1. Validación de un formulario

1.1. Repositorio completo del proyecto

(Ver Proyecto)
Repositorio.

1.2. Formulario con validaciones - Parte 1

(Ver Proyecto)
Maquetación.
novalidate
Propiedad de los elementos HTML que es un valor booleano que indica si un elemento **<form>** pasará las Restricciones de Validación.

1.3. Formulario con validaciones - Parte 2

(Ver Proyecto)
Estilos.

1.4. Formulario con validaciones - Parte 3

(Ver Proyecto)
event.preventDefault();
Método del objeto **event**, cancela un evento si este es cancelable, sin detener el resto del funcionamiento del evento, es decir, puede ser llamado de nuevo. Esto significa que la acción por defecto que pertenece al evento, no ocurrirá. Para las acciones **submit** de los **<form>**, suele utilizarse para evitar que la página se recargue luego de ejecutar el evento.

element.textContent;
element.textContent = texto;
Propiedad que representa el contenido de texto de un elemento y sus descendientes. Puede ser obtenida o seteada.

1.5. Formulario con validaciones - Parte 4

(Ver Carpeta)
Expresiones regulares:
Son patrones que se utilizan para hacer coincidir combinaciones de cadenas de caracteres. En JavaScript, las expresiones regulares también son objetos.

Hay dos formas de crear un objeto de expresión regular:

- Notación literal:

Los parámetros de la notación literal se encierran entre barras y no utilizan comillas:
let er = /ab+c/i; // notación literal
Da como resultado la compilación de la expresión regular cuando se evalúa la expresión. Utiliza la notación literal cuando la expresión regular permanecerá constante.

- Constructor:

Los parámetros de la función constructora no se encierran entre barras, pero utilizan comillas:
let er = new RegExp("ab+c", "i"); // constructor con patrón de cadena como primer argumento
let er = new RegExp(/ab+c/, "i"); // constructor con expresión regular literal como primer argumento

Da como resultado la compilación en tiempo de ejecución de la expresión regular. Utiliza la función constructora cuando sepas que el patrón de expresión regular cambiará, o no conozcas el patrón y lo obtienes de otra fuente.

new RegExp(patrón, flag);
Constructor que crea el objeto que se utiliza para hacer coincidir texto con un patrón. Recibe como argumentos un **patrón**, que es una expresión regular en forma de cadena de texto o notación literal, y una **flag**, que son modificadores que alteran el comportamiento de la expresión regular.

Flags disponibles:

Flag	**Descripción**
“g”	Búsqueda global
“i”	Ignora mayúsculas y minúsculas
“m”	Multilínea, permite que **^** y **$** coincidan en cada línea
“s”	Hace que **.** coincida con saltos de línea
“u”	Soporta Unicode
“y”	Sticky (coincide sólo desde la posición actual en la cadena).

Los patrones o expresiones regulares se pueden utilizar con métodos tanto del constructor como métodos de string:

- Métodos del constructor, RegEx():

.exec(cadena)
Método que ejecuta una búsqueda sobre las coincidencias de una expresión regular en una **cadena** específica. Devuelve el resultado como *array*, o *null*.

.test(cadena)
Método que ejecuta la búsqueda de una ocurrencia entre una expresión regular y una **cadena** especificada. Devuelve *true* o *false*.

- Métodos de string:

.match(regex)
Método que devuelve todas las ocurrencias de una expresión regular (**regex**) dentro de una cadena. Retorna un Array cuyo contenido depende de la presencia de la bandera global (g), o null si no se encuentran coincidencias.

.matchAll(regex)
Método que retorna un iterador de todos los resultados de ocurrencia en una cadena de texto contra una expresión regular (**regex**), incluyendo grupos de captura.

.replace(patrón, reemplazo)
Método que devuelve una nueva cadena con una, algunas, o todas las coincidencias de un **patrón**, siendo cada una de estas coincidencias reemplazadas por un **reemplazo**. El

patrón puede ser una cadena o un objeto **RegExp**, y el reemplazo puede ser una cadena o una función que será llamada para cada coincidencia. Si el patrón es una cadena, sólo la primera coincidencia será reemplazada. La cadena original permanecerá inalterada.

.replaceAll(patrón, reemplazo)
Método que retorna una nueva cadena con todas las coincidencias de un **patrón** reemplazadas por un **reemplazo**. El **patrón** puede ser una cadena o un objeto **RegExp**, y el **reemplazo** puede ser una cadena o una función que será llamada para cada coincidencia. La cadena original permanecerá inalterada.

string.search(regex)
Método que ejecuta una búsqueda que encaje entre una expresión regular (**regex**) y el objeto **string** desde el que se llama.

.split(separador, límite)
Método que divide una cadena de texto en un array de cadenas mediante la separación de la cadena en subcadenas. El parámetro **separador** especifica el carácter a usar para la separación de la cadena. Es tratado como una cadena o como una expresión regular. Si se omite el separador, el array devuelto contendrá un sólo elemento con la cadena completa. El parámetro **límite** es opcional y especifica un límite sobre el número de divisiones a realizar.

1.6. Formulario con validaciones - Parte 5
Subida a GitHub y Netlify

2. Login
2.1. Repositorio completo del proyecto
(Ver Carpeta)
Repositorio.

2.2. Práctica de Login - Parte 1
(Ver Carpeta)
Maquetación.

2.3. Práctica de Login - Parte 2
(Ver Carpeta)
Estilos.

2.4. Práctica de Login - Parte 3
(Ver Carpeta)
Script.
DOMContentLoaded
Evento que se activa cuando el documento **HTML** se ha analizado por completo y todos los scripts se han descargado y ejecutado. No espera a que terminen de cargarse otros elementos, como imágenes, submarcos y scripts asincrónicos.

blur
Evento que es disparado cuando un elemento ha perdido su foco.

submit
Evento que define el objeto que se utiliza para representar el evento de un formulario **HTMLsubmit** . Este evento se activa cuando **<form>** se invoca la acción de envío del formulario.

change
Evento que se activa para los elementos **<input>**, **<select>** y **<textarea>** cuando el usuario modifica el valor del elemento.

string.trim();
Método de cadenas de texto que elimina los espacios en blanco en ambos extremos del string. Los espacios en blanco en este contexto, son todos los caracteres sin contenido (espacio, tabulación, etc.) y todos los caracteres de nuevas líneas.

2.5. Práctica de Login - Parte 4

(Ver Carpeta)
Completo, subida a Github y Netlify.