Problem Statement:

1. Write a C program to implement Bresenham line drawing algorithm.

Source Code:

```
#include<stdio.h>

#include<conio.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; }
}
main()
{
int gdriver=DETECT, gmode, error, x0, y0, x1, y1;
initgraph(&gdriver, &gmode, "c:\\turboc3\\bgi");
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();
}
```

Output:

```
Enter co-ordinates of first point: 100 100

Enter co-ordinates of second point: 300 300
```

Problem Statement:

2.Write a C program to implement DDA line drawing algorithm.

Source Code:

```
#include<graphics.h>

#include<conio.h>

#include<stdio.h>

main()

{

   float x,y,dx,dy,steps;

   int gdriver=DETECT, gmode, error, x0, y0, x1, y1, i;

   initgraph(&gdriver, &gmode, "c:\\turboc3\\bgi");

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

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

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

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

   dx = (float)(x1 - x0);

   dy = (float)(y1 - y0);

   if(dx>=dy)
```

```
       {
    steps = dx;
}
else
       {
    steps = dy;
}
dx = dx/steps;
dy = dy/steps;
x = x0;
y = y0;
i = 1;
while(i<= steps)
{
    putpixel(x, y, RED);
    x += dx;
    y += dy;
    i=i+1;
}
getch();
```

```
    closegraph();
}
```

Output:

```
Enter co-ordinates of first point: 100 200
Enter co-ordinates of second point: 600 200
```

Problem Statement:

3. Write a C program to implement Mid-Point line drawing algorithm.

Source Code:

```
#include <stdio.h>

#include <conio.h>

#include <graphics.h>

#include <math.h>

#include <dos.h>

void lineMidPoint(int x1, int y1, int x2, int y2){

    int dx = x2 - x1;

    int dy = y2 - y1;

    int d = 2 * dy - dx;

    int incrE = 2 * dy;

    int incrNE = 2 * (dy - dx);

    int x = x1;

    int y = y1;

    putpixel(x, y, WHITE);

    while(x < x2){
        if(d<=0){
            d += incrE;
            x++;
            }
        else{
            d += incrNE;
            x++;
            y++;
            }
        putpixel(x, y, WHITE);
        }
    }
int main(){
    int gd=DETECT, gm;
    initgraph(&gd, &gm, "\\tc");
    int x0,y0,x1,y1;
    printf("Enter co-ordinates of first point: ");
    scanf("%d%d", &x0, &y0);
    printf("Enter co-ordinates of second point: ");
    scanf("%d%d", &x1, &y1);
    lineMidPoint(x0, y0, x1, y1);
    getch();
    return 0;
}
```

Output:

```
Enter co-ordinates of first point: 100 200
Enter co-ordinates of second point: 500 300
```

Problem Statement:

4. Write a C program to implement Bresenham circle drawing algorithm.

Source Code:

```
#include <graphics.h>

#include <stdlib.h>

#include <stdio.h>

#include <conio.h>

#include <math.h>

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

  {

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

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

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

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

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

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

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

   putpixel(-y+xc,x+yc,WHITE);
```

```
}


 void BresenhamCircle(int xc,int yc,int r)
{
 int x=0,y=r,d=3-(2*r);
 EightWaySymmetricPlot(xc,yc,x,y);


 while(x<=y)
  {
   if(d<=0)
          {
     d=d+(4*x)+6;
    }
  else
   {
     d=d+(4*x)-(4*y)+10;
     y=y-1;
    }
    x=x+1;
    EightWaySymmetricPlot(xc,yc,x,y);
```

```
      }
    }
    int  main(void)
   {
    int xc,yc,r,gdriver = DETECT, gmode, errorcode;
     initgraph(&gdriver, &gmode, "C:\\TURBOC3\\BGI");
     errorcode = graphresult();
      if (errorcode != grOk)
     {
        printf("Graphics error: %s\n",
grapherrormsg(errorcode));
        printf("Press any key to halt:");
        getch();
        exit(1);
     }
       printf("Enter the centre points of the circle xc and
yc :");
       scanf("%d%d",&xc,&yc);
       printf("Enter the value of radius :");
       scanf("%d",&r);
       BresenhamCircle(xc,yc,r);
    getch();
    closegraph();
    return 0;
   }
```

Output:

```
Enter the centre points of the circle xc and yc :250 200
Enter the value of radius :60
```

Problem Statement:

5.Write a C program to implement Mid-Point circle drawing algorithm.

Source Code:

```
#include<stdio.h>

#include<graphics.h>

#include<conio.h>

void drawcircle(int x0, int y0, int radius)

{

   int x = radius;

   int y = 0;

   int err = 0;

   while (x >= y)

   {

putpixel(x0 + x, y0 + y, 7);

putpixel(x0 + y, y0 + x, 7);

putpixel(x0 - y, y0 + x, 7);

putpixel(x0 - x, y0 + y, 7);

putpixel(x0 - x, y0 - y, 7);
putpixel(x0 - y, y0 - x, 7);
putpixel(x0 + y, y0 - x, 7);
putpixel(x0 + x, y0 - y, 7);
if (err <= 0){
    y += 1;
    err += 2*y + 1;}
if (err > 0){
    x -= 1;
err -= 2*x + 1;}}}
int main()
{
int gdriver=DETECT, gmode, error, x, y, r;
initgraph(&gdriver, &gmode, "c:\\turboc3\\bgi");
printf("Enter radius of circle: ");
scanf("%d", &r);
printf("Enter co-ordinates of centre (x and y): ");
scanf("%d%d", &x, &y);
drawcircle(x, y, r);
getch();
return 0;
```

```
}
```

Output:

```
Enter radius of circle: 100 100

Enter co-ordinates of centre (x and y): 200
```

Problem Statement:

6.Write a C program to implement 2D shearing in X-axis.

Source Code:

```
#include<stdio.h>

#include<graphics.h>

#include<conio.h>

 main()

{

int gd=DETECT,gm;

int x,y,x1,y1,x2,y2,shear_f;

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

printf("\n please enter first coordinate = ");

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

printf("\n please enter second coordinate = ");

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

printf("\n please enter third coordinate = ");

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

printf("\n please enter shearing factor x = ");
scanf("%d",&shear_f);
cleardevice();
line(x,y,x1,y1);
line(x1,y1,x2,y2);
line(x2,y2,x,y);
setcolor(RED);
x=x+ y*shear_f;
x1=x1+ y1*shear_f;
x2=x2+ y2*shear_f;
line(x,y,x1,y1);
line(x1,y1,x2,y2);
line(x2,y2,x,y);
getch();
closegraph();
}
```

Output:

```
 please enter first coordinate = 30 50
 please enter second coordinate = 70 80
 please enter third coordinate = 100 130
 please enter shearing factor x = 2
```

Problem Statement:

7. Write a C program to implement 2D shearing in Y-axis.

Source Code:

```
#include<stdio.h>

#include<graphics.h>

#include<conio.h>

main()

{

int gd=DETECT,gm;

int x,y,x1,y1,x2,y2,shear_f;

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

printf("\n please enter first coordinate = ");

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

printf("\n please enter second coordinate = ");

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

printf("\n please enter third coordinate = ");

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

printf("\n please enter shearing factor x = ");
scanf("%d",&shear_f);
cleardevice();
line(x,y,x1,y1);
line(x1,y1,x2,y2);
line(x2,y2,x,y);
setcolor(RED);
y=y+ x*shear_f;
y1=y1+ x1*shear_f;
y2=y2+ x2*shear_f;
line(x,y,x1,y1);
line(x1,y1,x2,y2);
line(x2,y2,x,y);
getch();
closegraph();
}
```

Output:

```
please enter first coordinate = 30 60
 please enter second coordinate = 70 90
 please enter third coordinate = 100 130
 please enter shearing factor x = 3
```

Problem Statement:

8. Write a C program to implement 2D reflection in X-axis.

Source Code:

```
#include<stdio.h>

#include<conio.h>

#include<graphics.h>

#include<stdlib.h>

int main()

{

   int gd=DETECT,gm;

   int x1,y1,x2,y2,x3,y3;

   char a;

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

   printf("\n Enter the coordinates of triangle
x1,y1,x2,y2,x3,y3 ");

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);
      x1=x1;
      x2=x2;
      x3=x3;
      y1=y1+240;
      y2=y2+240;
      y3=y3+240;
   line(x1,y1,x2,y2);
   line(x2,y2,x3,y3);
   line(x3,y3,x1,y1);
   getch();
   closegraph();
}
```

Output:

```
 Enter the coordinates of triangle x1,y1,x2,y2,x3,y3 20
50 70 100 120 200
```

Problem Statement:

9. Write a C program to implement 2D reflection in Y-axis.

Source Code:

```
#include<stdio.h>

#include<conio.h>

#include<graphics.h>

#include<stdlib.h>

int main()

{

   int gd=DETECT,gm;

   int x1,y1,x2,y2,x3,y3;

   char a;

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

   printf("\n Enter the coordinates of triangle
x1,y1,x2,y2,x3,y3 ");

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);
      y1=y1;
      y2=y2;
      y3=y3;
      x1+=320;
      x2+=320;
      x3+=320;
   line(x1,y1,x2,y2);
   line(x2,y2,x3,y3);
   line(x3,y3,x1,y1);
   getch();
   closegraph();
}
```

Output:

```
 Enter the coordinates of triangle x1,y1,x2,y2,x3,y3 20
50 70 100 120 250
```

Problem Statement:

10. Write a C program to implement 2D scaling.

Source Code:

```
#include<stdio.h>
#include<graphics.h>
void findNewCoordinate(int s[][2], int p[][1])
{
	int temp[2][1] = { 0 };
	for (int i = 0; i < 2; i++)
		for (int j = 0; j < 1; j++)
			for (int k = 0; k < 2; k++)
				temp[i][j] += (s[i][k] * p[k][j]);
	p[0][0] = temp[0][0];
	p[1][0] = temp[1][0];
}
void scale(int x[], int y[], int sx, int sy)
{
	line(x[0], y[0], x[1], y[1]);
	line(x[1], y[1], x[2], y[2]);
```

```
    line(x[2], y[2], x[0], y[0]);
    int s[2][2] = { sx, 0, 0, sy };
    int p[2][1];
    for (int i = 0; i < 3; i++)
    {
        p[0][0] = x[i];
        p[1][0] = y[i];
        findNewCoordinate(s, p);
        x[i] = p[0][0];
        y[i] = p[1][0];
    }
    line(x[0], y[0], x[1], y[1]);
    line(x[1], y[1], x[2], y[2]);
    line(x[2], y[2], x[0], y[0]);
}
int main()
{
    int x[] = { 100, 200, 300 };
    int y[] = { 200, 100, 200 };
    int sx, sy;
```

```
        printf("Enter scaling factor sx & sy: ");
        scanf("%d%d", &sx, &sy);
        int gd, gm;
        detectgraph(&gd, &gm);
        initgraph(&gd, &gm," c:\\turboc3\\bgi ");
        scale(x, y, sx,sy);
        getch();
        return 0;
}
```

Output:

```
Enter scaling factor sx & sy: 4 4
```

Problem Statement:

11. Write a C program to implement 2D rotation.

Source Code:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<process.h>
#include<math.h>
void TriAngle(int x1, int y1, int x2, int y2, int x3, int y3);
void Rotate(int x1, int y1, int x2, int y2, int x3, int y3);
 main() {
   int gd = DETECT, gm;
   int x1, y1, x2, y2, x3, y3;
   initgraph(&gd, &gm, "c:\\turboc3\\bgi ");
   printf("Enter the 1st point for the triangle:");
   scanf("%d%d", &x1, &y1);
   printf("Enter the 2nd point for the triangle:");
   scanf("%d%d", &x2, &y2);
   printf("Enter the 3rd point for the triangle:");
```

```
    scanf("%d%d", &x3, &y3);
    TriAngle(x1, y1, x2, y2, x3, y3);
    Rotate(x1, y1, x2, y2, x3, y3);
    setcolor(1);
    TriAngle(x1, y1, x2, y2, x3, y3);
    getch();
}
void TriAngle(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 Rotate(int x1, int y1, int x2, int y2, int x3, int y3) {
    int x, y, a1, b1, a2, b2, a3, b3, p = x2, q = y2;
    float Angle;
    printf("Enter the angle for rotation:");
    scanf("%f", &Angle);
    cleardevice();
    Angle = (Angle * 3.14) / 180;
    a1 = p + (x1 - p) * cos(Angle)-(y1 - q) * sin(Angle);
```

```
  b1 = q + (x1 - p) * sin(Angle)+(y1 - q) * cos(Angle);
  a2 = p + (x2 - p) * cos(Angle)-(y2 - q) * sin(Angle);
  b2 = q + (x2 - p) * sin(Angle)+(y2 - q) * cos(Angle);
  a3 = p + (x3 - p) * cos(Angle)-(y3 - q) * sin(Angle);
  b3 = q + (x3 - p) * sin(Angle)+(y3 - q) * cos(Angle);
  TriAngle(a1, b1, a2, b2, a3, b3);
}
```

Output:

```
Enter the 1st point for the triangle:50 50
Enter the 2nd point for the triangle:90 90
Enter the 3rd point for the triangle:100 140
Enter the angle for rotation:60
```

Problem Statement:

12.Write a C program to implement 2D translation.

Source Code:

```
#include<conio.h>
#include<graphics.h>
#include<stdio.h>
 main()
{
int gd=DETECT,gm;
int l[2][2],v[2]={10,15},i=0,j;
initgraph(&gd,&gm," c:\\turboc3\\bgi ");
printf("Enter the initial and final coordinates of a line
");
while(i<2)
{
printf("x%d and y%d = ",i,i);
j=0;
scanf("%d",&l[i][j]);
scanf("%d",&l[i][j+1]);
i++;
}
line(l[0][0],l[0][1],l[1][0],l[1][1]);
setcolor(YELLOW);
line(l[0][0]+v[0],l[0][1]+v[1],l[1][0]+v[0],l[1][1]+v[1]);
getch();
closegraph();
}
```

Output:

```
Enter the initial and final coordinates of a line x0 and
y0 = 200 200
x1 and y1 = 60 60
```