Computer Graphics (ITCO9)

Practical File

Submitted To:
Mr. Jagdeep Singh
TRF-IT

Submitted By:
Shrid Pant
2017UIT2623

NETAJI SUBHAS UNIVERSITY OF TECHNOLOGY

DWARKA SECTOR-3, DELHI-110078

S.No.	Problem Statement
1	Study of basic graphics functions in graphics.h
2	Write a program to draw ellipse using Midpoint Algorithm
3	Write a program to rotate a point about origin
4	Write a program to scale a triangle
5	Write a program for line clipping
6	Write a program for rotation of 3D object about any arbitrary axis
7	Draw a hut with two windows
8	Draw a full moon with three stars
9	Draw a rotating coin on table

10	Write your name in Hindi using Bezier Curve

1. Study of basic graphics functions in graphics.h

getpixel() and putpixel()
The getpixel() function returns an integer to represent the color of the pixel at the co-ordinates specified.

cleardevice()
This function clears the graphics screen and positions the cursor at 0, 0.

arc();
This function is useful in drawing a circular arc.

initgraph();
This function initialises the graphics mode.

closegraph();
This function shuts down the graphics mode.

circle();
This function draws a circle with the specified coordinates.

line();
This function draws a line from one point to the other.

rectangle();
This function draws a rectangle about the specified points.

setcolor();
This function gives colour to the specified object.

setfillstyle();
setfillstyle function sets the current fill pattern and fill color.

void drawArc(GRectangle bounds, double start, double sweep); Draws an elliptical arc inscribed in a rectangle.

void drawImage(string filename, double x, double y); Draws the image from the specified file with its upper left corner at the specified point. The forms of the call that include the bounds scale the image so that it fits inside the specified rectangle.

void fillArc(GRectangle bounds, double start, double sweep); Fills a wedge-shaped area of an elliptical arc.

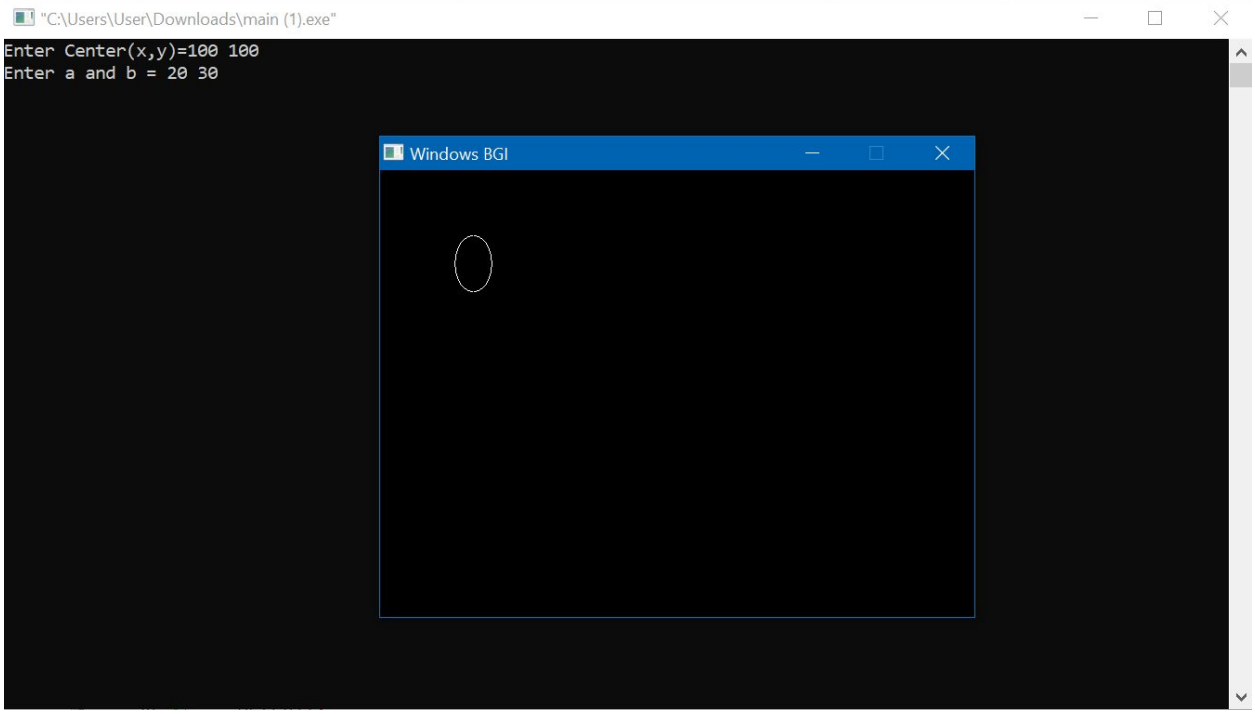
void fillOval(double x, double y, double width, double height); Fills the frame of a oval with the specified bounds.

