

Universidad Tecnológica Metropolitana



Estructuras de Datos Aplicadas

ISC. Ruth Betsaida Martínez Domínguez, MGTI

Prácticas 11-13

Soberanis Acosta Jimena Monserrat

Desarrollo de Software Multiplataforma

Cuarto Cuatrimestre

4°B

Parcial I

Viernes, 29 de septiembre de 2023

PRACTICA 11

Comencé con la estructura del HTML:

Existe 3 espacios para cada de las matrices que se crearán, la matriz principal y las otras 2 que son el resultado de las operaciones verticales y horizontales.

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Matriz y Resultados</title>
7      <link rel="stylesheet" href="styles.css">
8  </head>
9  <body>
10     <div class = "contenedor">
11         <div class="columna-izq">
12             <h1>Matriz</h1>
13             <div class="matriz-container">
14                 <table id="matriz"></table>
15             </div>
16
17             <div class="resultados-columna">
18                 <table>
19                     <tbody id="tablaResultadosColumna"></tbody>
20                 </table>
21             </div>
22         </div>
23
24         <div class="resultados-fila">
25             <table>
26                 <tbody id="tablaResultadosFila"></tbody>
27             </table>
28         </div>
29     </div>
30     <script src="script.js"></script>
31 </body>
32 </html>
```

Después, para la parte lógica de la práctica, está el script:

Definí una clase llamada MatrizOperaciones, el cual contendrá los mpetodos para calcular las operaciones de la matriz principal, el script comienza con un constructor que toma dos argumentos 'filas' y 'columnas', dentro de este constructor se crea la propiedad 'matriz' como un arreglo/matriz vacía con esos dos argumentos, que después llama al método llenarMatriz para llenarla con valores aleatorios.

El método llenarMatriz llena la matriz con números aleatorios entre 1 y 100, utiliza dos bucles for anidados para recorrer cada una de las filas y columnas de la matriz y le asigna un valor aleatorio a cada celda.

```
JS script.js > MatrizOperaciones > calcularResultados
1  class MatrizOperaciones {
2      constructor(filas, columnas) {
3          this.matriz = Array.from({ length: filas }, () => Array(columnas));
4          this.llenarMatriz();
5      }
6
7      llenarMatriz() {
8          for (let fila = 0; fila < this.matriz.length; fila++) {
9              for (let columna = 0; columna < this.matriz[fila].length; columna++) {
10                 this.matriz[fila][columna] = Math.floor(Math.random() * 100) + 1;
11             }
12         }
13     }
14 }
```

Luego, sigue el método calcularResultados, el cual calcula la suma y el promedio por fila y columna de la matriz principal, crea primeramente 4 arreglos vacíos que serán llenados por los resultados de cada operación de la fila y columna, inicia con el primer bucle for que recorre cada fila de la matriz, dentro de este se crea la variable sumaFila para tener la suma de los elementos de la fila actual.

En el segundo bucle anidado se recorre cada columna de la fila, y suma el valor actual de la matriz y se guarda en sumaPorColumna.

```
15  calcularResultados() {
16      const sumaPorFila = [];
17      const promedioPorFila = [];
18      const sumaPorColumna = [];
19      const promedioPorColumna = [];
20
21      for (let fila = 0; fila < this.matriz.length; fila++) {
22          let sumaFila = 0;
23          for (let columna = 0; columna < this.matriz[fila].length; columna++) {
24              sumaFila += this.matriz[fila][columna];
25              sumaPorColumna[columna] = (sumaPorColumna[columna] || 0) + this.matriz[fila][columna];
26          }
27          sumaPorFila.push(sumaFila);
28          promedioPorFila.push(sumaFila / this.matriz[fila].length);
29      }
30  }
```

Luego, se encuentre otro bucle for para calcular el promedio por columna, recorre las columnas de la matriz, este se logra con la suma de sumaPorColumna entre el número de filas y el resultado se agrega al arreglo promedioPorColumna. De último se devuelve cada uno de los arreglos calculados.

```
31      for (let columna = 0; columna < this.matriz[0].length; columna++) {
32          promedioPorColumna.push(sumaPorColumna[columna] / this.matriz.length);
33      }
34
35      return {
36          sumaPorFila,
37          promedioPorFila,
38          sumaPorColumna,
39          promedioPorColumna,
40      };
41  }
42  }
```

Proseguí con el método crearTablaMatriz:

En esta parte es donde inicialmente se crea un nuevo elemento de tipo tabla, primeramente, con un bucle que recorre cada 'fila' y en esta crea las filas reales, después con un segundo bucle anidado recorre las tanto fila como columna para crear las celdas para cada elemento. Al final devuelve la tabla construida.

```
43  ✓  crearTablaMatriz() {
44      const tabla = document.createElement('table');
45
46  ✓      for (let fila = 0; fila < this.matriz.length; fila++) {
47          const filaTabla = document.createElement('tr');
48
49  ✓          for (let column = 0; column < this.matriz[fila].length; column++) {
50              const celda = document.createElement('td');
51              celda.textContent = this.matriz[fila][column];
52              filaTabla.appendChild(celda);
53          }
54
55          tabla.appendChild(filaTabla);
56      }
57
58      return tabla;
59  }
60  }
61
```

Para la siguiente parte, debía de guardar los resultados de los cálculos en sus respectivas tablas, por lo que creé primero el método crearTablaResultadosFila, se inicia calculando los resultados que serán los datos necesarios para llenar la tabla, se crean dos elementos nuevos, un table y la cabecera (tr) de la tabla, que tienen como etiqueta A y B.

Se inicia un bucle for que recorre los resultados de las filas tomando de referencia el arreglo sumaPorFila, por cada iteración se crea una nueva fila de datos, se van añadiendo la suma y el promedio correspondientes. Por último, se devuelve la tabla.

```
60  ✓  crearTablaResultadosFila() {
61      const resultados = this.calcularResultados();
62      const tabla = document.createElement('table');
63      const cabecera = document.createElement('tr');
64      cabecera.innerHTML = '<th>A</th><th>B</th>';
65      tabla.appendChild(cabecera);
66
67  ✓      for (let fila = 0; fila < resultados.sumaPorFila.length; fila++) {
68          const filaHTML = document.createElement('tr');
69          filaHTML.innerHTML = `<td>${resultados.sumaPorFila[fila]}</td><td>${resultados.promedioPorFila[fila].toFixed(2)}</td>`;
70          tabla.appendChild(filaHTML);
71      }
72
73      return tabla;
74  }
```

El otro método crearTablaResultadosColumna es similar al anterior, pero se enfoca en crear una tabla para los resultados por columna. Se obtienen los resultados, se crea una tabla y se crean dos filas nuevas que serán la suma y el promedio. De igual manera, se crean los encabezados C y D.

Después se inicia el bucle que recorre los resultados de la columna tomando de referencia a `sumaPorColumna`, con ello se crean las celdas para la suma y el promedio, y están se agregan a su fila correspondiente. Por último, se devuelve la tabla.

```
76     crearTablaResultadosColumna() {
77         const resultados = this.calcularResultados();
78         const tabla = document.createElement('table');
79         const filaSuma = document.createElement('tr');
80         const filaPromedio = document.createElement('tr');
81
82         const cabeceraSuma = document.createElement('th');
83         cabeceraSuma.textContent = 'C';
84         filaSuma.appendChild(cabeceraSuma);
85
86         const cabeceraPromedio = document.createElement('th');
87         cabeceraPromedio.textContent = 'D';
88         filaPromedio.appendChild(cabeceraPromedio);
89
90         for (let fila = 0; fila < resultados.sumaPorColumna.length; fila++) {
91             const celdaSuma = document.createElement('td');
92             celdaSuma.textContent = resultados.sumaPorColumna[fila];
93             filaSuma.appendChild(celdaSuma);
94
95             const celdaPromedio = document.createElement('td');
96             celdaPromedio.textContent = resultados.promedioPorColumna[fila].toFixed(2);
97             filaPromedio.appendChild(celdaPromedio);
98         }
99
100        tabla.appendChild(filaSuma);
101        tabla.appendChild(filaPromedio);
102
103        return tabla;
104    }
105 }
```

Fuera de la parte principal se creó una instancia con la clase `MatrizOperaciones`, que crea la matriz base y la llena de valores aleatorios.

Poco a poco se van llamando a ciertas partes del HTML para crear cada una de las tablas con la instancia creada anteriormente.

```
107 const matrizOperaciones = new MatrizOperaciones(5, 10);
108 matrizOperaciones.llenarMatriz();
109
110 const matrizContainer = document.querySelector('.matriz-container');
111 matrizContainer.appendChild(matrizOperaciones.crearTablaMatriz());
112
113 const tablaResultadosFila = document.getElementById('tablaResultadosFila');
114 tablaResultadosFila.appendChild(matrizOperaciones.crearTablaResultadosFila());
115
116 const tablaResultadosColumna = document.getElementById('tablaResultadosColumna');
117 tablaResultadosColumna.appendChild(matrizOperaciones.crearTablaResultadosColumna());
118 }
```

Por último, le añadí estilo con el CSS:

```

1  √ body {
2    |   font-family: Arial, sans-serif;
3    |   text-align: center;
4    |   margin: 10px;
5    | }
6
7  √ h1 {
8    |   margin-bottom: 20px;
9    | }
10
11 √ .contenedor{
12 |   display: flex;
13 | }
14
15 √ table {
16 |   border-collapse: collapse;
17 |   margin-right: 10px;
18 | }
19
20 √ .matriz-container h1{
21 |   align-items: center;
22 |   justify-content: center;
23 | }
24

```

```

25 .matriz-container table{
26 |   width: 93%;
27 | }
28
29 td {
30 |   border: 1px solid #ccc;
31 |   padding: 6px;
32 | }
33
34 .matriz-container{
35 |   display: flex;
36 |   flex-direction: column;
37 |   align-items: end;
38 | }
39
40 .resultados-fila{
41 |   margin-top: 60px ;
42 | }
43
44 .resultados-columna{
45 |   flex: 1;
46 |   padding: 10px;
47 | }
48

```

```

49 √ .resultados-columna table{
50 |   flex: 1;
51 |   padding: 10px;
52 |   width: 100%;
53 | }
54
55 √ ul {
56 |   list-style-type: none;
57 |   padding: 0;
58 | }
59
60 √ ul li {
61 |   margin-bottom: 8px;
62 | }
63

```

Y con ayuda del framework Electron creé la aplicación, el cual luce así:

| Matriz y Resultados | | | | | | | | | | |
|----------------------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-----------|
| File Edit View Window Help | | | | | | | | | | |
| Matriz | | | | | | | | | | |
| | | | | | | | | | | A B |
| 61 | 82 | 93 | 97 | 43 | 95 | 95 | 75 | 68 | 46 | 755 75.50 |
| 89 | 98 | 51 | 8 | 38 | 48 | 62 | 98 | 44 | 47 | 583 58.30 |
| 52 | 46 | 64 | 3 | 30 | 23 | 25 | 59 | 99 | 52 | 453 45.30 |
| 33 | 86 | 3 | 4 | 53 | 83 | 42 | 20 | 79 | 14 | 417 41.70 |
| 24 | 82 | 72 | 75 | 36 | 62 | 37 | 81 | 32 | 80 | 581 58.10 |
| | | | | | | | | | | |
| C | 259 | 394 | 283 | 187 | 200 | 311 | 261 | 333 | 322 | 239 |
| D | 51.80 | 78.80 | 56.60 | 37.40 | 40.00 | 62.20 | 52.20 | 66.60 | 64.40 | 47.80 |

| Matriz y Resultados | | | | | | | | | | |
|----------------------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-----------|
| File Edit View Window Help | | | | | | | | | | |
| Matriz | | | | | | | | | | |
| | | | | | | | | | | A B |
| 62 | 39 | 53 | 16 | 5 | 85 | 78 | 31 | 91 | 81 | 541 54.10 |
| 6 | 57 | 16 | 48 | 56 | 87 | 43 | 98 | 53 | 25 | 489 48.90 |
| 92 | 67 | 54 | 1 | 12 | 85 | 26 | 78 | 32 | 67 | 514 51.40 |
| 93 | 87 | 73 | 21 | 84 | 98 | 77 | 12 | 66 | 86 | 697 69.70 |
| 4 | 9 | 76 | 92 | 96 | 62 | 22 | 74 | 73 | 44 | 552 55.20 |
| | | | | | | | | | | |
| C | 257 | 259 | 272 | 178 | 253 | 417 | 246 | 293 | 315 | 303 |
| D | 51.40 | 51.80 | 54.40 | 35.60 | 50.60 | 83.40 | 49.20 | 58.60 | 63.00 | 60.60 |

PRÁCTICA 12

Comencé con la estructura de HTML:

Existe una tabla para las ventas, y cada uno de los títulos para los resultados correspondientes, por último, creará la lista de ventas por día.

```
> index.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Resumen de Ventas</title>
7      <link rel="stylesheet" href="styles.css">
8  </head>
9  <body>
10     <div class="container">
11         <h1>Resumen de Ventas</h1>
12         <table id="ventas-table">
13
14         </table>
15         <div class="resultados">
16             <p>Menor venta realizada: <span id="menor-venta"></span></p>
17             <p>Mayor venta realizada: <span id="mayor-venta"></span></p>
18             <p>Venta total: <span id="venta-total"></span></p>
19             <p>Venta por día:</p>
20             <ul id="venta-por-dia">
21
22             </ul>
23         </div>
24     </div>
25     <script src="script.js"></script>
26 </body>
27 </html>
28
```

Luego, seguí con el script:

Comencé con la creación de la Clase VentasAnalyzer: Esta clase se encargará de manejar todas las operaciones relacionadas con las ventas. Le proporcioné un parámetro llamado ventas. En el constructor de la clase, especifiqué que tomará la matriz de ventas como argumento y la almacenará en una propiedad llamada ventas.

Después está encontrarMenorVenta: Se encarga de buscar la venta más baja en la matriz de ventas. Empecé inicializando algunas variables, como menorVenta, mes, y día. Luego, utilicé bucles anidados para recorrer la matriz y comparar cada valor con la venta más baja encontrada hasta ese momento. Si se encontraba una venta más baja, se actualiza las variables correspondientes. Al final, devuelve un objeto con la información de la venta más baja.


```

JS script.js > VentasAnalyzer > encontrarMenorVenta
1  class VentasAnalyzer {
2      constructor(ventas) {
3          this.ventas = ventas;
4      }
5
6      encontrarMenorVenta() {
7          let menorVenta = this.ventas[0][0];
8          let mes = 0;
9          let dia = 0;
10
11         for (let fila = 0; fila < this.ventas.length; fila++) {
12             for (let columna = 0; columna < this.ventas[fila].length; columna++) {
13                 if (this.ventas[fila][columna] < menorVenta) {
14                     menorVenta = this.ventas[fila][columna];
15                     mes = fila + 1;
16                     dia = columna + 1;
17                 }
18             }
19         }
20
21         return { venta: menorVenta, mes, dia };
22     }

```

Proseguí con el método encontrarMayorVenta: Es similar al método anterior, busca la venta más alta en la matriz de ventas. Utilicé esa lógica similar para encontrar la venta más alta y que devuelva la información relacionada.

```

24     encontrarMayorVenta() {
25         let mayorVenta = this.ventas[0][0];
26         let mes = 0;
27         let dia = 0;
28
29         for (let fila = 0; fila < this.ventas.length; fila++) {
30             for (let columna = 0; columna < this.ventas[fila].length; columna++) {
31                 if (this.ventas[fila][columna] > mayorVenta) {
32                     mayorVenta = this.ventas[fila][columna];
33                     mes = fila + 1;
34                     dia = columna + 1;
35                 }
36             }
37         }
38
39         return { venta: mayorVenta, mes, dia };
40     }
41

```

Seguí con el método calcularVentaTotal: Calcula la suma total de todas las ventas en la matriz. Comencé inicializando una variable total en cero y luego utilicé los bucles anidados para sumar todos los valores en la matriz. Finalmente, devuelve el total.

```

41
42 ✓   calcularVentaTotal() {
43       let total = 0;
44
45 ✓       for (let fila = 0; fila < this.ventas.length; fila++) {
46 ✓           for (let columna = 0; columna < this.ventas[fila].length; columna++) {
47               total += this.ventas[fila][columna];
48           }
49       }
50
51       return total;
52   }
53

```

Para las ventas por día, hice el método `calcularVentaPorDia`: calcula la suma de ventas por día de la semana. Inicié un arreglo llamado `ventaPorDia` con valores iniciales en cero para cada día de la semana. Luego, utilicé los bucles anidados para recorrer la matriz y agrega cada venta al día correspondiente en el arreglo. Devuelve el arreglo con las sumas por día.

```

54 ✓   calcularVentaPorDia() {
55       const ventaPorDia = [0, 0, 0, 0, 0, 0, 0];
56
57 ✓       for (let fila = 0; fila < this.ventas.length; fila++) {
58 ✓           for (let columna = 0; columna < this.ventas[fila].length; columna++) {
59               ventaPorDia[columna] += this.ventas[fila][columna];
60           }
61       }
62
63       return ventaPorDia;
64   }
65

```

Esta el siguiente método llamado `imprimirResultados`: Utiliza todos los métodos anteriores para obtener información sobre las ventas, como la venta más baja, la venta más alta, la venta total y las ventas por día. Luego, actualiza los elementos HTML en la aplicación para mostrar estos resultados.

```

66 ✓   imprimirResultados() {
67       const menorVentaInfo = this.encontrarMenorVenta();
68       const mayorVentaInfo = this.encontrarMayorVenta();
69       const ventaTotal = this.calcularVentaTotal();
70       const ventaPorDia = this.calcularVentaPorDia();
71
72       document.getElementById('menor-venta').textContent = `$$${menorVentaInfo.venta} (Mes ${menorVentaInfo.mes}, Día ${menorVentaInfo.dia})`;
73       document.getElementById('mayor-venta').textContent = `$$${mayorVentaInfo.venta} (Mes ${mayorVentaInfo.mes}, Día ${mayorVentaInfo.dia})`;
74       document.getElementById('venta-total').textContent = `$$${ventaTotal}`;
75
76       const listaVentaPorDia = document.getElementById('venta-por-dia');
77       const diasSemana = ['Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes', 'Sábado', 'Domingo'];
78
79 ✓       for (let i = 0; i < ventaPorDia.length; i++) {
80           const listItem = document.createElement('li');
81           listItem.textContent = `$$${diasSemana[i]}: $$${ventaPorDia[i]}`;
82           listaVentaPorDia.appendChild(listItem);
83       }
84   }

```

Seguí con el método `llenarTablaVentas`: Este se encarga de llenar una tabla en la aplicación con los valores de la matriz de ventas. Crea elementos de tabla y celdas HTML y establece los valores de las celdas con las ventas correspondientes.

```

86     llenarTablaVentas() {
87         const tablaVentas = document.getElementById('ventas-table');
88
89         for (let fila = 0; fila < this.ventas.length; fila++) {
90             const filaTabla = document.createElement('tr');
91
92             for (let columna = 0; columna < this.ventas[fila].length; columna++) {
93                 const celda = document.createElement('td');
94                 celda.textContent = `$$${this.ventas[fila][columna]}`;
95                 filaTabla.appendChild(celda);
96             }
97
98             tablaVentas.appendChild(filaTabla);
99         }
100     }
101 }

```

Creé una Instancia de la Clase VentasAnalyzer: Este pasa la matriz de ventas como argumento. Esto permite utilizar todos los métodos y propiedades que definí dentro de la clase.

Finalmente, llamé a los métodos llenarTablaVentas e imprimirResultados en la instancia de VentasAnalyzer.

```

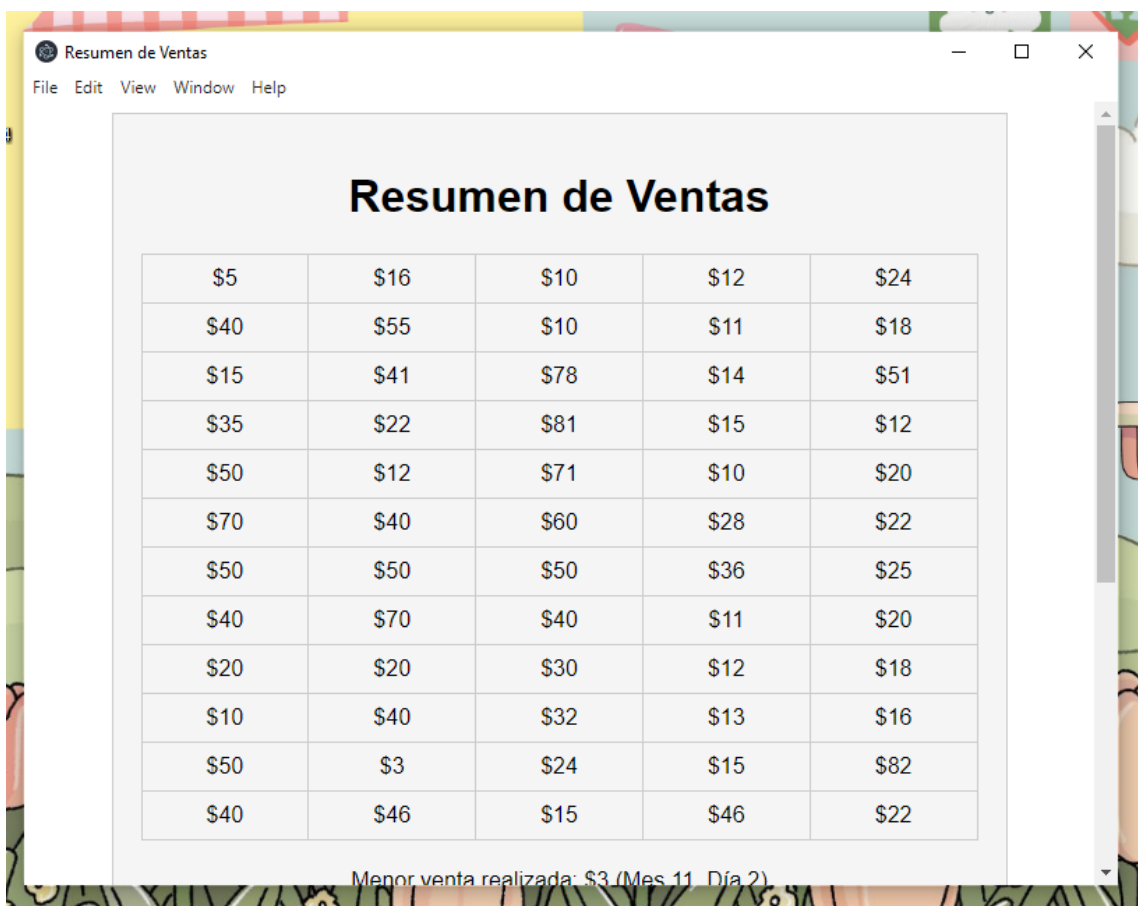
102
103 const ventasAnalyzer = new VentasAnalyzer([
104     [5, 16, 10, 12, 24],
105     [40, 55, 10, 11, 18],
106     [15, 41, 78, 14, 51],
107     [35, 22, 81, 15, 12],
108     [50, 12, 71, 10, 20],
109     [70, 40, 60, 28, 22],
110     [50, 50, 50, 36, 25],
111     [40, 70, 40, 11, 20],
112     [20, 20, 30, 12, 18],
113     [10, 40, 32, 13, 16],
114     [50, 3, 24, 15, 82],
115     [40, 46, 15, 46, 22]
116 ]);
117
118 ventasAnalyzer.llenarTablaVentas();
119 ventasAnalyzer.imprimirResultados();
120

```

Le definí un estilo con CSS:

```
# styles.css > ...
1  body {
2      font-family: Arial, sans-serif;
3      text-align: center;
4  }
5
6  .container {
7      max-width: 600px;
8      margin: 0 auto;
9      padding: 20px;
10     border: 1px solid #ccc;
11     background-color: #f5f5f5;
12 }
13
14 table {
15     width: 100%;
16     margin-top: 20px;
17     border-collapse: collapse;
18 }
19
20 table, th, td {
21     border: 1px solid #ccc;
22     padding: 8px;
23 }
24
25 .resultados {
26     margin-top: 20px;
27 }
28
29 #venta-por-dia {
30     list-style-type: none;
31     padding: 0;
32 }
33
34 #venta-por-dia li {
35     margin-bottom: 8px;
36 }
37
```

Y con Electron, lancé la aplicación que luce así:



| \$5 | \$16 | \$10 | \$12 | \$24 |
|------|------|------|------|------|
| \$40 | \$55 | \$10 | \$11 | \$18 |
| \$15 | \$41 | \$78 | \$14 | \$51 |
| \$35 | \$22 | \$81 | \$15 | \$12 |
| \$50 | \$12 | \$71 | \$10 | \$20 |
| \$70 | \$40 | \$60 | \$28 | \$22 |
| \$50 | \$50 | \$50 | \$36 | \$25 |
| \$40 | \$70 | \$40 | \$11 | \$20 |
| \$20 | \$20 | \$30 | \$12 | \$18 |
| \$10 | \$40 | \$32 | \$13 | \$16 |
| \$50 | \$3 | \$24 | \$15 | \$82 |
| \$40 | \$46 | \$15 | \$46 | \$22 |

Menor venta realizada: \$3 (Mes 11, Día 2)

| | | | | |
|------|------|------|------|------|
| \$40 | \$46 | \$15 | \$46 | \$22 |
|------|------|------|------|------|

Menor venta realizada: \$3 (Mes 11, Día 2)

Mayor venta realizada: \$82 (Mes 11, Día 5)

Venta total: \$1894

Venta por día:

Lunes: \$425

Martes: \$415

Miércoles: \$501

Jueves: \$223

Viernes: \$330

Sábado: \$0

Domingo: \$0

PRACTICA 13

Comencé con esta práctica con el HTML, que crea una tabla con la información de los alumnos y sus calificaciones de sus parciales:

```

index.html > html > body > table > tbody > tr > td
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6    <title>Calificaciones</title>
7    <link rel="stylesheet" type="text/css" href="styles.css">
8  </head>
9  <body>
10   <h1>Calificaciones de Alumnos</h1>
11   <table>
12     <thead>
13       <tr>
14         <th>Alumno</th>
15         <th>Parcial 1</th>
16         <th>Parcial 2</th>
17         <th>Parcial 3</th>
18         <th>Parcial 4</th>
19         <th>Promedio</th>
20       </tr>
21     </thead>
22     <tbody>
23       <tr>
24         <td>Alumno 1</td>
25         <td>8.5</td>
26         <td>7.0</td>
27         <td>6.5</td>
28         <td>9.0</td>
29         <td></td>
30       </tr>

```

Y debajo la información que se obtendrá referente a la tabla.

```
31      <tr>
32          <td>Alumno 2</td>
33          <td>6.0</td>
34          <td>5.5</td>
35          <td>7.5</td>
36          <td>6.0</td>
37          <td></td>
38      </tr>
39      <tr>
40          <td>Alumno 3</td>
41          <td>8.6</td>
42          <td>7.0</td>
43          <td>10.0</td>
44          <td>9.5</td>
45          <td></td>
46      </tr>
47  </tbody>
48 </table>
49
50 <h2>Información:</h2>
51 <p>Promedio de cada alumno:</p>
52 <ul id="promedios"></ul>
53 <p>Promedio más alto: <span id="promedio-alto"></span></p>
54 <p>Promedio más bajo: <span id="promedio-bajo"></span></p>
55 <p>Cantidad de parciales reprobados: <span id="parciales-reprobados"></span></p>
56 <p>Distribución de calificaciones finales:</p>
57 <ul id="distribucion"></ul>
58
59 <script src="script.js"></script>
60 </body>
61 </html>
```

Proseguí con el script:

Definí una clase llamada EstadisticasCalificaciones para encapsular todas las operaciones relacionadas con el cálculo de estadísticas de calificaciones. Esta clase tiene un constructor que inicializa las propiedades necesarias, como table, filas, promedioAlto, promedioBajo, parcialesReprobados y distribución (los rangos de calificación). La propiedad table almacena la tabla HTML que contiene las calificaciones, mientras que filas almacenan todas las filas de esa tabla.

```
1  class EstadisticasCalificaciones {
2      constructor() {
3          this.table = document.querySelector('table');
4          this.filas = this.table.querySelectorAll('tbody tr');
5          this.promedioAlto = -Infinity;
6          this.promedioBajo = Infinity;
7          this.parcialesReprobados = 0;
8          this.distribucion = {
9              '0-4.9': 0,
10             '5.0-5.9': 0,
11             '6.0-6.9': 0,
12             '7.0-7.9': 0,
13             '8.0-8.9': 0,
14             '9.0-10': 0
15         };
16     }
```

Dentro de la clase EstadisticasCalificaciones, creé dos métodos principales: calcularEstadisticas y mostrarResultados.

El primero, calcularEstadisticas, realiza todos los cálculos necesarios para obtener el promedio alto, el promedio bajo, el número de parciales reprobados y la distribución de calificaciones.

```
17
18  ✓   calcularEstadisticas() {
19  ✓       for (const fila of this.filas) {
20           const celdas = fila.querySelectorAll('td');
21           let suma = 0;
22
23       ✓       for (let i = 1; i < celdas.length - 1; i++) {
24           const calificacion = parseFloat(celdas[i].textContent);
25
26       ✓           if (!isNaN(calificacion)) {
27               suma += calificacion;
28
29       ✓           if (calificacion < 7.0) {
30               this.parcialesReprobados++;
31           }
32       }
33     }
34
```

Este método recorre cada fila de la tabla, luego, para cada fila, recorre las celdas correspondientes a las calificaciones. Se suman todas las calificaciones válidas para calcular el promedio y se identifican los parciales reprobados (calificaciones menores a 7.0). Luego, se actualizan las propiedades promedioAlto y promedioBajo, si se encuentra un promedio más alto o más bajo que los actuales, respectivamente. También se actualiza la distribución de calificaciones en el objeto distribucion según el rango de promedio obtenido.

```
35           const promedio = suma / (celdas.length - 2);
36           celdas[celdas.length - 1].textContent = promedio.toFixed(2);
37
38           this.promedioAlto = Math.max(this.promedioAlto, promedio);
39           this.promedioBajo = Math.min(this.promedioBajo, promedio);
40
41           if (promedio >= 0 && promedio <= 4.9) {
42               this.distribucion['0-4.9']++;
43           } else if (promedio >= 5.0 && promedio <= 5.9) {
44               this.distribucion['5.0-5.9']++;
45           } else if (promedio >= 6.0 && promedio <= 6.9) {
46               this.distribucion['6.0-6.9']++;
47           } else if (promedio >= 7.0 && promedio <= 7.9) {
48               this.distribucion['7.0-7.9']++;
49           } else if (promedio >= 8.0 && promedio <= 8.9) {
50               this.distribucion['8.0-8.9']++;
51           } else if (promedio >= 9.0 && promedio <= 10) {
52               this.distribucion['9.0-10']++;
53           }
54       }
55     }
56
```

El segundo método, `mostrarResultados`, se encarga de actualizar los elementos HTML con los resultados calculados. Los valores de `promedioAlto`, `promedioBajo` y `parcialesReprobados` se establecen en los elementos correspondientes del documento HTML, mientras que la distribución de calificaciones se muestra en una lista en el documento.

```
57 mostrarResultados() {
58   document.getElementById('promedio-alto').textContent = this.promedioAlto.toFixed(2);
59   document.getElementById('promedio-bajo').textContent = this.promedioBajo.toFixed(2);
60   document.getElementById('parciales-reprobados').textContent = this.parcialesReprobados;
61
62   const distribucionList = document.getElementById('distribucion');
63   for (const rango in this.distribucion) {
64     const li = document.createElement('li');
65     li.textContent = `${rango}: ${this.distribucion[rango]} Alumnos`;
66     distribucionList.appendChild(li);
67   }
68 }
69 }
```

Finalmente, fuera de la clase, creé una instancia llamada `estadisticas` de `EstadisticasCalificaciones` que llama a sus métodos `calcularEstadisticas` y `mostrarResultados` para realizar los cálculos y mostrar los resultados.

```
71 const estadisticas = new EstadisticasCalificaciones();
72 estadisticas.calcularEstadisticas();
73 estadisticas.mostrarResultados();
74
```

Le agregué el estilo que quería con CSS:

```
# styles.css > $ #distribucion
1 table {
2   border-collapse: collapse;
3   width: 80%;
4   margin: 20px auto;
5   font-family: Arial, sans-serif;
6 }
7
8 th, td {
9   border: 1px solid #ddd;
10  padding: 8px;
11  text-align: center;
12 }
13
14 th {
15   background-color: #f2f2f2;
16 }
17
18 h1, h2 {
19   text-align: center;
20 }
21
22 ul {
23   list-style-type: none;
24   padding: 0;
25 }
26
27 li {
28   margin: 5px 0;
29   font-size: 16px;
30 }
31
32 #distribucion {
33   width: 40%;
34   margin: 10px;
35 }
36
37 #promedio-alto, #promedio-bajo, #parciales-reprobados, #promedios {
38   font-weight: bold;
39   color: #333;
40 }
41
```

Y con ayuda de Electron lancé la aplicación, el cual luce así.

Calificaciones

File Edit View Window Help

Calificaciones de Alumnos

| Alumno | Parcial 1 | Parcial 2 | Parcial 3 | Parcial 4 | Promedio |
|----------|-----------|-----------|-----------|-----------|----------|
| Alumno 1 | 8.5 | 7.0 | 6.5 | 9.0 | 7.75 |
| Alumno 2 | 6.0 | 5.5 | 7.5 | 6.0 | 6.25 |
| Alumno 3 | 8.6 | 7.0 | 10.0 | 9.5 | 8.78 |

Información:

Promedio de cada alumno:

Promedio más alto: 8.78

Promedio más bajo: 6.25

Cantidad de parciales reprobados: 4

Distribución de calificaciones finales:

0-4.9: 0 Alumnos

5.0-5.9: 0 Alumnos

6.0-6.9: 1 Alumnos

7.0-7.9: 1 Alumnos

8.0-8.9: 1 Alumnos

9.0-10: 0 Alumnos