Universidad Tecnológica Metropolitana

Estructuras de Datos Aplicadas

ISC. Ruth Betsaida Martínez Domínguez, MGTI

Prácticas 9-10

Soberanis Acosta Jimena Monserrat

Desarrollo de Software Multiplataforma

Cuarto Cuatrimestre

4°B

Parcial I

Jueves, 28 de septiembre de 2023

PRACTICA 9

Primeramente, para este práctica definí cada matriz con lo que se trabajará

```
const matriz1 = [
    [10, 5],
    [8, 2]
];

const matriz2 = [
    [2, 4],
    [6, 8]
];
```

Después, creé una función para cada operación:

Para cada una de las funciones se creó una nueva matriz llamada resultado(nombre de la operación), el cual se utilizará para almacenar el resultado de dicha operación.

Se creó bucles for anidades para que recorran tanto las filas como columnas de las matrices, este bucle se repetirá solamente 2 veces, pues ese es el tamaño máximo.

La operación funciona de tal modo que se sumará cada elemento de matriz1[fila][columna] con su correspondiente posición en la matriz2[fila][columna], y este resultado se guardará en resultadoSuma y lo devolverá.

```
function sumaMatrices(){
    const resultadoSuma =[];

    for (let fila = 0; fila < 2; fila++){
        resultadoSuma.push([]);

        for (let columna = 0; columna < 2; columna++){
            resultadoSuma[fila].push(matriz1[fila][columna]+ matriz2[fila][columna]);
        }
    }
    return resultadoSuma;
}
```

La siguiente operación funciona de manera similar, solo que en esta se realizará una resta de la matriz1[fila][columna] con su correspondiente posición en la matriz2[fila][columna], este resultado se guardará en resultadoResta y se devolverá cuando termine el bucle.

```
function restaMatrices(){
    const resultadoResta =[];

    for (let fila = 0; fila < 2; fila++){
        resultadoResta.push([]);

        for (let columna = 0; columna < 2; columna++){
            resultadoResta[fila].push(matriz1[fila][columna] - matriz2[fila][columna]);
        }
    }
    return resultadoResta;
}
```

De la misma manera funciona la operación del producto, solo que en este en vez de que se realice una suma o resta, se realiza una multiplicación de la matriz1[fila][columna] con su correspondiente posición en la matriz2[fila][columna],

```
function productoMatrices(){
    const resultadoProducto =[];

    for (let fila = 0; fila < 2; fila++){
        resultadoProducto.push([]);

        for (let columna = 0; columna < 2; columna++){
            resultadoProducto[fila].push(matriz1[fila][columna] * matriz2[fila][columna]);
        }
    }
    return resultadoProducto;
}
```

En esta última operación se realiza una comprobación, pues la matriz2 es la divide a la matriz 1, y en tal caso que la matriz divisora (matriz2) sea igual a cero, se mostrará en la consola que no se pude dividir entre cero y se retornará un null como resultado; en caso contrario, si es diferente de 0 se realiza la operación normalmente, pero el resultado se dará con 2 decimales.

```
function divisonMatrices(){
    const resultadoDivision =[];

    for (let fila = 0; fila < 2; fila++){
        resultadoDivision.push([]);

        for (let columna = 0; columna < 2; columna++){
            if(matriz2[fila][columna]!==0){
                const resultado = matriz1[fila][columna] / matriz2[fila][columna];

                const resultadoDivisionRedondeado = resultado.toFixed(2);
                resultadoDivision[fila].push(parseFloat(resultadoDivisionRedondeado));
            }else{
                console.log("No se puede dividir entre cero.");
                return null;
            }
        }
    }
    return resultadoDivision;
}
```

Después de realizar los cálculos de las operaciones entre las 2 matrices, se mostrará primero las matrices con las que se trabajaron, y después los resultados de cada una de las operaciones, en cada apartado se llamó a las funciones con los parámetros de matriz1 y matriz2.

```
68    console.log("Matriz 1: ");
69    console.log(matriz1);
70
71    console.log("Matriz 2: ");
72    console.log(matriz2);
73    console.log('\t');
74
75    console.log("Suma de las matrices:");
76    console.log(sumaMatrices(matriz1, matriz2));
77    console.log('\t');
78
79    console.log("Resta de las matrices:");
80    console.log(restaMatrices(matriz1, matriz2));
81    console.log('\t');
82
83    console.log("Producto de las matrices:");
84    console.log(productoMatrices(matriz1, matriz2));
85    console.log('\t');
86
87    console.log("Division de las matrices:");
88    console.log(divisonMatrices(matriz1, matriz2));
89    console.log('\t');
90
```

Ejemplo:

