Divyansh Mishra

23BCE5071

**Web Programming**

**Assignment – 12**

Q1

**CODE:**

<!DOCTYPE html>

<html>

<head>

  <title>Pond Scene using JavaScript Canvas</title>

  <style>

    canvas {

      border: 1px solid black;

      display: block;

      margin: auto;

      margin-top: 20px;

    }

    h2 {

      text-align: center;

      font-family: 'Arial';

    }

  </style>

</head>

<body>

  <h2>Pond Scene using JavaScript Canvas</h2>

  <canvas id="pondCanvas" width="900" height="500"></canvas>

  <script>

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

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

    // House

    ctx.fillStyle = "orange";

    ctx.strokeStyle = "blue";

    ctx.fillRect(100, 200, 100, 100);

    ctx.strokeRect(100, 200, 100, 100);

    ctx.beginPath();

    ctx.moveTo(100, 200);

    ctx.lineTo(150, 150);

    ctx.lineTo(200, 200);

    ctx.closePath();

    ctx.fillStyle = "red";

    ctx.fill();

    ctx.stroke();

    ctx.fillStyle = "blue";

    ctx.fillRect(110, 210, 20, 20); // window

    ctx.fillStyle = "brown";

    ctx.fillRect(140, 250, 20, 50); // door

    // Pond (Oval)

    ctx.beginPath();

    ctx.ellipse(500, 270, 150, 80, 0, 0, 2 \* Math.PI);

    ctx.fillStyle = "#b3d9ff";

    ctx.fill();

    ctx.strokeStyle = "blue";

    ctx.stroke();

    // Boat (Quadrilateral - polygon)

    ctx.beginPath();

    ctx.moveTo(470, 260);

    ctx.lineTo(490, 280);

    ctx.lineTo(510, 280);

    ctx.lineTo(530, 260);

    ctx.closePath();

    ctx.fillStyle = "brown";

    ctx.fill();

    ctx.stroke();

    // Sun (circle with lines)

    ctx.beginPath();

    ctx.arc(750, 100, 40, 0, 2 \* Math.PI);

    ctx.fillStyle = "yellow";

    ctx.fill();

    ctx.stroke();

    for (let i = 0; i < 12; i++) {

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

      const x1 = 750 + Math.cos(angle) \* 50;

      const y1 = 100 + Math.sin(angle) \* 50;

      const x2 = 750 + Math.cos(angle) \* 70;

      const y2 = 100 + Math.sin(angle) \* 70;

      ctx.beginPath();

      ctx.moveTo(x1, y1);

      ctx.lineTo(x2, y2);

      ctx.stroke();

    }

    // Duck (Two Circles)

    ctx.beginPath(); // body

    ctx.arc(700, 330, 25, 0, 2 \* Math.PI);

    ctx.fillStyle = "yellow";

    ctx.fill();

    ctx.stroke();

    ctx.beginPath(); // head

    ctx.arc(730, 310, 20, 0, 2 \* Math.PI);

    ctx.fillStyle = "yellow";

    ctx.fill();

    ctx.stroke();

    ctx.beginPath(); // eye

    ctx.arc(738, 305, 4, 0, 2 \* Math.PI);

    ctx.fillStyle = "black";

    ctx.fill();

    ctx.beginPath(); // beak (triangle)

    ctx.moveTo(750, 310);

    ctx.lineTo(765, 305);

    ctx.lineTo(765, 315);

    ctx.closePath();

    ctx.fillStyle = "orange";

    ctx.fill();

    // Flower

    ctx.beginPath(); // stem

    ctx.moveTo(800, 200);

    ctx.lineTo(800, 270);

    ctx.strokeStyle = "green";

    ctx.lineWidth = 5;

    ctx.stroke();

    ctx.lineWidth = 1;

    // leaves

    ctx.beginPath();

    ctx.arc(790, 240, 15, 0, Math.PI, true);

    ctx.fillStyle = "green";

    ctx.fill();

    ctx.beginPath();

    ctx.arc(810, 240, 15, 0, Math.PI, true);

    ctx.fill();

    // petals

    ctx.fillStyle = "pink";

    ctx.beginPath();

    ctx.arc(800, 180, 20, 0, 2 \* Math.PI);

    ctx.fill();

    ctx.beginPath();

    ctx.arc(780, 200, 20, 0, 2 \* Math.PI);

    ctx.fill();

    ctx.beginPath();

    ctx.arc(820, 200, 20, 0, 2 \* Math.PI);

    ctx.fill();

    ctx.beginPath();

    ctx.arc(800, 220, 20, 0, 2 \* Math.PI);

    ctx.fill();

    // center

    ctx.beginPath();

    ctx.arc(800, 200, 10, 0, 2 \* Math.PI);

    ctx.fillStyle = "yellow";

    ctx.fill();

    // Stones (left)

    let stones = [

      { x: 130, y: 320, r: 15 },

      { x: 110, y: 330, r: 10 },

      { x: 150, y: 330, r: 20 }

    ];

    ctx.fillStyle = "gray";

    for (let s of stones) {

      ctx.beginPath();

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

      ctx.fill();

      ctx.stroke();

    }

    // Stones (right)

    let stones2 = [

      { x: 700, y: 380, r: 15 },

      { x: 720, y: 390, r: 10 },

      { x: 740, y: 380, r: 15 }

    ];

    for (let s of stones2) {

      ctx.beginPath();

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

      ctx.fill();

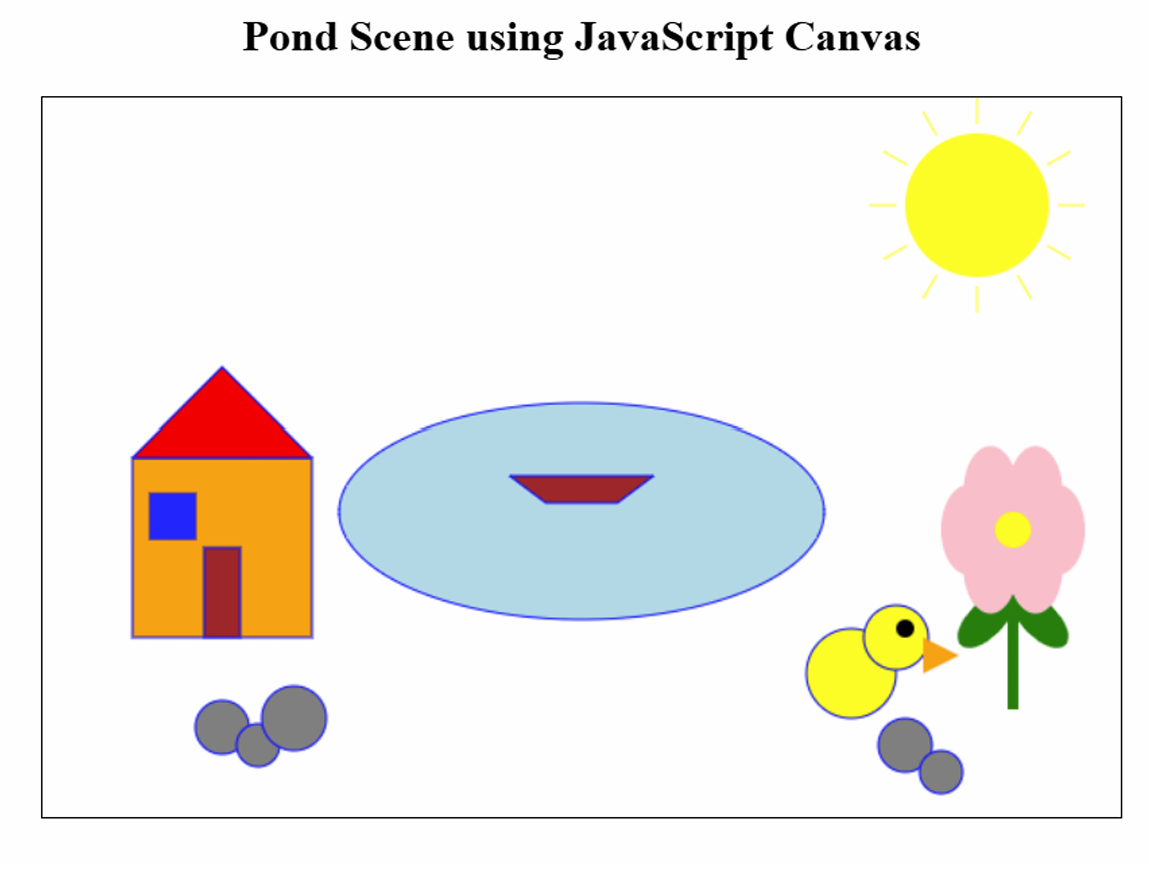
      ctx.stroke();

    }

  </script>

</body>

</html>



Q2.

Code:

<!DOCTYPE html>

<html>

<head>

  <title>Pond Scene using JavaScript Canvas</title>

  <style>

    canvas {

      border: 1px solid black;

      display: block;

      margin: auto;

      margin-top: 20px;

    }

    h2 {

      text-align: center;

      font-family: 'Arial';

    }

  </style>

</head>

<body>

  <h2>Pond Scene using JavaScript Canvas</h2>

  <canvas id="pondCanvas" width="900" height="500"></canvas>

  <script>

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

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

    let boatOffset = 0;

    let direction = 1;

    function drawScene() {

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

      // House

      ctx.fillStyle = "orange";

      ctx.strokeStyle = "blue";

      ctx.fillRect(100, 200, 100, 100);

      ctx.strokeRect(100, 200, 100, 100);

      ctx.beginPath();

      ctx.moveTo(100, 200);

      ctx.lineTo(150, 150);

      ctx.lineTo(200, 200);

      ctx.closePath();

      ctx.fillStyle = "red";

      ctx.fill();

      ctx.stroke();

      ctx.fillStyle = "blue";

      ctx.fillRect(110, 210, 20, 20); // window

      ctx.fillStyle = "brown";

      ctx.fillRect(140, 250, 20, 50); // door

      // Pond (Oval)

      ctx.beginPath();

      ctx.ellipse(500, 270, 150, 80, 0, 0, 2 \* Math.PI);

      ctx.fillStyle = "#b3d9ff";

      ctx.fill();

      ctx.strokeStyle = "blue";

      ctx.stroke();

      // Boat (Animated Quadrilateral)

      const baseX = 500 + boatOffset;

      ctx.beginPath();

      ctx.moveTo(baseX - 30, 260);

      ctx.lineTo(baseX - 10, 280);

      ctx.lineTo(baseX + 10, 280);

      ctx.lineTo(baseX + 30, 260);

      ctx.closePath();

      ctx.fillStyle = "brown";

      ctx.fill();

      ctx.stroke();

      // Sun

      ctx.beginPath();

      ctx.arc(750, 100, 40, 0, 2 \* Math.PI);

      ctx.fillStyle = "yellow";

      ctx.fill();

      ctx.stroke();

      for (let i = 0; i < 12; i++) {

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

        const x1 = 750 + Math.cos(angle) \* 50;

        const y1 = 100 + Math.sin(angle) \* 50;

        const x2 = 750 + Math.cos(angle) \* 70;

        const y2 = 100 + Math.sin(angle) \* 70;

        ctx.beginPath();

        ctx.moveTo(x1, y1);

        ctx.lineTo(x2, y2);

        ctx.stroke();

      }

      // Duck (Two Circles)

      ctx.beginPath(); // body

      ctx.arc(700, 330, 25, 0, 2 \* Math.PI);

      ctx.fillStyle = "yellow";

      ctx.fill();

      ctx.stroke();

      ctx.beginPath(); // head

      ctx.arc(730, 310, 20, 0, 2 \* Math.PI);

      ctx.fillStyle = "yellow";

      ctx.fill();

      ctx.stroke();

      ctx.beginPath(); // eye

      ctx.arc(738, 305, 4, 0, 2 \* Math.PI);

      ctx.fillStyle = "black";

      ctx.fill();

      ctx.beginPath(); // beak (triangle)

      ctx.moveTo(750, 310);

      ctx.lineTo(765, 305);

      ctx.lineTo(765, 315);

      ctx.closePath();

      ctx.fillStyle = "orange";

      ctx.fill();

      // Flower

      ctx.beginPath(); // stem

      ctx.moveTo(800, 200);

      ctx.lineTo(800, 270);

      ctx.strokeStyle = "green";

      ctx.lineWidth = 5;

      ctx.stroke();

      ctx.lineWidth = 1;

      // leaves

      ctx.beginPath();

      ctx.arc(790, 240, 15, 0, Math.PI, true);

      ctx.fillStyle = "green";

      ctx.fill();

      ctx.beginPath();

      ctx.arc(810, 240, 15, 0, Math.PI, true);

      ctx.fill();

      // petals

      ctx.fillStyle = "pink";

      ctx.beginPath();

      ctx.arc(800, 180, 20, 0, 2 \* Math.PI);

      ctx.fill();

      ctx.beginPath();

      ctx.arc(780, 200, 20, 0, 2 \* Math.PI);

      ctx.fill();

      ctx.beginPath();

      ctx.arc(820, 200, 20, 0, 2 \* Math.PI);

      ctx.fill();

      ctx.beginPath();

      ctx.arc(800, 220, 20, 0, 2 \* Math.PI);

      ctx.fill();

      // center

      ctx.beginPath();

      ctx.arc(800, 200, 10, 0, 2 \* Math.PI);

      ctx.fillStyle = "yellow";

      ctx.fill();

      // Stones (left)

      let stones = [

        { x: 130, y: 320, r: 15 },

        { x: 110, y: 330, r: 10 },

        { x: 150, y: 330, r: 20 }

      ];

      ctx.fillStyle = "gray";

      for (let s of stones) {

        ctx.beginPath();

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

        ctx.fill();

        ctx.stroke();

      }

      // Stones (right)

      let stones2 = [

        { x: 700, y: 380, r: 15 },

        { x: 720, y: 390, r: 10 },

        { x: 740, y: 380, r: 15 }

      ];

      for (let s of stones2) {

        ctx.beginPath();

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

        ctx.fill();

        ctx.stroke();

      }

    }

    function animate() {

      // Animate boat offset

      boatOffset += direction \* 0.5;

      if (boatOffset > 10 || boatOffset < -10) direction \*= -1;

      drawScene();

      requestAnimationFrame(animate);

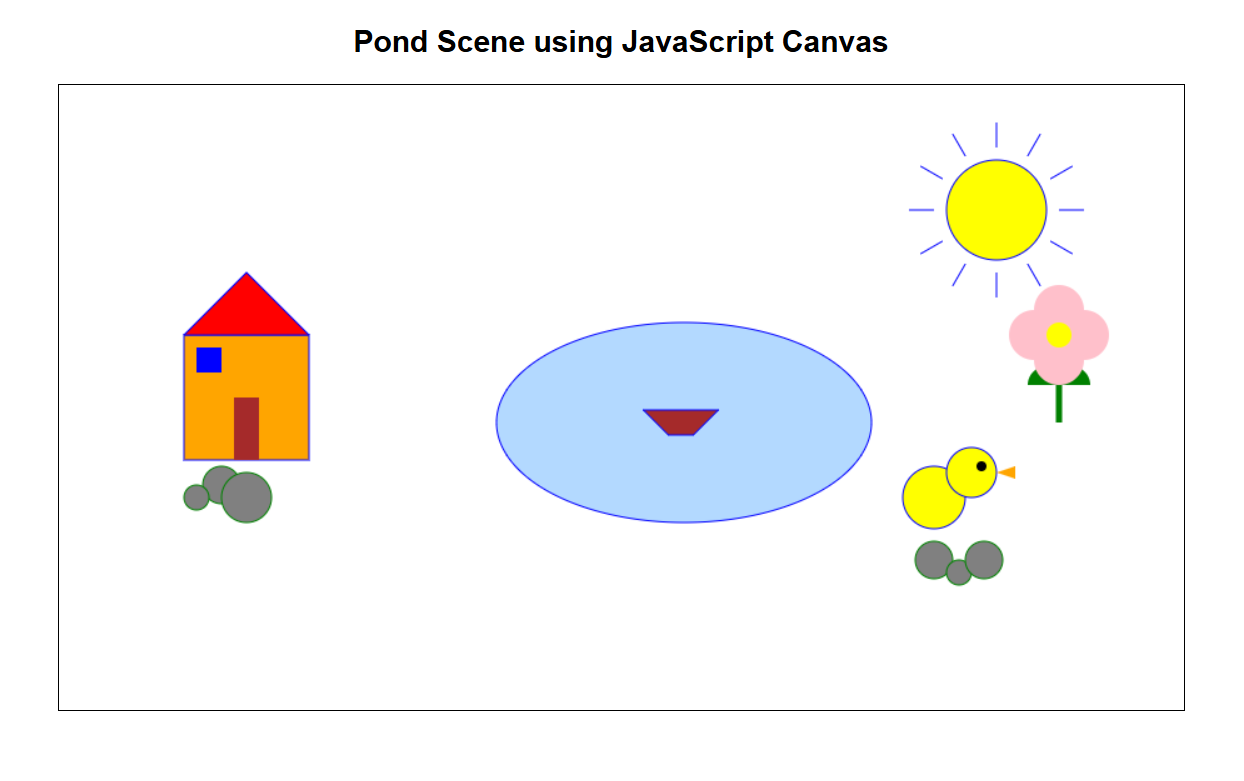
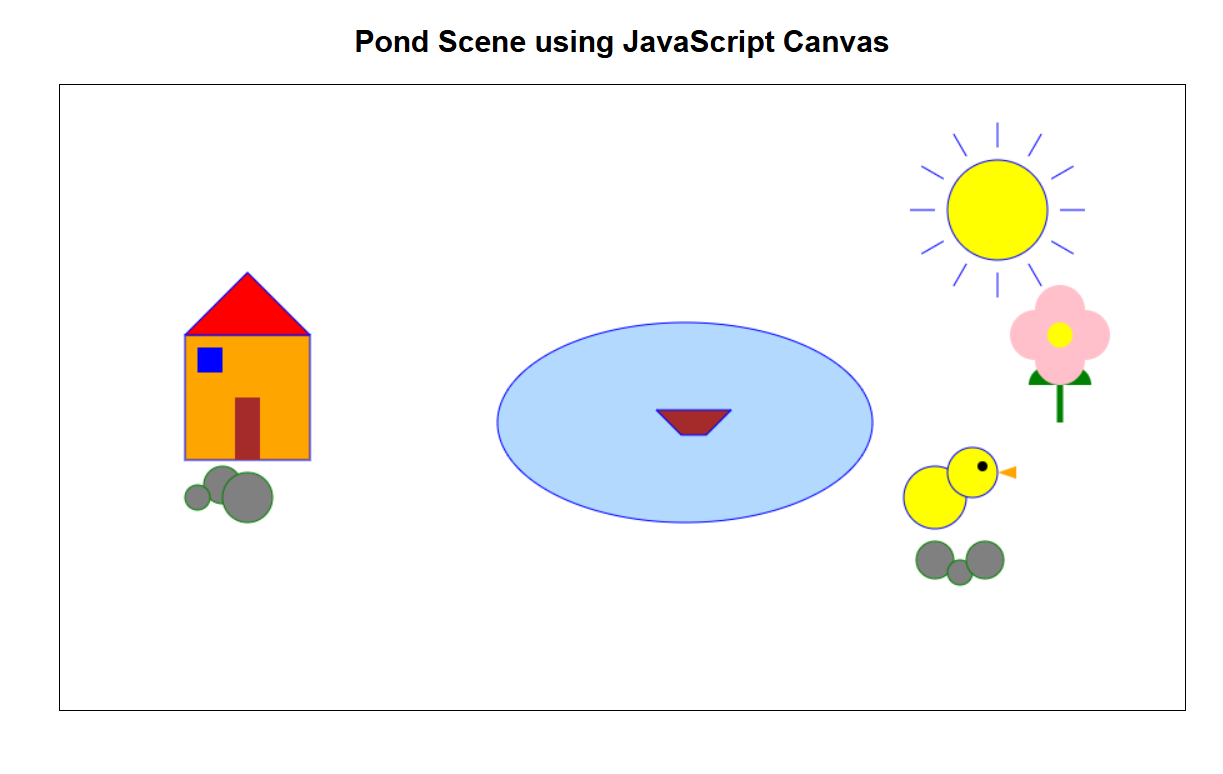
    }

    animate();

  </script>

</body>

</html>

Q3.

Code:

<!DOCTYPE html>

<html>

<head>

  <title>Analog Clock using Canvas</title>

  <style>

    body {

      display: flex;

      justify-content: center;

      align-items: center;

      height: 100vh;

      background: #f0f0f0;

    }

    canvas {

      background: #fff;

      border: 5px solid #333;

      border-radius: 50%;

    }

  </style>

</head>

<body>

  <canvas id="clockCanvas" width="400" height="400"></canvas>

  <script>

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

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

    const radius = canvas.width / 2;

    ctx.translate(radius, radius); // Move 0,0 to center

    const clockRadius = radius \* 0.90;

    function drawClock() {

      drawFace(ctx, clockRadius);

      drawNumbers(ctx, clockRadius);

      drawTime(ctx, clockRadius);

    }

    function drawFace(ctx, radius) {

      // Outer circle

      ctx.beginPath();

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

      ctx.fillStyle = '#fffacd'; // light yellow

      ctx.fill();

      // Border

      ctx.strokeStyle = '#333';

      ctx.lineWidth = 6;

      ctx.stroke();

      // Center dot

      ctx.beginPath();

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

      ctx.fillStyle = '#333';

      ctx.fill();

    }

    function drawNumbers(ctx, radius) {

      ctx.font = radius \* 0.15 + "px Arial";

      ctx.textBaseline = "middle";

      ctx.textAlign = "center";

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

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

        let x = radius \* 0.85 \* Math.sin(angle);

        let y = -radius \* 0.85 \* Math.cos(angle);

        ctx.fillText(num.toString(), x, y);

      }

    }

    function drawTime(ctx, radius) {

      const now = new Date();

      let hour = now.getHours();

      let minute = now.getMinutes();

      let second = now.getSeconds();

      // Hour

      hour %= 12;

      hour = (hour \* Math.PI / 6) +

             (minute \* Math.PI / (6 \* 60)) +

             (second \* Math.PI / (360 \* 60));

      drawHand(ctx, hour, radius \* 0.5, 7);

      // Minute

      minute = (minute \* Math.PI / 30) +

               (second \* Math.PI / (30 \* 60));

      drawHand(ctx, minute, radius \* 0.75, 5);

      // Second

      second = (second \* Math.PI / 30);

      drawHand(ctx, second, radius \* 0.85, 2, "red");

    }

    function drawHand(ctx, pos, length, width, color = "#333") {

      ctx.beginPath();

      ctx.lineWidth = width;

      ctx.lineCap = "round";

      ctx.strokeStyle = color;

      ctx.moveTo(0, 0);

      ctx.rotate(pos);

      ctx.lineTo(0, -length);

      ctx.stroke();

      ctx.rotate(-pos); // Reset rotation

    }

    // Update clock every second

    function updateClock() {

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

      drawClock();

    }

    setInterval(updateClock, 1000);

    updateClock(); // Initial draw

  </script>

</body>

</html>



Q4.

Code:

<!DOCTYPE html>

<html>

<head>

  <title>Dynamic Charts with Plotly.js</title>

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

  <style>

    body {

      font-family: Arial, sans-serif;

      margin: 20px;

      background: #f9f9f9;

    }

    h2 {

      text-align: center;

    }

    .chart-container {

      margin: 40px auto;

      width: 80%;

    }

  </style>

</head>

<body>

  <h2>Charts using Plotly.js</h2>

  <div class="chart-container">

    <div id="barChart"></div>

  </div>

  <div class="chart-container">

    <div id="lineChart"></div>

  </div>

  <div class="chart-container">

    <div id="pieChart"></div>

  </div>

  <div class="chart-container">

    <div id="donutChart"></div>

  </div>

  <script>

    // Sample Data

    const categories = ['Math', 'Science', 'English', 'History', 'Art'];

    const scores = [85, 90, 78, 88, 95];

    const colors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd'];

    // Bar Chart

    Plotly.newPlot('barChart', [{

      x: categories,

      y: scores,

      type: 'bar',

      marker: { color: colors }

    }], {

      title: 'Student Scores in Subjects (Bar Chart)',

      xaxis: { title: 'Subjects' },

      yaxis: { title: 'Scores' },

    });

    // Line Chart

    Plotly.newPlot('lineChart', [{

      x: categories,

      y: scores,

      type: 'scatter',

      mode: 'lines+markers',

      line: { color: '#17becf', width: 3 },

      marker: { color: colors, size: 10 }

    }], {

      title: 'Student Scores Trend (Line Chart)',

      xaxis: { title: 'Subjects' },

      yaxis: { title: 'Scores' },

    });

    // Pie Chart

    Plotly.newPlot('pieChart', [{

      labels: categories,

      values: scores,

      type: 'pie',

      marker: { colors: colors },

      textinfo: 'label+percent',

      insidetextorientation: 'radial'

    }], {

      title: 'Score Distribution (Pie Chart)',

      showlegend: true

    });

    // Donut Chart

    Plotly.newPlot('donutChart', [{

      labels: categories,

      values: scores,

      type: 'pie',

      hole: 0.4,

      marker: { colors: colors },

      textinfo: 'label+value'

    }], {

      title: 'Score Distribution (Donut Chart)',

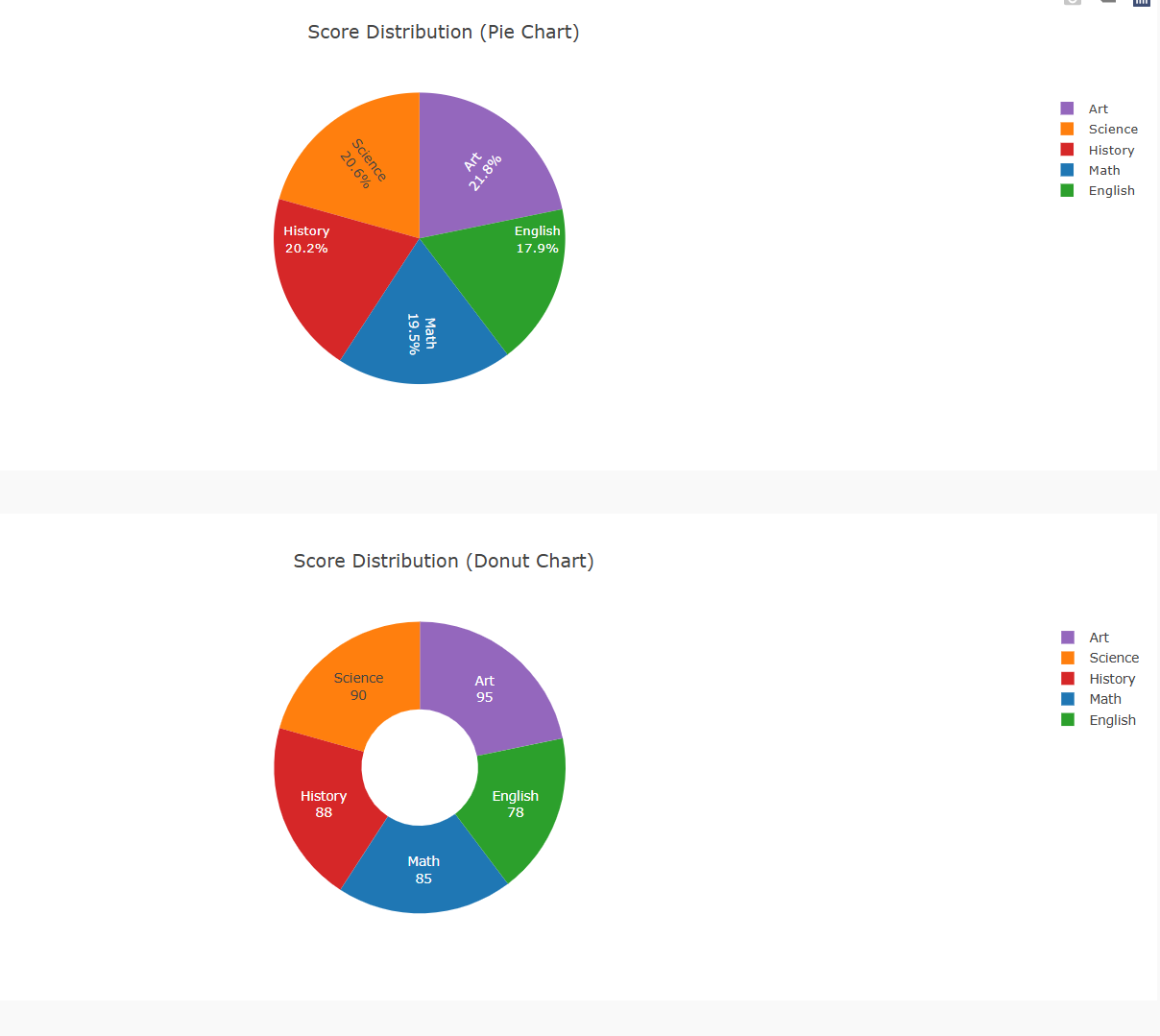
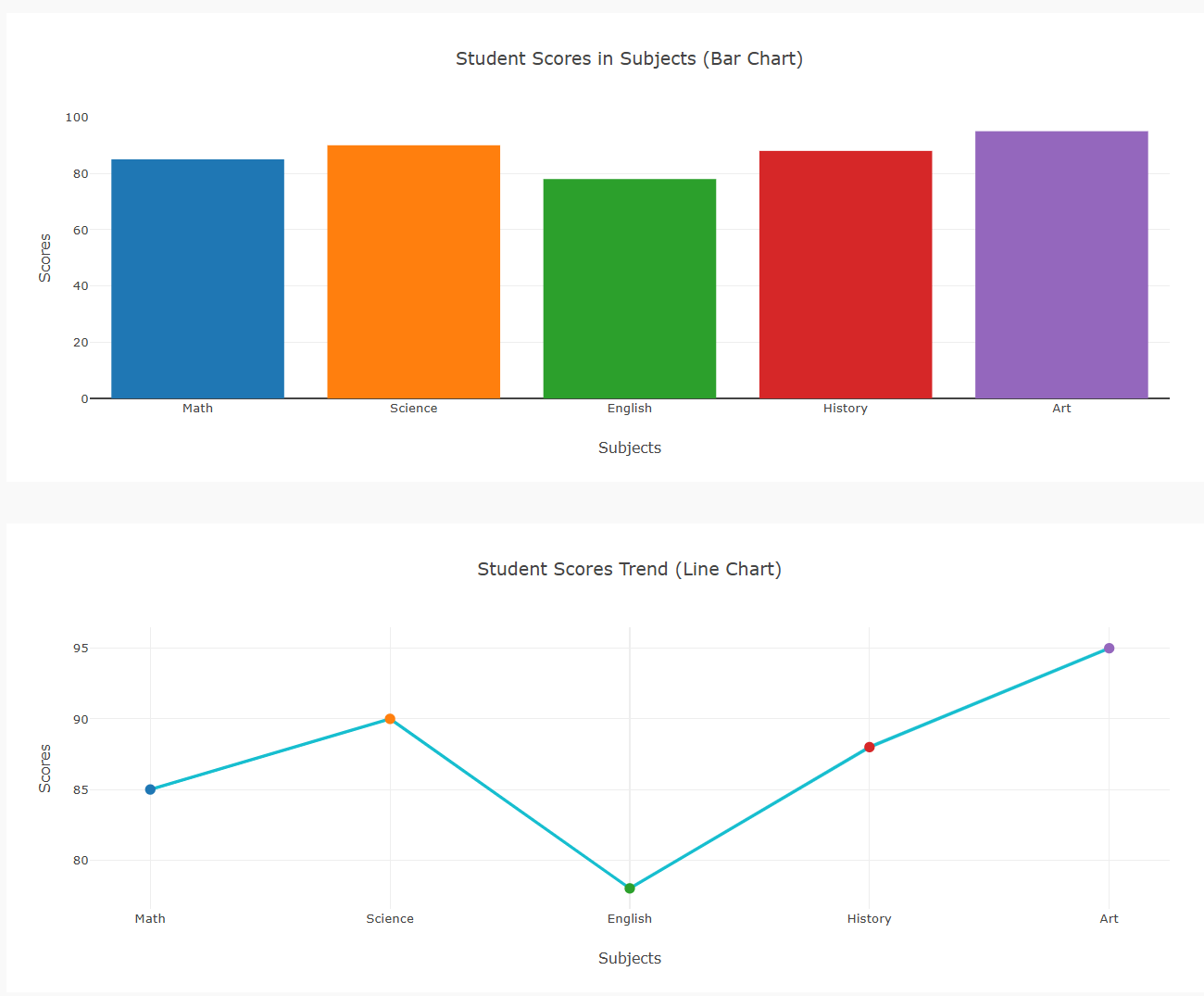
      showlegend: true

    });

  </script>

</body>

</html>



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 Demo</title>

    <style>

        .container {

            position: relative;

            width: 500px;

            height: 400px;

            margin: 50px auto;

            border: 1px solid #ccc;

        }

        .box {

            position: absolute;

            width: 150px;

            height: 150px;

            display: flex;

            justify-content: center;

            align-items: center;

            color: white;

            font-weight: bold;

            border: 3px solid black;

        }

        #box1 {

            background-color: rgba(255, 0, 0, 0.7);

            top: 50px;

            left: 50px;

            z-index: 1;

        }

        #box2 {

            background-color: rgba(0, 255, 0, 0.7);

            top: 100px;

            left: 150px;

            z-index: 2;

        }

        #box3 {

            background-color: rgba(0, 0, 255, 0.7);

            top: 150px;

            left: 250px;

            z-index: 3;

        }

        .controls {

            margin: 20px auto;

            width: 500px;

            display: flex;

            flex-direction: column;

            gap: 10px;

        }

        .box-control {

            display: flex;

            align-items: center;

            gap: 10px;

        }

        button {

            padding: 5px 10px;

            cursor: pointer;

        }

    </style>

</head>

<body>

    <div class="container">

        <div id="box1" class="box">Box 1<br>z-index: <span class="z-value">1</span></div>

        <div id="box2" class="box">Box 2<br>z-index: <span class="z-value">2</span></div>

        <div id="box3" class="box">Box 3<br>z-index: <span class="z-value">3</span></div>

    </div>

    <div class="controls">

        <div class="box-control">

            <span style="width: 80px;">Box 1:</span>

            <button onclick="decreaseZIndex('box1')">-</button>

            <input type="number" id="box1-z-index" value="1" min="-10" max="10" onchange="updateZIndex('box1', this.value)">

            <button onclick="increaseZIndex('box1')">+</button>

            <button onclick="bringToFront('box1')">Bring to Front</button>

            <button onclick="sendToBack('box1')">Send to Back</button>

        </div>

        <div class="box-control">

            <span style="width: 80px;">Box 2:</span>

            <button onclick="decreaseZIndex('box2')">-</button>

            <input type="number" id="box2-z-index" value="2" min="-10" max="10" onchange="updateZIndex('box2', this.value)">

            <button onclick="increaseZIndex('box2')">+</button>

            <button onclick="bringToFront('box2')">Bring to Front</button>

            <button onclick="sendToBack('box2')">Send to Back</button>

        </div>

        <div class="box-control">

            <span style="width: 80px;">Box 3:</span>

            <button onclick="decreaseZIndex('box3')">-</button>

            <input type="number" id="box3-z-index" value="3" min="-10" max="10" onchange="updateZIndex('box3', this.value)">

            <button onclick="increaseZIndex('box3')">+</button>

            <button onclick="bringToFront('box3')">Bring to Front</button>

            <button onclick="sendToBack('box3')">Send to Back</button>

        </div>

    </div>

    <script src="script.js"></script>

</body>

</html>

