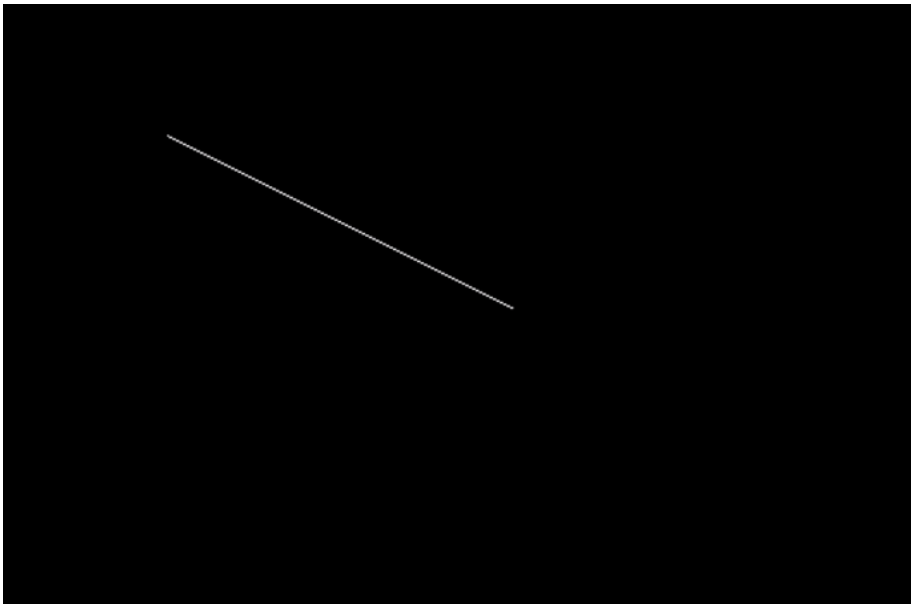
COMPUTER GRAPHICS

1. drawing with using simple equation y=mx+c

```
#include <graphics.h>
#include <conio.h>
void drawLineUsingEquation(int x1, int y1, int x2, int y2) {
    int dx = x2 - x1;
    int dy = y2 - y1;

    float m = (float)dy / dx;
    int x = x1;
    float y = y1;

    for (x = x1; x <= x2; x++) {
        putpixel(x, (int)(y + 0.5), WHITE);
        y += m;
    }
}
int main() {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");
    int x1 = 100, y1 = 100, x2 = 300, y2 = 200;
    drawLineUsingEquation(x1, y1, x2, y2);

    getch();
    closegraph();
    return 0;
```

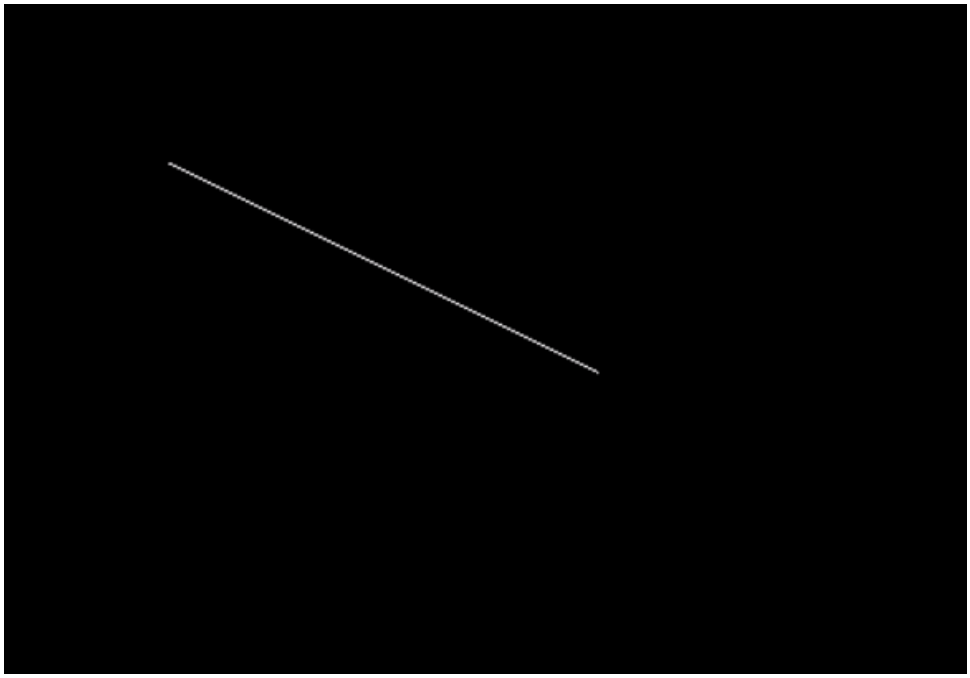
}



2. DDA Line Drawing algorithm

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

void DDA_Line(int x1, int y1, int x2, int y2) {
    int dx = x2 - x1;
    int dy = y2 - y1;

    int steps = abs(dx) > abs(dy) ? abs(dx) : abs(dy);
    float Xinc = dx / (float)steps;
    float Yinc = dy / (float)steps;



    float x = x1;
    float y = y1;
```

```
    for (int i = 0; i <= steps; i++) {
      putpixel((int)x, (int)y, WHITE);
      x += Xinc;
      y += Yinc;
    }
}

int main() {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");
    int x1 = 100, y1 = 100, x2 = 300, y2 = 200;
    DDA_Line(x1, y1, x2, y2);

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



3. Bresenham line drawing algorithm

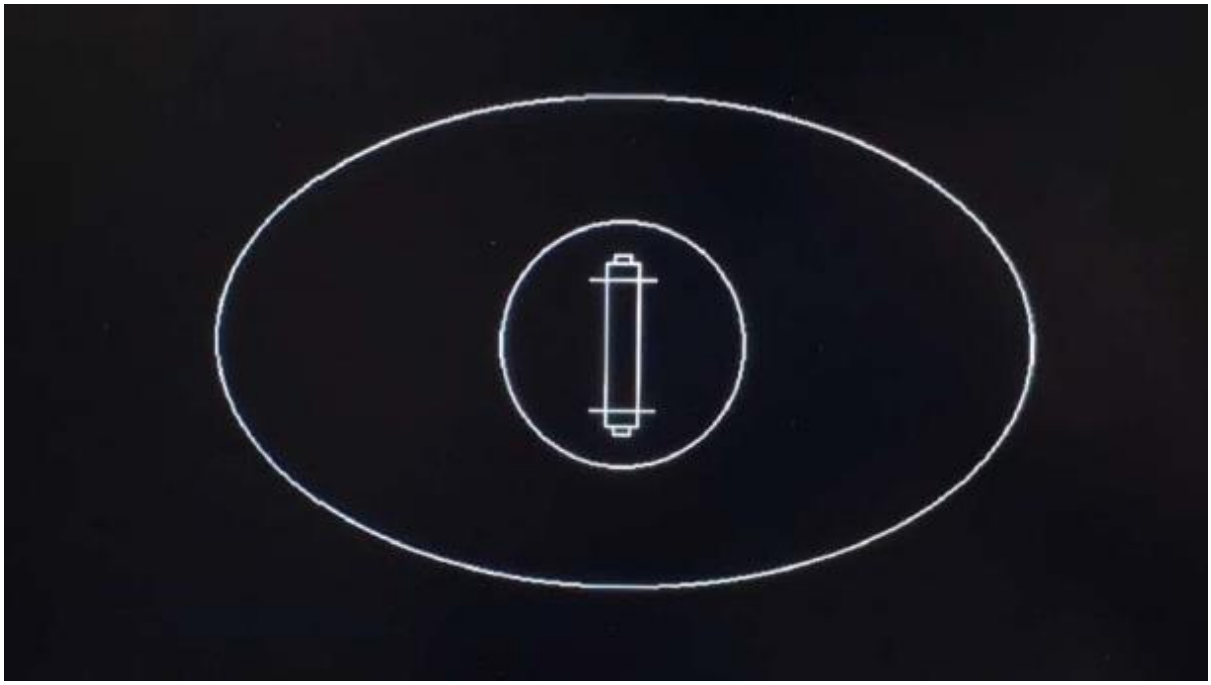
```
#include <graphics.h>
#include <conio.h>

void bresenhamLine(int x1, int y1, int x2, int y2) {
    int dx = x2 - x1;
    int dy = y2 - y1;
    int p = 2 * dy - dx;

    for (int x = x1, y = y1; x <= x2; x++) {
        putpixel(x, y, WHITE);
        if (p >= 0) {
            y++;
            p += 2 * (dy - dx);
        } else {
            p += 2 * dy;
        }
    }
}

int main() {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\Turboc3\\BGI");

    // Example: Drawing a line
    bresenhamLine(100, 100, 200, 200);
```

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

4. draw the cricket ground and football ground using computer graphics functions.

```
#include <graphics.h>
#include <conio.h>
void drawCricketGround()
{
int midX = getmaxx() / 2;
int midY = getmaxy() / 2;
ellipse(midX, midY, 0, 360, 250, 150);
rectangle(midX - 10, midY - 50, midX + 10, midY + 50);
rectangle(midX - 5, midY - 55, midX + 5, midY - 50);
rectangle(midX - 5, midY + 50, midX + 5, midY + 55);
ellipse(midX, midY, 0, 360, 75, 75);
line(midX - 20, midY - 60, midX + 20, midY - 60);
line(midX - 20, midY + 60, midX + 20, midY + 60);
}
int main()
{
    int gd = DETECT, gm;
    , initgraph(&gd, &gm, "C:\\Turboc3\\BGI");
    drawCricketGround();
```

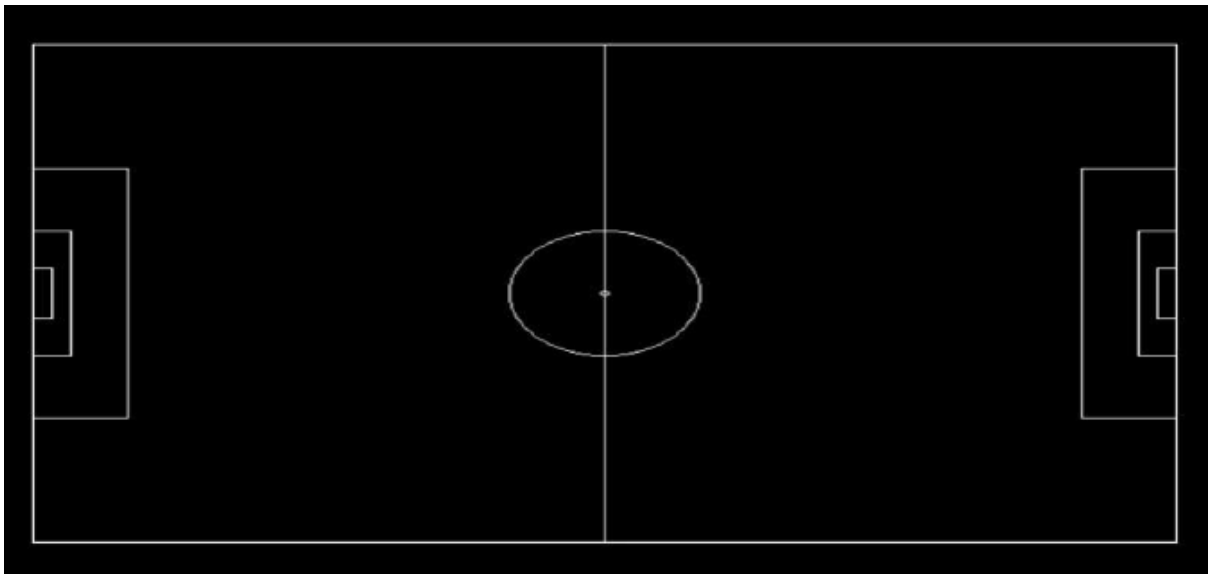
```
    getch();
    closegraph();
    return 0;
}
```



FOOTBALL GROUND:

```
#include <graphics.h>
 #include <conio.h>
void drawFootballGround()
{
    int midX = getmaxx() / 2;
    int midY = getmaxy() / 2;
    rectangle(midX - 300, midY - 200, midX + 300, midY + 200);
    circle(midX, midY, 50);
    circle(midX, midY, 2);
```

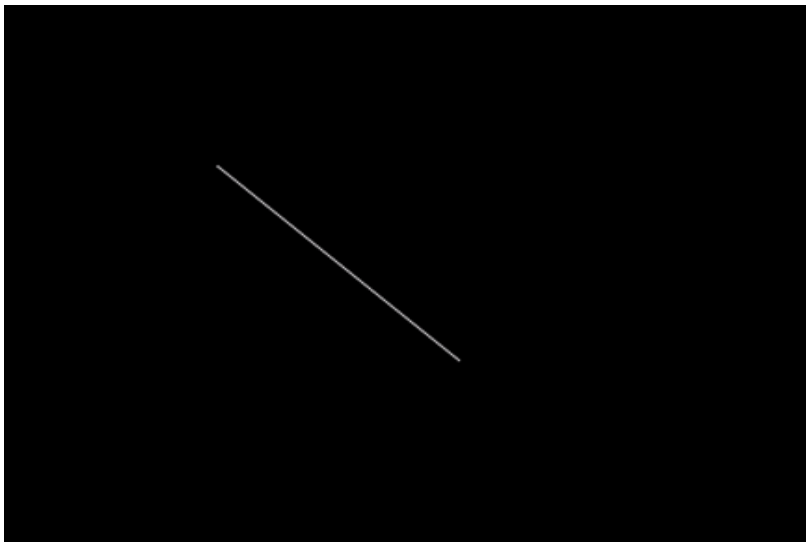
```
    rectangle(midX - 300, midY - 50, midX - 280, midY + 50);
    rectangle(midX + 280, midY - 50, midX + 300, midY + 50);
    rectangle(midX - 300, midY - 100, midX - 250, midY + 100);
    rectangle(midX + 250, midY - 100, midX + 300, midY + 100);
    rectangle(midX - 300, midY - 20, midX - 290, midY + 20);
    rectangle(midX + 290, midY - 20, midX + 300, midY + 20);
    line(midX, midY - 200, midX, midY + 200);
}
int main()
{
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\Turboc3\\BGI");
    drawFootballGround();
    getch();
    closegraph();
    return 0;
}
```



5. Midpoint line drawing

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

void midpointLine(int x1, int y1, int x2, int y2) {
   int dx = x2 - x1;
   int dy = y2 - y1;

   int d = dy - (dx / 2);
   int x = x1, y = y1;

   putpixel(x, y, WHITE);

   while (x < x2) {
      x++;
      if (d < 0) {
         d = d + dy;
         d = d + (dy - dx);
         y++;
      }
      putpixel(x, y, WHITE);
   }
}
int main() {
   int gd = DETECT, gm;
   initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");
```

```
    int x1 = 100, y1 = 100, x2 = 300, y2 = 200;
    midpointLine(x1, y1, x2, y2);
    getch();
    closegraph();
    return 0;
}
```

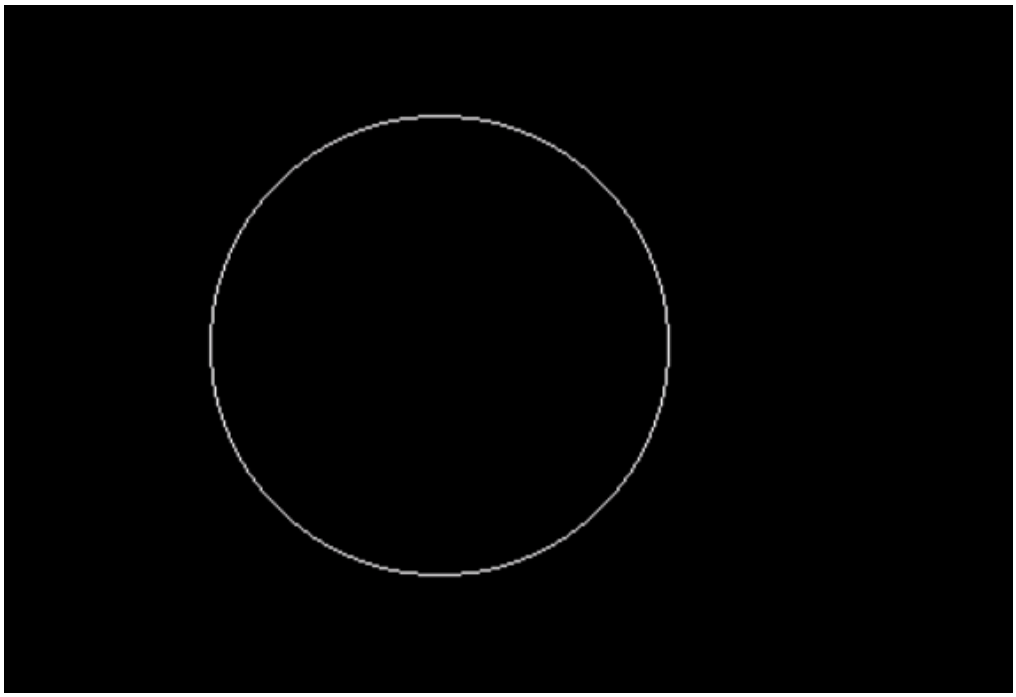


6. midpoint circle drawing

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

void midpointCircle(int xc, int yc, int r) {
    int x = 0, y = r;
    int d = 1 - r;  // Initial decision parameter

    // Draw the initial points of the circle
    putpixel(xc + x, yc + y, WHITE);
    putpixel(xc - x, yc + y, WHITE);
```

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

while (x < y) {
   x++;
   if (d < 0) {
      d = d + 2 * x + 1;  // Move horizontally
   } else {
      y--;
      d = d + 2 * (x - y) + 1;  // Move diagonally
   }


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

```
}

int main() {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");

    int xc = 250, yc = 250, r = 100;
    midpointCircle(xc, yc, r);

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

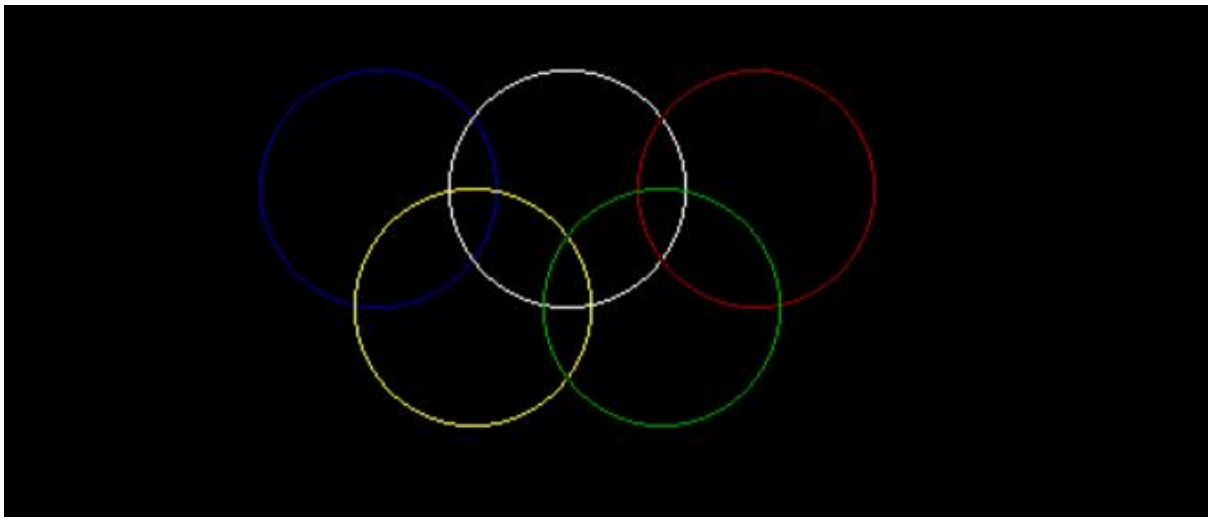


7. draw the olympic logo using graphics function

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

```
#include <conio.h>

void drawOlympicLogo() {

    int radius = 50;
    int gap = 10;



    setcolor(BLUE);
    circle(200, 150, radius);
    setcolor(WHITE);
    circle(280, 150, radius);
    setcolor(RED);
    circle(360, 150, radius);



    setcolor(YELLOW);
    circle(240, 200, radius);
    setcolor(GREEN);
    circle(320, 200, radius);
}

int main() {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");
```

```
    drawOlympicLogo();

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



8. Display your name using computer graphics in-built functions.

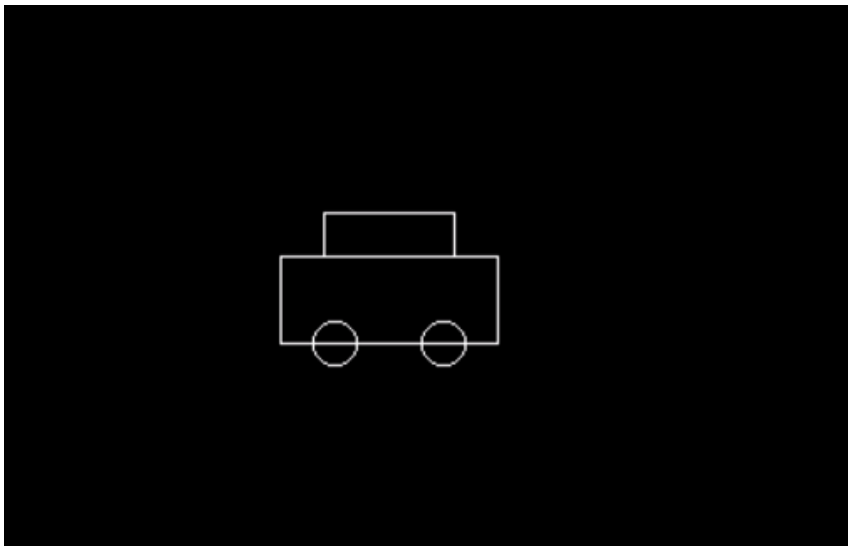
```
#include <graphics.h>
#include <conio.h>

int main() {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");


     cleardevice();

    setcolor(WHITE);
```

```
    settextstyle(DEFAULT_FONT, HORIZ_DIR, 3);

    outtextxy(100, 200, "VRUNDA RADADIYA");

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

9. make moving car 🚗 using computer graphics in-built functions.

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

void drawCar(int x, int y) {

    rectangle(x, y, x + 100, y + 40);
    rectangle(x + 20, y - 20, x + 80, y);

    circle(x + 25, y + 40, 10);
    circle(x + 75, y + 40, 10);
}

int main() {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");
```

```
    int x = 0, y = 300;
    while (x < getmaxx() - 100) {
        // Clear previous frame
        cleardevice();

        drawCar(x, y);

        delay(50);

        x += 5;
    }

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



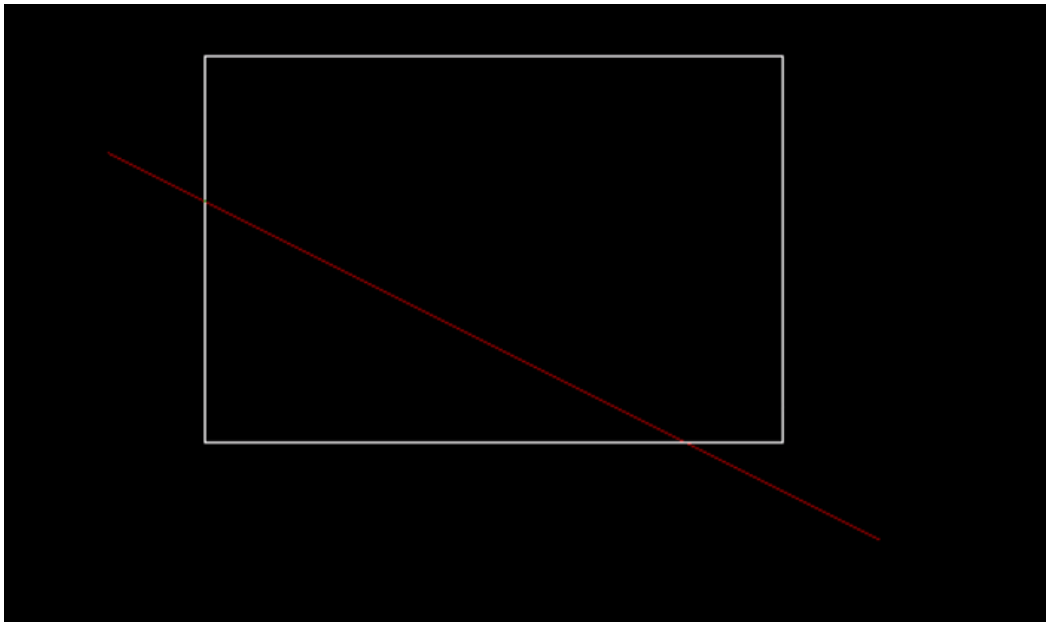
10. Line clipping algorithm

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

const int LEFT = 1, RIGHT = 2, BOTTOM = 4, TOP = 8;
int xmin = 100, ymin = 100, xmax = 400, ymax = 300;

int computeCode(int x, int y) {
   int code = 0;
   if (x < xmin) code |= LEFT;
   if (x > xmax) code |= RIGHT;
   if (y < ymin) code |= BOTTOM;
   if (y > ymax) code |= TOP;
   return code;
}

void Clip(int x1, int y1, int x2, int y2) {
   int code1 = computeCode(x1, y1);
   int code2 = computeCode(x2, y2);
   int accept = 0;

   while (1) {
   if ((code1 == 0) && (code2 == 0)) {
      accept = 1; // Both points inside
      break;
   } else if (code1 & code2) {
```

```
    break; // Both points share an outside region, reject
  } else {
    int code_out;
    int x, y;

    // Pick an outside point
    if (code1 != 0) code_out = code1;
    else code_out = code2;

    // Find intersection point using slope of the line
    if (code_out & TOP) {
    x = x1 + (x2 - x1) * (ymax - y1) / (y2 - y1);
    y = ymax;
    } else if (code_out & BOTTOM) {
    x = x1 + (x2 - x1) * (ymin - y1) / (y2 - y1);
    y = ymin;
    } else if (code_out & RIGHT) {
    y = y1 + (y2 - y1) * (xmax - x1) / (x2 - x1);
    x = xmax;
    } else if (code_out & LEFT) {
    y = y1 + (y2 - y1) * (xmin - x1) / (x2 - x1);
    x = xmin;
    }

    if (code_out == code1) {
```

```
        x1 = x;
        y1 = y;
        code1 = computeCode(x1, y1);
        } else {
        x2 = x;
        y2 = y;
        code2 = computeCode(x2, y2);
        }
    }
    }

    if (accept) {
    setcolor(WHITE);
    rectangle(xmin, ymin, xmax, ymax); // Draw clipping rectangle
    setcolor(GREEN);
    line(x1, y1, x2, y2); // Draw the clipped line
    }
}

int main() {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");

    int x1 = 50, y1 = 150, x2 = 450, y2 = 350;

    setcolor(RED);
```

```
    line(x1, y1, x2, y2);

    Clip(x1, y1, x2, y2);

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



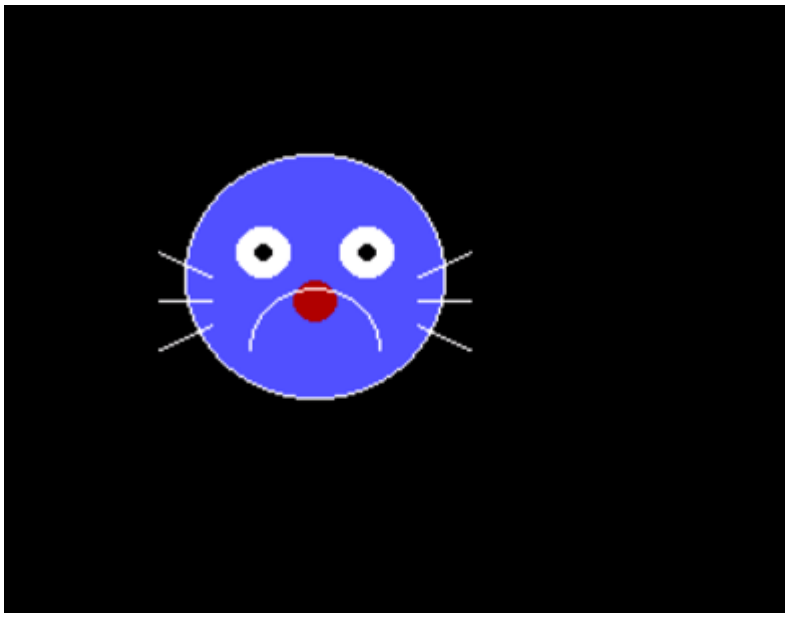
11. create any one cartoon character and apply any one transformation technique.

```
#include <graphics.h>
#include <conio.h>
#include <stdio.h>
#include <dos.h> // For delay function

void drawDoraemon(int x, int y) {
```

```
// Head
setcolor(WHITE);
circle(x, y, 50);
setfillstyle(SOLID_FILL, LIGHTBLUE);
floodfill(x, y, WHITE);

// Eyes
setcolor(WHITE);
circle(x - 20, y - 10, 10);
circle(x + 20, y - 10, 10);
setfillstyle(SOLID_FILL, WHITE);
floodfill(x - 20, y - 10, WHITE);
floodfill(x + 20, y - 10, WHITE);

// Pupils
setcolor(BLACK);
circle(x - 20, y - 10, 3);
circle(x + 20, y - 10, 3);
setfillstyle(SOLID_FILL, BLACK);
floodfill(x - 20, y - 10, BLACK);
floodfill(x + 20, y - 10, BLACK);

// Nose
setcolor(RED);
circle(x, y + 10, 8);
setfillstyle(SOLID_FILL, RED);
```

```
    floodfill(x, y + 10, RED);

    // Mouth
    setcolor(WHITE);
    arc(x, y + 30, 0, 180, 25);

    // Whiskers
    setcolor(WHITE);
    line(x - 40, y + 10, x - 60, y + 10); // Left whisker
    line(x + 40, y + 10, x + 60, y + 10); // Right whisker
    line(x - 40, y + 20, x - 60, y + 30); // Left bottom whisker
    line(x + 40, y + 20, x + 60, y + 30); // Right bottom whisker
    line(x - 40, y, x - 60, y - 10); // Left top whisker
    line(x + 40, y, x + 60, y - 10); // Right top whisker
}

int main() {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\Turboc3\\BGI");

    int x = 100, y = 200;

    while (!kbhit()) {  // Loop until a key is pressed
        cleardevice();
            drawDoraemon(x, y);
```

```
        delay(100);

        // Move Doraemon to the right
        x += 5;
        if (x > getmaxx()) {
            x = -50;
        }
    }

    closegraph();
    return 0;
}
```

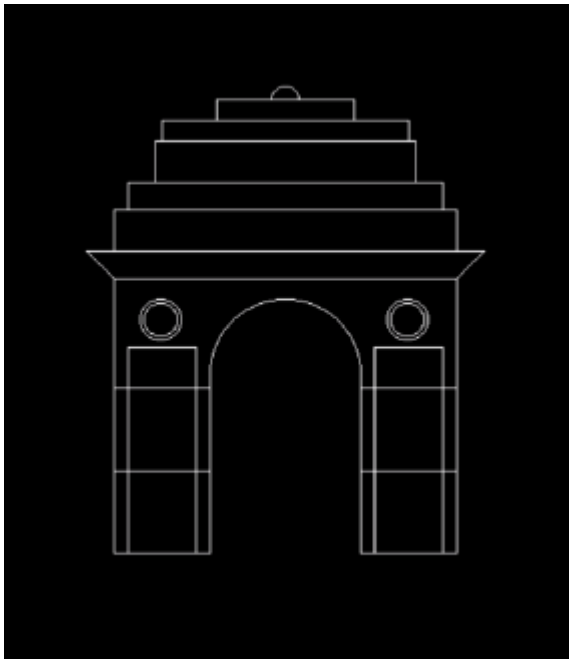


12. Draw India gate using computer graphics functions

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

```
void main()
{
   int gd = DETECT, gm;
   initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");
   line(100, 400, 170, 400);
   line(100, 400, 100, 200);
   line(100, 200, 350, 200);
   line(110, 400, 110, 250);
   line(110, 250, 160, 250);
   line(160, 400, 160, 250);
   line(100, 280, 170, 280);
   circle(134, 230, 15);
   circle(134, 230, 12);
   line(100, 340, 170, 340);
   line(350, 200, 350, 400);
   line(350, 400, 280, 400);
   line(170, 400, 170, 270);
   line(280, 400, 280, 270);
   line(290, 400, 290, 250);
   line(340, 400, 340, 250);
   line(290, 250, 340, 250);
   line(280, 280, 350, 280);
   line(280, 340, 350, 340);
   circle(314, 230, 15);
   circle(314, 230, 12);
   arc(225, 270, 0, 180, 55);
```

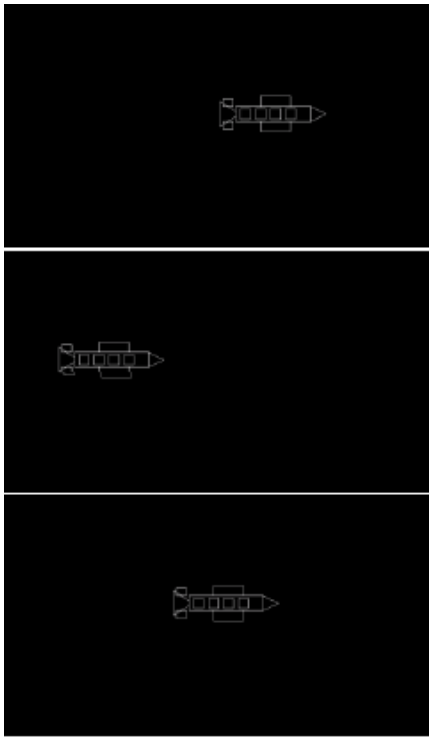
```
    line(100, 200, 80, 180);
    line(350, 200, 370, 180);
    line(80, 180, 370, 180);
    line(100, 180, 100, 150);
    line(350, 180, 350, 150);
    line(100, 150, 350, 150);
    line(110, 150, 110, 130);
    line(340, 150, 340, 130);
    line(110, 130, 340, 130);
    line(130, 130, 130, 100);
    line(320, 130, 320, 100);
    line(130, 100, 320, 100);
    line(135, 100, 135, 85);
    line(315, 100, 315, 85);
    line(135, 85, 315, 85);
    line(175, 85, 175, 70);
    line(275, 85, 275, 70);
    line(175, 70, 275, 70);
    arc(225, 70, 0, 180, 10);
    getch();
    closegraph();
}
```



13. flying plane ✈ animation

```
#include <stdio.h>
#include <graphics.h>
#include <conio.h>
#include <dos.h>
void drawPlane(int x, int y) {
rectangle(x, y, x + 150, y + 30);
line(x + 150, y, x + 180, y + 15);
line(x + 150, y + 30, x + 180, y + 15);
rectangle(x + 50, y - 20, x + 110, y);
rectangle(x + 50, y + 30, x + 110, y + 50);
line(x, y + 10, x - 30, y - 10);
line(x, y + 20, x - 30, y + 40);
line(x - 30, y - 10, x - 30, y + 40);
rectangle(x - 25, y - 15, x - 5, y);
rectangle(x - 25, y + 30, x - 5, y + 45);
```

```
rectangle(x + 10, y + 5, x + 30, y + 25);
rectangle(x + 40, y + 5, x + 60, y + 25);
rectangle(x + 70, y + 5, x + 90, y + 25);
rectangle(x + 100, y + 5, x + 120, y + 25);
}
int main() {
int gd = DETECT, gm;
int x = 0, y = 200;
initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");
while (!kbhit()) {
cleardevice();
drawPlane(x, y);
x += 5;
delay(100);
if (x > getmaxx()) {
x = -180;
}
}
getch();
closegraph();
return 0;
}
```



14. Draw solar system using computer graphics functions

```
#include <stdio.h>#include<conio.h>#include<graphics.h>
int main()
{
   int gd = DETECT, gm, i;
   initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");
   rectangle(100, 100, 400, 400);
   rectangle(100, 100, 220, 220);
   rectangle(120, 120, 200, 200);
   rectangle(280, 100, 400, 220);
   rectangle(100, 280, 220, 400);
   rectangle(280, 280, 400, 400);
   rectangle(220, 220, 280, 280);
   rectangle(300, 120, 380, 200);
   rectangle(120, 300, 200, 380);
```

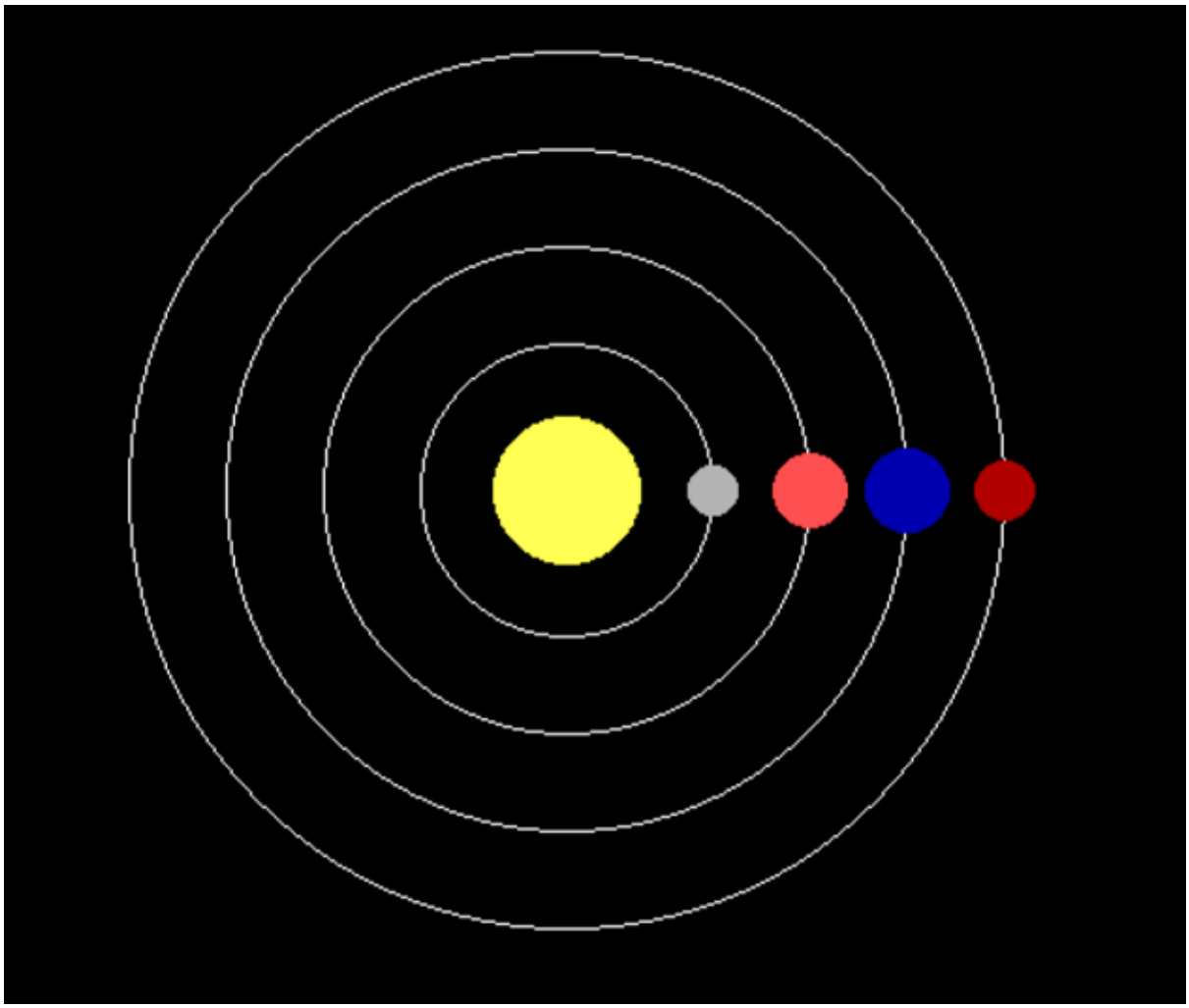
```
rectangle(300, 300, 380, 380);
line(220, 220, 280, 280);
line(280, 220, 220, 280);
for (i = 0; i < 280; i = i + 20)
{
   if ((i == 120 || i == 140 || i == 160))
   {
      if (i == 120)
      {
         line(240, 100, 240, 220);
         line(240, 280, 240, 400);
      }
      if (i == 140)
      {
         line(260, 100, 260, 220);
         line(260, 280, 260, 400);
      }
   }
   else
   {
      line(120 + i, 220, 120 + i, 280);
   }
}
for (i = 0; i < 280; i = i + 20)
{
   if ((i == 120 || i == 140 || i == 160))
```

```
    {
        if (i == 120)
        {
            line(100, 240, 220, 240);
            line(280, 240, 400, 240);
        }
        if (i == 140)
        {
            line(100, 260, 220, 260);
            line(280, 260, 400, 260);
        }
    }
    else
    {
        line(220, 120 + i, 280, 120 + i);
    }
}
circle(140, 140, 10);
circle(180, 140, 10);
circle(140, 180, 10);
circle(180, 180, 10);
circle(320, 140, 10);
circle(360, 140, 10);
circle(320, 180, 10);
circle(360, 180, 10);
circle(140, 320, 10);
```

```
circle(180, 320, 10);
circle(140, 360, 10);
circle(180, 360, 10);
circle(320, 320, 10);
circle(360, 320, 10);
circle(320, 360, 10);
circle(360, 360, 10);
setfillstyle(SOLID_FILL, LIGHTBLUE);
floodfill(110, 290, WHITE);
floodfill(140, 320, WHITE);
floodfill(180, 320, WHITE);
floodfill(140, 360, WHITE);
floodfill(180, 360, WHITE);
floodfill(221, 361, WHITE);
floodfill(241, 361, WHITE);
floodfill(241, 341, WHITE);
floodfill(241, 321, WHITE);
floodfill(241, 301, WHITE);
floodfill(241, 281, WHITE);
floodfill(241, 261, WHITE);
setfillstyle(SOLID_FILL, YELLOW);
floodfill(110, 110, WHITE);
floodfill(140, 140, WHITE);
floodfill(180, 140, WHITE);
floodfill(140, 180, WHITE);
floodfill(180, 180, WHITE);
```

```
floodfill(121, 221, WHITE);
floodfill(121, 241, WHITE);
floodfill(141, 241, WHITE);
floodfill(161, 241, WHITE);
floodfill(181, 241, WHITE);
floodfill(201, 241, WHITE);
floodfill(221, 241, WHITE);
setfillstyle(SOLID_FILL, LIGHTRED);
floodfill(290, 110, WHITE);
floodfill(320, 140, WHITE);
floodfill(360, 140, WHITE);
floodfill(320, 180, WHITE);
floodfill(360, 180, WHITE);
floodfill(261, 121, WHITE);
floodfill(241, 121, WHITE);
floodfill(241, 141, WHITE);
floodfill(241, 161, WHITE);
floodfill(241, 181, WHITE);
floodfill(241, 201, WHITE);
floodfill(241, 221, WHITE);
setfillstyle(SOLID_FILL, GREEN);
floodfill(290, 290, WHITE);
floodfill(320, 320, WHITE);
floodfill(360, 320, WHITE);
floodfill(320, 360, WHITE);
floodfill(360, 360, WHITE);
```

```
    floodfill(361, 261, WHITE);
    floodfill(361, 241, WHITE);
    floodfill(341, 241, WHITE);
    floodfill(321, 241, WHITE);
    floodfill(301, 241, WHITE);
    floodfill(281, 241, WHITE);
    floodfill(261, 241, WHITE);
    getch();
    closegraph();
    return 0;
}
```



15. design ludo board using computer graphics functions.

```
#include <stdio.h>#include<conio.h>#include<graphics.h>
int main()
{
   int gd = DETECT, gm, i;
   initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");
   rectangle(100, 100, 400, 400);
   rectangle(100, 100, 220, 220);
   rectangle(120, 120, 200, 200);
   rectangle(280, 100, 400, 220);
   rectangle(100, 280, 220, 400);
   rectangle(280, 280, 400, 400);
   rectangle(220, 220, 280, 280);
   rectangle(300, 120, 380, 200);
   rectangle(120, 300, 200, 380);
   rectangle(300, 300, 380, 380);
   line(220, 220, 280, 280);
   line(280, 220, 220, 280);
   for (i = 0; i < 280; i = i + 20)
   {
      if ((i == 120 || i == 140 || i == 160))
      {
         if (i == 120)
         {
            line(240, 100, 240, 220);
            line(240, 280, 240, 400);
```

```
        }
        if (i == 140)
        {
            line(260, 100, 260, 220);
            line(260, 280, 260, 400);
        }
    }
    else
    {
        line(120 + i, 220, 120 + i, 280);
    }
}
for (i = 0; i < 280; i = i + 20)
{
    if ((i == 120 || i == 140 || i == 160))
    {
        if (i == 120)
        {
            line(100, 240, 220, 240);
            line(280, 240, 400, 240);
        }
        if (i == 140)
        {
            line(100, 260, 220, 260);
            line(280, 260, 400, 260);
        }
```

```
    }
    else
    {
        line(220, 120 + i, 280, 120 + i);
    }
}
circle(140, 140, 10);
circle(180, 140, 10);
circle(140, 180, 10);
circle(180, 180, 10);
circle(320, 140, 10);
circle(360, 140, 10);
circle(320, 180, 10);
circle(360, 180, 10);
circle(140, 320, 10);
circle(180, 320, 10);
circle(140, 360, 10);
circle(180, 360, 10);
circle(320, 320, 10);
circle(360, 320, 10);
circle(320, 360, 10);
circle(360, 360, 10);
setfillstyle(SOLID_FILL, LIGHTBLUE);
floodfill(110, 290, WHITE);
floodfill(140, 320, WHITE);
floodfill(180, 320, WHITE);
```

```
floodfill(140, 360, WHITE);
floodfill(180, 360, WHITE);
floodfill(221, 361, WHITE);
floodfill(241, 361, WHITE);
floodfill(241, 341, WHITE);
floodfill(241, 321, WHITE);
floodfill(241, 301, WHITE);
floodfill(241, 281, WHITE);
floodfill(241, 261, WHITE);
setfillstyle(SOLID_FILL, YELLOW);
floodfill(110, 110, WHITE);
floodfill(140, 140, WHITE);
floodfill(180, 140, WHITE);
floodfill(140, 180, WHITE);
floodfill(180, 180, WHITE);
floodfill(121, 221, WHITE);
floodfill(121, 241, WHITE);
floodfill(141, 241, WHITE);
floodfill(161, 241, WHITE);
floodfill(181, 241, WHITE);
floodfill(201, 241, WHITE);
floodfill(221, 241, WHITE);
setfillstyle(SOLID_FILL, LIGHTRED);
floodfill(290, 110, WHITE);
floodfill(320, 140, WHITE);
floodfill(360, 140, WHITE);
```

```
    floodfill(320, 180, WHITE);
    floodfill(360, 180, WHITE);
    floodfill(261, 121, WHITE);
    floodfill(241, 121, WHITE);
    floodfill(241, 141, WHITE);
    floodfill(241, 161, WHITE);
    floodfill(241, 181, WHITE);
    floodfill(241, 201, WHITE);
    floodfill(241, 221, WHITE);
    setfillstyle(SOLID_FILL, GREEN);
    floodfill(290, 290, WHITE);
    floodfill(320, 320, WHITE);
    floodfill(360, 320, WHITE);
    floodfill(320, 360, WHITE);
    floodfill(360, 360, WHITE);
    floodfill(361, 261, WHITE);
    floodfill(361, 241, WHITE);
    floodfill(341, 241, WHITE);
    floodfill(321, 241, WHITE);
    floodfill(301, 241, WHITE);
    floodfill(281, 241, WHITE);
    floodfill(261, 241, WHITE);
    getch();
    closegraph();
    return 0;
}
```