Importante sobre esta relación de ejercicios

Las actividades de esta relación de ejercicios se definen como ejercicios de práctica y desarrollo de competencias que no son calificables dentro del sistema de evaluación. Estas actividades tienen como propósito principal proporcionar retroalimentación valiosa y apoyar el proceso de aprendizaje, pero no contribuyen al cálculo de la nota de los Resultados de Aprendizaje (RA) y Criterios de Evaluación (CE) vinculados a esta unidad didáctica.

Estas actividades se implementan con los siguientes objetivos fundamentales:

- Proporcionar un espacio seguro para cometer errores y aprender de ellos.
- Permitir la práctica repetitiva de habilidades sin presión evaluativa.
- Ofrecer retroalimentación detallada y personalizada para mejorar el desempeño.
- Favorecer la autoevaluación y la reflexión sobre el propio aprendizaje.
- Preparar al estudiante para las actividades calificativas posteriores.

Es crucial entender que, aunque estas actividades no influyen directamente en la calificación, constituyen una parte esencial del proceso formativo. El compromiso con estas tareas se reflejará indirectamente en el desempeño durante las evaluaciones calificadas, ya que permiten consolidar los conocimientos y desarrollar las competencias necesarias para alcanzar los resultados de aprendizaje establecidos.

La participación activa en estas actividades no calificables es altamente recomendable para obtener el máximo beneficio del proceso educativo y asegurar el desarrollo adecuado de las competencias profesionales que se pretenden alcanzar.

Esta relación de ejercicios está diseñada específicamente para contribuir al desarrollo y logro del siguiente Resultado de Aprendizaje (RA) y sus Criterios de Evaluación (CE) vinculados:

RA 3. Accede y manipula documentos web utilizando lenguajes de script de cliente.

- CE3a: Se han identificado y clasificado los lenguajes de script de cliente relacionados con la web y sus diferentes versiones y estándares.
- CE3b: Se ha identificado la sintaxis básica de los lenguajes de script de cliente.
- CE3c: Se han utilizado métodos para la selección y acceso de los diferentes elementos de un documento web.
- CE3d: Se han creado y modificado elementos de documentos web.
- CE3e: Se han eliminado elementos de documentos web.
- CE3f: Se han realizado modificaciones sobre los estilos de un documento web.

Se recuerda al alumnado que **no está permitido el uso de herramientas de inteligencia artificial (IA) para la realización total o parcial de este trabajo**.

ÍNDICE DE EJERCICIOS

Ejercicios básicos 3
Ejercicios de DOM 4
Ejercicios de validación de formularios 31

EJERCICIOS BÁSICOS

Para los ejercicios básicos inserta el código JavaScript en el mismo fichero index.html del ejercicio.

Ejercicio 01: Declara una variable llamada nombre y asígnale tu nombre. Luego, declara una constante llamada edad y asígnale tu edad. Muestra ambos valores en el documento web. Deberás incrustar ese texto en un párrafo y mediante CSS cambiarle el color y el tamaño de letra.

Ejercicio 02: Declara dos variables llamadas a y b, asígnales los valores 12 y 5, respectivamente. Realiza la suma, resta, producto y división de ambos, calcula también el resto de dividir uno por el otro, y muestra el resultado en el documento web. Deberás incrustar los resultados un párrafo cada uno y mediante CSS cambiarle varias propiedades a cada operación.

Ejercicio 03: Declara una variable llamada edad e inicialízala con un número. Muestra en el documento web si la edad corresponde a una persona mayor de edad o no. Mediante CSS cambia propiedades de ese párrafo teniendo estilos diferentes si se es mayor o menor de edad.

Ejercicio 04: Declara una variable llamada mes con un valor numérico entre 1 y 12. Usa un switch para mostrar en un párrafo el nombre del mes correspondiente. Si el mes no es válido deberá mostrar un error. Mediante CSS cambia propiedades de ese párrafo.

Ejercicio 05: Muestra los números del 1 al 10 utilizando un bucle while, un bucle do while y un bucle for. Mediante CSS cambia propiedades de ese párrafo. Para cada bucle crea un estilo diferente.

Ejercicio 06: Utilizando un bucle genera automáticamente los 6 encabezados disponible en HTML5. Para cada encabezado crea un estilo diferente.

EJERCICIOS DE DOM

A partir de aquí deberás crear el código JavaScript de cada ejercicio en un fichero aparte en su carpeta correspondiente.

Ejercicio 07: En una página web crea un div con un nombre y mediante JavaScript cambia el texto de ese div. Deberás hacerlo mediante un botón que al hacerle clic cambie el texto.

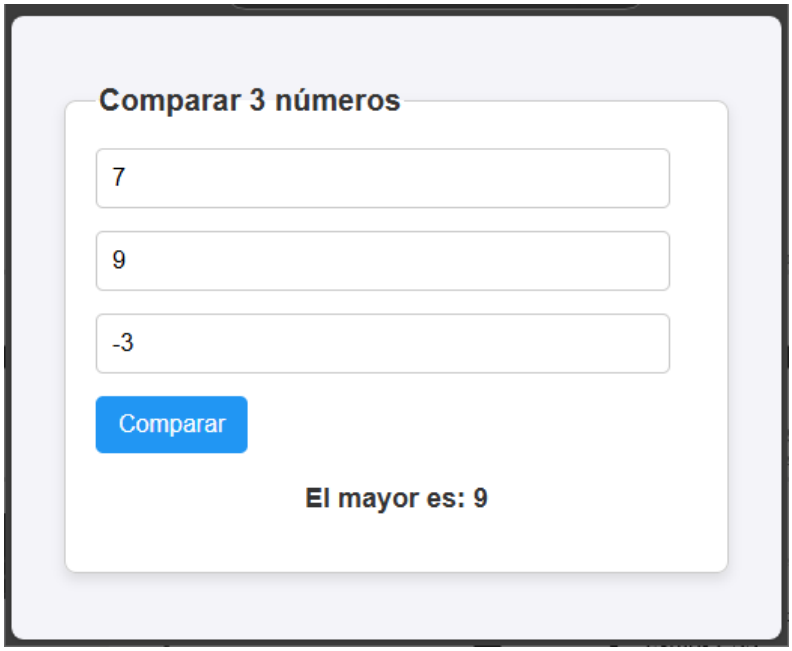
Ejercicio 08: Crea un párrafo con un texto. Mediante JavaScript cambia el texto de ese párrafo y además cambia su color de texto a rojo.

Ejercicio 09: Crea una aplicación web sencilla que permita al usuario introducir tres números y, al presionar un botón, determine cuál de los tres números es el mayor. La funcionalidad debe implementarse utilizando JavaScript, mientras que el diseño del formulario y la presentación del resultado se manejarán mediante HTML y CSS.

Requisitos:

- Estructura HTML:
  - El documento HTML debe contener un formulario que incluya:
  - Tres campos de entrada (<input>) de tipo number, donde el usuario podrá ingresar los números a comparar.
  - Un botón (<button>) que, al ser pulsado, ejecutará la función de comparación.
  - Un elemento de texto (puede ser un o <p>) donde se mostrará el resultado de la comparación.
- Estilo CSS:
  - El formulario debe estar estilizado con CSS para tener un diseño atractivo, que incluya:
    - Un fondo claro y un campo de entrada centrado y con bordes redondeados.
    - Un botón colorido que cambie de color al pasar el ratón sobre él.
    - Una presentación clara y legible del resultado.
- Funcionalidad JavaScript:
  - Se debe implementar la lógica de comparación a través de una función JavaScript, que:
    - Obtenga los valores de los tres campos de entrada.
    - Compare los números y determine cuál es el mayor.
    - Maneje posibles errores de entrada, como campos vacíos o entradas no numéricas, y muestre un mensaje de error en caso de ser necesario.
    - Muestra el resultado de la comparación en el área designada del formulario.
- Validaciones:
  - Antes de realizar la comparación, la aplicación debe verificar que todos los campos de entrada contengan números válidos.
  - Si un campo está vacío o contiene un valor no numérico, debe mostrarse un mensaje de advertencia, indicando que es necesario ingresar números válidos.

Ejercicio 10: Crea una lista desordenada (ul) en HTML con tres elementos (li). Usa JavaScript para agregar un nuevo elemento a la lista al hacer clic en un botón.



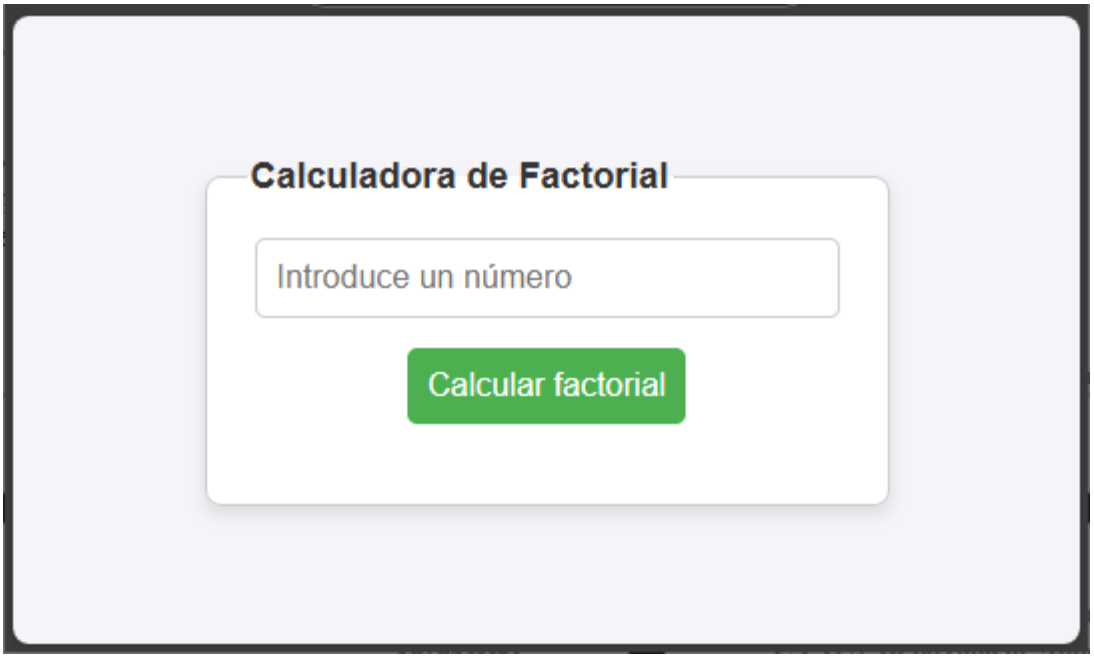
Ejercicio 11: Crea una aplicación web sencilla que permita al usuario calcular el factorial de un número entero no negativo. La aplicación debe contar con un formulario que incluya un campo de entrada para el número y un botón para realizar el cálculo. El resultado debe mostrarse de forma clara y legible en la página. La funcionalidad debe implementarse utilizando JavaScript y los estilos visuales deben ser atractivos mediante CSS.

