MEMORIA LM

# Ejercicio1:

## Calculadora.js

Este archivo define la clase Calculadora, que contiene los métodos para realizar operaciones matemáticas básicas y algunas operaciones extendidas como porcentaje, cuadrado, y raíz cuadrada.

* **sumar, restar, multiplicar, dividir**: Realizan las operaciones matemáticas básicas.
* **porcentaje**: Calcula el porcentaje de un número.
* **cuadrado**: Retorna el cuadrado de un número.
* **raizCuadrada**: Retorna la raíz cuadrada de un número.

## Display.js

Este archivo maneja la interfaz de usuario de la calculadora, mostrando los números y resultados en el display y vinculando las entradas de usuario con las operaciones matemáticas.

* **Constructor**: Inicializa la calculadora con displays vacíos y crea una instancia de **Calculadora**.
* **borrar()**: Borra el último dígito del valor actual.
* **borrarTodo()**: Reinicia todos los valores de la calculadora.
* **agregarNumero(numero)**: Añade un número al valor actual que se está introduciendo.
* **imprimirValores()**: Actualiza los displays de la calculadora con los valores actuales.
* **calcular()**: Ejecuta la operación matemática seleccionada.
* **computar(tipo)**: Define el tipo de operación a realizar y ejecuta el cálculo.

## Index.js

Este archivo vincula los eventos de los botones con las funcionalidades definidas en **Display.js**.

* **DOMContentLoaded**: Asegura que el código JavaScript se ejecuta después de que el contenido HTML ha sido completamente cargado.
* Establece los manejadores de eventos para los botones de la calculadora.

## Index.css

Este archivo personaliza los estilos básicos proporcionados por Bootstrap y define estéticas específicas para la calculadora, como colores, márgenes y tipografías.

* **.container**: Aplica un ancho máximo, centrado automáticamente con margen, un poco de padding para espacio interno, un sombreado suave para dar profundidad, y un color de fondo claro.
* **.calculadora**: Añade padding adicional dentro del contenedor específico de la calculadora para espacio entre los bordes y los botones.
* **.display**: Aplica un margen inferior para separar visualmente el display de los botones.
* **.button-row**: Usado para cualquier fila adicional de botones, añadiendo margen inferior para separación.
* **.btn**: Ajusta los márgenes entre botones y aumenta el tamaño de fuente para hacer los botones más legibles y fáciles de presionar, especialmente en dispositivos táctiles.

## Index.html

Este archivo es la base de la estructura de la página web para la calculadora. Utiliza HTML para organizar los elementos visuales y conecta los recursos de CSS y JavaScript necesarios para el estilo y la funcionalidad.

* **Doctype y HTML**: Define el tipo de documento y la versión de HTML.
* **Head**: Contiene metadatos, enlaces a hojas de estilo, y scripts.
  + **Charset UTF-8**: Especifica la codificación de caracteres para el contenido del documento.
  + **Viewport**: Asegura que la página es responsive, adaptándose al ancho del dispositivo.
  + **Title**: El título del documento que aparece en la pestaña del navegador.
  + **Links a CSS y Google Fonts**: Importa Bootstrap para los estilos y una fuente específica de Google Fonts.
  + **Scripts**: Vincula los archivos JavaScript necesarios para la funcionalidad de la calculadora, con **defer** para que se carguen después del contenido HTML.
* **Body**:
  + **Container**: Un div que usa la clase de Bootstrap **container**, que centra y contiene el contenido principal de la página.
  + **Calculadora**: Dentro del contenedor, este div agrupa todos los elementos de la calculadora.
  + **Display**: Muestra los valores ingresados y los resultados de las operaciones.
  + **Botones**: Se organizan en filas (**row**) y columnas (**col-**) utilizando las utilidades de grid de Bootstrap, facilitando un diseño responsivo y adaptable.

# Ejercicio2:

Index.html: Este archivo es la estructura fundamental de la página web. Define la disposición del contenido que verán los usuarios y enlaza los recursos externos como CSS y JavaScript que mejoran la funcionalidad y la apariencia de la página.

* **<!DOCTYPE html>**: Declara que este documento es HTML5.
* **<html lang="es">**: Establece el idioma del contenido de la página en español.
* **<head>**: Contiene metadatos y enlaces a recursos externos.
  + **<meta charset="UTF-8">**: Especifica la codificación de caracteres para el contenido del documento.
  + **<meta name="viewport" content="width=device-width, initial-scale=1.0">**: Asegura que la página es responsive y se ve bien en todos los dispositivos.
  + **<link rel="stylesheet" href="styles.css">**: Enlaza el archivo de estilos CSS externo que controla la apariencia visual de la página.
* **<body>**: Contiene todos los elementos visibles de la página web.
  + **<h1>**: Título de la página.
  + **<select id="provinceSelector">**: Un menú desplegable para seleccionar una provincia. Las opciones se cargarán dinámicamente mediante JavaScript.
  + **<button id="fetchData">**: Botón que el usuario puede pulsar para cargar los datos meteorológicos.
  + **<div id="weatherContainer">**: Contenedor donde se mostrarán los datos meteorológicos.
  + **<script src="script.js">**: Enlaza el archivo JavaScript que añade la funcionalidad dinámica a la página.

## Styles.css:

Este archivo define los estilos visuales de la página, como colores, márgenes, fuentes y otros aspectos del diseño.

* **Estilos generales (body, h1)**: Define la tipografía, color de fondo, alineación y otros estilos básicos.
* **Estilos de formularios (select, button)**: Ajusta la apariencia de los elementos de formulario para que sean más atractivos y coherentes.
* **Estilos interactivos (button:hover)**: Añade un efecto visual cuando el usuario pasa el cursor sobre el botón.
* **Estilos de contenedores de datos (weather-report, #weatherContainer)**: Asegura que la presentación de los datos meteorológicos sea clara y organizada.

## Script.js

Este archivo añade la interactividad a la página, permitiendo cargar y mostrar datos meteorológicos dinámicamente según la provincia seleccionada.

* **Inicialización y carga de datos**: Al cargar la página, el script realiza una petición a la API para obtener una lista de provincias, que luego se añaden al menú desplegable.
* **Interacción del usuario**: Cuando se selecciona una provincia y se hace clic en el botón, se carga la información meteorológica para esa provincia específica.
* **Manejo de errores**: Se incluyen capturas de errores para asegurar que cualquier fallo en la carga de datos se maneje adecuadamente, informando al usuario de posibles problemas.