Desarrollo WEB entorno cliente.
UD 2 –Sintaxis del lenguaje JavaScript.
Ejercicios 2.1- Básicos

1. Sintaxis Básica y Lógica de Control (Ejercicios 1 - 21)

Esta sección se centra en variables, entrada/salida de datos, condicionales y bucles.

- **1:** Pedir datos (nombre, edad, profesión) y mostrarlos con `document.write`.
- **2:** Determinar si un número es par o impar.
- **3:** Generar una tabla de multiplicar (1 al 10) usando bucles y HTML.
- **4:** Calculadora básica (suma, resta, etc.) implementada con `if` y luego con `Switch`.
- **5:** Comparar dos números para ver cuál es mayor.
- **6:** Calcular el factorial de un número.
- **7:** Juego de "Adivina el número secreto" (bucle `do...while`).
- **8, 13, 14:** Generación de pirámides numéricas y de asteriscos (lógica de bucles anidados).
- **9:** Verificar personajes de Star Wars (uso de operadores lógicos `||`).
- **10:** Calculadora de precio de billete de autobús con lógica compleja de descuentos (edad, día de la semana).
- **11:** Contar cuántas cifras tiene un número entero.
- **12:** Manejo de comillas simples y dobles dentro de un string.
- **15:** Ordenar alfabéticamente 4 palabras introducidas.
- **16:** Dibujar un rectángulo hueco con caracteres (borde y relleno).
- **17:** Analizar un carácter (si es vocal, número, mayúscula, etc.) usando `Switch`.
- **18:** Mostrar números impares sin usar el operador módulo (`%`).
- **19:** Generar una tabla numérica y calcular sumas de filas y columnas.
- **20:** Juego avanzado de adivinar número con niveles de dificultad y pistas.
- **21:** Calculadora visual con botones y alertas (lógica de acumuladores).

2. Arrays (Ejercicios 1 - 5)

Ejercicios enfocados en la manipulación de vectores/listas.

- **1:** Calcular la media aritmética de un vector de enteros.
- **2:** Validar la letra del DNI usando un array de letras predefinido.
- **3:** Eliminar duplicados de un array y ordenarlo (Uso de `Set`).
- **4:** Ordenar un array de nombres de forma normal e inversa.
- **5:** Generador de calendario que marca festivos almacenados en un array bidimensional.

3. Cadenas de Texto / Strings (Ejercicios 6 - 10)

Manipulación de texto y Expresiones Regulares.

- **6:** Validar formato de email y extraer usuario/dominio.
- **7:** Detector de palíndromos (invirtiendo cadenas y limpiando caracteres).
- **8:** Formatear nombre completo (cambiar orden de Nombre Apellido).
- **9:** Capitalizar frases (Tipo Título: Primera Letra Mayúscula).
- **10:** Validar fecha (dd-mm-aaaa) sin usar el objeto `Date`.

4. Fechas / Objeto Date (Ejercicios 11 - 16)

Uso del objeto `Date` para cálculos temporales.

- **11:** Calcular tiempo transcurrido entre dos clics (cronómetro simple).
- **12:** Formatear fecha de nacimiento.
- **13:** Formatear fecha a texto largo ("Jueves, 26 de noviembre...").
- **14:** Comprobar si un año es bisiesto.
- **15:** Calcular la edad exacta de una persona.
- **16:** Generador de Horario Escolar complejo calculando horas y recreos dinámicamente.

5. DOM y Eventos (Ejercicios 1 - 9)

Interacción dinámica con el HTML y ratón/teclado.

- **1:** Añadir elementos `<li>` aleatorios a una lista.
- **2:** Crear tabla 100x100 y colorear números "casi primos".
- **3:** Gestión masiva de Checkboxes (marcar/desmarcar todos).
- **4:** Ocultar elementos al hacer clic y eliminarlos al hacer doble clic.
- **5:** Mostrar coordenadas (X, Y) del ratón en tiempo real.
- **6:** Uso de `setInterval` para alertas automáticas.
- **7:** Buscador de DNIs que contengan una letra específica al pulsar una tecla.
- **8:** Cambiar color de fondo aleatoriamente con doble clic.
- **9:** *Drag and Drop* (Arrastrar y soltar): Tirar una bola de papel a una papelera.

6. Formularios y Validaciones (Ejercicios 1 - 13)

Validación de datos antes del envío (`onsubmit`, `onblur`, `onchange`).

- **1:** Validación de DNI (formato y letra).
- **2:** Validación de Email (formato y lista blanca de servidores).
- **3:** Validador de Anagramas (comparar dos campos de texto).
- **4:** Selects dependientes (Al elegir Provincia, cargan sus Ciudades).
- **5:** `Textarea` con límite de caracteres que bloquea la escritura al llegar al tope (excepto borrar).
- **6:** Cambiar el fondo de la web usando `Radio Buttons`.
- **7:** Configurar Pizza (Checkboxes) y mostrar resumen en tiempo real en un textarea.
- **8:** Formulario de contacto completo con múltiples validaciones (Regex, edad, obligatorios).
- **9:** Validación de contraseña segura (Mayúscula, minúscula, número, longitud) en tiempo real.
- **10:** Calculadora de precio con descuentos y cupones.
- **11:** Reserva de cita médica (selectores de especialidad y hora dinámica).
- **12:** Registro de usuario con validación de subida de foto (tamaño y tipo).
- **13:** Buscador de productos con filtros múltiples (texto, categoría, precio, disponibilidad).

7. Comunicación Asíncrona (Fetch API & JSON)

Consumo de APIs externas y renderizado de datos.

- **1:** Listar personajes de **Star Wars** (SWAPI) y ver detalles en tarjetas.
- **2:** Listar personajes de **Rick and Morty** con imágenes.
- **Extra 5:** Lista de usuarios con filtrado en tiempo real (JSONPlaceholder).
- **Extra 6:** Galería de fotos con paginación y modal (JSONPlaceholder).
- **Extra 7:** Blog: Ver posts y cargar sus comentarios dinámicamente debajo.
- **Extra 8:** Conversor de monedas en tiempo real (ExchangeRate API).
- **Extra 9:** Buscador de **Pokémon** (PokéAPI) con carga de detalles y estadísticas.

8. Ejercicios CRUD Completos (Extras)

Ejercicios repetitivos para practicar las 4 operaciones (GET, POST, PUT, DELETE) contra APIs de prueba (principalmente JSONPlaceholder y FakeStore).

- **Gestión de:** Tareas (TODOs), Usuarios, Posts de Blog, Comentarios, Álbumes, Productos de Tienda, Empleados, Libros, Clientes y Pedidos.
- **Mecánica común:** Listar en tabla, botón para crear (prompt), botón para editar (prompt) y botón para borrar (por ID), actualizando la interfaz tras cada operación `fetch`.

Crea un documento HTML distinto para cada ejercicio que se llamará, **ejercicioX.html**. Aunque no es lo más aconsejable, pon el código javascript dentro de ese archivo. Recuerda poner comentarios.

1. Crea un script que pida al usuario su nombre, edad y profesión, almacene estos datos en unas variables y los imprima en la página.
   Utiliza document.write();

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejercicio 1: Datos del Usuario</title>
</head>
<body>
    <h1>Ejercicio 1: Pedir Datos al Usuario</h1>

    <script>
        // 1. Pedir los datos al usuario usando prompt()
        // El texto que escriba el usuario se guardará en cada variable.
        const nombre = prompt('Introduce tu nombre:');
        const edad = prompt('Introduce tu edad:');
        const profesion = prompt('Introduce tu profesión:');

        // 2. Almacenar (los datos ya están guardados en las variables de arriba)

        // 3. Imprimir los datos en la página usando document.write()
        // Escribimos un poco de HTML para que se vea ordenado.
        document.write('<h2>Información Recibida:</h2>');
        document.write('<p><strong>Nombre:</strong> ' + nombre + '</p>');
        document.write('<p><strong>Edad:</strong> ' + edad + '</p>');
        document.write('<p><strong>Profesión:</strong> ' + profesion + '</p>');
    </script>

    <noscript>
        <p>Por favor, activa JavaScript para ver este ejercicio.</p>
    </noscript>

</body>
</html>
```

2. Crea un script que pida al usuario un número y muestre en la página un párrafo indicando si el número introducido es par o impar.

```html
<!DOCTYPE html>
<html lang="es">
<head>
```

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejercicio 2: Par o Impar</title>
</head>
<body>
    <h1>Ejercicio 2: Determinar si un número es Par o Impar</h1>

    <script>
        // 1. Pedir al usuario un número
        const numeroInput = prompt('Introduce un número:');

        // 2. Convertir el texto introducido a un número
        // parseInt() convierte un string (texto) en un número entero.
        const numero = parseInt(numeroInput);

        // 3. Comprobar si lo introducido es realmente un número válido
        // isNaN() significa "Is Not a Number" (No es un Número)
        if (isNaN(numero)) {
            document.write('<p>Error: No has introducido un número válido.</p>');
        } else {
            // 4. Comprobar si es par o impar usando el operador módulo (%)
            // El operador módulo (%) devuelve el resto de una división.
            // Si el resto de dividir por 2 es 0, es par.
            if (numero % 2 === 0) {
                // Es par
                document.write('<p>El número ' + numero + ' es
<strong>par</strong>.</p>');
            } else {
                // Es impar
                document.write('<p>El número ' + numero + ' es <strong>im-
par</strong>.</p>');
            }
        }
    </script>

    <noscript>
        <p>Por favor, activa JavaScript para ver este ejercicio.</p>
    </noscript>

</body>
</html>
```

3. Crea un script que genere la tabla de multiplicar de los 10 primeros números. Deben escribirse dentro de una sección y dentro de un table.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <title>Ejercicio 3: Tabla de Multiplicar</title>
</head>
<body>
    <h1>Ejercicio 3: Tabla de Multiplicar (1 al 10)</h1>

    <script>
        // Usamos document.write() para crear los elementos HTML

        // 1. Crear la sección
        document.write('<section>');
        document.write('<h2>Tabla de Multiplicar</h2>');

        // 2. Crear la tabla (le añado border="1" para que se vean las celdas)
        document.write('<table border="1" style="border-collapse: collapse; text-
align: center;">');

        // 3. Crear la fila de cabecera (los números del 1 al 10)
        document.write('<thead>');
        document.write('<tr>');
        document.write('<th>&times;</th>'); // Celda de la esquina
        for (let i = 1; i <= 10; i++) {
            document.write('<th>' + i + '</th>');
        }
        document.write('</tr>');
        document.write('</thead>');

        // 4. Crear el cuerpo de la tabla con los resultados
        document.write('<tbody>');
        // Bucle para las filas (número a multiplicar)
        for (let i = 1; i <= 10; i++) {
            document.write('<tr>');
            // Primera celda de la fila (cabecera de fila)
            document.write('<th>' + i + '</th>');

            // Bucle para las columnas (multiplicador)
            for (let j = 1; j <= 10; j++) {
                // Celda con el resultado
                document.write('<td>' + (i * j) + '</td>');
            }
            document.write('</tr>');
        }
        document.write('</tbody>');

        // 5. Cerrar la tabla y la sección
        document.write('</table>');
        document.write('</section>');
    </script>

    <noscript>
        <p>Por favor, activa JavaScript para ver este ejercicio.</p>
```

```
    </noscript>

</body>
</html>
```

4. Crea un script que tenga definidas dos variables con datos numéricos. Pregunta al usuario la operación que quiere realizar y en función de dicha operación escribe en la página el resultado, indicando toda la operación. Impleméntalo con un if y luego con un Switch.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejercicio 4: Operaciones</title>
</head>
<body>
    <h1>Ejercicio 4: Operaciones con dos números</h1>

    <script>
        // 1. Definir dos variables numéricas
        const num1 = 20;
        const num2 = 4;

        // 2. Preguntar al usuario la operación
        // .toLowerCase() convierte la entrada a minúsculas para facilitar la compa-
ración
        const operacion = prompt('Introduce la operación (suma, resta, multiplica-
cion, division):').toLowerCase();

        // 3. Implementación con 'if...else if...else'
        document.write('<h2>Implementación con <code>if</code>:</h2>');

        if (operacion === 'suma') {
            document.write('<p>' + num1 + ' + ' + num2 + ' = ' + (num1 + num2) +
'</p>');
        } else if (operacion === 'resta') {
            document.write('<p>' + num1 + ' - ' + num2 + ' = ' + (num1 - num2) +
'</p>');
        } else if (operacion === 'multiplicacion') {
            document.write('<p>' + num1 + ' * ' + num2 + ' = ' (num1 * num2) +
'</p>');
        } else if (operacion === 'division') {
            // Comprobación simple para evitar dividir por cero
            if (num2 !== 0) {
                document.write('<p>' + num1 + ' / ' + num2 + ' = ' + (num1 / num2) +
'</p>');
            } else {
```

```
                document.write('<p>Error: No se puede dividir por cero.</p>');
            }
        } else {
            document.write('<p>Operación no reconocida: "' + operacion + '". Por fa-
vor, usa: suma, resta, multiplicacion o division.</p>');
        }

        // 4. Implementación con 'switch'
        document.write('<h2>Implementación con <code>switch</code>:</h2>');

        switch (operacion) {
            case 'suma':
                document.write('<p>' + num1 + ' + ' + num2 + ' = ' + (num1 + num2) +
'</p>');
                break;
            case 'resta':
                document.write('<p>' + num1 + ' - ' + num2 + ' = ' + (num1 - num2) +
'</p>');
                break;
            case 'multiplicacion':
                document.write('<p>' + num1 + ' * ' + num2 + ' = ' + (num1 * num2) +
'</p>');
                break;
            case 'division':
                // Comprobación simple para evitar dividir por cero
                if (num2 !== 0) {
                    document.write('<p>' + num1 + ' / ' + num2 + ' = ' + (num1 /
num2) + '</p>');
                } else {
                    document.write('<p>Error: No se puede dividir por cero.</p>');
                }
                break;
            default:
                document.write('<p>Operación no reconocida: "' + operacion + '". Por
favor, usa: suma, resta, multiplicacion o division.</p>');
        }
    </script>

    <noscript>
        <p>Por favor, activa JavaScript para ver este ejercicio.</p>
    </noscript>

</body>
</html>
```

5. Crea un script que dados dos números que introducirá el usuario por pantalla diga cuál es mayor de los dos.

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejercicio 5: Comparar Dos Números</title>
</head>
<body>
    <h1>Ejercicio 5: ¿Qué número es mayor?</h1>

    <script>
        // 1. Pedir al usuario el primer número
        const num1Input = prompt('Introduce el primer número:');

        // 2. Pedir al usuario el segundo número
        const num2Input = prompt('Introduce el segundo número:');

        // 3. Convertir las entradas a números (usamos parseFloat para permitir decimales)
        const num1 = parseFloat(num1Input);
        const num2 = parseFloat(num2Input);

        // 4. Comprobar si las entradas son números válidos
        if (isNaN(num1) || isNaN(num2)) {
            document.write('<p>Error: Uno o ambos valores introducidos no son números válidos.</p>');
        } else {
            // 5. Comparar los números y mostrar el resultado
            if (num1 > num2) {
                document.write('<p>El primer número (' + num1 + ') es <strong>mayor</strong> que el segundo número (' + num2 + ').</p>');
            } else if (num2 > num1) {
                document.write('<p>El segundo número (' + num2 + ') es <strong>mayor</strong> que el primer número (' + num1 + ').</p>');
            } else {
                document.write('<p>Ambos números son <strong>iguales</strong> (' + num1 + ' y ' + num2 + ').</p>');
            }
        }
    </script>

    <noscript>
        <p>Por favor, activa JavaScript para ver este ejercicio.</p>
    </noscript>

</body>
</html>
```

6. Crea un script que dado un número que introducirá un usuario por pantalla, calcule su factorial. Recuerda que el factorial de un número es el producto de todos los números enteros desde 1 hasta dicho número.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejercicio 6: Factorial de un número</title>
</head>
<body>
    <h1>Ejercicio 6: Factorial de un número</h1>

    <script>
        // 1. Pedir al usuario un número
        const numInput = prompt('Introduce un número para calcular su factorial:');

        // 2. Convertir la entrada a un número entero
        const numero = parseInt(numInput);

        // 3. Comprobar si es un número válido y no negativo
        if (isNaN(numero) || numero < 0) {
            document.write('<p>Error: Por favor, introduce un número entero no negativo (0 o mayor).</p>');
        } else {
            // 4. Calcular el factorial

            let factorial = 1;

            // El factorial de 0 es 1 (caso especial)
            // Para números mayores que 0, calculamos con un bucle
            if (numero > 0) {
                for (let i = 1; i <= numero; i++) {
                    factorial = factorial * i;
                    // Alternativa: factorial *= i;
                }
            }

            // 5. Mostrar el resultado
            document.write('<p>El factorial de ' + numero + ' (es decir, ' + numero + '!) es <strong>' + factorial + '</strong>.</p>');
        }
    </script>

    <noscript>
        <p>Por favor, activa JavaScript para ver este ejercicio.</p>
    </noscript>

</body>
</html>
```

7. Crea un script que pida al usuario la entrada de un número. Este cuadro de diálogo aparecerá hasta que el usuario introduzca un número en concreto que debes escoger a tu elección.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejercicio 7: Adivina el Número</title>
</head>
<body>
    <h1>Ejercicio 7: Adivina el Número Secreto</h1>

    <script>
        // 1. Definir el número secreto (a elección)
        const numeroSecreto = 7;

        let numeroUsuario;

        // 2. Usar un bucle 'do...while' para pedir el número hasta que acierte
        // El bucle 'do...while' se ejecuta al menos una vez, pidiendo el número.
        do {
            // 3. Pedir al usuario un número
            const numInput = prompt('Adivina el número secreto (pista: está entre 1
y 10):');

            // 4. Convertir la entrada a un número entero
            numeroUsuario = parseInt(numInput);

            // El bucle continuará mientras el número introducido (numeroUsuario)
            // sea diferente (!==) al número secreto.

        } while (numeroUsuario !== numeroSecreto);

        // 5. Cuando el bucle termina, significa que el usuario ha acertado.
        document.write('<p>¡Felicidades! Has adivinado el número secreto: <strong>'
+ numeroSecreto + '</strong>.</p>');
    </script>

    <noscript>
        <p>Por favor, activa JavaScript para ver este ejercicio.</p>
    </noscript>

</body>
</html>
```

8. Crea un script que genera una pirámide con los números que tenga la siguiente pinta:
   1

22
333
4444
55555

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        for (let i = 1; i <= 5; i++) {
            for (let j = 1; j <= i; j++) {
                document.write(i);
            }
            document.write("<br>");
        }
    </script>
</body>
</html>
```

9. Crea un script en el que el usuario si introduce (C3PO, R2, Vader o luck) nos responda diciendo que son personajes de Star Wars y si introduce cualquier otro nos responda que no son personajes de la película. Pista: Los datos podrán ser introducidos tanto en minúsculas como en mayúsculas: miCadena.toUppercase();

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Verificador Star Wars</title>
</head>
<body>

    <h2>Comprobar personaje de Star Wars</h2>
    <p>Introduce un nombre (Ej: C3PO, R2, Vader, luck):</p>

    <input type="text" id="nombrePersonaje">

    <button onclick="comprobarPersonaje()">Comprobar</button>

    <p id="resultado"></p>

    <script>
        function comprobarPersonaje() {
```

```
            // 1. Obtener el texto del input
            let nombre = document.getElementById("nombrePersonaje").value;

            // 2. Convertir el texto a mayúsculas (como dice la pista)
            let nombreMayusculas = nombre.toUpperCase();

            // 3. Obtener el elemento donde escribiremos la respuesta
            let pResultado = document.getElementById("resultado");

            // 4. Comprobar si el nombre está en la lista
            if (nombreMayusculas === "C3PO" ||
                nombreMayusculas === "R2"   ||
                nombreMayusculas === "VADER" ||
                nombreMayusculas === "LUCK") {

                // Si coincide, mostramos este mensaje
                pResultado.textContent = nombre + " es un personaje de Star Wars.";

            } else {

                // Si no coincide, mostramos este otro
                pResultado.textContent = nombre + " no es un personaje de la película.";
            }
        }
    </script>

</body>
</html>
```

10. Crea un script que calcule el precio de un billete de autobús, IVA incluido. Para ello, se deben seguir los siguientes criterios:
    • Pedir: nombre, edad y día de la semana que quiere viajar.
    • Si tiene entre 18 y 26 años o es mayor de 60 se le aplicará un descuento del 15%.
    • Si viaja miércoles o viernes, además se le aplicará un descuento del 5%.
    • El precio de un billete simple es de 12,5 euros SIN IVA. • El resultado debe expresar el motivo del descuento y un resumen del precio con IVA y SIN IVA.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Calculadora de Billetes</title>
</head>
<body>

    <h2>Calculadora de Precio de Billete</h2>
```

```
<div>
    <label for="nombre">Nombre:</label>
    <input type="text" id="nombre">
</div>
<div>
    <label for="edad">Edad:</label>
    <input type="number" id="edad">
</div>
<div>
    <label for="dia">Día de la semana:</label>
    <select id="dia">
        <option value="lunes">Lunes</option>
        <option value="martes">Martes</option>
        <option value="miércoles">Miércoles</option>
        <option value="jueves">Jueves</option>
        <option value="viernes">Viernes</option>
        <option value="sábado">Sábado</option>
        <option value="domingo">Domingo</option>
    </select>
</div>

<button onclick="calcularPrecio()">Calcular Precio</button>

<div id="resultado"></div>

<script>
    function calcularPrecio() {
        // --- Constantes ---
        const PRECIO_BASE = 12.50; // Precio sin IVA
        const IVA_TASA = 0.10; // 10% de IVA
        const DESCUENTO_EDAD_TASA = 0.15; // 15%
        const DESCUENTO_DIA_TASA = 0.05;  // 5%

        // --- 1. Obtener datos de entrada ---
        let nombre = document.getElementById("nombre").value;
        let edad = parseInt(document.getElementById("edad").value);
        let dia = document.getElementById("dia").value;

        // Elemento donde escribiremos la respuesta
        let divResultado = document.getElementById("resultado");

        // Validación simple
        if (nombre === "" || isNaN(edad)) {
            divResultado.innerHTML = "Por favor, introduce un nombre y una edad válidos.";
            return; // Detiene la función si faltan datos
        }

        // --- 2. Calcular Descuentos ---
        let tasaDescuentoTotal = 0;
        let motivoDescuento = "";
```

```
            // Comprobar descuento por edad
            if ((edad >= 18 && edad <= 26) || edad > 60) {
                tasaDescuentoTotal += DESCUENTO_EDAD_TASA; // Sumamos 0.15
                motivoDescuento += "Descuento por edad (15%). ";
            }

            // Comprobar descuento por día
            if (dia === "miércoles" || dia === "viernes") {
                tasaDescuentoTotal += DESCUENTO_DIA_TASA; // Sumamos 0.05
                motivoDescuento += "Descuento por día (5%). ";
            }

            // Si no hay motivos, ponemos un mensaje
            if (motivoDescuento === "") {
                motivoDescuento = "Sin descuentos aplicados.";
            }

            // --- 3. Calcular Precios ---
            let descuentoEnEuros = PRECIO_BASE * tasaDescuentoTotal;
            let precioSinIVA = PRECIO_BASE - descuentoEnEuros;
            let precioConIVA = precioSinIVA * (1 + IVA_TASA);

            // --- 4. Mostrar el resultado ---

            // Usamos .innerHTML para poder poner saltos de línea <br>
            // Usamos .toFixed(2) para mostrar siempre 2 decimales
            let resumen = `
                Hola, ${nombre}. Aquí tienes tu resumen:<br><br>
                <b>Motivo del descuento:</b> ${motivoDescuento}<br>
                <hr>
                Precio Base (sin IVA): ${PRECIO_BASE.toFixed(2)} €<br>
                Descuentos aplicados: -${descuentoEnEuros.toFixed(2)} €<br>
                <hr>
                <b>Total (SIN IVA): ${precioSinIVA.toFixed(2)} €</b><br>
                <b>Total (CON 10% IVA): ${precioConIVA.toFixed(2)} €</b>
            `;

            divResultado.innerHTML = resumen;
        }
    </script>

</body>
</html>
```

11. Escribe un algoritmo en JavaScript que pregunte un numero entero, y nos diga el número de cifras que tiene. Si no es un número que siga pidiendo hasta que el usuario ponga un número.

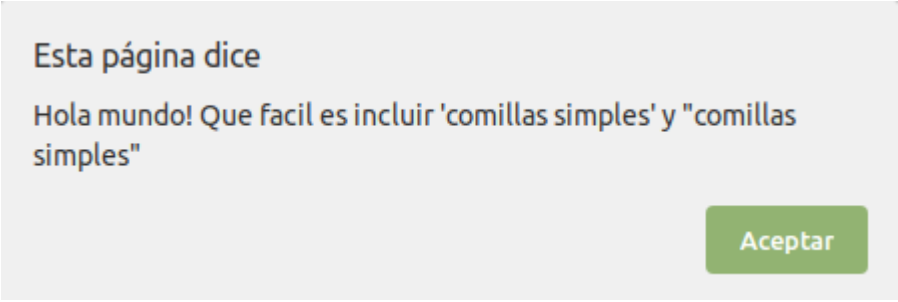
```
<!DOCTYPE html>
<html lang="es">
<head>
```

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Contador de Cifras</title>
</head>
<body>

    <h1>Ejercicio de Contar Cifras</h1>
    <p>Abre la consola o mira los pop-ups para interactuar.</p>

    <button onclick="iniciarConteo()">Empezar Script</button>

    <script>
        function iniciarConteo() {
            let inputUsuario;
            let numero;
            let esEnteroValido = false;

            // 1. Bucle 'do...while' para "seguir pidiendo"
            // Se ejecutará al menos una vez y se repetirá MIENTRAS (while)
            // la condición sea verdadera (es decir, mientras no sea un número válido).
            do {
                inputUsuario = prompt("Introduce un número entero:");

                // Si el usuario pulsa "Cancelar" (inputUsuario es null)
                if (inputUsuario === null) {
                    alert("Operación cancelada por el usuario.");
                    return; // Salimos de la función
                }

                // Intentamos convertir la entrada a un número
                // Usamos parseFloat para poder detectar decimales
                let numeroFlotante = parseFloat(inputUsuario);

                // Comprobamos si es un número Y si es un entero
                // 1. isNaN(numeroFlotante) -> Comprueba si es "hola"
                // 2. numeroFlotante % 1 !== 0 -> Comprueba si es "12.5" (el resto de dividir por 1 es 0.5)
                if (isNaN(numeroFlotante)) {
                    alert("'" + inputUsuario + "' no es un número. Inténtalo de nuevo.");
                    esEnteroValido = false;
                } else if (numeroFlotante % 1 !== 0) {
                     alert("'" + inputUsuario + "' es un decimal, no un entero. Inténtalo de nuevo.");
                    esEnteroValido = false;
                } else {
                    // Si es un número y no tiene decimales, es válido
                    esEnteroValido = true;
                    numero = parseInt(inputUsuario); // Guardamos la versión entera
```

```
            }

        } while (esEnteroValido === false);

        // 2. Si salimos del bucle, 'numero' es un entero válido.
        // Procedemos a contar las cifras.

        // Usamos Math.abs() para quitar el signo negativo (ej: -123)
        // Convertimos el número a un string (texto)
        // Obtenemos la longitud (length) de ese texto
        // Esto funciona para 0 (Math.abs(0).toString() -> "0".length -> 1)
        let cantidadCifras = Math.abs(numero).toString().length;

        // 3. Mostramos el resultado
        alert("El número " + numero + " tiene " + cantidadCifras + " cifras.");
    }

    // Opcional: puedes llamar a la función directamente al cargar la página
    // quitando el botón y descomentando la línea de abajo:
    // iniciarConteo();

</script>

</body>
</html>
```

12. Crea un script para que muestre en pantalla la siguiente imagen. Ojo ha de ir todo en una única cadena.

    Hola mundo! Que fácil es incluir ‘comillas simples’ y “comillas dobles”




```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejercicio de Cadena Única</title>
</head>
<body>
```

```
    <script>

        // --- Solución 1 (Usando comillas dobles y escapando) ---
        // Se inicia la cadena con "
        // Las comillas simples ' se pueden usar sin problema.
        // Las comillas dobles " internas deben "escaparse" con \ para que no finalicen la ca-
dena.

        const miCadena = "Hola mundo! Que fácil es incluir 'comillas simples' y \"comillas do-
bles\"";

        // Mostramos la cadena en una alerta
        alert(miCadena);


        /*
        --- Solución 2 (Usando comillas simples y escapando) ---
        // Esta alternativa también es válida y cumple el requisito.

        const miCadenaAlternativa = 'Hola mundo! Que fácil es incluir \'comillas simples\' y
"comillas dobles"';
        // alert(miCadenaAlternativa);
        */


        /*
        --- Solución 3 (Moderna, con template literals) ---
        // Esta es la forma más fácil hoy en día.
        // Usando comillas invertidas (backticks), puedes incluir ' y " sin escapar.

        const miCadenaModerna = `Hola mundo! Que fácil es incluir 'comillas simples' y "comillas
dobles"`;
        // alert(miCadenaModerna);
        */

    </script>

</body>
</html>
```

13. Crea un script que pida un número y nos muestre la siguiente estructura hasta el impar más próximo al número.

Ej. para número = 8

1
333
55555
7777777

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Estructura Impar</title>
</head>
<body>

    <h2>Estructura de Números Impares</h2>
    <button onclick="generarEstructura()">Pedir número y generar</button>

    <pre id="resultado"></pre>

    <script>
        function generarEstructura() {
            // 1. Pedir el número y validarlo
            let input = prompt("Introduce un número entero (ej. 8):");
            let numero = parseInt(input);

            if (isNaN(numero) || numero < 1) {
                alert("Por favor, introduce un número entero positivo.");
                return;
            }

            // 2. Encontrar el impar más próximo (hacia abajo)
            let limiteImpar;
            if (numero % 2 === 0) {
                // Si es par (como 8), el límite es el anterior (7)
                limiteImpar = numero - 1;
            } else {
                // Si es impar (como 7), el límite es él mismo (7)
                limiteImpar = numero;
            }

            // 3. Generar la estructura
            let output = ""; // Aquí guardaremos todo el texto

            // Bucle 'for' que avanza de 2 en 2 (1, 3, 5, 7...)
            for (let i = 1; i <= limiteImpar; i += 2) {

                // --- Esta es la lógica clave ---
                // Calculamos cuántos asteriscos tocan en esta línea.
                // Es la diferencia entre el límite y el número actual, dividido por 2.
                // Ej. (límite 7):
                // i=1 -> (7-1)/2 = 3 asteriscos
                // i=3 -> (7-3)/2 = 2 asteriscos
                // i=5 -> (7-5)/2 = 1 asterisco
                // i=7 -> (7-7)/2 = 0 asteriscos
                let numAsteriscos = (limiteImpar - i) / 2;

                // Creamos las cadenas usando .repeat()
                let asteriscos = "*".repeat(numAsteriscos);
                let numeros = i.toString().repeat(i);
```

```
                // Juntamos la línea y añadimos un salto de línea (\n)
                output += asteriscos + numeros + asteriscos + "\n";
            }

            // 4. Mostrar el resultado en la etiqueta <pre>
            // Usamos .textContent para que interprete bien los \n (saltos de línea)
            document.getElementById("resultado").textContent = output;
        }
    </script>

</body>
</html>
```

14. Crea un script que pida un número y nos muestre la siguiente estructura hasta el número correspondiente.

Ej. Para número=5
55555
*4444
**333
***22
****1

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Estructura Invertida</title>
</head>
<body>

    <h2>Generador de Estructura Invertida</h2>
    <button onclick="generarEstructura()">Pedir número (ej. 5)</button>

    <pre id="resultado"></pre>

    <script>
        function generarEstructura() {
            // 1. Pedir el número y validarlo
            let input = prompt("Introduce un número entero (ej. 5):");
            let numero = parseInt(input);

            if (isNaN(numero) || numero < 1) {
                alert("Por favor, introduce un número entero positivo.");
                return;
            }

            // 2. Preparar la variable donde guardaremos el texto
            let output = "";
```

```
            // 3. Bucle 'for' que cuenta hacia atrás
            // Empezamos en el 'numero' (ej. 5) y bajamos hasta 1
            for (let i = numero; i >= 1; i--) {

                // 4. Calcular el número de asteriscos
                // La fórmula es: (numero_total - numero_actual)
                // Ej. si numero=5:
                // i=5 -> (5-5) = 0 asteriscos
                // i=4 -> (5-4) = 1 asterisco
                // i=3 -> (5-3) = 2 asteriscos
                let numAsteriscos = numero - i;

                // 5. Crear las cadenas de texto
                // .repeat() crea una cadena repitiendo el texto N veces
                let asteriscos = "*".repeat(numAsteriscos);
                let numeros = i.toString().repeat(i); // 'i' se repite 'i' veces

                // 6. Unir todo y añadir un salto de línea (\n)
                output += asteriscos + numeros + "\n";
            }

            // 7. Mostrar el resultado final en la etiqueta <pre>
            document.getElementById("resultado").textContent = output;
        }
    </script>

</body>
</html>
```

15. Crea un script que pida 4 palabras y las muestre ordenadas alfabéticamente.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ordenar Palabras</title>
</head>
<body>

    <h2>Ordenar 4 Palabras Alfabéticamente</h2>
    <button onclick="iniciarOrden()">Empezar</button>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em;"></div>

    <script>
        function iniciarOrden() {
            // 1. Creamos un array (lista) vacío para guardar las palabras
            let palabras = [];
```

```
            // 2. Pedimos las 4 palabras con un bucle
            // El bucle 'for' se repetirá 4 veces (i=1, i=2, i=3, i=4)
            for (let i = 1; i <= 4; i++) {
                let palabra = prompt("Introduce la palabra #" + i + ":");

                // Verificamos si el usuario pulsó "Cancelar" o dejó el campo vacío
                if (palabra === null || palabra === "") {
                    alert("Operación cancelada o palabra vacía. Inténtalo de nuevo.");
                    return; // Salimos de la función
                }

                // 3. Añadimos la palabra al array
                palabras.push(palabra);
            }

            // 4. Ordenamos el array
            // El método .sort() ordena alfabéticamente los strings por defecto
            palabras.sort();

            // 5. Mostramos el resultado
            let divResultado = document.getElementById("resultado");

            divResultado.innerHTML = "<strong>Palabras ordenadas:</strong><br>" +
                                     palabras.join("<br>"); // .join() une los elementos
        }
    </script>

</body>
</html>
```

16. Crea un script que dado un par de números, alto y ancho cree un rectángulo, el borde y el relleno serán caracteres pedidos. Puedes utilizar las etiquetas <tt> y <pre>

Ej. Alto = 4 y ancho = 6, borde =’%’; relleno=’&’

%%%%%%
%&&&&%
%&&&&%
%%%%%%

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Dibujar Rectángulo</title>
</head>
<body>
```

```html
<h2>Generador de Rectángulos</h2>
<button onclick="dibujar()">Dibujar</button>

<pre id="rectangulo"></pre>

<script>
    function dibujar() {
        // 1. Pedir todos los datos
        let alto = parseInt(prompt("Introduce el alto (ej. 4):"));
        let ancho = parseInt(prompt("Introduce el ancho (ej. 6):"));
        let borde = prompt("Introduce el carácter de borde (ej. %):");
        let relleno = prompt("Introduce el carácter de relleno (ej. &):");

        // Validación simple
        if (isNaN(alto) || isNaN(ancho) || alto <= 0 || ancho <= 0) {
            alert("El alto y el ancho deben ser números positivos.");
            return;
        }
        if (!borde || !relleno) {
            alert("Debes introducir un carácter para borde y relleno.");
            return;
        }

        // 2. Variable para guardar el resultado
        let output = "";

        // 3. Bucle para recorrer las filas (el alto)
        for (let i = 1; i <= alto; i++) {

            // 4. Comprobar si es la PRIMERA o la ÚLTIMA fila
            if (i === 1 || i === alto) {
                // Si es la primera o última, es todo borde
                output += borde.repeat(ancho) + "\n";

            } else {
                // 5. Si es una fila INTERMEDIA

                // Si el ancho es 1, solo se pone un borde
                if (ancho === 1) {
                     output += borde + "\n";
                } else {
                    // Si es mayor que 1, se pone borde + relleno + borde
                    // El relleno es el ancho total menos los 2 bordes
                    let rellenoInterior = relleno.repeat(ancho - 2);
                    output += borde + rellenoInterior + borde + "\n";
                }
            }
        }
```

```
            // 6. Mostrar el resultado en la etiqueta <pre>
            document.getElementById("rectangulo").textContent = output;
        }
    </script>

</body>
</html>
```

17. Crea un script que, dado un carácter, nos diga si es un número, si es vocal, si es consonante, si está en mayúsculas o si es otro distinto a los anteriores. Utiliza Switch case

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Analizador de Carácter</title>
</head>
<body>

    <h2>Analizador de Carácter (con Switch)</h2>
    <button onclick="analizarCaracter()">Analizar Carácter</button>

    <script>
        function analizarCaracter() {
            // 1. Pedir el carácter
            let input = prompt("Introduce UN solo carácter:");
            let mensaje = ""; // Variable para guardar el resultado

            // 2. Validar la entrada
            if (!input || input.length !== 1) {
                alert("Entrada no válida. Debes introducir un solo carácter.");
                return; // Salimos de la función si no es válido
            }

            // 'input' ya es nuestro carácter
            let c = input;

            // 3. Usar 'switch (true)' para evaluar condiciones
            // Esto funciona como una cadena de 'if...else if...else'
            // Comprueba el primer 'case' que sea verdadero y se detiene.
            switch (true) {

                // Caso 1: ¿Es un número?
                case (c >= '0' && c <= '9'):
                    mensaje = "Es un NÚMERO.";
                    break; // Importante salir
```

```
                // Caso 2: ¿Es una vocal minúscula?
                case (c === 'a' || c === 'e' || c === 'i' || c === 'o' || c === 'u'):
                    mensaje = "Es una VOCAL.";
                    break;

                // Caso 3: ¿Es una consonante minúscula?
                // (Es una letra minúscula PERO no es una vocal)
                case (c >= 'a' && c <= 'z'):
                    mensaje = "Es una CONSONANTE.";
                    break;

                // Caso 4: ¿Está en mayúsculas?
                // (Comprobamos si es cualquier letra mayúscula, vocal o consonante)
                case (c >= 'A' && c <= 'Z'):
                    mensaje = "Está en MAYÚSCULAS.";
                    break;

                // Caso 5: Si no es nada de lo anterior
                default:
                    mensaje = "Es OTRO CARÁCTER distinto (ej. $, &, ?).";
            }

            // 4. Mostrar el resultado
            alert("El carácter '" + c + "'... \n" + mensaje);
        }
    </script>

</body>
</html>
```

18. Crea un script que dado un número nos muestre por pantalla todos los impares. Nota sin usar operador módulo (%)

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Mostrar Impares sin Módulo</title>
</head>
<body>

    <h2>Mostrar Impares (Sin usar %)</h2>
    <p>Introduce un número para ver todos los impares hasta él.</p>

    <button onclick="mostrarImpares()">Pedir Número</button>

    <!-- Aquí mostraremos la lista de números -->
    <div id="resultado" style="margin-top: 15px; font-size: 1.2em; line-height: 1.5;"></div>
```

```
    <script>
        function mostrarImpares() {
            // 1. Pedir el número
            let input = prompt("Introduce un número entero:");

            // 2. Convertir a número entero
            let numero = parseInt(input);

            // 3. Validar la entrada
            if (isNaN(numero)) {
                alert("Por favor, introduce un número válido.");
                return; // Salimos de la función si no es un número
            }

            let divResultado = document.getElementById("resultado");

            // Limpiamos el resultado anterior y ponemos un título
            divResultado.innerHTML = "<h4>Impares hasta " + numero + ":</h4>";

            // Si el número es menor que 1, no hay impares que mostrar
            if (numero < 1) {
                divResultado.innerHTML += "No hay números impares.";
                return;
            }

            // 4. Bucle for que empieza en 1 y suma 2 en cada paso
            // Esta es la clave para no usar el operador módulo (%)
            // El bucle solo "tocará" los números 1, 3, 5, 7...
            for (let i = 1; i <= numero; i += 2) {

                // 5. Añadimos cada número impar al div con un salto de línea
                divResultado.innerHTML += i + "<br>";
            }
        }
    </script>

</body>
</html>
```

19. Crea un script que dado un par de números (filas y columnas) escriba una tabla como la del ejemplo

Ej filas=4; columnas =5

20	19	18	17	16	Suma(fil1)=90
15	14	13	12	11	Suma(fil2)=65
10	9	8	7	6	Suma(fil3)=40
5	4	3	2	1	Suma(fil4)=15
Suma(col1)=50	Suma(col2)=46	Suma(col3)=42	Suma(col4)=38	Suma(col5)=34	Suma(fil5)=210

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Generador de Tabla</title>
</head>
<body>

    <h2>Generador de Tabla Numérica</h2>
    <button onclick="generarTabla()">Generar Tabla (Ej: 4 filas, 5 cols)</button>

    <div id="tabla-container"></div>

    <script>
        function generarTabla() {
            // 1. Pedir y validar los datos
            let filas = parseInt(prompt("Introduce el número de FILAS (ej. 4):"));
            let columnas = parseInt(prompt("Introduce el número de COLUMNAS (ej. 5):"));

            if (isNaN(filas) || isNaN(columnas) || filas <= 0 || columnas <= 0) {
                alert("Por favor, introduce números positivos válidos.");
                return;
            }

            // 2. Inicializar variables

            // El contador empieza en el número total (ej. 4 * 5 = 20)
            let contador = filas * columnas;

            // Variable para construir el HTML de la tabla
            // Usamos border='1' para que se vea la cuadrícula
            let htmlTabla = "<table border='1' style='border-collapse: collapse;'>";

            // Arrays para guardar las sumas.
            // .fill(0) crea un array con ceros, ej: [0, 0, 0, 0]
            let sumaFilas = new Array(filas).fill(0);
            let sumaColumnas = new Array(columnas).fill(0);
            let sumaTotal = 0;

            // 3. Bucle para las FILAS (de 0 a filas-1)
            for (let r = 0; r < filas; r++) {
                htmlTabla += "<tr>"; // Empezar una nueva fila

                // 4. Bucle para las COLUMNAS (de 0 a columnas-1)
                for (let c = 0; c < columnas; c++) {
                    let valorActual = contador;

                    // Sumamos el valor a sus totales correspondientes
                    sumaFilas[r] += valorActual;
```

```
                    sumaColumnas[c] += valorActual;
                    sumaTotal += valorActual;

                    // Añadimos la celda (<td>) con el valor
                    htmlTabla += "<td style='padding: 5px;'>" + valorActual + "</td>";

                    // Decrementamos el contador para la siguiente celda
                    contador--;
                }
                // Fin del bucle de columnas

                // 5. Añadir la celda de SUMA DE FILA (al final de la fila)
                htmlTabla += "<td style='padding: 5px;'>Suma(fil" + (r + 1) + ")=" + sumaFi-
las[r] + "</td>";
                htmlTabla += "</tr>"; // Cerrar la fila
            }
            // Fin del bucle de filas

            // 6. Generar la ÚLTIMA FILA (con las sumas de las columnas)
            htmlTabla += "<tr>";
            for (let c = 0; c < columnas; c++) {
                htmlTabla += "<td style='padding: 5px;'>Suma(col" + (c + 1) + ")=" + sumaColum-
nas[c] + "</td>";
            }

            // 7. Añadir la celda de SUMA TOTAL
            // (El ejemplo "Suma(fil5)=210" es confuso, usamos "Suma(total)")
            htmlTabla += "<td style='padding: 5px;'>Suma(total)=" + sumaTotal + "</td>";
            htmlTabla += "</tr>";

            // 8. Cerrar la tabla y mostrarla en el HTML
            htmlTabla += "</table>";
            document.getElementById("tabla-container").innerHTML = htmlTabla;
        }
    </script>

</body>
</html>
```

20. Crea un script que pida dos números, y genere un aleatorio entre ambos números (incluidos). Hay que controlar que sean números entero, si no lo son, los ha de volver a pedir. Ojo que pueden darnos primero el número más alto o el más bajo, lo debemos controlar.