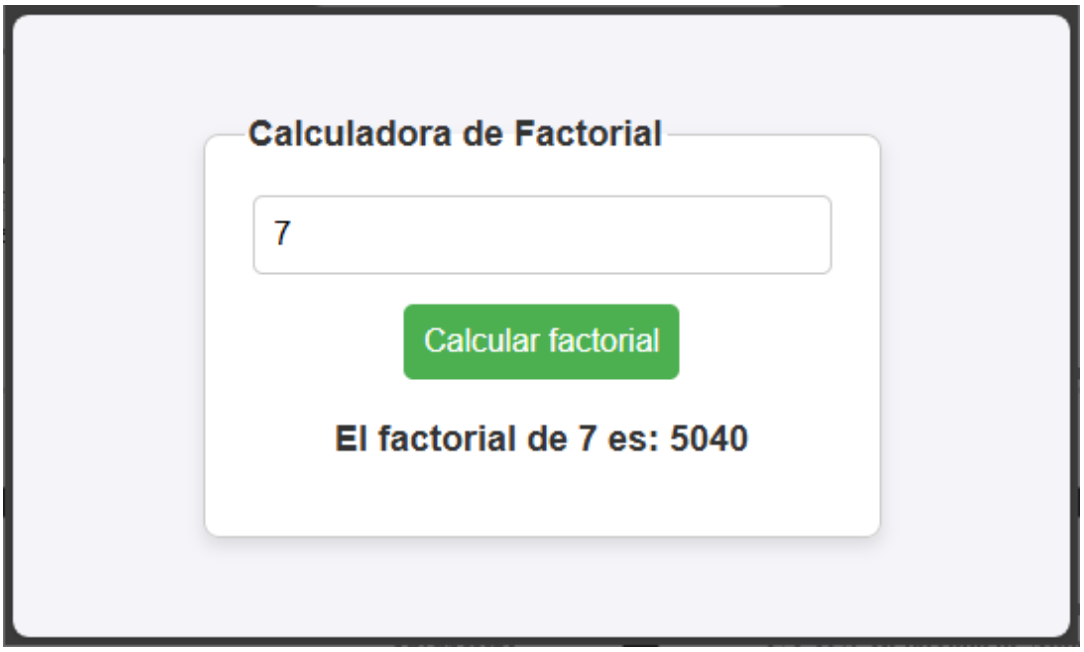
El factorial de un entero positivo n, el factorial de n o n factorial se define en principio como el producto de todos los números enteros positivos desde 1 (es decir, los números naturales) hasta n. Por ejemplo:

$$5! = 5 * 4 * 3 * 2 * 1$$

Requisitos:

- Formulario HTML:
  - El formulario debe contener:
    - Un campo de entrada (<input>) de tipo number que permita al usuario introducir un número entero no negativo.
    - Un botón (<button>) que, al ser presionado, desencadene la función para calcular el factorial del número ingresado.
    - Debe incluirse un área (o <p>) donde se muestre el resultado del cálculo.
- Validaciones:
  - El sistema debe validar que el número introducido en el campo de entrada sea un entero no negativo. Si el usuario introduce un número negativo o un valor no numérico, se debe mostrar un mensaje de error indicándolo.
  - El programa debe manejar el caso en que se introduzca un número muy grande que podría causar un desbordamiento o un rendimiento ineficiente en el cálculo.
- Interactividad:
  - El botón de cálculo debe ser de fácil acceso y debe proporcionar una respuesta inmediata al usuario al hacer clic en él. El resultado debe mostrarse en el área designada sin necesidad de recargar la página.
- Estilos CSS:
  - La aplicación debe tener un diseño limpio y moderno, utilizando CSS para mejorar la experiencia del usuario.
  - El campo de entrada y el botón deben ser de un tamaño adecuado, centrados en la página, y con bordes redondeados.
  - Los mensajes de error deben ser claramente visibles y diferenciables del resultado normal, utilizando colores o estilos diferentes para llamar la atención del usuario.

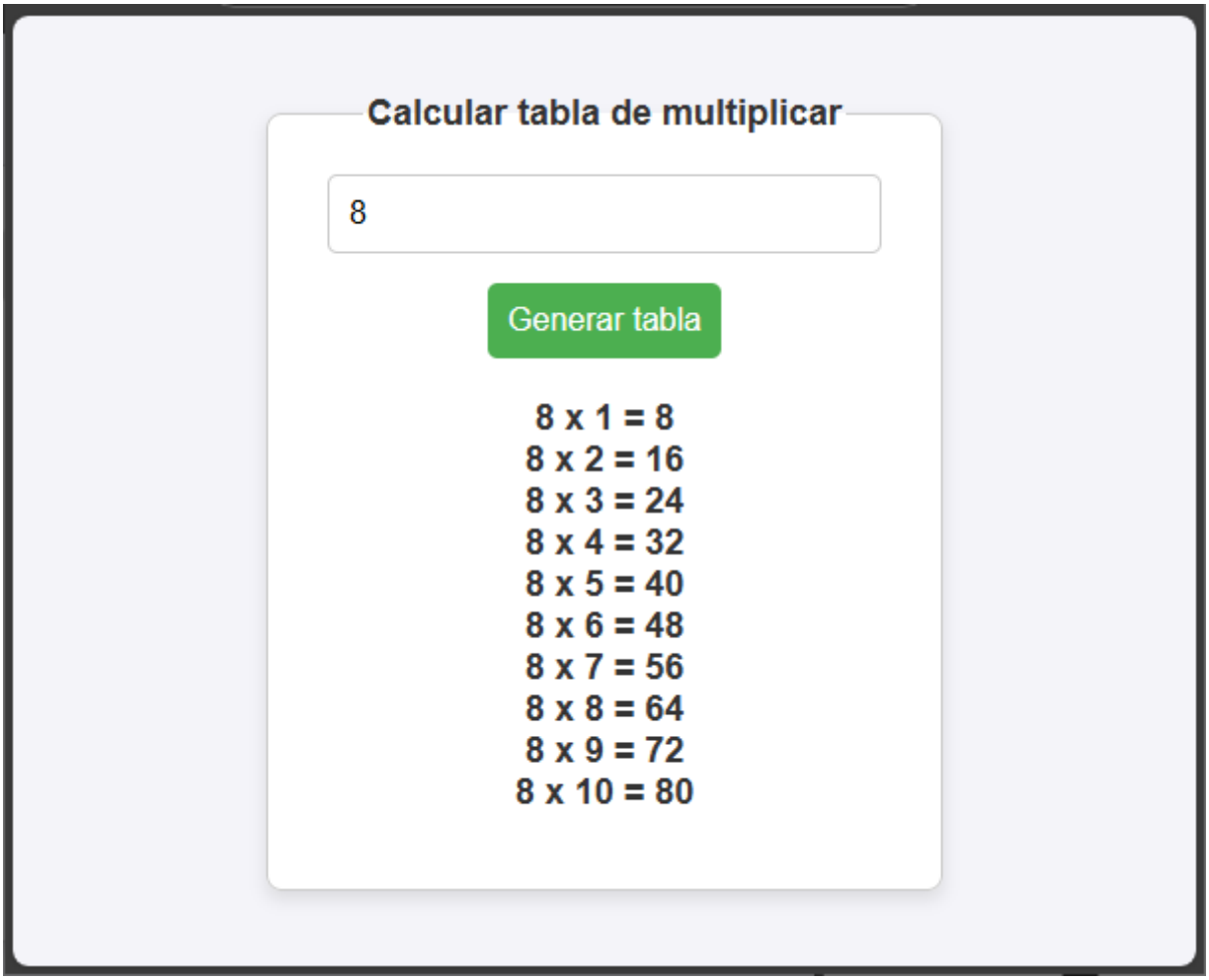

Calculadora de Factorial
Introduce un número
Calcular factorial


Calculadora de Factorial
7
Calcular factorial
El factorial de 7 es: 5040

Ejercicio 12: Crea una aplicación web sencilla que permita al usuario ingresar un número y, al hacer clic en un botón, se muestre la tabla de multiplicar correspondiente a ese número. Esta aplicación se desarrollará utilizando HTML5 para la estructura del formulario, CSS3 para el estilo visual y JavaScript para la lógica de la funcionalidad.

Requisitos:

- Estructura HTML:
  - Se debe crear un formulario que contenga:
    - Un campo de entrada (<input>) donde el usuario pueda introducir un número entero positivo.
    - Un botón (<button>) que, al hacer clic, desencadene la acción de generar y mostrar la tabla de multiplicar.
    - Un área de resultados (o <p>) donde se mostrará la tabla de multiplicar generada.
- Interactividad con JavaScript:
  - Se debe implementar una función que:
    - Obtenga el valor introducido en el campo de entrada.
    - Valide que el número introducido sea un entero positivo. Si el número es negativo o no es un número válido, se debe mostrar un mensaje de error en lugar de la tabla.
    - Calcule la tabla de multiplicar para el número ingresado (del 1 al 10).
    - Muestra la tabla de multiplicar en el área designada cuando el usuario haga clic en el botón.
  - Si el usuario introduce un número no válido, se debe mostrar un mensaje que indique que debe ingresar un número entero positivo.
- Estilo CSS:
  - El formulario debe tener un diseño limpio y moderno, con un fondo que resalte la tabla de multiplicar.
  - Los elementos del formulario (inputs y botones) deben estar alineados y con márgenes adecuados.
  - La tabla de multiplicar debe presentarse de manera clara y organizada, con estilos que hagan que los resultados sean fáciles de leer.
  - Los botones deben tener efectos visuales al pasar el ratón sobre ellos, proporcionando retroalimentación visual al usuario.

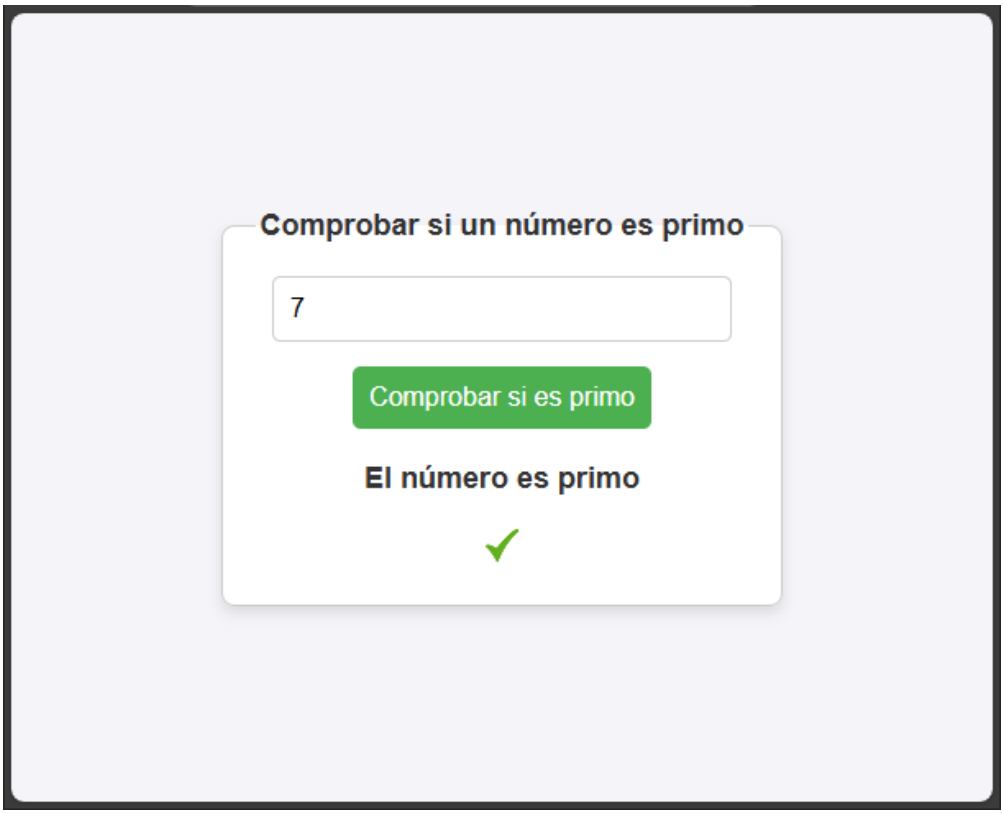


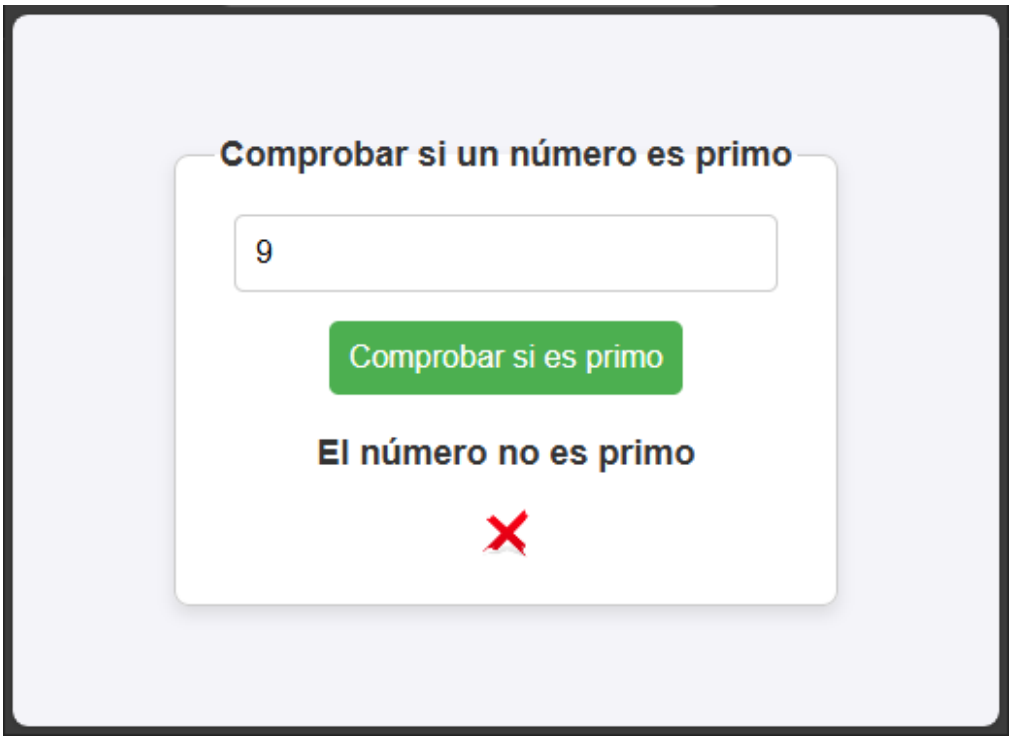
Ejercicio 13: Crea una aplicación web que permita al usuario verificar si un número ingresado es primo. La interfaz constará de un formulario sencillo que contendrá un campo de entrada para el número y un botón que, al ser presionado, mostrará si el número es primo o no. La lógica de verificación será implementada en JavaScript, y se aplicarán estilos modernos utilizando CSS3 para una mejor presentación.

En matemáticas, un número primo es un número natural mayor que 1 que tiene únicamente dos divisores positivos distintos: él mismo y el 1. Si el número es primo se debe mostrar la imagen ✓ mientras que si no lo es se debe mostrar la imagen ✗.

Requisitos:

- Estructura HTML:
  - Crear un archivo HTML5 que contenga:
    - Un formulario que incluya:
      - Un campo de entrada (<input>) de tipo numérico donde el usuario podrá ingresar un número entero positivo.
      - Un botón (<button>) que el usuario podrá pulsar para ejecutar la verificación del número primo.
    - Un área visible (por ejemplo, un div o un p) donde se mostrará el resultado de la verificación, indicando si el número es primo o no.
- Validaciones:
  - El número ingresado debe ser un entero positivo. Se debe validar la entrada y mostrar un mensaje de error si el usuario intenta ingresar un número negativo, cero o no entero.
  - En caso de que el campo esté vacío al presionar el botón, también se deberá mostrar un mensaje indicando que es necesario ingresar un número.
- Estilos CSS:
  - Crear un archivo CSS3 que estilice la interfaz de la aplicación:
  - Aplicar un diseño limpio y moderno.
  - Usar colores agradables y una tipografía legible.
  - Centrar el formulario en la página.
  - Estilizar el campo de entrada y el botón para que sean visualmente atractivos.
  - Proporcionar retroalimentación visual cuando se pase el ratón sobre el botón (por ejemplo, cambiando el color de fondo).



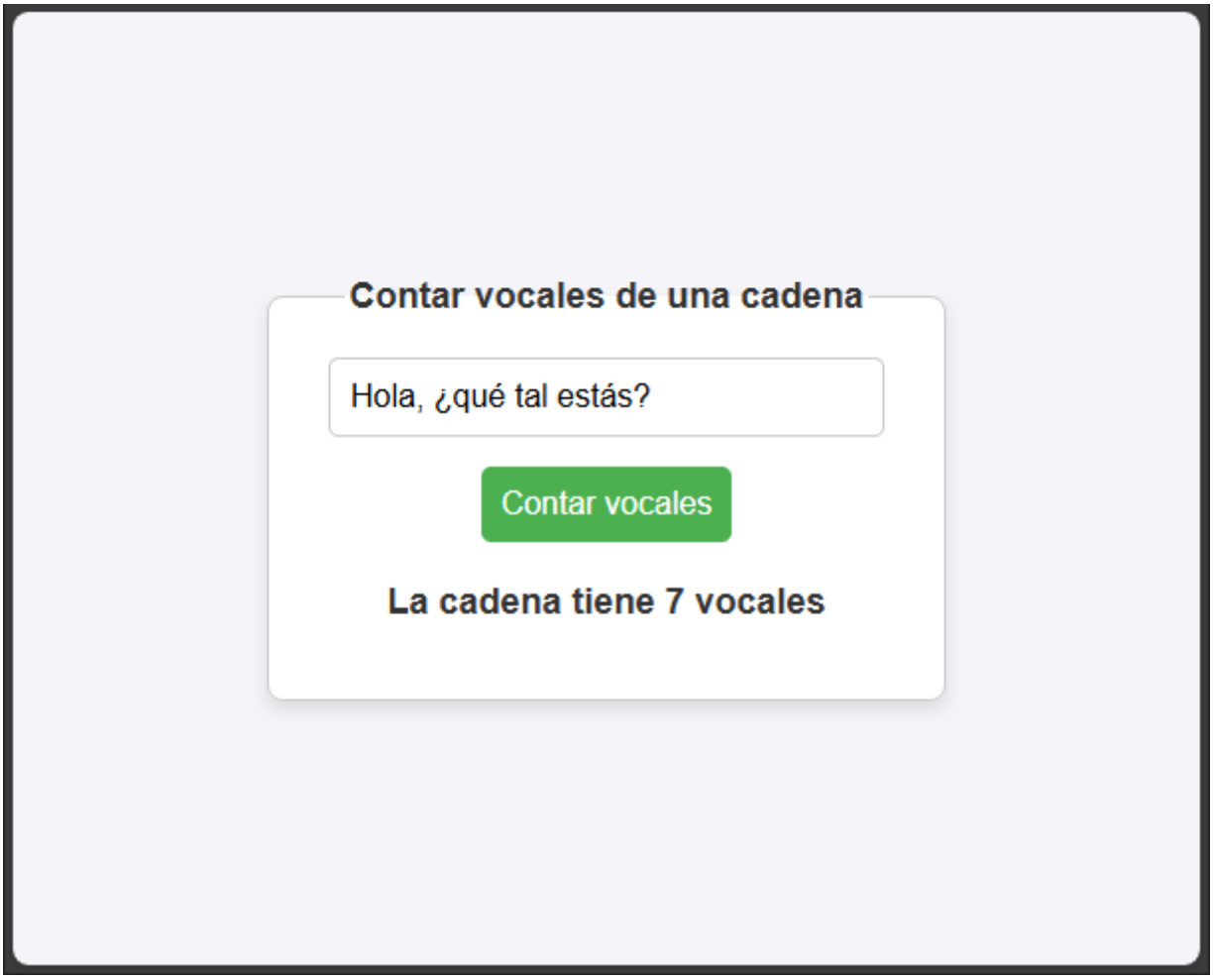


Ejercicio 14: Crea una aplicación web sencilla que permita al usuario introducir una cadena de texto en un formulario y, al pulsar un botón, contar el número de vocales presentes en la cadena. Las vocales a considerar incluirán tanto las minúsculas como las mayúsculas, así como las vocales acentuadas. La funcionalidad se implementará utilizando HTML5, CSS3 y JavaScript.

Requisitos:

- Estructura HTML:
  - Crear un archivo index.html que contenga:
    - Un formulario con un campo de entrada (<input>) donde el usuario pueda escribir una cadena de texto.
    - Un botón (<button>) que, al ser pulsado, ejecute la función para contar las vocales.
  - Un área de resultados (puede ser un o un <p>) donde se mostrará el número de vocales encontradas.
- Validaciones:
  - Al pulsar el botón, se debe verificar si el campo de entrada está vacío. Si el campo está vacío, se deberá mostrar un mensaje de error, indicando que se debe introducir texto.
  - Si el texto ingresado es válido, el programa debe proceder a contar las vocales.
- Estilos CSS:
  - Crear un archivo estilos.css para aplicar estilos atractivos al formulario. Las características del diseño deben incluir:
    - Un fondo claro y agradable para la vista.
    - Bordes redondeados para los campos de entrada y el botón.
    - Un diseño centrado que sea responsivo y fácil de usar.
    - Un efecto visual en el botón al pasar el ratón sobre él.

- El área de resultados debe ser claramente visible y tener un estilo que lo distinga del resto del formulario.



Ejercicio 15: Crea una página que muestre la hora actual actualizándose cada segundo. El formato debe ser *HH:MM:SS*. Muestra la hora en un div.

Ejercicio 16: Crea una web que permita al usuario iniciar un temporizador en segundos. El temporizador debe descontar hasta llegar a 0 y mostrar el tiempo restante en un div. Cuanto llegue a 0 deberá parar de contar y mostrar el mensaje ¡Tiempo acabado!.

Ejercicio 17: Crea una web que cuente cuántas veces un usuario ha hecho clic en un botón. Muestra el número de clics en un div.

Ejercicio 18: Crea una aplicación web sencilla que permita a los usuarios navegar entre varias imágenes utilizando botones de "Anterior" y "Siguiente". Esta aplicación tiene como finalidad ofrecer una interfaz intuitiva y visualmente atractiva para mostrar imágenes en secuencia.

Requisitos:

- Estructura del Proyecto:
  - La aplicación debe consistir en tres archivos:
    - HTML (index.html): Para la estructura básica de la página.
    - CSS (estilos.css): Para aplicar estilos y mejorar la presentación de la aplicación.
    - JavaScript (script.js): Para implementar la lógica de navegación entre imágenes.
- Interfaz de Usuario:
  - La interfaz debe incluir:
    - Una imagen que se mostrará en el centro de la página.
      - Dos botones:
        - Anterior: Permite al usuario retroceder a la imagen anterior en la secuencia.
        - Siguiente: Permite al usuario avanzar a la siguiente imagen en la secuencia.
  - La disposición de los elementos debe ser visualmente equilibrada y centrada en la pantalla.
- Funcionalidad de Navegación:
  - El sistema debe almacenar un conjunto de imágenes en un arreglo en el archivo JavaScript.
  - La lógica de navegación debe permitir al usuario:
    - Avanzar a la siguiente imagen cuando se haga clic en el botón "Siguiente".
    - Retroceder a la imagen anterior cuando se haga clic en el botón "Anterior".
    - Si el usuario está en la primera imagen y hace clic en "Anterior", debe mostrarse la última imagen del arreglo (navegación cíclica).
    - Si el usuario está en la última imagen y hace clic en "Siguiente", debe mostrarse la primera imagen del arreglo.
- Estilos CSS:
  - Los estilos deben ser agradables y modernos, con un diseño responsivo que se adapte a diferentes tamaños de pantalla.
  - La imagen debe estar centrada, y los botones deben ser fácilmente accesibles y visualmente atractivos.
  - Los botones deben incluir un efecto de hover (cambio de color) para mejorar la experiencia del usuario.
- Validaciones y Consideraciones:
  - No es necesario validar entradas del usuario, ya que la aplicación solo se basa en la navegación de imágenes.
  - La aplicación debe ser fácil de usar y entender, asegurando que los usuarios puedan navegar entre imágenes sin confusión.

Ejercicio 19: Crea una aplicación web que permita a los usuarios convertir temperaturas entre grados Celsius y grados Fahrenheit. La aplicación contará con un formulario que incluye un campo de entrada para la temperatura, un menú desplegable (select) para elegir el tipo de conversión y un botón que ejecutará la conversión. El resultado se mostrará al usuario de manera clara y amigable.

Requisitos:

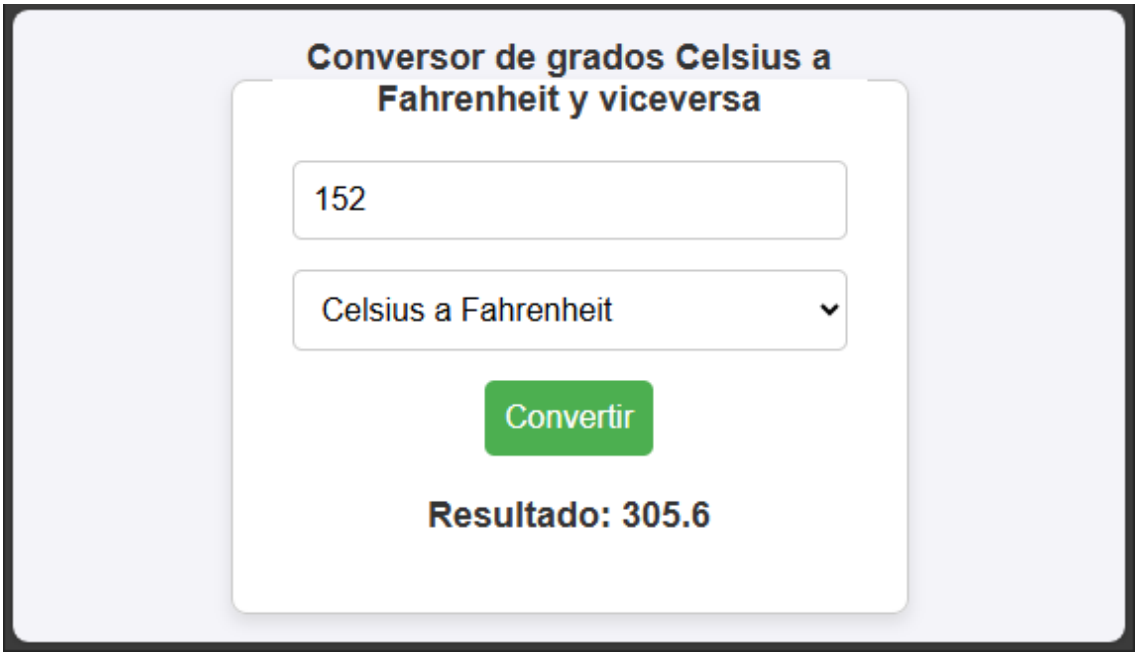
- Formulario HTML:
  - El formulario debe contener:
    - Un campo de entrada (<input>) para que el usuario introduzca la temperatura a convertir. Este campo debe ser de tipo number para permitir solo la entrada de valores numéricos.
    - Un menú desplegable (<select>) que permita al usuario seleccionar entre las siguientes opciones:
      - Celsius a Fahrenheit
      - Fahrenheit a Celsius
    - Un botón (<button>) que, al ser presionado, iniciará el proceso de conversión.
- Cálculos de Conversión:
  - La conversión de temperatura debe realizarse mediante las siguientes fórmulas matemáticas:
    - De Celsius a Fahrenheit:

$$F = C\ x\ \frac{9}{5} + 32$$

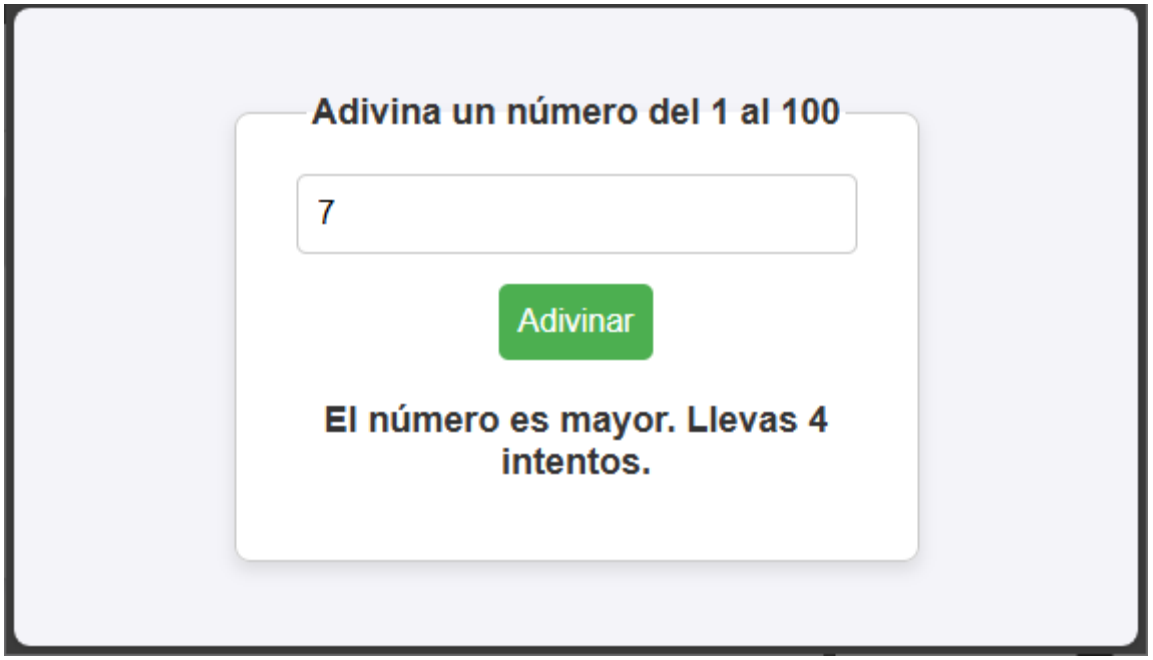
      - Donde F es la temperatura en grados Fahrenheit y C es la temperatura en grados Celsius.
    - De Fahrenheit a Celsius:

$$C = (F - 32)\ x\ \frac{5}{9}$$

    - Donde C es la temperatura en grados Celsius y F es la temperatura en grados Fahrenheit.
- Validaciones:
    - Antes de realizar la conversión, se debe validar que el campo de entrada no esté vacío. Si el campo está vacío, debe mostrarse un mensaje de error indicando que se necesita un valor.
    - El usuario debe recibir un mensaje claro sobre el resultado de la conversión en un área visible de la página.
- Estilos CSS:
    - Se deben aplicar estilos CSS para mejorar la apariencia del formulario. Esto incluye:
    - Un diseño limpio y moderno con un fondo agradable.
    - Campos de entrada y botones con bordes redondeados.
    - Un diseño centrado en la página que sea responsive y accesible en diferentes dispositivos.



Ejercicio 20: Crea una web que genere un número aleatorio entre 1 y 100. El usuario debe intentar adivinar el número introduciéndolo en un formulario. El programa debe indicar si el número es mayor, menor o correcto. Deberá mostrar también el número de intentos que llevas y una vez adivinado el número mostrará el número de intentos que has utilizado.



Ejercicio 21: Crea una página web utilizando HTML5, CSS3 y JavaScript que permita al usuario introducir una serie de números separados por comas en un formulario. El usuario podrá elegir si desea ordenar los números de forma ascendente o descendente mediante la selección de un grupo de radiobuttons. Al hacer clic en un botón, los números serán ordenados y se mostrarán en pantalla.

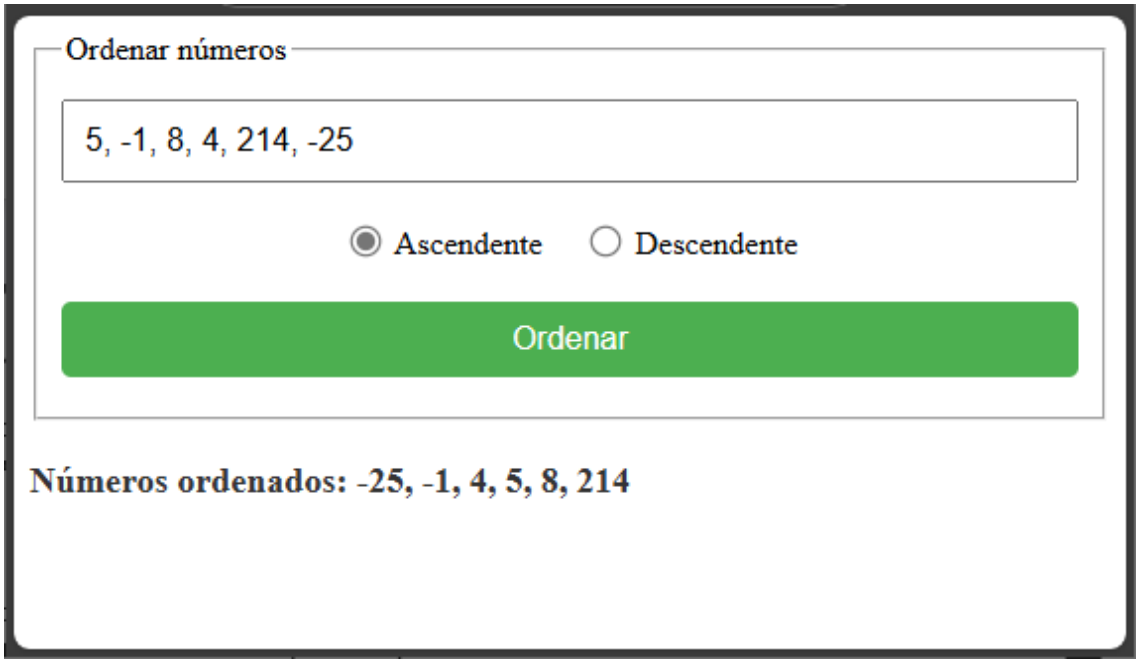
Requisitos funcionales:

- Campo de entrada:
  - El formulario debe contener un campo de texto (<input type="text">) en el cual el usuario podrá introducir una lista de números separados por comas.
  - Ejemplo de entrada válida: 5, 10, 2, 8, 3.
- Selección del criterio de ordenación:
  - Deben aparecer dos opciones representadas por radiobuttons:
    - Ascendente: Ordenará los números de menor a mayor.
    - Descendente: Ordenará los números de mayor a menor.
  - Uno de los radiobuttons debe estar seleccionado por defecto, siendo el ascendente.
- Botón de ordenar:
  - El formulario debe contener un botón que, al ser presionado, ejecute la lógica de ordenación según el criterio seleccionado (ascendente o descendente).
  - Este botón no debe enviar el formulario a otro lugar ni recargar la página, sino que debe procesar la información utilizando JavaScript.