2. Program to draw ellipse using midpoint algorithm.

```
 #include<bits/stdc++.h>
#include<graphics.h>
using namespace std;

void ellipse(double x, double y, double xc, double yc) {

        putpixel(xc + x, yc + y, 15);
        putpixel(xc + x, yc - y, 15);
        putpixel(xc - x, yc + y, 15);
        putpixel(xc - x, yc - y, 15);

}

int main() {
        int gd = DETECT, gm;
        initgraph(&gd, &gm, "C:\\TC\\BGI");
        double xc, yc;
        cout << "Enter Center(x,y)=";
        cin >> xc >> yc;
        cout << "Enter a and b = ";
        double rx, ry;
        cin >> rx >> ry;
        double x = 0, y = ry;
        double po = pow(ry, 2) - (pow(rx, 2) * ry) + (pow(rx, 2));
        while (rx * rx * y >= ry * ry * x) {
                if (po <= 0) {
```

```
                y = y;
                po = po + 2 * ry * ry * (x + 1) + ry * ry;
            }
            else {
                y = y - 1;
                po = po + 2 * ry * ry * (x + 1) + ry * ry - 2 * rx * rx * (y - 1);
            }
            x = x + 1;
            ellipse(x, y, xc, yc);
        }
        po = ((x + (1 / 2.0)) * (x + (1 / 2.0)) * ry * ry + (y - 1) * (y - 1) * rx * rx) - rx * rx
* ry * ry;
        while (y != 0) {
            if (po < 0) {
                x = x + 1;
                po = po - 2 * rx * rx * (y - 1) + rx * rx + 2 * ry * ry * (x + 1);
            }
            else {
                x = x;
                po = po - 2 * rx * rx * (y - 1) + rx * rx;
            }
            y = y - 1;
            ellipse(x, y, xc, yc);
        }
        getch();
}
```

Output:

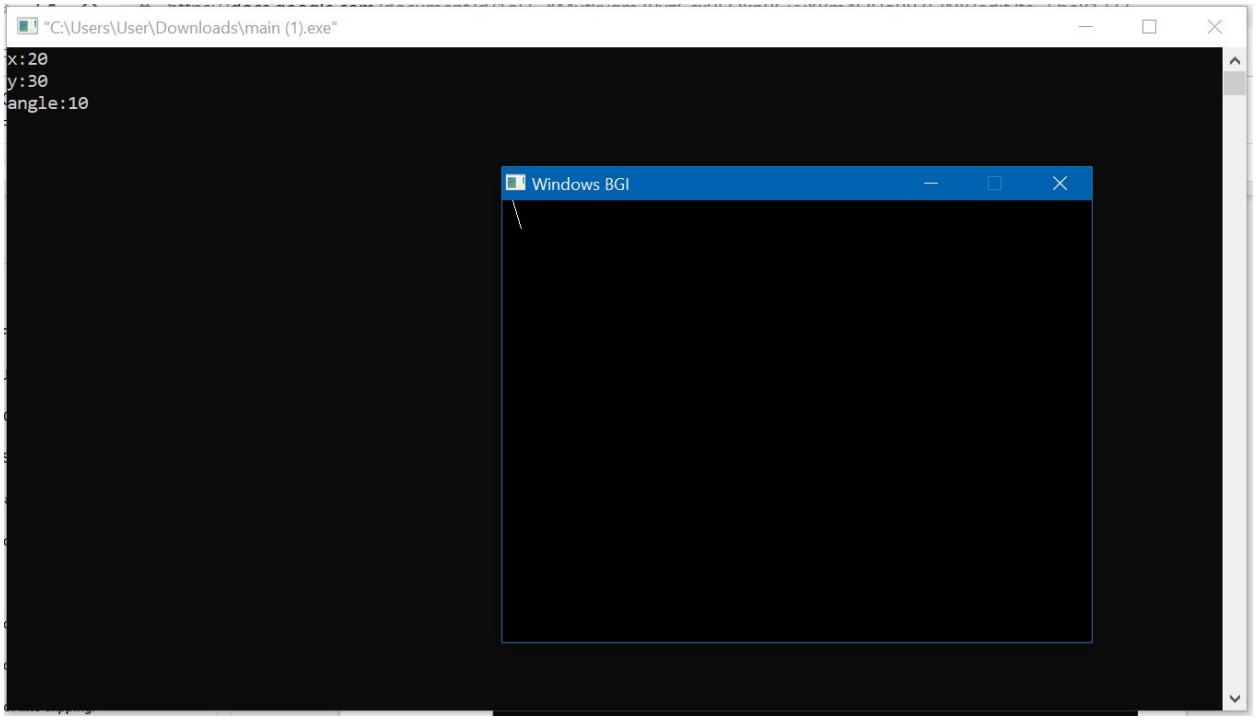


3. Program to rotate a point about origin.

```
#include<bits/stdc++.h>
#include<graphics.h>

using namespace std;

int main(){
int gd=DETECT,gm;
initgraph(&gd,&gm," ");
float x,y,angle;
cout<<"x:";cin>>x;
cout<<"y:";cin>>y;
putpixel(x,y,RED);
cout<<"angle:";cin>>angle;
int x_new=x*cos(angle)-y*sin(angle);
int y_new=x*sin(angle)+y*cos(angle);
putpixel(x_new,y_new,WHITE);
line(x,y,x_new,y_new);
getch();
```

```
closegraph();
return 0;
}
```

Output:



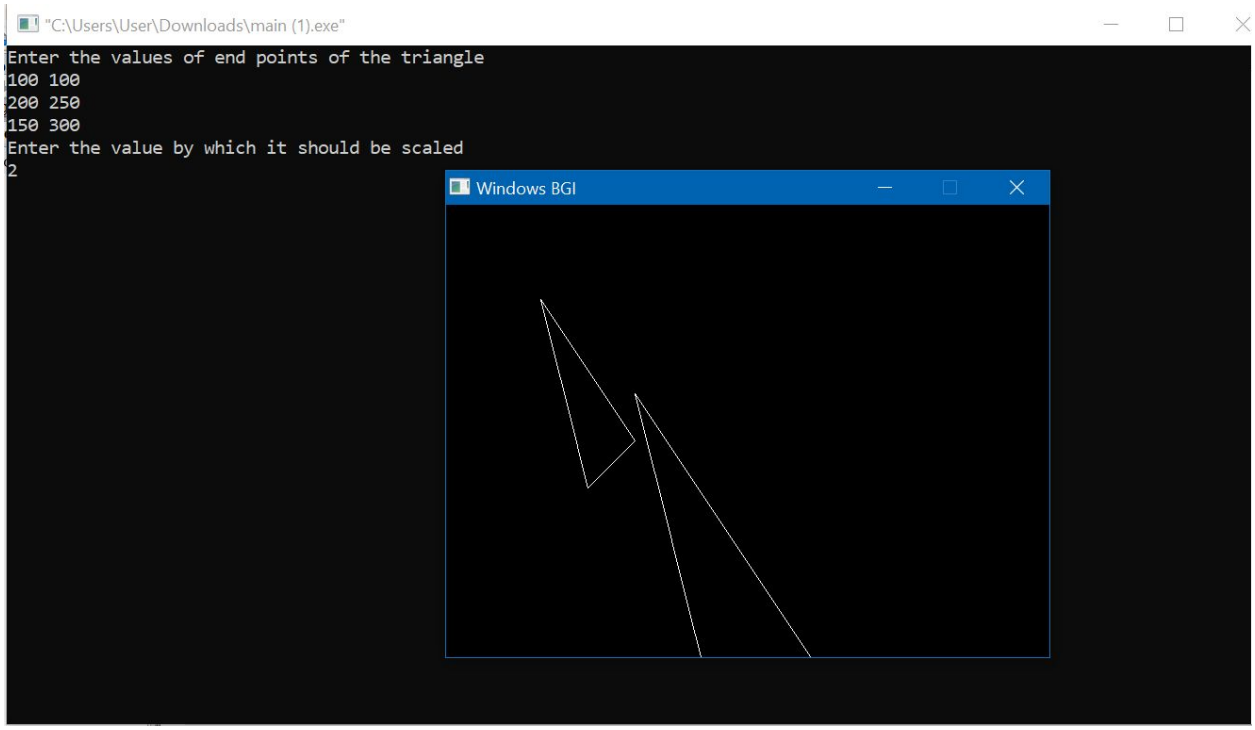
4. Program to scale a triangle.

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<process.h>
#include<math.h>
#include <iostream>

using namespace std;

int x1,yone,x2,y2,x3,y3,mx,my;
void draw();
void scale();
```

```
int main()
{
   int gd=DETECT,gm;
   int c;
   initgraph(&gd,&gm,"..\\bgi");
   cout << "Enter the 1st point for the triangle:";
   cin >>x1 >> yone;
   cout << "Enter the 2nd point for the triangle:";
   cin >> x2 >> y2;
   cout << "Enter the 3rd point for the triangle:";
   cin >> x3 >> y3;
   draw();
   scale();

   return 0;
}

 void draw()
{
   line(x1,yone,x2,y2);
   line(x2,y2,x3,y3);
   line(x3,y3,x1,yone);
}
void scale()
{
   int x,y,a1,a2,a3,b1,b2,b3;
   int mx,my;
   cout << "Enter the scaling coordinates";
   cin >> x >> y;
   mx=(x1+x2+x3)/3;
   my=(yone+y2+y3)/3;
   cleardevice();
   a1=mx+(x1-mx)*x;
   b1=my+(yone-my)*y;
   a2=mx+(x2-mx)*x;
    b2=my+(y2-my)*y;
    a3=mx+(x3-mx)*x;
   b3=my+(y3-my)*y;
   line(a1,b1,a2,b2);
```

```
  line(a2,b2,a3,b3);
  line(a3,b3,a1,b1);
  draw();
  getch();

}
```

Output:



5. Program for line clipping.

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

// Defining region codes
const int INSIDE = 0; // 0000
const int LEFT = 1;   // 0001
const int RIGHT = 2;  // 0010
```

