LAB-12

WEB PROGRAMMING

Name: Vinit Yadav

Reg No: 23BCE1767

Q1)

HTML

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>23BCE1767</title>

</head>

<body>

    <canvas id="scene" width="800" height="400"></canvas>

    <script src="script.js"></script>

</body>

</html>

JS

window.onload = function() {

    const canvas = document.getElementById("scene");

    const ctx = canvas.getContext("2d");

    // Draw Pond (Oval)

    ctx.fillStyle = "#1E90FF";

    ctx.beginPath();

    ctx.ellipse(250, 300, 120, 60, 0, 0, Math.PI \* 2);

    ctx.fill();

    // Draw Boat (Quadrilateral with curved edges)

    ctx.fillStyle = "#8B4513";

    ctx.beginPath();

    ctx.moveTo(200, 270);

    ctx.quadraticCurveTo(250, 300, 300, 270);

    ctx.lineTo(270, 250);

    ctx.lineTo(230, 250);

    ctx.closePath();

    ctx.fill();

    // Draw Duck (Two Circles)

    ctx.fillStyle = "#FFD700";

    ctx.beginPath();

    ctx.arc(180, 290, 15, 0, Math.PI \* 2);

    ctx.fill();

    ctx.beginPath();

    ctx.arc(165, 280, 10, 0, Math.PI \* 2);

    ctx.fill();

    // Draw Sun (Large Circle with Lines)

    ctx.fillStyle = "#FFA500";

    ctx.beginPath();

    ctx.arc(50, 50, 30, 0, Math.PI \* 2);

    ctx.fill();

    for (let i = 0; i < 12; i++) {

        let angle = (i \* Math.PI) / 6;

        let x1 = 50 + Math.cos(angle) \* 30;

        let y1 = 50 + Math.sin(angle) \* 30;

        let x2 = 50 + Math.cos(angle) \* 40;

        let y2 = 50 + Math.sin(angle) \* 40;

        ctx.beginPath();

        ctx.moveTo(x1, y1);

        ctx.lineTo(x2, y2);

        ctx.stroke();

    }

    // Draw House (Rectangle + Triangle)

    ctx.fillStyle = "#8B0000";

    ctx.fillRect(320, 200, 80, 80);

    ctx.fillStyle = "#A52A2A";

    ctx.beginPath();

    ctx.moveTo(310, 200);

    ctx.lineTo(360, 150);

    ctx.lineTo(410, 200);

    ctx.closePath();

    ctx.fill();

    // Draw Flower (Ellipse, Line, Curved Leaves)

    ctx.fillStyle = "#008000";

    ctx.fillRect(100, 320, 5, 50);

    ctx.beginPath();

    ctx.moveTo(100, 340);

    ctx.quadraticCurveTo(90, 330, 80, 340);

    ctx.quadraticCurveTo(90, 350, 100, 340);

    ctx.fill();

    ctx.beginPath();

    ctx.moveTo(105, 350);

    ctx.quadraticCurveTo(115, 340, 125, 350);

    ctx.quadraticCurveTo(115, 360, 105, 350);

    ctx.fill();

    ctx.fillStyle = "#FF0000";

    ctx.beginPath();

    ctx.ellipse(100, 310, 15, 10, 0, 0, Math.PI \* 2);

    ctx.fill();

    // Draw Stones (Multiple Small Circles)

    ctx.fillStyle = "#808080";

    for (let i = 0; i < 5; i++) {

        let x = 220 + i \* 15;

        let y = 320 + (i % 2) \* 5;

        ctx.beginPath();

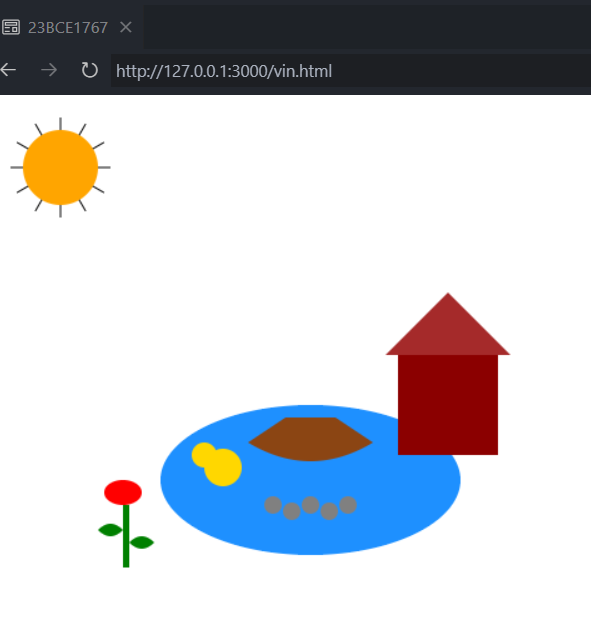
        ctx.arc(x, y, 7, 0, Math.PI \* 2);

        ctx.fill();

    }

};

Output:



Q2)

Code: <!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>23BCE1767</title>

</head>

<body>

    <canvas id="scene" width="800" height="400"></canvas>

    <script src="q2.js"></script>

</body>

</html>

Js

const canvas = document.getElementById("scene");

const ctx = canvas.getContext("2d");

let boatX = 50; // Initial x position of the boat

const boatSpeed = 2; // Speed of movement

function drawScene() {

    ctx.clearRect(0, 0, canvas.width, canvas.height); // Clear canvas

    // Draw Pond (Oval)

    ctx.fillStyle = "#1E90FF";

    ctx.beginPath();

    ctx.ellipse(250, 300, 200, 50, 0, 0, Math.PI \* 2);

    ctx.fill();

    // Draw Boat (Polygon - Quadrilateral with curved edges)

    ctx.fillStyle = "brown";

    ctx.beginPath();

    ctx.moveTo(boatX, 290);

    ctx.lineTo(boatX + 50, 270);

    ctx.lineTo(boatX + 100, 290);

    ctx.lineTo(boatX + 80, 310);

    ctx.lineTo(boatX + 20, 310);

    ctx.closePath();

    ctx.fill();

    // Draw Duck (Two Circles)

    ctx.fillStyle = "yellow";

    ctx.beginPath();

    ctx.arc(180, 280, 10, 0, Math.PI \* 2);

    ctx.arc(195, 290, 15, 0, Math.PI \* 2);

    ctx.fill();

    // Draw Sun (Large Circle with outward lines)

    ctx.fillStyle = "yellow";

    ctx.beginPath();

    ctx.arc(400, 50, 30, 0, Math.PI \* 2);

    ctx.fill();

    for (let i = 0; i < 12; i++) {

        let angle = (Math.PI / 6) \* i;

        ctx.beginPath();

        ctx.moveTo(400 + Math.cos(angle) \* 30, 50 + Math.sin(angle) \* 30);

        ctx.lineTo(400 + Math.cos(angle) \* 40, 50 + Math.sin(angle) \* 40);

        ctx.stroke();

    }

    // Draw House (Rectangle + Triangle)

    ctx.fillStyle = "red";

    ctx.fillRect(300, 250, 60, 50);

    ctx.fillStyle = "brown";

    ctx.beginPath();

    ctx.moveTo(300, 250);

    ctx.lineTo(330, 220);

    ctx.lineTo(360, 250);

    ctx.closePath();

    ctx.fill();

    // Draw Flower (Ellipse with stem, leaves, petals)

    ctx.fillStyle = "green";

    ctx.fillRect(80, 280, 5, 40);

    ctx.beginPath();

    ctx.arc(82, 275, 5, 0, Math.PI \* 2);

    ctx.fill();

    // Draw Stones (Small Circles)

    ctx.fillStyle = "gray";

    for (let i = 0; i < 5; i++) {

        ctx.beginPath();

        ctx.arc(250 + i \* 15, 350, 7, 0, Math.PI \* 2);

        ctx.fill();

    }

}

function animate() {

    boatX += boatSpeed;

    if (boatX > canvas.width) {

        boatX = -100; // Reset when it moves off-screen

    }

    drawScene();

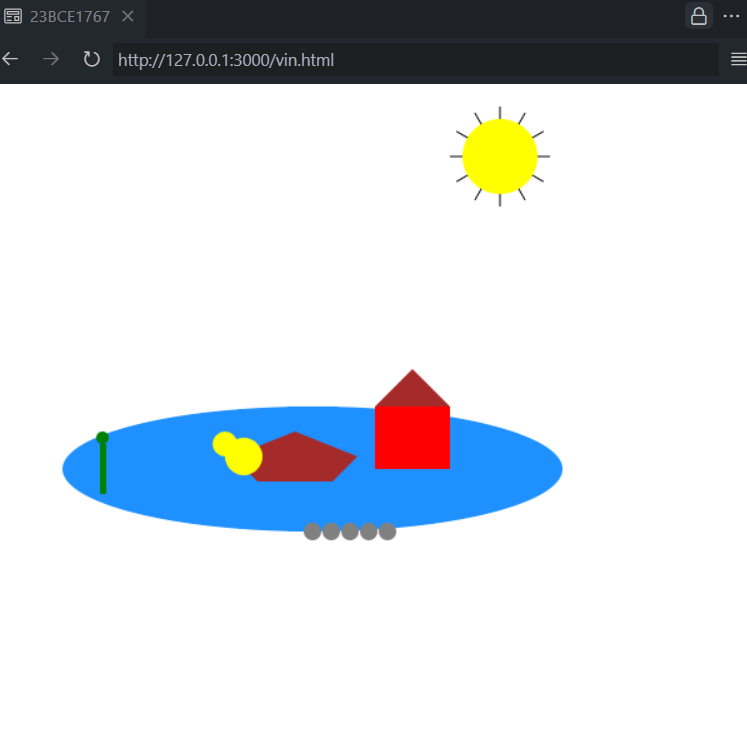
    requestAnimationFrame(animate);

}

drawScene(); // Initial draw

animate(); // Start animation

Output:



Q3)

Code:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>23BCE1767</title>

</head>

<body>

    <canvas id="scene" height="1000"></canvas>

    <script src="q3.js"></script>

</body>

</html>

Js

const canvas = document.getElementById("scene");

const ctx = canvas.getContext("2d");

const radius = canvas.width / 3;

ctx.translate(radius, radius);

function drawClock() {

    ctx.clearRect(-radius, -radius, canvas.width, canvas.height);

    drawFace(ctx, radius);

    drawNumbers(ctx, radius);

    drawTime(ctx, radius);

}

function drawFace(ctx, radius) {

    ctx.beginPath();

    ctx.arc(0, 0, radius - 5, 0, 2 \* Math.PI);

    ctx.fillStyle = "white";

    ctx.fill();

    ctx.lineWidth = 5;

    ctx.stroke();

    ctx.beginPath();

    ctx.arc(0, 0, 5, 0, 2 \* Math.PI);

    ctx.fillStyle = "black";

    ctx.fill();

}

function drawNumbers(ctx, radius) {

    ctx.font = "16px Arial";

    ctx.textAlign = "center";

    ctx.textBaseline = "middle";

    for (let num = 1; num <= 12; num++) {

        let angle = (num \* Math.PI) / 6;

        let x = (radius - 20) \* Math.cos(angle - Math.PI / 2);

        let y = (radius - 20) \* Math.sin(angle - Math.PI / 2);

        ctx.fillText(num, x, y);

    }

}

function drawTime(ctx, radius) {

    let now = new Date();

    let hours = now.getHours() % 12;

    let minutes = now.getMinutes();

    let seconds = now.getSeconds();

    drawHand(ctx, (hours \* 30 + minutes / 2) \* (Math.PI / 180), radius \* 0.5, 6);

    drawHand(ctx, (minutes \* 6 + seconds / 10) \* (Math.PI / 180), radius \* 0.8, 4);

    drawHand(ctx, seconds \* 6 \* (Math.PI / 180), radius \* 0.9, 2, "red");

}

function drawHand(ctx, angle, length, width, color = "black") {

    ctx.beginPath();

    ctx.lineWidth = width;

    ctx.strokeStyle = color;

    ctx.moveTo(0, 0);

    ctx.lineTo(length \* Math.cos(angle - Math.PI / 2), length \* Math.sin(angle - Math.PI / 2));

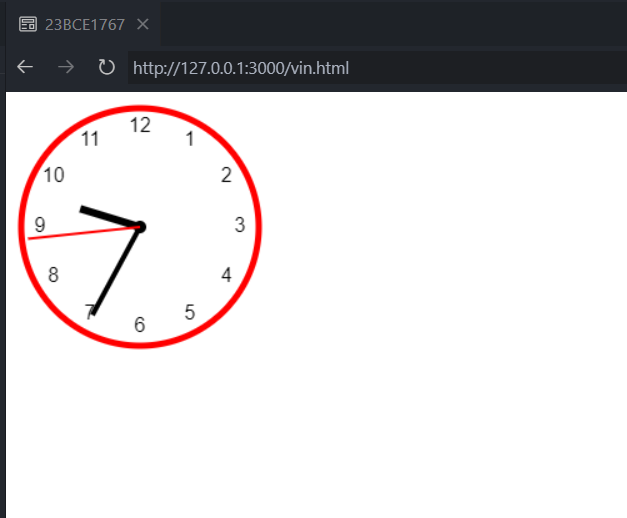
    ctx.stroke();

}

setInterval(drawClock, 1000);

drawClock();

Output:



Q4)

Code:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>23BCE1767</title>

</head>

<body>

    <div id="barChart"></div>

<div id="lineChart"></div>

<div id="pieChart"></div>

<div id="donutChart"></div>

    <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>

    <script src="script.js"></script>

</body>

</html>

Js

document.addEventListener("DOMContentLoaded", function () {

    drawBarChart();

    drawLineChart();

    drawPieChart();

    drawDonutChart();

});

function drawBarChart() {

    let data = [{

        x: ["A", "B", "C", "D"],

        y: [10, 20, 15, 25],

        type: "bar",

        marker: { color: "blue" }

    }];

    let layout = { title: "Bar Chart", xaxis: { title: "Categories" }, yaxis: { title: "Values" } };

    Plotly.newPlot("barChart", data, layout);

}

function drawLineChart() {

    let data = [{

        x: [1, 2, 3, 4, 5],

        y: [10, 15, 7, 12, 20],

        type: "scatter",

        mode: "lines+markers",

        marker: { color: "green" }

    }];

    let layout = { title: "Line Chart", xaxis: { title: "X Axis" }, yaxis: { title: "Y Axis" } };

    Plotly.newPlot("lineChart", data, layout);

}

function drawPieChart() {

    let data = [{

        values: [30, 20, 50],

        labels: ["Category A", "Category B", "Category C"],

        type: "pie"

    }];

    let layout = { title: "Pie Chart" };

    Plotly.newPlot("pieChart", data, layout);