- Mostrar el resultado:
  - Los números ordenados deben mostrarse en una sección o párrafo debajo del formulario. El resultado debe actualizarse cada vez que el usuario haga clic en el botón "Ordenar".
  - El resultado debe estar claramente visible para el usuario.

Requisitos:

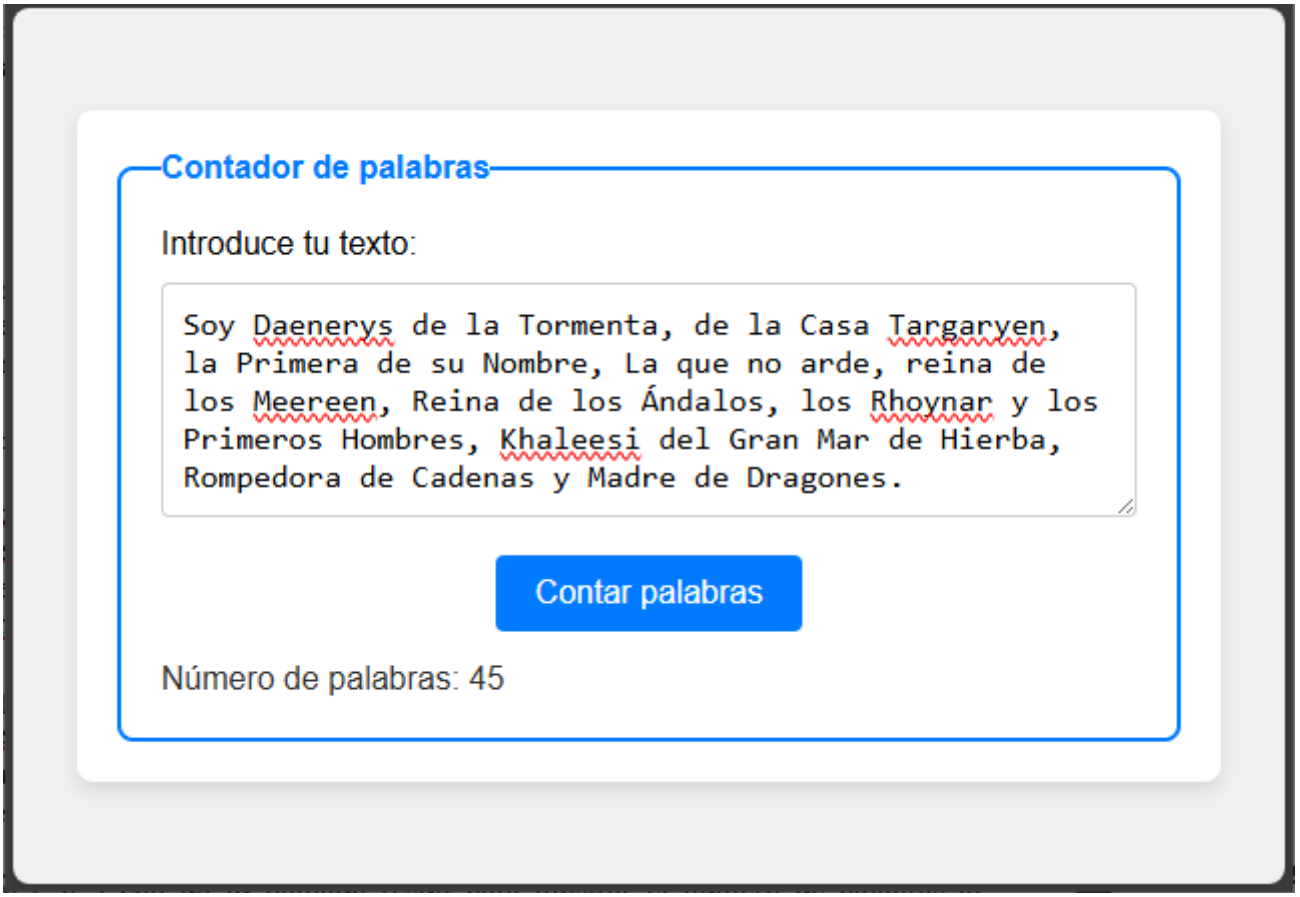
- HTML5: Estructura de la página con un formulario que incluya:
  - Un campo de texto para la introducción de los números.
  - Dos radiobuttons para seleccionar el tipo de orden (ascendente o descendente).
  - Un botón para ejecutar la ordenación.
- CSS3: Estilos visuales coherentes para los siguientes elementos:
  - El campo de entrada, los radiobuttons y el botón deben estar estilizados de manera uniforme, con márgenes, padding y colores adecuados.
  - Los radiobuttons deben estar centrados horizontalmente en la página.
  - Los resultados deben mostrarse con una tipografía clara y visible.
- JavaScript: Lógica de funcionamiento que:
  - Capture el valor introducido en el campo de texto y lo convierta en un array de números.
  - Determine qué radiobutton está seleccionado y aplique la ordenación ascendente o descendente.
  - Actualice la interfaz para mostrar los números ordenados.



Ejercicio 22: Crea una página web sencilla que contenga un formulario para que el usuario introduzca un texto. La página deberá contar el número de palabras en el texto ingresado y mostrar el resultado al usuario al pulsar un botón. El ejercicio requiere la implementación de HTML5, CSS3 y JavaScript, asegurando que todos los elementos visuales y funcionales estén organizados y estilizados de manera profesional.

Requisitos:

- Estructura del Proyecto:
  - La página HTML debe llamarse index.html.
  - Debe incluir un archivo de estilos CSS llamado estilos.css.
  - El JavaScript debe colocarse en un archivo independiente llamado javascript.js, ubicado dentro de una carpeta llamada js.
- HTML5:
  - Crear un documento HTML que estructure un formulario dentro de un contenedor principal (<main>).
  - El formulario (<form>) debe contener los siguientes elementos:
    - Un fieldset que agrupe todos los elementos relacionados del formulario. Este fieldset debe contener:
    - Una legend que describa el propósito del formulario con el texto: "Contador de palabras".
    - Un label que describa el campo de entrada con el texto: "Introduce tu texto:".
    - Un textarea para que el usuario introduzca el texto. Debe tener los siguientes atributos:
      - Un id para poder acceder a él desde JavaScript (por ejemplo, textInput).
      - Debe tener un placeholder que indique: "Escribe algo aquí...".
    - Un button de tipo button con el texto "Contar palabras" y un id llamado countButton.
    - Un párrafo (<p>) con un id llamado result para mostrar el número de palabras al usuario. Inicialmente debe mostrar "Número de palabras: 0".
- CSS3 (estilos.css):
  - Aplicar un estilo moderno y limpio a la página, teniendo en cuenta:
    - El body debe estar centrado horizontal y verticalmente en la ventana del navegador.
    - El fieldset debe tener un borde con un color agradable y bordes redondeados para hacerlo visualmente atractivo.
    - El legend debe destacar con un color que contraste con el fondo del fieldset.
    - El textarea debe ajustarse al 100% del ancho disponible dentro del fieldset, tener bordes redondeados y ser redimensionable solo en la dirección vertical.
  - El botón debe:
    - Estar centrado horizontalmente dentro del formulario.
    - Cambiar de color al hacer hover para indicar que es interactivo.
  - El párrafo que muestra el resultado debe estar estilizado con un color de texto adecuado para que sea legible.



Ejercicio 23: Crea una aplicación web sencilla que permita al usuario agregar tareas a una lista interactiva. Junto a cada elemento de la lista deberá aparecer un botón que permita eliminar dicha tarea. La aplicación utilizará HTML5, CSS3 y JavaScript. El ejercicio se centrará en la implementación de la estructura, el estilo y la funcionalidad, con el fin de aplicar conceptos básicos y fundamentales en el desarrollo web.

Requisitos:

- Estructura del Proyecto:
  - El proyecto estará compuesto por tres archivos principales:
    - Un archivo HTML (index.html) que contendrá la estructura de la página.
    - Un archivo CSS (estilos.css) que proporcionará el diseño visual de la aplicación.
    - Un archivo JavaScript (javascript.js) que se encargará de la lógica de la aplicación.
    - Además, el archivo de JavaScript deberá estar ubicado dentro de una carpeta llamada js.
- Estructura del Documento HTML (index.html):
  - El documento HTML debe seguir la estructura básica de HTML5 y contener los siguientes elementos:
    - Encabezado principal: Un h1 con el título "Lista de Tareas".
    - Formulario de entrada: Un fieldset que contenga:
      - Un legend con la leyenda "Agregar tarea nueva".

- Un input de tipo texto para que el usuario introduzca una tarea. El input debe tener un atributo placeholder con el texto "Escribe una tarea..." y un id llamado "nueva-tarea".
- Un button que se utilizará para agregar la tarea a la lista. El botón debe tener el texto "Agregar" y un id llamado "agregar-btn".
- Lista de tareas: Una lista desordenada (ul) que se usará para mostrar las tareas agregadas por el usuario. Esta lista debe tener un id llamado "lista-tareas".

- Estilo de la Página (CSS - estilos.css):
  - La página debe tener un diseño limpio y agradable a la vista. Usa colores suaves y estilos redondeados para hacerla atractiva.
  - El archivo CSS debe contener comentarios detallados en cada línea o bloque de código, explicando la función de cada estilo o regla aplicada.
  - Aplica los siguientes estilos básicos:
    - Un fondo claro y una fuente legible para el cuerpo de la página.
    - Centra el fieldset y añade bordes redondeados para darle un aspecto moderno.
    - Diseña el input y el button para que se vean agradables y estén alineados horizontalmente.
    - Estiliza la lista (ul y li) para que las tareas se muestren de manera clara, con un fondo de color suave y bordes redondeados.

Ejercicio 24: Crea una página web con un cronómetro funcional utilizando HTML5, CSS3 y JavaScript. La página debe incluir botones para iniciar, detener y reiniciar el cronómetro.

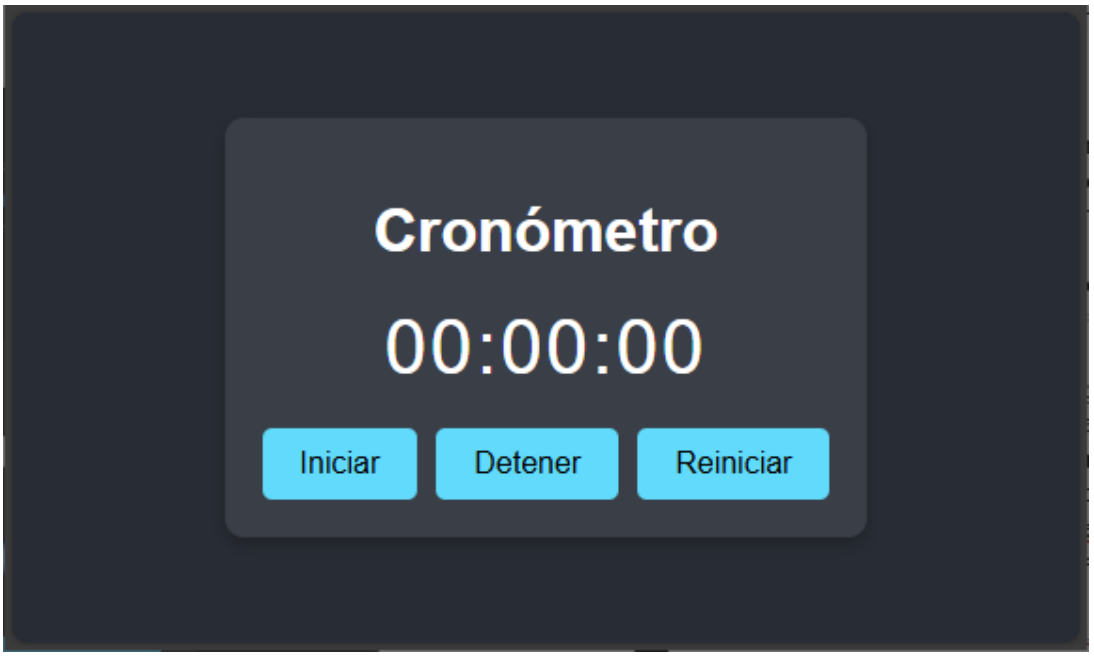
Requisitos:

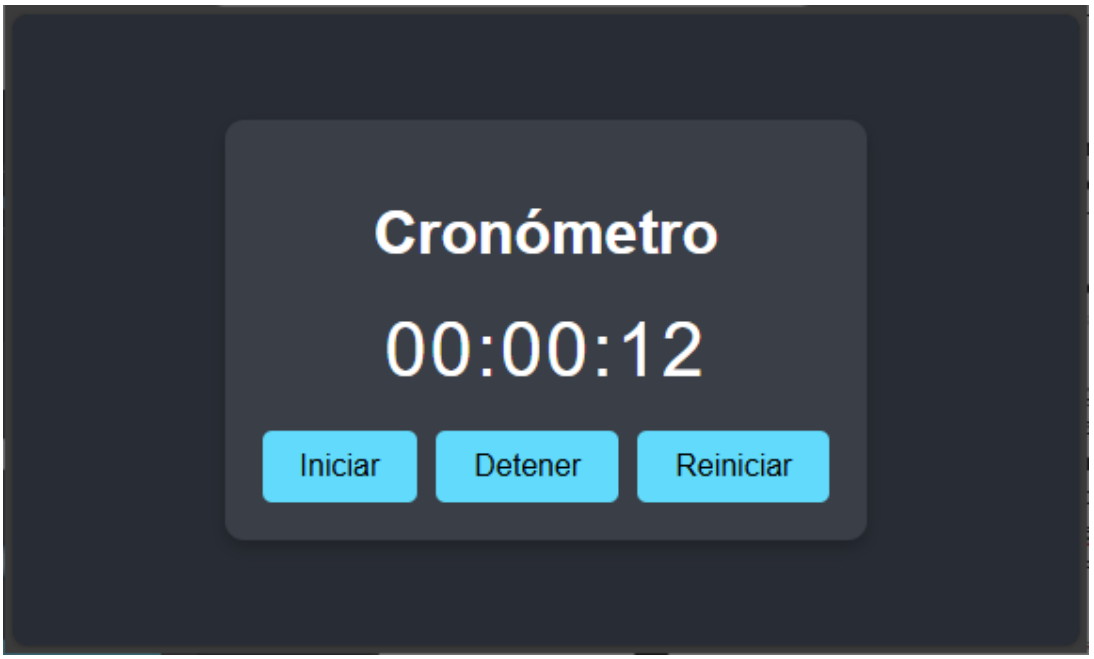
- Archivos y estructura del proyecto:
  - El proyecto debe estar organizado en la siguiente estructura:
    - El archivo index.html será el archivo principal donde se estructurará la página.
    - El archivo estilos.css se utilizará para los estilos CSS.
    - El archivo javascript.js (dentro de la carpeta js) contendrá el código JavaScript necesario para el funcionamiento del cronómetro.
- Funciones JavaScript necesarias:
  - Date.now(): Esta función se utilizará para capturar el tiempo en milisegundos desde el inicio del cronómetro y calcular el tiempo transcurrido.
  - setInterval(): Se usará para actualizar el cronómetro cada segundo. Este método permitirá ejecutar una función periódicamente para actualizar el tiempo mostrado en pantalla.
  - clearInterval(): Esta función se utilizará para detener la ejecución del intervalo cuando el cronómetro se detenga o se reinicie.
  - formatTime(): Esta función personalizable será utilizada para convertir el tiempo en milisegundos a un formato legible de horas:minutos:segundos.

Especificaciones de la página web:

- Estructura del HTML (index.html):
  - El HTML debe contener una estructura sencilla con los siguientes elementos:
    - Un título principal (<h1>) que indique "Cronómetro".
    - Un contenedor ( ) para mostrar el tiempo transcurrido, con un ID display que permita su actualización desde JavaScript.
    - Tres botones (<button>) con los IDs start, stop y reset respectivamente, para controlar el cronómetro.
- Estilos CSS (estilos.css):
  - El diseño debe ser moderno y minimalista, utilizando una paleta de colores clara y oscura para dar contraste.
  - Paleta de colores sugerida:
    - Fondo general: #282c34 (gris oscuro).
    - Contenedor: #3a3f47 (gris intermedio).
    - Texto principal: #FFFFFF (blanco).
    - Botones: #61dafb (azul claro).
    - Botones al pasar el cursor: #21a1f1 (azul más oscuro).
  - Los estilos deben centrarse en crear un diseño responsive y accesible. Deben incluirse sombras suaves para darle un efecto moderno y bordes redondeados para suavizar el aspecto de los elementos.
- Lógica del cronómetro en JavaScript (javascript.js):
  - El código JavaScript debe cumplir con las siguientes especificaciones:
    - Capturar los elementos del DOM como el display y los botones utilizando document.getElementById().
    - Utilizar Date.now() para obtener el tiempo actual en milisegundos y calcular el tiempo transcurrido.
    - Usar setInterval() para ejecutar una función cada segundo que actualice el tiempo en el display.

- Implementar clearInterval() para detener el cronómetro cuando se presione el botón de detener o reiniciar.
- Crear una función formatTime() para convertir los milisegundos en un formato hh:mm:ss.





Ejercicio 25: Crea una página web que permita al usuario introducir una lista de números separados por comas y calcular el promedio de dichos números utilizando HTML5, CSS3 y JavaScript. La interfaz debe ser moderna, accesible y funcional, proporcionando una experiencia de usuario clara y visualmente atractiva.

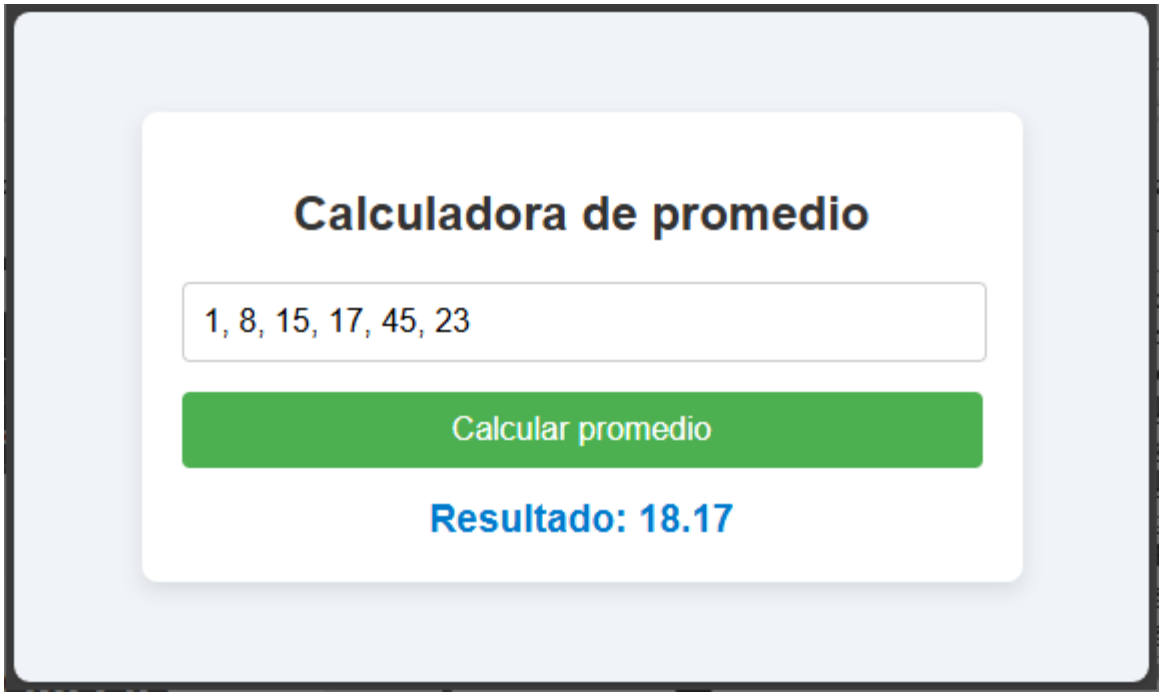
Requisitos:

- Archivos y estructura del proyecto:
  - El proyecto debe tener la siguiente estructura:
    - El archivo index.html será el archivo principal que define la estructura de la página web.
    - El archivo estilos.css contendrá los estilos CSS para darle formato y diseño a la página.
    - El archivo javascript.js estará ubicado en una carpeta llamada js y contendrá la lógica para calcular el promedio.
- Funciones JavaScript necesarias:
  - split(','): Esta función dividirá la cadena de texto ingresada por el usuario en un array, utilizando la coma como separador.
  - .trim(): Se utilizará para eliminar espacios en blanco que puedan rodear los números en el array.
  - .map(Number): Convertirá cada elemento del array en un número, si es posible.
  - .filter(num => !isNaN(num)): Filtrará los valores no numéricos, dejando solo números válidos en el array.
  - .reduce((acc, num) => acc + num, 0): Suma todos los elementos del array para obtener el total.
  - average.toFixed(2): Redondeará el promedio a dos decimales para presentarlo de manera precisa y legible.

Especificaciones de la página web:

- Estructura del HTML (index.html):
  - El HTML debe contener los siguientes elementos:
    - Un título principal (<h1>) que indique "Calculadora de Promedio".
    - Un campo de entrada de texto (<input>) con el id="numberInput" para que el usuario pueda introducir los números separados por comas.
    - Un botón (<button>) con el id="calculateButton" para activar el cálculo del promedio.
    - Un contenedor de resultados ( ) con el id="result" donde se mostrará el resultado calculado o mensajes de error en caso de entrada incorrecta.
- Estilos CSS (estilos.css):
  - La página debe tener un diseño moderno y minimalista, utilizando una paleta de colores claros y brillantes para dar un aspecto fresco y profesional:
    - Fondo general: #f0f4f8 (gris claro).
    - Contenedor: #ffffff (blanco).
    - Texto principal: #333333 (gris oscuro).
    - Botón principal: #4caf50 (verde brillante).
    - Hover del botón: #45a049 (verde oscuro).
    - Resultado: #007acc (azul para resaltar).
  - Los estilos deben ser responsivos y centrados en la accesibilidad, con un uso adecuado de sombras, bordes redondeados y transiciones suaves para crear un efecto visual agradable.
- Lógica del cronómetro en JavaScript (javascript.js):
  - El archivo javascript.js deberá capturar el valor del input, dividirlo y limpiar el array de números, convirtiendo cada valor a un número usando Number().
  - Si no hay números válidos, se mostrará un mensaje de error en el área de resultado.

- Se calculará la suma total de los números y se calculará el promedio, que se mostrará en el contenedor de resultado.
- El código debe estar bien estructurado y comentado para facilitar la comprensión.



Ejercicio 26: Crea una página web que valide si un número de teléfono introducido por el usuario sigue el formato específico: +34 seguido de 9 dígitos. Para ello, se utilizarán HTML5, CSS3 y JavaScript, organizados en una estructura de archivos bien definida:

- El archivo HTML se llamará index.html y contendrá la estructura de la página.
- El archivo de estilos CSS se llamará estilos.css y definirá el diseño visual de la página.
- El archivo JavaScript se llamará javascript.js, ubicado en una carpeta llamada js, y será responsable de la lógica de validación del número de teléfono.

Especificaciones de la página web:

- Estructura del Archivo HTML (index.html):
  - El archivo HTML debe contener la estructura básica de la página y los elementos necesarios para interactuar con el usuario:
    - Un título principal (h1) que indique la funcionalidad de la página: "Validador de Número de Teléfono".
    - Un formulario con un campo de entrada (input) donde el usuario introducirá el número de teléfono. Este campo será de tipo text.
    - Un botón para enviar el formulario y validar el número.
    - Un elemento de texto (div) donde se mostrará el mensaje de validación al usuario (indicando si el número es válido o no).

        - Enlaza el archivo CSS (estilos.css) para aplicar los estilos y el archivo JavaScript (javascript.js) para ejecutar la lógica.
- Estilos CSS (estilos.css):
    - Los estilos deben darle a la página un aspecto moderno y limpio, utilizando una paleta de colores fresca y una estructura simple que sea fácil de entender.
    - Paleta de Colores:
        - Fondo general: #e0f7fa (azul claro).
        - Texto: #333 (gris oscuro).
        - Fondo de los formularios: #ffffff (blanco).
        - Botones: #00796b (verde oscuro) con un efecto hover que cambia el color a #004d40 (verde más oscuro).
        - Mensajes de error: #d32f2f (rojo).
        - Mensajes de éxito: #388e3c (verde claro).
- Lógica JavaScript (javascript.js):
    - El archivo JavaScript implementa la lógica para validar el número de teléfono siguiendo un formato específico. La estructura del archivo está detalladamente comentada para que sea fácil de entender y modificar. La lógica está dividida en funciones y bloques específicos:
        - validatePhoneNumber(phoneNumber): Esta función utiliza una expresión regular para verificar si el número de teléfono introducido cumple con el formato del número de teléfono en España: +34 seguido de 9 dígitos continuos.
        - Manejo del Evento submit:
            - Se añade un evento al formulario que se activa al intentar enviarlo. Este evento evita que la página se recargue y, en su lugar, ejecuta la lógica de validación.
            - Se obtiene el valor introducido en el campo de texto y se le aplica la función validatePhoneNumber.
            - Si el número es válido, se muestra un mensaje verde indicando que es correcto; si es inválido, se muestra un mensaje rojo indicando que debe cumplir con el formato +34 seguido de 9 dígitos.

Ejemplos de números válidos e inválidos

- Válidos:
    - +34666666666
    - +34987654321
- Inválidos:
    - +34 666 666 666 (contiene espacios)
    - +34987654 (menos de 9 dígitos)
    - 34666666666 (falta el +)

Ejemplo de utilización de una expresión regular en JavaScript

```
// Función que valida si una cadena contiene solo dígitos
function validateNumbers(input) {
    // Expresión regular para comprobar si la cadena contiene solo dígitos
    const numberRegex = /^[0-9]+$/;

    // Utilizamos el método test() para comprobar si la entrada cumple con la expresión regular
    return numberRegex.test(input);
}

// Ejemplos de uso
const input1 = "123456"; // Solo dígitos
const input2 = "123a56"; // Contiene letras
const input3 = "4567890"; // Solo dígitos

// Imprimimos el resultado de las validaciones en la consola
console.log(validateNumbers(input1)); // true, es válido
console.log(validateNumbers(input2)); // false, no es válido
console.log(validateNumbers(input3)); // true, es válido
```

Explicación del código:

- numberRegex: La expresión regular /^[0-9]+$/ se utiliza para validar que la cadena de texto contiene solo dígitos y sigue esta estructura:
  - ^: Indica el inicio de la cadena.
  - [0-9]+: Indica que debe contener uno o más dígitos (del 0 al 9).
  - $: Indica el final de la cadena, asegurando que no hay otros caracteres adicionales.
- test(): Este método de la expresión regular retorna true si la cadena de texto cumple con el patrón de la expresión regular y false si no lo cumple.

Llamadas a validateNumbers:

input1: "123456" es una cadena válida y la función retorna true.
input2: "123a56" no es válida (contiene una letra), por lo que retorna false.
input3: "4567890" es válida y la función retorna true.


Validador de número de teléfono
666666666
Validar
Número de teléfono no válido. Por favor, introduce un formato correcto.


Validador de número de teléfono
+34666666666
Validar
Número de teléfono válido

Ejercicio 27: Crea una aplicación web que permita a los usuarios calcular su Índice de Masa Corporal (IMC) introduciendo su peso en kilogramos y su altura en centímetros. El IMC es una medida utilizada para evaluar si una persona tiene un peso saludable en relación con su altura.

Cálculo del IMC

El IMC se calcula utilizando la siguiente fórmula:

$$IMC = \frac{peso\ (kg)}{(altura\ (m))^2}$$

Dado que el usuario ingresará la altura en centímetros, se convertirá a metros dividiendo entre 100.

Categorías de IMC

Una vez calculado el IMC, la aplicación debe clasificarlo en las siguientes categorías:

- Peso inferior al normal: IMC < 18.5
- Normal: 18.5 ≤ IMC < 25
- Peso superior al normal: 25 ≤ IMC < 30
- Obesidad: IMC ≥ 30

Cada categoría tendrá un estilo visual específico que facilitará la comprensión del resultado.

Especificaciones de la página web:

- Estilos (estilos.css):
  - Los estilos de la aplicación están diseñados para ser modernos, claros y fáciles de entender.
  - Paleta de Colores:
    - Fondo general: Un tono claro (#f0f4f8) que proporciona un contraste suave con los elementos del formulario.
    - Contenedor principal: Blanco (#ffffff) con bordes redondeados y sombra para crear una sensación de profundidad.
    - Texto del título: Un verde oscuro (#00796b) que transmite frescura y salud.
    - Texto de etiquetas: Gris oscuro (#333333) para asegurar buena legibilidad.
  - Estilos de Formulario:
    - Entradas de texto: Ancho completo, bordes redondeados y un fondo blanco con borde gris claro.
    - Botón: Verde oscuro (#00796b) con texto blanco, que cambia a un tono más oscuro (#00574b) al pasar el ratón sobre él.
  - Estilos para Categorías de IMC:
    - Peso inferior al normal: Verde (#388e3c), que transmite salud y bienestar.
    - Normal: Verde oscuro (#00796b), que refleja un estado óptimo.
    - Peso superior al normal: Naranja (#ffa000), que sugiere precaución.
    - Obesidad: Rojo (#d32f2f), que advierte de un riesgo potencial para la salud.
- Funcionalidad del JavaScript (javascript.js):
  - El archivo JavaScript es responsable de la lógica del cálculo del IMC y la manipulación del DOM. A continuación, se describen sus componentes:
    - Selección de elementos: Utiliza document.getElementById para seleccionar el formulario y el área de resultados.

- Función calcularIMC: Toma como parámetros el peso y la altura (convertida a metros) y devuelve el IMC calculado.
- Manejo del evento de envío del formulario:
    - Previene el comportamiento por defecto del formulario que recarga la página.
    - Obtiene los valores introducidos por el usuario.
    - Llama a la función calculateIMC para obtener el IMC.
    - Determina la categoría del IMC y aplica la clase CSS correspondiente para estilos.
    - Muestra el resultado en el área de resultados.

Para prevenir el comportamiento por defecto de recargar la página podemos utilizar el siguiente código como primera línea del funcionamiento del evento de envío del formulario:

```
1. event.preventDefault();
```

EJERCICIOS DE VALIDACIÓN DE FORMULARIOS

Ejercicio 28: Crea una aplicación web que permita a los usuarios introducir un nombre de usuario que debe cumplir con ciertas restricciones de longitud. Específicamente, el nombre de usuario debe tener entre 5 y 15 caracteres. La validación no se realizará utilizando los atributos minlength y maxlength que ofrece el elemento <input> de HTML5, sino que se llevará a cabo mediante JavaScript, lo que permite un mayor control sobre la lógica de validación y el manejo de los mensajes de retroalimentación para el usuario.

Requisitos de la validación:

- Longitud del nombre de usuario:
  - El nombre de usuario debe tener un mínimo de 5 caracteres.
  - El nombre de usuario no puede exceder los 15 caracteres.

Mensajes de retroalimentación:

- Si el nombre de usuario cumple con los requisitos, se mostrará un mensaje de éxito.
- Si el nombre de usuario no cumple con los requisitos, se mostrará un mensaje de error que indicará la necesidad de ajustar la longitud.

Especificaciones de la página web:

- Estilos (estilos.css):
  - Los estilos de la aplicación están diseñados para proporcionar una experiencia visual moderna y clara. A continuación, se detalla la paleta de colores y los estilos aplicados:
    - Paleta de colores
      - Fondo general: Un tono suave de azul claro (#e1f5fe), que crea un ambiente acogedor y limpio.
      - Contenedor principal: Blanco (#ffffff) con bordes redondeados y sombra para dar profundidad y destacar del fondo.
      - Texto del título: Un verde oscuro (#00796b) que transmite frescura y positividad.
      - Texto de etiquetas: Gris oscuro (#333333), asegurando una buena legibilidad.
    - Estilos de formulario
      - Entradas de texto: Ancho completo del contenedor, con bordes redondeados y un fondo blanco que brinda un contraste suave.
      - Botón: Verde oscuro (#00796b) con texto blanco, que cambia a un tono más oscuro (#00574b) al pasar el ratón sobre él, proporcionando retroalimentación visual.
    - Estilos para resultados
      - Área de resultados: Se utiliza un tamaño de fuente mayor y se alinean los textos al centro. Los mensajes de éxito se presentan en verde (#388e3c), mientras que los mensajes de error se muestran en rojo (#d32f2f), facilitando la identificación de la situación.


Registro de usuario
Nombre de usuario (5-15 caracteres):
asdf
Validar
Error: El nombre de usuario debe tener entre 5 y 15 caracteres.


Registro de usuario
Nombre de usuario (5-15 caracteres):
usuariodeprueba
Validar
Nombre de usuario válido.
¡Bienvenido!

Ejercicio 29: Crea una página web utilizando HTML5, CSS3 y JavaScript que permita al usuario subir un archivo PDF a través de un formulario. El archivo debe ser validado para asegurarse de que es un PDF y que su tamaño no exceda los 2MB. Utiliza la validación nativa que ofrece el atributo accept="application/pdf" en HTML5 para que el fichero a subir sea del tipo PDF. La validación de tamaño se debe hacer manualmente a través de código JavaScript.

Requisitos del formulario:

- El formulario debe permitir al usuario seleccionar un archivo mediante un elemento de tipo file.
- El atributo accept="application/pdf" se utilizará en el elemento <input> para validar que el fichero sea del tipo PDF. La validación del tamaño de archivo se hará en JavaScript.
- El formulario deberá tener:
    - Un título claro que indique la acción: "Sube tu archivo PDF".
    - Un botón de "Subir" para enviar el archivo.
    - Un área para mostrar mensajes de error en caso de que el archivo no cumpla con los requisitos.

Especificaciones de la página web:

- Estructura del archivo HTML (index.html):
    - El archivo index.html deberá contener la estructura base de un documento HTML5 y enlazar los archivos estilos.css y javascript.js para los estilos y la lógica, respectivamente.
    - Dentro del <body>, el formulario estará envuelto en un contenedor (div) que lo centrará en la pantalla. Este contenedor tendrá un título (<h1>), un <input type="file"> para subir el archivo, un botón para enviar el formulario (<button>), y un área (<p> o) para mostrar mensajes de error si se produce alguna validación fallida.
- Estilo de la página (estilos.css):
    - Paleta de colores: La página deberá tener un estilo moderno y cálido. Se utilizará una paleta con tonos crema, melocotón y naranja para crear un ambiente acogedor y amigable:
        - Fondo de la página: Un color crema suave (#FFF3E0) para no distraer y mantener la calidez.
        - Contenedor del formulario: Un tono melocotón claro (#FFE0B2) para resaltar el área del formulario de manera sutil.
        - Texto principal: Marrón oscuro (#4E342E) para mantener la legibilidad y complementar la calidez de la paleta.
    - Botones y acciones:
        - Botón de envío: Un naranja (#F4511E) que cambia a un tono más oscuro (#D84315) cuando se pasa el mouse para indicar interactividad.
        - Mensajes de error: Rojo (#D32F2F) para que destaquen y se entienda claramente cuando hay un problema.
    - Estilo moderno:
        - Se aplicarán bordes redondeados, sombras suaves y transiciones para los elementos interactivos (botones) para darle un aspecto profesional y limpio.
        - Se centrará el formulario en la página utilizando técnicas de flexbox para que siempre esté visible de manera óptima en cualquier tamaño de pantalla.


Sube tu archivo PDF
Elegir archivo
No se ha seleccionado ningún archivo
Subir


Sube tu archivo PDF
Elegir archivo
T05_Manual_Ref..._0_Oracle11g.pdf
Subir
El archivo debe ser menor de 2MB.

Ejercicio 30: Crea una página web utilizando HTML5, CSS3 y JavaScript que simula un formulario de registro para un usuario. El formulario debe contener campos para introducir un nombre de usuario, una contraseña y una confirmación de la contraseña. Se debe implementar una funcionalidad en JavaScript que valide que las contraseñas ingresadas en los dos campos coincidan. Si no coinciden, el formulario mostrará un mensaje de error adecuado sin enviar los datos.

La estructura del proyecto debe contener tres archivos principales:

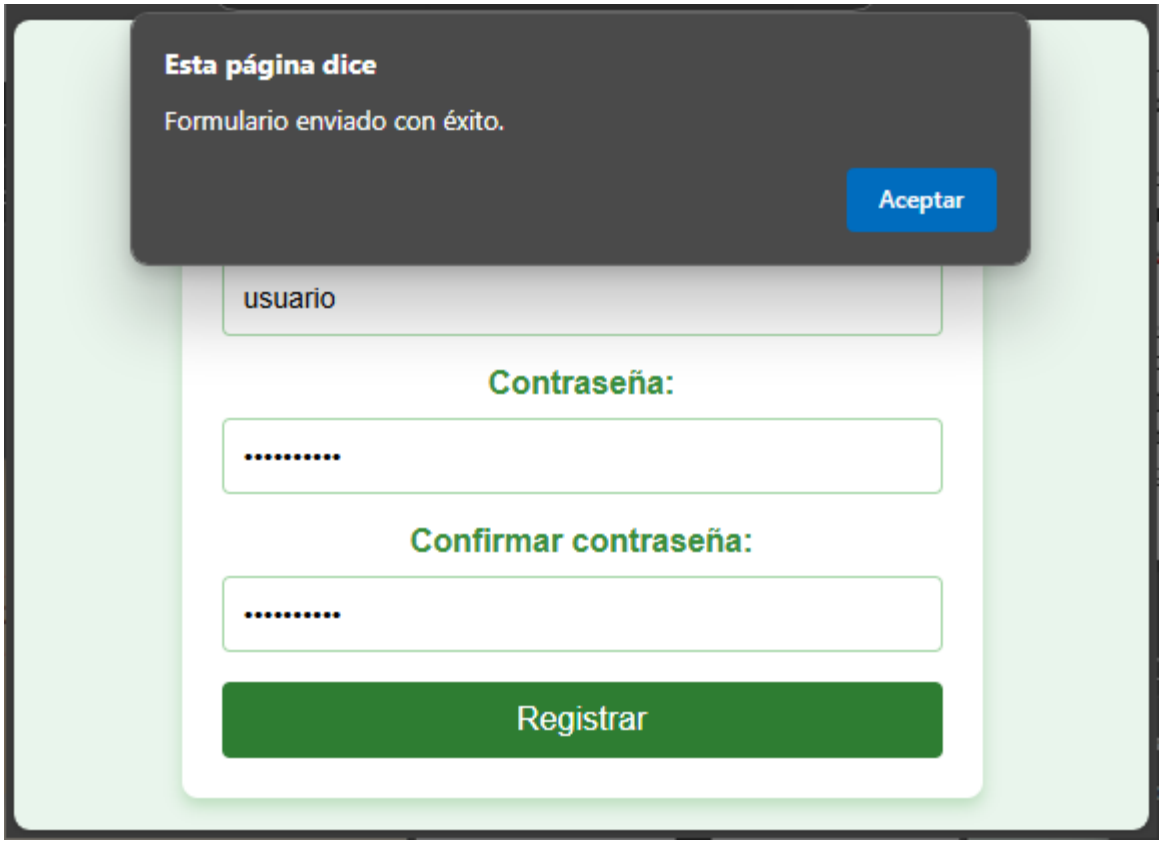
- index.html (estructura HTML del formulario).
- estilos.css (estilos para la apariencia del formulario y la página).
- javascript.js (ubicado en una carpeta llamada js, contiene la lógica de validación de contraseñas).

Especificaciones de la página web:

- Estilo de la página (estilos.css):
  - Paleta de colores:
    - Verde claro (#e9f5ec): Usado como fondo de la página para dar un tono suave y relajante.
    - Verde principal (#43a047): Utilizado en el botón principal para resaltar la acción de envío.
    - Verde medio (#388e3c): Aplicado en las etiquetas y bordes para mantener un contraste equilibrado y moderno.

- Verde oscuro (#2e7d32): Se usa en el título y en el botón al pasar el cursor para marcar jerarquía visual.
- Rojo (#d32f2f): Empleado para los mensajes de error, resaltando visualmente cuando ocurre un problema.

- Estilos generales:
  - Body: Se establece un fondo verde claro y se centra el contenedor principal con flexbox para asegurar una buena visualización en pantallas de diferentes tamaños.
  - Container: Un contenedor con bordes redondeados y una sombra verde suave para un aspecto moderno y limpio.
  - Campos de entrada (Inputs): Con bordes suaves en tonos verdes claros y un padding generoso para facilitar la lectura y la interacción.
  - Botón: Diseño interactivo que cambia de color cuando se pasa el cursor por encima.



Ejercicio 31: Crea una página web utilizando HTML5, CSS3 y JavaScript que simula un formulario de registro para un usuario. El formulario debe contener dos campos: uno para introducir un nombre de usuario y otro para ingresar la edad del usuario. La edad debe ser validada para que esté comprendida entre 18 y 65 años. Esta validación no debe realizarse utilizando los atributos min y max que proporciona el elemento <input> en HTML5, sino que se deberá implementar la validación mediante JavaScript.

El proyecto se estructurará en tres archivos principales:

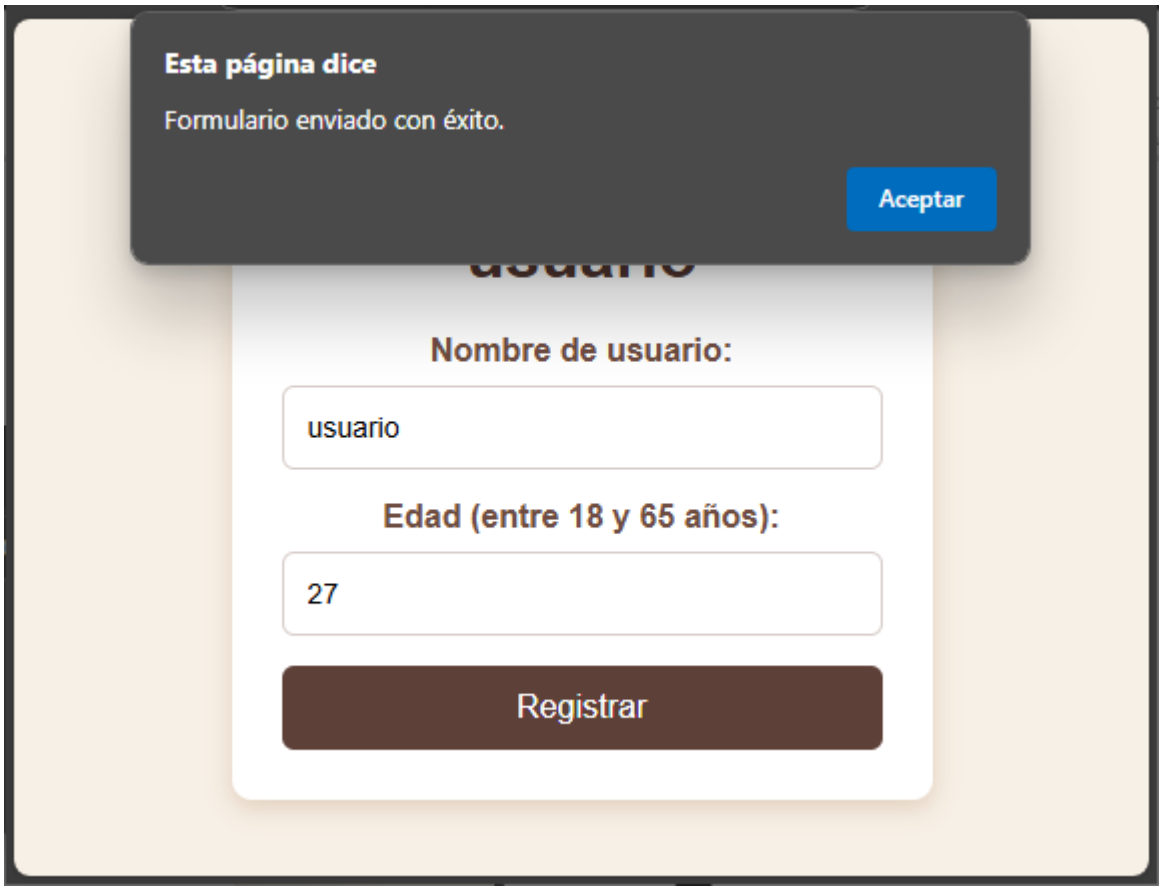
- index.html (estructura HTML del formulario).
- estilos.css (estilos para la apariencia del formulario y la página).
- javascript.js (ubicado en una carpeta llamada js, contiene la lógica de validación de la edad).

Especificaciones de la página web:

- Estilo de la página (estilos.css):
  - Paleta de colores:
    - Marrón claro (#f7f0e6): Utilizado como fondo de la página para crear un ambiente cálido y acogedor.
    - Marrón medio (#8d6e63): Utilizado en el botón principal para destacar la acción de envío.

- Marrón oscuro (#5d4037): Aplicado en el botón al pasar el cursor para mostrar interactividad.
- Marrón suave (#d7ccc8): Se usa en los bordes de los inputs para mantener un diseño limpio y suave.
- Rojo (#d32f2f): Se utiliza para los mensajes de error, asegurando que el usuario detecte visualmente cualquier problema.

- Estilos gGenerales:
  - Body: Se establece un fondo marrón claro y se centra el contenedor principal con flexbox para asegurar una buena visualización en pantallas de diferentes tamaños.
  - Container: Un contenedor con bordes redondeados y una sombra marrón suave para un aspecto limpio y profesional.
  - Campos de entrada (Inputs): Con bordes suaves en tonos marrones claros y un padding adecuado para facilitar la lectura y la interacción.
  - Botón: Diseño interactivo que cambia de color cuando se pasa el cursor por encima.


Esta página dice
Formulario enviado con éxito.
Aceptar
Nombre de usuario:
usuario
Edad (entre 18 y 65 años):
27
Registrar