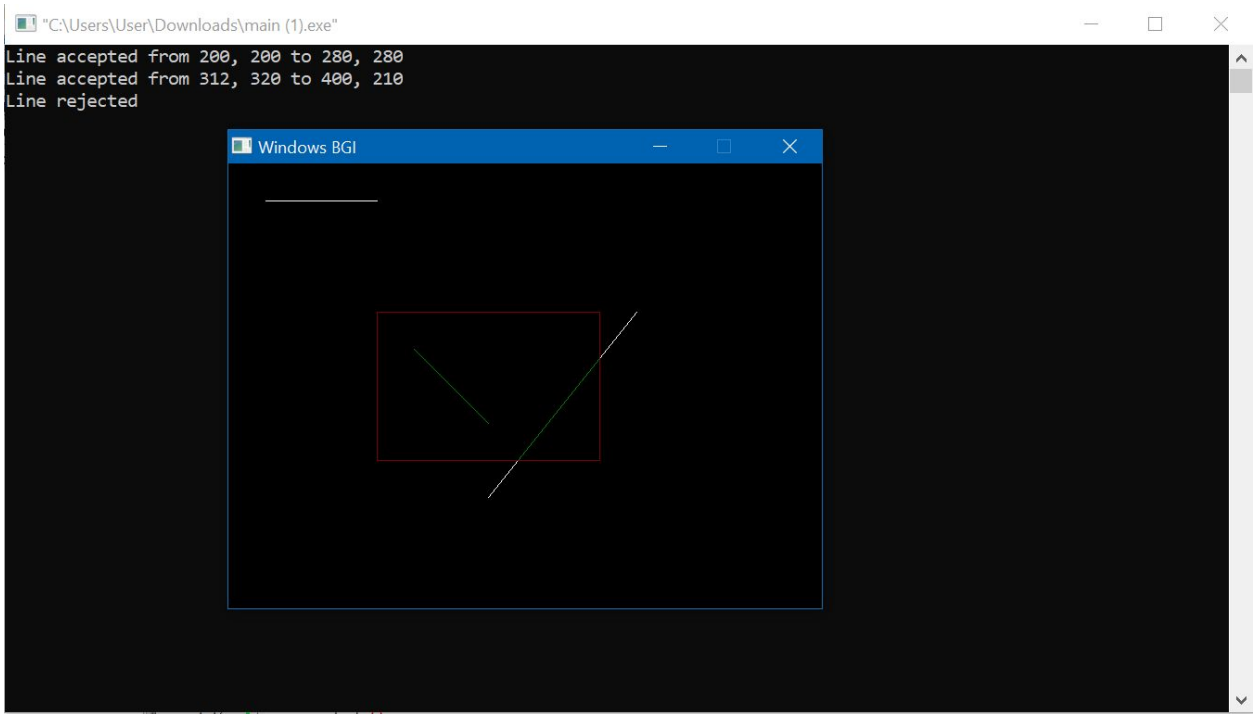
```
const int BOTTOM = 4; // 0100
const int TOP = 8;    // 1000
const int x_max = 400;
const int y_max = 320;
const int x_min = 160;
const int y_min = 160;

int computecode(double x, double y)
{

   int code = INSIDE;

   if (x < x_min)
      code |= LEFT;
   else if (x > x_max)
      code |= RIGHT;
   if (y < y_min)
      code |= BOTTOM;
   else if (y > y_max)
      code |= TOP;

   return code;
}


void lineclipping(double x1, double y1,
                  double x2, double y2)
{

   int code1 = computecode(x1, y1);
   int code2 = computecode(x2, y2);


   bool accept = false;

   while (true)
   {
      if ((code1 == 0) && (code2 == 0))
      {
```

```
    accept = true;
    break;
}
else if (code1 & code2)
{

    break;
}
else
{

    int code_out;
    double x, y;


    if (code1 != 0)
        code_out = code1;
    else
        code_out = code2;


    if (code_out & TOP)
    {

        x = x1 + (x2 - x1) * (y_max - y1) / (y2 - y1);
        y = y_max;
    }
    else if (code_out & BOTTOM)
    {

        x = x1 + (x2 - x1) * (y_min - y1) / (y2 - y1);
        y = y_min;
    }
    else if (code_out & RIGHT)
    {

        y = y1 + (y2 - y1) * (x_max - x1) / (x2 - x1);
        x = x_max;
```

```
            }
            else if (code_out & LEFT)
            {

               y = y1 + (y2 - y1) * (x_min - x1) / (x2 - x1);
               x = x_min;
            }

            if (code_out == code1)
            {
               x1 = x;
               y1 = y;
               code1 = computecode(x1, y1);
            }
            else
            {
               x2 = x;
               y2 = y;
               code2 = computecode(x2, y2);
            }
         }
      }
      if (accept)
      {
         cout <<"Line accepted from " << x1 << ", "
              << y1 << " to "<< x2 << ", " << y2 << endl;
         setcolor(GREEN);
         line(x1,y1,x2,y2);
      }
      else
         cout << "Line rejected" << endl;
}

int main()
{
   int gd = DETECT, gm;
         initgraph(&gd, &gm, "C:\\TC\\BGI");
         setcolor(RED);
         rectangle(x_min,y_min,x_max,y_max);
```

```
    setcolor(WHITE);
    line(280,360,440,160);
    line(40,40,160,40);
    line(200,200,280,280);
    lineclipping(200, 200, 280, 280);


    lineclipping(280, 360, 440, 160);

    lineclipping(40, 40, 160, 40);
    getch();
    closegraph();

    return 0;
}
```

