Actividad 10 Javascript

Crea una función llamada `multiplicarArray()` que reciba como parámetro un array de números enteros. La función debe encontrar **el número menor y el número mayor** dentro del array, y **retornar el resultado de multiplicarlos entre sí**.

Objetivo del ejercicio:

- Practicar el uso de funciones en JavaScript.
- Manipular arrays y aplicar métodos como `Math.min()` y `Math.max()`.
- Mostrar el resultado de forma dinámica en la página web.
- Aplicar estilos CSS básicos al formulario y al resultado.

Requisitos funcionales:

1. La página debe permitir al usuario **ingresar un array de números** (por ejemplo, separados por comas).
2. Al hacer clic en un botón, debe ejecutarse la función `multiplicarArray()`.
3. El resultado (la multiplicación del menor y mayor número del array) debe mostrarse en pantalla.

Orientaciones para resolverlo:

1. **Leer la entrada del usuario** desde un campo de texto (`input` o `textarea`) y convertirla en un array de números.
    - Pista: Usa `.split(',')` para separar los valores.
    - Usa `.map(Number)` para convertir los elementos a números.
2. **Encontrar el número mayor y menor** usando `Math.max(...array)` y `Math.min(...array)`.
3. **Multiplicar los valores** obtenidos y retornar el resultado.
4. **Mostrar el resultado en el HTML** dinámicamente con JavaScript (por ejemplo, usando `innerText` o `textContent`).

✅ Resultado esperado (ejemplo):

Entrada del usuario:
`3, 9, 1, 5`

Salida mostrada:
`El número menor es 1, el mayor es 9, y su multiplicación es 9.`

✅ Tareas de CSS que deben realizar:

1. **Diseñar un contenedor centralizado**
    - Crear una caja (`div`) con fondo blanco, bordes redondeados y sombra suave.
    - Centrarla horizontalmente en la página.
2. **Estilizar el campo de entrada (`input[type="text"]`)**
    - Aplicar un ancho adecuado (por ejemplo, 80% del contenedor).
    - Usar padding interno y bordes redondeados.
    - Cambiar el color del borde al hacer foco.
3. **Estilizar el botón**
    - Colores contrastantes (fondo verde, texto blanco).
    - Efecto `hover` que cambie ligeramente el color de fondo.
    - Bordes redondeados y cursor tipo puntero.
4. **Estilizar el área de resultados (`#resultado`)**
    - Dar margen superior.
    - Usar un tipo de letra en **negrita** y color gris oscuro.
    - Asegurarse de que el resultado se muestre de forma destacada.
5. **Estilos generales**
    - Aplicar una fuente legible como `Arial` o `Segoe UI`.
    - Usar un color de fondo claro para el `body`.
    - Añadir padding general para evitar que el contenido quede pegado a los bordes.

Retos opcionales de CSS

- Agregar una animación de entrada al contenedor (`@keyframes fadeIn`).
- Usar media queries para adaptar el diseño a pantallas pequeñas.
- Probar con variables CSS para los colores principales.
- Crear un diseño más moderno con `Flexbox` o `Grid`.

¿Qué es una animación de entrada?

Es un efecto que se aplica cuando un elemento aparece por primera vez en pantalla. Por ejemplo:

- Que el contenedor se desvanezca (`fade in`).
- Que suba ligeramente desde abajo mientras aparece (`slide up`).
- Que aumente su opacidad y tamaño de forma gradual.

Ejemplo:

```
@keyframes fadeIn {
  from {
    opacity: 0;
    transform: translateY(20px);
  }
  to {
    opacity: 1;
    transform: translateY(0);
  }
}

.contenedor {
  animation: fadeIn 0.8s ease-in-out;
}
```

Cuando la página se carga, el contenedor principal aparecerá **con una suave transición** desde una posición un poco más baja, mientras su opacidad pasa de 0 a 1.

Esto da un toque visual muy profesional sin necesidad de usar JavaScript.