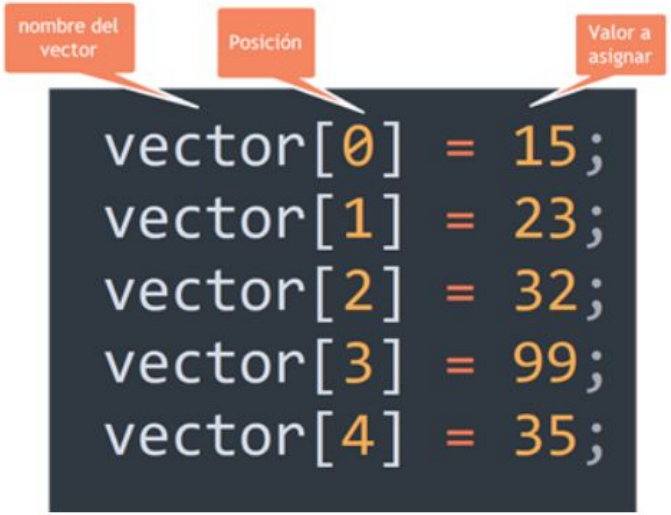
Argentina Programa

Clase 10: Programación Web

EJERCICIOS JAVA

¿COMO CARGAR Y RECORRER UN VECTOR?



Un vector que ya se encuentra cargado o con valores asignados (Ej: imagen 1), puede ser recorrido. Este recorrido se lleva a cabo, tanto para mostrar los valores que contiene el vector como para utilizarlos en caso de que sean necesarios. Para realizar este recorrido la mejor opción es utilizar la **estructura repetitiva for**.

Los vectores se recorren SIEMPRE de **manera secuencial**, es decir, posición a posición según un determinado orden que se establezca.

EJERCICIOS JAVA

¿COMO CARGAR Y RECORRER UN VECTOR?

Índice en el que inicia el vector

Obtenemos la longitud del vector para saber hasta donde recorrer

Indicamos de cuánto en cuánto aumentará el índice

Muestro por pantalla en que posición estoy

```
for (int i=0; i<vector.lenght; i++) {

    System.out.println("Estoy en la posición " + i);
    System.out.println("Contiene el número " + vector[i])
}
```

Muestro por pantalla el valor que contiene mi

EJERCICIOS JAVA

¿COMO CARGAR Y RECORRER UNA MATRIZ?

```java
no usages
public class Main {

    no usages
    public static void main(String[] args) {
        Scanner entrada = new Scanner(System.in);
        int[][] matriz = new int[4][4];

        // Carga de la matriz por teclado
        for (int i = 0; i < 4; i++) {
            for (int j = 0; j < 4; j++) {
                System.out.print("Ingrese el valor de la posición [" + i + "][" + j + "]: ");
                matriz[i][j] = entrada.nextInt();
            }
        }

        // Recorrido de la matriz
        System.out.println("Los valores de la matriz son:");
        for (int i = 0; i < 4; i++) {
            for (int j = 0; j < 4; j++) {
                System.out.print(matriz[i][j] + "\t");
            }
            System.out.println();
        }
    }
}
```

EJERCICIOS JAVA

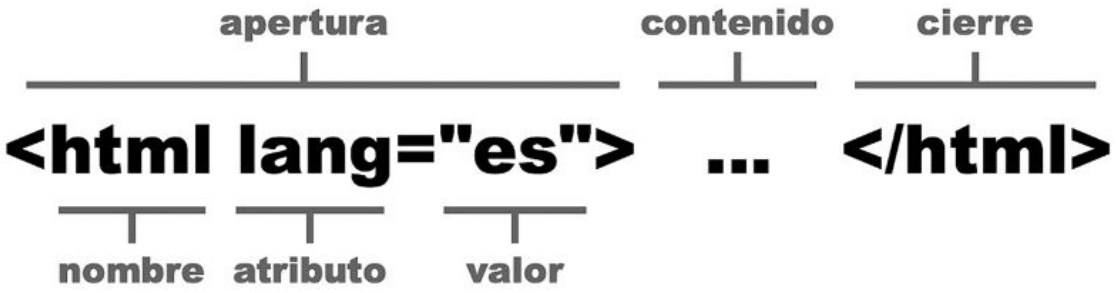
RECORRIDO Y CARGA DE VECTORES Y MATRICES

1. Cargar un vector de 10 elementos y mostrarlos en pantalla:
2. Cargar una matriz de 3x3 y mostrarla en pantalla:
3. Cargar un vector de 5 elementos y encontrar el elemento máximo y mínimo:
4. Cargar una matriz de 3x3 y encontrar el elemento máximo y mínimo:

HTML

HTML (HyperText Markup Language) es un lenguaje compuesto por un grupo de etiquetas definidas con un nombre rodeado de paréntesis angulares.

Los paréntesis angulares delimitan la etiqueta y el nombre define el tipo de contenido que representa.



¿QUÉ SON LAS ETIQUETAS HTML?

Las etiquetas HTML son códigos que se utilizan para definir la estructura y el contenido de una página web. Cada etiqueta se utiliza para definir un elemento específico de la página, como un título, un párrafo, una imagen, un enlace, etc.

¿QUÉ SON LAS ELEMENTOS HTML?

Los elementos HTML son los elementos que se definen con las etiquetas HTML. Por ejemplo, la etiqueta <p> define un párrafo, la etiqueta <img> define una imagen, la etiqueta <a> define un enlace, etc. Cada elemento puede tener diferentes atributos que se utilizan para definir su comportamiento y apariencia..

Anatomía de una etiqueta HTML

1. **La etiqueta de apertura**: es el nombre de la etiqueta, como <p>, <img>, <a>, etc. Esta etiqueta se utiliza para definir el inicio del elemento.
2. **Atributos:** son valores que se utilizan para definir las características del elemento, como el tamaño de una imagen, el destino de un enlace, etc. Los atributos se definen dentro de la etiqueta de apertura.
3. **Contenido**: es el contenido que se encuentra dentro del elemento, como el texto de un párrafo, la fuente de una imagen, etc.
4. **La etiqueta de cierre:** es la misma que la etiqueta de apertura, pero se incluye una barra diagonal antes del nombre de la etiqueta. Esta etiqueta se utiliza para definir el final del elemento.

HTML

```
<!DOCTYPE html>
<html lang="es">
    <head>
      <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width,
initial-scale=1.0">

            <title>Mi primer documento HTML</title>
    </head>
    <body>
            <p>HOLA MUNDO!</p>
    </body>
```

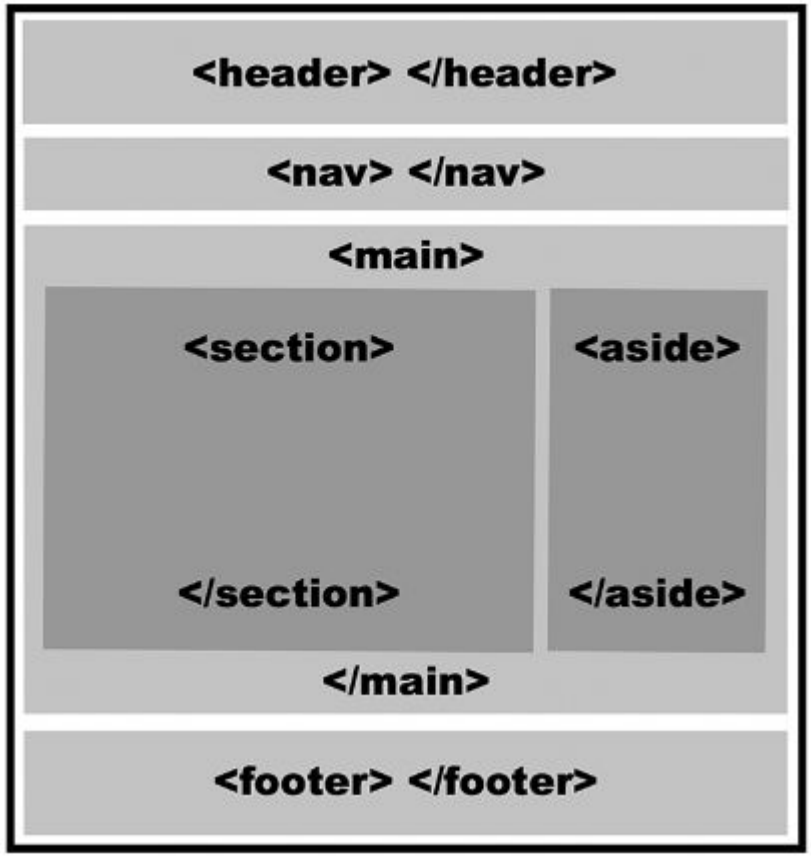
HTML

```
<!DOCTYPE html>
<html lang="es">
    <head>
        <title>Este texto es el título del documento</title>
        <meta charset="utf-8">
        <meta name="description" content="Este es un documento HTML5">
        <meta name="keywords" content="HTML, CSS, JavaScript">
        <link rel="stylesheet" href="misestilos.css">
    </head>
    <body>
    </body>
</html>
```

HTML

Cabecera
Barra de Navegación
Información Principal
Barra Lateral
Barra Institucional

HTML


<header> </header>
<nav> </nav>
<main>
<section>
</section>
<aside>
</aside>
</main>
<footer> </footer>

HTML

```
<!DOCTYPE html>
<html lang="es">
	<head>
		<title>Este texto es el título del documento</title>
	</head>
	<body>
		<header>
			Este es el título
		</header>
		<nav>Principal | Fotos | Videos | Contacto</nav>
		<main>
			<section>
				<article> Este es el texto de mi primer artículo</article>
				<article> Este es el texto de mi segundo artículo</article>
			</section>
		<aside>
			Cita del artículo uno
		</aside>
		</main>
			<footer>  &copy; Derechos Reservados 2016 </footer>
	</body>
</html>
```

ETIQUETAS DE ENCABEZADO

Las etiquetas de encabezado (**<h1>, <h2>, <h3>, <h4>, <h5>, <h6>**) son utilizadas para definir los encabezados o títulos de una página web. Estas etiquetas se utilizan para organizar y jerarquizar la información de una página.

La etiqueta **<h1>** se utiliza para el título principal de la página y es la más importante de todas. La etiqueta **<h2>** se utiliza para los títulos de sección, que son menos importantes que el título principal. Las etiquetas **<h3>** a **<h6>** se utilizan para los títulos de subsecciones y son aún menos importantes que los títulos de sección.

ETIQUETAS DE ENCABEZADO

```
<!DOCTYPE html>
<html>
<head>
	<title>Ejemplo de etiquetas de encabezado</title>
</head>
<body>
	<h1>Título principal</h1>
	<h2>Título de sección</h2>
	<p>Contenido de la sección</p>
	<h3>Título de subsección</h3>
	<p>Contenido de la subsección</p>
	<h4>Título de sub-subsección</h4>
	<p>Contenido de la sub-subsección</p>
</body>
</html>
```

ETIQUETAS DE TEXTO

Las etiquetas de texto en HTML se utilizan para dar formato y estructura al contenido textual de una página web. Las etiquetas de texto más comunes son:

- **<p>:** se utiliza para definir un párrafo.
- ****: se utiliza para enfatizar una palabra o frase. El texto entre las etiquetas se muestra en cursiva por defecto.
- ****: se utiliza para enfatizar una palabra o frase de manera más fuerte que con la etiqueta ****. El texto entre las etiquetas **** se muestra en negrita por defecto.
- ****: se utiliza para aplicar estilos o atributos específicos a una sección del texto. Es una etiqueta genérica que se puede utilizar para cualquier tipo de texto.