Ayudate de la siguiente funcion:

Math.floor(Math.random() * (MAX - MIN + 1)) + MIN;

Ha de pedir también al usuario el nivel de dificultad. Fácil, normal o difícil.

- Fácil significa que el usuario tiene N/2 intentos siendo N el rango del número aleatorio
- Normal significa que el usuario tiene N/4 intentos siendo N el rango del número aleatorio
- Difícil significa que el usuario tiene N/8 intentos siendo N el rango del número aleatorio

Después se le dirá al usuario que acierte el número, haciendo que inserte un número.

- Si acierta antes de que se le acaben los intentos. El juego terminará felicitando al usuario y pidiéndole su nombre para colocar la puntuación (Número de intentos), la dificultad en la página. Además, se le preguntará si quiere jugar de nuevo, volviendo a pedir todos los datos del juego.
- Si falla se le pedirá otro número hasta que agote los intentos. Además, si el nivel de dificultad es “fácil” o “normal”, cada vez que falle le indicará si el número es mayor o menor al que ha puesto

Pon el código en una función llamada jugar() y haz que se ejecute en atributo onclick del botón. `<button onclick="myFunction()">Click me</button>`

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Juego de Adivinar el Número</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 20px;
        }
        button {
            font-size: 1.2em;
            padding: 10px 20px;
            cursor: pointer;
        }
        #salonFama {
            margin-top: 20px;
            padding: 10px;
            border: 1px solid #ccc;
            background-color: #f9f9f9;
        }
        #salonFama li {
            list-style-type: none;
            margin-bottom: 5px;
            border-bottom: 1px dotted #aaa;
        }
    </style>
</head>
<body>

    <h2>Juego de Adivinar el Número</h2>

    <button onclick="jugar()">¡Empezar a Jugar!</button>

    <div id="salonFama">
        <h3>Puntuaciones:</h3>
        <ul id="listaPuntuaciones">
            </ul>
    </div>
```

```
<script>

    /**
     * Función auxiliar para pedir y validar un NÚMERO ENTERO.
     * Seguirá pidiendo hasta que se introduzca un entero válido.
     * Si el usuario pulsa "Cancelar", devuelve null.
     */
    function pedirNumeroEntero(texto) {
        let input, num;
        do {
            input = prompt(texto);
            if (input === null) {
                return null; // El usuario canceló
            }
            // Usamos parseFloat para detectar decimales
            num = parseFloat(input);

            // Comprobamos si NO es un número (isNaN) o si tiene decimales
            if (isNaN(num) || num !== parseInt(input)) {
                alert("Error: Debes introducir un NÚMERO ENTERO.");
            }

        } while (isNaN(num) || num !== parseInt(input));

        return parseInt(input); // Devolvemos el número ya como entero
    }


    /**
     * Función principal del juego.
     */
    function jugar() {

        // --- 1. PEDIR RANGO Y VALIDAR ---

        let num1 = pedirNumeroEntero("Introduce el primer número entero del rango:");
        // Si el usuario cancela, detenemos el juego.
        if (num1 === null) {
            alert("Juego cancelado.");
            return;
        }

        let num2 = pedirNumeroEntero("Introduce el segundo número entero del rango:");
        if (num2 === null) {
            alert("Juego cancelado.");
            return;
        }

        // Controlamos que el orden no importe
        const MIN = Math.min(num1, num2);
```

```
            const MAX = Math.max(num1, num2);

            // --- 2. PEDIR DIFICULTAD ---

            let dificultad;
            do {
                dificultad = prompt("Elige la dificultad: fácil, normal o difícil");
                if (dificultad === null) {
                    alert("Juego cancelado.");
                    return;
                }
                dificultad = dificultad.toLowerCase().trim(); // Limpiamos la entrada
            } while (dificultad !== "fácil" && dificultad !== "normal" && dificultad !== "difí-
cil");

            // --- 3. CONFIGURAR JUEGO ---

            // Usamos la fórmula que nos diste
            const numeroSecreto = Math.floor(Math.random() * (MAX - MIN + 1)) + MIN;

            // N es el rango de números (sin incluir el min)
            const N_rango = MAX - MIN;
            let totalIntentos;

            switch (dificultad) {
                case "fácil":
                    // Math.ceil() redondea hacia arriba (ej. 4.5 -> 5)
                    // Math.max(1, ...) asegura que haya al menos 1 intento
                    totalIntentos = Math.max(1, Math.ceil(N_rango / 2));
                    break;
                case "normal":
                    totalIntentos = Math.max(1, Math.ceil(N_rango / 4));
                    break;
                case "difícil":
                    totalIntentos = Math.max(1, Math.ceil(N_rango / 8));
                    break;
            }

            alert(`--- ¡Juego Iniciado! ---\nAdivina el número entre ${MIN} y ${MAX}.\nNivel
${dificultad}.\nTienes ${totalIntentos} intentos.`);

            // --- 4. BUCLE DE JUEGO ---

            let intentosUsados = 0;
            let haAcertado = false;

            while (intentosUsados < totalIntentos) {

                let guess = pedirNumeroEntero(`Intento ${intentosUsados + 1} de ${totalInten-
tos}.\nIntroduce tu número:`);
```

```
                if (guess === null) {
                    alert("Juego cancelado.");
                    return;
                }

                intentosUsados++;

                if (guess === numeroSecreto) {
                    haAcertado = true;
                    break; // ¡Victoria! Salimos del bucle
                } else {
                    // Falló
                    let intentosRestantes = totalIntentos - intentosUsados;

                    // Si es "difícil", no damos pistas
                    if (dificultad === "difícil") {
                        if (intentosRestantes > 0) {
                            alert(`Incorrecto. Te quedan ${intentosRestantes} intentos.`);
                        }
                    } else {
                        // "fácil" o "normal" sí damos pistas
                        let pista = (guess < numeroSecreto) ? "MAYOR" : "MENOR";
                        if (intentosRestantes > 0) {
                            alert(`Incorrecto. El número secreto es ${pista}.\nTe quedan ${in-
tentosRestantes} intentos.`);
                        }
                    }
                }
            } // Fin del bucle while

            // --- 5. FIN DE JUEGO (VICTORIA O DERROTA) ---

            if (haAcertado) {
                alert(`¡FELICIDADES! Has adivinado el número ${numeroSecreto} en ${intentosUsa-
dos} intentos.`);

                let nombre = prompt("Introduce tu nombre para el salón de la fama:");
                if (nombre) { // Si el usuario no cancela el nombre
                    let lista = document.getElementById("listaPuntuaciones");
                    lista.innerHTML += `<li><b>${nombre}</b> - ${intentosUsados} intentos (Ni-
vel: ${dificultad}, Rango: ${MIN}-${MAX})</li>`;
                }
            } else {
                alert(`¡GAME OVER! Agotaste tus ${totalIntentos} intentos. El número secreto era
${numeroSecreto}.`);
            }

            // --- 6. JUGAR DE NUEVO ---
```

```
            // confirm() muestra un pop-up con "Aceptar" (true) o "Cancelar" (false)
            if (confirm("¿Quieres jugar de nuevo?")) {
                jugar(); // Volvemos a llamar a la función
            } else {
                alert("Gracias por jugar. ¡Hasta la próxima!");
            }
        }
    </script>

</body>
</html>
```

21. Ya sabemos usar algo, botones enlazarlos a funciones javascript. Realiza una calculadora sencilla como la del dibujo.

Cada vez que pulsemos un digito ha de aparecer en un alert el número, junto con el resto números. Por ejemplo, pulsamos el 3 aparecerá “3” si a continuación pulsamos el cuatro aparecer un alert con “34”. Si ya tenemos el primer número y la operación y pulsamos otro número aparecerá “34 + 7” en un alert.

Si pulsamos una operación antes de tener el primer número no hará nada, sino nos mostrará el primer número más la operación. Ejemplo “34 +” en un alert

Si pulsamos igual antes de tener ambos números a sumar y la operación tampoco hará nada, sino nos mostrará toda la operación. Por ejemplo: “34 + 7 = 41” en un alert

Si pulsamos Reset, la calculadora empezaría de nuevo.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Calculadora con Alertas</title>
    <style>
```

```
        /* Estilos opcionales para que se parezca a la imagen */
        table {
            border-spacing: 5px;
        }
        button {
            width: 100%;
            height: 40px;
            font-size: 1.2em;
            font-family: monospace;
        }
        /* El botón '=' ocupa 2 columnas */
        #btn-igual {
            width: 100%; /* Ajusta al 100% de las 2 celdas combinadas */
        }
    </style>
</head>
<body>

    <h2>Calculadora Sencilla (con Alertas)</h2>

    <table>
        <tr>
            <td><button onclick="pulsarNumero('1')">1</button></td>
            <td><button onclick="pulsarNumero('2')">2</button></td>
            <td><button onclick="pulsarNumero('3')">3</button></td>
            <td><button onclick="pulsarOperacion('+')">+</button></td>
        </tr>
        <tr>
            <td><button onclick="pulsarNumero('4')">4</button></td>
            <td><button onclick="pulsarNumero('5')">5</button></td>
            <td><button onclick="pulsarNumero('6')">6</button></td>
            <td><button onclick="pulsarOperacion('-')">-</button></td>
        </tr>
        <tr>
            <td><button onclick="pulsarNumero('7')">7</button></td>
            <td><button onclick="pulsarNumero('8')">8</button></td>
            <td><button onclick="pulsarNumero('9')">9</button></td>
            <td><button onclick="pulsarOperacion('*')">*</button></td>
        </tr>
        <tr>
            <td><button onclick="pulsarNumero('0')">0</button></td>
            <td colspan="2">
                <button id="btn-igual" onclick="pulsarIgual()">=</button>
            </td>
            <td><button onclick="pulsarOperacion('/')">/</button></td>
        </tr>
        <tr>
            <td><button onclick="pulsarReset()">C</button></td>
            <td></td>
            <td></td>
```

```
            <td></td>
        </tr>
    </table>


    <script>

        // --- Variables Globales ---
        // Estas variables guardan el estado de la calculadora
        let operando1 = "";
        let operando2 = "";
        let operacion = "";

        /**
         * Se llama al pulsar un botón de número (0-9)
         */
        function pulsarNumero(num) {
            // Si 'operacion' está vacía, estamos escribiendo el primer número
            if (operacion === "") {
                operando1 += num; // Concatenamos el número (ej: "3" + "4" = "34")
                alert(operando1);
            }
            // Si 'operacion' YA tiene algo (ej: "+"), estamos escribiendo el segundo número
            else {
                operando2 += num;
                alert(operando1 + " " + operacion + " " + operando2);
            }
        }

        /**
         * Se llama al pulsar un botón de operación (+, -, *, /)
         */
        function pulsarOperacion(op) {
            // Si no se ha escrito el primer número, no hacemos nada
            if (operando1 === "") {
                return;
            }

            // Guardamos la operación
            operacion = op;
            // Mostramos el estado actual
            alert(operando1 + " " + operacion);
        }

        /**
         * Se llama al pulsar el botón de igual (=)
         */
        function pulsarIgual() {
            // Si falta alguno de los 3 componentes, no hacemos nada
            if (operando1 === "" || operando2 === "" || operacion === "") {
```

```
            return;
        }

        // Convertimos los strings a números para poder calcular
        const num1 = parseFloat(operando1);
        const num2 = parseFloat(operando2);
        let resultado;

        // Usamos un switch para realizar el cálculo correcto
        switch (operacion) {
            case '+':
                resultado = num1 + num2;
                break;
            case '-':
                resultado = num1 - num2;
                break;
            case '*':
                resultado = num1 * num2;
                break;
            case '/':
                // Controlamos la división por cero
                if (num2 === 0) {
                    alert("Error: No se puede dividir por cero. Reseteando.");
                    pulsarReset();
                    return; // Salimos de la función
                }
                resultado = num1 / num2;
                break;
            default:
                // Esto no debería ocurrir
                alert("Operación desconocida.");
                return;
        }

        // Mostramos el alert final con el resultado
        alert(operando1 + " " + operacion + " " + operando2 + " = " + resultado);

        // Importante: Guardamos el resultado como el nuevo 'operando1'
        // para poder encadenar operaciones (ej: 5 + 2 = 7, y luego + 3 = 10)
        operando1 = resultado.toString();
        operando2 = "";
        operacion = "";
    }

    /**
     * Se llama al pulsar el botón 'C' (Reset)
     */
    function pulsarReset() {
        operando1 = "";
        operando2 = "";
```

```
            operacion = "";
            alert("Calculadora reseteada.");
        }

    </script>

</body>
</html>
```

EJERCICIOS ARRAYS

1. Crea un script en el que se pida al usuario un vector de números enteros e indique en pantalla la media aritmética de todos sus elementos. Para parar la introducción de número, teclea un 0.

Comprueba que todos los valores son números, antes de hacer la operación.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Media Aritmética de un Vector</title>
</head>
<body>

    <h2>Calculadora de Media Aritmética</h2>
    <p>Introduce números enteros uno por uno. Teclea <b>0</b> para parar y calcular.</p>

    <button onclick="calcularMedia()">Empezar a introducir números</button>

    <!-- Aquí mostraremos el resultado -->
    <div id="resultado" style="margin-top: 20px; font-size: 1.1em; line-height: 1.6;"></div>

    <script>
        function calcularMedia() {
            let vector = []; // Array para guardar los números (el "vector")
            let suma = 0;
            let inputUsuario;
            let numero;

            // 1. Bucle 'while(true)' para pedir números indefinidamente
            while (true) {
                inputUsuario = prompt("Introduce un número entero (teclea 0 para parar):");

                // Si el usuario pulsa "Cancelar"
                if (inputUsuario === null) {
                    alert("Operación cancelada.");
                    return; // Salimos de la función
                }

                // 2. Validar que es un número
                // Usamos parseFloat para poder detectar decimales
                numero = parseFloat(inputUsuario);

                if (isNaN(numero)) {
                    alert("'" + inputUsuario + "' no es un número. Inténtalo de nuevo.");
                    continue; // Vuelve al inicio del bucle
```

```
                }

                // 3. Validar que es un NÚMERO ENTERO
                // (Comprobamos si tiene decimales)
                if (numero % 1 !== 0) {
                     alert("'" + inputUsuario + "' es un decimal. Introduce solo números ENTE-
ROS.");
                     continue; // Vuelve al inicio del bucle
                }

                // 4. Comprobar la condición de parada (el 0)
                if (numero === 0) {
                    break; // Salimos del bucle
                }

                // 5. Si es un número válido y no es 0, lo guardamos
                vector.push(numero);
                suma += numero;
            }

            // 6. Calcular y mostrar el resultado
            let divResultado = document.getElementById("resultado");

            // Comprobamos si se introdujo al menos un número
            if (vector.length === 0) {
                divResultado.innerHTML = "No se introdujo ningún número (además del 0).";
                alert("No se introdujo ningún número para calcular.");
            } else {
                let media = suma / vector.length;

                let resultadoTexto = `
                    <b>Números introducidos (Vector):</b> [${vector.join(", ")}]<br>
                    <b>Suma total:</b> ${suma}<br>
                    <b>Cantidad de elementos:</b> ${vector.length}<br>
                    <hr>
                    <b>Media Aritmética: ${media.toFixed(2)}</b>
                `;

                divResultado.innerHTML = resultadoTexto;

                // También mostramos un alert como pide el ejercicio
                alert("La media aritmética de los " + vector.length + " números es: " + me-
dia.toFixed(2));
            }
        }
    </script>

</body>
</html>
```

2. El cálculo de la letra del Documento Nacional de Identidad (DNI) es un proceso matemático sencillo que se basa en obtener el resto de la división entera del número de DNI y el número 23. Si el resto es 0 será la letra 'T', si es 1 la letra 'R', etc. Se adjunta el array con las letras correspondientes a los restos.

var letras = ['T', 'R', 'W', 'A', 'G', 'M', 'Y', 'F', 'P', 'D', 'X', 'B', 'N', 'J', 'Z', 'S', 'Q', 'V', 'H', 'L', 'C', 'K', 'E', 'T'];

Pídele un DNI al usuario e indícale si el DNI introducido es válido o no.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Validador de DNI</title>
</head>
<body>

    <h2>Validador de Letra de DNI</h2>
    <p>Introduce tu DNI completo (8 números y 1 letra) para comprobar si es válido.</p>

    <button onclick="validarDNI()">Validar DNI</button>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em; font-family: monos-
pace;"></div>

    <script>
        function validarDNI() {

            // 1. El array de letras (tal como se proporcionó)
            var letras = ['T', 'R', 'W', 'A', 'G', 'M', 'Y', 'F', 'P', 'D', 'X', 'B', 'N', 'J',
'Z', 'S', 'Q', 'V', 'H', 'L', 'C', 'K', 'E', 'T'];

            // 2. Pedimos el DNI completo
            let input = prompt("Introduce tu DNI completo (ej: 12345678Z):");
            let divResultado = document.getElementById("resultado");

            // --- 3. Validación del formato ---
            if (!input) {
                return; // El usuario pulsó "Cancelar"
            }

            // Limpiamos la entrada: quitamos espacios y la ponemos en mayúsculas
            let dniCompleto = input.toUpperCase().trim();

            // Usamos una expresión regular (RegEx) para validar el formato 8 números + 1 letra
            let regex = /^\d{8}[A-Z]$/; // 8 dígitos (\d{8}) y 1 letra ([A-Z])

            if (!regex.test(dniCompleto)) {
                let mensajeError = "Formato incorrecto. Debe ser 8 números seguidos de 1 le-
tra.";
                alert(mensajeError);
```

```
                divResultado.innerHTML = mensajeError;
                divResultado.style.color = "red";
                return;
            }

            // --- 4. Extracción de datos y Cálculo ---

            // Separamos los números de la letra
            let numeroStr = dniCompleto.substring(0, 8);
            let letraUsuario = dniCompleto.substring(8); // Coge la letra (último carácter)

            // Convertimos la parte numérica a un número entero
            let numeroDNI = parseInt(numeroStr);

            // Calculamos el resto
            let resto = numeroDNI % 23;

            // Obtenemos la letra que DEBERÍA tener
            let letraCalculada = letras[resto];

            // --- 5. Comparación y Resultado ---
            if (letraCalculada === letraUsuario) {
                // Si la letra calculada coincide con la del usuario, es VÁLIDO
                let mensajeOK = `El DNI ${dniCompleto} es VÁLIDO.`;
                alert(mensajeOK);
                divResultado.innerHTML = mensajeOK;
                divResultado.style.color = "green";
            } else {
                // Si no coinciden, NO es válido
                let mensajeError = `El DNI ${dniCompleto} NO ES VÁLIDO.<br>
                                    La letra correcta para ${numeroDNI} es la <b>${letraCalcu-
lada}</b>.`;
                alert(`El DNI ${dniCompleto} NO ES VÁLIDO. La letra correcta es la ${letraCalcu-
lada}.`);
                divResultado.innerHTML = mensajeError;
                divResultado.style.color = "red";
            }
        }
    </script>

</body>
</html>
```

3. Dado el siguiente array: [4,0,3,4,7,3,5,8,1,8,8,0,2,3,1,2,5,7,3,2,5,1]. Genera un nuevo array con los elementos originales sin repetir y ordenado.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <title>Array Sin Repetir y Ordenado</title>
</head>
<body>

    <h2>Array Sin Repetir y Ordenado</h2>
    <p>Este script procesa un array, elimina los duplicados y lo ordena.</p>

    <button onclick="procesarArray()">Procesar Array</button>

    <div id="resultado" style="margin-top: 15px; font-size: 1.2em; font-family: monos-
pace;"></div>

    <script>
        function procesarArray() {
            // 1. Array original
            const arrayOriginal = [4, 0, 3, 4, 7, 3, 5, 8, 1, 8, 8, 0, 2, 3, 1, 2, 5, 7, 3, 2,
5, 1];

            // 2. Generar el nuevo array

            // a) Creamos un 'Set' a partir del array.
            // Un Set es un objeto que, por definición, solo guarda valores ÚNICOS.
            let setUnico = new Set(arrayOriginal);

            // b) Convertimos el Set de vuelta a un Array
            let arraySinRepetir = Array.from(setUnico);

            // c) Ordenamos el array numéricamente
            // Usamos (a, b) => a - b para asegurar un orden numérico (no alfabético)
            arraySinRepetir.sort((a, b) => a - b);


            // --- Mostrar Resultados ---
            let divResultado = document.getElementById("resultado");
            divResultado.innerHTML = `
                <b>Array Original:</b><br> [${arrayOriginal.join(', ')}]<br><br>
                <b>Array Nuevo (Sin repetir y ordenado):</b><br> [${arraySinRepetir.join(', ')}]
            `;

            // También mostramos un alert
            alert("Array Nuevo: [" + arraySinRepetir.join(', ') + "]");
        }
    </script>

</body>
</html>
```

4. Genera un array con los siguientes nombres: Julián, Rodrigo, Mateo, Valdomero. Haz una ordenación de los mismos en orden normal e inverso.

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ordenar Nombres</title>
</head>
<body>

    <h2>Ordenar Nombres</h2>
    <p>Ordena el array: [Julián, Rodrigo, Mateo, Valdomero]</p>

    <button onclick="ordenarNombres()">Ordenar</button>

    <div id="resultado" style="margin-top: 15px; font-size: 1.2em; font-family: monospace; line-height: 1.6;"></div>

    <script>
        function ordenarNombres() {
            // 1. Array original
            const nombres = ['Julián', 'Rodrigo', 'Mateo', 'Valdomero'];

            // 2. Ordenación Normal (Alfabética)
            // Hacemos una copia con [...nombres] para no modificar el original
            // Usamos .localeCompare('es') para que ordene 'Julián' correctamente con la tilde
            const ordenNormal = [...nombres].sort((a, b) => a.localeCompare(b, 'es'));

            // 3. Ordenación Inversa
            // Hacemos lo mismo, pero invertimos la lógica de comparación (b compara con a)
            const ordenInverso = [...nombres].sort((a, b) => b.localeCompare(a, 'es'));

            // 4. Mostrar resultados
            let divResultado = document.getElementById("resultado");
            divResultado.innerHTML = `
                <b>Array Original:</b><br> [${nombres.join(', ')}]<br><br>
                <b>Orden Normal:</b><br> [${ordenNormal.join(', ')}]<br><br>
                <b>Orden Inverso:</b><br> [${ordenInverso.join(', ')}]
            `;

            // También en un alert
            alert(
                "Original: [" + nombres.join(', ') + "]\n\n" +
                "Normal: [" + ordenNormal.join(', ') + "]\n\n" +
                "Inverso: [" + ordenInverso.join(', ') + "]"
            );
        }
    </script>

</body>
</html>
```

5. Crea un calendario del mes que le pidas al usuario.

Marca en color rojo los días festivos. Guarda los festivos en un array doble indexado por el número del mes (0 a 11), así:

OCTUBRE

Lunes	Martes	Miercoles	Jueves	Viernes	Sabado	Domingo
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

Este array nos dice que en enero hay 2 festivos, el 1 Y el 6 de enero. En febrero, ninguno
Array de festivos de 2025:

var festivos = [[1,6],[],[],[17,18,23],[1],[],[],[15],[],[12],[1],[6,8,25]];

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Generador de Calendario</title>
</head>
<body>

    <h2>Calendario del Mes (2025)</h2>
    <p>Introduce un número del 1 (Enero) al 12 (Diciembre)</p>

    <button onclick="generarCalendario()">Generar Calendario</button>

    <div id="calendario-container"></div>

    <script>
        // Array de festivos de 2025 (0=Enero, 1=Febrero, ...)
        var festivos = [[1,6],[],[],[17,18,23],[1],[],[],[15],[],[12],[1],[6,8,25]];

        // El año es fijo (2025) para que coincida con el array de festivos
        const ANIO = 2025;

        function generarCalendario() {
            // 1. Pedir y validar el mes
            let inputMes = prompt("Introduce el número del mes (1-12):");
            let mes = parseInt(inputMes) - 1; // Convertimos a índice 0-11
```

```
            if (isNaN(mes) || mes < 0 || mes > 11) {
                alert("Mes no válido. Introduce un número entre 1 y 12.");
                return;
            }

            // 2. Obtener datos clave del mes usando el objeto Date

            // Obtenemos el nombre del mes (ej. "octubre")
            let fecha = new Date(ANIO, mes, 1);
            let nombreMes = fecha.toLocaleString('es-ES', { month: 'long' });

            // Obtenemos el día de la semana en que empieza el mes
            // .getDay() da 0(Dom) a 6(Sáb). Lo ajustamos para 0(Lun) a 6(Dom)
            let diaSemanaInicio = (fecha.getDay() + 6) % 7;

            // Obtenemos el número total de días del mes
            // Truco: pedimos el día 0 del mes siguiente (que es el último del mes actual)
            let totalDiasMes = new Date(ANIO, mes + 1, 0).getDate();

            // 3. Empezar a construir el HTML de la tabla
            let html = `<h2>${nombreMes.toUpperCase()} ${ANIO}</h2>`;
            html += "<table>";

            // Fila de cabecera (Lunes a Domingo)
            html += "<thead><tr>";
            const diasSemana = ['Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes', 'Sábado',
'Domingo'];
            for (let dia of diasSemana) {
                html += `<th>${dia}</th>`;
            }
            html += "</tr></thead>";

            // Cuerpo del calendario
            html += "<tbody><tr>";

            let diaActual = 1; // Contador del día que estamos escribiendo

            // 4. Añadir celdas vacías al principio (padding)
            for (let i = 0; i < diaSemanaInicio; i++) {
                html += "<td></td>";
            }

            // 5. Rellenar los días del mes
            let diaSemanaActual = diaSemanaInicio;
            while (diaActual <= totalDiasMes) {

                // Comprobar si el día es festivo
                // festivos[mes] -> [12] (para Octubre)
                // .includes(diaActual) -> ¿[12] incluye el 12? Sí.
                let esFestivo = festivos[mes].includes(diaActual);
```

```
                if (esFestivo) {
                    html += `<td class="festivo">${diaActual}</td>`;
                } else {
                    html += `<td>${diaActual}</td>`;
                }

                diaActual++;
                diaSemanaActual++;

                // Si llegamos al Domingo (día 7), cerramos la fila y abrimos una nueva
                if (diaSemanaActual === 7) {
                    html += "</tr><tr>";
                    diaSemanaActual = 0; // Reseteamos al Lunes
                }
            }

            // 6. Añadir celdas vacías al final (padding)
            while (diaSemanaActual < 7 && diaSemanaActual !== 0) {
                html += "<td></td>";
                diaSemanaActual++;
            }

            // 7. Cerrar la tabla
            html += "</tr></tbody></table>";

            // 8. Mostrar el HTML en la página
            document.getElementById("calendario-container").innerHTML = html;
        }
    </script>

</body>
</html>
```

EJERCICIOS CADENAS.

6. Crear un script que compruebe si un correo electrónico es correcto:

Y caso de serlo muestre:

- el nombre de usuario
- el nombre de dominio

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Validador de Email</title>
</head>
<body>

    <h2>Comprobar Correo Electrónico</h2>
    <p>Introduce un email para validarlo y extraer sus partes.</p>

    <button onclick="validarEmail()">Validar Correo</button>

    <div id="resultado" style="margin-top: 20px; font-size: 1.1em; line-height: 1.6;"></div>

    <script>
        function validarEmail() {
            // 1. Pedir el correo al usuario
            let email = prompt("Introduce el correo electrónico:");
            let divResultado = document.getElementById("resultado");

            if (!email) {
                return; // Salir si el usuario pulsa "Cancelar" o no escribe nada
            }

            // 2. Expresión Regular para validar un email
            // Esta es una regex común y simple:
            // [caracteres]@[caracteres].[caracteres]
            // ^\S+@\S+\.\S+$
            const regex = /^\S+@\S+\.\S+$/;

            // 3. Comprobar si el email cumple el formato
            if (regex.test(email)) {

                // 4. Si es válido, separamos las partes
                // .split('@') divide el string en un array usando '@' como separador
                // ej: "info@dominio.com" -> ["info", "dominio.com"]
                let partes = email.split('@');

                let nombreUsuario = partes[0];
                let nombreDominio = partes[1];

                // 5. Mostramos el resultado VÁLIDO
                let mensajeOK = `
                    El correo <b>${email}</b> es VÁLIDO.<br><br>
                    <b>Nombre de usuario:</b> ${nombreUsuario}<br>
                    <b>Nombre de dominio:</b> ${nombreDominio}
                `;
```

```
                divResultado.innerHTML = mensajeOK;
                divResultado.style.color = "green";

            } else {

                // 6. Mostramos el resultado NO VÁLIDO
                let mensajeError = `El correo <b>${email}</b> NO ES VÁLIDO.`;
                divResultado.innerHTML = mensajeError;
                divResultado.style.color = "red";
            }
        }
    </script>

</body>
</html>
```

7. Crear un script que solicite una cadena y diga si es palíndromo, es decir, se escribe igual de derecha a izquierda y de izquierda a derecha.

Ejemplo de palíndromo: Allí ves Sevilla.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Verificador de Palíndromos</title>
</head>
<body>

    <h2>Verificador de Palíndromos</h2>
    <p>Introduce una frase para ver si es un palíndromo (ej: Allí ves Sevilla).</p>

    <button onclick="verificarPalindromo()">Verificar Frase</button>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em;"></div>

    <script>
        function verificarPalindromo() {
            // 1. Pedir la cadena al usuario
            let fraseOriginal = prompt("Introduce la cadena a verificar:");

            if (fraseOriginal === null || fraseOriginal === "") {
                alert("No has introducido nada.");
                return;
            }

            // 2. Limpiar la cadena
            // a) Convertir a minúsculas
            let cadenaLimpia = fraseOriginal.toLowerCase();
```

```
            // b) Quitar tildes (ej: á -> a)
            // normalize("NFD") separa la letra de la tilde (ej: 'á' -> 'a' + '´')
            // replace(/[\u0300-\u036f]/g, "") elimina todos los caracteres de tilde
            cadenaLimpia = cadenaLimpia.normalize("NFD").replace(/[\u0300-\u036f]/g, "");

            // c) Quitar espacios, comas, puntos, etc.
            // Reemplaza todo lo que NO sea (^) una letra (a-z) o un número (0-9)
            cadenaLimpia = cadenaLimpia.replace(/[^a-z0-9]/g, "");


            // 3. Invertir la cadena limpia
            let cadenaInvertida = cadenaLimpia.split('').reverse().join('');

            // Explicación de la línea anterior:
            // .split('') -> Convierte el string en un array (ej: "hola" -> ['h','o','l','a'])
            // .reverse() -> Invierte el array (ej: ['a','l','o','h'])
            // .join('')  -> Vuelve a unir el array en un string (ej: "aloh")


            // 4. Comparar y mostrar el resultado
            let divResultado = document.getElementById("resultado");

            if (cadenaLimpia === cadenaInvertida) {
                let mensaje = `¡Es un palíndromo!`;
                divResultado.innerHTML = `"${fraseOriginal}"<br><b>${mensaje}</b>`;
                divResultado.style.color = "green";
                alert(mensaje);
            } else {
                let mensaje = `No es un palíndromo.`;
                divResultado.innerHTML = `"${fraseOriginal}"<br><b>${mensaje}</b>
                                          <br><small>(Limpio: ${cadenaLimpia} vs ${cadenaInver-
tida})</small>`;
                divResultado.style.color = "red";
                alert(mensaje);
            }
        }
    </script>

</body>
</html>
```

8. Crear un script que solicite el nombre y apellidos de una persona (en una sola vez) y luego los muestre como “apellidos, nombre”.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Formatear Nombre</title>
```

```
</head>
<body>

    <h2>Formatear Nombre y Apellidos</h2>
    <p>Introduce tu nombre completo (ej: Juan Pérez García) para formatearlo.</p>

    <button onclick="formatearNombre()">Empezar</button>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em;"></div>

    <script>
        function formatearNombre() {
            // 1. Pedir el nombre completo
            let nombreCompleto = prompt("Introduce tu nombre y apellidos:");

            // 2. Validar la entrada
            if (!nombreCompleto) {
                alert("No has introducido nada.");
                return;
            }

            // 3. Procesar la cadena

            // a) Limpiamos espacios al inicio/final
            let nombreLimpio = nombreCompleto.trim();

            // b) Dividimos el string en un array de palabras
            // ej: "Juan Pérez García" -> ['Juan', 'Pérez', 'García']
            let partes = nombreLimpio.split(' ');

            // c) Comprobamos que haya al menos 2 palabras (nombre y apellido)
            if (partes.length < 2) {
                alert("Formato incorrecto. Debes introducir al menos un nombre y un apellido.");
                return;
            }

            // d) Extraemos el primer elemento (el nombre)
            // .shift() saca el primer elemento del array y lo devuelve
            let nombre = partes.shift();

            // e) Lo que queda en el array 'partes' son los apellidos
            // .join(' ') los vuelve a unir con un espacio
            // ej: ['Pérez', 'García'] -> "Pérez García"
            let apellidos = partes.join(' ');

            // 4. Mostrar el resultado
            let resultadoFormateado = `${apellidos}, ${nombre}`;

            document.getElementById("resultado").textContent = resultadoFormateado;
            alert("Formato: " + resultadoFormateado);
```

```
        }
    </script>

</body>
</html>
```

9. Crear un script que solicite una frase y convierta a mayúsculas la primera letra de cada palabra y a minúsculas el resto.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Capitalizar Frase</title>
</head>
<body>

    <h2>Convertir a Mayúscula Inicial (Tipo Título)</h2>
    <p>Introduce una frase (ej: hola mundo) y se convertirá a (Hola Mundo).</p>

    <button onclick="capitalizarFrase()">Convertir Frase</button>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em;"></div>

    <script>
        function capitalizarFrase() {
            // 1. Pedir la frase
            let frase = prompt("Introduce la frase:");

            if (!frase) {
                alert("No has introducido nada.");
                return;
            }

            // 2. Convertir toda la frase a minúsculas primero
            // Esto asegura que si el usuario escribe "hOlA", se convierta en "hola"
            let fraseMinusculas = frase.toLowerCase();

            // 3. Dividir la frase en un array de palabras
            let palabras = fraseMinusculas.split(' ');

            // 4. Recorrer el array y capitalizar cada palabra
            // Usamos .map() que crea un NUEVO array basado en el anterior
            let palabrasCapitalizadas = palabras.map(palabra => {

                // Si la palabra está vacía (por un doble espacio), la devolvemos
                if (palabra.length === 0) {
                    return "";
                }
```

```
                // Cogemos la primera letra (índice 0) y la ponemos en mayúscula
                let primeraLetra = palabra.charAt(0).toUpperCase();

                // Cogemos el resto de la palabra (desde el índice 1)
                let restoPalabra = palabra.slice(1);

                // Las unimos
                return primeraLetra + restoPalabra;
            });

            // 5. Unimos el array de palabras capitalizadas de nuevo en una frase
            let resultado = palabrasCapitalizadas.join(' ');

            // 6. Mostramos el resultado
            document.getElementById("resultado").textContent = resultado;
            alert("Resultado: " + resultado);
        }
    </script>

</body>
</html>
```

10. Crear un script que compruebe una fecha en formato dd-mm-aaaa es correcta. No uses un objeto Date.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Validador de Fecha</title>
</head>
<body>

    <h2>Validador de Fecha (dd-mm-aaaa)</h2>
    <p>Introduce una fecha para comprobar si es correcta (sin usar el objeto Date).</p>

    <button onclick="validarFecha()">Validar Fecha</button>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em;"></div>

    <script>
        function validarFecha() {
            let input = prompt("Introduce la fecha en formato dd-mm-aaaa:");
            let divResultado = document.getElementById("resultado");

            if (!input) {
                return; // Salir si el usuario pulsa "Cancelar"
            }

            // --- 1. Validación de Formato (con Expresión Regular) ---
```

```
        // ^\d{2} -> Empieza con 2 dígitos
        // -\d{2}- -> Sigue con un guion, 2 dígitos y un guion
        // \d{4}$ -> Termina con 4 dígitos
        const regex = /^\d{2}-\d{2}-\d{4}$/;

        if (!regex.test(input)) {
            let mensaje = `Formato incorrecto. Debe ser dd-mm-aaaa.`;
            divResultado.innerHTML = `<b>${input}</b><br>${mensaje}`;
            divResultado.style.color = "red";
            alert(mensaje);
            return;
        }

        // --- 2. Extracción de partes (ya sabemos que el formato es correcto) ---
        let partes = input.split('-');
        let dia = parseInt(partes[0], 10);
        let mes = parseInt(partes[1], 10);
        let anio = parseInt(partes[2], 10);

        // --- 3. Validación de Rango (Mes y Año) ---
        if (mes < 1 || mes > 12) {
            let mensaje = `Fecha incorrecta: El mes (${mes}) debe estar entre 01 y 12.`;
            divResultado.innerHTML = `<b>${input}</b><br>${mensaje}`;
            divResultado.style.color = "red";
            alert(mensaje);
            return;
        }

        if (anio <= 0) {
            let mensaje = `Fecha incorrecta: El año (${anio}) debe ser positivo.`;
            divResultado.innerHTML = `<b>${input}</b><br>${mensaje}`;
            divResultado.style.color = "red";
            alert(mensaje);
            return;
        }

        // --- 4. Validación del Día (basado en el mes y año bisiesto) ---

        // Creamos un array con los días de cada mes
        // (Usamos el índice 0 como placeholder para que mes 1 sea Enero)
        const diasPorMes = [0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31];

        // Comprobar si es bisiesto (para Febrero, mes 2)
        // Un año es bisiesto si es divisible por 4,
        // excepto si es divisible por 100, a menos que también sea divisible por 400.
        if ( (anio % 4 === 0 && anio % 100 !== 0) || (anio % 400 === 0) ) {
            diasPorMes[2] = 29; // Febrero tiene 29 días
        }

        // Obtenemos el máximo de días para el mes introducido
```

```
            let maxDias = diasPorMes[mes];

            if (dia < 1 || dia > maxDias) {
                let mensaje = `Fecha incorrecta: El día (${dia}) no es válido para ${anio}-
${mes}. (Máx: ${maxDias})`;
                divResultado.innerHTML = `<b>${input}</b><br>${mensaje}`;
                divResultado.style.color = "red";
                alert(mensaje);
                return;
            }

            // --- 5. Si pasa todas las validaciones ---
            let mensaje = `La fecha ${input} es VÁLIDA.`;
            divResultado.innerHTML = `<b>${mensaje}</b>`;
            divResultado.style.color = "green";
            alert(mensaje);
        }
    </script>

</body>
</html>
```

EJERCICIOS DE FECHAS.

