CGM ASSIGNMENT

(DIKSHANT AGRAWAL-189301074)

1. Draw a Circle using MidPoint circle algorithm and Bresenham's Circle algorithm.

```
#include <stdio.h>
#include <graphics.h>
void circles(int x,int y,int r)
{
   int X=0,Y=r;
   putpixel(X+x,Y+y,WHITE);
   int p=1-r;
   while(X<=Y)
   {
      X++;
      if(p<=0)
         p=p+2*X+1;
      else
      {
         Y--;
         p=p+2*X-2*Y+1;
      }
      putpixel(X+x,Y+y,WHITE);
      putpixel(Y+x,X+y,WHITE);
      putpixel(Y+x,-X+y,WHITE);
      putpixel(X+x,-Y+y,WHITE);
      putpixel(-X+x,-Y+y,WHITE);
      putpixel(-Y+x,-X+y,WHITE);
      putpixel(-Y+x,X+y,WHITE);
      putpixel(-X+x,Y+y,WHITE);
```

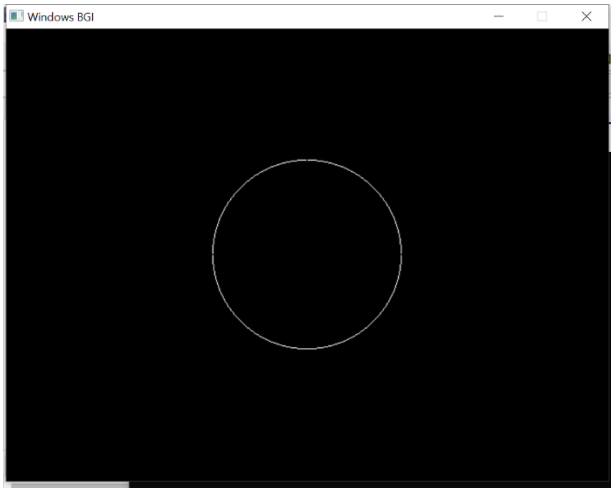
```
    }
}
int main()
{
    int gd = DETECT, gm;
    initgraph(&gd, &gm, NULL);
    int x=getmaxx()/2,y=getmaxy()/2,r=100;
    circles(x,y,r);
    getch();
    closegraph();
}
```

BRESENHAM'S CIRCLE

```
#include <stdio.h>
#include <graphics.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 values of 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;
}
```



2. Draw an Ellipse for Region 1 and Region 2 using Midpoint Ellipse Algorithm

```
#include<graphics.h>
#include <stdio.h>
    int main(){
     long x,y,x_center,y_center;
     long a_sqr,b_sqr, fx,fy, d,a,b,tmp1,tmp2;
     int g_driver=DETECT,g_mode;

   initgraph(&g_driver,&g_mode,"C:\\NULL\\BGI");
   printf("********* MID POINT ELLIPSE ALGORITHM *********");
   printf("\n\n Enter coordinate x and y = ");
   scanf("%ld%ld",&x_center,&y_center);
   printf("\n Now enter constants a and b = ");
   scanf("%ld%ld",&a,&b);
   x=0;
   y=b;
   a_sqr=a*a;
   b_sqr=b*b;
   fx=2*b_sqr*x;
   fy=2*a_sqr*y;
  d=b_sqr-(a_sqr*b)+(a_sqr*0.25);
  do
   {
  putpixel(x_center+x,y_center+y,1);
  putpixel(x_center-x,y_center-y,1);
  putpixel(x_center+x,y_center-y,1);
```

```
putpixel(x_center-x,y_center+y,1);

 if(d<0)
  {
d=d+fx+b_sqr;
  }
 else
{
y=y-1;
d=d+fx+-fy+b_sqr;
fy=fy-(2*a_sqr);
}
x=x+1;
fx=fx+(2*b_sqr);
delay(10);

 }
 while(fx<fy);
 tmp1=(x+0.5)*(x+0.5);
 tmp2=(y-1)*(y-1);
 d=b_sqr*tmp1+a_sqr*tmp2-(a_sqr*b_sqr);
 do
 {
putpixel(x_center+x,y_center+y,1);
putpixel(x_center-x,y_center-y,1);
putpixel(x_center+x,y_center-y,1);
putpixel(x_center-x,y_center+y,1);

 if(d>=0)
d=d-fy+a_sqr;
 else
```

```
{
x=x+1;
d=d+fx-fy+a_sqr;
fx=fx+(2*b_sqr);
}
 y=y-1;
 fy=fy-(2*a_sqr);
 }
 while(y>0);
 getch();
 closegraph();
}
```