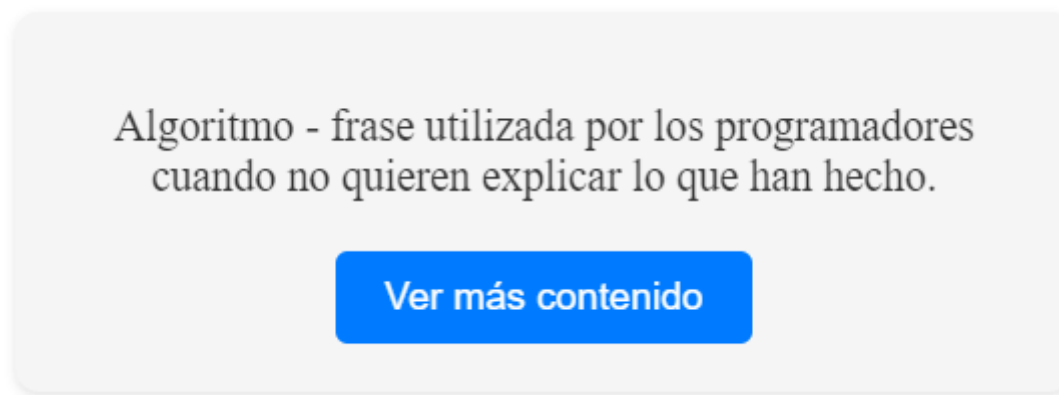


Ejercicios JavaScript Eventos

1. Ejercicio JavaScript tipo examen: Mostrar frases aleatorias mediante CSS

Se proporciona un código HTML y un código CSS para crear un contenedor que vaya mostrando frases de un array de forma aleatoria cuando el usuario interaccione con un botón. Tu objetivo es crear un código JavaScript para que cuando el usuario presione el botón con id texto-informatica aparezca una frase aleatoria (de un array dentro del código JS) al que llamaremos frasesFrikisInformatica, el cual contiene frases frikis o graciosas de informática.



Fichero HTML:

```
<div id="contenido-informatica">  
  
<p id="texto-informatica">Algoritmo - frase utilizada por los programadores cuando no  
quieren explicar lo que han hecho.</p>  
  
<button id="ver-mas-boton">Ver más contenido</button>  
  
</div>
```

Fichero CSS:

```
#contenido-informatica {  
  
background-color: #f5f5f5;  
  
border-radius: 10px;  
  
padding: 20px;  
  
box-shadow: 0px 2px 4px rgba(0, 0, 0, 0.1);  
  
text-align: center;  
  
max-width: 400px;
```

```

margin: 0 auto;
}

#texto-informatica {
font-size: 18px;
color: #333;
margin-bottom: 20px;
}

#ver-mas-boton {
background-color: #007bff;
color: #fff;
border: none;
border-radius: 5px;
padding: 10px 20px;
font-size: 16px;
cursor: pointer;
transition: background-color 0.3s ease;
}

#ver-mas-boton:hover {
background-color: #0056b3;
}

```

El código JavaScript puede ser una función, un fichero .js, etc. Se puede usar el siguiente guión para realizarlo:

- Definir un array con frases frikis o graciosas sobre informática
- Función para cambiar el texto de forma aleatoria
- Obtener un índice aleatorio del array de frases
- Obtener el elemento de texto
- Cambiar el contenido del elemento con una frase aleatoria
- Asignar evento al botón

2. Ejercicio JavaScript: Cambio de color de forma aleatoria

Modifica el código del ejercicio anterior para que además de salir una frase de forma aleatoria, cambie de color el texto y el botón de forma aleatoria, según un array de colores definidos por ti. Por ejemplo, puedes usar el siguiente array de colores:


```
var colores = [  
  "#ff0000", // Rojo  
  "#00ff00", // Verde  
  "#0000ff", // Azul  
  "#ff00ff", // Magenta  
  "#00ffff", // Cyan  
  "#ff8000", // Naranja  
  "#8000ff", // Morado  
  "#00ff80", // Verde claro  
  "#ff0080", // Rosa  
  "#80ff00", // Verde lima  
  "#0080ff", // Azul claro  
  "#ffbf00", // Amarillo oscuro  
  "#bf00ff", // Violeta oscuro  
  "#00ffbf", // Turquesa  
  "#bf00ff", // Púrpura  
  "#00bfff", // Azul claro  
  "#ff00bf", // Magenta oscuro  
  "#80ff80", // Verde claro pastel  
  "#ff80ff", // Rosa pastel  
  "#80ffff" // Cian claro  
];
```


3. Ejercicio JavaScript: Lista de compras


Se proporciona un código HTML y un código CSS para crear una lista de compras como la siguiente:

Lista de Compras

Agregar

Huevos ☒ 

Harina ☒ 

Limón ☒ 

Escribe el código JavaScript necesario para que el formulario funcione correctamente.

Instrucciones:

- Selecciona los elementos necesarios: el input para ingresar los elementos de la lista, el botón para agregar elementos y la lista donde se mostrarán los elementos.
- Implementa una función llamada `addItem` que se ejecutará cuando se haga clic en el botón «Agregar». Esta función debe hacer lo siguiente:
 - Obtener el valor del input y asegurarse de que no esté vacío.
 - Crear un nuevo elemento de lista (``) y establecer su contenido con el valor del input.
 - Crear dos botones dentro del nuevo elemento de lista: uno para marcar como completado y otro para eliminar el elemento.
 - Agregar eventos a los botones de completado y eliminación.
 - Agregar el nuevo elemento a la lista de elementos.
 - Limpiar el input después de agregar el elemento.

- Asocia la función addItem al evento click del botón «Agregar».

Fichero HTML:

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link rel="stylesheet" href="styles.css">

<link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">

<title>Lista de Compras</title>

</head>

<body>

<div id="taskContainer">

<h1>Lista de Compras</h1>

<div class="inputContainer">

<input type="text" id="newItemInput" placeholder="Inserta un nuevo ítem">

<button id="addItemBtn">Agregar</button>

</div>

<ul id="itemsList">

<!-- Aquí se agregarán los ítems -->

</ul>

</div>

<script src="script.js"></script>

</body>

</html>
```

CSS:

```
body {

font-family: Arial, sans-serif;

background-color: #f8f8f8;

margin: 0;
```

```
padding: 0;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

}

#taskContainer {

width: 400px;

background-color: #fff;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

padding: 20px;

}

h1 {

text-align: center;

color: #FF5722;

}

.inputContainer {

display: flex;

align-items: center;

margin-bottom: 10px;

}

input[type="text"] {

flex: 1;

padding: 10px;

margin-right: 10px;

border: 1px solid #ccc;

border-radius: 5px;

font-size: 16px;

}

.inputContainer button {
```

```
padding: 10px;

background-color: #FF5722;

color: #fff;

border: none;

border-radius: 5px;

cursor: pointer;

font-size: 16px;

transition: background-color 0.3s;

}

inputContainer button:hover{

background-color: #E64A19;

}

ul {

list-style-type: none;

padding: 0;

}

li {

padding: 10px;

border-bottom: 1px solid #eee;

display: flex;

align-items: center;

}

li:last-child {

border-bottom: none;

}

.completed {

text-decoration: line-through;

color: #999;

}

.completeBtn, .deleteBtn {

background-color: transparent;
```

```
border: none;

font-size: 16px;

cursor: pointer;

}

.completeBtn:hover, .deleteBtn:hover{

color: #FF5722;

}

.material-icons {

font-size: 24px;

color: black;

cursor: pointer;

transition: color 0.3s;

}

.material-icons:hover{

color: #FF5722;

}

.deleteBtn {

order: 3;

}
```


4. Ejercicio JavaScript: Ocultar y mostrar elementos mediante JavaScript

Implemente la funcionalidad en JavaScript para alternar la visibilidad de la caja al hacer clic en el botón.

Instrucciones:

Utiliza el código HTML proporcionado que contiene una caja y un botón.

```
<div id="container">  
  
<div id="box"></div>  
  
<button id="toggleBtn">Ocultar Caja</button>  
  
</div>
```

Utiliza el código CSS proporcionado para dar estilo a los elementos. No es necesario modificar el CSS.

```
body {  
  
  font-family: Arial, sans-serif;  
  
  display: flex;  
  
  justify-content: center;  
  
  align-items: center;  
  
  margin: 0;  
  
  background-color: #f4f4f4;  
  
}  
  
#container {  
  
  text-align: center;  
  
}  
  
#box {  
  
  width: 100px;  
  
  height: 100px;  
  
  background-color: #3498db;  
  
  margin: 20px auto;  
  
}  
  
button {  
  
  margin: 10px;  
  
  padding: 10px 20px;  
  
  border: none;
```

```
border-radius: 5px;  
cursor: pointer;  
background-color: #2ecc71;  
color: #fff;  
font-size: 16px;  
transition: background-color 0.3s;  
}  
button:hover{  
background-color: #27ae60;  
}
```

Pasos:

- Crea una función llamada toggleBox que se activará al hacer clic en el botón. Esta función debe alternar la visibilidad de la caja cambiando el valor de la propiedad display entre 'none' y 'block'.
- La función también debe cambiar el texto del botón entre «Ocultar Caja» y «Mostrar Caja» según corresponda.

Agrega un evento al botón para que llame a la función toggleBox cuando se haga clic en él.

Verifica que el programa funcione correctamente. Al hacer clic en el botón, la caja debe desaparecer o aparecer según su estado actual, y el texto del botón debe cambiar en consecuencia.

5. Ejercicio JavaScript: Agregar tareas a un listado mediante

Implementa la funcionalidad de agregar tareas a una lista cuando se hace clic en el botón «Agregar tarea».

Lista de tareas

Agregar tarea

Tarea 1

X

Tarea 2

X

Tarea 3

X

Instrucciones:

Utiliza el código HTML proporcionado que contiene un input para insertar nuevas tareas y un botón para agregarlas, junto con una lista (ul) donde se mostrarán las tareas.

```
<div id="taskContainer">  
  
<h1>Lista de tareas</h1>  
  
<div class="inputContainer">  
  
<input type="text" id="newTaskInput" placeholder="Inserta una nueva tarea">  
  
<button id="addTaskBtn">Agregar tarea</button>  
  
</div>  
  
<ul id="taskList">  
  
<!-- Aquí se agregarán las tareas -->  
  
</ul>  
  
</div>
```

Utiliza el código CSS proporcionado para dar estilo a los elementos.

```
body {  
  
  font-family: Arial, sans-serif;  
  
  background-color: #f4f4f4;  
  
  margin: 0;  
  
  padding: 0;  
  
  display: flex;  
  
  justify-content: center;  
  
  align-items: center;  
  
  margin: 20px;  
  
}  
  
#taskContainer {  
  
  width: 400px;  
  
  background-color: #fff;  
  
  border-radius: 8px;  
  
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);  
  
  padding: 20px;  
  
}  
  
h1 {  
  
  text-align: center;  
  
}  
  
.inputContainer {  
  
  display: flex;  
  
  align-items: center;  
  
  margin-bottom: 10px;  
  
}  
  
input[type="text"] {  
  
  flex: 1;  
  
  padding: 10px;  
  
  margin-right: 10px;  
  
  border: 1px solid #ccc;  
  
  border-radius: 5px;  
  
  font-size: 16px;
```

```
}  
  
button {  
  
padding: 10px 20px;  
  
background-color: #4caf50;  
  
color: #fff;  
  
border: none;  
  
border-radius: 5px;  
  
cursor: pointer;  
  
font-size: 16px;  
  
transition: background-color 0.3s;  
  
}  
  
button:hover {  
  
background-color: #45a049;  
  
}  
  
ul {  
  
list-style-type: none;  
  
padding: 0;  
  
}  
  
li {  
  
padding: 10px;  
  
border-bottom: 1px solid #eee;  
  
display: flex;  
  
justify-content: space-between;  
  
align-items: center;  
  
}  
  
li:last-child {  
  
border-bottom: none;  
  
}  
  
.deleteBtn {  
  
background-color: #dc3545;  
  
color: #fff;  
  
border: none;  
  
border-radius: 5px;
```

```
padding: 5px 10px;

cursor: pointer;

transition: background-color 0.3s;

}

.deleteBtn:hover{

background-color: #d32f2f;

}
```

Completa el código JavaScript para agregar tareas a la lista cuando se hace clic en el botón «Agregar tarea»:

- Seleccionar elementos
- Función para agregar una nueva tarea
- Agregar evento al botón de agregar tarea
 - Cada vez que se agrega una nueva tarea, el campo de texto debe limpiarse.
 - Cada tarea agregada debe tener un botón «X» que, al hacer clic en él, elimine esa tarea de la lista.

Pasos:

- Selecciona los elementos necesarios: el input de nueva tarea, el botón de agregar tarea y la lista de tareas.
- Crea una función llamada addTask que se ejecutará cuando se haga clic en el botón de agregar tarea.
- Dentro de la función addTask:
 - Obten el valor del input de nueva tarea.
 - Verifica que el campo de texto no esté vacío.
 - Si el campo no está vacío:
 - Crea un nuevo elemento de lista (li) y establece su texto como el valor de la nueva tarea.
 - Crea un botón para eliminar la tarea (button) con el texto «X».
 - Agrega un evento al botón de eliminar para que, al hacer clic en él, se elimine la tarea correspondiente.
 - Agrega la clase deleteBtn al botón de eliminar.
 - Agrega el botón de eliminar al elemento de lista.
 - Agrega el elemento de lista a la lista de tareas.
 - Limpia el campo de texto.
- Agrega un evento al botón de agregar tarea para que llame a la función addTask cuando se haga clic en él.

6. Ejercicio JavaScript: Selección aleatoria de nombres

Se proporciona un archivo HTML y CSS que contiene un formulario para insertar nombres de personas o cosas. Implementa la funcionalidad en JavaScript para agregar nombres a una lista y luego seleccionar aleatoriamente uno de ellos, tal y como se muestra a continuación:

Selección aleatoria

Agregar

Iván

Patricia

Andrés

Manuel - Se ha elegido: Manuel

Seleccionar aleatoriamente

Instrucciones:

Utiliza el código HTML proporcionado que contiene un input para insertar nombres, un botón para agregarlos, y una lista donde se mostrarán los nombres.

```
<div id="container">
```

```
<h1>Selección aleatoria</h1>
```

```
<input type="text" id="inputName" placeholder="Inserta un nombre">
```

```
<button id="addBtn">Agregar</button>
```

```
<ul id="nameList"></ul>
```

```
<button id="randomBtn">Seleccionar aleatoriamente</button>
```

```
</div>
```

Utiliza el código CSS proporcionado para dar estilo a los elementos. No es necesario modificar el CSS.

```
body {  
  
  font-family: Arial, sans-serif;  
  
  display: flex;  
  
  justify-content: center;  
  
  align-items: center;  
  
  margin: 0;  
  
  background-color: #f9f9f9;  
  
}  
  
#container {  
  
  text-align: center;  
  
  background-color: #fff;  
  
  border-radius: 10px;  
  
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);  
  
  padding: 20px;  
  
  margin: 20px;  
  
}  
  
h1 {  
  
  margin-top: 0;  
  
  color: #333;  
  
}  
  
input[type="text"] {  
  
  padding: 10px;  
  
  border: 1px solid #ccc;  
  
  border-radius: 5px;  
  
  font-size: 16px;  
  
  margin-bottom: 10px;  
  
  width: 200px;  
  
}  
  
button {  
  
  padding: 10px 20px;  
  
  border: none;
```



```
border-radius: 5px;

cursor: pointer;

background-color: #4caf50;

color: #fff;

font-size: 16px;

margin: 10px;

transition: background-color 0.3s;

}

button:hover{

background-color: #45a049;

}

ul{

list-style: none;

padding: 0;

margin-top: 20px;

text-align: left;

}

li{

padding: 10px;

border-bottom: 1px solid #eee;

}

li:last-child{

border-bottom: none;

}

.randomBtn{

background-color: #2196f3;

}

.randomBtn:hover{

background-color: #1e87db;

}

.selected{

background-color: #ffff00;

}
```

```
.selected: hover {  
  
  background-color: #ffff00;  
  
}
```

Pasos:

- Crea un evento que se active al hacer clic en el botón «Agregar». Este evento debe llamar a una función que obtenga el valor del input, agregue el nombre a una lista en memoria y luego actualice la lista visible.
- Crea una función para renderizar los nombres en la lista. Esta función debe actualizar el contenido del elemento con los nombres almacenados en la lista en memoria.
- Crea un evento que se active al hacer clic en el botón «Seleccionar aleatoriamente». Esta función debe seleccionar aleatoriamente un nombre de la lista en memoria y resaltarlo en la lista visible. Además, se debe mostrar un mensaje indicando que se ha elegido ese nombre.

Verifica que el programa funcione correctamente. Al hacer clic en el botón «Seleccionar aleatoriamente», debe aparecer un nombre resaltado en la lista con el mensaje indicando que ha sido seleccionado aleatoriamente.