En la consola, con las matrices definidas arriba, estos son los resultados:

```
PS C:\Users\Jimena\Documents\UTM\Cuatrimestre 4\Estructura de datos aplicados\PARCIAL I\Practica9> node script9.js
Matriz 1:
[ [ 10, 5 ], [ 8, 2 ] ]
Matriz 2:
[ [ 2, 4 ], [ 6, 8 ] ]

Suma de las matrices:
[ [ 12, 9 ], [ 14, 10 ] ]

Resta de las matrices:
[ [ 8, 1 ], [ 2, -6 ] ]

Producto de las matrices:
[ [ 20, 20 ], [ 48, 16 ] ]

Division de las matrices:
[ [ 5, 1.25 ], [ 1.33, 0.25 ] ]
```

Primero se muestran las matrices y después cada uno de los resultados de las operaciones.

PRÁCTICA 10

Comencé en un HTML simple en donde solo se tiene un input para ingresar el numero para el tamaño del cuadrado y un botón para poder enviar la cantidad.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Matriz Cuadrada</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <div class="container">
        <h1>Generador de Matriz Cuadrada</h1>
        <label for="tamanioMatriz">Tamaño del Cuadro:</label>
        <input type="number" id="tamanioMatriz" min="1" step="1">
        <button id="generarMatriz">Generar Matriz</button>
        <div id="tabla-container"></div>
    </div>
    <script src="script.js"></script>
</body>
</html>
```

En la primera función se trata de crear el tamaño del cuadro, qué números contendrá y dónde.

En este función se toma el parámetro número que representa el número de tamaño deseado, inicialmente se crea una matriz vacía, con el uso de dos bucles for anidados se recorre las filas y columnas de la matriz. Se establece para cada elemento que si la fila y la columna son iguales (por ejemplo, si se encuentra en la fila 2 y en la columna 2) este elemento o lugar se le colocará el número 1, en cualquier otro caso o lugar este elemento se le colocará el número 0. Este bucle cesará cuando se llegué el mismo número que el que se ingresó.

```javascript
function crearMatrizCuadrada(numero){
    const matriz = [];

    for (let fila = 0; fila < numero; fila++){
        matriz.push([]);

        for(let columna = 0; columna < numero; columna++){
            if (fila === columna){
                matriz[fila].push(1);
            }else{
                matriz[fila].push(0);
            }
        }
    }
    return matriz;
}
```

En la función anterior, cuando al final se devuelve la matriz (ahora con datos), esta se envía a la siguiente función, que es crear la tabla en el HTML, el cual recibe como parámetro esta matriz anterior.

Se crea un nuevo elemento tabla, se utiliza de nuevo los dos bucles anidades para recorrer as filas y columnas. Por cada elemento de la matriz se crea una fila de la tabla (tr) y un elemento de celda (td) para cada valor en la matriz guardada.

Se establece el contenido de cada celda sea el valor correspondiente de la matriz, se agrega esta fila a la tabla y se continua con la siguiente, y al final devuelve esta tabla.

```
17
18  function crearTablaHTML(matriz) {
19      const tabla = document.createElement('table');
20
21      for (let fila = 0; fila < matriz.length; fila++) {
22          const filaTabla = document.createElement('tr');
23
24          for (let columna = 0; columna < matriz[fila].length; columna++) {
25              const celda = document.createElement('td');
26              celda.textContent = matriz[fila][columna];
27              filaTabla.appendChild(celda);
28          }
29
30          tabla.appendChild(filaTabla);
31      }
32
33      return tabla;
34  }
```

Por último, se tiene al evento que ocurre cuando se oprime el botón, este crea una función que obtiene el tamaño deseado para la matriz desde el input en el HTML. Se verifica que sea un número válido y mayor que 0 para poder continuar. Si el valor es válido, se usa la función para crear los datos de la matriz y luego se llama a crear la tabla en este mismo mundo.

Antes de agregar la nueva tabla, los datos de la tabla anterior es eliminada para nueva información.

```
document.getElementById('generarMatriz').addEventListener('click', function() {
    const tamanoMatriz = parseInt(document.getElementById('tamanioMatriz').value);

    if (!isNaN(tamanoMatriz) && tamanoMatriz > 0) {
        const matriz = crearMatrizCuadrada(tamanoMatriz);
        const tablaHTML = crearTablaHTML(matriz);

        const tablaContainer = document.getElementById('tabla-container');
        tablaContainer.innerHTML = '';
        tablaContainer.appendChild(tablaHTML);
    } else {
        alert("Por favor, ingrese un tamaño válido para el cuadro.");
    }
});
```

Decidí colocarle un diseño con CSS y con ayuda Electron pude levantar la aplicación, el cual quedó de esta manera: