Programación Prisma

Prisma.html

```
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Prisma Hexagonal análisis y solución</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <div class="container">
    <h1>Análisis del Prisma Hexagonal</h1>
    <form id="prismForm">
      <label for="height">Altura del prisma (cm):</label>
      <input type="number" id="height" name="height" min="1" required>
      <label for="apothem">Apotema del hexágono (cm):</label>
      <input type="number" id="apothem" name="apothem" min="1" required>
      <label for="side">Lado del hexágono (cm):</label>
      <input type="number" id="side" name="side" min="1" required>
      <button type="submit">Calcular</button>
    </form>
    <div id="results" style="display: none;">
      <canvas id="canvas" width="600" height="600"></canvas>
      <table id="resultsTable">
        <thead>
          <tr>
            <th>Descripción</th>
            <th>Valor</th>
          </tr>
```

```
            </thead>
            <tbody>
              <tr>
                <td>Área del hexágono</td>
                <td id="hexArea">N/A</td>
              </tr>
              <tr>
                <td>Área del rectángulo</td>
                <td id="rectArea">N/A</td>
              </tr>
              <tr>
                <td>Volumen del prisma</td>
                <td id="volume">N/A</td>
              </tr>
            </tbody>
          </table>
        </div>
    </div>
    <script src="script.js"></script>
</body>
</html>
```

Styles.css

```
body {
    font-family: Arial, sans-serif;
    background-color: #f0f0f0;
    display: flex;
    justify-content: center;
    align-items: center;
```

```
    height: 100vh;
    margin: 0;
}

.container {
    background-color: white;
    padding: 20px;
    border-radius: 10px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    text-align: center;
    width: 400px;
}

h1 {
    margin-bottom: 20px;
}

form {
    display: flex;
    flex-direction: column;
    margin-bottom: 20px;
}

label {
    margin-bottom: 5px;
}

input {
    margin-bottom: 15px;
```

```
    padding: 5px;
    border: 1px solid #ccc;
    border-radius: 5px;
}

button {
    padding: 10px;
    background-color: #007bff;
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}

button:hover {
    background-color: #0056b3;
}

canvas {
    border: 1px solid #ccc;
    background-color: #fff;
    margin-top: 20px;
}

#resultsTable {
    margin-top: 20px;
    width: 100%;
    border-collapse: collapse;
    background-color: green;
```

```
    color: black;
    font-family: Arial, sans-serif;
    font-size: 14px;
}

#resultsTable th, #resultsTable td {
    border: 1px solid #ccc;
    padding: 10px;
    text-align: center;
}
```

Script.js

```
document.getElementById('prismForm').addEventListener('submit', function(event) {
    event.preventDefault();
    const height = parseFloat(document.getElementById('height').value);
    const apothem = parseFloat(document.getElementById('apothem').value);
    const side = parseFloat(document.getElementById('side').value);

    const hexArea = calculateHexArea(side, apothem);
    const rectArea = calculateRectArea(height, side);
    const volume = calculateVolume(hexArea, height);

    document.getElementById('hexArea').innerText = `${hexArea.toFixed(2)} cm²`;
    document.getElementById('rectArea').innerText = `${rectArea.toFixed(2)} cm²`;
    document.getElementById('volume').innerText = `${volume.toFixed(2)} cm³`;

    drawPrism(height, side, apothem);
    document.getElementById('results').style.display = 'block';
});
```

```
function calculateHexArea(side, apothem) {
   return (3 * side * apothem);
}

function calculateRectArea(height, side) {
   return (height * side);
}

function calculateVolume(hexArea, height) {
   return (hexArea * height);
}

function drawPrism(height, side, apothem) {
   const canvas = document.getElementById('canvas');
   const ctx = canvas.getContext('2d');
   ctx.clearRect(0, 0, canvas.width, canvas.height);

   const centerX = canvas.width / 2;
   const centerY = canvas.height / 2 + height / 4;

   const hexPoints = [];
   for (let i = 0; i < 6; i++) {
      const angle = (Math.PI / 3) * i;
      const x = centerX + side * Math.cos(angle);
      const y = centerY + side * Math.sin(angle);
      hexPoints.push({ x, y });
   }
```

```
const colors = ['red', 'green', 'blue', 'orange', 'purple', 'cyan'];

// Draw bottom hexagon
ctx.fillStyle = 'lightgray';
ctx.beginPath();
ctx.moveTo(hexPoints[0].x, hexPoints[0].y);
hexPoints.forEach(point => {
  ctx.lineTo(point.x, point.y);
});
ctx.closePath();
ctx.fill();
ctx.stroke();

// Draw top hexagon
const topHexPoints = hexPoints.map(point => ({
  x: point.x,
  y: point.y - height
}));
ctx.fillStyle = 'lightgray';
ctx.beginPath();
ctx.moveTo(topHexPoints[0].x, topHexPoints[0].y);
topHexPoints.forEach(point => {
  ctx.lineTo(point.x, point.y);
});
ctx.closePath();
ctx.fill();
ctx.stroke();

// Draw lines connecting hexagons
```

```
    for (let i = 0; i < 6; i++) {
        ctx.strokeStyle = colors[i];
        ctx.beginPath();
        ctx.moveTo(hexPoints[i].x, hexPoints[i].y);
        ctx.lineTo(topHexPoints[i].x, topHexPoints[i].y);
        ctx.stroke();
        ctx.closePath();
    }

    // Fill the faces of the prism
    for (let i = 0; i < 6; i++) {
        ctx.fillStyle = colors[i];
        ctx.beginPath();
        ctx.moveTo(hexPoints[i].x, hexPoints[i].y);
        ctx.lineTo(topHexPoints[i].x, topHexPoints[i].y);
        ctx.lineTo(topHexPoints[(i + 1) % 6].x, topHexPoints[(i + 1) % 6].y);
        ctx.lineTo(hexPoints[(i + 1) % 6].x, hexPoints[(i + 1) % 6].y);
        ctx.closePath();
        ctx.fill();
        ctx.stroke();
    }
}
```