Output:



6. Program to draw a hut with 2 windows.

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
int main()
{
int gd=DETECT,gm;

initgraph(&gd,&gm,"c:\\tc\\bgi");
setcolor(3);
line(100,200,300,200);

line(100,400,300,400);

line(100,400,100,200);

line(300,400,300,200);

line(300,200,500,200);

line(300,400,500,400);

line(500,200,500,400);

line(170,250,270,250);

line(170,200,270,200);

line(170,250,170,400);

line(270,250,270,400);

line(200,50,400,50);

line(400,50,500,200);

line(300,200,200,50);

line(100,200,200,50);
```

```
getch();
closegraph();

return 0;
}
```

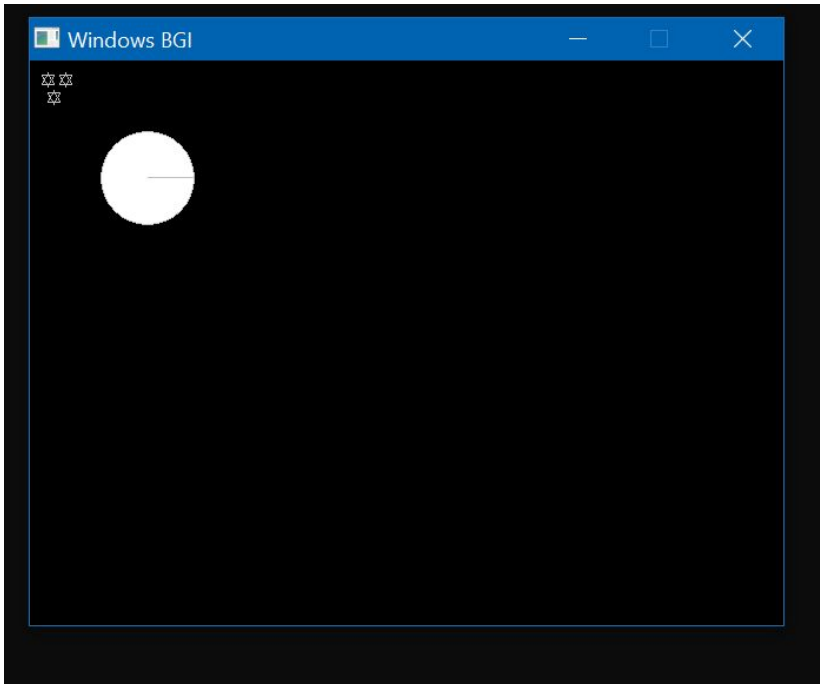
Output:

7. Program to draw a moon with 3 stars.

```
#include<bits/stdc++.h>
#include<graphics.h>
using namespace std;

int main()
{
   int gd = DETECT, gm;
         initgraph(&gd, &gm, "C:\\TC\\BGI");
```

```
        setcolor(LIGHTGRAY);
        pieslice(100,100,0,360,40);
        line(15,10,10,20);
    line(10,20,20,20);
    line(20,20.0,15,10);
    line(10,12.5,20,12.5);
    line(10,12.5,15,22.5);
    line(15,22.5,20,12.5);

    line(30,10,25,20);
    line(25,20,35,20);
    line(35,20.0,30,10);
    line(25,12.5,35,12.5);
    line(25,12.5,30,22.5);
    line(30,22.5,35,12.5);

    line(20,25,15,35);
    line(15,35,25,35);
    line(25,35.0,20,25);
    line(15,27.5,25,27.5);
    line(15,27.5,20,37.5);
    line(20,37.5,25,27.5);
        getch();
        closegraph();
        return 0;
}
```

Output:

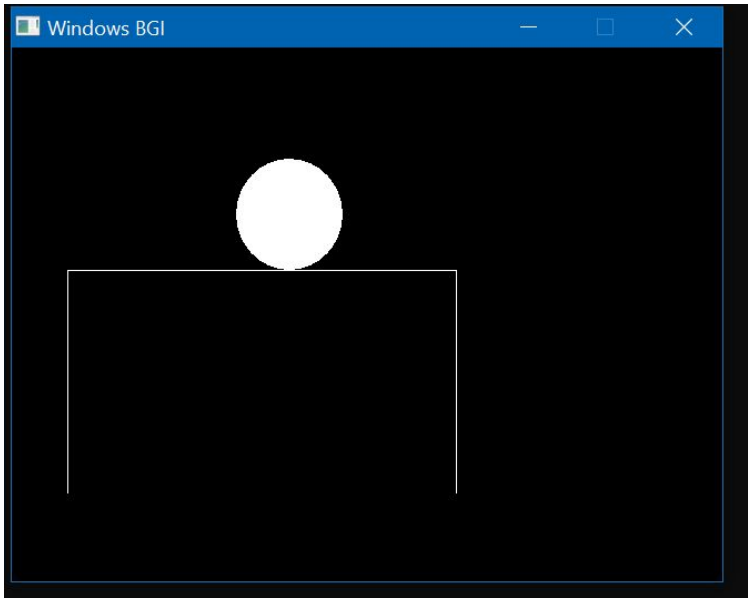


8. Program to draw a rotating coin on the table.

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

int main(){
 int gd = DETECT,gm;
   initgraph(&gd, &gm, "X:\\TC\\BGI");
   int i=0;

   line(50,200,400,200);


   while(true){
   while(true){
      ellipse(250,150,0,360,i,50);
      line(50,200,400,200);
      line(50,200,50,400);
      line(400,200,400,400);
      setfillstyle(SOLID_FILL, WHITE);
      floodfill(250, 150, WHITE);
```

```
        if(i==50){
            break;
        }
        i++;
        delay(100);
        cleardevice();
    }
    delay(100);
    cleardevice();
    while(true){
        ellipse(250,150,0,360,i,50);
        line(50,200,400,200);
        line(50,200,50,400);
        line(400,200,400,400);
        setfillstyle(SOLID_FILL, WHITE);
        floodfill(250, 150, WHITE);
        if(i==0){
            break;
        }
        i--;
        delay(100);
        cleardevice();
    }
        delay(100);
        cleardevice();
    }
    getch();
    closegraph();
    return 0;
}
```

Output:



9. Program to write name in Hindi.

```
#include<iostream>
#include<conio.h>
#include<dos.h>
#include<graphics.h>
using namespace std;
void curve(int x[4],int y[4])
{
   int px,py,i;
   double t;
   for(t=0.0;t<=1.0;t+=0.001)
   {
   px=(1-t)*(1-t)*(1-t)*x[0]+3*t*(1-t)*(1-t)*x[1]+3*t*t*(1-t)*x[2]+t*t*t*x[3];
   py=(1-t)*(1-t)*(1-t)*y[0]+3*t*(1-t)*(1-t)*y[1]+3*t*t*(1-t)*y[2]+t*t*t*y[3];
   putpixel(px,py,WHITE);
   }
}

int main()
```

```
{
   int gd=DETECT,gm;
   initgraph(&gd,&gm,"C:\\TC\\BGI");

     line(50,100,50,200);

     line (50, 160, 20, 110);

     int x[4],y[4],px,py,i;
     x[0] = 20; y[0] = 110;
     x[1] = 30; y[1] = 97;
     x[2] = 35; y[2] = 110;
     x[3] = 25; y[3] = 127;
     curve(x, y);

line (25, 127, 15, 160);
line (50, 160, 40, 190);

line (60, 100, 60, 200);

   x[0] = 60; y[0] = 100;
   x[1] = 60; y[1] = 80;
   x[2] = 50; y[2] = 80;
   x[3] = 50; y[3] = 100;
   curve(x, y);

   line (90, 100, 90, 140);

   x[0] = 90; y[0] = 140;
   x[1] = 70; y[1] = 140;
   x[2] = 70; y[2] = 200;
   x[3] = 90; y[3] = 200;
   curve(x, y);

   x[0] = 90; y[0] = 200;
   x[2] = 80; y[2] = 185;
   x[1] = 90; y[1] = 185;
   x[3] = 80; y[3] = 200;
   curve(x, y);
```

```
    line (45, 100, 95, 100);

    getch();
    closegraph();

}
```

Output: