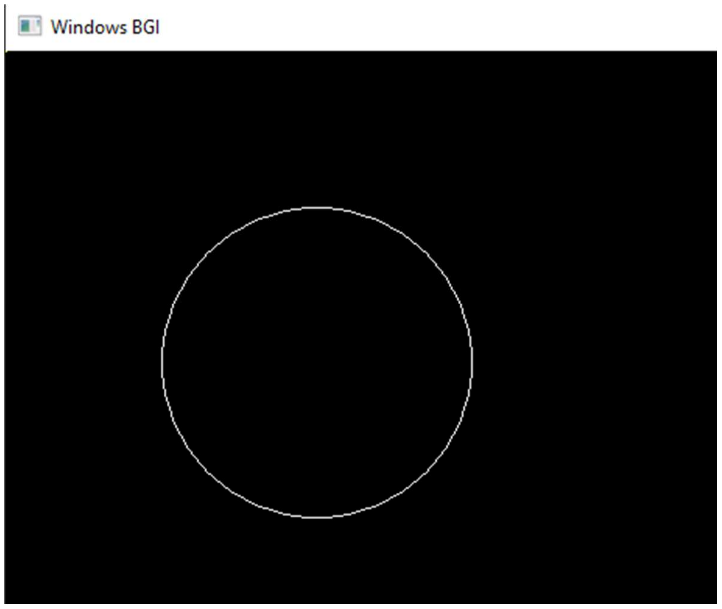
1. Write a C++ program to draw a circle.

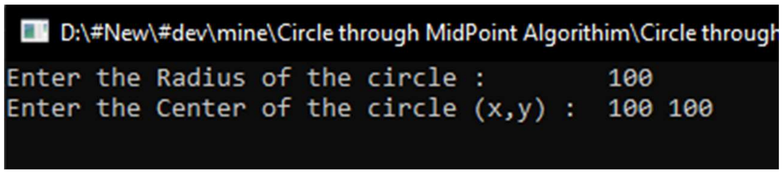
```
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
int main(void)
{
  /* request auto detection */
  int gdriver = DETECT, gmode, errorcode;
  int midx, midy;
  /* initialize graphics and local variables */
  initgraph(&gdriver, &gmode, "C:\\TURBOC3\\BGI");
  /* read result of initialization */
  errorcode = graphresult();
  if (errorcode != grOk)  /* an error occurred */
  {
    printf("Graphics error: %s\n", grapherrormsg(errorcode));
    printf("Press any key to halt:");
    getch();
    exit(1); /* terminate with an error code */
  }
  setcolor(getmaxcolor());
  /* draw the circle */
  circle(200,200, 100);
  /* clean up */
  getch();
  closegraph();
  return 0;
```



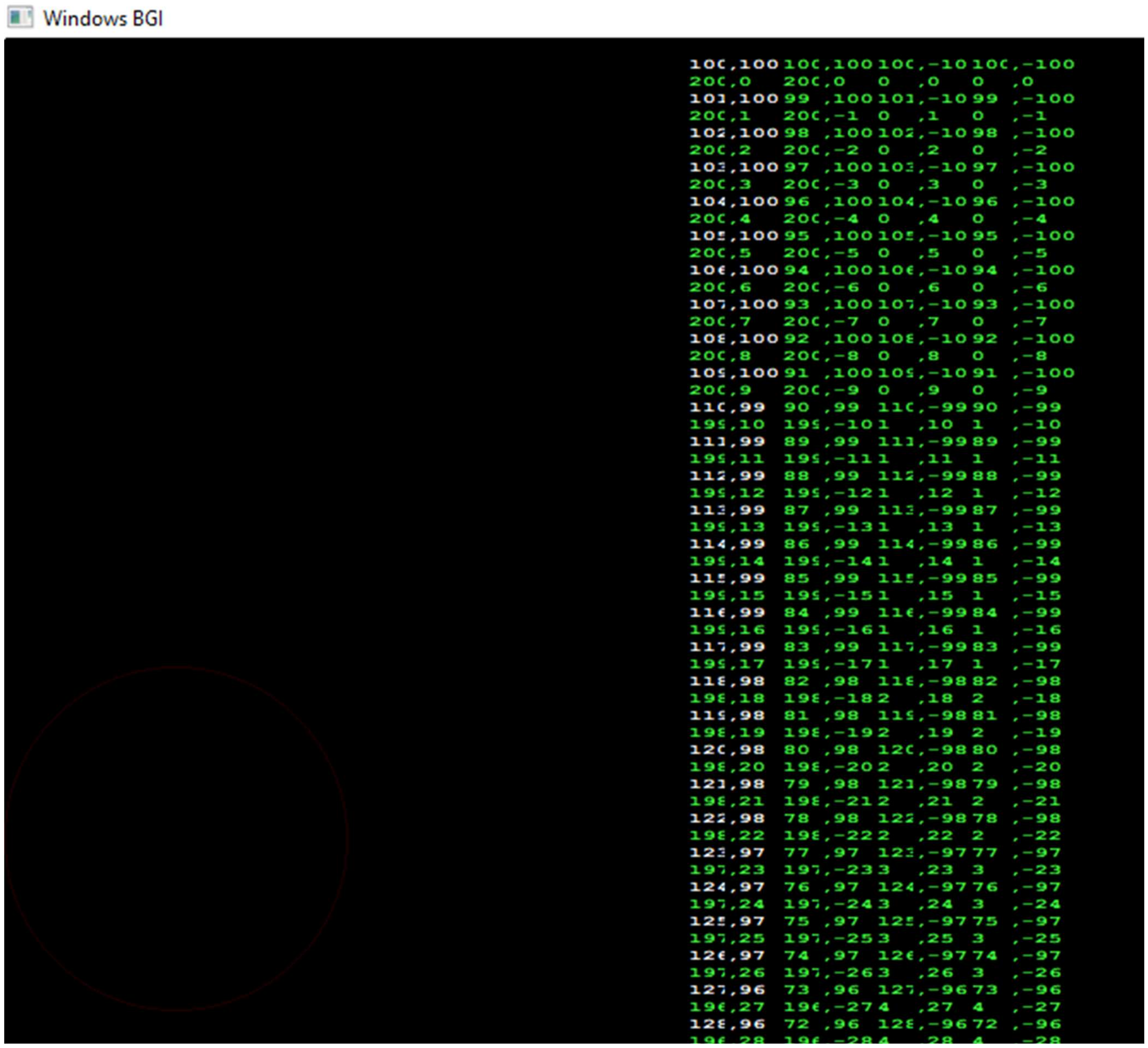
2. Circle through Midpoint Algorithm

```
#include <stdlib.h>
#include <graphics.h>
#include <math.h>
#include <conio.h>
#include <iostream>
using namespace std;
int xc, yc, showat=10;
char ruf[10];
void tellpoint(int *num1, int *num2)
{
  int color; color=getcolor();
  int a,b;
  a = xc+*num1; b = getmaxy()-yc+*num2;
  outtextxy(400,showat,itoa(a,ruf,10));  outtextxy(422,showat,",");  outtextxy(429,showat,itoa(b,ruf,10));
  a=xc-*num1; b=getmaxy()-yc+*num2; setcolor(10);
  outtextxy(455,showat,itoa(a,ruf,10));  outtextxy(477,showat,",");  outtextxy(484,showat,itoa(b,ruf,10));
  a=xc+*num1; b=getmaxy()-yc-*num2; setcolor(20);
  outtextxy(510,showat,itoa(a,ruf,10));  outtextxy(532,showat,",");  outtextxy(539,showat,itoa(b,ruf,10));
  a=xc-*num1; b=getmaxy()-yc-*num2; setcolor(30);


 outtextxy(565,showat,itoa(a,ruf,10));  outtextxy(587,showat,",");  outtextxy(594,showat,itoa(b,ruf,10));
  showat+=10;
  a = xc+*num2; b = getmaxy()-yc+*num1; setcolor(110);


  outtextxy(400,showat,itoa(a,ruf,10));  outtextxy(422,showat,",");  outtextxy(429,showat,itoa(b,ruf,10));
  a=xc+*num2; b=getmaxy()-yc-*num1; setcolor(70);
  outtextxy(455,showat,itoa(a,ruf,10));  outtextxy(477,showat,",");  outtextxy(484,showat,itoa(b,ruf,10));
  a=xc-*num2; b=getmaxy()-yc+*num1; setcolor(140);
  outtextxy(510,showat,itoa(a,ruf,10));  outtextxy(532,showat,",");  outtextxy(539,showat,itoa(b,ruf,10));
  a=xc-*num2; b=getmaxy()-yc-*num1; setcolor(150);
  outtextxy(565,showat,itoa(a,ruf,10));  outtextxy(587,showat,",");  outtextxy(594,showat,itoa(b,ruf,10));
  setcolor(color); showat+=10;
```

```
}
void drawpoint(int x, int y)
{
  putpixel (xc+x, yc+y, 30);        putpixel (xc-x, yc+y, 30);       putpixel (xc+x, yc-y, 30);
  putpixel (xc-x, yc-y, 30);        putpixel (xc+y, yc+x, 30);       putpixel (xc-y, yc+x, 30);
  putpixel (xc+y, yc-x, 30);        putpixel (xc-y, yc-x, 30);
}
int main()
{
            int gdriver=DETECT, gmode, ecode;
            initgraph(&gdriver, &gmode, "c:\\Turboc3\\bgi");
            ecode = graphresult();
            if (ecode != grOk)
    {
               cout << "Graphic error ...";
        cout << "Press any key ...";
                     getch();
                     exit(1);
            }
int x,y,r, Pk;
char comma;
cout <<"Enter the Radius of the circle :\t";
cin >> r;
cout << "Enter the Center of the circle (x,y) :\t";
cin >>xc >>comma >>yc;
yc = getmaxy()-yc;
x=0;
y=r;
drawpoint(x,y);
tellpoint(&x, &y);
Pk = 1 - r;
while (x<y)
 {
       if (Pk<0)
       x +=1;
              else
       {
            x +=1;
                            y -=1;
            }
             drawpoint(x,y);
             tellpoint(&x, &y);



if (Pk<0)
 Pk = Pk + 2*x +1;
 else
     Pk = Pk + 2*(x-y) +1;


}
            getch();
            return 0;
}
```



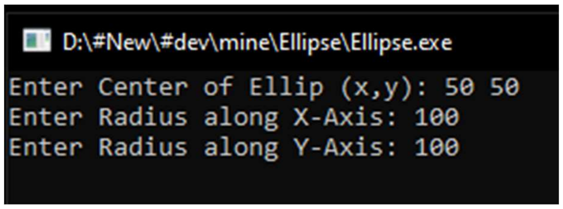
```
D:\#New\#dev\mine\Circle through MidPoint Algorithim\Circle through
Enter the Radius of the circle :        100
Enter the Center of the circle (x,y) :  100 100
```



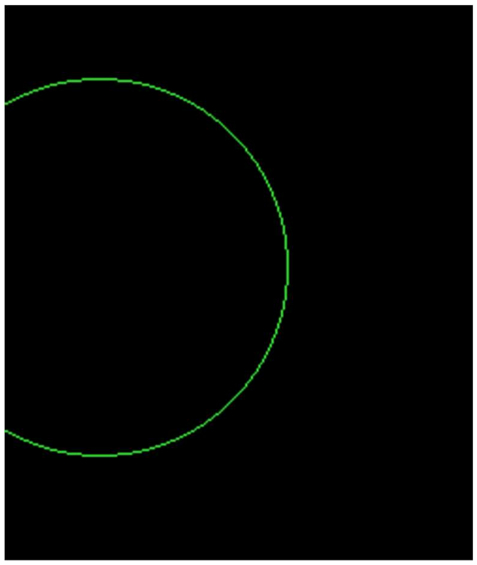
```
Windows BGI
100,100100,100100,-10100,-100
200,0   200,0   0   ,0   0   ,0
101,10099 ,100101,-1099 ,-100
200,1   200,-1  0   ,1   0   ,-1
102,10098 ,100102,-1098 ,-100
200,2   200,-2  0   ,2   0   ,-2
103,10097 ,100103,-1097 ,-100
200,3   200,-3  0   ,3   0   ,-3
104,10096 ,100104,-1096 ,-100
200,4   200,-4  0   ,4   0   ,-4
105,10095 ,100105,-1095 ,-100
200,5   200,-5  0   ,5   0   ,-5
106,10094 ,100106,-1094 ,-100
200,6   200,-6  0   ,6   0   ,-6
107,10093 ,100107,-1093 ,-100
200,7   200,-7  0   ,7   0   ,-7
108,10092 ,100108,-1092 ,-100
200,8   200,-8  0   ,8   0   ,-8
109,10091 ,100109,-1091 ,-100
200,9   200,-9  0   ,9   0   ,-9
110,99  90 ,99  110,-9990 ,-99
199,10  199,-101   ,10 1  ,-10
111,99  89 ,99  111,-9989 ,-99
199,11  199,-111   ,11 1  ,-11
112,99  88 ,99  112,-9988 ,-99
199,12  199,-121   ,12 1  ,-12
113,99  87 ,99  113,-9987 ,-99
199,13  199,-131   ,13 1  ,-13
114,99  86 ,99  114,-9986 ,-99
199,14  199,-141   ,14 1  ,-14
115,99  85 ,99  115,-9985 ,-99
199,15  199,-151   ,15 1  ,-15
116,99  84 ,99  116,-9984 ,-99
199,16  199,-161   ,16 1  ,-16
117,99  83 ,99  117,-9983 ,-99
199,17  199,-171   ,17 1  ,-17
118,98  82 ,98  118,-9882 ,-98
198,18  198,-182   ,18 2  ,-18
119,98  81 ,98  119,-9881 ,-98
198,19  198,-192   ,19 2  ,-19
120,98  80 ,98  120,-9880 ,-98
198,20  198,-202   ,20 2  ,-20
121,98  79 ,98  121,-9879 ,-98
198,21  198,-212   ,21 2  ,-21
122,98  78 ,98  122,-9878 ,-98
198,22  198,-222   ,22 2  ,-22
123,97  77 ,97  123,-9777 ,-97
197,23  197,-233   ,23 3  ,-23
124,97  76 ,97  124,-9776 ,-97
197,24  197,-243   ,24 3  ,-24
125,97  75 ,97  125,-9775 ,-97
197,25  197,-253   ,25 3  ,-25
126,97  74 ,97  126,-9774 ,-97
197,26  197,-263   ,26 3  ,-26
127,96  73 ,96  127,-9673 ,-96
196,27  196,-274   ,27 4  ,-27
128,96  72 ,96  128,-9672 ,-96
196,28  196,-284   ,28 4  ,-28
```

3. Ellipse Program

```
#include <stdio.h>
#include <conio.h>
#include<iostream>
#include <graphics.h>
#include <stdlib.h>
#include<math.h>
#define ROUND(x)  ((int)(x+0.5))
using namespace std;
float xc,yc,rx,ry;
void drawpoint(int x, int y){
 putpixel(xc+x,yc+y,10);
  putpixel(xc-x,yc+y,10);
 putpixel(xc+x,yc-y,10);
 putpixel(xc-x,yc-y,10);}
int main(){
        float x,y;
        int gdriver=DETECT, gmode, ecode;
        char comma;
        initgraph(&gdriver, &gmode, "C:\\TURBOC3\\BGI");
/* read result of initialization */
        ecode = graphresult();
        if (ecode != grOk){
                printf("Graphics error:\n");
                printf("Press any key...");
                getch();
                exit(1);
        }
float p;
        cout<<"Enter Center of Ellip (x,y): ";
        cin>>xc>>comma>>yc;
        cout<<"Enter Radius along X-Axis: ";
        cin>>rx;
        cout<<"Enter Radius along Y-Axis: ";
        cin>>ry;
        yc = getmaxy()-yc;
        x=0;
        y=ry;
        drawpoint(x,y);
        p=ROUND(ry*ry-rx*rx*ry+0.25*(rx*rx)); //ry2-rx2*ry+(.25*rx2)
        while((ry*ry)*x<(rx*rx)*y)
        {       x=x+1;
                if(p>=0)
                {       y=y-1;
                        p=p+2*(ry*ry)*x+ry*ry-2*(rx*rx)*y;
                }
                else
                {       p=p+2*(ry*ry)*x+ry*ry;
                }
                drawpoint(x,y);
        }
        p=ROUND(ry*ry*((x+.5)*(x+.5))+rx*rx*((y-1)*(y-1))-(rx*rx)*(ry*ry));
        while(y>0)
        {       y=y-1;
                if(p<=0)
                {       x=x+1;
                        p=p-2*(rx*rx)*y+rx*rx+2*(ry*ry)*x;
                }
                else
                {
                        p=p-2*(rx*rx)*y+rx*rx;
                }
                drawpoint(x,y);
        }
        getch();
        closegraph();
                                                            }
```

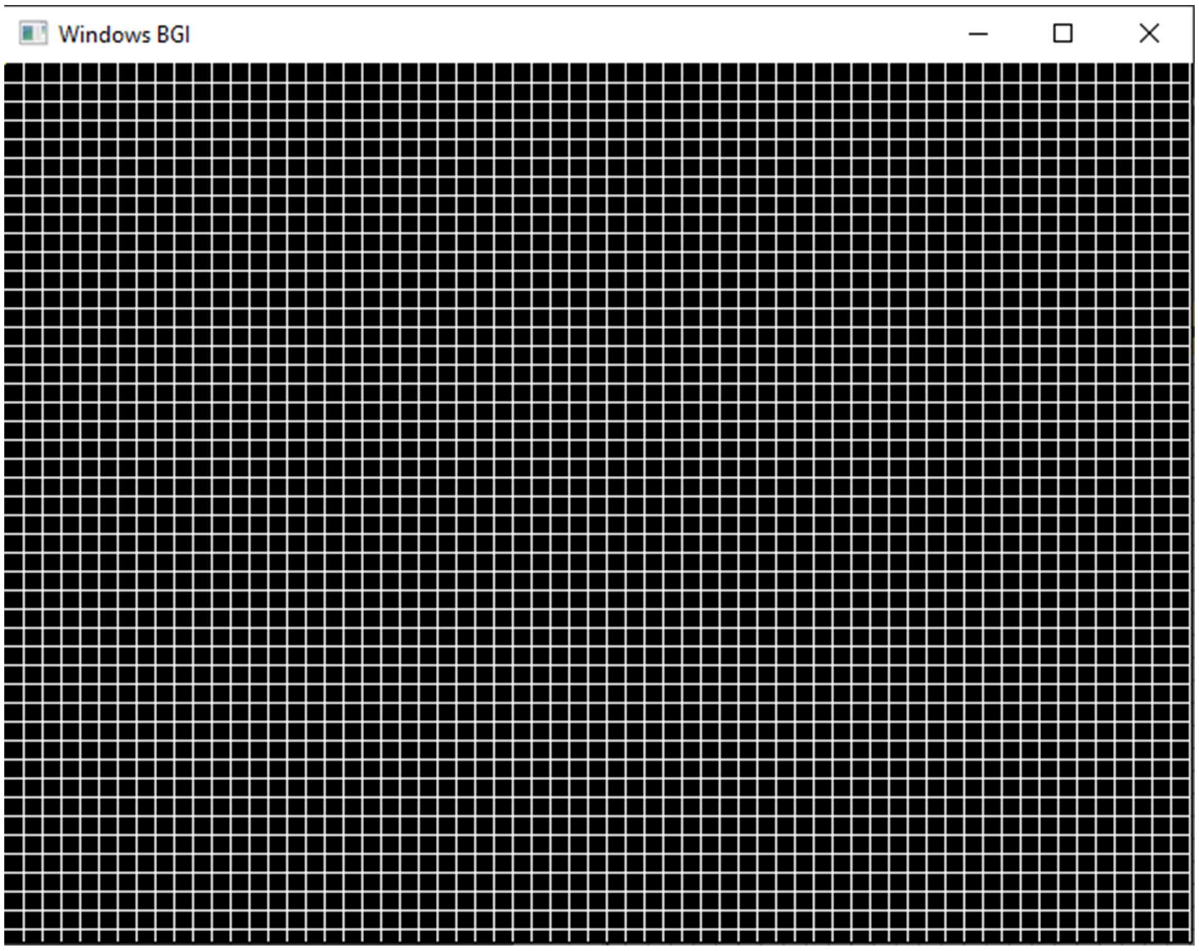


```
D:\#New\#dev\mine\Ellipse\Ellipse.exe
Enter Center of Ellip (x,y): 50 50
Enter Radius along X-Axis: 100
Enter Radius along Y-Axis: 100
```



4. Full Graph Program

```
#include<iostream>
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
using namespace std;
int main(void)
{
  /* request auto detection */
  int gdriver = DETECT, gmode, errorcode;
  int xmax, ymax;
  /* initialize graphics and local variables */
  initgraph(&gdriver, &gmode, "C:\\TURBOC3\\BGI");
  /* read result of initialization */
  errorcode = graphresult();
    if (errorcode != grOk)
  {printf("Graphics error: %s\n", grapherrormsg(errorcode));
    printf("Press any key to halt:");
    getch();
    exit(1); }
  setcolor(getmaxcolor());
  xmax = getmaxx();
  ymax = getmaxy();
  for(int i=10;i<xmax;i+=10)
  line(i,0,i,ymax);
  for(int i=10;i<ymax;i+=10)
  line(0,i,xmax,i);
     getch();
  closegraph();
  return 0;
}
```

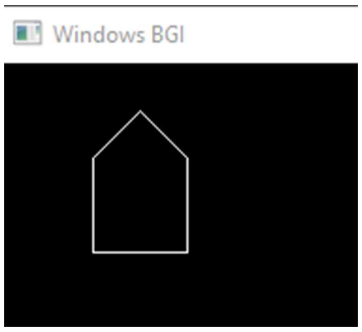


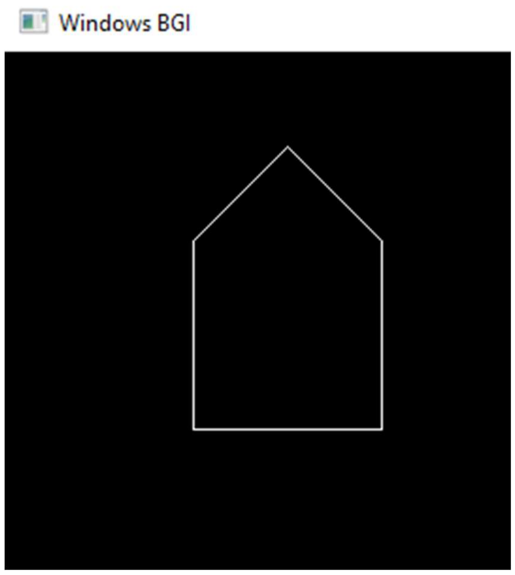
5. Different Functions at Object

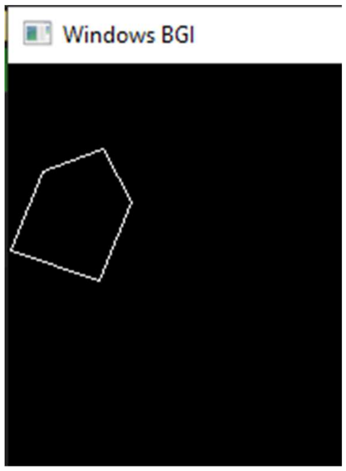
```
#include<iostream>
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
using namespace std;

int arr[10]={50,50, 50,100, 100,100, 100,50, 75,25};
int i;
        void draw(){
                for(i=0;i<=6;i+=2)
                        line(arr[i],arr[i+1],arr[i+2],arr[i+3]);
                        line(arr[i],arr[i+1],arr[0],arr[1]);  }
        void translate(int tx, int ty){
                for(i=0;i<=8;i+=2)
                        {arr[i]=arr[i]+tx;
```

```
                    arr[i+1]=arr[i+1]+ty;}
            }
        void scaling_inc(int sx, int sy){
                for(i=0;i<=8;i+=2){
                        arr[i]=arr[i]*sx;
                        arr[i+1]=arr[i+1]*sy;}
                }
        void scaling_dec(int sx, int sy){
                for(i=0;i<=8;i+=2){
                        arr[i]=arr[i]/sx;
                        arr[i+1]=arr[i+1]/sy;}}
 void rotate(float cos, float sin){
                    for(i=0;i<=8;i+=2){
                        arr[i]=(arr[i]*cos)-(arr[i+1]*sin);
                        arr[i+1]=(arr[i+1]*cos)+(arr[i]*sin);}
                }
int main(void){
  int gdriver = DETECT, gmode, errorcode;
  int xmax, ymax;
  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);}
  char c;
  draw();
  do{    c=getch();
                switch(c){
                case '5':
                rotate(0.9396,0.3420);
                break;
                case '1':
                scaling_inc(2,2);
                break;
                case '3':
```

```
                scaling_dec(2,2);
                break;
                case '2':
                translate(0,10);
                break;
                case '6':
                translate(10,0);
                break;
                case '4':
                translate(-10,0);
                break;
                case '8':
                translate(0,-10);
                break;
                }
        cleardevice();
        draw();
    }
    while(c!='q');
    getch();
    closegraph();
    return 0;
}
```

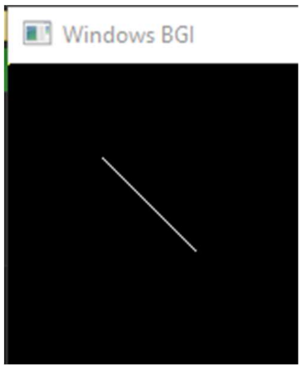






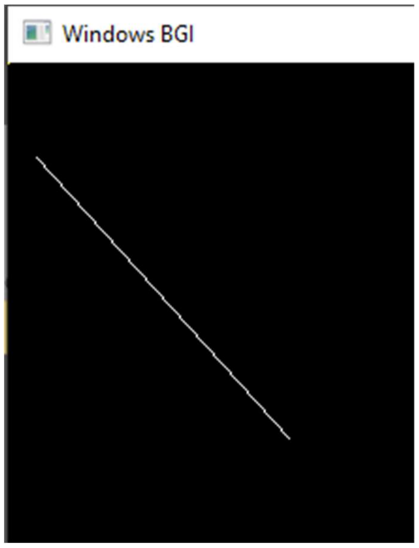
6. Draw a Line with the help of C++ program.

```
#include <iostream>
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
using namespace std;
int main(void)
{
  /* request auto detection */
  int gdriver=DETECT, gmode, errorcode;
  initgraph(&gdriver,&gmode,"C:\\TURBOC3\\BGI");
  int xmax, ymax;
  errorcode = graphresult();
  /* an error occurred */
  if (errorcode != grOk)
  {
    printf("Graphics error: %s\n", grapherrormsg(errorcode));
    printf("Press any key to halt:");
    getch();
    exit(1);
  }
  setcolor(getmaxcolor());
  /* draw a diagonal line */
  line(50, 50, 100,100);
 cout<<"that was line";
  getch();
  closegraph();
  return 0;
}
```



7. Another C++ program to draw a line at different points.

```
#include <iostream>
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
using namespace std;
int main(void)
{
 int gdriver = DETECT, gmode, errorcode;
   int xmax, ymax;
   /* initialize graphics and local variables */
   initgraph(&gdriver, &gmode, "C:\\TURBOC3\\BGI");
   /* read result of initialization */
   errorcode = graphresult();
   /* an error occurred */
   if (errorcode != grOk)
   {
      printf("Graphics error: %s\n", grapherrormsg(errorcode));
      printf("Press any key to halt:");
      getch();
      exit(1);
   }
  setcolor(getmaxcolor());

   /* draw a diagonal line */
   line(15, 50, 150,200);

   /* clean up */
   cout<<"that was line";
   getch();
   closegraph();
   return 0;
}
```



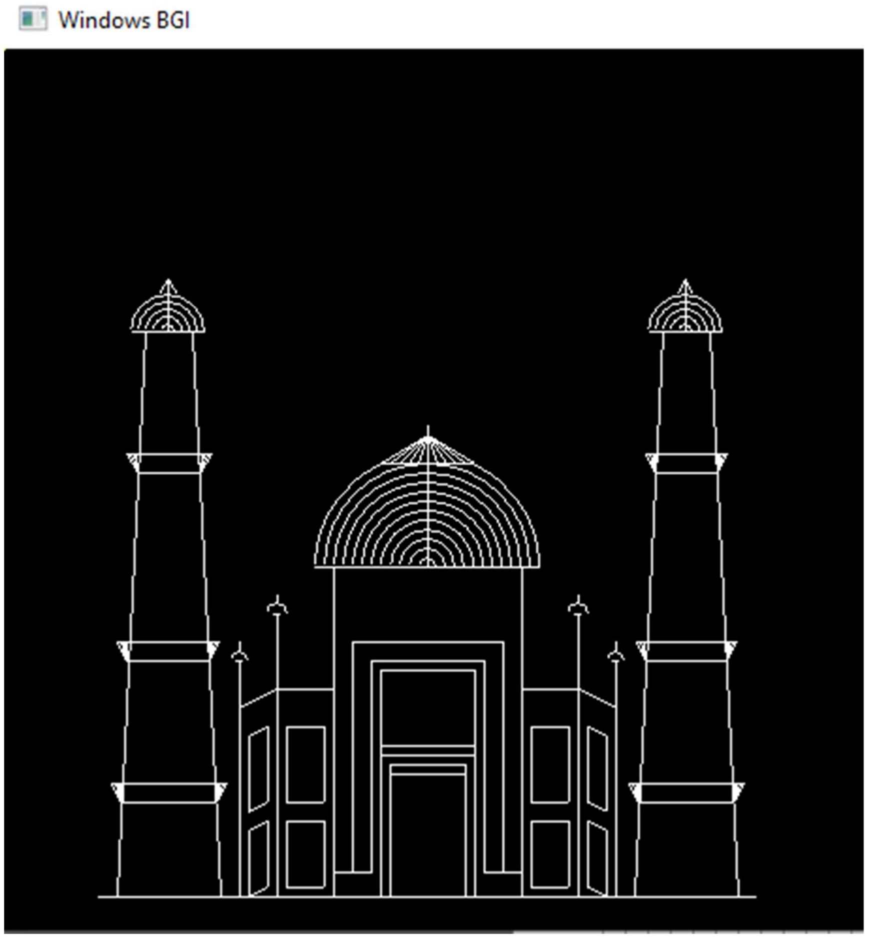
8. C++ program to draw a graphical image (lineCode).

```
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
int main(void)
{
  /* request auto detection */
  int gdriver = DETECT, gmode, errorcode;
  int xmax, ymax;

  /* initialize graphics and local variables */
  initgraph(&gdriver, &gmode, "C:\\turboc3\\bgi");

  /* read result of initialization */
  errorcode = graphresult();
  /* an error occurred */
  if (errorcode != grOk)
  {
    printf("Graphics error: %s\n", grapherrormsg(errorcode));
    printf("Press any key to halt:");
    getch();
    exit(1);
  }

  setcolor(getmaxcolor());
  xmax = getmaxx();
  ymax = getmaxy();

  /* draw a diagonal line */
  line(50, 450, 400, 450);
  line(75,150,60,450);
  line(100,150,115,450);
  line(375,150,390,450);
  line(350,150,335,450);
  line(125,325,125,450);
  line(325,325,325,450);

  line(145,300,145,450);
  line(305,300,305,450);

  line(175,275,175,450);
  line(275,275,275,450);

  line(185,315,185,437);
  line(265,315,265,437);
  line(185,315,265,315);

  line(195,325,195,437);
  line(255,325,255,437);
  line(195,325,255,325);

  line(195,437,175,437);
  line(255,437,275,437);

  line(200,330,200,450);
  line(250,330,250,450);
  line(200,330,250,330);
  line(200,370,250,370);
  line(200,375,250,375);

  line(205,380,205,450);
  line(245,380,245,450);
  line(205,380,245,380);
  line(205,385,245,385);



  line(125,350,145,340);
  line(145,340,175,340);

  line(275,340,305,340);
  line(305,340,325,350);

  line(150,360,170,360);
  line(150,360,150,400);
  line(170,360,170,400);
  line(150,400,170,400);

  line(130,365,140,360);
  line(140,360,140,400);
  line(130,365,130,405);
  line(130,405,140,400);

  line(150,410,170,410);
  line(150,410,150,445);
  line(170,410,170,445);
  line(150,445,170,445);

  line(140,410,140,445);
  line(130,415,130,450);
  line(130,415,140,410);
  line(130,450,140,445);

  //Right Windowz
  line(280,360,300,360);
  line(280,360,280,400);
  line(300,360,300,400);
  line(280,400,300,400);

  line(280,410,300,410);
  line(280,410,280,445);
  line(300,410,300,445);
  line(280,445,300,445);

  line(310,360,310,400);
  line(310,360,320,365);
  line(310,360,310,400);
  line(310,400,320,405);
```

```
 line(320,365,320,405);

 line(310,410,310,445);
 line(320,415,320,450);
 line(310,410,320,415);
 line(310,445,320,450);

 //tomb arc
 arc(225,275,0,65,60);
 arc(225,275,115,180,60);
//right little tomb
 arc(325,325,0,180,5);
 line(325,315,325,320);
 line(323,325,327,325);
 // Right menaar tomb....
 arc(362.5,150,0,180,20);
 line(343,150,382,150);
 line(362.5,122,358,130);
 line(362.5,122,366,130);
 line(362.5,122,362.5,150);
 arc(362.5,150,0,180,16);
 arc(362.5,150,0,180,12);
 arc(362.5,150,0,180,8);
 arc(362.5,150,0,180,4);

 // Left Menaar Tomb....
 arc(87.5,150,0,180,20);
 line(68,150,107,150);
 line(87.5,122,87.5,150);
 line(87.5,122,82.5,130);
 line(87.5,122,92.5,130);
 arc(87.5,150,0,180,16);
 arc(87.5,150,0,180,12);
 arc(87.5,150,0,180,8);
 arc(87.5,150,0,180,4);

 // left little tomb arc
 arc(125,325,0,180,5);
 line(123,325,127,325);
 line(125,315,125,320);

 arc(225,275,0,180,55);
 arc(225,275,0,180,50);
 arc(225,275,0,180,45);
 arc(225,275,0,180,40);
 arc(225,275,0,180,35);
 arc(225,275,0,180,30);
 arc(225,275,0,180,25);
 arc(225,275,0,180,20);
 arc(225,275,0,180,15);
 arc(225,275,0,180,10);
 arc(225,275,0,180,5);

 line(200,220,250,220);
 line(225,220,225,275);

// arc(350,150,0,180,10);
line(165,275,285,275);
line(225,205,200,220);
line(225,205,250,220);
line(225,205,225,220);

line(225,205,230,220);
line(225,205,235,220);
line(225,205,240,220);
line(225,205,245,220);

line(225,205,220,220);
line(225,205,215,220);
line(225,205,210,220);
line(225,205,205,220);

line(225,200,225,205);
```

```
 arc(145,300,0,180,6);
 line(143,300,147,300);
 line(145,290,145,294);

 arc(305,300,0,180,6);
 line(303,300,307,300);
 line(305,290,305,294);

 line(62,400,113,400);
 line(57,390,118,390);
 line(57,390,62,400);
 line(59,390,62,400);
 line(61,390,62,400);
line(60,390,62,400);
line(62,390,62,400);

 line(118,390,113,400);
 line(116,390,113,400);
 line(114,390,113,400);
 line(113,390,113,400);

 line(65,325,109,325);
 line(60,315,114,315);
 line(60,315,65,325);
 line(62,315,65,325);
 line(63,315,65,325);
 line(64,315,65,325);
 line(65,315,65,325);
 line(114,315,109,325);
 line(112,315,109,325);
 line(110,315,109,325);
 line(109,315,109,325);

 line(71,225,104,225);
 line(65,215,110,215);
 line(65,215,71,225);
 line(67,215,71,225);
line(69,215,71,225);
line(70,215,71,225);
line(110,215,104,225);
line(108,215,104,225);
line(106,215,104,225);
line(105,215,104,225);

line(337,400,388,400);
line(332,390,393,390);
line(332,390,337,400);
line(334,390,337,400);
line(336,390,337,400);
line(337,390,337,400);
line(393,390,388,400);
line(391,390,388,400);
line(389,390,388,400);
line(388,390,388,400);

line(341,325,384,325);
line(336,315,389,315);
line(336,315,341,325);
line(338,315,341,325);
line(340,315,341,325);
line(389,315,384,325);
line(387,315,384,325);
line(385,315,384,325);
line(384,315,384,325);

line(346,225,379,225);
line(341,215,384,215);
line(341,215,346,225);
line(343,215,346,225);
line(345,215,346,225);
line(346,215,346,225);

line(384,215,379,225);
line(382,215,379,225);
```

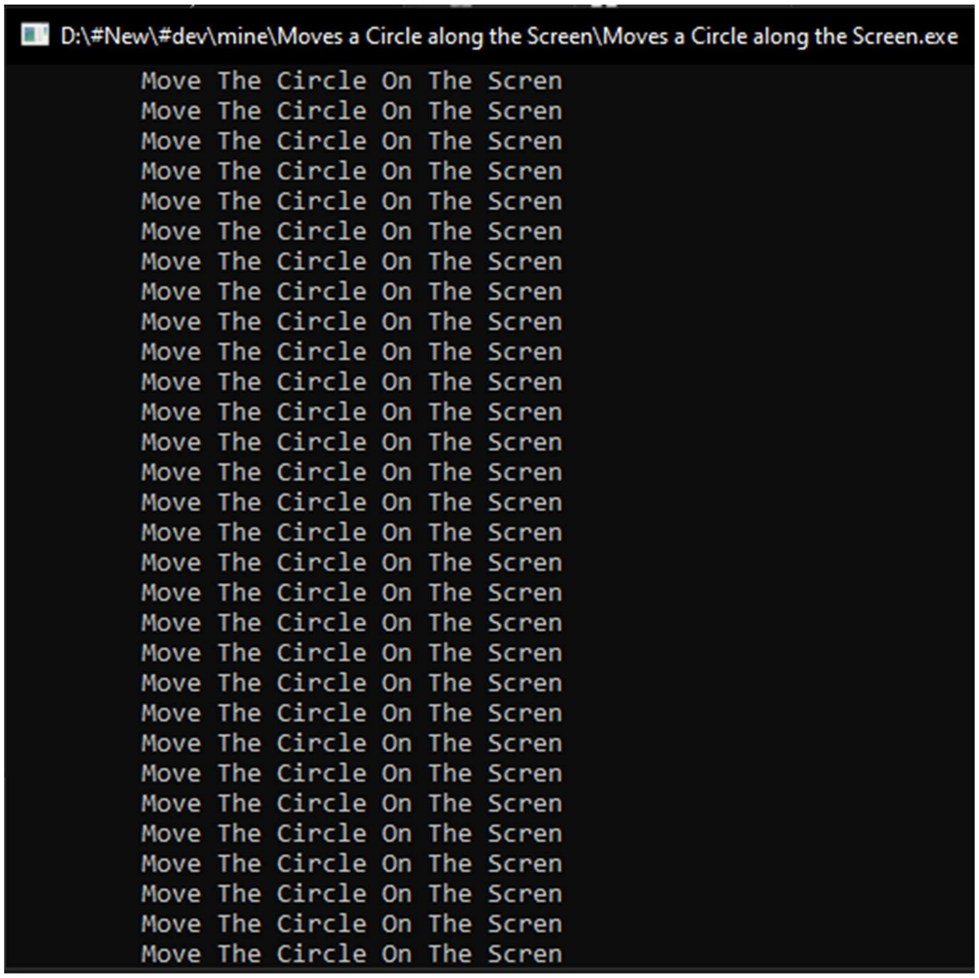
```
 line(380,215,379,225);
 line(379,215,379,225);



  /* clean up */
  getch();
  closegraph();
  return 0;
}
```



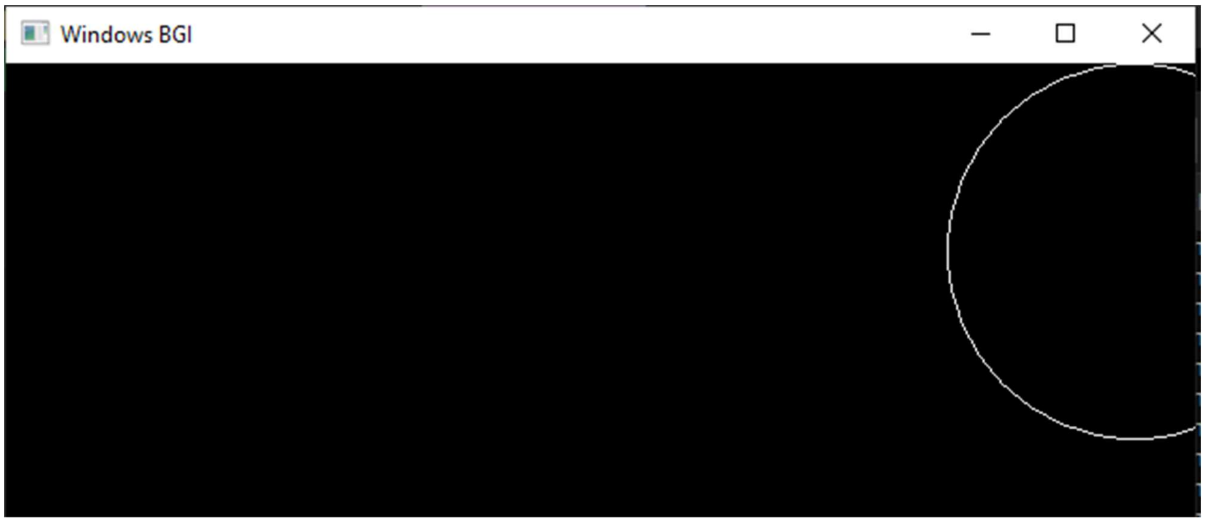
9. C++ program to move a circle along the screen.

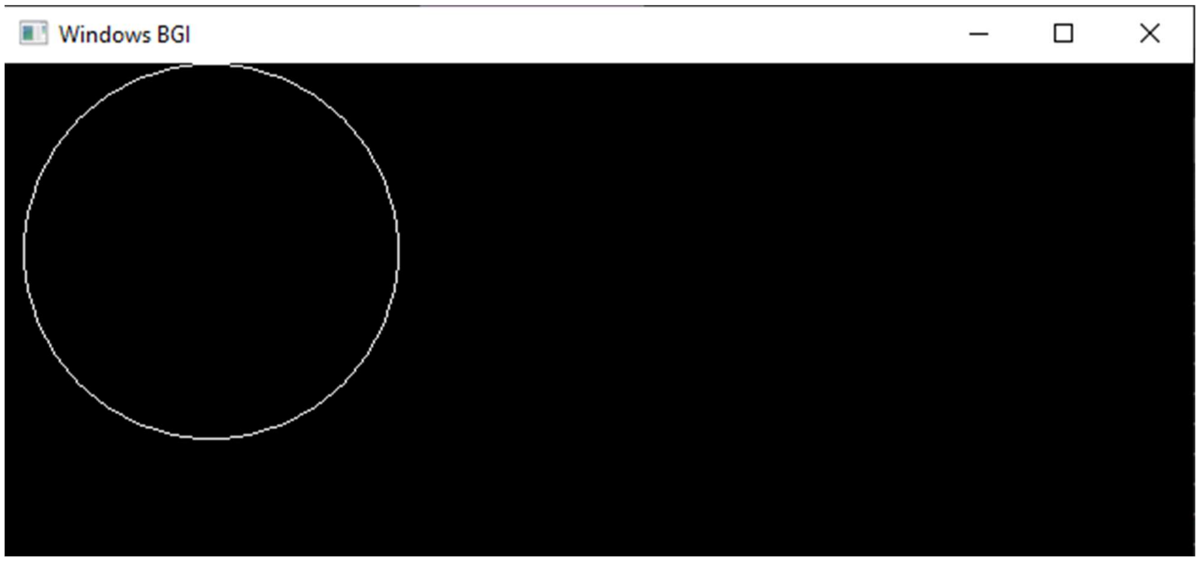
```
#include <iostream>
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <dos.h>
using namespace std;
int main()
{
  int gdriver = DETECT, gmode, errorcode;
  int xmax, ymax;
  initgraph(&gdriver, &gmode, "c:\\turboc3\\bgi");
  errorcode = graphresult();
  if (errorcode != grOk)
  {
    printf("Graphics error: %s\n", grapherrormsg(errorcode));
    printf("Press any key to halt:");
    getch();
    exit(1);
  }
  xmax = getmaxx();
  ymax = getmaxy();
 {
        while(!kbhit())
{
for(int i=0;i<=800;i+=10)
{
                delay (100);

               cleardevice();
                printf("\n \t Move The Circle On The Scren");
               circle(i,100,100);
}
for(int i=800;i>=20;i-=10)
              {
               delay(100);
```

```
                    cleardevice();
                    printf("\n \t Move The Circle On The Scren");
                   circle(i-50,100,100);
                    }
    }
          getch();
          closegraph();
          return 0;
    }
    }
```



```
D:\#New\#dev\mine\Moves a Circle along the Screen\Moves a Circle along the Screen.exe
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
         Move The Circle On The Scren
```

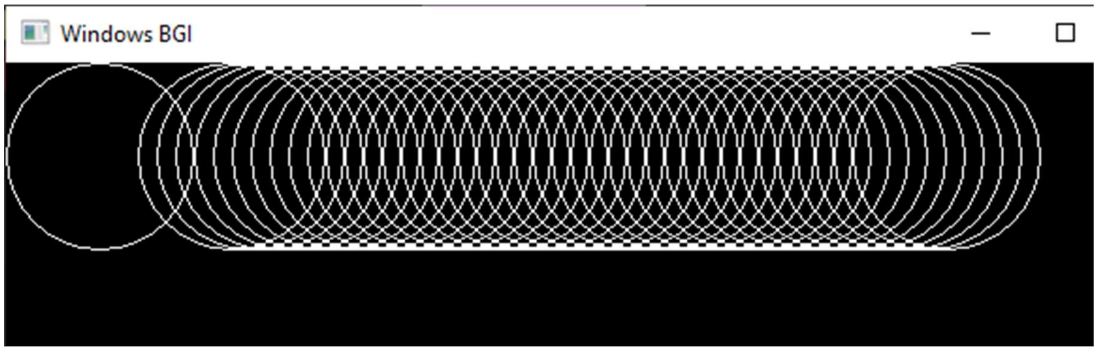




10. C++ program to draw a moving circle.

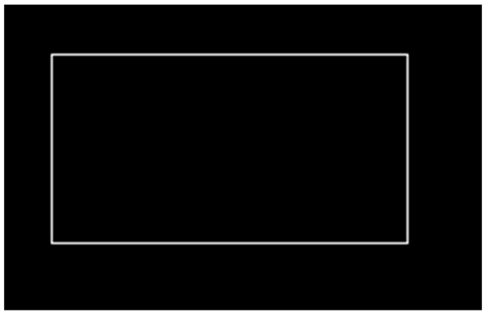
```
# include <iostream>
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
# include <dos.h>
using namespace std;
int main(void)
{
  /* request auto detection */
  int gdriver = DETECT, gmode, errorcode;
  int xmax, ymax;
  int i;
  /* initialize graphics and local variables */
  initgraph(&gdriver, &gmode, "C:\\TURBOC3\\BGI");
  /* read result of initialization */
  errorcode = graphresult();
  /* an error occurred */
  if (errorcode != grOk)
  {
    printf("Graphics error: %s\n", grapherrormsg(errorcode));
    printf("Press any key to halt:");
    getch();
```

```
      exit(1);
    }

  {        while(!kbhit())
{          for(int i=0;i<=500;i+=10)
                       delay(100);
                      cleardevice();
                      circle(i,50,50);}
for(i=550;i>=20;i-=10)
{
                      delay(100);
                      circle(i-50,50,50);}
}
  /* clean up */
   getch();
   closegraph();
   return 0;
}
```



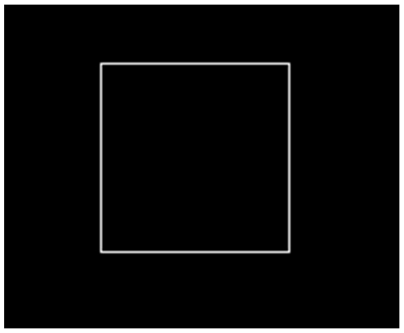
11. C++ program to draw a rectangle.

```
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
int main()
{
  int gdriver = DETECT, gmode, errorcode;
  int xmax, ymax;
  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);
}
  setcolor(getmaxcolor());
  xmax = getmaxx();
  ymax = getmaxy()  ;
int left = getmaxx() / 2 ;

int top = getmaxy() / 2 - 50;
int right = getmaxx() /5;
int bottom = getmaxy() / 2 +50;
printf("\n Display the Retanguler");
rectangle(left,top,right,bottom);
getch();
closegraph();
return 0;
```



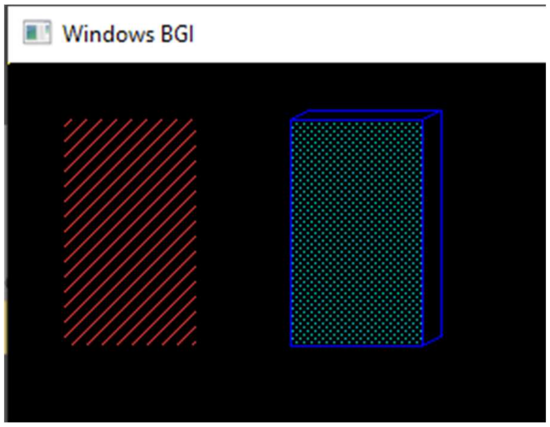
12. C++ program to draw a square.

```
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
int main()
{
   int gdriver = DETECT, gmode, errorcode;
   int xmax, ymax;
   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);
   }
  setcolor(getmaxcolor());
   xmax = getmaxx();
   ymax = getmaxy()  ;


int left = getmaxx()/2-50 ;
int top = getmaxy()/ 2 - 50;
int right = getmaxx()/2+50;
int bottom = getmaxy()/2+50;
rectangle(left,top,right,bottom);
getch();
closegraph();
return 0;
}
```



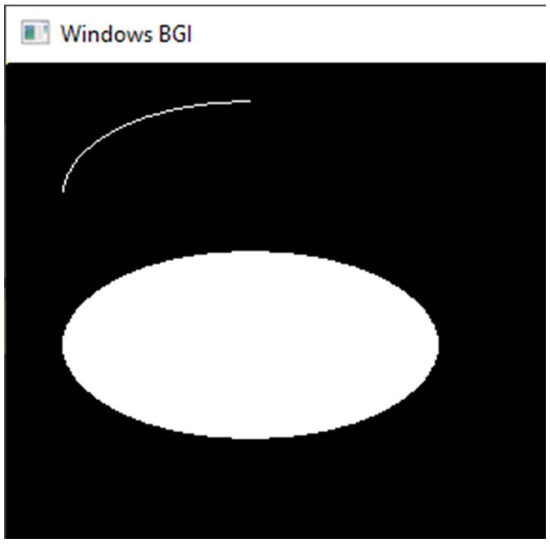
13. Write a C++ program that draws 2D and 3D rectangles with patterns and colors on Windows BGI using bar() and bar3d() functions in graphical programming.

```
#include <iostream>
#include <graphics.h>
using namespace std;
int main()
{
        int gd=DETECT, gm, err,x=30,y=30;
        initgraph(&gd,&gm,"");
        err=graphresult();
        if(err!=grOk)
        {
                cout<<"Graphics Error "<<grapherrormsg(err);
                exit(1);
        }
        setcolor(BLUE);
        setfillstyle(3,LIGHTRED);
        bar(30,30,100,150);
        setfillstyle(11,CYAN);
        bar3d(150,30,220,150,10,1);
        getch();
        closegraph();
        return 0;
}
```



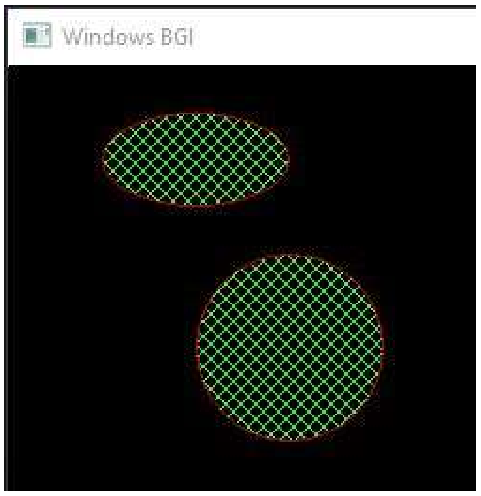
14. Write a C++ program that draws an arc and ellipse on BGI Window using ellipse() and fillellipse() functions in graphical programming.

```
#include <iostream>
#include <graphics.h>
using namespace std;
int main()
{
	int gd=DETECT, gm, err;
	initgraph(&gd,&gm,"");
	err=graphresult();
	if(err!=grOk)
	{
		cout<<"Graphics Error "<<grapherrormsg(err);
		exit(1);
	}
	ellipse(130,70,90,180,100,50);
	fillellipse(130,150,100,50);
	getch();
	closegraph();
	return 0;
}
```



15. Write a program that draws ellipse and circle with patterns and colors on Windows BGI using setfillstyle() and floodfill() functions in graphical programming.

```
#include <iostream>
#include <graphics.h>
using namespace std;
int main()
{
	int gd=DETECT, gm, err,x=30,y=30;
	initgraph(&gd,&gm,"");
	err=graphresult();
	if(err!=grOk)
	{
		cout<<"Graphics Error "<<grapherrormsg(err);
		exit(1);
	}
	setcolor(RED);
	setfillstyle(8,10);
	fillellipse(100,50,50,25);
	circle(150,150,50);
	floodfill(150,150,4);
	getch();
	closegraph();
	return 0;
}
```


Windows BGI