}

function drawDonutChart() {

    let data = [{

        values: [25, 35, 40],

        labels: ["X", "Y", "Z"],

        type: "pie",

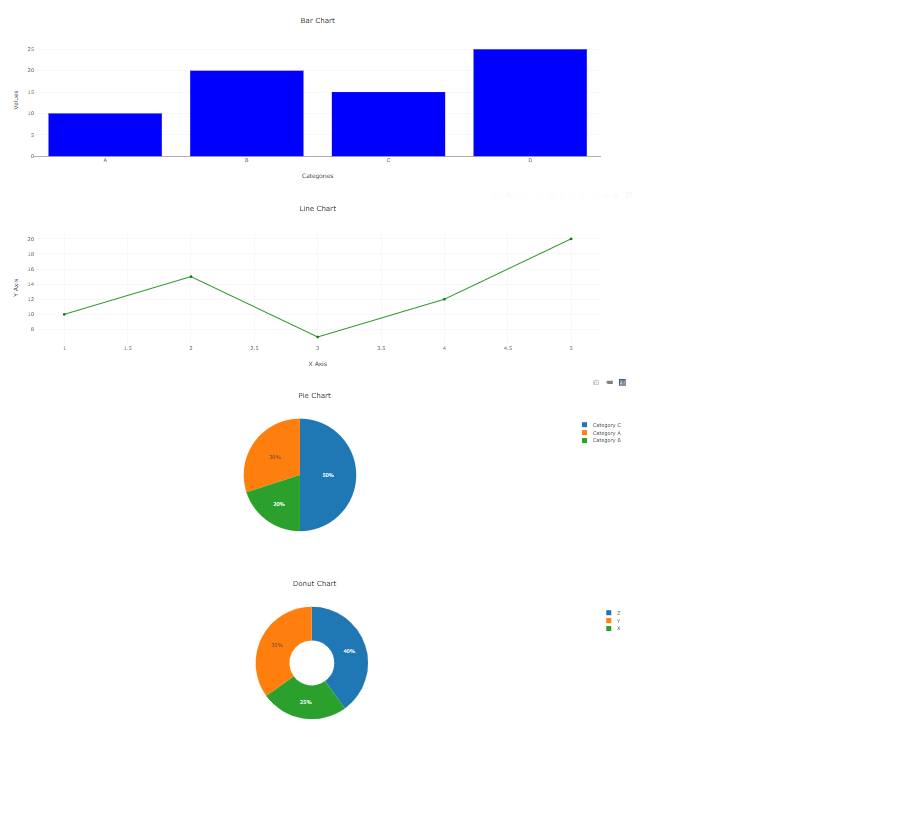
        hole: 0.4

    }];

    let layout = { title: "Donut Chart" };

    Plotly.newPlot("donutChart", data, layout);

}



Q5)

Code:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Z-Index Manipulation</title>

    <style>

        .container {

            position: relative;

            width: 300px;

            height: 250px;

            border: 2px solid black;

            margin-bottom: 20px;

        }

        .box {

            position: absolute;

            width: 100px;

            height: 100px;

            text-align: center;

            line-height: 100px;

            font-weight: bold;

            color: white;

        }

        #box1 { background: red; top: 50px; left: 50px; z-index: 1; }

        #box2 { background: blue; top: 80px; left: 80px; z-index: 2; }

        #box3 { background: green; top: 110px; left: 110px; z-index: 3; }

        .controls {

            display: flex;

            gap: 10px;

            align-items: center;

        }

    </style>

</head>

<body>

    <h2>Change Z-Index of Elements</h2>

    <div class="container">

        <div id="box1" class="box">Box 1</div>

        <div id="box2" class="box">Box 2</div>

        <div id="box3" class="box">Box 3</div>

    </div>

    <div class="controls">

        <label for="boxSelect">Select Box:</label>

        <select id="boxSelect">

            <option value="box1">Box 1</option>

            <option value="box2">Box 2</option>

            <option value="box3">Box 3</option>

        </select>

        <button onclick="changeZIndex(1)">Increase Z-Index</button>

        <button onclick="changeZIndex(-1)">Decrease Z-Index</button>

    </div>

    <h3>Current Z-Indexes:</h3>

    <p>Box 1: <span id="zBox1">1</span></p>

    <p>Box 2: <span id="zBox2">2</span></p>

    <p>Box 3: <span id="zBox3">3</span></p>

    <script>

        function updateZIndexes() {

            document.getElementById('zBox1').innerText = document.getElementById('box1').style.zIndex;

            document.getElementById('zBox2').innerText = document.getElementById('box2').style.zIndex;

            document.getElementById('zBox3').innerText = document.getElementById('box3').style.zIndex;

        }

        function changeZIndex(change) {

            let selectedBox = document.getElementById(document.getElementById("boxSelect").value);

            let currentZ = parseInt(window.getComputedStyle(selectedBox).zIndex);

            selectedBox.style.zIndex = currentZ + change;

            updateZIndexes();

        }

        updateZIndexes();

    </script>

</body>

</html>

Output:

