Lab – 7

14. Draw solar system using computer graphics functions.

#include <graphics.h>

#include <conio.h>

#include <math.h>

void drawPlanet(int x, int y, int radius, int color, char name[])

{

    setcolor(color);

    setfillstyle(SOLID\_FILL, color);

    fillellipse(x, y, radius, radius);

    setcolor(WHITE);

    outtextxy(x - radius - 2, y + radius + 5, name);

}

void drawSolarSystem(int centerX, int centerY)

{

    setcolor(YELLOW);

    setfillstyle(SOLID\_FILL, YELLOW);

    fillellipse(centerX, centerY, 50, 50);

    outtextxy(centerX + 60, centerY - 10, "Sun");

    setcolor(WHITE);

    circle(centerX, centerY, 100);

    circle(centerX, centerY, 150);

    circle(centerX, centerY, 200);

    circle(centerX, centerY, 250);

    circle(centerX, centerY, 300);

    circle(centerX, centerY, 350);

    circle(centerX, centerY, 400);

    circle(centerX, centerY, 450);

    drawPlanet(centerX + 100, centerY, 5, LIGHTGRAY, "Mercury");

    drawPlanet(centerX + 150, centerY, 12, LIGHTGREEN, "Venus");

    drawPlanet(centerX + 200, centerY, 15, BLUE, "Earth");

    drawPlanet(centerX + 250, centerY, 10, RED, "Mars");

    drawPlanet(centerX + 300, centerY, 20, BROWN, "Jupiter");

    drawPlanet(centerX + 350, centerY, 18, CYAN, "Saturn");

    drawPlanet(centerX + 400, centerY, 15, LIGHTBLUE, "Uranus");

    drawPlanet(centerX + 450, centerY, 14, BLUE, "Neptune");

}

int main()

{

    int gd = DETECT, gm;

    initgraph(&gd, &gm, NULL);

    int centerX = getmaxx() / 2;

    int centerY = getmaxy() / 2;

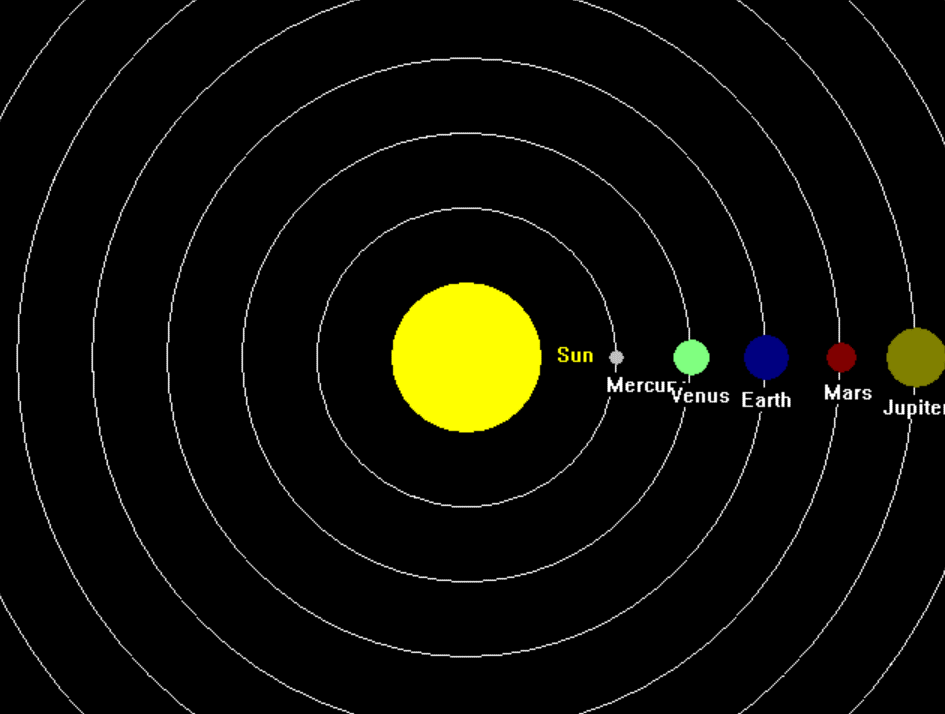
    drawSolarSystem(centerX, centerY);

    getch();

    closegraph();

    return 0;

}



15. ⁠design ludo board using computer graphics functions.

#include <graphics.h>

#include <conio.h>

void drawLudoBoard(int startX, int startY, int cellSize)

{

*// Colors for each player's base*

    int redBaseColor = RED;

    int greenBaseColor = GREEN;

    int yellowBaseColor = YELLOW;

    int blueBaseColor = BLUE;

    int pathColor = LIGHTGRAY;

*// Draw the red base*

    setfillstyle(SOLID\_FILL, redBaseColor);

    rectangle(startX, startY, startX + 4 \* cellSize, startY + 4 \* cellSize);

    floodfill(startX + 1, startY + 1, WHITE);

*// Draw the green base*

    setfillstyle(SOLID\_FILL, greenBaseColor);

    rectangle(startX + 5 \* cellSize, startY, startX + 9 \* cellSize, startY + 4 \* cellSize);

    floodfill(startX + 5 \* cellSize + 1, startY + 1, WHITE);

*// Draw the yellow base*

    setfillstyle(SOLID\_FILL, yellowBaseColor);

    rectangle(startX + 5 \* cellSize, startY + 5 \* cellSize, startX + 9 \* cellSize, startY + 9 \* cellSize);

    floodfill(startX + 5 \* cellSize + 1, startY + 5 \* cellSize + 1, WHITE);

*// Draw the blue base*

    setfillstyle(SOLID\_FILL, blueBaseColor);

    rectangle(startX, startY + 5 \* cellSize, startX + 4 \* cellSize, startY + 9 \* cellSize);

    floodfill(startX + 1, startY + 5 \* cellSize + 1, WHITE);

*// Draw paths and center*

    setfillstyle(SOLID\_FILL, pathColor);

*// Vertical path on the left (red to blue)*

    rectangle(startX + 4 \* cellSize, startY, startX + 5 \* cellSize, startY + 5 \* cellSize);

    floodfill(startX + 4 \* cellSize + 1, startY + 1, WHITE);

*// Horizontal path on the top (red to green)*

    rectangle(startX, startY + 4 \* cellSize, startX + 5 \* cellSize, startY + 5 \* cellSize);

    floodfill(startX + 1, startY + 4 \* cellSize + 1, WHITE);

*// Vertical path on the right (green to yellow)*

    rectangle(startX + 5 \* cellSize, startY + 4 \* cellSize, startX + 9 \* cellSize, startY + 5 \* cellSize);

    floodfill(startX + 5 \* cellSize + 1, startY + 4 \* cellSize + 1, WHITE);

*// Horizontal path on the bottom (blue to yellow)*

    rectangle(startX + 4 \* cellSize, startY + 5 \* cellSize, startX + 5 \* cellSize, startY + 9 \* cellSize);

    floodfill(startX + 4 \* cellSize + 1, startY + 5 \* cellSize + 1, WHITE);

*// Center square (safe zone)*

    setfillstyle(SOLID\_FILL, WHITE);

    rectangle(startX + 4 \* cellSize, startY + 4 \* cellSize, startX + 5 \* cellSize, startY + 5 \* cellSize);

    floodfill(startX + 4 \* cellSize + 1, startY + 4 \* cellSize + 1, WHITE);

*// Drawing the diagonal lines in the center*

    setcolor(WHITE);

    line(startX + 4 \* cellSize, startY + 4 \* cellSize, startX + 5 \* cellSize, startY + 5 \* cellSize);

    line(startX + 4 \* cellSize, startY + 5 \* cellSize, startX + 5 \* cellSize, startY + 4 \* cellSize);

    setfillstyle(SOLID\_FILL, WHITE);

*// Red Quadrant*

    for (int i = 0; i < 2; i++)

    {

        for (int j = 0; j < 2; j++)

        {

            fillellipse(startX + (i \* 2 + 1) \* cellSize, startY + (j \* 2 + 1) \* cellSize, cellSize / 3, cellSize / 3);

        }

    }

*// Green Quadrant*

    for (int i = 0; i < 2; i++)

    {

        for (int j = 0; j < 2; j++)

        {

            fillellipse(startX + (5 + i \* 2 + 1) \* cellSize, startY + (j \* 2 + 1) \* cellSize, cellSize / 3, cellSize / 3);

        }

    }

*// Yellow Quadrant*

    for (int i = 0; i < 2; i++)

    {

        for (int j = 0; j < 2; j++)

        {

            fillellipse(startX + (5 + i \* 2 + 1) \* cellSize, startY + (5 + j \* 2 + 1) \* cellSize, cellSize / 3, cellSize / 3);

        }

    }

*// Blue Quadrant*

    for (int i = 0; i < 2; i++)

    {

        for (int j = 0; j < 2; j++)

        {

            fillellipse(startX + (i \* 2 + 1) \* cellSize, startY + (5 + j \* 2 + 1) \* cellSize, cellSize / 3, cellSize / 3);

        }

    }

}

int main()

{

    int gd = DETECT, gm;

    initgraph(&gd, &gm, "");

    int startX = 200, startY = 100;

    int cellSize = 30;

    drawLudoBoard(startX, startY, cellSize);

    getch();

    closegraph();

    return 0;

}

