PRÁCTICA 4.05 Generador de formularios

Normas de entrega

- En cuanto al **código**:
  - en la **presentación interna**, importan los **comentarios**, la claridad del código, la significación de los nombres elegidos; todo esto debe permitir considerar al programa como **autodocumentado**. No será necesario explicar que es un `if` un `for` pero sí su funcionalidad. Hay que comentar las cosas destacadas y, sobre todo, las **funciones** y **clases** empleadas. La ausencia de comentarios será penalizada,
  - en la **presentación externa**, importan las leyendas aclaratorias, información en pantalla y avisos de ejecución que favorezcan el uso de la aplicación,
  - todo el código debe estar situado dentro del evento `window.onload = () => {};` o a través del evento `document.addEventListener("DOMContentLoaded", () => {});` ,
  - si no se especifica lo contrario, la información resultante del programa debe aparecer en la consola del navegador `console.log(información)`,
  - los ejercicios deben realizarse usando ***JavaScript* ES6** y usar el modo estricto (`use strict`) No se podrá utilizar *jQuery* ni cualquier otra biblioteca (si no se especifica lo contrario en el enunciado),
  - para el nombre de **variables**, **constantes** y **funciones** se utilizará *lowerCamelCase,*
  - para la asignación de eventos se utilizará `addEventListener()` indicando sus tres parámetros en su definición,
  - se usarán las funcionalidades `import` y `export` para crear bibliotecas de funciones temáticas (no debe haber declaración de funciones ni objetos en el documento principal).
- En cuanto a la **entrega** de los archivos que componen los ejercicios:
  - todos los ejercicios en **una carpeta** (creando las **subcarpetas** necesarias para documentación anexa como imágenes o estilos) cuyo nombre queda a discreción del discente,
  - el nombre de los ficheros necesarios para resolver el ejercicio será el número de ejercicio que contenga,
  - el código contendrá ejemplos de ejecución, si procede, y
  - la carpeta será comprimida en formato **ZIP** y será subida a **Aules** de forma puntual.

Ejercicio 1 - Generador de formularios

Crea una aplicación que te permitirá generar formularios de manera dinámica. Para ello necesitarás un formulario que realice las siguientes acciones:

- crear un **`<input>`** de tipo texto. Le preguntará al usuario mediante un **`prompt`** qué nombre (atributo **`id`**) tiene el **`<input>`**
- crear un **`<input>`** de tipo contraseña. Le preguntará al usuario mediante un **`prompt`** qué nombre (atributo **`id`**) tiene el **`<input>`**
- crear un **`<textarea>`**. Le preguntará al usuario el nombre (atributo **`id`**) y generará automáticamente uno de cuarenta columnas y cinco filas
- crear un **`<label>`**. Preguntará al usuario a qué **`<input>`** va referido (atributo **`for`**) y cuál es el texto. La función debe comprobar si ese elemento existe en el formulario actual
- crear una imagen. Preguntará al usuario qué ruta tiene la imagen (atributo **`src`**) y su nombre (atributo **`id`**)
- crear un **`<checkbox>`**. Preguntará al usuario el nombre y el valor (atributos **`name`** y **`value`**)
- crear un **`<radio>`**. Preguntará al usuario el nombre y el valor (atributos **`name`** y **`value`**)
- crear un botón (`submit`). Preguntará al usuario el nombre y el valor (atributos **`id`** y `value`)

El programa debe **comprobar que ni el nombre ni el id elegidos por el usuario esté en uso en el formulario** actual. De ser así, debe informar debidamente al ~~idiota~~ usuario.