Usa objetos Date para realizar los siguientes ejercicios:

11. Calcula el tiempo transcurrido entre la pulsación de dos botones rotulados “Empezar a contar”, “Finalizar”.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Tiempo Transcurrido</title>
</head>
<body>

    <h2>Calculadora de Tiempo Transcurrido</h2>
    <p>Pulsa "Empezar" y luego "Finalizar" para medir el tiempo.</p>

    <button id="btnEmpezar" onclick="empezar()">Empezar a contar</button>
    <button id="btnFinalizar" onclick="finalizar()" disabled>Finalizar</button>

    <h3 id="resultado" style="margin-top: 20px;"></h3>

    <script>
        // Variable global para guardar el momento de inicio
        let tiempoInicio;

        function empezar() {
            // 1. Guardamos el momento exacto del clic
```

```
            // new Date() crea un objeto Date con la fecha y hora actuales
            tiempoInicio = new Date();

            // Mostramos feedback y ajustamos los botones
            document.getElementById("resultado").textContent = "Contando...";
            document.getElementById("resultado").style.color = "blue";

            document.getElementById("btnEmpezar").disabled = true;
            document.getElementById("btnFinalizar").disabled = false;
        }

        function finalizar() {
            // 1. Guardamos el momento exacto del segundo clic
            const tiempoFin = new Date();

            // 2. Calculamos la diferencia
            // Al restar dos objetos Date, JavaScript nos da la diferencia en MILISEGUNDOS
            const diferenciaEnMs = tiempoFin - tiempoInicio;

            // 3. Convertimos los milisegundos a segundos (dividiendo por 1000)
            const segundos = diferenciaEnMs / 1000;

            // 4. Mostramos el resultado
            let mensaje = `Tiempo transcurrido: ${segundos.toFixed(2)} segundos.`;
            document.getElementById("resultado").textContent = mensaje;
            document.getElementById("resultado").style.color = "green";

            // 5. Reseteamos los botones para poder jugar de nuevo
            document.getElementById("btnEmpezar").disabled = false;
            document.getElementById("btnFinalizar").disabled = true;
            tiempoInicio = null; // Limpiamos la variable
        }
    </script>

</body>
</html>
```

12. Crea un script en el que se pida la fecha de nacimiento y devuelva la fecha en formato: dd-mm-aaaa, es decir, 26-11-2015.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Formatear Fecha de Nacimiento</title>
</head>
<body>

    <h2>Formatear Fecha de Nacimiento</h2>
    <p>Introduce los datos de tu fecha de nacimiento para formatearla.</p>
```

```
    <button onclick="pedirFecha()">Introducir Fecha</button>

    <h3 id="resultado" style="margin-top: 20px;"></h3>

    <script>
        function pedirFecha() {
            // 1. Pedir las partes de la fecha
            let dia = prompt("Introduce el DÍA de nacimiento (ej: 26):");
            let mes = prompt("Introduce el MES de nacimiento (ej: 11):");
            let anio = prompt("Introduce el AÑO de nacimiento (ej: 2015):");

            // 2. Convertir a números
            let diaNum = parseInt(dia);
            let mesNum = parseInt(mes);
            let anioNum = parseInt(anio);

            // 3. Validación simple de que sean números
            if (isNaN(diaNum) || isNaN(mesNum) || isNaN(anioNum)) {
                alert("Entrada no válida. Por favor, introduce solo números.");
                return;
            }

            // 4. Crear el objeto Date
            // IMPORTANTE: El mes en JavaScript es 0-indexado (0=Enero, 11=Diciembre)
            // Por eso restamos 1 al mes introducido por el usuario.
            const fecha = new Date(anioNum, mesNum - 1, diaNum);

            // 5. Validación extra: Comprobar si la fecha es válida
            // (p.ej. si el usuario puso "31-02-2015", el objeto Date lo "corregiría" a Marzo)
            // Comprobamos si los datos que guardó el objeto Date coinciden con los introducidos
            if (fecha.getFullYear() !== anioNum || fecha.getMonth() !== (mesNum - 1) || fe-
cha.getDate() !== diaNum) {
                alert("La fecha introducida (" + dia + "-" + mes + "-" + anio + ") no es una fe-
cha válida.");
                return;
            }

            // 6. Formatear la salida (queremos 'dd' y 'mm')

            // .padStart(2, '0') añade un '0' a la izquierda si el string no tiene 2 caracteres
            let diaFormateado = diaNum.toString().padStart(2, '0');
            let mesFormateado = mesNum.toString().padStart(2, '0');

            let resultado = `${diaFormateado}-${mesFormateado}-${anioNum}`;

            // 7. Mostrar el resultado
            document.getElementById("resultado").textContent = "Fecha formateada: " + resultado;
            alert("Fecha: " + resultado);
        }
```

```
    </script>

</body>
</html>
```

13. Crea un script que cuando le pases un objeto de tipo Date genere una cadena con el formato: Día de la semana, dd de mm de aaaa, es decir, jueves, 26 de noviembre de 2015.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Formatear Fecha</title>
</head>
<body>

    <h2>Formatear Fecha a Cadena Larga</h2>
    <p>Este script formatea una fecha (ej: 26-11-2015) al formato: "Día de la semana, dd de mm de aaaa".</p>

    <button onclick="mostrarFormato()">Formatear Fecha de Ejemplo</button>

    <h3 id="resultado" style="margin-top: 20px;"></h3>

    <script>

        /**
         * Función que recibe un objeto Date y lo formatea
         * al formato largo en español.
         * @param {Date} fecha - El objeto Date que se va a formatear.
         * @returns {string} - La fecha formateada.
         */
        function formatearFechaLarga(fecha) {

            // 1. Opciones de formato para toLocaleDateString
            // Queremos el día de la semana largo, el mes largo, etc.
            const opciones = {
                weekday: 'long',  // "jueves"
                day: 'numeric',   // "26"
                month: 'long',    // "noviembre"
                year: 'numeric'   // "2015"
            };

            // 2. Usar toLocaleDateString con el local 'es-ES' (Español de España)
            // Esto automáticamente construye la cadena "jueves, 26 de noviembre de 2015"
            // también maneja la capitalización y las comas correctamente.
            return fecha.toLocaleDateString('es-ES', opciones);
        }

        /**
         * Función que se llama con el botón
```

```
         */
        function mostrarFormato() {
            // 1. Creamos un objeto Date de ejemplo (el de la pregunta)
            // Ojo: los meses en JavaScript son 0-indexados (Enero=0, Noviembre=10)
            const fechaEjemplo = new Date(2015, 10, 26); // 26 de Noviembre de 2015

            // 2. Llamamos a nuestra función de formateo
            const fechaFormateada = formatearFechaLarga(fechaEjemplo);

            // 3. Mostramos el resultado
            let mensaje = `La fecha formateada es:<br><b>${fechaFormateada}</b>`;
            document.getElementById("resultado").innerHTML = mensaje;

            alert(fechaFormateada);
        }
    </script>

</body>
</html>
```

14. Crea un script que diga si la fecha es de un año bisiesto.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Comprobar Año Bisiesto</title>
</head>
<body>

    <h2>Comprobar si un Año es Bisiesto</h2>
    <p>Introduce un año (o pulsa el botón para usar la fecha actual).</p>

    <button onclick="comprobarAnioInput()">Introducir un Año</button>
    <button onclick="comprobarAnioActual()">Usar Año Actual</button>

    <h3 id="resultado" style="margin-top: 20px;"></h3>

    <script>

        /**
         * Función principal que comprueba si un año es bisiesto.
         * @param {number} anio - El año a comprobar.
         */
        function esBisiesto(anio) {
            // Un año es bisiesto si es divisible por 4,
            // excepto si es divisible por 100, a menos que también sea divisible por 400.
            if ( (anio % 4 === 0 && anio % 100 !== 0) || (anio % 400 === 0) ) {
                return true;
            } else {
```

```
                return false;
            }
        }

        /**
         * Pide un año al usuario y lo comprueba.
         */
        function comprobarAnioInput() {
            let input = prompt("Introduce el año (ej: 2024):");
            let anioNum = parseInt(input);

            if (isNaN(anioNum) || anioNum <= 0) {
                alert("Por favor, introduce un número de año válido.");
                return;
            }

            let resultado = esBisiesto(anioNum);
            mostrarResultado(anioNum, resultado);
        }

        /**
         * Comprueba el año actual (basado en el reloj del sistema).
         */
        function comprobarAnioActual() {
            // 1. Creamos un objeto Date con la fecha y hora actuales
            const fechaActual = new Date();

            // 2. Obtenemos el año de esa fecha
            const anioActual = fechaActual.getFullYear();

            let resultado = esBisiesto(anioActual);
            mostrarResultado(anioActual, resultado);
        }

        /**
         * Muestra el resultado en la página.
         */
        function mostrarResultado(anio, resultado) {
            let divResultado = document.getElementById("resultado");
            if (resultado) {
                divResultado.textContent = `El año ${anio} SÍ es bisiesto.`;
                divResultado.style.color = "green";
            } else {
                divResultado.textContent = `El año ${anio} NO es bisiesto.`;
                divResultado.style.color = "red";
            }
        }
    </script>
</body>
</html>
```

15. Crea un script que nos devuelva la edad actual de una persona a partir de su fecha de nacimiento.

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Calcular Edad</title>
</head>
<body>

    <h2>Calculadora de Edad</h2>
    <p>Introduce tu fecha de nacimiento para saber tu edad actual.</p>

    <button onclick="iniciarCalculo()">Calcular Edad</button>

    <h3 id="resultado" style="margin-top: 20px;"></h3>

    <script>
        function iniciarCalculo() {
            // 1. Pedir los datos por separado (es más fiable)
            let anio = parseInt(prompt("Introduce el AÑO de nacimiento (ej: 1990):"));
            let mes = parseInt(prompt("Introduce el MES de nacimiento (ej: 11):"));
            let dia = parseInt(prompt("Introduce el DÍA de nacimiento (ej: 26):"));

            if (isNaN(anio) || isNaN(mes) || isNaN(dia)) {
                alert("Datos incorrectos. Introduce solo números.");
                return;
            }

            // 2. Creamos los objetos Date
            // El mes en JS es 0-indexado (0=Enero, 11=Diciembre), por eso restamos 1
            const fechaNacimiento = new Date(anio, mes - 1, dia);
            const fechaActual = new Date();

            // 3. Comprobamos si la fecha es válida (p.ej. que no sea 31/02/1990)
            if (fechaNacimiento.getFullYear() !== anio ||
                fechaNacimiento.getMonth() !== (mes - 1) ||
                fechaNacimiento.getDate() !== dia) {

                alert("La fecha de nacimiento introducida no es válida.");
                return;
            }

            // 4. Calculamos la edad (lógica principal)

            // Resta simple de años
            let edad = fechaActual.getFullYear() - fechaNacimiento.getFullYear();
```

```
            // Comprobamos si el cumpleaños de este año ya ha pasado
            const mesActual = fechaActual.getMonth();
            const diaActual = fechaActual.getDate();
            const mesNac = fechaNacimiento.getMonth();
            const diaNac = fechaNacimiento.getDate();

            // Si el mes actual es anterior al del cumpleaños,
            // O si es el mismo mes pero un día anterior,
            // entonces aún no ha cumplido años este año.
            if (mesActual < mesNac || (mesActual === mesNac && diaActual < diaNac)) {
                edad--; // Restamos un año
            }

            // 5. Mostrar resultado
            let mensaje = `Si naciste el ${dia}-${mes}-${anio}, tu edad actual es: ${edad}
años.`;
            document.getElementById("resultado").textContent = mensaje;
            alert(mensaje);
        }
    </script>

</body>
</html>
```

EJERCICIOS DE COMBINADOS.

16. Crea un script que genere un horario similar a este:

Horario 2º DAM

	Lunes	Martes	Miércoles	Jueves	Viernes
15:45-16:35	IPPE	PROY	IPPE	AD	SGE
16:40-17:30	DI	SGE	AD	AD	SGE
17:35-18:25	DI	SGE	AD	AD	SASP
18:25-18:40	RECREO	RECREO	RECREO	RECREO	RECREO
18:40-19:30	DWC	PMDM	PMDM	PROY	DI
19:35-20:25	DWC	PMDM	PMDM	PSP	DI
20:30-21:20	DWC	PMDM	PMDM	PSP	DI

```
let horario=[
        ["IPPE", "PROY", "IPPE", "AD", "SGE"],
        ["DI", "SGE", "AD", "AD", "SGE"],
        ["DI", "SGE", "AD", "AD", "SASP"],
        ["RECREO","RECREO","RECREO","RECREO","RECREO"],
        ["DWC", "PMDM", "PMDM", "PROY", "DI"],
        ["DWC", "PMDM", "PMDM", "PSP", "DI"],
        ["DWC", "PMDM", "PMDM", "PSP", "DI"]
```

];

- Usa un array bidimensional para guardar los datos que vas a mostrar. Sin horas ni días.
- Usa objetos Date para mostrar las fechas.
- Las horas de la primera columna, se deben de calcular.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Generador de Horario</title>
    <style>
        /* Estilos para que se parezca a la imagen */
        body {
            font-family: Arial, sans-serif;
        }
        table {
            border-collapse: collapse;
            text-align: center;
            margin: 20px 0;
        }
        th, td {
            border: 1px solid black;
            padding: 10px;
            min-width: 80px;
        }
        /* La primera fila (cabecera) */
        thead th {
            background-color: #f0f0f0;
        }
        /* La primera columna (horas) */
        tbody th {
            background-color: #f0f0f0;
            white-space: nowrap; /* Evita que la hora se parta en dos líneas */
        }
        /* Estilo especial para la celda del recreo */
        .recreo {
            background-color: #f7f7f7;
            font-weight: bold;
        }
    </style>
</head>
<body>

    <h2>Horario 2º DAM</h2>
    <button onclick="generarHorario()">Generar Horario</button>

    <div id="horario-container"></div>
```

```
<script>
    // --- 1. DATOS ---

    // Array bidimensional con las asignaturas (como se pidió)
    const horario = [
        ["IPPE", "PROY", "IPPE", "AD", "SGE"],
        ["DI", "SGE", "AD", "AD", "SGE"],
        ["DI", "SGE", "AD", "AD", "SASP"],
        ["RECREO", "RECREO", "RECREO", "RECREO", "RECREO"],
        ["DWC", "PMDM", "PMDM", "PROY", "DI"],
        ["DWC", "PMDM", "PMDM", "PSP", "DI"],
        ["DWC", "PMDM", "PMDM", "PSP", "DI"]
    ];

    // Array para la cabecera de los días
    const dias = ["Lunes", "Martes", "Miércoles", "Jueves", "Viernes"];


    // --- 2. FUNCIÓN DE AYUDA ---

    /**
     * Función auxiliar para formatear un objeto Date a "HH:MM".
     * @param {Date} fecha - El objeto Date del que extraer la hora.
     * @returns {string} - La hora formateada (ej: "15:45").
     */
    function formatTime(fecha) {
        let horas = fecha.getHours().toString().padStart(2, '0');
        let minutos = fecha.getMinutes().toString().padStart(2, '0');
        return `${horas}:${minutos}`;
    }


    // --- 3. FUNCIÓN PRINCIPAL ---

    function generarHorario() {

        // --- 3a. Cálculo de las horas (usando objetos Date) ---
        const horasSlots = [];
        let horaInicio = new Date(); // Objeto Date para los cálculos
        horaInicio.setHours(15, 45, 0, 0); // Establecemos la hora de inicio: 15:45

        for (let i = 0; i < horario.length; i++) { // 7 filas
            let duracionMinutos;
            let pausaMinutos;

            // Definimos la duración de esta franja
            if (i === 3) { // Fila 3 (RECREO)
                duracionMinutos = 15;
            } else {
```

```
                duracionMinutos = 50; // Clase normal
            }

            // Calculamos la hora de fin sumando los minutos
            let horaFin = new Date(horaInicio.getTime() + duracionMinutos * 60000);

            // Guardamos el string "HH:MM-HH:MM"
            horasSlots.push(`${formatTime(horaInicio)}-${formatTime(horaFin)}`);

            // Definimos la pausa para la *siguiente* franja
            // (Entre la 3ª clase y el recreo, y entre el recreo y la 4ª, no hay pausa de 5
min)
            if (i === 2 || i === 3) {
                pausaMinutos = 0;
            } else {
                pausaMinutos = 5;
            }

            // Preparamos la hora de inicio de la siguiente franja
            horaInicio = new Date(horaFin.getTime() + pausaMinutos * 60000);
        }

        // --- 3b. Generación del HTML de la tabla ---
        let html = "<table>";

        // Fila 1: Cabecera (Días)
        html += "<thead><tr><th></th>"; // Celda vacía esquina
        for (let dia of dias) {
            html += `<th>${dia}</th>`;
        }
        html += "</tr></thead>";

        // Filas 2-8: Horas y Asignaturas
        html += "<tbody>";
        for (let i = 0; i < horario.length; i++) { // 7 filas
            html += "<tr>";

            // Columna 1: Horas (Calculadas)
            html += `<th>${horasSlots[i]}</th>`;

            // Columnas 2-6: Asignaturas
            for (let j = 0; j < horario[i].length; j++) { // 5 columnas
                let asignatura = horario[i][j];

                if (asignatura === "RECREO") {
                    html += `<td class="recreo">${asignatura}</td>`;
                } else {
                    html += `<td>${asignatura}</td>`;
                }
            }
```

```
                html += "</tr>";
            }
            html += "</tbody></table>";

            // --- 4. Mostrar en la página ---
            document.getElementById("horario-container").innerHTML = html;
        }
    </script>

</body>
</html>
```

Ejercicios Relacionados Tema 3:

Ejercicio 1: Tabla de Multiplicar Automática

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Tabla de Multiplicar</title>
</head>
<body>
    <h2>Generador de Tabla de Multiplicar</h2>
    <div id="resultado"></div>

    <script>
        let num = parseInt(prompt("Introduce un número (1-10):"));

        if (isNaN(num) || num < 1 || num > 10) {
            alert("Número no válido");
        } else {
            let html = "<table border='1'>";
            html += "<tr><th>Operación</th><th>Resultado</th></tr>";

            for (let i = 1; i <= 10; i++) {
                let resultado = num * i;
                html += `<tr><td>${num} x ${i}</td><td>${resultado}</td></tr>`;
            }

            html += "</table>";
            document.getElementById("resultado").innerHTML = html;
        }
    </script>
</body>
</html>
```

Ejercicio 2: Lista de Alumnos con Notas

```
<!DOCTYPE html>
<html lang="es">
<head>
```

```
    <meta charset="UTF-8">
    <title>Calificaciones</title>
</head>
<body>
    <h2>Sistema de Calificaciones</h2>
    <div id="tabla-notas"></div>

    <script>
        var alumnos = ["Ana García", "Luis Martínez", "María López", "Carlos Ruiz", "Elena To-
rres"];
        var notas = [8.5, 4.2, 9.0, 5.5, 3.8];

        let html = "<table border='1'>";
        html += "<tr><th>Alumno</th><th>Nota</th><th>Estado</th></tr>";

        for (let i = 0; i < alumnos.length; i++) {
            let estado = notas[i] >= 5 ? "APROBADO" : "SUSPENSO";
            let color = notas[i] >= 5 ? "green" : "red";

            html += "<tr>";
            html += `<td>${alumnos[i]}</td>`;
            html += `<td>${notas[i]}</td>`;
            html += `<td style='color:${color}; font-weight:bold'>${estado}</td>`;
            html += "</tr>";
        }

        html += "</table>";
        document.getElementById("tabla-notas").innerHTML = html;
    </script>
</body>
</html>
```

Ejercicio 3: Horario Semanal

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Horario Escolar</title>
    <style>

        td {
            width: 150px;
        }

    </style>
</head>
<body>
    <h2>Mi Horario Semanal</h2>
    <div id="horario"></div>
```

```
    <script>
        var horario = [
            ["Matemáticas", "Lengua", "Inglés", "Educación Física", "Historia"],
            ["Física", "Química", "Matemáticas", "Inglés", "Lengua"],
            ["Inglés", "Historia", "Matemáticas", "Física", "Química"],
            ["Lengua", "Matemáticas", "Educación Física", "Historia", "Inglés"],
            ["Química", "Física", "Lengua", "Matemáticas", "Historia"]
        ];

        var dias = ["Lunes", "Martes", "Miércoles", "Jueves", "Viernes"];
        var horas = ["8:00-9:00", "9:00-10:00", "10:00-11:00", "11:30-12:30", "12:30-13:30"];

        let html = "<table border='1'>";

        // Cabecera con los días
        html += "<tr><th>Hora</th>";
        for (let dia of dias) {
            html += `<th>${dia}</th>`;
        }
        html += "</tr>";

        // Filas con las asignaturas
        for (let i = 0; i < horas.length; i++) {
            html += `<tr><td><b>${horas[i]}</b></td>`;
            for (let j = 0; j < dias.length; j++) {
                html += `<td>${horario[j][i]}</td>`;
            }
            html += "</tr>";
        }

        html += "</table>";
        document.getElementById("horario").innerHTML = html;
    </script>
</body>
</html>
```

Ejercicio 4: Generador de Días del Mes

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Días del Mes</title>
</head>
<body>
    <h2>Cuántos días tiene el mes</h2>
    <div id="resultado"></div>

    <script>
        var meses = ["Enero", "Febrero", "Marzo", "Abril", "Mayo", "Junio",
                     "Julio", "Agosto", "Septiembre", "Octubre", "Noviembre", "Diciembre"];
```

```
        let numMes = parseInt(prompt("Introduce el número del mes (1-12):"));

        if (isNaN(numMes) || numMes < 1 || numMes > 12) {
            alert("Mes no válido");
        } else {
            let anio = parseInt(prompt("Introduce el año:"));

            // Calcular días del mes
            let dias = new Date(anio, numMes, 0).getDate();
            let nombreMes = meses[numMes - 1];

            let html = `<h3>${nombreMes} de ${anio} tiene ${dias} días</h3>`;
            html += "<table border='1'><tr>";

            for (let i = 1; i <= dias; i++) {
                html += `<td>${i}</td>`;
                if (i % 7 === 0) {
                    html += "</tr><tr>";
                }
            }

            html += "</tr></table>";
            document.getElementById("resultado").innerHTML = html;
        }
    </script>
</body>
</html>
```

Ejercicio 5: Lista de Tareas con Prioridades

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Lista de Tareas</title>
</head>
<body>
    <h2>Mis Tareas Pendientes</h2>
    <div id="lista-tareas"></div>

    <script>
        var tareas = [
            "Estudiar JavaScript",
            "Hacer ejercicios HTML",
            "Repasar arrays",
            "Practicar DOM",
            "Revisar fechas"
        ];

        var prioridades = [3, 1, 2, 3, 1]; // 1=Alta, 2=Media, 3=Baja
```

```
        let html = "<table border='1'>";
        html += "<tr><th>Nº</th><th>Tarea</th><th>Prioridad</th></tr>";

        for (let i = 0; i < tareas.length; i++) {
            let textoprioridad;
            let color;

            if (prioridades[i] === 1) {
                textoprioridad = "ALTA";
                color = "red";
            } else if (prioridades[i] === 2) {
                textoprioridad = "MEDIA";
                color = "orange";
            } else {
                textoprioridad = "BAJA";
                color = "green";
            }

            html += "<tr>";
            html += `<td>${i + 1}</td>`;
            html += `<td>${tareas[i]}</td>`;
            html += `<td style='color:${color}; font-weight:bold'>${textoprioridad}</td>`;
            html += "</tr>";
        }

        html += "</table>";
        document.getElementById("lista-tareas").innerHTML = html;
    </script>
</body>
</html>
```

Ejercicio 6: Calendario Simplificado

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Calendario Simple</title>
</head>
<body>
    <h2>Generador de Calendario 2025</h2>
    <div id="calendario"></div>

    <script>
        var festivos = [[1,6],[],[],[17,18,23],[1],[],[],[15],[],[12],[1],[6,8,25]];
        const ANIO = 2025;

        let inputMes = prompt("Introduce el número del mes (1-12):");
        let mes = parseInt(inputMes) - 1;
```

```
        if (isNaN(mes) || mes < 0 || mes > 11) {
            alert("Mes no válido. Introduce un número entre 1 y 12.");
        } else {
            let fecha = new Date(ANIO, mes, 1);
            let nombreMes = fecha.toLocaleString('es-ES', { month: 'long' });
            let diaSemanaInicio = (fecha.getDay() + 6) % 7;
            let totalDiasMes = new Date(ANIO, mes + 1, 0).getDate();

            let html = `<h2>${nombreMes.toUpperCase()} ${ANIO}</h2>`;
            html += "<table border='1'>";

            html += "<thead><tr>";
            const diasSemana = ['Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes', 'Sábado',
'Domingo'];
            for (let dia of diasSemana) {
                html += `<th>${dia}</th>`;
            }
            html += "</tr></thead>";

            html += "<tbody><tr>";

            let diaActual = 1;

            for (let i = 0; i < diaSemanaInicio; i++) {
                html += "<td></td>";
            }

            let diaSemanaActual = diaSemanaInicio;
            while (diaActual <= totalDiasMes) {
                let esFestivo = festivos[mes].includes(diaActual);

                if (esFestivo) {
                    html += `<td style='color:red; font-weight:bold'>${diaActual}</td>`;
                } else {
                    html += `<td>${diaActual}</td>`;
                }

                diaActual++;
                diaSemanaActual++;

                if (diaSemanaActual === 7) {
                    html += "</tr><tr>";
                    diaSemanaActual = 0;
                }
            }

            while (diaSemanaActual < 7 && diaSemanaActual !== 0) {
                html += "<td></td>";
                diaSemanaActual++;
            }
```

```
            html += "</tr></tbody></table>";
            document.getElementById("calendario").innerHTML = html;
        }
    </script>
</body>
</html>
```

DOM Y EVENTOS

1. Haz un programa que cuando se pulse un botón “Nuevo número”, añada un elemento con un número aleatorio a una lista desordenada (elemento UL).

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Añadir Número Aleatorio</title>
</head>
<body>

    <h2>Lista de Números Aleatorios</h2>

    <button onclick="anadirNumero()">Nuevo número</button>

    <ul id="listaNumeros">
        </ul>

    <script>
        function anadirNumero() {
            // 1. Obtener el elemento <ul> donde añadiremos el número
            let lista = document.getElementById("listaNumeros");

            // 2. Generar un número aleatorio (entre 0 y 99)
            // Math.random() da un decimal entre 0 y 1 (ej: 0.123)
            // Lo multiplicamos por 100 (ej: 12.3)
            // Math.floor() lo redondea hacia abajo (ej: 12)
            let numeroAleatorio = Math.floor(Math.random() * 100);

            // 3. Crear un nuevo elemento <li> (un ítem de lista)
            let nuevoElemento = document.createElement("li");

            // 4. Asignar el número aleatorio como texto del <li>
            nuevoElemento.textContent = numeroAleatorio;

            // 5. Añadir (append) el nuevo <li> a la lista <ul>
            lista.appendChild(nuevoElemento);
        }
```

```
    </script>

</body>
</html>
```

2. Realiza un programa que cree dinámicamente una tabla de 100x100. Cada elemento de la tabla tendrá un número único, que empezará en 1 y se irá incrementando de 1 en 1.
Esta página además tendrá un botón que será “Calcular numero casi primos”. Este botón hará que todas las celdas de la tabla que tengan números “Casi primos” se pongan con un fondo amarillo.
Numero casi primo: es un número que solo es divisible por sí mismo, la unidad y por un solo número que no sea ni la unidad ni si mismos.
Ejemplo:
• 2 no es un número casi primo, porque es divisible por 1 y por 2, pero no por otro número.
• 4 es un número casi primo, porque es divisible por 1, por 4 y por 2.
• 8 no es un número casi primo, porque es divisible por 1, por 8 y por 2, pero además también es divisible por 4.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Números Casi Primos</title>
    <style>
        /* Estilos para que la tabla 100x100 sea visible */
        table {
            border-collapse: collapse;
            table-layout: fixed; /* Ancho fijo */
        }
        td {
            border: 1px solid #ddd;
            width: 12px;  /* Celdas muy pequeñas */
            height: 12px;
            font-size: 8px; /* Fuente muy pequeña */
            text-align: center;
            overflow: hidden; /* Ocultar números que no quepan */
        }
        button {
            margin: 15px;
            font-size: 1.2em;
        }
    </style>
</head>
<body>

    <h2>Tabla 100x100 y Números Casi Primos</h2>
    <button onclick="calcularCasiPrimos()">Calcular numero casi primos</button>

    <div id="tabla-container"></div>
```

```
<script>

    /**
     * Función 1: Crea la tabla 100x100 al cargar la página
     */
    function crearTabla() {
        let container = document.getElementById("tabla-container");
        let tabla = document.createElement("table");
        let tbody = document.createElement("tbody");
        let numero = 1;

        for (let i = 0; i < 100; i++) { // 100 filas
            let tr = document.createElement("tr");
            for (let j = 0; j < 100; j++) { // 100 columnas
                let td = document.createElement("td");
                td.textContent = numero;
                // Asignamos un ID único a cada celda para encontrarla después
                td.id = "celda-" + numero;
                tr.appendChild(td);
                numero++;
            }
            tbody.appendChild(tr);
        }
        tabla.appendChild(tbody);
        container.appendChild(tabla);
    }

    /**
     * Función 2: Comprueba si un número es Primo
     * (Necesaria para la función 3)
     */
    function esPrimo(num) {
        if (num <= 1) return false;
        if (num === 2) return true;
        // Un truco: solo necesitamos comprobar hasta la raíz cuadrada
        for (let i = 2; i <= Math.sqrt(num); i++) {
            if (num % i === 0) {
                return false;
            }
        }
        return true;
    }

    /**
     * Función 3: Comprueba si un número es "Casi Primo"
     * (Según la definición, es el cuadrado de un número primo)
     */
    function esCasiPrimo(num) {
        // 1. Obtenemos la raíz cuadrada
```

```
            const raiz = Math.sqrt(num);

            // 2. Si la raíz no es un entero, no es un cuadrado perfecto
            if (raiz % 1 !== 0) {
                return false;
            }

            // 3. Si la raíz es un entero, comprobamos si esa raíz es prima
            return esPrimo(raiz);
        }

        /**
         * Función 4: Se llama al pulsar el botón
         * Recorre todas las celdas y colorea las necesarias.
         */
        function calcularCasiPrimos() {
            // El bucle va hasta 10000 (100 * 100)
            for (let n = 1; n <= 10000; n++) {

                // Si el número es "casi primo"...
                if (esCasiPrimo(n)) {
                    // Seleccionamos la celda por su ID
                    let celda = document.getElementById("celda-" + n);

                    // (Comprobamos que exista, por si acaso)
                    if (celda) {
                        celda.style.backgroundColor = "yellow";
                    }
                }
            }
        }

        // --- Ejecutamos la creación de la tabla cuando se carga la página ---
        window.onload = crearTabla;

    </script>

</body>
</html>
```

3. Haz un programa que cree 100 elementos “checkbox” con números aleatorios. Además, la página tendrá un botón “Marcar todos” y un botón “Desmarcar todos”.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <title>Checkboxes Aleatorios</title>
    <style>
        /* Estilos opcionales para que la lista no ocupe toda la página */
        #checkbox-container {
            border: 1px solid #ccc;
            height: 300px; /* Altura fija */
            overflow-y: auto; /* Scroll vertical */
            padding: 10px;
            margin-top: 15px;
            width: 300px;
        }
        /* Cada checkbox en su propia línea */
        #checkbox-container div {
            display: block;
            margin-bottom: 5px;
        }
    </style>
</head>
<body>

    <h2>100 Checkboxes Aleatorios</h2>

    <button onclick="marcarTodos()">Marcar todos</button>
    <button onclick="desmarcarTodos()">Desmarcar todos</button>

    <!-- El contenedor donde se generarán los checkboxes -->
    <div id="checkbox-container"></div>

    <script>

        /**
         * Función 1: Crea los 100 checkboxes al cargar la página
         */
        function crearCheckboxes() {
            // Seleccionamos el contenedor
            let container = document.getElementById("checkbox-container");

            for (let i = 0; i < 100; i++) {
                // Generamos un número aleatorio (ej. 0-999)
                let numAleatorio = Math.floor(Math.random() * 1000);
                let idUnico = "cb-" + i;

                // Creamos un <div> para agrupar el check y su label
                let wrapper = document.createElement("div");

                // 1. Crear el <input type="checkbox">
                let checkbox = document.createElement("input");
                checkbox.type = "checkbox";
                checkbox.id = idUnico; // ID único
```

```
                // 2. Crear el <label> para el número
                let label = document.createElement("label");
                label.textContent = " " + numAleatorio; // El texto es el número
                label.setAttribute("for", idUnico); // Lo enlazamos al checkbox

                // 3. Añadimos el checkbox y el label al wrapper
                wrapper.appendChild(checkbox);
                wrapper.appendChild(label);

                // 4. Añadimos el wrapper al contenedor principal
                container.appendChild(wrapper);
            }
        }

        /**
         * Función 2: Marca todos los checkboxes
         */
        function marcarTodos() {
            // Seleccionamos TODOS los checkboxes que están dentro del contenedor
            let checkboxes = document.querySelectorAll("#checkbox-container input[type='check-
box']");

            // Recorremos la lista de checkboxes y los marcamos
            checkboxes.forEach(function(cb) {
                cb.checked = true;
            });
        }

        /**
         * Función 3: Desmarca todos los checkboxes
         */
        function desmarcarTodos() {
            // Seleccionamos TODOS los checkboxes
            let checkboxes = document.querySelectorAll("#checkbox-container input[type='check-
box']");

            // Recorremos la lista y los desmarcamos
            checkboxes.forEach(function(cb) {
                cb.checked = false;
            });
        }

        // --- Ejecutamos la creación de checkboxes cuando la página termine de cargar ---
        window.onload = crearCheckboxes;

    </script>

</body>
</html>
```

4. Haz un programa que tenga 3 elementos <p> y al hacer clic sobre ellos desaparezcan (se oculten) y al hacer doble clic los elimine. También deberá tener un botón “Reaparecer” que hará que aparezcan todos los elementos desaparecidos (pero no los eliminados).

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ocultar y Eliminar Elementos</title>
</head>
<body>

    <h2>Clic para ocultar, Doble Clic para eliminar</h2>

    <button id="btnReaparecer">Reaparecer</button>
    <hr>

    <p class="parrafo-interactivo">Párrafo 1</p>
    <p class="parrafo-interactivo">Párrafo 2</p>
    <p class="parrafo-interactivo">Párrafo 3</p>

    <script>
        // --- 1. Seleccionamos todos los párrafos ---
        // Usamos querySelectorAll para obtener una lista de todos los elementos
        // que coinciden con la clase '.parrafo-interactivo'
        const parrafos = document.querySelectorAll('.parrafo-interactivo');

        // --- 2. Añadimos los listeners a cada párrafo ---

        // Recorremos la lista de párrafos con un forEach
        parrafos.forEach(function(p) {

            // Evento 1: Clic Sencillo (Ocultar)
            p.addEventListener('click', function(event) {
                // 'event.target' es el párrafo específico que se ha clicado
                event.target.style.display = 'none';
            });

            // Evento 2: Doble Clic (Eliminar)
            p.addEventListener('dblclick', function(event) {
                // .remove() elimina el elemento del DOM permanentemente
                event.target.remove();
            });
        });

        // --- 3. Funcionalidad del botón "Reaparecer" ---

        // Seleccionamos el botón
        const btn = document.getElementById('btnReaparecer');
```

```
        btn.addEventListener('click', function() {
            // Volvemos a seleccionar TODOS los párrafos que AÚN EXISTAN
            // (Los eliminados no serán encontrados por querySelectorAll)
            const parrafosOcultos = document.querySelectorAll('.parrafo-interactivo');

            parrafosOcultos.forEach(function(p) {
                // Restablecemos su estilo a 'block' (que es el normal para un <p>)
                p.style.display = 'block';
            });
        });
    </script>

</body>
</html>
```

5. Haz un programa que mediante eventos y el uso del objeto event, te muestre en todo momento la posición actual del ratón en pantalla.
Para mostrarlo modificaremos de forma dinámica un elemento XHTML (Ejemplo, un <p>) que nos muestre la posición actual del ratón.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Posición del Ratón</title>
    <style>

    </style>
</head>
<body>

    <h1>Sigue la posición del ratón</h1>
    <p>Mueve el ratón por la ventana para ver las coordenadas.</p>

    <p id="posicion-raton">Posición X: 0, Posición Y: 0</p>

    <script>
        // 1. Seleccionamos el párrafo donde mostraremos la info
        const pElemento = document.getElementById('posicion-raton');

        // 2. Añadimos un 'event listener' a toda la ventana
        // 'mousemove' es el evento que se dispara cada vez que el ratón se mueve
        window.addEventListener('mousemove', function(event) {

            // 3. El objeto 'event' contiene la información del movimiento
            // 'event.clientX' es la posición X (horizontal)
            // 'event.clientY' es la posición Y (vertical)
            let posX = event.clientX;
            let posY = event.clientY;
```

```
            // 4. Actualizamos el texto del párrafo
            pElemento.textContent = `Posición X: ${posX}, Posición Y: ${posY}`;
        });
    </script>

</body>
</html>
```

6. Realizar un programa con dos botones “Comenzar Saludos” y “Parar Saludos”. Al hacer click en “Comenzar Saludos”, lance un setInterval que cada 5 segundos muestre un alert con “Hola”. El botón “Parar Saludos” parará esa secuencia.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Saludos Automáticos</title>
</head>
<body>

    <h2>Saludos Automáticos</h2>
    <p>Haz click en "Comenzar Saludos" para iniciar una secuencia de alertas cada 5 segun-
dos.</p>

    <button onclick="comenzarSaludos()">Comenzar Saludos</button>
    <button onclick="pararSaludos()">Parar Saludos</button>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em; font-family: monos-
pace;"></div>

    <script>
        // Variable global para guardar el identificador del setInterval
        let intervalo = null;

        function comenzarSaludos() {
            // --- 1. Verificamos si ya hay un intervalo en ejecución ---
            if (intervalo !== null) {
                alert("Los saludos ya están en ejecución.");
                return; // Salimos de la función para no crear múltiples intervalos
            }

            // --- 2. Iniciamos el setInterval ---
            // setInterval ejecuta una función cada X milisegundos (5000ms = 5 segundos)
            intervalo = setInterval(function() {
                alert("Hola");
            }, 5000);

            // --- 3. Mostramos mensaje de confirmación ---
            let mensaje = "Saludos iniciados. Se mostrará un 'Hola' cada 5 segundos.";
```

```
            document.getElementById("resultado").innerHTML = mensaje;
            document.getElementById("resultado").style.color = "green";

            alert("Saludos iniciados. Se mostrará un 'Hola' cada 5 segundos.");
        }

        function pararSaludos() {
            // --- 1. Verificamos si hay un intervalo activo ---
            if (intervalo === null) {
                alert("No hay ningún saludo en ejecución.");
                return; // No hay nada que parar
            }

            // --- 2. Detenemos el setInterval ---
            // clearInterval() detiene el intervalo usando su identificador
            clearInterval(intervalo);

            // --- 3. Reseteamos la variable a null ---
            intervalo = null;

            // --- 4. Mostramos mensaje de confirmación ---
            let mensaje = "Saludos detenidos.";
            document.getElementById("resultado").innerHTML = mensaje;
            document.getElementById("resultado").style.color = "red";

            alert("Saludos detenidos.");
        }
    </script>

</body>
</html>
```

7. Haz un programa que cuando se presione una tecla, calcule cuantos DNIs de 4 cifras (del 0001 al 9999) tienen esa letra y te mostrará los DNIs y sus letras en un elemento XHTML (Ejemplo, un <p>).

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Buscador de DNIs por Letra</title>
</head>
<body>

    <h2>Buscador de DNIs por Letra</h2>
    <p>Presiona cualquier tecla (letra) para buscar todos los DNIs de 4 cifras (0001-9999) que tienen esa letra.</p>

    <div id="resultado" style="margin-top: 20px; font-size: 1em; font-family: monospace; white-space: pre-wrap;"></div>
```

```
    <script>
        // --- 1. Array de letras del DNI ---
        var letras = ['T', 'R', 'W', 'A', 'G', 'M', 'Y', 'F', 'P', 'D', 'X', 'B', 'N', 'J', 'Z',
'S', 'Q', 'V', 'H', 'L', 'C', 'K', 'E'];

        // --- 2. Evento para detectar cuando se presiona una tecla ---
        document.addEventListener('keypress', function(evento) {

            // Obtenemos la tecla presionada y la convertimos a mayúscula
            let teclaPresionada = evento.key.toUpperCase();

            // --- 3. Validamos que sea una letra válida del array ---
            if (!letras.includes(teclaPresionada)) {
                document.getElementById("resultado").innerHTML =
                    `La tecla '${teclaPresionada}' no es una letra válida de DNI.\n` +
                    `Letras válidas: ${letras.join(', ')}`;
                document.getElementById("resultado").style.color = "red";
                return;
            }

            // --- 4. Buscamos todos los DNIs con esa letra ---
            let resultados = []; // Array para guardar los DNIs encontrados
            let contador = 0; // Contador de DNIs encontrados

            // Recorremos todos los números del 1 al 9999
            for (let numeroDNI = 1; numeroDNI <= 9999; numeroDNI++) {

                // Calculamos el resto al dividir entre 23
                let resto = numeroDNI % 23;

                // Obtenemos la letra correspondiente
                let letraCalculada = letras[resto];

                // Si la letra coincide con la tecla presionada
                if (letraCalculada === teclaPresionada) {
                    contador++;

                    // Formateamos el número con ceros a la izquierda (0001, 0002, etc.)
                    let numeroFormateado = numeroDNI.toString().padStart(4, '0');

                    // Guardamos el resultado en el array
                    resultados.push(`${numeroFormateado}${letraCalculada}`);
                }
            }

            // --- 5. Mostramos los resultados ---
            let divResultado = document.getElementById("resultado");

            if (contador > 0) {
```

```
                // Creamos el mensaje con todos los DNIs encontrados
                let mensaje = `Se encontraron ${contador} DNIs con la letra '${teclaPresio-
nada}':\n\n`;
                mensaje += resultados.join('\n');

                divResultado.innerHTML = mensaje;
                divResultado.style.color = "green";
            } else {
                divResultado.innerHTML = `No se encontraron DNIs con la letra '${teclaPresio-
nada}'.`;
                divResultado.style.color = "red";
            }
        });

        // --- 6. Mensaje inicial ---
        document.getElementById("resultado").innerHTML = "Esperando que presiones una tecla...";
        document.getElementById("resultado").style.color = "blue";
    </script>

</body>
</html>
```

8. Haz un programa que al hacer doble click en la pantalla del navegador, cambie el fondo a un color aleatorio.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Cambiar Color de Fondo</title>
</head>
<body>

    <h2>Cambiar Color de Fondo Aleatorio</h2>
    <p>Haz <strong>doble click</strong> en cualquier parte de la pantalla para cambiar el color
de fondo a un color aleatorio.</p>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em; font-family: monos-
pace;"></div>

    <script>
        // --- 1. Evento para detectar doble click en cualquier parte del documento ---
        document.addEventListener('dblclick', function() {

            // --- 2. Generamos tres valores RGB aleatorios (0-255) ---
            // Math.random() genera un número entre 0 y 1
            // Lo multiplicamos por 256 y redondeamos hacia abajo para obtener un número entre 0
y 255
            let r = Math.floor(Math.random() * 256); // Valor Rojo (Red)
```

```
            let g = Math.floor(Math.random() * 256); // Valor Verde (Green)
            let b = Math.floor(Math.random() * 256); // Valor Azul (Blue)

            // --- 3. Creamos el string del color en formato RGB ---
            let colorAleatorio = `rgb(${r}, ${g}, ${b})`;

            // --- 4. Aplicamos el color al fondo del body ---
            document.body.style.backgroundColor = colorAleatorio;

            // --- 5. Mostramos información del color generado ---
            let mensaje = `Color de fondo cambiado a: ${colorAleatorio}<br>`;
            mensaje += `RGB(${r}, ${g}, ${b})`;

            document.getElementById("resultado").innerHTML = mensaje;

            // Calculamos si el color es claro u oscuro para ajustar el color del texto
            // Fórmula de luminosidad: (0.299*R + 0.587*G + 0.114*B)
            let luminosidad = (0.299 * r + 0.587 * g + 0.114 * b);

            // Si el fondo es oscuro, ponemos el texto en blanco; si es claro, en negro
            if (luminosidad < 128) {
                document.body.style.color = "white";
            } else {
                document.body.style.color = "black";
            }
        });

        // --- 6. Mensaje inicial ---
        document.getElementById("resultado").innerHTML = "Haz doble click para cambiar el color
de fondo...";
    </script>

</body>
</html>
```

9. Realiza un programa que tenga una imagen de una bola de papel y una papelera vacía. Cuando se arrastre la bola de papel a la papelera vacía, deberá cambiar la imagen de la papelera vacía a una papelera llena.

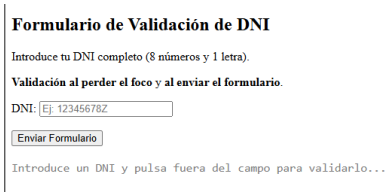
```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Arrastrar Papel a la Papelera</title>
</head>
<body>

    <h2>Arrastra el Papel a la Papelera</h2>
    <p>Arrastra la bola de papel hacia la papelera para tirarla.</p>
```

```
    <!-- Imagen de la bola de papel (draggable) -->
    <img id="papel"
         src="https://cdn-icons-png.flaticon.com/512/3524/3524388.png"
         alt="Bola de papel"
         draggable="true"
         style="width: 100px; cursor: move; margin: 20px;">

    <br><br>

    <!-- Imagen de la papelera (zona de destino) -->
    <img id="papelera"
         src="https://cdn-icons-png.flaticon.com/512/3687/3687412.png"
         alt="Papelera vacía"
         style="width: 150px; margin: 20px; border: 3px dashed #ccc; padding: 10px;">

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em; font-family: monos-
pace;"></div>

    <script>
        // --- 1. Referencias a los elementos ---
        let papel = document.getElementById("papel");
        let papelera = document.getElementById("papelera");
        let divResultado = document.getElementById("resultado");

        // URLs de las imágenes de la papelera
        const PAPELERA_VACIA = "https://cdn-icons-png.flaticon.com/512/3687/3687412.png";
        const PAPELERA_LLENA = "https://cdn-icons-png.flaticon.com/512/3687/3687408.png";

        // --- 2. Evento cuando SE EMPIEZA a arrastrar el papel ---
        papel.addEventListener('dragstart', function(evento) {
            // Guardamos el id del elemento que estamos arrastrando
            evento.dataTransfer.setData('text', evento.target.id);
            divResultado.innerHTML = "Arrastrando papel...";
            divResultado.style.color = "blue";
        });

        // --- 3. Evento cuando el papel pasa POR ENCIMA de la papelera ---
        papelera.addEventListener('dragover', function(evento) {
            // Por defecto, no se permite soltar elementos
            // Debemos prevenir el comportamiento por defecto para permitirlo
            evento.preventDefault();

            // Cambiamos el borde para indicar que es una zona válida
            papelera.style.border = "3px solid green";
        });

        // --- 4. Evento cuando el papel SALE de la zona de la papelera ---
        papelera.addEventListener('dragleave', function(evento) {
            // Restauramos el borde original
```

```
            papelera.style.border = "3px dashed #ccc";
        });

        // --- 5. Evento cuando SE SUELTA el papel sobre la papelera ---
        papelera.addEventListener('drop', function(evento) {
            // Prevenimos el comportamiento por defecto
            evento.preventDefault();

            // Obtenemos el id del elemento que se arrastró
            let idElemento = evento.dataTransfer.getData('text');
            let elementoArrastrado = document.getElementById(idElemento);

            // --- Verificamos que sea el papel ---
            if (idElemento === 'papel') {
                // Cambiamos la imagen de la papelera a "llena"
                papelera.src = PAPELERA_LLENA;
                papelera.alt = "Papelera llena";

                // Ocultamos el papel (lo "tiramos")
                elementoArrastrado.style.display = "none";

                // Restauramos el borde
                papelera.style.border = "3px solid #ccc";

                // Mostramos mensaje de éxito
                divResultado.innerHTML = "¡Papel tirado a la papelera correctamente! ✓";
                divResultado.style.color = "green";
            }
        });

        // --- 6. Mensaje inicial ---
        divResultado.innerHTML = "Arrastra la bola de papel hacia la papelera...";
        divResultado.style.color = "gray";
    </script>

</body>
</html>
```

Formularios

1.Realiza un formulario donde se pueda introducir y enviar un DNI con letra. El formulario deberá validar si la letra es correcta al: Ejer 1-a) Perder el foco del campo de texto donde se introduce el DNI. Ejer 1-b) Enviar el formulario, cancelando el envío si el formato no es correcto.



```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Validación de DNI en Formulario</title>
</head>
<body>

    <h2>Formulario de Validación de DNI</h2>
    <p>Introduce tu DNI completo (8 números y 1 letra).</p>
    <p><strong>Validación al perder el foco</strong> y <strong>al enviar el formula-
rio</strong>.</p>

    <!-- Formulario -->
    <form id="formularioDNI" onsubmit="return validarFormulario()">

        <label for="campoDNI">DNI:</label>
        <input type="text"
               id="campoDNI"
               name="dni"
               placeholder="Ej: 12345678Z"
               onblur="validarDNIEnFoco()"
               maxlength="9"
               required>

        <br><br>

        <button type="submit">Enviar Formulario</button>

    </form>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em; font-family: monos-
pace;"></div>

    <script>
        // --- 1. Array de letras del DNI ---
        var letras = ['T', 'R', 'W', 'A', 'G', 'M', 'Y', 'F', 'P', 'D', 'X', 'B', 'N', 'J', 'Z',
'S', 'Q', 'V', 'H', 'L', 'C', 'K', 'E'];

        // --- 2. Función para validar el DNI (reutilizable) ---
        function validarDNI(dniCompleto) {
            // Limpiamos la entrada: quitamos espacios y ponemos en mayúsculas
            dniCompleto = dniCompleto.toUpperCase().trim();

            // Expresión regular: 8 dígitos + 1 letra
            let regex = /^\d{8}[A-Z]$/;

            // Verificamos el formato
            if (!regex.test(dniCompleto)) {
```

```
                return {
                    valido: false,
                    mensaje: "Formato incorrecto. Debe ser 8 números seguidos de 1 letra."
                };
            }

            // Extraemos número y letra
            let numeroStr = dniCompleto.substring(0, 8);
            let letraUsuario = dniCompleto.substring(8);

            // Convertimos a número
            let numeroDNI = parseInt(numeroStr);

            // Calculamos el resto
            let resto = numeroDNI % 23;

            // Obtenemos la letra correcta
            let letraCalculada = letras[resto];

            // Comparamos
            if (letraCalculada === letraUsuario) {
                return {
                    valido: true,
                    mensaje: `El DNI ${dniCompleto} es VÁLIDO ✓`
                };
            } else {
                return {
                    valido: false,
                    mensaje: `El DNI ${dniCompleto} NO ES VÁLIDO. La letra correcta es '${letra-
Calculada}'.`
                };
            }
        }

        // --- 3. EJERCICIO 1-a) Validar al perder el foco (onblur) ---
        function validarDNIEnFoco() {
            let campoDNI = document.getElementById("campoDNI");
            let divResultado = document.getElementById("resultado");
            let valorDNI = campoDNI.value;

            // Solo validamos si hay algo escrito
            if (valorDNI.trim() === "") {
                divResultado.innerHTML = "";
                return;
            }

            // Llamamos a la función de validación
            let resultado = validarDNI(valorDNI);

            // Mostramos el resultado
```

```
        divResultado.innerHTML = resultado.mensaje;

        if (resultado.valido) {
            // DNI válido: color verde y borde verde
            divResultado.style.color = "green";
            campoDNI.style.border = "2px solid green";
        } else {
            // DNI inválido: color rojo y borde rojo
            divResultado.style.color = "red";
            campoDNI.style.border = "2px solid red";
        }
    }

    // --- 4. EJERCICIO 1-b) Validar al enviar el formulario (onsubmit) ---
    function validarFormulario() {
        let campoDNI = document.getElementById("campoDNI");
        let divResultado = document.getElementById("resultado");
        let valorDNI = campoDNI.value;

        // Validamos si el campo está vacío
        if (valorDNI.trim() === "") {
            alert("El campo DNI no puede estar vacío.");
            divResultado.innerHTML = "Error: El campo DNI está vacío.";
            divResultado.style.color = "red";
            campoDNI.style.border = "2px solid red";
            return false; // Cancelamos el envío
        }

        // Llamamos a la función de validación
        let resultado = validarDNI(valorDNI);

        if (resultado.valido) {
            // DNI válido: permitimos el envío
            alert("✓ Formulario válido. DNI correcto: " + valorDNI.toUpperCase());
            divResultado.innerHTML = "Formulario enviado correctamente ✓";
            divResultado.style.color = "green";
            campoDNI.style.border = "2px solid green";

            // En un caso real, aquí se enviaría el formulario
            // return true; permitiría el envío
            // Por ahora lo cancelamos para que no recargue la página
            return false;
        } else {
            // DNI inválido: cancelamos el envío
            alert("✗ Error: " + resultado.mensaje);
            divResultado.innerHTML = resultado.mensaje;
            divResultado.style.color = "red";
            campoDNI.style.border = "2px solid red";
            return false; // Cancelamos el envío
        }
```

```
        }

        // --- 5. Mensaje inicial ---
        document.getElementById("resultado").innerHTML = "Introduce un DNI y pulsa fuera del
campo para validarlo...";
        document.getElementById("resultado").style.color = "gray";
    </script>

</body>
</html>
```

2. Realiza un formulario que pida una dirección de email y la valide antes de enviarla: Ejer 2-a) Debe validar si el email sigue el formato texto@servidor.loquesea Ejer 2-b) Además de validar el formato anterior, debe comprobar que servidor.loquesea este como servidor admitido en un array de servidores llamado "listaServidores". Dicho array debe ser definido a mano en el código. Ejemplo listaServidores=["terra.es","google.com","marca.es","yahoo.es"];


```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Validación de Email</title>
</head>
<body>

    <h2>Formulario de Validación de Email</h2>
    <p>Introduce tu dirección de correo electrónico.</p>
    <p><strong>Validación de formato</strong> y <strong>servidor permitido</strong>.</p>

    <!-- Formulario -->
    <form id="formularioEmail" onsubmit="return validarFormulario()">
```

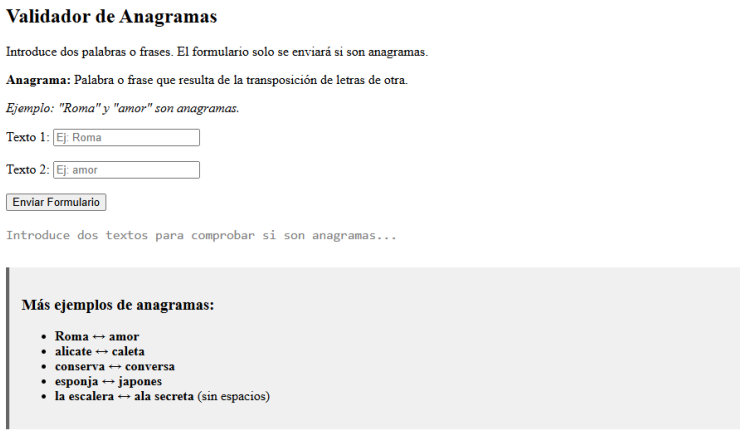
```
        <label for="campoEmail">Email:</label>
        <input type="text"
               id="campoEmail"
               name="email"
               placeholder="Ej: usuario@google.com"
               onblur="validarEmailEnFoco()"
               required>

        <br><br>

        <button type="submit">Enviar Formulario</button>

    </form>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em; font-family: monos-
pace;"></div>

    <div id="infoServidores" style="margin-top: 20px; font-size: 0.9em; font-family: monospace;
color: #666;">
        <strong>Servidores permitidos:</strong> <span id="listaServidoresTexto"></span>
    </div>

    <script>
        // --- 1. Array de servidores admitidos ---
        let listaServidores = ["terra.es", "google.com", "marca.es", "yahoo.es"];

        // Mostramos la lista de servidores permitidos en pantalla
        document.getElementById("listaServidoresTexto").innerHTML = listaServidores.join(", ");

        // --- 2. EJERCICIO 2-a) Función para validar formato básico del email ---
        function validarFormatoEmail(email) {
            // Expresión regular para validar formato: texto@servidor.dominio
            // ^ = inicio de cadena
            // [^\s@]+ = uno o más caracteres que NO sean espacios ni @
            // @ = arroba obligatoria
            // [^\s@]+ = uno o más caracteres que NO sean espacios ni @ (servidor)
            // \. = punto literal
            // [^\s@]+ = uno o más caracteres que NO sean espacios ni @ (dominio)
            // $ = fin de cadena
            let regex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

            return regex.test(email);
        }

        // --- 3. EJERCICIO 2-b) Función para validar si el servidor está en la lista ---
        function validarServidorPermitido(email) {
            // Extraemos la parte del servidor (después del @)
            // Ejemplo: usuario@google.com → google.com
            let partes = email.split('@');
```

```
            // Verificamos que haya exactamente 2 partes (antes y después del @)
            if (partes.length !== 2) {
                return {
                    valido: false,
                    servidor: null
                };
            }

            let servidor = partes[1].toLowerCase(); // Convertimos a minúsculas

            // Comprobamos si el servidor está en la lista
            if (listaServidores.includes(servidor)) {
                return {
                    valido: true,
                    servidor: servidor
                };
            } else {
                return {
                    valido: false,
                    servidor: servidor
                };
            }
        }

        // --- 4. Función completa de validación (formato + servidor) ---
        function validarEmail(email) {
            // Limpiamos la entrada
            email = email.trim().toLowerCase();

            // Verificamos si el campo está vacío
            if (email === "") {
                return {
                    valido: false,
                    mensaje: "El campo email no puede estar vacío."
                };
            }

            // PASO 1: Validamos el formato
            if (!validarFormatoEmail(email)) {
                return {
                    valido: false,
                    mensaje: "Formato de email incorrecto. Debe seguir el formato: texto@servi-
dor.dominio"
                };
            }

            // PASO 2: Validamos el servidor
            let resultadoServidor = validarServidorPermitido(email);
```

```
            if (!resultadoServidor.valido) {
                return {
                    valido: false,
                    mensaje: `El servidor '${resultadoServidor.servidor}' NO está permitido.<br>
                             Servidores válidos: ${listaServidores.join(", ")}`
                };
            }

            // Si llegamos aquí, todo es válido
            return {
                valido: true,
                mensaje: `✓ Email válido: ${email}<br>Servidor permitido: ${resultadoServi-
dor.servidor}`
            };
        }

        // --- 5. Validar al perder el foco (onblur) ---
        function validarEmailEnFoco() {
            let campoEmail = document.getElementById("campoEmail");
            let divResultado = document.getElementById("resultado");
            let valorEmail = campoEmail.value;

            // Solo validamos si hay algo escrito
            if (valorEmail.trim() === "") {
                divResultado.innerHTML = "";
                campoEmail.style.border = "";
                return;
            }

            // Llamamos a la función de validación
            let resultado = validarEmail(valorEmail);

            // Mostramos el resultado
            divResultado.innerHTML = resultado.mensaje;

            if (resultado.valido) {
                // Email válido: color verde y borde verde
                divResultado.style.color = "green";
                campoEmail.style.border = "2px solid green";
            } else {
                // Email inválido: color rojo y borde rojo
                divResultado.style.color = "red";
                campoEmail.style.border = "2px solid red";
            }
        }

        // --- 6. Validar al enviar el formulario (onsubmit) ---
        function validarFormulario() {
            let campoEmail = document.getElementById("campoEmail");
            let divResultado = document.getElementById("resultado");
```

```
            let valorEmail = campoEmail.value;

            // Llamamos a la función de validación
            let resultado = validarEmail(valorEmail);

            if (resultado.valido) {
                // Email válido: permitimos el envío
                alert("✓ Formulario válido. Email correcto: " + valorEmail.toLowerCase());
                divResultado.innerHTML = "Formulario enviado correctamente ✓";
                divResultado.style.color = "green";
                campoEmail.style.border = "2px solid green";

                // return true; permitiría el envío real
                // Por ahora lo cancelamos para que no recargue la página
                return false;
            } else {
                // Email inválido: cancelamos el envío
                alert("✗ Error de validación:\n" + resultado.mensaje.replace(/<br>/g, '\n'));
                divResultado.innerHTML = resultado.mensaje;
                divResultado.style.color = "red";
                campoEmail.style.border = "2px solid red";
                return false; // Cancelamos el envío
            }
        }

        // --- 7. Mensaje inicial ---
        document.getElementById("resultado").innerHTML = "Introduce un email y pulsa fuera del
campo para validarlo...";
        document.getElementById("resultado").style.color = "gray";
    </script>

</body>
</html>
```

3. Realiza un formulario que tenga dos campos de texto. Deberá validar antes de enviarse si uno de los campos es un anagrama del otro y enviarse solo si lo es. Se ignorarán mayúsculas, minúsculas y espacios. https://es.wikipedia.org/wiki/Anagrama Ejemplo de anagrama Roma y amor



```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Validador de Anagramas</title>
</head>
<body>

    <h2>Validador de Anagramas</h2>
    <p>Introduce dos palabras o frases. El formulario solo se enviará si son anagramas.</p>
    <p><strong>Anagrama:</strong> Palabra o frase que resulta de la transposición de letras de otra.</p>
    <p><em>Ejemplo: "Roma" y "amor" son anagramas.</em></p>

    <!-- Formulario -->
    <form id="formularioAnagrama" onsubmit="return validarFormulario()">

        <label for="texto1">Texto 1:</label>
        <input type="text"
               id="texto1"
               name="texto1"
               placeholder="Ej: Roma"
               onblur="validarAnagramaEnFoco()"
               required>

        <br><br>

        <label for="texto2">Texto 2:</label>
        <input type="text"
               id="texto2"
               name="texto2"
               placeholder="Ej: amor"
               onblur="validarAnagramaEnFoco()"
               required>

        <br><br>

        <button type="submit">Enviar Formulario</button>

    </form>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em; font-family: monospace;"></div>

    <div style="margin-top: 30px; padding: 15px; background-color: #f0f0f0; border-left: 4px solid #666;">
        <h3>Más ejemplos de anagramas:</h3>
        <ul>
            <li><strong>Roma</strong> ↔ <strong>amor</strong></li>
```

```
        <li><strong>alicate</strong> ↔ <strong>caleta</strong></li>
        <li><strong>conserva</strong> ↔ <strong>conversa</strong></li>
        <li><strong>esponja</strong> ↔ <strong>japones</strong></li>
        <li><strong>la escalera</strong> ↔ <strong>ala secreta</strong> (sin espacios)</li>
    </ul>
</div>

<script>
    // --- 1. Función para normalizar texto (quitar espacios, acentos, minúsculas) ---
    function normalizarTexto(texto) {
        // Convertimos a minúsculas
        texto = texto.toLowerCase();

        // Quitamos todos los espacios en blanco
        texto = texto.replace(/\s+/g, '');

        // Quitamos tildes/acentos (opcional pero recomendado)
        texto = texto.normalize("NFD").replace(/[\u0300-\u036f]/g, "");

        return texto;
    }

    // --- 2. Función para ordenar las letras de un texto ---
    function ordenarLetras(texto) {
        // Convertimos el string en un array de caracteres
        // Ordenamos alfabéticamente
        // Volvemos a unir en un string
        return texto.split('').sort().join('');
    }

    // --- 3. Función principal: validar si son anagramas ---
    function sonAnagramas(texto1, texto2) {
        // Normalizamos ambos textos (minúsculas, sin espacios)
        let texto1Normalizado = normalizarTexto(texto1);
        let texto2Normalizado = normalizarTexto(texto2);

        // Verificamos que no estén vacíos
        if (texto1Normalizado === "" || texto2Normalizado === "") {
            return {
                sonAnagramas: false,
                mensaje: "Ambos campos deben tener contenido."
            };
        }

        // Verificamos que tengan la misma longitud (obligatorio para ser anagramas)
        if (texto1Normalizado.length !== texto2Normalizado.length) {
            return {
                sonAnagramas: false,
                mensaje: `NO son anagramas.<br>
                          '${texto1}' tiene ${texto1Normalizado.length} letras.<br>
```

```
                    '${texto2}' tiene ${texto2Normalizado.length} letras.`
            };
        }

        // Ordenamos las letras de ambos textos
        let texto1Ordenado = ordenarLetras(texto1Normalizado);
        let texto2Ordenado = ordenarLetras(texto2Normalizado);

        // Si los textos ordenados son iguales, son anagramas
        if (texto1Ordenado === texto2Ordenado) {
            return {
                sonAnagramas: true,
                mensaje: `✓ ¡SÍ son anagramas!<br>
                    '${texto1}' ↔ '${texto2}'<br>
                    Ordenados: '${texto1Ordenado}' = '${texto2Ordenado}'`
            };
        } else {
            return {
                sonAnagramas: false,
                mensaje: `✗ NO son anagramas.<br>
                    '${texto1}' → ordenado: '${texto1Ordenado}'<br>
                    '${texto2}' → ordenado: '${texto2Ordenado}'`
            };
        }
    }

    // --- 4. Validar al perder el foco (onblur) ---
    function validarAnagramaEnFoco() {
        let campo1 = document.getElementById("texto1");
        let campo2 = document.getElementById("texto2");
        let divResultado = document.getElementById("resultado");

        let valor1 = campo1.value;
        let valor2 = campo2.value;

        // Solo validamos si ambos campos tienen contenido
        if (valor1.trim() === "" || valor2.trim() === "") {
            divResultado.innerHTML = "";
            campo1.style.border = "";
            campo2.style.border = "";
            return;
        }

        // Llamamos a la función de validación
        let resultado = sonAnagramas(valor1, valor2);

        // Mostramos el resultado
        divResultado.innerHTML = resultado.mensaje;

        if (resultado.sonAnagramas) {
```

```
                // Son anagramas: color verde y borde verde
                divResultado.style.color = "green";
                campo1.style.border = "2px solid green";
                campo2.style.border = "2px solid green";
            } else {
                // No son anagramas: color rojo y borde rojo
                divResultado.style.color = "red";
                campo1.style.border = "2px solid red";
                campo2.style.border = "2px solid red";
            }
        }

        // --- 5. Validar al enviar el formulario (onsubmit) ---
        function validarFormulario() {
            let campo1 = document.getElementById("texto1");
            let campo2 = document.getElementById("texto2");
            let divResultado = document.getElementById("resultado");

            let valor1 = campo1.value;
            let valor2 = campo2.value;

            // Llamamos a la función de validación
            let resultado = sonAnagramas(valor1, valor2);

            if (resultado.sonAnagramas) {
                // Son anagramas: permitimos el envío
                alert("✓ Formulario válido. Son anagramas:\n'" + valor1 + "' y '" + valor2 +
"'");
                divResultado.innerHTML = "¡Formulario enviado correctamente! ✓";
                divResultado.style.color = "green";
                campo1.style.border = "2px solid green";
                campo2.style.border = "2px solid green";

                // return true; permitiría el envío real
                // Por ahora lo cancelamos para que no recargue la página
                return false;
            } else {
                // No son anagramas: cancelamos el envío
                alert("✗ Error: Los textos NO son anagramas.\nEl formulario no se puede en-
viar.");
                divResultado.innerHTML = resultado.mensaje;
                divResultado.style.color = "red";
                campo1.style.border = "2px solid red";
                campo2.style.border = "2px solid red";
                return false; // Cancelamos el envío
            }
        }

        // --- 6. Mensaje inicial ---
```

```
        document.getElementById("resultado").innerHTML = "Introduce dos textos para comprobar si
son anagramas...";
        document.getElementById("resultado").style.color = "gray";
    </script>

</body>
</html>
```

4. Realiza un formulario con dos elementos “select”. Al cambiar el primero, se actualizará el segundo. Al enviar el formulario, se comprobará que ambos han sido marcados. Cuando no tengan ninguna selección previa, los “select” mostrarán “Select no seleccionado”. Los valores del primer “select” serán “Huesca”, “Zaragoza”, “Teruel”. Por defecto no habrá ninguno seleccionado. Los valores del segundo “select” son: • Para Huesca: “Huesca Capital”, “Jaca”, “Barbastro”. • Para Zaragoza: “Zaragoza Capital”, “Calatayud”, “Ejea de los caballeros”. • Para Teruel: “Teruel Capital”, “Alcañiz”, “Calamocha”. (Aquí saldrá por defecto seleccionado “Alcañiz”).


```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Formulario con Select Dependientes</title>
</head>
<body>

    <h2>Formulario de Provincias y Ciudades de Aragón</h2>
    <p>Selecciona una provincia y luego una ciudad. Ambos campos son obligatorios.</p>

    <!-- Formulario -->
    <form id="formularioProvincias" onsubmit="return validarFormulario()">

        <label for="selectProvincia">Provincia:</label>
        <select id="selectProvincia" name="provincia" onchange="actualizarCiudades()">
            <option value="">Select no seleccionado</option>
            <option value="Huesca">Huesca</option>
            <option value="Zaragoza">Zaragoza</option>
            <option value="Teruel">Teruel</option>
        </select>
```

```
        <br><br>

        <label for="selectCiudad">Ciudad:</label>
        <select id="selectCiudad" name="ciudad">
            <option value="">Select no seleccionado</option>
        </select>

        <br><br>

        <button type="submit">Enviar Formulario</button>

    </form>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em; font-family: monos-
pace;"></div>

    <script>
        // --- 1. Definimos las ciudades para cada provincia ---
        let ciudadesPorProvincia = {
            "Huesca": ["Huesca Capital", "Jaca", "Barbastro"],
            "Zaragoza": ["Zaragoza Capital", "Calatayud", "Ejea de los caballeros"],
            "Teruel": ["Teruel Capital", "Alcañiz", "Calamocha"]
        };

        // --- 2. Función para actualizar el segundo select según la provincia seleccionada ---
        function actualizarCiudades() {
            // Obtenemos los elementos select
            let selectProvincia = document.getElementById("selectProvincia");
            let selectCiudad = document.getElementById("selectCiudad");

            // Obtenemos el valor seleccionado en el primer select
            let provinciaSeleccionada = selectProvincia.value;

            // Limpiamos el segundo select (borramos todas las opciones)
            selectCiudad.innerHTML = "";

            // Si no hay provincia seleccionada, mostramos solo la opción por defecto
            if (provinciaSeleccionada === "") {
                let opcionDefecto = document.createElement("option");
                opcionDefecto.value = "";
                opcionDefecto.textContent = "Select no seleccionado";
                selectCiudad.appendChild(opcionDefecto);

                // Reseteamos estilos
                selectCiudad.style.border = "";
                selectProvincia.style.border = "";
                document.getElementById("resultado").innerHTML = "";
                return;
            }
```

```
            // Obtenemos el array de ciudades correspondiente a la provincia
            let ciudades = ciudadesPorProvincia[provinciaSeleccionada];

            // Añadimos cada ciudad como una opción en el segundo select
            for (let i = 0; i < ciudades.length; i++) {
                let opcion = document.createElement("option");
                opcion.value = ciudades[i];
                opcion.textContent = ciudades[i];

                // CASO ESPECIAL: Para Teruel, "Alcañiz" debe estar seleccionado por defecto
                if (provinciaSeleccionada === "Teruel" && ciudades[i] === "Alcañiz") {
                    opcion.selected = true;
                }

                selectCiudad.appendChild(opcion);
            }

            // Mostramos mensaje informativo
            let divResultado = document.getElementById("resultado");
            divResultado.innerHTML = `Provincia seleccionada: ${provinciaSeleccionada}. Ahora
selecciona una ciudad.`;
            divResultado.style.color = "blue";

            // Reseteamos bordes
            selectProvincia.style.border = "";
            selectCiudad.style.border = "";
        }

        // --- 3. Función para validar el formulario antes de enviarlo ---
        function validarFormulario() {
            let selectProvincia = document.getElementById("selectProvincia");
            let selectCiudad = document.getElementById("selectCiudad");
            let divResultado = document.getElementById("resultado");

            let provinciaSeleccionada = selectProvincia.value;
            let ciudadSeleccionada = selectCiudad.value;

            // --- Verificamos que la provincia haya sido seleccionada ---
            if (provinciaSeleccionada === "") {
                alert("Error: Debes seleccionar una provincia.");
                divResultado.innerHTML = "Error: No has seleccionado ninguna provincia.";
                divResultado.style.color = "red";
                selectProvincia.style.border = "2px solid red";
                return false; // Cancelamos el envío
            }

            // --- Verificamos que la ciudad haya sido seleccionada ---
            if (ciudadSeleccionada === "") {
                alert("Error: Debes seleccionar una ciudad.");
                divResultado.innerHTML = "Error: No has seleccionado ninguna ciudad.";
```

```
                divResultado.style.color = "red";
                selectCiudad.style.border = "2px solid red";
                return false; // Cancelamos el envío
            }

            // --- Si ambos están seleccionados, permitimos el envío ---
            alert(`✓ Formulario válido:\nProvincia: ${provinciaSeleccionada}\nCiudad: ${ciu-
dadSeleccionada}`);

            divResultado.innerHTML = `✓ Formulario enviado correctamente:<br>
                                      Provincia: <strong>${provinciaSeleccionada}</strong><br>
                                      Ciudad: <strong>${ciudadSeleccionada}</strong>`;
            divResultado.style.color = "green";

            selectProvincia.style.border = "2px solid green";
            selectCiudad.style.border = "2px solid green";

            // return true; permitiría el envío real del formulario
            // Por ahora lo cancelamos para que no recargue la página
            return false;
        }

        // --- 4. Mensaje inicial ---
        document.getElementById("resultado").innerHTML = "Selecciona una provincia para comen-
zar...";
        document.getElementById("resultado").style.color = "gray";
    </script>

</body>
</html>
```

```html
<body>

    <h2>Formulario de Provincias y Ciudades de Aragón</h2>
    <p>Selecciona una provincia y luego una ciudad. Ambos campos son obligatorios.</p>

    <!-- Formulario -->
    <form id="formularioProvincias" onsubmit="return validarFormulario()">

        <label for="selectProvincia">Provincia:</label>
        <select id="selectProvincia" name="provincia" onchange="actualizarCiudades()">
            <option value="">Select no seleccionado</option>
            <option value="Huesca">Huesca</option>
            <option value="Zaragoza">Zaragoza</option>
            <option value="Teruel">Teruel</option>
        </select>

        <br><br>

        <label for="selectCiudad">Ciudad:</label>
        <select id="selectCiudad" name="ciudad">
            <option value="">Select no seleccionado</option>
        </select>

        <br><br>

        <button type="submit">Enviar Formulario</button>

    </form>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em; font-family: monospace;"></div>

    <script>
        // --- 1. Definimos las ciudades para cada provincia ---
        let ciudadesPorProvincia = {
            "Huesca": ["Huesca Capital", "Jaca", "Barbastro"],
            "Zaragoza": ["Zaragoza Capital", "Calatayud", "Ejea de los caballeros"],
            "Teruel": ["Teruel Capital", "Alcañiz", "Calamocha"]
        };

        // --- 2. Función para actualizar el segundo select según la provincia seleccionada ---
        function actualizarCiudades() {
            // Obtenemos los elementos select
            let selectProvincia = document.getElementById("selectProvincia");
            let selectCiudad = document.getElementById("selectCiudad");

            // Obtenemos el valor seleccionado en el primer select
            let provinciaSeleccionada = selectProvincia.value;

            // Limpiamos el segundo select (borramos todas las opciones)
            selectCiudad.innerHTML = "";

            // Si no hay provincia seleccionada, mostramos solo la opción por defecto
            if (provinciaSeleccionada === "") {
                let opcionDefecto = document.createElement("option");
                opcionDefecto.value = "";
                opcionDefecto.textContent = "Select no seleccionado";
                selectCiudad.appendChild(opcionDefecto);

                // Reseteamos estilos
                selectCiudad.style.border = "";
                selectProvincia.style.border = "";
                document.getElementById("resultado").innerHTML = "";
                return;
            }

            // Obtenemos el array de ciudades correspondiente a la provincia
            let ciudades = ciudadesPorProvincia[provinciaSeleccionada];

            // Añadimos cada ciudad como una opción en el segundo select
            for (let i = 0; i < ciudades.length; i++) {
                let opcion = document.createElement("option");
                opcion.value = ciudades[i];
                opcion.textContent = ciudades[i];

                // CASO ESPECIAL: Para Teruel, "Alcañiz" debe estar seleccionado por defecto
                if (provinciaSeleccionada === "Teruel" && ciudades[i] === "Alcañiz") {
                    opcion.selected = true;
                }

                selectCiudad.appendChild(opcion);
            }
```

```
        // Mostramos mensaje informativo
        let divResultado = document.getElementById("resultado");
        divResultado.innerHTML = `Provincia seleccionada: ${provinciaSeleccionada}. Ahora selecciona una ciudad.`;
        divResultado.style.color = "blue";

        // Reseteamos bordes
        selectProvincia.style.border = "";
        selectCiudad.style.border = "";
    }

    // --- 3. Función para validar el formulario antes de enviarlo ---
    function validarFormulario() {
        let selectProvincia = document.getElementById("selectProvincia");
        let selectCiudad = document.getElementById("selectCiudad");
        let divResultado = document.getElementById("resultado");

        let provinciaSeleccionada = selectProvincia.value;
        let ciudadSeleccionada = selectCiudad.value;

        // --- Verificamos que la provincia haya sido seleccionada ---
        if (provinciaSeleccionada === "") {
            alert("Error: Debes seleccionar una provincia.");
            divResultado.innerHTML = "Error: No has seleccionado ninguna provincia.";
            divResultado.style.color = "red";
            selectProvincia.style.border = "2px solid red";
            return false; // Cancelamos el envío
        }

        // --- Verificamos que la ciudad haya sido seleccionada ---
        if (ciudadSeleccionada === "") {
            alert("Error: Debes seleccionar una ciudad.");
            divResultado.innerHTML = "Error: No has seleccionado ninguna ciudad.";
            divResultado.style.color = "red";
            selectCiudad.style.border = "2px solid red";
            return false; // Cancelamos el envío
        }

        // --- Si ambos están seleccionados, permitimos el envío ---
        alert(`✓ Formulario válido:\nProvincia: ${provinciaSeleccionada}\nCiudad: ${ciudadSeleccionada}`);

        divResultado.innerHTML = `✓ Formulario enviado correctamente:<br>
                                  Provincia: <strong>${provinciaSeleccionada}</strong><br>
                                  Ciudad: <strong>${ciudadSeleccionada}</strong>`;
        divResultado.style.color = "green";

        selectProvincia.style.border = "2px solid green";
        selectCiudad.style.border = "2px solid green";

        // return true; permitiría el envío real del formulario
        // Por ahora lo cancelamos para que no recargue la página
        return false;
    }

    // --- 4. Mensaje inicial ---
    document.getElementById("resultado").innerHTML = "Selecciona una provincia para comenzar...";
    document.getElementById("resultado").style.color = "gray";
</script>

</body>
```

5. Crea un formulario con un textarea donde limites el número de caracteres a 200. Debes gestionarlo de tal manera, que cuando se alcance el límite, se acepten las pulsaciones de las teclas Suprimir, Borrado y flechas de teclado. • Puedes hacer un script adicional para comprobar el código de las teclas que debes permitir en caso de llegar al tope de caracteres. Desarrollo WEB entorno cliente – Unidad 5 2º DAM

Formulario con Límite de Caracteres

Escribe en el área de texto. El límite es de **200 caracteres**.

Al alcanzar el límite, solo se permiten: **Suprimir (Delete)**, **Borrado (Backspace)** y **flechas de teclado**.

Mensaje (máximo 200 caracteres):

Escribe aquí tu mensaje...

Caracteres: 0 / 200

Enviar Formulario

Escribe tu mensaje. El límite es de 200 caracteres.

Comprobador de Códigos de Teclas

Pulsa cualquier tecla en este campo para ver su código:

Pulsa una tecla aquí...

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Textarea con Límite de Caracteres</title>
</head>
<body>

    <h2>Formulario con Límite de Caracteres</h2>
    <p>Escribe en el área de texto. El límite es de <strong>200 caracteres</strong>.</p>
    <p>Al alcanzar el límite, solo se permiten: <strong>Suprimir (Delete)</strong>, <strong>Borrado (Backspace)</strong> y <strong>flechas de teclado</strong>.</p>

    <!-- Formulario -->
    <form id="formularioTexto" onsubmit="return enviarFormulario()">

        <label for="areaTexto">Mensaje (máximo 200 caracteres):</label>
        <br>
        <textarea id="areaTexto"
                  name="mensaje"
                  rows="8"
                  cols="50"
                  placeholder="Escribe aquí tu mensaje..."
                  onkeydown="controlarLimite(event)"
                  onkeyup="actualizarContador()"></textarea>

        <br>
```

```
        <!-- Contador de caracteres -->
        <div id="contador" style="font-size: 1.1em; font-family: monospace; margin-top: 5px;">
            Caracteres: <span id="numCaracteres">0</span> / 200
        </div>

        <br>

        <button type="submit">Enviar Formulario</button>

    </form>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em; font-family: monos-
pace;"></div>

    <!-- Script adicional para comprobar códigos de teclas -->
    <div style="margin-top: 30px; padding: 15px; background-color: #f0f0f0; border-left: 4px so-
lid #666;">
        <h3>Comprobador de Códigos de Teclas</h3>
        <p>Pulsa cualquier tecla en este campo para ver su código:</p>
        <input type="text"
               id="comprobadorTeclas"
               placeholder="Pulsa una tecla aquí..."
               onkeydown="mostrarCodigoTecla(event)"
               style="width: 100%; padding: 10px; font-size: 1em;">
        <div id="infoTecla" style="margin-top: 10px; font-family: monospace; color:
#333;"></div>
    </div>

    <script>
        // --- CONSTANTES ---
        const LIMITE_CARACTERES = 200;

        // Códigos de teclas permitidas cuando se alcanza el límite
        const TECLAS_PERMITIDAS = {
            8: "Backspace (Borrado)",
            46: "Delete (Suprimir)",
            37: "ArrowLeft (Flecha Izquierda)",
            38: "ArrowUp (Flecha Arriba)",
            39: "ArrowRight (Flecha Derecha)",
            40: "ArrowDown (Flecha Abajo)",
            36: "Home (Inicio)",
            35: "End (Fin)"
        };

        // --- 1. Función principal para controlar el límite de caracteres ---
        function controlarLimite(evento) {
            let textarea = document.getElementById("areaTexto");
            let longitudActual = textarea.value.length;
```

```
            // Obtenemos el código de la tecla pulsada
            let codigoTecla = evento.keyCode || evento.which;

            // --- Si NO hemos alcanzado el límite, permitimos cualquier tecla ---
            if (longitudActual < LIMITE_CARACTERES) {
                return true; // Permitir la tecla
            }

            // --- Si hemos alcanzado el límite ---
            // Verificamos si la tecla pulsada está en la lista de permitidas
            if (TECLAS_PERMITIDAS.hasOwnProperty(codigoTecla)) {
                // Es una tecla permitida (Backspace, Delete, flechas)
                return true; // Permitir la tecla
            }

            // Permitir también Ctrl+A, Ctrl+C, Ctrl+V, Ctrl+X (para copiar/pegar/seleccionar)
            if (evento.ctrlKey || evento.metaKey) {
                return true; // Permitir combinaciones con Ctrl/Cmd
            }

            // Si llegamos aquí, la tecla NO está permitida
            evento.preventDefault(); // Cancelamos la acción de la tecla

            // Mostramos mensaje de advertencia
            let divResultado = document.getElementById("resultado");
            divResultado.innerHTML = "⚠ Límite de caracteres alcanzado (200/200). Solo puedes
borrar o usar flechas.";
            divResultado.style.color = "red";

            return false; // Bloqueamos la tecla
        }

        // --- 2. Función para actualizar el contador de caracteres ---
        function actualizarContador() {
            let textarea = document.getElementById("areaTexto");
            let longitudActual = textarea.value.length;
            let spanContador = document.getElementById("numCaracteres");
            let divContador = document.getElementById("contador");
            let divResultado = document.getElementById("resultado");

            // Actualizamos el número en el contador
            spanContador.textContent = longitudActual;

            // Cambiamos colores según el porcentaje de uso
            if (longitudActual >= LIMITE_CARACTERES) {
                // Límite alcanzado: rojo
                divContador.style.color = "red";
                divContador.style.fontWeight = "bold";
                divResultado.innerHTML = "⚠ Límite alcanzado. Solo puedes borrar texto.";
                divResultado.style.color = "red";
```

```
            } else if (longitudActual >= LIMITE_CARACTERES * 0.9) {
                // 90% o más: naranja (advertencia)
                divContador.style.color = "orange";
                divContador.style.fontWeight = "bold";
                divResultado.innerHTML = `Quedan ${LIMITE_CARACTERES - longitudActual} caracte-
res.`;
                divResultado.style.color = "orange";
            } else if (longitudActual >= LIMITE_CARACTERES * 0.7) {
                // 70% o más: azul (info)
                divContador.style.color = "blue";
                divContador.style.fontWeight = "normal";
                divResultado.innerHTML = `Llevas ${longitudActual} caracteres de ${LIMITE_CARAC-
TERES}.`;
                divResultado.style.color = "blue";
            } else {
                // Menos del 70%: verde (OK)
                divContador.style.color = "green";
                divContador.style.fontWeight = "normal";
                divResultado.innerHTML = "";
            }
        }

        // --- 3. Función para enviar el formulario ---
        function enviarFormulario() {
            let textarea = document.getElementById("areaTexto");
            let divResultado = document.getElementById("resultado");
            let mensaje = textarea.value.trim();

            // Validamos que no esté vacío
            if (mensaje === "") {
                alert("Error: El mensaje no puede estar vacío.");
                divResultado.innerHTML = "Error: Debes escribir algo antes de enviar.";
                divResultado.style.color = "red";
                return false; // Cancelamos el envío
            }

            // Validamos que no exceda el límite (por seguridad)
            if (mensaje.length > LIMITE_CARACTERES) {
                alert(`Error: El mensaje excede el límite de ${LIMITE_CARACTERES} caracteres.`);
                divResultado.innerHTML = `Error: Mensaje demasiado largo (${mensaje.len-
gth}/${LIMITE_CARACTERES}).`;
                divResultado.style.color = "red";
                return false; // Cancelamos el envío
            }

            // Si todo es válido
            alert(`✓ Formulario enviado correctamente.\nMensaje (${mensaje.length} caracte-
res):\n"${mensaje}"`);
            divResultado.innerHTML = `✓ Formulario enviado con éxito.<br>Caracteres enviados:
${mensaje.length}`;
```

```
            divResultado.style.color = "green";

            // return true; permitiría el envío real
            return false; // Por ahora cancelamos para no recargar
        }

        // --- 4. SCRIPT ADICIONAL: Comprobador de códigos de teclas ---
        function mostrarCodigoTecla(evento) {
            let codigoTecla = evento.keyCode || evento.which;
            let nombreTecla = evento.key;
            let divInfo = document.getElementById("infoTecla");

            // Verificamos si la tecla está en la lista de permitidas
            let esPermitida = TECLAS_PERMITIDAS.hasOwnProperty(codigoTecla);

            // Construimos el mensaje
            let mensaje = `<strong>Tecla presionada:</strong> ${nombreTecla}<br>`;
            mensaje += `<strong>Código (keyCode):</strong> ${codigoTecla}<br>`;

            if (esPermitida) {
                mensaje += `<strong>Estado:</strong> <span style="color: green;">✓ PERMITIDA
cuando hay límite (${TECLAS_PERMITIDAS[codigoTecla]})</span>`;
            } else {
                mensaje += `<strong>Estado:</strong> <span style="color: red;">✗ BLOQUEADA
cuando hay límite</span>`;
            }

            divInfo.innerHTML = mensaje;

            // Prevenimos que se escriba en el campo de prueba
            evento.preventDefault();
        }

        // --- 5. Inicialización al cargar la página ---
        window.onload = function() {
            actualizarContador();
            document.getElementById("resultado").innerHTML = "Escribe tu mensaje. El límite es
de 200 caracteres.";
            document.getElementById("resultado").style.color = "gray";
        };
    </script>

</body>
</html>
```

6. Genera un formulario con 4 inputs de tipo radio. Cada uno de estos controles, debe llevar como valor un código en hexadecimal. Al presionar uno de estos controles, se debe cambiar, automáticamente, el color de fondo de la página.

Selector de Color de Fondo

Selecciona un color para cambiar el fondo de la página automáticamente.

Elige un color:

- ○ Azul Claro (#3498db)
- ○ Verde (#2ecc71)
- ○ Naranja (#e67e22)
- ○ Morado (#9b59b6)

Selecciona un color para cambiar el fondo de la página...

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Cambiar Color con Radio Buttons</title>
</head>
<body>

    <h2>Selector de Color de Fondo</h2>
    <p>Selecciona un color para cambiar el fondo de la página automáticamente.</p>

    <!-- Formulario -->
    <form id="formularioColores">

        <h3>Elige un color:</h3>

        <!-- Radio button 1: Azul claro -->
        <input type="radio"
               id="color1"
               name="colorFondo"
               value="#3498db"
               onchange="cambiarColorFondo()">
        <label for="color1">Azul Claro (#3498db)</label>
        <br><br>

        <!-- Radio button 2: Verde -->
        <input type="radio"
               id="color2"
               name="colorFondo"
```

```
               value="#2ecc71"
               onchange="cambiarColorFondo()">
        <label for="color2">Verde (#2ecc71)</label>
        <br><br>

        <!-- Radio button 3: Naranja -->
        <input type="radio"
               id="color3"
               name="colorFondo"
               value="#e67e22"
               onchange="cambiarColorFondo()">
        <label for="color3">Naranja (#e67e22)</label>
        <br><br>

        <!-- Radio button 4: Morado -->
        <input type="radio"
               id="color4"
               name="colorFondo"
               value="#9b59b6"
               onchange="cambiarColorFondo()">
        <label for="color4">Morado (#9b59b6)</label>
        <br><br>

    </form>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em; font-family: monos-
pace;"></div>

    <script>
        // --- 1. Función para cambiar el color de fondo ---
        function cambiarColorFondo() {
            // Obtenemos todos los radio buttons con el nombre "colorFondo"
            let radios = document.getElementsByName("colorFondo");

            let colorSeleccionado = null;
            let labelSeleccionado = null;

            // --- 2. Recorremos todos los radio buttons para encontrar cuál está seleccionado -
--
            for (let i = 0; i < radios.length; i++) {
                if (radios[i].checked) {
                    // Este radio button está seleccionado
                    colorSeleccionado = radios[i].value; // Obtenemos el código hexadecimal

                    // Obtenemos el texto del label asociado
                    let label = document.querySelector(`label[for="${radios[i].id}"]`);
                    labelSeleccionado = label.textContent;

                    break; // Salimos del bucle porque ya encontramos el seleccionado
                }
```

```
        }

        // --- 3. Si encontramos un color seleccionado, cambiamos el fondo ---
        if (colorSeleccionado) {
            // Aplicamos el color al fondo del body
            document.body.style.backgroundColor = colorSeleccionado;

            // Convertimos el color hexadecimal a RGB para mostrarlo
            let rgb = hexToRgb(colorSeleccionado);

            // Mostramos información del color seleccionado
            let divResultado = document.getElementById("resultado");
            divResultado.innerHTML = `Color de fondo cambiado:<br>
                                      <strong>${labelSeleccionado}</strong><br>
                                      RGB: (${rgb.r}, ${rgb.g}, ${rgb.b})`;
            divResultado.style.color = "#333";
            divResultado.style.backgroundColor = "rgba(255, 255, 255, 0.8)";
            divResultado.style.padding = "10px";
            divResultado.style.borderRadius = "5px";

            // Ajustamos el color del texto si el fondo es muy oscuro
            ajustarColorTexto(colorSeleccionado);
        }
    }

    // --- 4. Función auxiliar para convertir hexadecimal a RGB ---
    function hexToRgb(hex) {
        // Quitamos el # si existe
        hex = hex.replace('#', '');

        // Convertimos cada par de caracteres hexadecimales a decimal
        let r = parseInt(hex.substring(0, 2), 16); // Primeros 2 caracteres = Rojo
        let g = parseInt(hex.substring(2, 4), 16); // Caracteres 3-4 = Verde
        let b = parseInt(hex.substring(4, 6), 16); // Caracteres 5-6 = Azul

        return { r: r, g: g, b: b };
    }

    // --- 5. Función para ajustar el color del texto según la luminosidad del fondo ---
    function ajustarColorTexto(colorHex) {
        // Convertimos a RGB
        let rgb = hexToRgb(colorHex);

        // Calculamos la luminosidad usando la fórmula estándar
        // Fórmula: (0.299 * R + 0.587 * G + 0.114 * B)
        let luminosidad = (0.299 * rgb.r + 0.587 * rgb.g + 0.114 * rgb.b);

        // Si el fondo es oscuro (luminosidad < 128), texto blanco
        // Si el fondo es claro (luminosidad >= 128), texto negro
        if (luminosidad < 128) {
```

```
                document.body.style.color = "white";
            } else {
                document.body.style.color = "black";
            }
        }

        // --- 6. Mensaje inicial ---
        document.getElementById("resultado").innerHTML = "Selecciona un color para cambiar el
fondo de la página...";
        document.getElementById("resultado").style.color = "#666";
    </script>

</body>
</html>
```

7. Crea un pequeño formulario que permita al usuario seleccionar los ingredientes de una pizza. Debajo de los ingredientes, debe aparecer un textarea con las opciones seleccionadas. Los ingredientes de los que disponemos son los siguientes: tomate, atún, champiñón, pimiento rojo, mozzarella di buffala y cebolla. Cada vez que se selecciona un elemento, se debe apuntar en el textarea el ingrediente seleccionado. Debes gestionar también que cuando el usuario deselecciona un ingrediente, se borre de la lista.

🍕 Crea tu Pizza Personalizada

Selecciona los ingredientes que deseas en tu pizza. La lista se actualizará automáticamente.

Ingredientes disponibles:

- [ ] 🍅 Tomate
- [ ] 🐟 Atún
- [ ] 🍄 Champiñón
- [ ] 🌶️ Pimiento rojo
- [ ] 🧀 Mozzarella di buffala
- [ ] 🧅 Cebolla

Tu pizza llevará:

No has seleccionado ningún ingrediente aún.

¡Elige los ingredientes para tu pizza!

Ver Resumen del Pedido | Limpiar Selección

🍕 Pizza sin ingredientes. ¡Selecciona al menos uno!

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Selector de Ingredientes para Pizza</title>
</head>
<body>

    <h2>🍕 Crea tu Pizza Personalizada</h2>
    <p>Selecciona los ingredientes que deseas en tu pizza. La lista se actualizará automática-
mente.</p>

    <!-- Formulario -->
    <form id="formularioPizza">

        <h3>Ingredientes disponibles:</h3>

        <!-- Checkbox 1: Tomate -->
        <input type="checkbox"
```

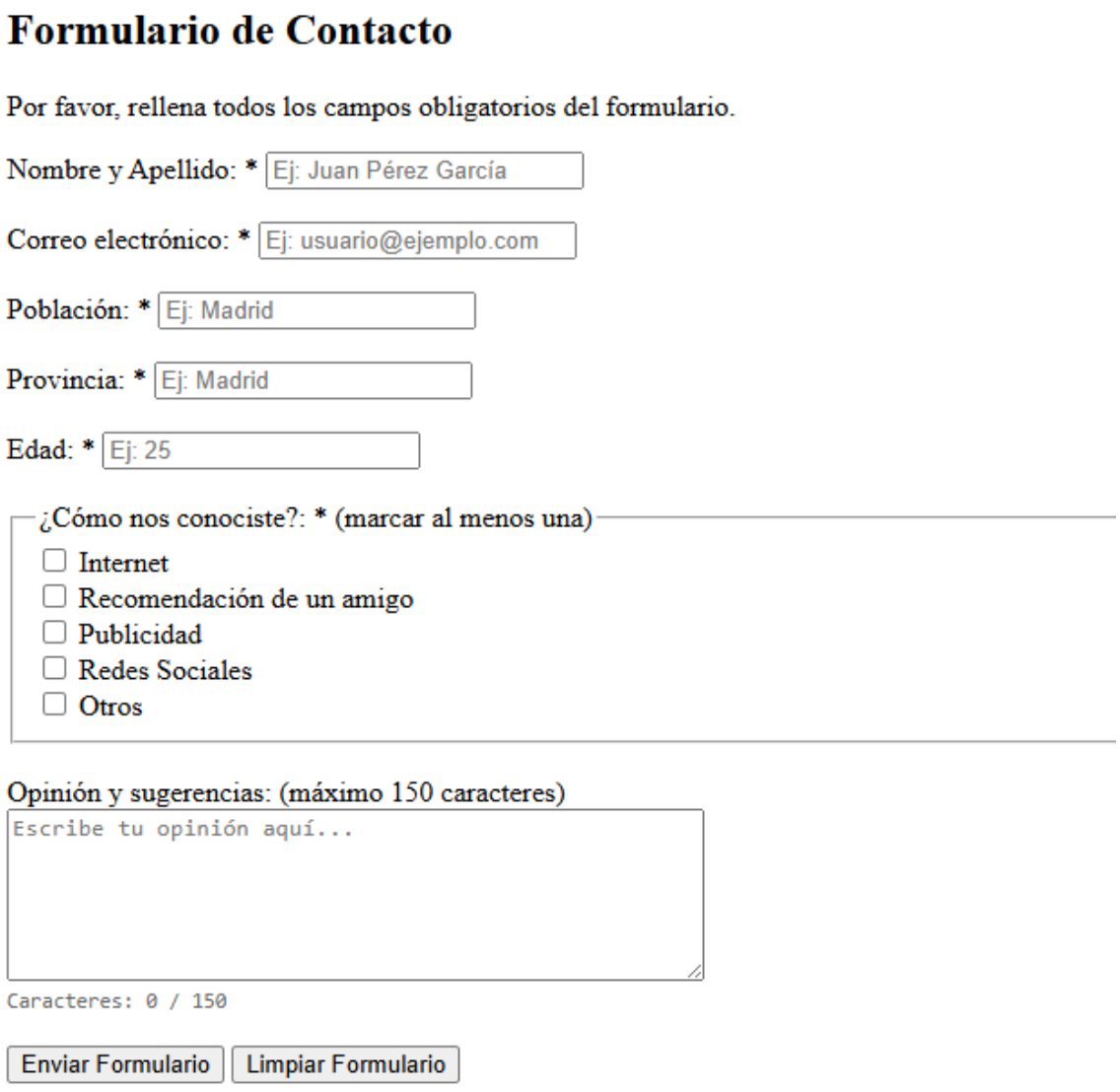
```
            id="tomate"
            name="ingredientes"
            value="Tomate"
            onchange="actualizarIngredientes()">
        <label for="tomate">🍅 Tomate</label>
        <br>

        <!-- Checkbox 2: Atún -->
        <input type="checkbox"
            id="atun"
            name="ingredientes"
            value="Atún"
            onchange="actualizarIngredientes()">
        <label for="atun">🐟 Atún</label>
        <br>

        <!-- Checkbox 3: Champiñón -->
        <input type="checkbox"
            id="champinon"
            name="ingredientes"
            value="Champiñón"
            onchange="actualizarIngredientes()">
        <label for="champinon">🍄 Champiñón</label>
        <br>

        <!-- Checkbox 4: Pimiento rojo -->
        <input type="checkbox"
            id="pimiento"
            name="ingredientes"
            value="Pimiento rojo"
            onchange="actualizarIngredientes()">
        <label for="pimiento">🌶️ Pimiento rojo</label>
        <br>

        <!-- Checkbox 5: Mozzarella di buffala -->
        <input type="checkbox"
            id="mozzarella"
            name="ingredientes"
            value="Mozzarella di buffala"
            onchange="actualizarIngredientes()">
        <label for="mozzarella">🧀 Mozzarella di buffala</label>
        <br>

        <!-- Checkbox 6: Cebolla -->
        <input type="checkbox"
            id="cebolla"
            name="ingredientes"
            value="Cebolla"
            onchange="actualizarIngredientes()">
        <label for="cebolla">🧅 Cebolla</label>
```

```
        <br><br>

        <hr>

        <!-- Área de texto para mostrar ingredientes seleccionados -->
        <h3>Tu pizza llevará:</h3>
        <textarea id="listaIngredientes"
                  rows="8"
                  cols="50"
                  readonly
                  placeholder="Selecciona ingredientes para ver tu pedido aquí..."></textarea>

        <br><br>

        <button type="button" onclick="mostrarResumen()">Ver Resumen del Pedido</button>
        <button type="button" onclick="limpiarSeleccion()">Limpiar Selección</button>

    </form>

    <div id="resultado" style="margin-top: 20px; font-size: 1.2em; font-family: monos-
pace;"></div>

    <script>
        // --- 1. Función principal para actualizar el textarea con los ingredientes ---
        function actualizarIngredientes() {
            // Obtenemos todos los checkboxes del formulario
            let checkboxes = document.getElementsByName("ingredientes");

            // Array para guardar los ingredientes seleccionados
            let ingredientesSeleccionados = [];

            // --- 2. Recorremos todos los checkboxes ---
            for (let i = 0; i < checkboxes.length; i++) {
                // Si el checkbox está marcado (checked = true)
                if (checkboxes[i].checked) {
                    // Añadimos su valor al array
                    ingredientesSeleccionados.push(checkboxes[i].value);
                }
            }

            // --- 3. Actualizamos el contenido del textarea ---
            let textarea = document.getElementById("listaIngredientes");

            if (ingredientesSeleccionados.length > 0) {
                // Si hay ingredientes seleccionados, los mostramos
                // Usamos join('\n') para poner cada ingrediente en una línea nueva
                textarea.value = "Ingredientes seleccionados:\n\n";

                // Añadimos cada ingrediente con un número
                for (let i = 0; i < ingredientesSeleccionados.length; i++) {
```

```
                    textarea.value += `${i + 1}. ${ingredientesSeleccionados[i]}\n`;
                }

                // Añadimos el total al final
                textarea.value += `\nTotal de ingredientes: ${ingredientesSeleccionados.len-
gth}`;

            } else {
                // Si no hay ingredientes seleccionados, mostramos mensaje
                textarea.value = "No has seleccionado ningún ingrediente aún.\n\n¡Elige los in-
gredientes para tu pizza!";
            }

            // --- 4. Actualizamos el mensaje informativo ---
            actualizarMensajeInfo(ingredientesSeleccionados.length);
        }

        // --- 5. Función para actualizar el mensaje informativo ---
        function actualizarMensajeInfo(cantidad) {
            let divResultado = document.getElementById("resultado");

            if (cantidad === 0) {
                divResultado.innerHTML = "🍕 Pizza sin ingredientes. ¡Selecciona al menos uno!";
                divResultado.style.color = "gray";
            } else if (cantidad === 1) {
                divResultado.innerHTML = "🍕 Has seleccionado 1 ingrediente.";
                divResultado.style.color = "blue";
            } else if (cantidad <= 3) {
                divResultado.innerHTML = `🍕 Has seleccionado ${cantidad} ingredientes. ¡Buen
equilibrio!`;
                divResultado.style.color = "green";
            } else if (cantidad <= 5) {
                divResultado.innerHTML = `🍕 Has seleccionado ${cantidad} ingredientes. ¡Pizza
completa!`;
                divResultado.style.color = "orange";
            } else {
                divResultado.innerHTML = `🍕 Has seleccionado ${cantidad} ingredientes. ¡Pizza
suprema!`;
                divResultado.style.color = "red";
            }
        }

        // --- 6. Función para mostrar un resumen del pedido ---
        function mostrarResumen() {
            let checkboxes = document.getElementsByName("ingredientes");
            let ingredientesSeleccionados = [];

            // Recogemos los ingredientes seleccionados
            for (let i = 0; i < checkboxes.length; i++) {
                if (checkboxes[i].checked) {
```

```
                    ingredientesSeleccionados.push(checkboxes[i].value);
                }
            }

            // Mostramos el resumen en un alert
            if (ingredientesSeleccionados.length > 0) {
                let mensaje = "🍕 RESUMEN DE TU PEDIDO 🍕\n\n";
                mensaje += "Tu pizza personalizada llevará:\n\n";

                for (let i = 0; i < ingredientesSeleccionados.length; i++) {
                    mensaje += `• ${ingredientesSeleccionados[i]}\n`;
                }

                mensaje += `\nTotal de ingredientes: ${ingredientesSeleccionados.length}`;
                mensaje += `\nPrecio estimado: ${8 + (ingredientesSeleccionados.length *
1.5)}€`;

                alert(mensaje);
            } else {
                alert("⚠️ No has seleccionado ningún ingrediente.\n\nPor favor, selecciona al
menos uno para crear tu pizza.");
            }
        }

        // --- 7. Función para limpiar toda la selección ---
        function limpiarSeleccion() {
            // Obtenemos todos los checkboxes
            let checkboxes = document.getElementsByName("ingredientes");

            // Desmarcamos todos los checkboxes
            for (let i = 0; i < checkboxes.length; i++) {
                checkboxes[i].checked = false;
            }

            // Actualizamos el textarea
            actualizarIngredientes();

            // Mostramos mensaje de confirmación
            alert("✓ Selección limpiada. Puedes empezar de nuevo.");
        }

        // --- 8. Inicialización al cargar la página ---
        window.onload = function() {
            // Llamamos a la función para inicializar el textarea
            actualizarIngredientes();
        };
    </script>

</body>
</html>
```

8. Crea un formulario con la estructura y formato de la captura. Debes añadir la funcionalidad necesaria para que se compruebe lo siguiente antes de proceder a su envío:
• Nombre y apellido sólo texto y que esté relleno
• Correo electrónico: Formato a@b.dom y que esté relleno
• Población: Solo texto y que esté relleno.
• Provincia: Solo texto y que esté relleno.
• Edad: Solo número, que esté relleno y la edad esté comprendida entre 18 y 100 años.
• Cómo nos conociste (Checkbox): Que se marque por lo menos una opción.
• Opinión y sugerencias. Sólo texto. Debe tener una longitud máxima de 150 caracteres.




```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Formulario de Contacto Completo</title>
</head>
<body>
```

```
<h2>Formulario de Contacto</h2>
<p>Por favor, rellena todos los campos obligatorios del formulario.</p>

<!-- Formulario -->
<form id="formularioContacto" onsubmit="return validarFormularioCompleto()">

    <!-- Nombre y Apellido -->
    <label for="nombre">Nombre y Apellido: *</label>
    <input type="text"
           id="nombre"
           name="nombre"
           placeholder="Ej: Juan Pérez García"
           required>
    <br><br>

    <!-- Correo electrónico -->
    <label for="email">Correo electrónico: *</label>
    <input type="text"
           id="email"
           name="email"
           placeholder="Ej: usuario@ejemplo.com"
           required>
    <br><br>

    <!-- Población -->
    <label for="poblacion">Población: *</label>
    <input type="text"
           id="poblacion"
           name="poblacion"
           placeholder="Ej: Madrid"
           required>
    <br><br>

    <!-- Provincia -->
    <label for="provincia">Provincia: *</label>
    <input type="text"
           id="provincia"
           name="provincia"
           placeholder="Ej: Madrid"
           required>
    <br><br>

    <!-- Edad -->
    <label for="edad">Edad: *</label>
    <input type="text"
           id="edad"
           name="edad"
           placeholder="Ej: 25"
           required>
```

```
        <br><br>

        <!-- Cómo nos conociste (Checkboxes) -->
        <fieldset>
            <legend>¿Cómo nos conociste?: * (marcar al menos una)</legend>

            <input type="checkbox"
                   id="internet"
                   name="conociste"
                   value="Internet">
            <label for="internet">Internet</label>
            <br>

            <input type="checkbox"
                   id="amigo"
                   name="conociste"
                   value="Amigo">
            <label for="amigo">Recomendación de un amigo</label>
            <br>

            <input type="checkbox"
                   id="publicidad"
                   name="conociste"
                   value="Publicidad">
            <label for="publicidad">Publicidad</label>
            <br>

            <input type="checkbox"
                   id="redes"
                   name="conociste"
                   value="Redes Sociales">
            <label for="redes">Redes Sociales</label>
            <br>

            <input type="checkbox"
                   id="otros"
                   name="conociste"
                   value="Otros">
            <label for="otros">Otros</label>
        </fieldset>
        <br>

        <!-- Opinión y sugerencias -->
        <label for="opinion">Opinión y sugerencias: (máximo 150 caracteres)</label>
        <br>
        <textarea id="opinion"
                  name="opinion"
                  rows="6"
                  cols="50"
                  placeholder="Escribe tu opinión aquí..."
```

```
                onkeyup="actualizarContadorOpinion()"></textarea>
        <br>
        <div id="contadorOpinion" style="font-size: 0.9em; font-family: monospace; color:
#666;">
            Caracteres: <span id="numCaracteresOpinion">0</span> / 150
        </div>
        <br>

        <button type="submit">Enviar Formulario</button>
        <button type="reset" onclick="limpiarMensajes()">Limpiar Formulario</button>

    </form>

    <div id="resultado" style="margin-top: 20px; font-size: 1.1em; font-family: monos-
pace;"></div>

    <script>
        // --- 1. Función para validar que un campo solo contenga texto (letras y espacios) ---
        function soloTexto(texto) {
            // Expresión regular: solo letras (incluyendo acentos), espacios y guiones
            let regex = /^[a-zA-ZáéíóúÁÉÍÓÚñÑüÜ\s-]+$/;
            return regex.test(texto);
        }

        // --- 2. Función para validar formato de email ---
        function validarFormatoEmail(email) {
            // Expresión regular: texto@texto.dominio
            let regex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
            return regex.test(email);
        }

        // --- 3. Función para validar que sea solo número ---
        function soloNumero(texto) {
            // Expresión regular: solo dígitos
            let regex = /^\d+$/;
            return regex.test(texto);
        }

        // --- 4. Función para validar nombre y apellido ---
        function validarNombreApellido() {
            let campo = document.getElementById("nombre");
            let valor = campo.value.trim();

            // Verificar que no esté vacío
            if (valor === "") {
                return {
                    valido: false,
                    mensaje: "El campo 'Nombre y Apellido' es obligatorio."
                };
            }
```

```
            // Verificar que solo contenga texto
            if (!soloTexto(valor)) {
                return {
                    valido: false,
                    mensaje: "El campo 'Nombre y Apellido' solo puede contener letras y espa-
cios."
                };
            }

            return { valido: true, mensaje: "" };
        }

        // --- 5. Función para validar correo electrónico ---
        function validarEmail() {
            let campo = document.getElementById("email");
            let valor = campo.value.trim();

            // Verificar que no esté vacío
            if (valor === "") {
                return {
                    valido: false,
                    mensaje: "El campo 'Correo electrónico' es obligatorio."
                };
            }

            // Verificar formato
            if (!validarFormatoEmail(valor)) {
                return {
                    valido: false,
                    mensaje: "El formato del correo electrónico debe ser: usuario@dominio.com"
                };
            }

            return { valido: true, mensaje: "" };
        }

        // --- 6. Función para validar población ---
        function validarPoblacion() {
            let campo = document.getElementById("poblacion");
            let valor = campo.value.trim();

            // Verificar que no esté vacío
            if (valor === "") {
                return {
                    valido: false,
                    mensaje: "El campo 'Población' es obligatorio."
                };
            }
```

```
        // Verificar que solo contenga texto
        if (!soloTexto(valor)) {
            return {
                valido: false,
                mensaje: "El campo 'Población' solo puede contener letras y espacios."
            };
        }

        return { valido: true, mensaje: "" };
    }

    // --- 7. Función para validar provincia ---
    function validarProvincia() {
        let campo = document.getElementById("provincia");
        let valor = campo.value.trim();

        // Verificar que no esté vacío
        if (valor === "") {
            return {
                valido: false,
                mensaje: "El campo 'Provincia' es obligatorio."
            };
        }

        // Verificar que solo contenga texto
        if (!soloTexto(valor)) {
            return {
                valido: false,
                mensaje: "El campo 'Provincia' solo puede contener letras y espacios."
            };
        }

        return { valido: true, mensaje: "" };
    }

    // --- 8. Función para validar edad ---
    function validarEdad() {
        let campo = document.getElementById("edad");
        let valor = campo.value.trim();

        // Verificar que no esté vacío
        if (valor === "") {
            return {
                valido: false,
                mensaje: "El campo 'Edad' es obligatorio."
            };
        }

        // Verificar que sea solo número
        if (!soloNumero(valor)) {
```

```
            return {
                valido: false,
                mensaje: "El campo 'Edad' solo puede contener números."
            };
        }

        // Convertir a número y verificar rango
        let edad = parseInt(valor);

        if (edad < 18 || edad > 100) {
            return {
                valido: false,
                mensaje: "La edad debe estar comprendida entre 18 y 100 años."
            };
        }

        return { valido: true, mensaje: "" };
    }

    // --- 9. Función para validar checkboxes (al menos uno marcado) ---
    function validarCheckboxes() {
        let checkboxes = document.getElementsByName("conociste");
        let alMenosUno = false;

        // Verificar si al menos un checkbox está marcado
        for (let i = 0; i < checkboxes.length; i++) {
            if (checkboxes[i].checked) {
                alMenosUno = true;
                break;
            }
        }

        if (!alMenosUno) {
            return {
                valido: false,
                mensaje: "Debes marcar al menos una opción en '¿Cómo nos conociste?'."
            };
        }

        return { valido: true, mensaje: "" };
    }

    // --- 10. Función para validar opinión (máximo 150 caracteres) ---
    function validarOpinion() {
        let campo = document.getElementById("opinion");
        let valor = campo.value;

        // Verificar longitud máxima
        if (valor.length > 150) {
            return {
```

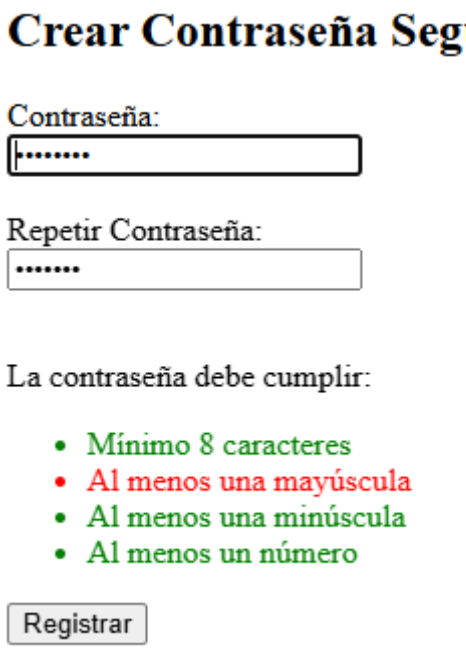
```
                    valido: false,
                    mensaje: `El campo 'Opinión y sugerencias' no puede superar los 150 caracte-
res. Actualmente tiene ${valor.length} caracteres.`
                };
            }

            // Si hay texto, verificar que solo contenga caracteres válidos (opcional)
            // En este caso permitimos cualquier carácter

            return { valido: true, mensaje: "" };
        }

        // --- 11. Función para actualizar contador de opinión ---
        function actualizarContadorOpinion() {
            let textarea = document.getElementById("opinion");
            let longitudActual = textarea.value.length;
            let spanContador = document.getElementById("numCaracteresOpinion");
            let divContador = document.getElementById("contadorOpinion");

            // Actualizamos el contador
            spanContador.textContent = longitudActual;

            // Cambiamos color según el porcentaje
            if (longitudActual > 150) {
                divContador.style.color = "red";
                divContador.style.fontWeight = "bold";
            } else if (longitudActual >= 135) {
                divContador.style.color = "orange";
                divContador.style.fontWeight = "bold";
            } else if (longitudActual >= 100) {
                divContador.style.color = "blue";
                divContador.style.fontWeight = "normal";
            } else {
                divContador.style.color = "#666";
                divContador.style.fontWeight = "normal";
            }
        }

        // --- 12. Función principal de validación del formulario ---
        function validarFormularioCompleto() {
            let divResultado = document.getElementById("resultado");
            let errores = [];

            // Validar cada campo
            let validacionNombre = validarNombreApellido();
            let validacionEmail = validarEmail();
            let validacionPoblacion = validarPoblacion();
            let validacionProvincia = validarProvincia();
            let validacionEdad = validarEdad();
            let validacionCheckboxes = validarCheckboxes();
```

```
        let validacionOpinion = validarOpinion();

        // Recopilar errores
        if (!validacionNombre.valido) errores.push(validacionNombre.mensaje);
        if (!validacionEmail.valido) errores.push(validacionEmail.mensaje);
        if (!validacionPoblacion.valido) errores.push(validacionPoblacion.mensaje);
        if (!validacionProvincia.valido) errores.push(validacionProvincia.mensaje);
        if (!validacionEdad.valido) errores.push(validacionEdad.mensaje);
        if (!validacionCheckboxes.valido) errores.push(validacionCheckboxes.mensaje);
        if (!validacionOpinion.valido) errores.push(validacionOpinion.mensaje);

        // Si hay errores, mostrarlos y cancelar envío
        if (errores.length > 0) {
            let mensajeError = "❌ Se encontraron los siguientes errores:\n\n";
            for (let i = 0; i < errores.length; i++) {
                mensajeError += `${i + 1}. ${errores[i]}\n`;
            }

            alert(mensajeError);

            // Mostrar errores en pantalla también
            divResultado.innerHTML = "<strong>Errores encontrados:</strong><br>";
            for (let i = 0; i < errores.length; i++) {
                divResultado.innerHTML += `${i + 1}. ${errores[i]}<br>`;
            }
            divResultado.style.color = "red";
            divResultado.style.backgroundColor = "#ffe6e6";
            divResultado.style.padding = "15px";
            divResultado.style.borderRadius = "5px";
            divResultado.style.border = "2px solid red";

            return false; // Cancelar envío
        }

        // Si todo es válido, mostrar resumen
        let nombre = document.getElementById("nombre").value.trim();
        let email = document.getElementById("email").value.trim();
        let poblacion = document.getElementById("poblacion").value.trim();
        let provincia = document.getElementById("provincia").value.trim();
        let edad = document.getElementById("edad").value.trim();
        let opinion = document.getElementById("opinion").value;

        // Obtener checkboxes marcados
        let checkboxes = document.getElementsByName("conociste");
        let opcionesSeleccionadas = [];
        for (let i = 0; i < checkboxes.length; i++) {
            if (checkboxes[i].checked) {
                opcionesSeleccionadas.push(checkboxes[i].value);
            }
        }
```

```
            let mensajeExito = `✓ FORMULARIO VÁLIDO\n\n`;
            mensajeExito += `Nombre: ${nombre}\n`;
            mensajeExito += `Email: ${email}\n`;
            mensajeExito += `Población: ${poblacion}\n`;
            mensajeExito += `Provincia: ${provincia}\n`;
            mensajeExito += `Edad: ${edad} años\n`;
            mensajeExito += `Nos conociste por: ${opcionesSeleccionadas.join(", ")}\n`;
            if (opinion) {
                mensajeExito += `Opinión: ${opinion}\n`;
            }
            mensajeExito += `\n¿Deseas enviar el formulario?`;

            alert(mensajeExito);

            // Mostrar mensaje de éxito en pantalla
            divResultado.innerHTML = `<strong>✓ Formulario validado correcta-
mente</strong><br><br>
                                      Nombre: ${nombre}<br>
                                      Email: ${email}<br>
                                      Población: ${poblacion}<br>
                                      Provincia: ${provincia}<br>
                                      Edad: ${edad} años<br>
                                      Nos conociste por: ${opcionesSeleccionadas.join(", ")}<br>
                                      ${opinion ? "Opinión: " + opinion : ""}`;
            divResultado.style.color = "green";
            divResultado.style.backgroundColor = "#e6ffe6";
            divResultado.style.padding = "15px";
            divResultado.style.borderRadius = "5px";
            divResultado.style.border = "2px solid green";

            // return true; permitiría el envío real
            return false; // Por ahora cancelamos para no recargar
        }

        // --- 13. Función para limpiar mensajes al resetear ---
        function limpiarMensajes() {
            document.getElementById("resultado").innerHTML = "";
            actualizarContadorOpinion();
        }

        // --- 14. Inicialización ---
        window.onload = function() {
            actualizarContadorOpinion();
        };
    </script>

</body>
</html>
```

Ejercicios Extra Formularios

Ejercicio 9: Validación de Contraseña Segura (VALIDACIONES)

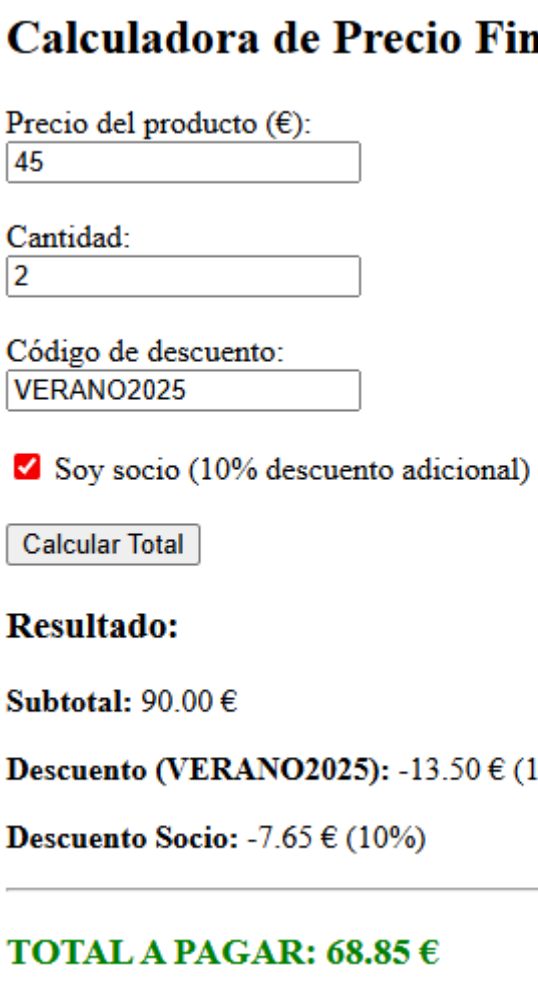



```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Ejercicio 9 - Validación de Contraseña</title>
</head>
<body>
    <h2>Crear Contraseña Segura</h2>
    <form id="formPassword">
        <label>Contraseña:</label><br>
        <input type="password" id="password" name="password"><br><br>

        <label>Repetir Contraseña:</label><br>
        <input type="password" id="password2" name="password2"><br><br>

        <div id="requisitos">
            <p>La contraseña debe cumplir:</p>
            <ul>
                <li id="req-longitud">Mínimo 8 caracteres</li>
                <li id="req-mayuscula">Al menos una mayúscula</li>
                <li id="req-minuscula">Al menos una minúscula</li>
                <li id="req-numero">Al menos un número</li>
            </ul>
        </div>

        <input type="submit" value="Registrar">
    </form>
```

```
    <script>
        // Obtener elementos del formulario
var formPassword = document.getElementById("formPassword");
var password = document.getElementById("password");
var password2 = document.getElementById("password2");

// Validar requisitos de contraseña mientras se escribe
password.addEventListener("keyup", function() {
    validarRequisitos(password.value);
});

// Función para validar los requisitos de contraseña
function validarRequisitos(pass) {
    // Verificar longitud mínima
    if (pass.length >= 8) {
        document.getElementById("req-longitud").style.color = "green";
    } else {
        document.getElementById("req-longitud").style.color = "red";
    }

    // Verificar mayúscula
    if (/[A-Z]/.test(pass)) {
        document.getElementById("req-mayuscula").style.color = "green";
    } else {
        document.getElementById("req-mayuscula").style.color = "red";
    }

    // Verificar minúscula
    if (/[a-z]/.test(pass)) {
        document.getElementById("req-minuscula").style.color = "green";
    } else {
        document.getElementById("req-minuscula").style.color = "red";
    }

    // Verificar número
    if (/[0-9]/.test(pass)) {
        document.getElementById("req-numero").style.color = "green";
    } else {
        document.getElementById("req-numero").style.color = "red";
    }
}

// Validar al enviar el formulario
formPassword.addEventListener("submit", function(evento) {
    evento.preventDefault();

    var pass1 = password.value;
    var pass2 = password2.value;

    // Validar que cumple todos los requisitos
```

```
    if (pass1.length < 8 || !/[A-Z]/.test(pass1) || !/[a-z]/.test(pass1) || !/[0-
9]/.test(pass1)) {
        alert("La contraseña no cumple todos los requisitos");
        return;
    }

    // Validar que las contraseñas coinciden
    if (pass1 !== pass2) {
        alert("Las contraseñas no coinciden");
        return;
    }

    alert("Contraseña válida. Formulario enviado correctamente");
    formPassword.submit();
});
    </script>
</body>
</html>
```

Ejercicio 10: Calculadora de Precio con Descuentos



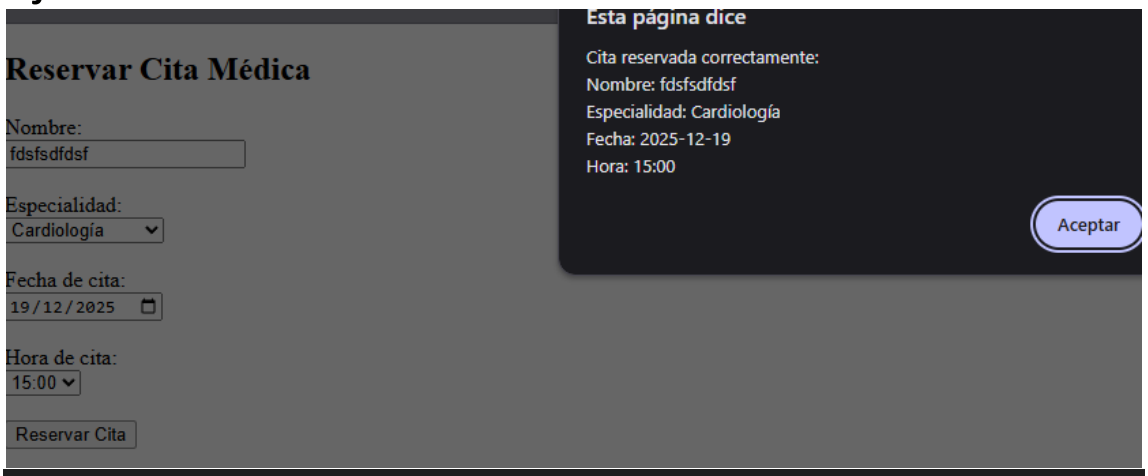

```
<!DOCTYPE html>
<html lang="es">
```

```
<head>
    <meta charset="UTF-8">
    <title>Ejercicio 10 - Calculadora de Precio</title>
</head>

<body>
    <h2>Calculadora de Precio Final</h2>
    <form id="formPrecio">
        <label>Precio del producto (€):</label><br>
        <input type="text" id="precio" name="precio"><br><br>

        <label>Cantidad:</label><br>
        <input type="text" id="cantidad" name="cantidad"><br><br>

        <label>Código de descuento:</label><br>
        <input type="text" id="codigo" name="codigo" placeholder="Opcional"><br><br>

        <label>
            <input type="checkbox" id="socio" name="socio">
            Soy socio (10% descuento adicional)
        </label><br><br>

        <input type="button" value="Calcular Total" id="btnCalcular">

        <h3>Resultado:</h3>
        <div id="resultado"></div>
    </form>

    <script>
        // Códigos de descuento válidos
        var codigosDescuento = {
            "VERANO2025": 15,
            "PRIMERACOMPRA": 20,
            "DESCUENTO10": 10
        };

        // Obtener elementos
        var btnCalcular = document.getElementById("btnCalcular");
        var precio = document.getElementById("precio");
        var cantidad = document.getElementById("cantidad");
        var codigo = document.getElementById("codigo");
        var socio = document.getElementById("socio");
        var resultado = document.getElementById("resultado");

        // Evento click del botón
        btnCalcular.addEventListener("click", calcularPrecio);

        function calcularPrecio() {
            // Validar que precio sea un número
            var precioValor = parseFloat(precio.value);
```

```
            if (isNaN(precioValor) || precioValor <= 0) {
                alert("Introduce un precio válido");
                return;
            }

            // Validar que cantidad sea un número entero
            var cantidadValor = parseInt(cantidad.value);
            if (isNaN(cantidadValor) || cantidadValor <= 0) {
                alert("Introduce una cantidad válida");
                return;
            }

            // Calcular subtotal
            var subtotal = precioValor * cantidadValor;
            var descuentoTotal = 0;
            var html = "";

            html += "<p><b>Subtotal:</b> " + subtotal.toFixed(2) + " €</p>";

            // Aplicar código de descuento si existe
            var codigoIngresado = codigo.value.trim().toUpperCase();
            if (codigoIngresado !== "") {
                if (codigosDescuento[codigoIngresado]) {
                    var descuentoCodigo = codigosDescuento[codigoIngresado];
                    var montoDescuento = subtotal * (descuentoCodigo / 100);
                    descuentoTotal += montoDescuento;
                    html += "<p><b>Descuento (" + codigoIngresado + "):</b> -" + montoDes-
cuento.toFixed(2) + " € (" + descuentoCodigo + "%)</p>";
                } else {
                    html += "<p style='color:red'>Código de descuento no válido</p>";
                }
            }

            // Aplicar descuento de socio
            if (socio.checked) {
                var descuentoSocio = (subtotal - descuentoTotal) * 0.10;
                descuentoTotal += descuentoSocio;
                html += "<p><b>Descuento Socio:</b> -" + descuentoSocio.toFixed(2) + " €
(10%)</p>";
            }

            // Calcular total final
            var total = subtotal - descuentoTotal;
            html += "<hr>";
            html += "<h3 style='color:green'>TOTAL A PAGAR: " + total.toFixed(2) + " €</h3>";

            resultado.innerHTML = html;
        }
    </script>
</body>
```

```
</html>
```

Ejercicio 11: Selector de Fecha de Cita



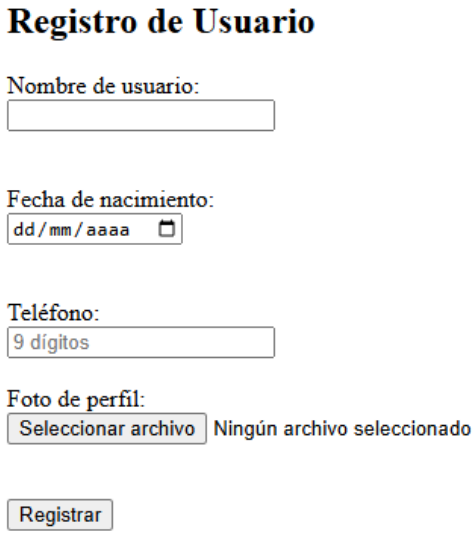

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Ejercicio 11 - Reserva de Cita</title>
</head>
<body>
    <h2>Reservar Cita Médica</h2>
    <form id="formCita">
        <label>Nombre:</label><br>
        <input type="text" id="nombre" name="nombre"><br><br>

        <label>Especialidad:</label><br>
        <select id="especialidad" name="especialidad">
            <option value="">-- Seleccionar --</option>
            <option value="traumatologia">Traumatología</option>
            <option value="cardiologia">Cardiología</option>
            <option value="dermatologia">Dermatología</option>
        </select><br><br>

        <label>Fecha de cita:</label><br>
        <input type="date" id="fecha" name="fecha"><br><br>

        <label>Hora de cita:</label><br>
        <select id="hora" name="hora">
            <option value="">-- Seleccionar fecha primero --</option>
        </select><br><br>

        <input type="submit" value="Reservar Cita">
    </form>

    <script>
        // Obtener elementos
var formCita = document.getElementById("formCita");
```

```
var nombre = document.getElementById("nombre");
var especialidad = document.getElementById("especialidad");
var fecha = document.getElementById("fecha");
var hora = document.getElementById("hora");

// Horarios disponibles por especialidad
var horarios = {
    "traumatologia": ["09:00", "10:00", "11:00", "12:00"],
    "cardiologia": ["09:30", "11:30", "15:00", "16:30"],
    "dermatologia": ["10:00", "11:30", "15:30", "17:00"]
};

// Establecer fecha mínima (hoy)
var hoy = new Date();
var dd = String(hoy.getDate()).padStart(2, '0');
var mm = String(hoy.getMonth() + 1).padStart(2, '0');
var yyyy = hoy.getFullYear();
fecha.min = yyyy + '-' + mm + '-' + dd;

// Actualizar horarios cuando cambia la especialidad
especialidad.addEventListener("change", function() {
    actualizarHorarios();
});

// También actualizar cuando cambia la fecha
fecha.addEventListener("change", function() {
    actualizarHorarios();
});

function actualizarHorarios() {
    var especialidadSeleccionada = especialidad.value;
    var fechaSeleccionada = fecha.value;

    // Limpiar opciones de hora
    hora.innerHTML = "";

    if (especialidadSeleccionada === "" || fechaSeleccionada === "") {
        hora.innerHTML = "<option value=''>-- Seleccionar especialidad y fecha --</option>";
        return;
    }

    // Agregar horarios disponibles
    var horariosDisponibles = horarios[especialidadSeleccionada];

    for (var i = 0; i < horariosDisponibles.length; i++) {
        var opcion = document.createElement("option");
        opcion.value = horariosDisponibles[i];
        opcion.textContent = horariosDisponibles[i];
        hora.appendChild(opcion);
    }
```

```
}

// Validar formulario al enviar
formCita.addEventListener("submit", function(evento) {
    evento.preventDefault();

    // Validar nombre
    if (nombre.value.trim() === "" || !/^[a-zA-ZáéíóúÁÉÍÓÚñÑ\s]+$/.test(nombre.value)) {
        alert("Introduce un nombre válido (solo letras)");
        return;
    }

    // Validar especialidad
    if (especialidad.value === "") {
        alert("Selecciona una especialidad");
        return;
    }

    // Validar fecha
    if (fecha.value === "") {
        alert("Selecciona una fecha");
        return;
    }

    // Validar que la fecha no sea en el pasado
    var fechaSeleccionada = new Date(fecha.value);
    var hoy = new Date();
    hoy.setHours(0, 0, 0, 0);

    if (fechaSeleccionada < hoy) {
        alert("No puedes seleccionar una fecha pasada");
        return;
    }

    // Validar hora
    if (hora.value === "") {
        alert("Selecciona una hora");
        return;
    }

    alert("Cita reservada correctamente:\n" +
          "Nombre: " + nombre.value + "\n" +
          "Especialidad: " + especialidad.options[especialidad.selectedIndex].text + "\n" +
          "Fecha: " + fecha.value + "\n" +
          "Hora: " + hora.value);

    formCita.submit();
});

    </script>
```

```
</body>
</html>
```

Ejercicio 12: Formulario de Registro con Foto




```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Ejercicio 12 - Registro con Foto</title>
</head>
<body>
    <h2>Registro de Usuario</h2>
    <form id="formRegistro">
        <label>Nombre de usuario:</label><br>
        <input type="text" id="username" name="username"><br>
        <small id="mensajeUsername"></small><br><br>

        <label>Fecha de nacimiento:</label><br>
        <input type="date" id="fechaNac" name="fechaNac"><br>
        <small id="mensajeEdad"></small><br><br>

        <label>Teléfono:</label><br>
        <input type="text" id="telefono" name="telefono" placeholder="9 dígitos"><br><br>

        <label>Foto de perfil:</label><br>
        <input type="file" id="foto" name="foto" accept="image/*"><br>
        <small id="mensajeFoto"></small><br><br>

        <input type="submit" value="Registrar">
    </form>

    <script>
        // Obtener elementos
var formRegistro = document.getElementById("formRegistro");
var username = document.getElementById("username");
```

```
var fechaNac = document.getElementById("fechaNac");
var telefono = document.getElementById("telefono");
var foto = document.getElementById("foto");

// Validar username al perder el foco
username.addEventListener("blur", function() {
    var mensajeUsername = document.getElementById("mensajeUsername");
    var valor = username.value.trim();

    // Username debe tener entre 4 y 15 caracteres alfanuméricos
    if (valor.length < 4 || valor.length > 15) {
        mensajeUsername.textContent = "El username debe tener entre 4 y 15 caracteres";
        mensajeUsername.style.color = "red";
    } else if (!/^[a-zA-Z0-9]+$/.test(valor)) {
        mensajeUsername.textContent = "Solo se permiten letras y números";
        mensajeUsername.style.color = "red";
    } else {
        mensajeUsername.textContent = "Username válido ✓";
        mensajeUsername.style.color = "green";
    }
});

// Validar edad al cambiar fecha
fechaNac.addEventListener("change", function() {
    var mensajeEdad = document.getElementById("mensajeEdad");
    var fechaSeleccionada = new Date(fechaNac.value);
    var hoy = new Date();

    // Calcular edad
    var edad = hoy.getFullYear() - fechaSeleccionada.getFullYear();
    var mes = hoy.getMonth() - fechaSeleccionada.getMonth();

    if (mes < 0 || (mes === 0 && hoy.getDate() < fechaSeleccionada.getDate())) {
        edad--;
    }

    if (edad < 18) {
        mensajeEdad.textContent = "Debes ser mayor de 18 años";
        mensajeEdad.style.color = "red";
    } else {
        mensajeEdad.textContent = "Edad: " + edad + " años ✓";
        mensajeEdad.style.color = "green";
    }
});

// Validar archivo de foto
foto.addEventListener("change", function() {
    var mensajeFoto = document.getElementById("mensajeFoto");
    var archivo = foto.files[0];
```

```
    if (!archivo) {
        mensajeFoto.textContent = "";
        return;
    }

    // Validar tamaño (máximo 2MB)
    var tamanoMB = archivo.size / (1024 * 1024);

    if (tamanoMB > 2) {
        mensajeFoto.textContent = "La imagen no puede superar 2MB";
        mensajeFoto.style.color = "red";
        foto.value = "";
        return;
    }

    // Validar tipo de archivo
    var tiposPermitidos = ["image/jpeg", "image/png", "image/jpg"];
    if (!tiposPermitidos.includes(archivo.type)) {
        mensajeFoto.textContent = "Solo se permiten archivos JPG o PNG";
        mensajeFoto.style.color = "red";
        foto.value = "";
        return;
    }

    mensajeFoto.textContent = "Imagen válida ✓";
    mensajeFoto.style.color = "green";
});

// Validar formulario completo
formRegistro.addEventListener("submit", function(evento) {
    evento.preventDefault();

    // Validar username
    if (username.value.trim().length < 4 || username.value.trim().length > 15) {
        alert("Username no válido");
        return;
    }

    // Validar fecha de nacimiento
    if (fechaNac.value === "") {
        alert("Selecciona tu fecha de nacimiento");
        return;
    }

    var fechaSeleccionada = new Date(fechaNac.value);
    var hoy = new Date();
    var edad = hoy.getFullYear() - fechaSeleccionada.getFullYear();

    if (edad < 18) {
        alert("Debes ser mayor de 18 años");
```

```
        return;
    }

    // Validar teléfono (9 dígitos)
    if (!/^[0-9]{9}$/.test(telefono.value)) {
        alert("El teléfono debe tener 9 dígitos");
        return;
    }

    // Validar que se haya subido una foto
    if (foto.files.length === 0) {
        alert("Debes subir una foto de perfil");
        return;
    }

    alert("Registro completado correctamente");
    formRegistro.submit();
});
    </script>
</body>
</html>
```

Ejercicio 13: Buscador con Filtros

Buscador de Productos

Buscar producto:
Escribe para buscar...

Categoría:
Todas

Precio máximo (€):
279 €

Solo productos disponibles

Buscar

Resultados (9):

Producto	Categoría	Precio	Estado
Camiseta Nike	ropa	35 €	Disponible
Lámpara LED	hogar	25 €	No disponible
Balón Adidas	deporte	20 €	Disponible
Teclado Logitech	electronica	50 €	Disponible
Pantalón Levi's	ropa	80 €	Disponible
Silla Oficina	hogar	120 €	Disponible
Raqueta Wilson	deporte	150 €	No disponible
Monitor Samsung	electronica	200 €	Disponible
Zapatillas Puma	ropa	65 €	Disponible

```
<!DOCTYPE html>
<html lang="es">

<head>
    <meta charset="UTF-8">
    <title>Ejercicio 13 - Buscador de Productos</title>
</head>

<body>
    <h2>Buscador de Productos</h2>
    <form id="formBuscar">
        <label>Buscar producto:</label><br>
        <input type="text" id="busqueda" name="busqueda" placeholder="Escribe para bus-
car..."><br><br>

        <label>Categoría:</label><br>
        <select id="categoria" name="categoria">
            <option value="">Todas</option>
```

```
            <option value="electronica">Electrónica</option>
            <option value="ropa">Ropa</option>
            <option value="hogar">Hogar</option>
            <option value="deporte">Deporte</option>
        </select><br><br>

        <label>Precio máximo (€):</label><br>
        <input type="range" id="precioMax" name="precioMax" min="0" max="500" value="500">
        <span id="valorPrecio">500</span> €<br><br>

        <label>
            <input type="checkbox" id="disponible" name="disponible" checked>
            Solo productos disponibles
        </label><br><br>

        <input type="button" value="Buscar" id="btnBuscar">
    </form>

    <div id="resultados"></div>

    <script>
        // Base de datos de productos (simulada)
        var productos = [
            { nombre: "Portátil HP", categoria: "electronica", precio: 450, disponible: true },
            { nombre: "Camiseta Nike", categoria: "ropa", precio: 35, disponible: true },
            { nombre: "Lámpara LED", categoria: "hogar", precio: 25, disponible: false },
            { nombre: "Balón Adidas", categoria: "deporte", precio: 20, disponible: true },
            { nombre: "Teclado Logitech", categoria: "electronica", precio: 50, disponible: true
},
            { nombre: "Pantalón Levi's", categoria: "ropa", precio: 80, disponible: true },
            { nombre: "Silla Oficina", categoria: "hogar", precio: 120, disponible: true },
            { nombre: "Raqueta Wilson", categoria: "deporte", precio: 150, disponible: false },
            { nombre: "Monitor Samsung", categoria: "electronica", precio: 200, disponible: true
},
            { nombre: "Zapatillas Puma", categoria: "ropa", precio: 65, disponible: true }
        ];

        // Obtener elementos
        var busqueda = document.getElementById("busqueda");
        var categoria = document.getElementById("categoria");
        var precioMax = document.getElementById("precioMax");
        var valorPrecio = document.getElementById("valorPrecio");
        var disponible = document.getElementById("disponible");
        var btnBuscar = document.getElementById("btnBuscar");
        var resultados = document.getElementById("resultados");

        // Actualizar valor del precio en tiempo real
        precioMax.addEventListener("input", function () {
            valorPrecio.textContent = precioMax.value;
        });
```

```
        // Buscar automáticamente al escribir
        busqueda.addEventListener("keyup", buscarProductos);
        categoria.addEventListener("change", buscarProductos);
        precioMax.addEventListener("input", buscarProductos);
        disponible.addEventListener("change", buscarProductos);
        btnBuscar.addEventListener("click", buscarProductos);

        function buscarProductos() {
            var textoBusqueda = busqueda.value.toLowerCase().trim();
            var categoriaSeleccionada = categoria.value;
            var precioMaximo = parseInt(precioMax.value);
            var soloDisponibles = disponible.checked;

            // Filtrar productos
            var productosFiltrados = productos.filter(function (producto) {
                // Filtro por texto de búsqueda
                var coincideTexto = producto.nombre.toLowerCase().includes(textoBusqueda);

                // Filtro por categoría
                var coincideCategoria = categoriaSeleccionada === "" || producto.categoria ===
categoriaSeleccionada;

                // Filtro por precio
                var coincidePrecio = producto.precio <= precioMaximo;

                // Filtro por disponibilidad
                var coincideDisponibilidad = !soloDisponibles || producto.disponible;

                return coincideTexto && coincideCategoria && coincidePrecio && coincideDisponi-
bilidad;
            });

            // Mostrar resultados
            mostrarResultados(productosFiltrados);
        }

        function mostrarResultados(productosFiltrados) {
            if (productosFiltrados.length === 0) {
                resultados.innerHTML = "<p>No se encontraron productos con esos criterios</p>";
                return;
            }

            var html = "<h3>Resultados (" + productosFiltrados.length + "):</h3>";
            html += "<table border='1'>";
            html += "<tr><th>Producto</th><th>Categoría</th><th>Precio</th><th>Es-
tado</th></tr>";

            for (var i = 0; i < productosFiltrados.length; i++) {
                var producto = productosFiltrados[i];
```

```
                var estado = producto.disponible ? "Disponible" : "No disponible";
                var color = producto.disponible ? "green" : "red";

                html += "<tr>";
                html += "<td>" + producto.nombre + "</td>";
                html += "<td>" + producto.categoria + "</td>";
                html += "<td>" + producto.precio + " €</td>";
                html += "<td style='color:" + color + "'>" + estado + "</td>";
                html += "</tr>";
            }

            html += "</table>";
            resultados.innerHTML = html;
        }

        // Mostrar todos los productos al cargar
        buscarProductos();
    </script>
</body>

</html>
```

UD7 – Comunicación asíncrona – Fetch APIs

A veces da problemas con la cache del navegador (en particular Chrome), y aunque modifiquemos los archivos, el navegador sigue usando la versión anterior del archivo.
1. Crea una página web que muestre una lista de personajes de Star Wars y permita ver más detalles al hacer clic en cada uno.
La dirección para descargar los datos es:
https://swapi.dev/api/people/
En la llamada nos muestra los 10 primeros personajes (no queremos al resto).
El json que devuelve tiene el siguiente formato

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Star Wars API - Personajes</title>
  <style>
    /* Sin estilos - solo HTML básico */
  </style>
</head>
<body>
  <h1>Personajes de Star Wars</h1>

  <!-- Contenedor donde se mostrarán las tarjetas de los personajes -->
  <div id="characters"></div>

  <!-- Contenedor donde se mostrarán los detalles del personaje seleccionado -->
```

```
<div id="details"></div>

<script>
  // URL de la API de Star Wars (devuelve los primeros 10 personajes)
  const apiURL = "https://swapi.dev/api/people/";

  // Obtener los personajes con fetch
  fetch(apiURL)
    .then(response => {
      // Verificar si la respuesta es correcta
      if (!response.ok) throw new Error("Error al cargar los personajes");
      // Convertir la respuesta a JSON
      return response.json();
    })
    .then(data => {
      // El array de personajes está en data.results
      const personajes = data.results;

      // Obtener el contenedor donde vamos a poner los personajes
      const container = document.getElementById("characters");

      // Recorrer cada personaje y crear su tarjeta
      personajes.forEach(personaje => {
        // Crear un div para cada personaje
        const card = document.createElement("div");

        // Crear un párrafo con el nombre del personaje
        const name = document.createElement("h3");
        name.textContent = personaje.name;

        // Añadir el nombre al div
        card.appendChild(name);

        // Añadir un evento de click para mostrar detalles
        card.addEventListener("click", () => showDetails(personaje));

        // Añadir la tarjeta al contenedor principal
        container.appendChild(card);
      });
    })
    .catch(error => console.error(error));

  // Función para mostrar detalles del personaje seleccionado
  function showDetails(character) {
    // Obtener el contenedor de detalles
    const details = document.getElementById("details");

    // Limpiar los detalles anteriores
    details.innerHTML = "";
```

```
      // Crear y añadir el título con el nombre
      const title = document.createElement("h2");
      title.textContent = character.name;
      details.appendChild(title);

      // Crear y añadir la altura
      const p1 = document.createElement("p");
      p1.innerHTML = "<strong>Altura:</strong> " + character.height + " cm";
      details.appendChild(p1);

      // Crear y añadir el peso
      const p2 = document.createElement("p");
      p2.innerHTML = "<strong>Peso:</strong> " + character.mass + " kg";
      details.appendChild(p2);

      // Crear y añadir el color de cabello
      const p3 = document.createElement("p");
      p3.innerHTML = "<strong>Color de cabello:</strong> " + character.hair_color;
      details.appendChild(p3);

      // Crear y añadir el color de piel
      const p4 = document.createElement("p");
      p4.innerHTML = "<strong>Color de piel:</strong> " + character.skin_color;
      details.appendChild(p4);

      // Crear y añadir el color de ojos
      const p5 = document.createElement("p");
      p5.innerHTML = "<strong>Color de ojos:</strong> " + character.eye_color;
      details.appendChild(p5);

      // Crear y añadir el año de nacimiento
      const p6 = document.createElement("p");
      p6.innerHTML = "<strong>Año de nacimiento:</strong> " + character.birth_year;
      details.appendChild(p6);

      // Crear y añadir el género
      const p7 = document.createElement("p");
      p7.innerHTML = "<strong>Género:</strong> " + character.gender;
      details.appendChild(p7);
    }
  </script>
</body>
</html>
```

2. Crea una página web que muestre una lista de personajes de Rick and Morty y permita ver más detalles al hacer clic en cada uno. La dirección para descargar los datos es: https://rickandmortyapi.com/api/character En la llamada nos muestra los 20 personajes. El json que devuelve tiene el siguiente format
No es necesario implementar los estilos

```html
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Rick and Morty API - Personajes</title>
  <style>
    /* Sin estilos - solo HTML básico */
  </style>
</head>
<body>
  <h1>Personajes de Rick and Morty</h1>

  <!-- Contenedor donde se mostrarán las tarjetas de los personajes -->
  <div id="characters"></div>

  <!-- Contenedor donde se mostrarán los detalles del personaje seleccionado -->
  <div id="details"></div>

  <script>
    // URL de la API de Rick and Morty (devuelve los primeros 20 personajes)
    const apiURL = "https://rickandmortyapi.com/api/character";

    // Obtener los personajes con fetch
    fetch(apiURL)
      .then(response => {
        // Verificar si la respuesta es correcta
        if (!response.ok) throw new Error("Error al cargar los personajes");
        // Convertir la respuesta a JSON
        return response.json();
      })
      .then(data => {
        // El array de personajes está en data.results
        const personajes = data.results;

        // Obtener el contenedor donde vamos a poner los personajes
        const container = document.getElementById("characters");

        // Recorrer cada personaje y crear su tarjeta
        personajes.forEach(personaje => {
          // Crear un div para cada personaje
          const card = document.createElement("div");

          // Crear una imagen del personaje
          const img = document.createElement("img");
          img.src = personaje.image;
          img.alt = personaje.name;

          // Crear un párrafo con el nombre del personaje
          const name = document.createElement("h3");
```

```
        name.textContent = personaje.name;

        // Añadir la imagen y el nombre al div
        card.appendChild(img);
        card.appendChild(name);

        // Añadir un evento de click para mostrar detalles
        card.addEventListener("click", () => showDetails(personaje));

        // Añadir la tarjeta al contenedor principal
        container.appendChild(card);
      });
    })
    .catch(error => console.error(error));

// Función para mostrar detalles del personaje seleccionado
function showDetails(character) {
    // Obtener el contenedor de detalles
    const details = document.getElementById("details");

    // Limpiar los detalles anteriores
    details.innerHTML = "";

    // Crear y añadir el título con el nombre
    const title = document.createElement("h2");
    title.textContent = character.name;
    details.appendChild(title);

    // Crear y añadir la imagen grande
    const img = document.createElement("img");
    img.src = character.image;
    img.alt = character.name;
    details.appendChild(img);

    // Crear y añadir el ID
    const p1 = document.createElement("p");
    p1.innerHTML = "<strong>ID:</strong> " + character.id;
    details.appendChild(p1);

    // Crear y añadir el estado (Alive, Dead, unknown)
    const p2 = document.createElement("p");
    p2.innerHTML = "<strong>Estado:</strong> " + character.status;
    details.appendChild(p2);

    // Crear y añadir la especie (Human, Alien, etc.)
    const p3 = document.createElement("p");
    p3.innerHTML = "<strong>Especie:</strong> " + character.species;
    details.appendChild(p3);

    // Crear y añadir el tipo (puede estar vacío)
```

```
      const p4 = document.createElement("p");
      p4.innerHTML = "<strong>Tipo:</strong> " + (character.type || "N/A");
      details.appendChild(p4);

      // Crear y añadir el género
      const p5 = document.createElement("p");
      p5.innerHTML = "<strong>Género:</strong> " + character.gender;
      details.appendChild(p5);

      // Crear y añadir el origen (character.origin.name)
      const p6 = document.createElement("p");
      p6.innerHTML = "<strong>Origen:</strong> " + character.origin.name;
      details.appendChild(p6);

      // Crear y añadir la ubicación actual (character.location.name)
      const p7 = document.createElement("p");
      p7.innerHTML = "<strong>Ubicación:</strong> " + character.location.name;
      details.appendChild(p7);

      // Crear y añadir el número de episodios
      const p8 = document.createElement("p");
      p8.innerHTML = "<strong>Número de episodios:</strong> " + character.episode.length;
      details.appendChild(p8);
    }
  </script>
</body>
</html>
```

Ejercicios Extra para repasar:

Ejercicio 5: Lista de Usuarios con Filtro

Crea una página que muestre usuarios de la API `https://jsonplaceholder.typicode.com/users`.

Requisitos:

- Mostrar una tabla con: ID, Nombre, Username, Email, Ciudad
- Añadir un campo de texto para filtrar usuarios por nombre (en tiempo real mientras escribes)
- Al hacer clic en un usuario, mostrar sus detalles completos incluyendo: teléfono, empresa y coordenadas geográficas
- Botón para "Limpiar filtro"

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8" />
```

```
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Lista de Usuarios con Filtro</title>
  <style>
    /* Sin estilos - solo HTML básico */
  </style>
</head>
<body>
  <h1>Lista de Usuarios</h1>

  <!-- Botón para cargar usuarios -->
  <button onclick="cargarUsuarios()">Cargar Usuarios</button>
  <button onclick="limpiarTodo()">Limpiar Todo</button>

  <hr>

  <!-- Campo de búsqueda -->
  <div id="campoBusqueda"></div>

  <hr>

  <!-- Contenedor para la tabla de usuarios -->
  <div id="tablaUsuarios"></div>

  <hr>

  <!-- Contenedor para los detalles del usuario -->
  <div id="detalleUsuario"></div>

  <script>
    // Array para guardar todos los usuarios
    let usuarios = [];

    // Array para guardar los usuarios filtrados
    let usuariosFiltrados = [];

    // Función para cargar los usuarios desde la API
    function cargarUsuarios() {
      // Obtener los contenedores
      const divBusqueda = document.getElementById("campoBusqueda");
      const divTabla = document.getElementById("tablaUsuarios");
      const divDetalle = document.getElementById("detalleUsuario");

      // Limpiar y mostrar mensaje de carga
      divBusqueda.innerHTML = "";
      divTabla.innerHTML = "<p>Cargando usuarios...</p>";
      divDetalle.innerHTML = "";

      // Hacer petición a la API
      fetch("https://jsonplaceholder.typicode.com/users")
        .then(response => {
```

```
        // Verificar si la respuesta es correcta
        if (!response.ok) throw new Error("Error al cargar usuarios");
        // Convertir a JSON
        return response.json();
      })
      .then(data => {
        // Guardar los usuarios
        usuarios = data;
        usuariosFiltrados = usuarios; // Al inicio, todos están filtrados

        console.log("Usuarios cargados:", usuarios.length);

        // Mostrar el campo de búsqueda
        mostrarCampoBusqueda();

        // Mostrar la tabla
        mostrarTabla();
      })
      .catch(error => {
        divTabla.innerHTML = "<p>Error: " + error.message + "</p>";
        console.error(error);
      });
  }

  // Función para mostrar el campo de búsqueda
  function mostrarCampoBusqueda() {
    const divBusqueda = document.getElementById("campoBusqueda");

    // Crear el HTML del campo de búsqueda
    let html = "<label for='inputBusqueda'><strong>Buscar por nombre:</strong></label><br>";
    html += "<input type='text' id='inputBusqueda' placeholder='Escribe un nombre...' onke-
yup='filtrarUsuarios()'>";
    html += " <button onclick='limpiarFiltro()'>Limpiar Filtro</button>";
    html += " <span id='contador'></span>";

    divBusqueda.innerHTML = html;
  }

  // Función para mostrar la tabla de usuarios
  function mostrarTabla() {
    const divTabla = document.getElementById("tablaUsuarios");

    // Si no hay usuarios filtrados
    if (usuariosFiltrados.length === 0) {
      divTabla.innerHTML = "<p>No se encontraron usuarios.</p>";
      document.getElementById("contador").textContent = "(0 resultados)";
      return;
    }

    // Actualizar contador
```

```
      document.getElementById("contador").textContent = "(" + usuariosFiltrados.length + " re-
sultados)";

      // Crear la tabla
      let html = "<h2>Usuarios (" + usuariosFiltrados.length + "):</h2>";
      html += "<table border='1'>";

      // Cabecera de la tabla
      html += "<thead>";
      html += "<tr>";
      html += "<th>ID</th>";
      html += "<th>Nombre</th>";
      html += "<th>Username</th>";
      html += "<th>Email</th>";
      html += "<th>Ciudad</th>";
      html += "</tr>";
      html += "</thead>";

      // Cuerpo de la tabla
      html += "<tbody>";

      // Recorrer cada usuario filtrado
      usuariosFiltrados.forEach(usuario => {
        html += "<tr onclick='verDetalle(" + usuario.id + ")' style='cursor: pointer;'>";
        html += "<td>" + usuario.id + "</td>";
        html += "<td>" + usuario.name + "</td>";
        html += "<td>" + usuario.username + "</td>";
        html += "<td>" + usuario.email + "</td>";
        html += "<td>" + usuario.address.city + "</td>"; // ciudad está en address.city
        html += "</tr>";
      });

      html += "</tbody>";
      html += "</table>";

      divTabla.innerHTML = html;
    }

    // Función para filtrar usuarios en tiempo real
    function filtrarUsuarios() {
      // Obtener el texto de búsqueda
      const inputBusqueda = document.getElementById("inputBusqueda");
      const textoBusqueda = inputBusqueda.value.toLowerCase().trim();

      // Si el campo está vacío, mostrar todos
      if (textoBusqueda === "") {
        usuariosFiltrados = usuarios;
        mostrarTabla();
        return;
      }
```

```
  // Filtrar usuarios cuyo nombre contenga el texto
  usuariosFiltrados = [];

  usuarios.forEach(usuario => {
    const nombreUsuario = usuario.name.toLowerCase();

    // Si el nombre contiene el texto de búsqueda
    if (nombreUsuario.indexOf(textoBusqueda) !== -1) {
      usuariosFiltrados.push(usuario);
    }
  });

  // Actualizar la tabla
  mostrarTabla();
}

// Función para ver los detalles de un usuario
function verDetalle(idUsuario) {
  const divDetalle = document.getElementById("detalleUsuario");

  // Buscar el usuario por ID
  let usuarioEncontrado = null;

  usuarios.forEach(usuario => {
    if (usuario.id === idUsuario) {
      usuarioEncontrado = usuario;
    }
  });

  // Si no se encontró
  if (usuarioEncontrado === null) {
    divDetalle.innerHTML = "<p>Usuario no encontrado.</p>";
    return;
  }

  // Crear el HTML con los detalles completos
  let html = "<h2>Detalles del Usuario</h2>";

  // Información básica
  html += "<h3>Información Personal</h3>";
  html += "<ul>";
  html += "<li><strong>ID:</strong> " + usuarioEncontrado.id + "</li>";
  html += "<li><strong>Nombre:</strong> " + usuarioEncontrado.name + "</li>";
  html += "<li><strong>Username:</strong> " + usuarioEncontrado.username + "</li>";
  html += "<li><strong>Email:</strong> " + usuarioEncontrado.email + "</li>";
  html += "<li><strong>Teléfono:</strong> " + usuarioEncontrado.phone + "</li>";
  html += "<li><strong>Website:</strong> " + usuarioEncontrado.website + "</li>";
  html += "</ul>";
```

```
      // Dirección
      html += "<h3>Dirección</h3>";
      html += "<ul>";
      html += "<li><strong>Calle:</strong> " + usuarioEncontrado.address.street + "</li>";
      html += "<li><strong>Suite:</strong> " + usuarioEncontrado.address.suite + "</li>";
      html += "<li><strong>Ciudad:</strong> " + usuarioEncontrado.address.city + "</li>";
      html += "<li><strong>Código Postal:</strong> " + usuarioEncontrado.address.zipcode +
"</li>";
      html += "</ul>";

      // Coordenadas geográficas
      html += "<h3>Coordenadas Geográficas</h3>";
      html += "<ul>";
      html += "<li><strong>Latitud:</strong> " + usuarioEncontrado.address.geo.lat + "</li>";
      html += "<li><strong>Longitud:</strong> " + usuarioEncontrado.address.geo.lng + "</li>";
      html += "</ul>";

      // Empresa
      html += "<h3>Empresa</h3>";
      html += "<ul>";
      html += "<li><strong>Nombre:</strong> " + usuarioEncontrado.company.name + "</li>";
      html += "<li><strong>Eslogan:</strong> " + usuarioEncontrado.company.catchPhrase +
"</li>";
      html += "<li><strong>Actividad:</strong> " + usuarioEncontrado.company.bs + "</li>";
      html += "</ul>";

      // Botón para cerrar detalles
      html += "<button onclick='cerrarDetalle()'>Cerrar Detalles</button>";

      divDetalle.innerHTML = html;
    }

    // Función para cerrar los detalles
    function cerrarDetalle() {
      document.getElementById("detalleUsuario").innerHTML = "";
    }

    // Función para limpiar el filtro
    function limpiarFiltro() {
      // Limpiar el input
      document.getElementById("inputBusqueda").value = "";

      // Resetear el array filtrado
      usuariosFiltrados = usuarios;

      // Actualizar la tabla
      mostrarTabla();

      // Cerrar detalles
      cerrarDetalle();
```

```
    }

    // Función para limpiar todo
    function limpiarTodo() {
      document.getElementById("campoBusqueda").innerHTML = "";
      document.getElementById("tablaUsuarios").innerHTML = "";
      document.getElementById("detalleUsuario").innerHTML = "";
      usuarios = [];
      usuariosFiltrados = [];
    }
  </script>
</body>
</html>
```

Ejercicio 6: Galería de Fotos con Paginación

Usa la API `https://jsonplaceholder.typicode.com/photos` que devuelve 5000 fotos.

Requisitos:

- Cargar solo las primeras 20 fotos (usa `?_limit=20`)
- Mostrar en una cuadrícula: thumbnail, título y ID
- Botones "Página Anterior" y "Página Siguiente" para navegar
- Al hacer clic en una foto, mostrar la imagen completa en grande
- Contador: "Mostrando fotos 1-20 de 5000"

Pista: Usa parámetros `_limit` y `_start` en la URL:

https://jsonplaceholder.typicode.com/photos?_start=0&_limit=20

https://jsonplaceholder.typicode.com/photos?_start=20&_limit=20

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Galería de Fotos con Paginación</title>
  <style>
```

```
    /* Sin estilos - solo HTML básico */
  </style>
</head>
<body>
  <h1>Galería de Fotos</h1>

  <!-- Botón para cargar la galería -->
  <button onclick="cargarPrimeraPagina()">Cargar Galería</button>
  <button onclick="limpiarGaleria()">Limpiar</button>

  <hr>

  <!-- Controles de navegación -->
  <div id="controles"></div>

  <hr>

  <!-- Contenedor para las fotos -->
  <div id="galeria"></div>

  <hr>

  <!-- Modal para mostrar la imagen grande -->
  <div id="modal"></div>

  <script>
    // Variables globales
    let fotos = []; // Array con las fotos de la página actual
    let paginaActual = 1; // Página actual (empezamos en 1)
    const fotosPorPagina = 20; // Número de fotos por página
    const totalFotos = 5000; // Total de fotos en la API
    const totalPaginas = Math.ceil(totalFotos / fotosPorPagina); // Total de páginas = 250

    // Función para cargar la primera página
    function cargarPrimeraPagina() {
      paginaActual = 1;
      cargarFotos();
    }

    // Función principal para cargar fotos de una página
    function cargarFotos() {
      const divGaleria = document.getElementById("galeria");
      const divControles = document.getElementById("controles");
      const divModal = document.getElementById("modal");

      // Limpiar y mostrar mensaje de carga
      divGaleria.innerHTML = "<p>Cargando fotos...</p>";
      divControles.innerHTML = "";
      divModal.innerHTML = "";
```

```
      // Calcular el índice de inicio según la página actual
      // Página 1: start = 0  (fotos 1-20)
      // Página 2: start = 20 (fotos 21-40)
      // Página 3: start = 40 (fotos 41-60)
      const start = (paginaActual - 1) * fotosPorPagina;

      // Construir la URL con los parámetros
      const url = "https://jsonplaceholder.typicode.com/photos?_start=" + start + "&_limit=" +
fotosPorPagina;

      console.log("Cargando URL:", url);
      console.log("Página:", paginaActual, "Start:", start);

      // Hacer petición fetch
      fetch(url)
        .then(response => {
          if (!response.ok) throw new Error("Error al cargar fotos");
          return response.json();
        })
        .then(data => {
          // Guardar las fotos
          fotos = data;

          console.log("Fotos cargadas:", fotos.length);

          // Mostrar la galería
          mostrarGaleria();

          // Mostrar los controles de navegación
          mostrarControles();
        })
        .catch(error => {
          divGaleria.innerHTML = "<p>Error: " + error.message + "</p>";
          console.error(error);
        });
    }

    // Función para mostrar la galería de fotos
    function mostrarGaleria() {
      const divGaleria = document.getElementById("galeria");

      if (fotos.length === 0) {
        divGaleria.innerHTML = "<p>No hay fotos para mostrar.</p>";
        return;
      }

      // Crear el HTML con las fotos
      let html = "";

      fotos.forEach((foto, index) => {
```

```
        // Cada foto es un div con thumbnail, título e ID
        html += "<div onclick='verImagenGrande(" + index + ")' style='display: inline-block;
margin: 10px; text-align: center; cursor: pointer; border: 1px solid #ccc; padding: 10px;'>";
        html += "<img src='" + foto.thumbnailUrl + "' alt='" + foto.title + "'>";
        html += "<p><strong>ID:</strong> " + foto.id + "</p>";
        html += "<p>" + foto.title + "</p>";
        html += "</div>";
      });

      divGaleria.innerHTML = html;
    }

    // Función para mostrar los controles de navegación
    function mostrarControles() {
      const divControles = document.getElementById("controles");

      // Calcular el rango de fotos que estamos mostrando
      const fotoInicio = (paginaActual - 1) * fotosPorPagina + 1;
      const fotoFin = Math.min(paginaActual * fotosPorPagina, totalFotos);

      let html = "";

      // Botón Página Anterior
      if (paginaActual > 1) {
        html += "<button onclick='paginaAnterior()'>Página Anterior</button> ";
      } else {
        html += "<button disabled>Página Anterior</button> ";
      }

      // Información de la página
      html += "<span>Página " + paginaActual + " de " + totalPaginas + "</span> ";

      // Botón Página Siguiente
      if (paginaActual < totalPaginas) {
        html += "<button onclick='paginaSiguiente()'>Página Siguiente</button>";
      } else {
        html += "<button disabled>Página Siguiente</button>";
      }

      html += "<br><br>";

      // Contador de fotos
      html += "<span>Mostrando fotos " + fotoInicio + "-" + fotoFin + " de " + totalFotos +
"</span>";

      divControles.innerHTML = html;
    }

    // Función para ir a la página anterior
    function paginaAnterior() {
```

```
      if (paginaActual > 1) {
        paginaActual--;
        cargarFotos();
      }
    }

    // Función para ir a la página siguiente
    function paginaSiguiente() {
      if (paginaActual < totalPaginas) {
        paginaActual++;
        cargarFotos();
      }
    }

    // Función para ver la imagen en grande (modal)
    function verImagenGrande(indice) {
      const foto = fotos[indice];
      const divModal = document.getElementById("modal");

      // Crear el HTML del modal
      let html = "<div onclick='cerrarModal()' style='position: fixed; top: 0; left: 0; width:
100%; height: 100%; background-color: rgba(0,0,0,0.8); z-index: 1000;'>";
      html += "<div style='text-align: center; padding-top: 50px;'>";
      html += "<span onclick='cerrarModal()' style='color: white; font-size: 40px; cursor: poin-
ter;'>×</span><br>";
      html += "<img src='" + foto.url + "' alt='" + foto.title + "' style='max-width: 90%; max-
height: 80vh;'><br>";
      html += "<p style='color: white; font-size: 18px;'>" + foto.title + " (ID: " + foto.id +
")</p>";
      html += "</div>";
      html += "</div>";

      divModal.innerHTML = html;
    }

    // Función para cerrar el modal
    function cerrarModal() {
      document.getElementById("modal").innerHTML = "";
    }

    // Función para limpiar la galería
    function limpiarGaleria() {
      document.getElementById("controles").innerHTML = "";
      document.getElementById("galeria").innerHTML = "";
      document.getElementById("modal").innerHTML = "";
      fotos = [];
      paginaActual = 1;
    }
  </script>
</body>
```

```
</html>
```

Ejercicio 7: Blog de Posts y Comentarios

Usa `https://jsonplaceholder.typicode.com/posts`

Requisitos:

- Mostrar lista de los primeros 10 posts (título y usuario ID)
- Al hacer clic en un post, mostrar su contenido completo
- Debajo del post, cargar y mostrar todos sus comentarios desde: `https://jsonplaceholder.typicode.com/posts/{id}/comments`
- Mostrar: nombre del comentarista, email y cuerpo del comentario
- Botón "Volver a la lista"

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Blog de Posts y Comentarios</title>
  <style>
    /* Sin estilos - solo HTML básico */
  </style>
</head>
<body>
  <h1>Blog de Posts</h1>

  <!-- Botón para cargar posts -->
  <button onclick="cargarPosts()">Cargar Posts</button>
  <button onclick="limpiarBlog()">Limpiar</button>

  <hr>

  <!-- Contenedor para la lista de posts -->
  <div id="listaPosts"></div>

  <!-- Contenedor para el post completo con comentarios -->
  <div id="postCompleto"></div>

  <script>
    // Array para guardar los posts
```

```
let posts = [];

// Función para cargar los primeros 10 posts
function cargarPosts() {
  const divLista = document.getElementById("listaPosts");
  const divPost = document.getElementById("postCompleto");

  // Limpiar y mostrar mensaje de carga
  divLista.innerHTML = "<p>Cargando posts...</p>";
  divPost.innerHTML = "";

  // Hacer petición para obtener los primeros 10 posts
  // Usamos ?_limit=10 para limitar los resultados
  fetch("https://jsonplaceholder.typicode.com/posts?_limit=10")
    .then(response => {
      if (!response.ok) throw new Error("Error al cargar posts");
      return response.json();
    })
    .then(data => {
      // Guardar los posts
      posts = data;

      console.log("Posts cargados:", posts.length);

      // Mostrar la lista
      mostrarListaPosts();
    })
    .catch(error => {
      divLista.innerHTML = "<p>Error: " + error.message + "</p>";
      console.error(error);
    });
}

// Función para mostrar la lista de posts
function mostrarListaPosts() {
  const divLista = document.getElementById("listaPosts");

  if (posts.length === 0) {
    divLista.innerHTML = "<p>No hay posts para mostrar.</p>";
    return;
  }

  // Crear el HTML con la lista
  let html = "<h2>Lista de Posts (" + posts.length + "):</h2>";
  html += "<ul>";

  // Recorrer cada post
  posts.forEach(post => {
    html += "<li>";
    html += "<a href='#' onclick='verPostCompleto(" + post.id + "); return false;'>";
```

```
        html += "<strong>" + post.title + "</strong>";
        html += "</a>";
        html += " (Usuario ID: " + post.userId + ")";
        html += "</li>";
      });

      html += "</ul>";

      divLista.innerHTML = html;
    }

    // Función para ver un post completo con sus comentarios
    function verPostCompleto(idPost) {
      const divLista = document.getElementById("listaPosts");
      const divPost = document.getElementById("postCompleto");

      // Ocultar la lista de posts
      divLista.innerHTML = "";

      // Mostrar mensaje de carga
      divPost.innerHTML = "<p>Cargando post y comentarios...</p>";

      // Buscar el post en nuestro array
      let postEncontrado = null;

      posts.forEach(post => {
        if (post.id === idPost) {
          postEncontrado = post;
        }
      });

      // Si no encontramos el post
      if (postEncontrado === null) {
        divPost.innerHTML = "<p>Error: Post no encontrado.</p>";
        return;
      }

      // Mostrar el post
      let html = "<h2>" + postEncontrado.title + "</h2>";
      html += "<p><strong>Usuario ID:</strong> " + postEncontrado.userId + " | <strong>Post
ID:</strong> " + postEncontrado.id + "</p>";
      html += "<hr>";
      html += "<p>" + postEncontrado.body + "</p>";
      html += "<hr>";

      // Botón para volver
      html += "<button onclick='volverAlista()'>Volver a la lista</button>";
      html += "<br><br>";

      // Sección de comentarios (inicialmente con mensaje de carga)
```

```
    html += "<div id='seccionComentarios'><p>Cargando comentarios...</p></div>";

    divPost.innerHTML = html;

    // Ahora cargamos los comentarios
    cargarComentarios(idPost);
  }

  // Función para cargar los comentarios de un post
  function cargarComentarios(idPost) {
    // URL para obtener comentarios de un post específico
    const url = "https://jsonplaceholder.typicode.com/posts/" + idPost + "/comments";

    console.log("Cargando comentarios desde:", url);

    // Hacer petición fetch
    fetch(url)
      .then(response => {
        if (!response.ok) throw new Error("Error al cargar comentarios");
        return response.json();
      })
      .then(comentarios => {
        console.log("Comentarios cargados:", comentarios.length);

        // Mostrar los comentarios
        mostrarComentarios(comentarios);
      })
      .catch(error => {
        document.getElementById("seccionComentarios").innerHTML =
          "<p>Error: " + error.message + "</p>";
        console.error(error);
      });
  }

  // Función para mostrar los comentarios
  function mostrarComentarios(comentarios) {
    const divComentarios = document.getElementById("seccionComentarios");

    // Verificar que haya comentarios
    if (comentarios.length === 0) {
      divComentarios.innerHTML = "<p>Este post no tiene comentarios.</p>";
      return;
    }

    // Crear el HTML con los comentarios
    let html = "<h3>Comentarios (" + comentarios.length + "):</h3>";

    // Recorrer cada comentario
    comentarios.forEach(comentario => {
      html += "<div>";
```

```
        html += "<hr>";
        html += "<h4>" + comentario.name + "</h4>";
        html += "<p><strong>Email:</strong> " + comentario.email + "</p>";
        html += "<p><strong>Comentario:</strong> " + comentario.body + "</p>";
        html += "</div>";
      });

      divComentarios.innerHTML = html;
    }

    // Función para volver a la lista de posts
    function volverAlista() {
      const divLista = document.getElementById("listaPosts");
      const divPost = document.getElementById("postCompleto");

      // Limpiar el post
      divPost.innerHTML = "";

      // Volver a mostrar la lista
      mostrarListaPosts();
    }

    // Función para limpiar todo
    function limpiarBlog() {
      document.getElementById("listaPosts").innerHTML = "";
      document.getElementById("postCompleto").innerHTML = "";
      posts = [];
    }
  </script>
</body>
</html>
```

Ejercicio 8: Conversor de Monedas en Tiempo Real

Usa la API gratuita: `https://api.exchangerate-api.com/v4/latest/USD`

Requisitos:

- Cargar las tasas de cambio actuales
- Dos selectores: "De" y "A" con diferentes monedas (USD, EUR, GBP, JPY, MXN, etc.)
- Un input para la cantidad a convertir

- Botón "Convertir" que muestre el resultado
- Mostrar la fecha de última actualización de las tasas
- Ejemplo: "100 USD = 84.50 EUR (actualizado: 2024-01-15)"

```html
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Conversor de Monedas</title>
  <style>
    /* Sin estilos - solo HTML básico */
  </style>
</head>
<body>
  <h1>Conversor de Monedas en Tiempo Real</h1>

  <!-- Botón para cargar tasas -->
  <button onclick="cargarTasas()">Cargar Tasas de Cambio</button>
  <button onclick="limpiarConversor()">Limpiar</button>

  <hr>

  <!-- Información de las tasas -->
  <div id="infoTasas"></div>

  <hr>

  <!-- Formulario de conversión -->
  <div id="formularioConversion"></div>

  <hr>

  <!-- Resultado de la conversión -->
  <div id="resultadoConversion"></div>

  <script>
    // Variables globales
    let tasas = {}; // Objeto con las tasas de cambio
    let monedaBase = ""; // Moneda base de las tasas
    let fechaActualizacion = ""; // Fecha de última actualización
    let monedasDisponibles = []; // Array con las monedas disponibles

    // Función para cargar las tasas de cambio
    function cargarTasas() {
      const divInfo = document.getElementById("infoTasas");
      const divFormulario = document.getElementById("formularioConversion");
      const divResultado = document.getElementById("resultadoConversion");
```

```
    // Limpiar y mostrar mensaje de carga
    divInfo.innerHTML = "<p>Cargando tasas de cambio...</p>";
    divFormulario.innerHTML = "";
    divResultado.innerHTML = "";

    // Hacer petición a la API
    fetch("https://api.exchangerate-api.com/v4/latest/USD")
      .then(response => {
        if (!response.ok) throw new Error("Error al cargar tasas");
        return response.json();
      })
      .then(data => {
        // Guardar los datos
        monedaBase = data.base; // "USD"
        tasas = data.rates; // Objeto con todas las tasas
        fechaActualizacion = data.date; // Fecha de actualización

        // Extraer las monedas disponibles (claves del objeto rates)
        // Object.keys() devuelve un array con las claves
        monedasDisponibles = Object.keys(tasas).sort(); // Ordenar alfabéticamente

        console.log("Moneda base:", monedaBase);
        console.log("Total de monedas:", monedasDisponibles.length);
        console.log("Fecha actualización:", fechaActualizacion);

        // Mostrar información
        mostrarInformacion();

        // Mostrar el formulario
        mostrarFormulario();
      })
      .catch(error => {
        divInfo.innerHTML = "<p>Error: " + error.message + "</p>";
        console.error(error);
      });
  }

  // Función para mostrar información de las tasas
  function mostrarInformacion() {
    const divInfo = document.getElementById("infoTasas");

    let html = "<h3>Información de las Tasas</h3>";
    html += "<p><strong>Moneda base:</strong> " + monedaBase + "</p>";
    html += "<p><strong>Total de monedas disponibles:</strong> " + monedasDisponibles.length +
"</p>";
    html += "<p><strong>Fecha de actualización:</strong> " + fechaActualizacion + "</p>";

    divInfo.innerHTML = html;
  }
```

```
// Función para mostrar el formulario de conversión
function mostrarFormulario() {
  const divFormulario = document.getElementById("formularioConversion");

  let html = "<h3>Convertir Moneda</h3>";

  // Input para la cantidad
  html += "<label for='cantidad'><strong>Cantidad:</strong></label><br>";
  html += "<input type='number' id='cantidad' value='100' min='0' step='0.01'>";
  html += "<br><br>";

  // Selector "De" (moneda origen)
  html += "<label for='monedaOrigen'><strong>De:</strong></label><br>";
  html += "<select id='monedaOrigen'>";

  // Recorrer todas las monedas disponibles
  monedasDisponibles.forEach(moneda => {
    // Marcar USD como seleccionado por defecto
    const selected = moneda === "USD" ? " selected" : "";
    html += "<option value='" + moneda + "'" + selected + ">" + moneda + "</option>";
  });

  html += "</select>";
  html += "<br><br>";

  // Selector "A" (moneda destino)
  html += "<label for='monedaDestino'><strong>A:</strong></label><br>";
  html += "<select id='monedaDestino'>";

  // Recorrer todas las monedas disponibles
  monedasDisponibles.forEach(moneda => {
    // Marcar EUR como seleccionado por defecto
    const selected = moneda === "EUR" ? " selected" : "";
    html += "<option value='" + moneda + "'" + selected + ">" + moneda + "</option>";
  });

  html += "</select>";
  html += "<br><br>";

  // Botón para convertir
  html += "<button onclick='convertirMoneda()'>Convertir</button>";

  divFormulario.innerHTML = html;
}

// Función para convertir moneda
function convertirMoneda() {
  const divResultado = document.getElementById("resultadoConversion");
```

```
      // Obtener los valores del formulario
      const cantidad = parseFloat(document.getElementById("cantidad").value);
      const monedaOrigen = document.getElementById("monedaOrigen").value;
      const monedaDestino = document.getElementById("monedaDestino").value;

      // Validar que la cantidad sea válida
      if (isNaN(cantidad) || cantidad <= 0) {
        divResultado.innerHTML = "<p>Error: Introduce una cantidad válida mayor que 0.</p>";
        return;
      }

      // Validar que las monedas sean diferentes
      if (monedaOrigen === monedaDestino) {
        divResultado.innerHTML = "<p>Advertencia: Las monedas son iguales.</p>";
        divResultado.innerHTML += "<p>" + cantidad + " " + monedaOrigen + " = " + cantidad + " "
+ monedaDestino + "</p>";
        return;
      }

      // CÁLCULO DE LA CONVERSIÓN
      // Las tasas están en base USD
      // Para convertir de cualquier moneda a otra, necesitamos dos pasos:

      // Paso 1: Convertir la moneda origen a USD
      let valorEnUSD;

      if (monedaOrigen === "USD") {
        // Si ya es USD, no hace falta conversión
        valorEnUSD = cantidad;
      } else {
        // Para convertir a USD, dividimos por la tasa de la moneda origen
        // Ejemplo: 100 EUR / 0.85 = 117.65 USD
        valorEnUSD = cantidad / tasas[monedaOrigen];
      }

      // Paso 2: Convertir de USD a la moneda destino
      let resultado;

      if (monedaDestino === "USD") {
        // Si el destino es USD, ya lo tenemos
        resultado = valorEnUSD;
      } else {
        // Multiplicamos el valor en USD por la tasa de la moneda destino
        // Ejemplo: 117.65 USD * 110.25 = 12970.31 JPY
        resultado = valorEnUSD * tasas[monedaDestino];
      }

      // Redondear a 2 decimales
      resultado = Math.round(resultado * 100) / 100;
```

```
      // Mostrar el resultado
      let html = "<h3>Resultado de la Conversión</h3>";
      html += "<p><strong>" + cantidad + " " + monedaOrigen + " = " + resultado + " " + moneda-
Destino + "</strong></p>";
      html += "<p>(Actualizado: " + fechaActualizacion + ")</p>";

      divResultado.innerHTML = html;
    }

    // Función para limpiar el conversor
    function limpiarConversor() {
      document.getElementById("infoTasas").innerHTML = "";
      document.getElementById("formularioConversion").innerHTML = "";
      document.getElementById("resultadoConversion").innerHTML = "";
      tasas = {};
      monedaBase = "";
      fechaActualizacion = "";
      monedasDisponibles = [];
    }
  </script>
</body>
</html>
```

Ejercicio 9: Buscador de Pokémon

Usa la PokéAPI: `https://pokeapi.co/api/v2/pokemon?limit=151` (primeros 151 Pokémon)

Requisitos:

- Cargar la lista de los 151 Pokémon originales
- Mostrar sus nombres en una lista
- Buscador: input de texto que filtre la lista en tiempo real
- Al hacer clic en un Pokémon, hacer una segunda petición a: `https://pokeapi.co/api/v2/pokemon/{nombre}`
- Mostrar: imagen (sprite), tipos, altura, peso, habilidades
- Mostrar las estadísticas (HP, Attack, Defense, etc.) en una lista

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <title>Buscador de Pokémon</title>
</head>
<body>

    <h1>Buscador de Pokémon</h1>
    <p>Explora los primeros 151 Pokémon de la primera generación.</p>

    <!-- Botones de control -->
    <button onclick="cargarPokemon()">Cargar Pokémon</button>
    <button onclick="limpiarBuscador()">Limpiar</button>

    <hr>

    <!-- Campo de búsqueda -->
    <div id="campoBusqueda"></div>

    <hr>

    <!-- Lista de Pokémon -->
    <div id="listaPokemon"></div>

    <hr>

    <!-- Detalles del Pokémon seleccionado -->
    <div id="detallePokemon"></div>

    <script>
        // --- 1. Variables globales ---
        let pokemonList = []; // Array con todos los Pokémon
        let pokemonFiltrados = []; // Array con Pokémon filtrados

        // --- 2. Función para cargar la lista de Pokémon ---
        function cargarPokemon() {
            let divBusqueda = document.getElementById("campoBusqueda");
            let divLista = document.getElementById("listaPokemon");
            let divDetalle = document.getElementById("detallePokemon");

            // Limpiamos y mostramos mensaje de carga
            divBusqueda.innerHTML = "";
            divLista.innerHTML = "<p>Cargando lista de Pokémon...</p>";
            divDetalle.innerHTML = "";

            // --- Realizamos la petición fetch para obtener los primeros 151 Pokémon ---
            fetch("https://pokeapi.co/api/v2/pokemon?limit=151")
                .then(function(response) {
                    // Verificamos que la respuesta sea correcta
                    if (!response.ok) {
                        throw new Error("Error en la petición: " + response.status);
                    }
                    return response.json();
```

```
            })
            .then(function(data) {
                // Guardamos la lista de Pokémon
                pokemonList = data.results;
                pokemonFiltrados = pokemonList; // Inicialmente todos están filtrados

                console.log("Total de Pokémon cargados:", pokemonList.length);
                console.log("Primer Pokémon:", pokemonList[0]);

                // Mostramos el campo de búsqueda
                mostrarCampoBusqueda();

                // Mostramos la lista
                mostrarListaPokemon();
            })
            .catch(function(error) {
                // Capturamos cualquier error
                divLista.innerHTML = "<p>Error al cargar los Pokémon: " + error.message +
"</p>";
                console.error("Error:", error);
            });
    }

    // --- 3. Función para mostrar el campo de búsqueda ---
    function mostrarCampoBusqueda() {
        let divBusqueda = document.getElementById("campoBusqueda");

        let html = "<label for='inputBusqueda'><strong>Buscar Pokémon:</strong></la-
bel><br>";
        html += "<input type='text' id='inputBusqueda' placeholder='Escribe el nombre de un
Pokémon...' onkeyup='filtrarPokemon()'>";
        html += " <span id='contadorResultados'></span>";

        divBusqueda.innerHTML = html;
    }

    // --- 4. Función para mostrar la lista de Pokémon ---
    function mostrarListaPokemon() {
        let divLista = document.getElementById("listaPokemon");

        // Verificamos que haya Pokémon
        if (pokemonFiltrados.length === 0) {
            divLista.innerHTML = "<p>No se encontraron Pokémon con ese nombre.</p>";
            document.getElementById("contadorResultados").textContent = "(0 resultados)";
            return;
        }

        // Actualizamos el contador
        document.getElementById("contadorResultados").textContent =
            "(" + pokemonFiltrados.length + " resultados)";
```

```
            // Creamos el HTML con la lista
            let html = "<h2>Lista de Pokémon (" + pokemonFiltrados.length + "):</h2>";
            html += "<ul>";

            for (let i = 0; i < pokemonFiltrados.length; i++) {
                let pokemon = pokemonFiltrados[i];

                // Capitalizamos la primera letra del nombre
                let nombreCapitalizado = pokemon.name.charAt(0).toUpperCase() +
pokemon.name.slice(1);

                html += "<li>";
                html += "<a href='#' onclick='verDetallePokemon(\"" + pokemon.name + "\"); re-
turn false;'>";
                html += nombreCapitalizado;
                html += "</a>";
                html += "</li>";
            }

            html += "</ul>";

            // Mostramos la lista
            divLista.innerHTML = html;
        }

        // --- 5. Función para filtrar Pokémon en tiempo real ---
        function filtrarPokemon() {
            let inputBusqueda = document.getElementById("inputBusqueda");
            let textoBusqueda = inputBusqueda.value.toLowerCase().trim();

            // Si el campo está vacío, mostramos todos
            if (textoBusqueda === "") {
                pokemonFiltrados = pokemonList;
                mostrarListaPokemon();
                return;
            }

            // Filtramos los Pokémon que contengan el texto en su nombre
            pokemonFiltrados = [];

            for (let i = 0; i < pokemonList.length; i++) {
                let nombrePokemon = pokemonList[i].name.toLowerCase();

                if (nombrePokemon.indexOf(textoBusqueda) !== -1) {
                    pokemonFiltrados.push(pokemonList[i]);
                }
            }

            // Actualizamos la lista
```

```
            mostrarListaPokemon();
        }

        // --- 6. Función para ver los detalles de un Pokémon ---
        function verDetallePokemon(nombrePokemon) {
            let divDetalle = document.getElementById("detallePokemon");

            // Mostramos mensaje de carga
            divDetalle.innerHTML = "<p>Cargando detalles de " + nombrePokemon + "...</p>";

            // Hacemos scroll hacia los detalles
            divDetalle.scrollIntoView({ behavior: 'smooth' });

            // --- Segunda petición fetch para obtener los detalles ---
            let url = "https://pokeapi.co/api/v2/pokemon/" + nombrePokemon;

            console.log("Cargando detalles desde:", url);

            fetch(url)
                .then(function(response) {
                    if (!response.ok) {
                        throw new Error("Error al cargar detalles: " + response.status);
                    }
                    return response.json();
                })
                .then(function(pokemon) {
                    console.log("Detalles del Pokémon:", pokemon);

                    // Mostramos los detalles
                    mostrarDetalles(pokemon);
                })
                .catch(function(error) {
                    divDetalle.innerHTML = "<p>Error al cargar los detalles: " + error.message +
"</p>";

                    console.error("Error:", error);
                });
        }

        // --- 7. Función para mostrar los detalles completos ---
        function mostrarDetalles(pokemon) {
            let divDetalle = document.getElementById("detallePokemon");

            // Capitalizamos el nombre
            let nombreCapitalizado = pokemon.name.charAt(0).toUpperCase() +
pokemon.name.slice(1);

            let html = "<h2>Detalles de " + nombreCapitalizado + "</h2>";

            // --- Imagen del Pokémon (sprite) ---
```

```
            html += "<img src='" + pokemon.sprites.front_default + "' alt='" + pokemon.name +
"'>";
            html += "<br><br>";

            // --- Información básica ---
            html += "<h3>Información Básica</h3>";
            html += "<ul>";
            html += "<li><strong>ID:</strong> #" + pokemon.id + "</li>";
            html += "<li><strong>Nombre:</strong> " + nombreCapitalizado + "</li>";
            html += "<li><strong>Altura:</strong> " + (pokemon.height / 10) + " m</li>";
            html += "<li><strong>Peso:</strong> " + (pokemon.weight / 10) + " kg</li>";
            html += "</ul>";

            // --- Tipos ---
            html += "<h3>Tipos</h3>";
            html += "<ul>";

            for (let i = 0; i < pokemon.types.length; i++) {
                let tipo = pokemon.types[i].type.name;
                // Capitalizamos
                tipo = tipo.charAt(0).toUpperCase() + tipo.slice(1);
                html += "<li>" + tipo + "</li>";
            }

            html += "</ul>";

            // --- Habilidades ---
            html += "<h3>Habilidades</h3>";
            html += "<ul>";

            for (let i = 0; i < pokemon.abilities.length; i++) {
                let habilidad = pokemon.abilities[i].ability.name;
                let esOculta = pokemon.abilities[i].is_hidden;

                // Capitalizamos y reemplazamos guiones por espacios
                habilidad = habilidad.replace(/-/g, ' ');
                habilidad = habilidad.charAt(0).toUpperCase() + habilidad.slice(1);

                html += "<li>" + habilidad;
                if (esOculta) {
                    html += " (Habilidad Oculta)";
                }
                html += "</li>";
            }

            html += "</ul>";

            // --- Estadísticas ---
            html += "<h3>Estadísticas Base</h3>";
            html += "<ul>";
```

```
        for (let i = 0; i < pokemon.stats.length; i++) {
            let stat = pokemon.stats[i];
            let nombreStat = stat.stat.name;
            let valorStat = stat.base_stat;

            // Traducimos los nombres de las estadísticas
            let nombreTraducido = traducirEstadistica(nombreStat);

            html += "<li><strong>" + nombreTraducido + ":</strong> " + valorStat + "</li>";
        }

        html += "</ul>";

        // Botón para cerrar detalles
        html += "<br>";
        html += "<button onclick='cerrarDetalles()'>Cerrar Detalles</button>";

        // Mostramos los detalles
        divDetalle.innerHTML = html;
    }

    // --- 8. Función auxiliar para traducir nombres de estadísticas ---
    function traducirEstadistica(nombreStat) {
        let traducciones = {
            "hp": "HP (Puntos de Salud)",
            "attack": "Ataque",
            "defense": "Defensa",
            "special-attack": "Ataque Especial",
            "special-defense": "Defensa Especial",
            "speed": "Velocidad"
        };

        return traducciones[nombreStat] || nombreStat;
    }

    // --- 9. Función para cerrar los detalles ---
    function cerrarDetalles() {
        document.getElementById("detallePokemon").innerHTML = "";
    }

    // --- 10. Función para limpiar el buscador ---
    function limpiarBuscador() {
        document.getElementById("campoBusqueda").innerHTML = "";
        document.getElementById("listaPokemon").innerHTML = "";
        document.getElementById("detallePokemon").innerHTML = "";
        pokemonList = [];
        pokemonFiltrados = [];
    }
</script>
```

```
</body>
</html>
```

Ejercicio completo de práctica (GET, POST, PUT, DELETE)

CARPETA EXTRAS CRUD

```
 <!DOCTYPE html>
<html lang="es">
<head>
<meta charset="UTF-8">
<title>CRUD Completo - Tareas API REST</title>
<style>
    body { font-family: Arial; padding: 20px; }
    table { border-collapse: collapse; width: 80%; margin-top: 20px; }
    th, td { border: 1px solid #ccc; padding: 8px; text-align: left; }
    th { background: #f0f0f0; }
    button { margin: 5px; padding: 10px; }
</style>
</head>
<body>

<h2>Gestión de Tareas - CRUD Completo</h2>

<!-- Botones de acciones -->
<button onclick="listarTareas()">GET - Listar Tareas</button>
<button onclick="crearTarea()">POST - Crear Tarea</button>
<button onclick="actualizarTarea()">PUT - Actualizar Tarea</button>
<button onclick="borrarTarea()">DELETE - Borrar Tarea</button>

<hr>

<!-- Contenedor para la tabla -->
<div id="resultado"></div>

<script>
  // URL de la API (cambiar por la del examen)
  const URL = "https://jsonplaceholder.typicode.com/todos"; // API de prueba
  let tareas = [];

  // ======================================
  // GET - Listar todas las tareas
```

```
// ========================================
function listarTareas() {
  console.log("GET: Listando tareas...");

  fetch(URL + "?_limit=10") // Limitar a 10 para pruebas
    .then(response => {
      if (!response.ok) throw new Error("Error al cargar tareas");
      return response.json();
    })
    .then(data => {
      tareas = data;
      console.log("Tareas cargadas:", tareas);
      mostrarTabla(tareas);
    })
    .catch(error => {
      console.error("Error:", error);
      alert("Error al cargar las tareas");
    });
}

// ========================================
// Función para mostrar la tabla
// ========================================
function mostrarTabla(arr) {
  if (arr.length === 0) {
    document.getElementById("resultado").innerHTML = "<p>No hay tareas para mostrar.</p>";
    return;
  }

  let html = `
    <table>
      <tr>
        <th>ID</th>
        <th>Usuario ID</th>
        <th>Título</th>
        <th>Completada</th>
      </tr>
  `;

  arr.forEach(item => {
    html += `
      <tr>
        <td>${item.id}</td>
        <td>${item.userId}</td>
        <td>${item.title}</td>
        <td>${item.completed ? "Sí" : "No"}</td>
      </tr>
    `;
  });
```

```
  html += "</table>";
  document.getElementById("resultado").innerHTML = html;
}

// ========================================
// POST - Crear una nueva tarea
// ========================================
function crearTarea() {
  const titulo = prompt("Introduce el título de la nueva tarea:");

  if (!titulo) {
    alert("Debes introducir un título");
    return;
  }

  console.log("POST: Creando tarea...");

  // Datos de la nueva tarea
  const nuevaTarea = {
    userId: 1,
    title: titulo,
    completed: false
  };

  fetch(URL, {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json'
    },
    body: JSON.stringify(nuevaTarea)
  })
  .then(response => {
    if (!response.ok) throw new Error("Error al crear tarea");
    return response.json();
  })
  .then(data => {
    console.log("Tarea creada:", data);
    alert("Tarea creada correctamente con ID: " + data.id);

    // Actualizar la tabla (añadir la nueva tarea)
    tareas.unshift(data); // Añadir al inicio
    mostrarTabla(tareas);
  })
  .catch(error => {
    console.error("Error:", error);
    alert("Error al crear la tarea");
  });
}

// ========================================
```

```
// PUT - Actualizar una tarea existente
// ======================================
function actualizarTarea() {
  const id = prompt("Introduce el ID de la tarea a actualizar:");

  if (!id) {
    alert("Debes introducir un ID");
    return;
  }

  const nuevoTitulo = prompt("Introduce el nuevo título:");

  if (!nuevoTitulo) {
    alert("Debes introducir un título");
    return;
  }

  console.log("PUT: Actualizando tarea ID " + id);

  // Datos actualizados
  const tareaActualizada = {
    userId: 1,
    title: nuevoTitulo,
    completed: false
  };

  fetch(URL + "/" + id, {
    method: 'PUT',
    headers: {
      'Content-Type': 'application/json'
    },
    body: JSON.stringify(tareaActualizada)
  })
  .then(response => {
    if (!response.ok) throw new Error("Error al actualizar tarea");
    return response.json();
  })
  .then(data => {
    console.log("Tarea actualizada:", data);
    alert("Tarea ID " + id + " actualizada correctamente");

    // Actualizar en el array local
    const index = tareas.findIndex(t => t.id == id);
    if (index !== -1) {
      tareas[index] = data;
      mostrarTabla(tareas);
    }
  })
  .catch(error => {
    console.error("Error:", error);
```

```
      alert("Error al actualizar la tarea");
    });
  }

  // ======================================
  // DELETE - Borrar una tarea
  // ======================================
  function borrarTarea() {
    const id = prompt("Introduce el ID de la tarea que deseas borrar:");

    if (!id) {
      alert("Debes introducir un ID");
      return;
    }

    console.log("DELETE: Borrando tarea ID " + id);

    fetch(URL + "/" + id, {
      method: 'DELETE'
    })
    .then(response => {
      if (!response.ok) throw new Error("Error al borrar tarea");
      console.log("Tarea eliminada correctamente");
      alert("Tarea ID " + id + " eliminada correctamente");

      // Eliminar del array local
      tareas = tareas.filter(t => t.id != id);
      mostrarTabla(tareas);
    })
    .catch(error => {
      console.error("Error:", error);
      alert("Error al borrar la tarea");
    });
  }

  // Cargar tareas al inicio
  window.onload = function() {
    listarTareas();
  };
</script>

</body>
</html>
```

Ejercicio 1: Gestión de Usuarios

API: `https://jsonplaceholder.typicode.com/users`

Requisitos:

- Botón **GET** para listar todos los usuarios en una tabla (ID, Nombre, Email, Ciudad)
- Botón **POST** para crear un nuevo usuario (pedir nombre y email con `prompt`)
- Botón **PUT** para actualizar el email de un usuario (pedir ID y nuevo email)
- Botón **DELETE** para borrar un usuario por ID
- Después de cada operación, actualizar la tabla

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Gestión de Usuarios</title>
</head>
<body>
  <h1>Gestión de Usuarios</h1>

  <button onclick="listarUsuarios()">GET - Listar Usuarios</button>
  <button onclick="crearUsuario()">POST - Crear Usuario</button>
  <button onclick="actualizarUsuario()">PUT - Actualizar Email</button>
  <button onclick="borrarUsuario()">DELETE - Borrar Usuario</button>

  <hr>
  <div id="resultado"></div>

  <script>
    const URL = "https://jsonplaceholder.typicode.com/users";
    let usuarios = [];

    // GET - Listar usuarios
    function listarUsuarios() {
      fetch(URL)
        .then(res => res.json())
        .then(data => {
          usuarios = data;
          mostrarTabla();
        })
        .catch(err => alert("Error: " + err));
    }

    // Mostrar tabla
    function mostrarTabla() {
      let html = "<table border='1'><tr><th>ID</th><th>Nombre</th><th>Email</th><th>Ciu-
dad</th></tr>";

      usuarios.forEach(user => {
        html += `<tr>
          <td>${user.id}</td>
```

```
        <td>${user.name}</td>
        <td>${user.email}</td>
        <td>${user.address.city}</td>
      </tr>`;
    });

    html += "</table>";
    document.getElementById("resultado").innerHTML = html;
}

// POST - Crear usuario
function crearUsuario() {
    const nombre = prompt("Nombre del usuario:");
    const email = prompt("Email del usuario:");

    if (!nombre || !email) {
      alert("Debes completar todos los campos");
      return;
    }

    const nuevoUsuario = {
      name: nombre,
      email: email,
      address: { city: "Madrid" }
    };

    fetch(URL, {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify(nuevoUsuario)
    })
    .then(res => res.json())
    .then(data => {
      alert("Usuario creado con ID: " + data.id);
      usuarios.unshift(data);
      mostrarTabla();
    })
    .catch(err => alert("Error: " + err));
}

// PUT - Actualizar email
function actualizarUsuario() {
    const id = prompt("ID del usuario a actualizar:");
    const nuevoEmail = prompt("Nuevo email:");

    if (!id || !nuevoEmail) {
      alert("Debes completar todos los campos");
      return;
    }
```

```
      fetch(URL + "/" + id, {
        method: 'PUT',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ email: nuevoEmail })
      })
      .then(res => res.json())
      .then(data => {
        alert("Email actualizado correctamente");
        const index = usuarios.findIndex(u => u.id == id);
        if (index !== -1) {
          usuarios[index].email = nuevoEmail;
          mostrarTabla();
        }
      })
      .catch(err => alert("Error: " + err));
    }

    // DELETE - Borrar usuario
    function borrarUsuario() {
      const id = prompt("ID del usuario a borrar:");

      if (!id) {
        alert("Debes introducir un ID");
        return;
      }

      fetch(URL + "/" + id, {
        method: 'DELETE'
      })
      .then(res => {
        alert("Usuario borrado correctamente");
        usuarios = usuarios.filter(u => u.id != id);
        mostrarTabla();
      })
      .catch(err => alert("Error: " + err));
    }
  </script>
</body>
</html>
```

Ejercicio 2: Gestión de Posts de Blog

API: `https://jsonplaceholder.typicode.com/posts`

Requisitos:

- Botón **GET** para listar los primeros 10 posts (ID, Título, Usuario ID)
- Botón **POST** para crear un nuevo post (pedir título y contenido)
- Botón **PUT** para actualizar el título de un post (pedir ID y nuevo título)

- Botón **DELETE** para borrar un post por ID
- Después de cada operación, actualizar la tabla

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Gestión de Posts</title>
</head>
<body>
  <h1>Gestión de Posts del Blog</h1>

  <button onclick="listarPosts()">GET - Listar Posts</button>
  <button onclick="crearPost()">POST - Crear Post</button>
  <button onclick="actualizarPost()">PUT - Actualizar Título</button>
  <button onclick="borrarPost()">DELETE - Borrar Post</button>

  <hr>
  <div id="resultado"></div>

  <script>
    const URL = "https://jsonplaceholder.typicode.com/posts";
    let posts = [];

    // GET - Listar posts
    function listarPosts() {
      fetch(URL + "?_limit=10")
        .then(res => res.json())
        .then(data => {
          posts = data;
          mostrarTabla();
        })
        .catch(err => alert("Error: " + err));
    }

    // Mostrar tabla
    function mostrarTabla() {
      let html = "<table border='1'><tr><th>ID</th><th>Usuario ID</th><th>Título</th></tr>";

      posts.forEach(post => {
        html += `<tr>
          <td>${post.id}</td>
          <td>${post.userId}</td>
          <td>${post.title}</td>
        </tr>`;
      });

      html += "</table>";
      document.getElementById("resultado").innerHTML = html;
    }
```

```
// POST - Crear post
function crearPost() {
  const titulo = prompt("Título del post:");
  const contenido = prompt("Contenido del post:");

  if (!titulo || !contenido) {
    alert("Debes completar todos los campos");
    return;
  }

  const nuevoPost = {
    userId: 1,
    title: titulo,
    body: contenido
  };

  fetch(URL, {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify(nuevoPost)
  })
  .then(res => res.json())
  .then(data => {
    alert("Post creado con ID: " + data.id);
    posts.unshift(data);
    mostrarTabla();
  })
  .catch(err => alert("Error: " + err));
}

// PUT - Actualizar título
function actualizarPost() {
  const id = prompt("ID del post a actualizar:");
  const nuevoTitulo = prompt("Nuevo título:");

  if (!id || !nuevoTitulo) {
    alert("Debes completar todos los campos");
    return;
  }

  fetch(URL + "/" + id, {
    method: 'PUT',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({ title: nuevoTitulo })
  })
  .then(res => res.json())
  .then(data => {
    alert("Título actualizado correctamente");
    const index = posts.findIndex(p => p.id == id);
```

```
        if (index !== -1) {
          posts[index].title = nuevoTitulo;
          mostrarTabla();
        }
      })
      .catch(err => alert("Error: " + err));
    }

    // DELETE - Borrar post
    function borrarPost() {
      const id = prompt("ID del post a borrar:");

      if (!id) {
        alert("Debes introducir un ID");
        return;
      }

      fetch(URL + "/" + id, {
        method: 'DELETE'
      })
      .then(res => {
        alert("Post borrado correctamente");
        posts = posts.filter(p => p.id != id);
        mostrarTabla();
      })
      .catch(err => alert("Error: " + err));
    }
  </script>
</body>
</html>
```

Ejercicio 3: Gestión de Comentarios

API: `https://jsonplaceholder.typicode.com/comments`

Requisitos:

- Botón **GET** para listar los primeros 15 comentarios (ID, Nombre, Email)
- Botón **POST** para crear un comentario (pedir nombre, email y texto)
- Botón **PUT** para actualizar el email de un comentario (pedir ID y nuevo email)
- Botón **DELETE** para borrar un comentario por ID

```
a <!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Gestión de Comentarios</title>
</head>
<body>
```

```
  <h1>Gestión de Comentarios</h1>

  <button onclick="listarComentarios()">GET - Listar Comentarios</button>
  <button onclick="crearComentario()">POST - Crear Comentario</button>
  <button onclick="actualizarComentario()">PUT - Actualizar Email</button>
  <button onclick="borrarComentario()">DELETE - Borrar Comentario</button>

  <hr>
  <div id="resultado"></div>

  <script>
    const URL = "https://jsonplaceholder.typicode.com/comments";
    let comentarios = [];

    // GET - Listar comentarios
    function listarComentarios() {
      fetch(URL + "?_limit=15")
        .then(res => res.json())
        .then(data => {
          comentarios = data;
          mostrarTabla();
        })
        .catch(err => alert("Error: " + err));
    }

    // Mostrar tabla
    function mostrarTabla() {
      let html = "<table border='1'><tr><th>ID</th><th>Nombre</th><th>Email</th><th>Comenta-
rio</th></tr>";

      comentarios.forEach(com => {
        html += `<tr>
          <td>${com.id}</td>
          <td>${com.name}</td>
          <td>${com.email}</td>
          <td>${com.body.substring(0, 50)}...</td>
        </tr>`;
      });

      html += "</table>";
      document.getElementById("resultado").innerHTML = html;
    }

    // POST - Crear comentario
    function crearComentario() {
      const nombre = prompt("Nombre:");
      const email = prompt("Email:");
      const texto = prompt("Texto del comentario:");

      if (!nombre || !email || !texto) {
```

```
    alert("Debes completar todos los campos");
    return;
  }

  const nuevoComentario = {
    postId: 1,
    name: nombre,
    email: email,
    body: texto
  };

  fetch(URL, {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify(nuevoComentario)
  })
  .then(res => res.json())
  .then(data => {
    alert("Comentario creado con ID: " + data.id);
    comentarios.unshift(data);
    mostrarTabla();
  })
  .catch(err => alert("Error: " + err));
}

// PUT - Actualizar email
function actualizarComentario() {
  const id = prompt("ID del comentario a actualizar:");
  const nuevoEmail = prompt("Nuevo email:");

  if (!id || !nuevoEmail) {
    alert("Debes completar todos los campos");
    return;
  }

  fetch(URL + "/" + id, {
    method: 'PUT',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({ email: nuevoEmail })
  })
  .then(res => res.json())
  .then(data => {
    alert("Email actualizado correctamente");
    const index = comentarios.findIndex(c => c.id == id);
    if (index !== -1) {
      comentarios[index].email = nuevoEmail;
      mostrarTabla();
    }
  })
  .catch(err => alert("Error: " + err));
```

```
    }

    // DELETE - Borrar comentario
    function borrarComentario() {
      const id = prompt("ID del comentario a borrar:");

      if (!id) {
        alert("Debes introducir un ID");
        return;
      }

      fetch(URL + "/" + id, {
        method: 'DELETE'
      })
      .then(res => {
        alert("Comentario borrado correctamente");
        comentarios = comentarios.filter(c => c.id != id);
        mostrarTabla();
      })
      .catch(err => alert("Error: " + err));
    }
  </script>
</body>
</html>
```

Ejercicio 4: Gestión de Álbumes de Fotos

API: `https://jsonplaceholder.typicode.com/albums`

Requisitos:

- Botón **GET** para listar los primeros 10 álbumes (ID, Usuario ID, Título)
- Botón **POST** para crear un álbum (pedir título)
- Botón **PUT** para actualizar el título de un álbum (pedir ID y nuevo título)
- Botón **DELETE** para borrar un álbum por ID

```
s <!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Gestión de Álbumes</title>
</head>
<body>
  <h1>Gestión de Álbumes</h1>

  <button onclick="listarAlbumes()">GET - Listar Álbumes</button>
  <button onclick="crearAlbum()">POST - Crear Álbum</button>
  <button onclick="actualizarAlbum()">PUT - Actualizar Título</button>
```

```
<button onclick="borrarAlbum()">DELETE - Borrar Álbum</button>

<hr>
<div id="resultado"></div>

<script>
  const URL = "https://jsonplaceholder.typicode.com/albums";
  let albumes = [];

  function listarAlbumes() {
    fetch(URL + "?_limit=10")
      .then(res => res.json())
      .then(data => {
        albumes = data;
        mostrarTabla();
      });
  }

  function mostrarTabla() {
    let html = "<table border='1'><tr><th>ID</th><th>Usuario ID</th><th>Título</th></tr>";
    albumes.forEach(album => {
      html += `<tr><td>${album.id}</td><td>${album.userId}</td><td>${album.title}</td></tr>`;
    });
    html += "</table>";
    document.getElementById("resultado").innerHTML = html;
  }

  function crearAlbum() {
    const titulo = prompt("Título del álbum:");
    if (!titulo) return;

    fetch(URL, {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ userId: 1, title: titulo })
    })
    .then(res => res.json())
    .then(data => {
      alert("Álbum creado con ID: " + data.id);
      albumes.unshift(data);
      mostrarTabla();
    });
  }

  function actualizarAlbum() {
    const id = prompt("ID del álbum:");
    const titulo = prompt("Nuevo título:");
    if (!id || !titulo) return;

    fetch(URL + "/" + id, {
```

```
        method: 'PUT',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ title: titulo })
      })
      .then(res => res.json())
      .then(data => {
        alert("Álbum actualizado");
        const index = albumes.findIndex(a => a.id == id);
        if (index !== -1) {
          albumes[index].title = titulo;
          mostrarTabla();
        }
      });
    }

    function borrarAlbum() {
      const id = prompt("ID del álbum a borrar:");
      if (!id) return;

      fetch(URL + "/" + id, { method: 'DELETE' })
      .then(res => {
        alert("Álbum borrado");
        albumes = albumes.filter(a => a.id != id);
        mostrarTabla();
      });
    }
  </script>
</body>
</html>
```

Ejercicio 5: Gestión de Tareas (TODO List)

API: `https://jsonplaceholder.typicode.com/todos`

Requisitos:

- Botón **GET** para listar las primeras 20 tareas (ID, Título, Completada)
- Botón **POST** para crear una tarea (pedir título)
- Botón **PUT** para marcar una tarea como completada (pedir ID)
- Botón **DELETE** para borrar una tarea por ID
- Mostrar tareas completadas en color verde y pendientes en rojo

```
s <!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Gestión de Tareas</title>
</head>
<body>
```

```
<h1>Lista de Tareas (TODO)</h1>

<button onclick="listarTareas()">GET - Listar Tareas</button>
<button onclick="crearTarea()">POST - Crear Tarea</button>
<button onclick="completarTarea()">PUT - Completar Tarea</button>
<button onclick="borrarTarea()">DELETE - Borrar Tarea</button>

<hr>
<div id="resultado"></div>

<script>
  const URL = "https://jsonplaceholder.typicode.com/todos";
  let tareas = [];

  // GET - Listar tareas
  function listarTareas() {
    fetch(URL + "?_limit=20")
      .then(res => res.json())
      .then(data => {
        tareas = data;
        mostrarTabla();
      })
      .catch(err => alert("Error: " + err));
  }

  // Mostrar tabla con colores
  function mostrarTabla() {
    let html = "<table border='1'>";
    html += "<tr><th>ID</th><th>Título</th><th>Estado</th></tr>";

    tareas.forEach(tarea => {
      const color = tarea.completed ? "green" : "red";
      const estado = tarea.completed ? "Completada" : "Pendiente";

      html += `<tr style='color: ${color}'>
        <td>${tarea.id}</td>
        <td>${tarea.title}</td>
        <td>${estado}</td>
      </tr>`;
    });

    html += "</table>";
    document.getElementById("resultado").innerHTML = html;
  }

  // POST - Crear tarea
  function crearTarea() {
    const titulo = prompt("Título de la tarea:");

    if (!titulo) {
```

```
    alert("Debes introducir un título");
    return;
  }

  const nuevaTarea = {
    userId: 1,
    title: titulo,
    completed: false
  };

  fetch(URL, {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify(nuevaTarea)
  })
  .then(res => res.json())
  .then(data => {
    alert("Tarea creada con ID: " + data.id);
    tareas.unshift(data);
    mostrarTabla();
  })
  .catch(err => alert("Error: " + err));
}

// PUT - Completar tarea
function completarTarea() {
  const id = prompt("ID de la tarea a completar:");

  if (!id) {
    alert("Debes introducir un ID");
    return;
  }

  fetch(URL + "/" + id, {
    method: 'PUT',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({ completed: true })
  })
  .then(res => res.json())
  .then(data => {
    alert("Tarea marcada como completada");
    const index = tareas.findIndex(t => t.id == id);
    if (index !== -1) {
      tareas[index].completed = true;
      mostrarTabla();
    }
  })
  .catch(err => alert("Error: " + err));
}
```

```
    // DELETE - Borrar tarea
    function borrarTarea() {
      const id = prompt("ID de la tarea a borrar:");

      if (!id) {
        alert("Debes introducir un ID");
        return;
      }

      fetch(URL + "/" + id, {
        method: 'DELETE'
      })
      .then(res => {
        alert("Tarea borrada correctamente");
        tareas = tareas.filter(t => t.id != id);
        mostrarTabla();
      })
      .catch(err => alert("Error: " + err));
    }
  </script>
</body>
</html>
```

Ejercicio 6: Gestión de Productos (Tienda)

API: `https://fakestoreapi.com/products`

Requisitos:

- Botón **GET** para listar los primeros 10 productos (ID, Título, Precio, Categoría)
- Botón **POST** para crear un producto (pedir título, precio y categoría)
- Botón **PUT** para actualizar el precio de un producto (pedir ID y nuevo precio)
- Botón **DELETE** para borrar un producto por ID

```
a <!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Gestión de Productos</title>
</head>
<body>
  <h1>Gestión de Productos de Tienda</h1>

  <button onclick="listarProductos()">GET - Listar Productos</button>
  <button onclick="crearProducto()">POST - Crear Producto</button>
  <button onclick="actualizarPrecio()">PUT - Actualizar Precio</button>
  <button onclick="borrarProducto()">DELETE - Borrar Producto</button>
```

```
<hr>
<div id="resultado"></div>

<script>
  const URL = "https://fakestoreapi.com/products";
  let productos = [];

  // GET - Listar productos
  function listarProductos() {
    fetch(URL + "?limit=10")
      .then(res => res.json())
      .then(data => {
        productos = data;
        mostrarTabla();
      })
      .catch(err => alert("Error: " + err));
  }

  // Mostrar tabla
  function mostrarTabla() {
    let html = "<table border='1'>";
    html += "<tr><th>ID</th><th>Título</th><th>Precio</th><th>Categoría</th></tr>";

    productos.forEach(prod => {
      html += `<tr>
        <td>${prod.id}</td>
        <td>${prod.title}</td>
        <td>${prod.price} €</td>
        <td>${prod.category}</td>
      </tr>`;
    });

    html += "</table>";
    document.getElementById("resultado").innerHTML = html;
  }

  // POST - Crear producto
  function crearProducto() {
    const titulo = prompt("Título del producto:");
    const precio = prompt("Precio del producto:");
    const categoria = prompt("Categoría:");

    if (!titulo || !precio || !categoria) {
      alert("Debes completar todos los campos");
      return;
    }

    const nuevoProducto = {
      title: titulo,
      price: parseFloat(precio),
```

```
      category: categoria,
      description: "Producto nuevo",
      image: "https://i.pravatar.cc"
    };

    fetch(URL, {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify(nuevoProducto)
    })
    .then(res => res.json())
    .then(data => {
      alert("Producto creado con ID: " + data.id);
      productos.unshift(data);
      mostrarTabla();
    })
    .catch(err => alert("Error: " + err));
  }

  // PUT - Actualizar precio
  function actualizarPrecio() {
    const id = prompt("ID del producto:");
    const nuevoPrecio = prompt("Nuevo precio:");

    if (!id || !nuevoPrecio) {
      alert("Debes completar todos los campos");
      return;
    }

    fetch(URL + "/" + id, {
      method: 'PUT',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ price: parseFloat(nuevoPrecio) })
    })
    .then(res => res.json())
    .then(data => {
      alert("Precio actualizado correctamente");
      const index = productos.findIndex(p => p.id == id);
      if (index !== -1) {
        productos[index].price = parseFloat(nuevoPrecio);
        mostrarTabla();
      }
    })
    .catch(err => alert("Error: " + err));
  }

  // DELETE - Borrar producto
  function borrarProducto() {
    const id = prompt("ID del producto a borrar:");
```

```
      if (!id) {
        alert("Debes introducir un ID");
        return;
      }

      fetch(URL + "/" + id, {
        method: 'DELETE'
      })
      .then(res => res.json())
      .then(data => {
        alert("Producto borrado correctamente");
        productos = productos.filter(p => p.id != id);
        mostrarTabla();
      })
      .catch(err => alert("Error: " + err));
    }
  </script>
</body>
</html>
```

Ejercicio 7: Gestión de Empleados

API: `https://dummy.restapiexample.com/api/v1/employees`

Requisitos:

- Botón **GET** para listar todos los empleados (ID, Nombre, Salario, Edad)
- Botón **POST** para crear un empleado (pedir nombre, salario y edad)
- Botón **PUT** para actualizar el salario de un empleado (pedir ID y nuevo salario)
- Botón **DELETE** para borrar un empleado por ID

```
a <!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Gestión de Empleados</title>
</head>
<body>
  <h1>Gestión de Empleados</h1>

  <button onclick="listarEmpleados()">GET - Listar Empleados</button>
  <button onclick="crearEmpleado()">POST - Crear Empleado</button>
  <button onclick="actualizarSalario()">PUT - Actualizar Salario</button>
  <button onclick="borrarEmpleado()">DELETE - Borrar Empleado</button>

  <hr>
  <div id="resultado"></div>

  <script>
```

```
const URL = "https://dummy.restapiexample.com/api/v1";
let empleados = [];

// GET - Listar empleados
function listarEmpleados() {
  fetch(URL + "/employees")
    .then(res => res.json())
    .then(response => {
      empleados = response.data;
      mostrarTabla();
    })
    .catch(err => alert("Error: " + err));
}

// Mostrar tabla
function mostrarTabla() {
  let html = "<table border='1'>";
  html += "<tr><th>ID</th><th>Nombre</th><th>Salario</th><th>Edad</th></tr>";

  empleados.forEach(emp => {
    html += `<tr>
      <td>${emp.id}</td>
      <td>${emp.employee_name}</td>
      <td>${emp.employee_salary} €</td>
      <td>${emp.employee_age}</td>
    </tr>`;
  });

  html += "</table>";
  document.getElementById("resultado").innerHTML = html;
}

// POST - Crear empleado
function crearEmpleado() {
  const nombre = prompt("Nombre del empleado:");
  const salario = prompt("Salario:");
  const edad = prompt("Edad:");

  if (!nombre || !salario || !edad) {
    alert("Debes completar todos los campos");
    return;
  }

  const nuevoEmpleado = {
    name: nombre,
    salary: salario,
    age: edad
  };

  fetch(URL + "/create", {
```

```
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify(nuevoEmpleado)
    })
    .then(res => res.json())
    .then(response => {
      alert("Empleado creado con ID: " + response.data.id);
      listarEmpleados(); // Recargar lista
    })
    .catch(err => alert("Error: " + err));
}

// PUT - Actualizar salario
function actualizarSalario() {
  const id = prompt("ID del empleado:");
  const nuevoSalario = prompt("Nuevo salario:");

  if (!id || !nuevoSalario) {
    alert("Debes completar todos los campos");
    return;
  }

  fetch(URL + "/update/" + id, {
    method: 'PUT',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({ salary: nuevoSalario })
  })
  .then(res => res.json())
  .then(response => {
    alert("Salario actualizado correctamente");
    listarEmpleados(); // Recargar lista
  })
  .catch(err => alert("Error: " + err));
}

// DELETE - Borrar empleado
function borrarEmpleado() {
  const id = prompt("ID del empleado a borrar:");

  if (!id) {
    alert("Debes introducir un ID");
    return;
  }

  fetch(URL + "/delete/" + id, {
    method: 'DELETE'
  })
  .then(res => res.json())
  .then(response => {
    alert("Empleado borrado correctamente");
```

```
        listarEmpleados(); // Recargar lista
      })
      .catch(err => alert("Error: " + err));
    }
  </script>
</body>
</html>
```

Ejercicio 8: Gestión de Libros

API de ejemplo (simulada con JSONPlaceholder posts)

Requisitos:

- Botón **GET** para listar libros (ID, Título, Autor)
- Botón **POST** para añadir un libro (pedir título y autor)
- Botón **PUT** para actualizar el título de un libro (pedir ID y nuevo título)
- Botón **DELETE** para borrar un libro por ID
- Filtrar libros por autor usando un input de búsqueda

```
a <!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Gestión de Libros</title>
</head>
<body>
  <h1>Gestión de Biblioteca</h1>

  <button onclick="listarLibros()">GET - Listar Libros</button>
  <button onclick="crearLibro()">POST - Añadir Libro</button>
  <button onclick="actualizarLibro()">PUT - Actualizar Título</button>
  <button onclick="borrarLibro()">DELETE - Borrar Libro</button>

  <hr>

  <label>Buscar por autor: </label>
  <input type="text" id="buscarAutor" onkeyup="filtrarPorAutor()" placeholder="Escribe un au-
tor...">

  <hr>
  <div id="resultado"></div>

  <script>
    const URL = "https://jsonplaceholder.typicode.com/posts";
    let libros = [];
    let librosFiltrados = [];

    // GET - Listar libros
    function listarLibros() {
```

```
  fetch(URL + "?_limit=15")
    .then(res => res.json())
    .then(data => {
      // Simulamos que el userId es el autor
      libros = data.map(post => ({
        id: post.id,
        titulo: post.title,
        autor: "Autor " + post.userId
      }));
      librosFiltrados = libros;
      mostrarTabla();
    })
    .catch(err => alert("Error: " + err));
}

// Mostrar tabla
function mostrarTabla() {
  let html = "<table border='1'>";
  html += "<tr><th>ID</th><th>Título</th><th>Autor</th></tr>";

  librosFiltrados.forEach(libro => {
    html += `<tr>
      <td>${libro.id}</td>
      <td>${libro.titulo}</td>
      <td>${libro.autor}</td>
    </tr>`;
  });

  html += "</table>";
  document.getElementById("resultado").innerHTML = html;
}

// Filtrar por autor en tiempo real
function filtrarPorAutor() {
  const textoBusqueda = document.getElementById("buscarAutor").value.toLowerCase();

  if (textoBusqueda === "") {
    librosFiltrados = libros;
  } else {
    librosFiltrados = libros.filter(libro =>
      libro.autor.toLowerCase().includes(textoBusqueda)
    );
  }

  mostrarTabla();
}

// POST - Crear libro
function crearLibro() {
  const titulo = prompt("Título del libro:");
```

```
  const autor = prompt("Autor del libro:");

  if (!titulo || !autor) {
    alert("Debes completar todos los campos");
    return;
  }

  const nuevoLibro = {
    userId: 1,
    title: titulo
  };

  fetch(URL, {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify(nuevoLibro)
  })
  .then(res => res.json())
  .then(data => {
    alert("Libro creado con ID: " + data.id);
    const libro = {
      id: data.id,
      titulo: titulo,
      autor: autor
    };
    libros.unshift(libro);
    librosFiltrados = libros;
    mostrarTabla();
  })
  .catch(err => alert("Error: " + err));
}

// PUT - Actualizar libro
function actualizarLibro() {
  const id = prompt("ID del libro:");
  const nuevoTitulo = prompt("Nuevo título:");

  if (!id || !nuevoTitulo) {
    alert("Debes completar todos los campos");
    return;
  }

  fetch(URL + "/" + id, {
    method: 'PUT',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({ title: nuevoTitulo })
  })
  .then(res => res.json())
  .then(data => {
    alert("Libro actualizado correctamente");
```

```
        const index = libros.findIndex(l => l.id == id);
        if (index !== -1) {
          libros[index].titulo = nuevoTitulo;
          librosFiltrados = libros;
          mostrarTabla();
        }
      })
      .catch(err => alert("Error: " + err));
    }

    // DELETE - Borrar libro
    function borrarLibro() {
      const id = prompt("ID del libro a borrar:");

      if (!id) {
        alert("Debes introducir un ID");
        return;
      }

      fetch(URL + "/" + id, {
        method: 'DELETE'
      })
      .then(res => {
        alert("Libro borrado correctamente");
        libros = libros.filter(l => l.id != id);
        librosFiltrados = libros;
        mostrarTabla();
      })
      .catch(err => alert("Error: " + err));
    }
  </script>
</body>
</html>
```

Ejercicio 9: Gestión de Clientes

API: `https://jsonplaceholder.typicode.com/users`

Requisitos:

- Botón **GET** para listar clientes (ID, Nombre, Email, Teléfono, Empresa)
- Botón **POST** para crear un cliente (pedir nombre, email y teléfono)
- Botón **PUT** para actualizar el teléfono de un cliente (pedir ID y nuevo teléfono)
- Botón **DELETE** para borrar un cliente por ID
- Contador de clientes totales

```
a <!DOCTYPE html>
<html lang="es">
```

```
<head>
  <meta charset="UTF-8">
  <title>Gestión de Clientes</title>
</head>
<body>
  <h1>Gestión de Clientes</h1>

  <button onclick="listarClientes()">GET - Listar Clientes</button>
  <button onclick="crearCliente()">POST - Crear Cliente</button>
  <button onclick="actualizarTelefono()">PUT - Actualizar Teléfono</button>
  <button onclick="borrarCliente()">DELETE - Borrar Cliente</button>

  <hr>

  <p id="contador"></p>

  <hr>
  <div id="resultado"></div>

  <script>
    const URL = "https://jsonplaceholder.typicode.com/users";
    let clientes = [];

    // GET - Listar clientes
    function listarClientes() {
      fetch(URL)
        .then(res => res.json())
        .then(data => {
          clientes = data;
          actualizarContador();
          mostrarTabla();
        })
        .catch(err => alert("Error: " + err));
    }

    // Actualizar contador
    function actualizarContador() {
      document.getElementById("contador").innerHTML =
        "<strong>Total de clientes: " + clientes.length + "</strong>";
    }

    // Mostrar tabla
    function mostrarTabla() {
      let html = "<table border='1'>";
      html += "<tr><th>ID</th><th>Nombre</th><th>Email</th><th>Teléfono</th><th>Em-
presa</th></tr>";

      clientes.forEach(cliente => {
        html += `<tr>
          <td>${cliente.id}</td>
```

```
        <td>${cliente.name}</td>
        <td>${cliente.email}</td>
        <td>${cliente.phone}</td>
        <td>${cliente.company.name}</td>
      </tr>`;
    });

    html += "</table>";
    document.getElementById("resultado").innerHTML = html;
}

// POST - Crear cliente
function crearCliente() {
    const nombre = prompt("Nombre del cliente:");
    const email = prompt("Email:");
    const telefono = prompt("Teléfono:");

    if (!nombre || !email || !telefono) {
        alert("Debes completar todos los campos");
        return;
    }

    const nuevoCliente = {
        name: nombre,
        email: email,
        phone: telefono,
        company: { name: "Empresa S.L." }
    };

    fetch(URL, {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify(nuevoCliente)
    })
    .then(res => res.json())
    .then(data => {
        alert("Cliente creado con ID: " + data.id);
        clientes.unshift(data);
        actualizarContador();
        mostrarTabla();
    })
    .catch(err => alert("Error: " + err));
}

// PUT - Actualizar teléfono
function actualizarTelefono() {
    const id = prompt("ID del cliente:");
    const nuevoTelefono = prompt("Nuevo teléfono:");

    if (!id || !nuevoTelefono) {
```

```
        alert("Debes completar todos los campos");
        return;
      }

      fetch(URL + "/" + id, {
        method: 'PUT',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ phone: nuevoTelefono })
      })
      .then(res => res.json())
      .then(data => {
        alert("Teléfono actualizado correctamente");
        const index = clientes.findIndex(c => c.id == id);
        if (index !== -1) {
          clientes[index].phone = nuevoTelefono;
          mostrarTabla();
        }
      })
      .catch(err => alert("Error: " + err));
    }

    // DELETE - Borrar cliente
    function borrarCliente() {
      const id = prompt("ID del cliente a borrar:");

      if (!id) {
        alert("Debes introducir un ID");
        return;
      }

      fetch(URL + "/" + id, {
        method: 'DELETE'
      })
      .then(res => {
        alert("Cliente borrado correctamente");
        clientes = clientes.filter(c => c.id != id);
        actualizarContador();
        mostrarTabla();
      })
      .catch(err => alert("Error: " + err));
    }
  </script>
</body>
</html>
```

Ejercicio 10: Gestión de Pedidos

API: `https://jsonplaceholder.typicode.com/posts` (simulando pedidos)

Requisitos:

- Botón **GET** para listar pedidos (ID, Cliente ID, Descripción, Estado)
- Botón **POST** para crear un pedido (pedir descripción)
- Botón **PUT** para cambiar el estado de un pedido a "Entregado" (pedir ID)
- Botón **DELETE** para cancelar un pedido (borrar por ID)
- Estados: "Pendiente" (por defecto), "Entregado" (después de PUT)

```
a <!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Gestión de Pedidos</title>
</head>
<body>
  <h1>Gestión de Pedidos</h1>

  <button onclick="listarPedidos()">GET - Listar Pedidos</button>
  <button onclick="crearPedido()">POST - Crear Pedido</button>
  <button onclick="entregarPedido()">PUT - Marcar como Entregado</button>
  <button onclick="cancelarPedido()">DELETE - Cancelar Pedido</button>

  <hr>
  <div id="resultado"></div>

  <script>
    const URL = "https://jsonplaceholder.typicode.com/posts";
    let pedidos = [];

    function listarPedidos() {
      fetch(URL + "?_limit=12")
        .then(res => res.json())
        .then(data => {
          pedidos = data.map(post => ({
            id: post.id,
            clienteId: post.userId,
            descripcion: post.title,
            estado: "Pendiente"
          }));
          mostrarTabla();
        });
    }

    function mostrarTabla() {
      let html = "<table border='1'>";
      html += "<tr><th>ID</th><th>Cliente ID</th><th>Descripción</th><th>Estado</th></tr>";

      pedidos.forEach(pedido => {
        const color = pedido.estado === "Entregado" ? "green" : "orange";
        html += `<tr style='color: ${color}'>
          <td>${pedido.id}</td>
          <td>${pedido.clienteId}</td>
```

```
        <td>${pedido.descripcion}</td>
        <td><strong>${pedido.estado}</strong></td>
      </tr>`;
    });

    html += "</table>";
    document.getElementById("resultado").innerHTML = html;
}

function crearPedido() {
    const descripcion = prompt("Descripción del pedido:");
    if (!descripcion) return;

    fetch(URL, {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ userId: 1, title: descripcion })
    })
    .then(res => res.json())
    .then(data => {
      alert("Pedido creado con ID: " + data.id);
      pedidos.unshift({
        id: data.id,
        clienteId: 1,
        descripcion: descripcion,
        estado: "Pendiente"
      });
      mostrarTabla();
    });
}

function entregarPedido() {
    const id = prompt("ID del pedido a marcar como entregado:");
    if (!id) return;

    fetch(URL + "/" + id, {
      method: 'PUT',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ estado: "Entregado" })
    })
    .then(res => res.json())
    .then(data => {
      alert("Pedido marcado como entregado");
      const index = pedidos.findIndex(p => p.id == id);
      if (index !== -1) {
        pedidos[index].estado = "Entregado";
        mostrarTabla();
      }
    });
}
```

```
    function cancelarPedido() {
      const id = prompt("ID del pedido a cancelar:");
      if (!id) return;

      fetch(URL + "/" + id, { method: 'DELETE' })
      .then(res => {
        alert("Pedido cancelado");
        pedidos = pedidos.filter(p => p.id != id);
        mostrarTabla();
      });
    }
  </script>
</body>
</html>
```

Metodos:

```
const URL = "TU_API_AQUI";
let datos = [];

// GET
function listar() {
 fetch(URL)
   .then(res => res.json())
   .then(data => {
     datos = data;
     mostrarTabla();
   });
}

// POST
function crear() {
 const valor = prompt("Valor:");

 fetch(URL, {
   method: 'POST',
   headers: { 'Content-Type': 'application/json' },
   body: JSON.stringify({ campo: valor })
 })
 .then(res => res.json())
 .then(data => {
   alert("Creado con ID: " + data.id);
   datos.unshift(data);
   mostrarTabla();
 });
}

// PUT
function actualizar() {
 const id = prompt("ID:");
```

```
  const nuevoValor = prompt("Nuevo valor:");

  fetch(URL + "/" + id, {
    method: 'PUT',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({ campo: nuevoValor })
  })
  .then(res => res.json())
  .then(data => {
    alert("Actualizado");
    const index = datos.findIndex(d => d.id == id);
    if (index !== -1) {
      datos[index].campo = nuevoValor;
      mostrarTabla();
    }
  });
}

// DELETE
function borrar() {
  const id = prompt("ID:");

  fetch(URL + "/" + id, { method: 'DELETE' })
  .then(res => {
    alert("Borrado");
    datos = datos.filter(d => d.id != id);
    mostrarTabla();
  });
}

// MOSTRAR TABLA
function mostrarTabla() {
  let html = "<table border='1'><tr><th>ID</th><th>Campo</th></tr>";
  datos.forEach(item => {
    html += `<tr><td>${item.id}</td><td>${item.campo}</td></tr>`;
  });
  html += "</table>";
  document.getElementById("resultado").innerHTML = html;
}
```