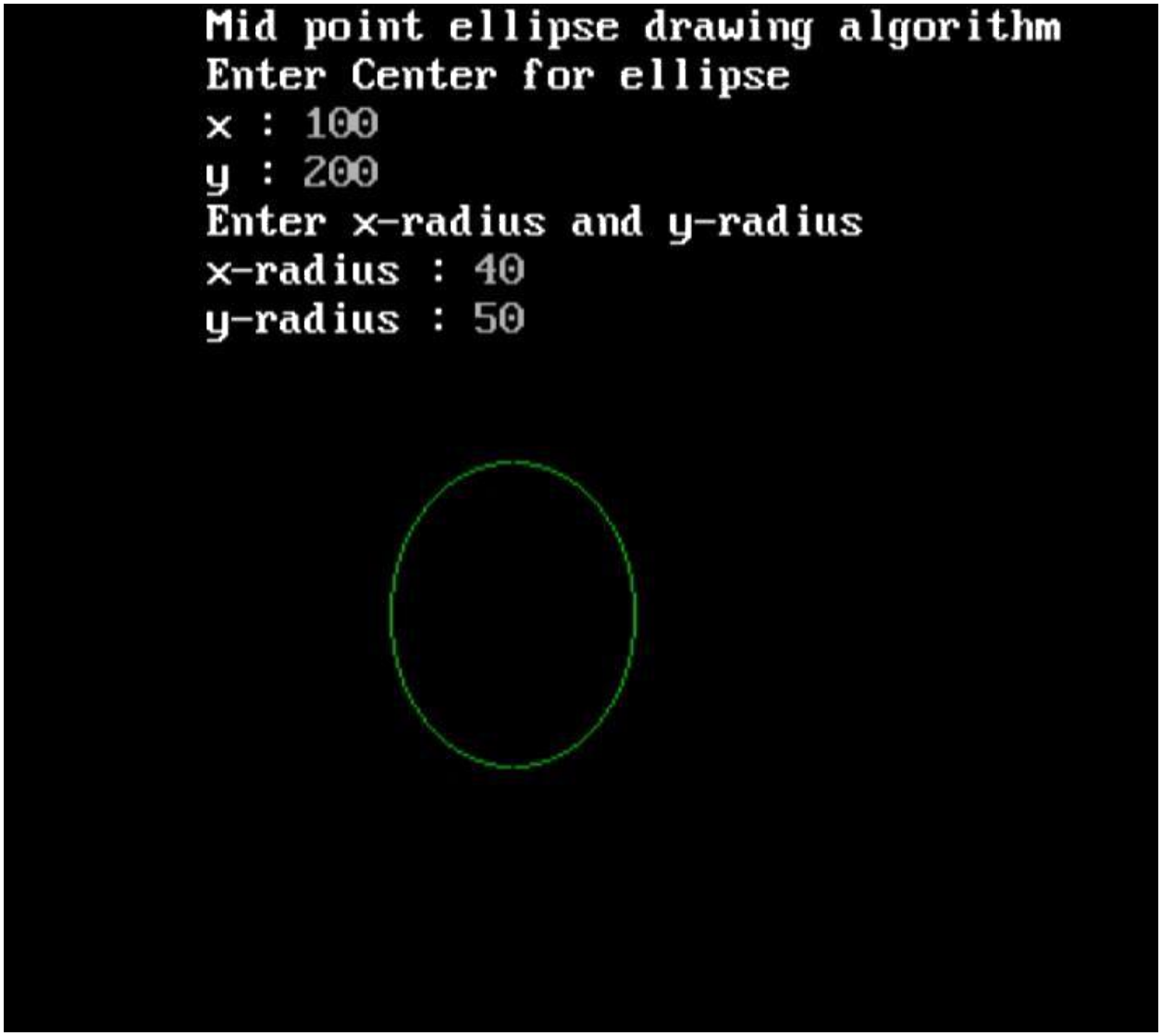
Roll No: 141	Name: TANAY PATIL	Div: D	Batch: D3

Aim: To implement midpoint ellipse algorithm

Program:

```
#include<conio.h>
#include<stdio.h> #include<graphics.h> void main()
{ int gd=DETECT,gm; float
x,y,xc,yc,rx,ry,pk,pk1;clrscr();
initgraph(&gd,&gm,"..\\bgi"); printf("Mid point
ellipse drawing algorithm\n"); printf("Enter Center
for ellipse\nx : "); scanf("%f",&xc); printf("y : ");
scanf("%f",&yc); printf("Enter x-radius and y-
radius\nx-radius : "); scanf("%f",&rx); printf("y-
radius : "); scanf("%f",&ry); x=0; y=ry; pk=(ry*ry)-
(rx*rx*ry)+((rx*rx)/4);
while((2*x*ry*ry)<(2*y*rx*rx)) { if(pk<=0) { x=x+1;
pk1=pk+(2*ry*ry*x)+(ry*ry);
} else { x=x+1; y=y-1; pk1=pk+(2*ry*ry*x)-
(2*rx*rx*y)+(ry*ry); } pk=pk1;putpixel(xc+x,yc+y,2);
putpixel(xc-x,yc+y,2); putpixel(xc+x,yc-y,2); putpixel(xc-
x,yc-y,2); } pk=((x+0.5)*(x+0.5)*ry*ry)+((y-1)*(y-
1)*rx*rx)-(rx*rx*ry*ry); while(y>0) { if(pk>0) { y=y-1;
pk1=pk-(2*rx*rx*y)+(rx*rx); } else { x=x+1; y=y-1;
pk1=pk+(2*ry*ry*x)-(2*rx*rx*y)+(rx*rx); } pk=pk1;
putpixel(xc+x,yc+y,2); putpixel(xc-x,yc+y,2);
putpixel(xc+x,yc-y,2); putpixel(xc-x,yc-y,2); } getch();}
```

Screenshot of output



Conclusion:

1. We encoded program for Midpoint Ellipse using its algorithm.

Universal College of Engineering, Kaman
Department of Computer Engineering
Subject: Computer Graphics

Experiment No: 5

RollNo:141	**Name:** TANAY PATIL	**Div: D**	**Batch: D3**

Aim: To implement area filling algorithm:

a. Boundary fill algorithm

b. Flood fill algorithm

Program:

a. Boundary fill algorithm

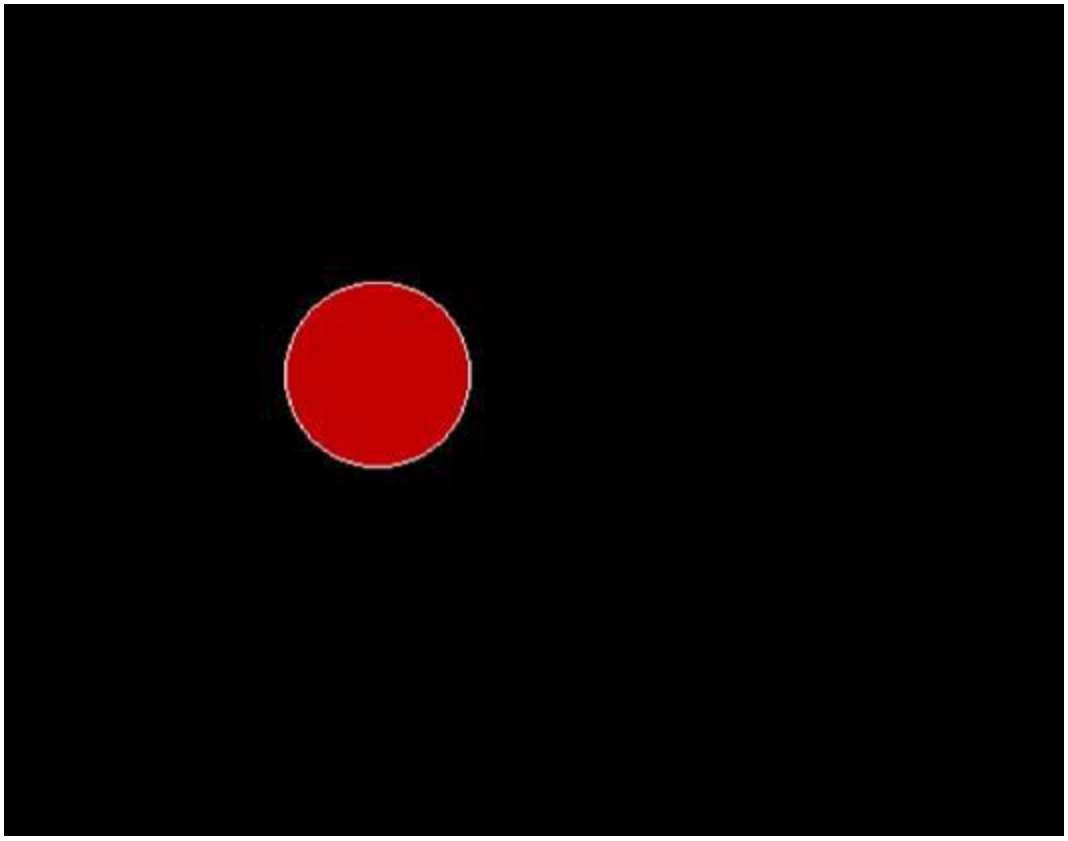
```
#include<stdio.h>
#include<graphics.h> #include<dos.h> void
boundaryfill(int x,inty,intf_color,intb_color)
{ if(getpixel(x,y)!=b_color&&getpixel(x,y)!=f_color)
{ putpixel(x,y,f_color);
boundaryfill(x+1,y,f_color,b_color); boundaryfill(x-
1,y,f_color,b_color);boundaryfill(x,y+1,f_color
,b_color); 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:\\turboc3\\bgi");
circle(x,y,radius); boundaryfill(x,y,4,15);
delay(5000); closegraph(); return 0; }
```

b. Flood fill algorithm

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<dos.h>
void flodfill(int x,inty,intf,int o)
{
if(getpixel(x,y)==o)
{ putpixel (x,y,f);
delay(10);
flodfill(x+1,y,f,o);
flodfill(x,y+1,f,o);
flodfill(x+1,y+1,f,o);
flodfill(x-1,y-1,f,o);
//flodfill(x-1,y,f,o);
//flodfill(x,y-1,f,o);
//flodfill(x-1,y+1,f,o);
// flodfill(x+1,y-1,f,o); } }
void main() { int
gd=DETECT,gm;
initgraph(&gd,&gm,"..\\bgi");
rectangle(50,50,100,100);
flodfill(51,51,4,0);
getch();}
```

Screenshot of output

A.



B.

Conclusion:

1. We Implemented Boundary fill algorithm and Flood fill algorithm.

Universal College of Engineering, Kaman
Department of Computer Engineering
Subject: Computer Graphics

Experiment No: 6

Roll No: 141	**Name: TANAY PATIL**	**Div: D**	**Batch: D3**

Aim: Implement 2D Transformations: Translation, Scaling, Rotation using switch case.

Program:

1. Program for translation:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
void main()
{
int gd=DETECT,gm;
int x1,y1,x2,y2,tx,ty,x3,y3,x4,y4; initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
printf("Enter the starting point of line segment:"); scanf("%d
%d",&x1,&y1); printf("Enter the ending point of line
segment:");scanf("%d %d",&x2,&y2); printf("Enter translation distances
tx,ty:\n"); scanf("%d%d",&tx,&ty);
setcolor(5); line(x1,y1,x2,y2);
outtextxy(x2+2,y2+2,"Original
line"); x3=x1+tx; y3=y1+ty;
x4=x2+tx; y4=y2+ty; setcolor(7);
line(x3,y3,x4,y4); outtextxy(x4+2,y4+2,"Line
after translation"); getch(); }
```

2. Program for scaling:

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

```
#include<graphics.h>#include<math.h>
void main()
{
int gd=DETECT,gm;
float x1,y1,x2,y2,sx,sy,x3,y3,x4,y4;
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
printf("Enter the starting point
coordinates:"); scanf("%f %f",&x1,&y1);
printf("Enter the ending point coordinates:");
scanf("%f %f",&x2,&y2); printf("Enter
scaling factors sx,sy:\n");
scanf("%f%f",&sx,&sy); setcolor(5);
line(x1,y1,x2,y2);
outtextxy(x2+2,y2+2,"Original line");
x3=x1*sx; y3=y1*sy; x4=x2*sx; y4=y2*sy;
setcolor(7); line(x3,y3,x4,y4);
outtextxy(x3+2,y3+2,"Line after scaling");
getch(); }
```

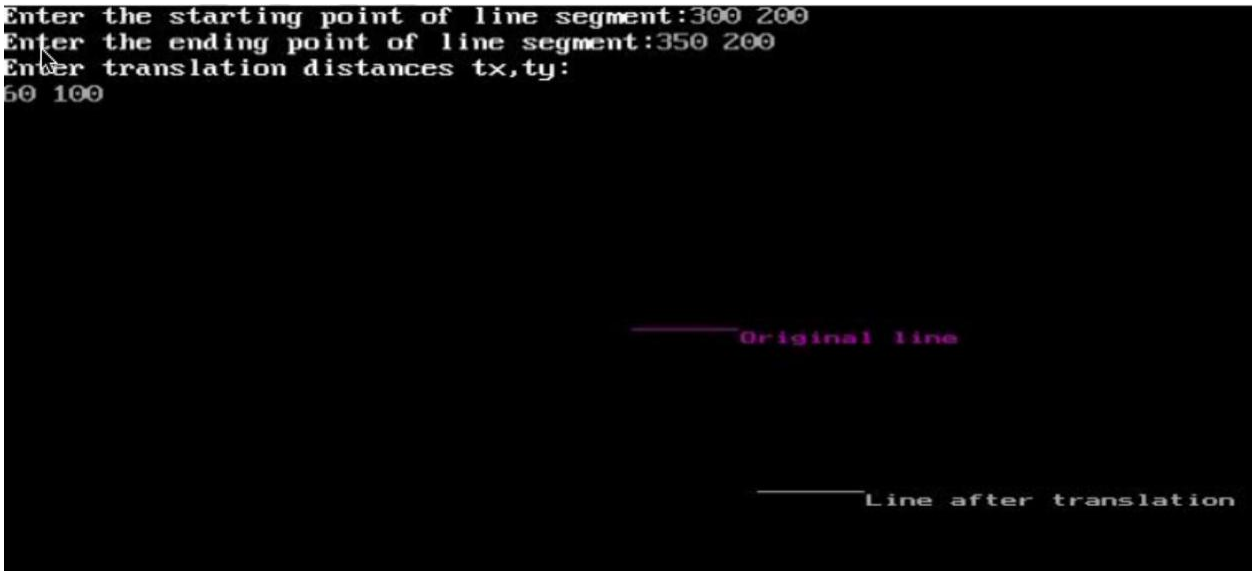
3. Program for Rotation:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h> void
main()
{
int gd=DETECT,gm;
float x1,y1,x2,y2,x3,y3,x4,y4,a,t;
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
printf("Enter coordinates of starting point:\n");
scanf("%f%f",&x1,&y1); printf("Enter
coordinates of ending point\n");
scanf("%f%f",&x2,&y2); printf("Enter angle for
rotation\n"); scanf("%f",&a); setcolor(5);
line(x1,y1,x2,y2); outtextxy(x2+2,y2+2,"Original
line");t=a*(3.14/180); x3=(x1*cos(t))-(y1*sin(t));
y3=(x1*sin(t))+(y1*cos(t)); x4=(x2*cos(t))-
(y2*sin(t)); y4=(x2*sin(t))+(y2*cos(t)); setcolor(7);
line(x3,y3,x4,y4); outtextxy(x3+2,y3+2,"Line after
rotation"); getch();
```

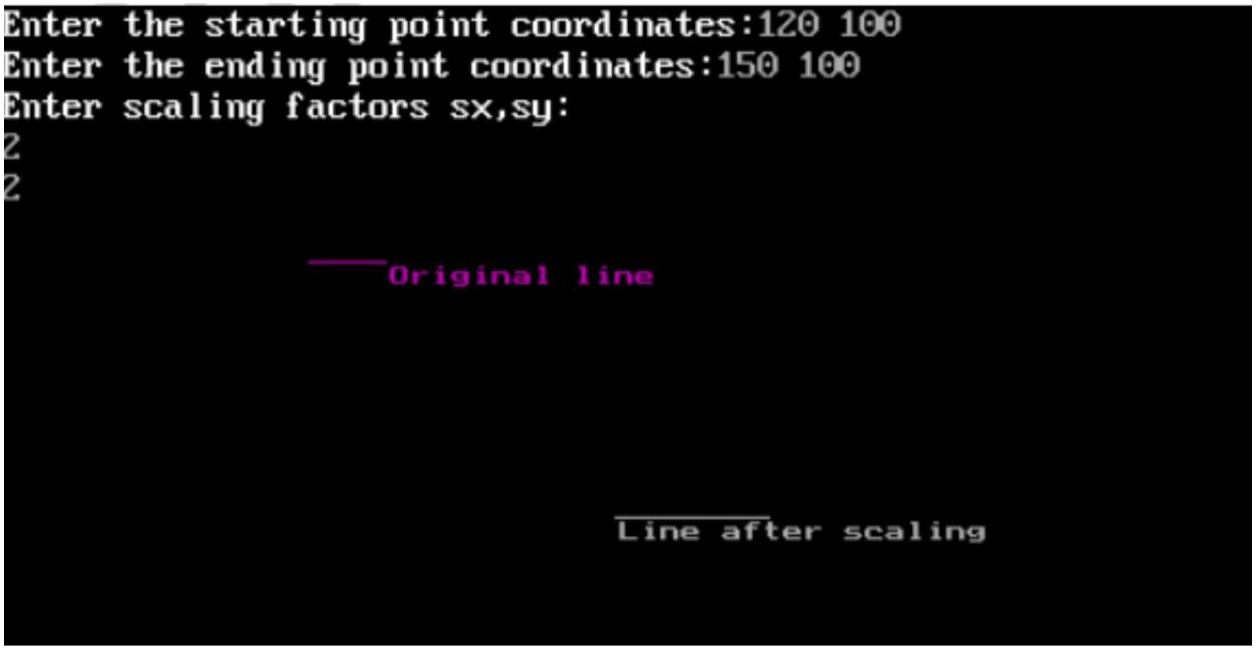
}

Screenshot of output

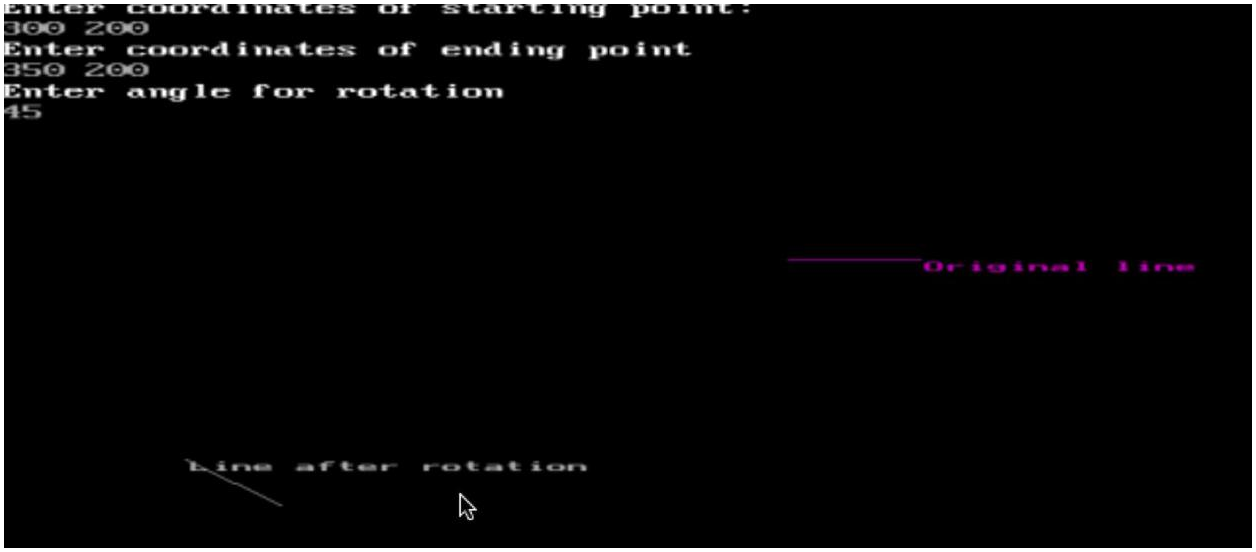
1. **Translation**



2.Scaling



3.Rotation



Conclusion:

1. we implemented 2D transformation: translation,scaling,rotation using c program.

Universal College of Engineering, Kaman
Department of Computer Engineering
Subject: Computer Graphics

Experiment No: 7

Roll No: 141	**Name:** TANAY PATIL	**Div: D**	**Batch: D1**

Aim: To implement Cohen Sutherland line clipping algorithm

Program:

```
#include<graphics.h>
#include<math.h>
#include<stdio.h>
#include<conio.h>
void bytecode();
void sutherland();
int a[5],b[5];
float m,xnew,ynew;
//float xl = 100,yl = 100, xh = 300, yh = 300,xa = 10,ya =
200,xb = 250, yb = 150;
//float
xl=100,yl=100,xh=300,yh=300,xa=170,ya=150,xb=250,yb=250; float
xl=90,yl=120,xh=300,yh=300,xa=50,ya=200,xb=200,yb=150; void
main()
{ intgd = DETECT,gm;
  initgraph(&gd,&gm,"C:\\TURBOC3\\BGI")
  ; setcolor(5); line(xa,ya,xb,yb);
  setcolor(12); rectangle(xl,yl,xh,yh);
  m = (yb-ya)/(xb-xa); bytecode();
  sutherland(); getch();
}
void bytecode()
{ if(xa <
  xl) a[3]
  = 1;
  else a[3] = 0;
```

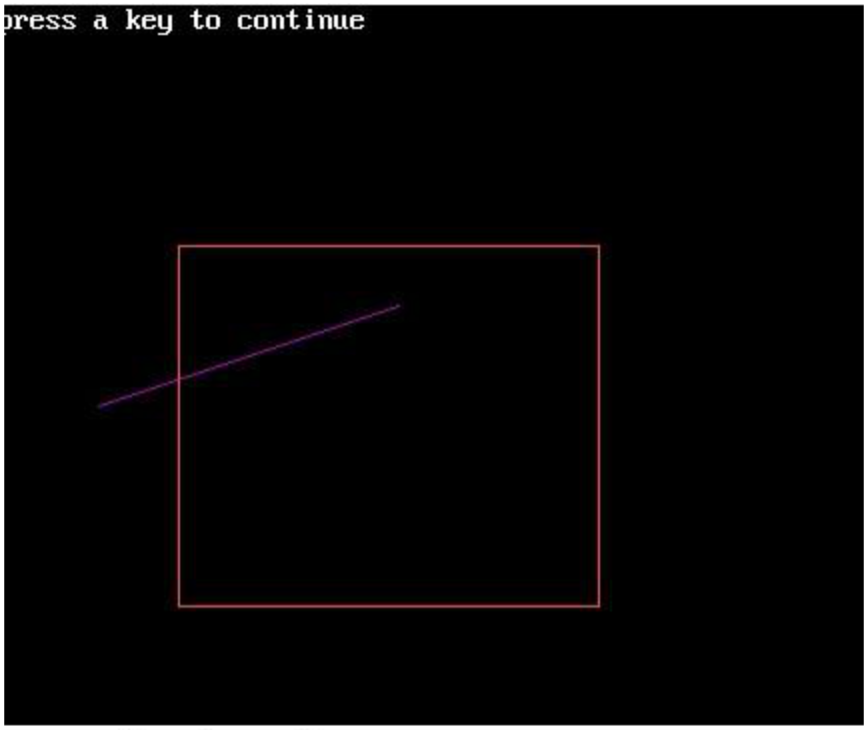
```
  if(xa>xh)
   a[2] = 1;
  else a[2] = 0;

  if(ya<yl) a[1]
   = 1;
  else a[1] = 0;

  if (ya>yh) a[0]
   = 1; else a[0]
   = 0;

  if(xb< xl) b[3]
    = 1;
  else b[3] = 0;

  if(xb>xh)
   b[2] = 1;
  else b[2] = 0;

  if(yb<yl) b[1]
   = 1;
  else b[1] = 0;

  if (yb>yh) b[0]
   = 1; else b[0]
   = 0;
}

void sutherland()
{ printf("press a key to continue");
  getch();
  if(a[0] == 0 && a[1] == 0 && a[2] == 0 && a[3] == 0 && b[0] ==
0 &&b[1] == 0 && b[2] == 0 && b[3] == 0 )
  {

   printf("no clipping");
   line(xa,ya,xb,yb);
```

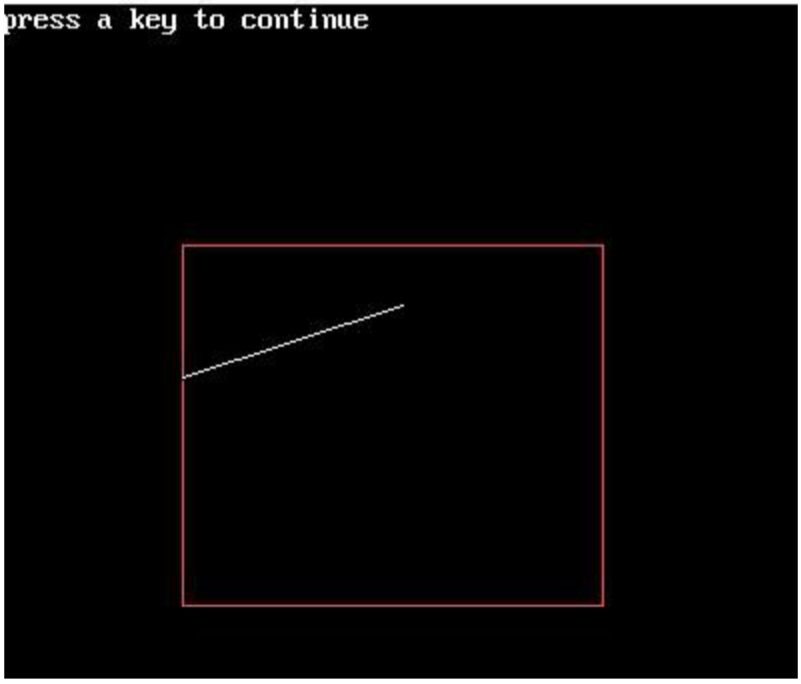
```
} else if(a[0]&&b[0] || a[1]&&b[1] || a[2]&&b[2] ||
a[3]&&b[3])
{ clrscr(); printf("line
 discarded");
 rectangle(xl,yl,xh,yh);
} else { if(a[3] == 1 &&
b[3]==0)
 {
  ynew = (m * (xl-xa)) + ya;
  setcolor(12);
  rectangle(xl,yl,xh,yh);
  setcolor(0);
  line(xa,ya,xb,yb);
  setcolor(15);
  line(xl,ynew,xb,yb);
 }
 else if(a[2] == 1 && b[2] == 0)
 {
  ynew = (m * (xh-xa)) + ya;
  setcolor(12);
  rectangle(xl,yl,xh,yh);
  setcolor(0);
  line(xa,ya,xb,yb);
  setcolor(15);
  line(xl,ynew,xb,yb);
 }
 else if(a[1] == 1 && b[1] == 0)
 { xnew = xa + (yl-
  ya)/m; setcolor(0);
  line(xa,ya,xb,yb);
  setcolor(15);
  line(xnew,yh,xb,yb)
  ;
 }

 else if(a[0] == 1 && b[0] == 0)
 { xnew = xa + (yh-
  ya)/m; setcolor(0);
  line(xa,ya,xb,yb);
  setcolor(15);
```

```
line(xnew,yh,xb,yb)
;
```

```
    }
  }
}
```

Screenshot of output





Conclusion:

1. we implemented Cohen Sutherland line clipping algorithm using c program.

Universal College of Engineering, Kaman
Department of Computer Engineering
Subject: Computer Graphics

Experiment No: 8

Roll No: 141	**Name: TANAY PATIL**	**Div: D**	**Batch: D3**

Aim: To implement 3D Transformation operations in C (only Translation & Scaling).

Program:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h> int
maxx,maxy,midx,midy;

void axis()
{ getch(); cleardevice();
 line(midx,0,midx,maxy)
 ;
 line(0,midy,maxx,midy);
} void
main()
{ int
```

```
gd,gm,x,y,z
,ang,x1,x2,
y1,y2;
detectgrap
h(&gd,&g
m);
initgraph(
&gd,&gm,
"C:/Turbo
c3/BGI");
 setfillstyle(3,25);
 maxx=getmaxx(); maxy=getmaxy();
 midx=maxx/2; midy=maxy/2;
 outtextxy(100,100,"ORIGINAL OBJECT");
 line(midx,0,midx,maxy); line(0,midy,maxx,midy);
 bar3d(midx+100,midy-20,midx+60,midy-90,20,5);
 axis();
 outtextxy(100,20,"TRANSLATION"); printf("\n\n Enter the
 Translation vector: "); scanf("%d%d",&x,&y);
 bar3d(midx+100,midy-20,midx+60,midy-90,20,5);
 bar3d(midx+(x+100),midy-(y+20),midx+(x+60),midy-(y+90),20,5);
 axis();
 outtextxy(100,20,"SCALING"); printf("\n Enter the Scaling Factor:
 "); scanf("%d%d%d",&x,&y,&z); bar3d(midx+100,midy-
```

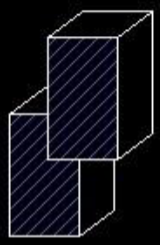
```
 20,midx+60,midy-90,20,5); bar3d(midx+(x*100),midy-
 (y*20),midx+(x*60),midy-(y*90),20*z,5); axis();

 outtextxy(100,20,"ROTATION"); printf("\n
 Enter the Rotation angle: "); scanf("%d",&ang);
 x1=100*cos(ang*3.14/180)-20*sin(ang*3.14/180);
 y1=100*sin(ang*3.14/180)+20*sin(ang*3.14/180);
 x2=60*cos(ang*3.14/180)-90*sin(ang*3.14/180);
 y2=60*sin(ang*3.14/180)+90*sin(ang*3.14/180);
 axis(); printf("\n After rotating about z-axis\n");
 bar3d(midx+100,midy-20,midx+60,midy-90,20,5);
 bar3d(midx+x1,midy-y1,midx+x2,midy-y2,20,5);
 axis(); printf("\n After rotating about x-axis\n");
 bar3d(midx+100,midy-20,midx+60,midy-90,20,5);
 bar3d(midx+100,midy-x1,midx+60,midy-x2,20,5);
 axis(); printf("\n After rotating about y-axis\n");
 bar3d(midx+100,midy-20,midx+60,midy-90,20,5);
 bar3d(midx+x1,midy-20,midx+x2,midy-90,20,5);
 axis();
 closegraph();
}
```

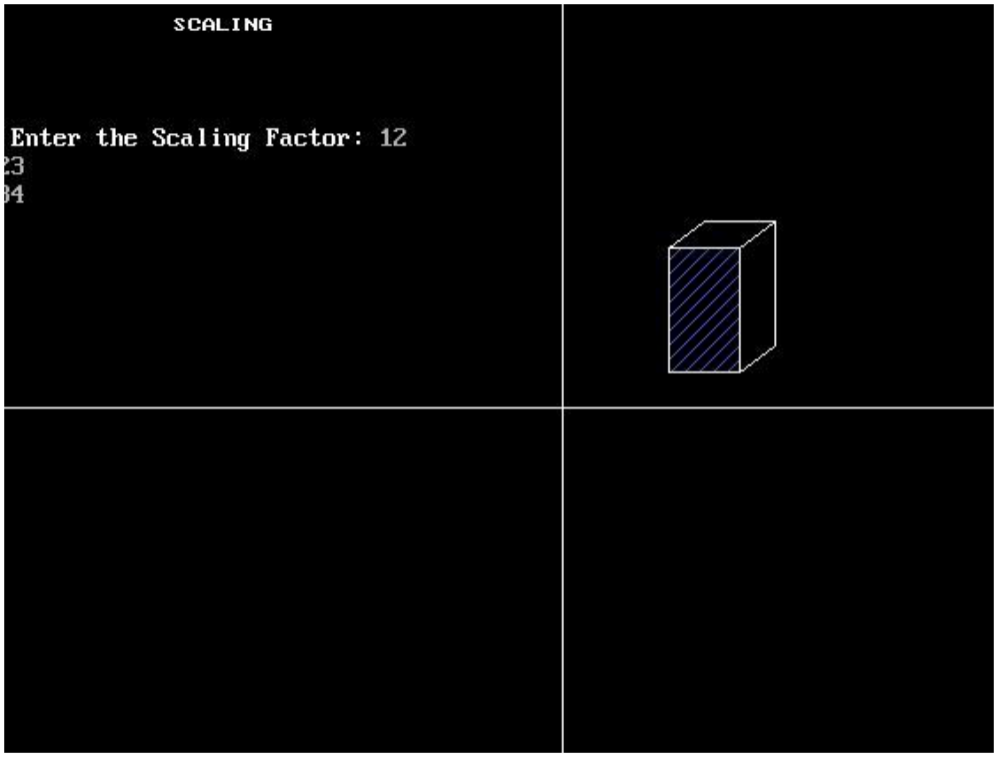
Screenshot of output

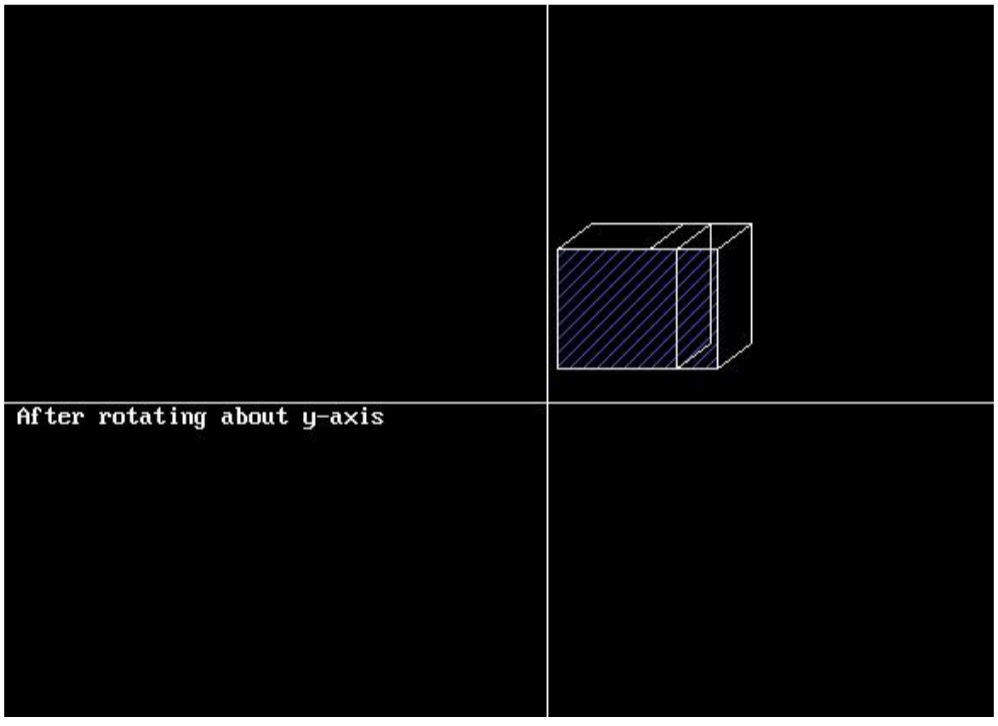
ORIGINAL OBJECT

TRANSLATION

Enter the Translation vector: 22
44







Conclusion:

We understood translation and scaling in 3D transformations.

Universal College of Engineering, Kaman
Department of Computer Engineering
Subject: Computer Graphics

Experiment No: 9

Roll No: 141	Name:TANAY PATIL	Div: D	Batch: D3

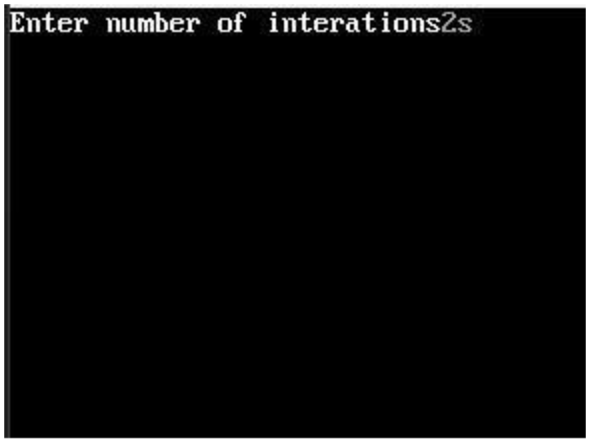
Aim: To implement Koch curve in C.

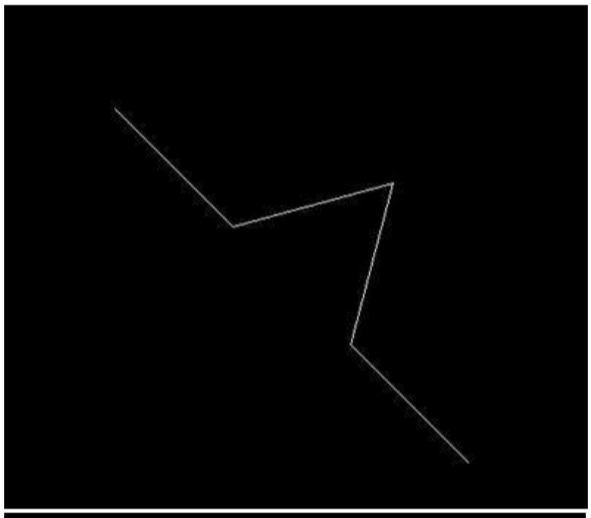
Program:

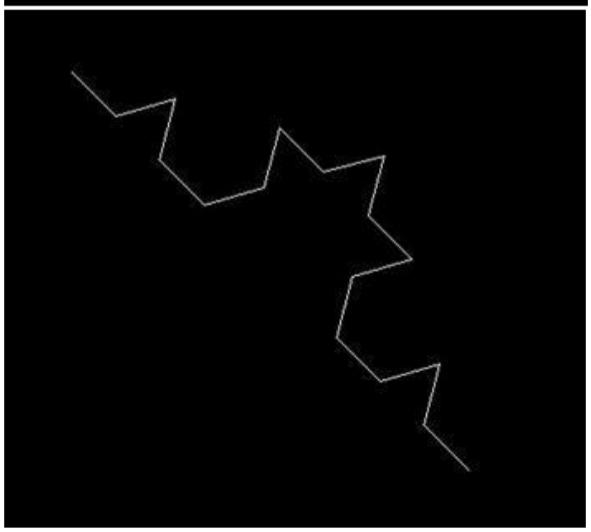
```
#include<graphics.h>

#include<conio.h>

#include<math.h>


void koch(int x1, int y1, int x2, int y2, int it)

{

float angle = 60*M_PI/180;

int x3 = (2*x1+x2)/3; int y3

= (2*y1+y2)/3;


int x4 = (x1+2*x2)/3; int

y4 = (y1+2*y2)/3;


int x = x3 + (x4-x3)*cos(angle)+(y4-y3)*sin(angle);

int y = y3 - (x4-x3)*sin(angle)+(y4-y3)*cos(angle);
```

```
if(it > 0) { koch(x1, y1, x3, y3, it-1); koch(x3, y3, x, y, it-
1); koch(x, y, x4, y4, it-1); koch(x4, y4, x2, y2, it-1);
} else
{

  line(x1, y1, x3, y3);
  line(x3, y3, x, y);
  line(x, y, x4, y4);
  line(x4, y4, x2, y2);
}
}

int main(void)
{
int gd = DETECT, gm;

int x1 = 100, y1 = 100, x2 = 400, y2 = 400;
initgraph(&gd, &gm, "C:\\Turboc3\\BGI"); koch(x1,
y1, x2, y2, 4);
getch(); return
0;
}
```

Screenshot of output







Conclusion:

We understood and implemented koch curve in c.

Universal College of Engineering, Kaman

Department of Computer Engineering

Subject: Computer Graphics

Roll No:141 **Name:** TANAY PATIL **Div: D** **Batch: D3**

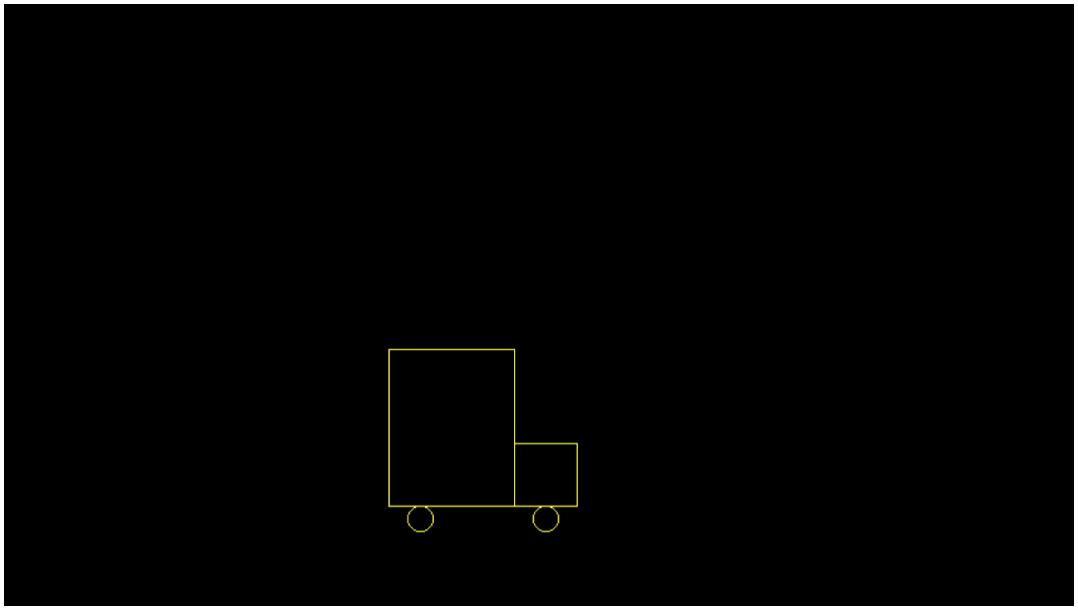
Aim: **Program to perform Animation on Moving Vehicle.**

Theory:

```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
#include<dos.h>
void main()
{
int gd=DETECT,gm;
int i ,j=0;
initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
settextstyle(GOTHIC_FONT,HORIZ_DIR,4);
getch();
i=0;
while(i<=420)
{
rectangle(150+i,350,200+i,400);
rectangle(50+i,275,150+i,400);
circle(175+i,410,10);
circle(75+i,410,10);
setcolor(j++);
delay(100);
i=i+10;
```

```
cleardevice();
}
getch();
closegraph();
}
```

Output:



Universal College of Engineering, Kaman

Department of Computer Engineering

Subject: Computer Graphics

Roll No:141 **Name:** TANAY PATIL **Div: D** **Batch: D3**

Aim: To implement Bezier Curve

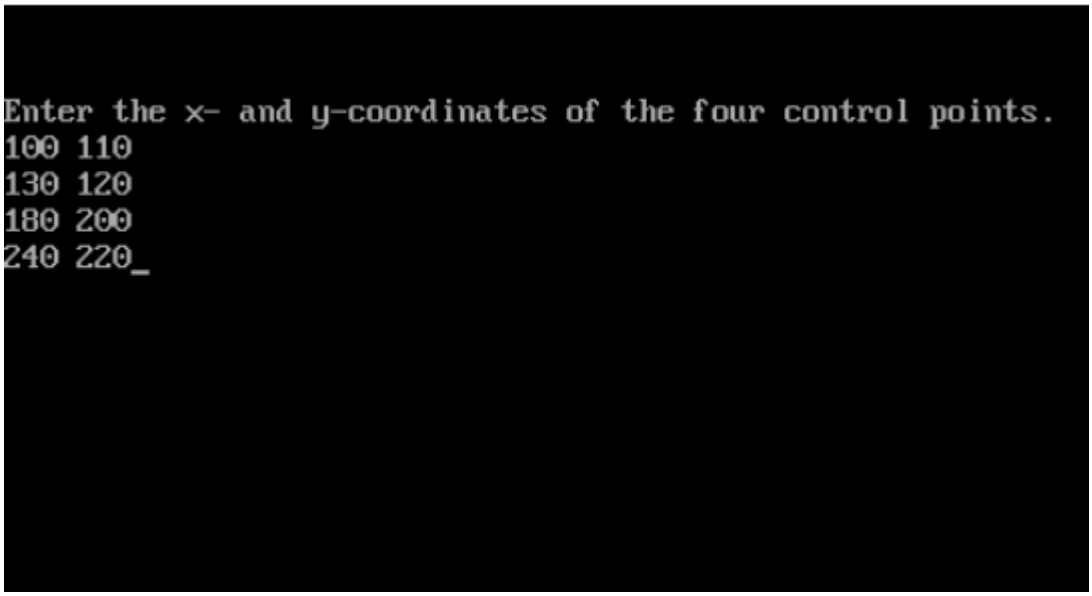
Theory:

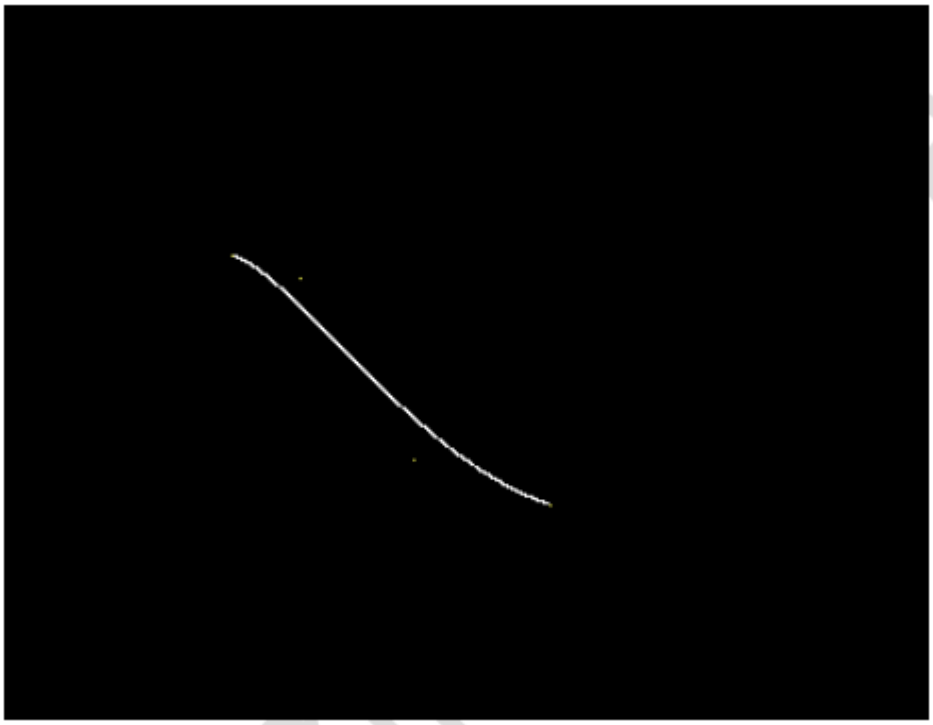
Program:

```
#include <stdio.h>
#include <stdlib.h>
#include <graphics.h>
#include <math.h>
void bezier (int x[4], int y[4])
{
 int gd = DETECT, gm;
 int i;
 double t;
 initgraph (&gd, &gm, "C:\\TurboC3\\BGI");
 for (t = 0.0; t < 1.0; t += 0.0005)
 {
double xt = pow (1-t, 3) * x[0] + 3 * t * pow (1-t, 2) * x[1] + 3 * pow (t, 2) * (1-t) * x[2] + pow (t, 3) * x[3];
double yt = pow (1-t, 3) * y[0] + 3 * t * pow (1-t, 2) * y[1] + 3 * pow (t, 2) * (1-t) * y[2] + pow (t, 3) *y[3];
putpixel (xt, yt, WHITE);
 }
 for (i=0; i<4; i++)
```

```
 putpixel (x[i], y[i], 5);
 getch();
 closegraph();
 return;
}
void main()
{
 int x[4], y[4];
 int i;
 printf ("Enter the x- and y-coordinates of the four control points.\n");
 for (i=0; i<4; i++)
 scanf ("%d%d", &x[i], &y[i]);
 bezier (x, y);
}
```

Output:





Conclusion: Implemented Bezier Curve successfully.