ETIQUETAS DE TEXTO

```
<!DOCTYPE html>
<html>
<head>
	<title>Ejemplo de etiquetas de texto</title>
</head>
<body>
	<p>Este es un párrafo de texto.</p>
	<p>Este es otro párrafo de texto con <em>una palabra enfatizada</em> y <strong>otra palabra enfatizada en negrita</strong>.</p>
	<p>Este es un <span style="color: blue;">texto con estilo</span>.</p>
</body>
</html>
```

¿QUÉ ES UN DIV?

La etiqueta ** ** es una etiqueta de tipo contenedor en HTML que se utiliza para crear secciones o bloques de contenido en una página web. Esta etiqueta no tiene una funcionalidad específica por sí misma, sino que se utiliza para agrupar elementos relacionados en una sección lógica de la página.

La etiqueta ** ** se utiliza comúnmente para agrupar elementos HTML relacionados y aplicarles estilos CSS en conjunto. Por ejemplo, se puede utilizar para agrupar elementos de navegación, encabezados, contenidos, pie de página, etc.

ETIQUETAS DE IMAGEN

Las etiquetas de imagen en HTML se utilizan para insertar imágenes en una página web. La etiqueta más común para las imágenes es la etiqueta <img>. A continuación, se explica cómo utilizarla y los atributos que se pueden utilizar con ella:

- **<img>:** se utiliza para insertar una imagen en una página web. Esta etiqueta no tiene etiqueta de cierre y se utiliza en forma de una etiqueta única. Los atributos más comunes de la etiqueta <img> son:
  - **src**: es el atributo requerido que especifica la URL de la imagen que se desea mostrar.
  - **alt**: es un atributo opcional que se utiliza para proporcionar una descripción textual de la imagen para usuarios que no pueden ver la imagen o para motores de búsqueda que indexan la página.
  - **width** y **height**: son atributos opcionales que se utilizan para especificar el ancho y alto de la imagen en píxeles.
  - **title**: es un atributo opcional que se utiliza para proporcionar un texto que aparecerá cuando el usuario pase el cursor sobre la imagen.

HTML

ETIQUETAS DE IMAGEN

```
<!DOCTYPE html>
<html>
<head>
	<title>Ejemplo de etiquetas de imagen</title>
</head>
<body>
	<img src="ruta/a/la/imagen.jpg" alt="Descripción de la imagen">
</body>
</html>
```

ETIQUETAS DE LISTA

Las etiquetas de lista en HTML se utilizan para crear listas de elementos. Hay dos tipos de listas principales: listas no ordenadas (**<ul>**) y listas ordenadas (**<ol>**). Los elementos de la lista se crean utilizando la etiqueta **<li>**. A continuación, se explica cómo utilizar estas etiquetas para crear listas en una página HTML:

- **<ul>**: se utiliza para crear una lista no ordenada. Los elementos de la lista se crean utilizando la etiqueta **<li>**.
- **<ol>:** se utiliza para crear una lista ordenada. Los elementos de la lista se crean utilizando la etiqueta **<li>**.
- **<li>**: se utiliza para crear un elemento de lista.

ETIQUETAS DE LISTA

```
<!DOCTYPE html>
<html>
<head>
	<title>Ejemplo de etiquetas de lista</title>
</head>
<body>
	<h2>Lista no ordenada</h2>
	<ul>
		<li>Elemento 1</li>
		<li>Elemento 2</li>
		<li>Elemento 3</li>
	</ul>

	<h2>Lista ordenada</h2>
	<ol>
		<li>Primer elemento</li>
		<li>Segundo elemento</li>
		<li>Tercer elemento</li>
	</ol>
</body>
</html>
```

ETIQUETAS DE ENLACE

Las etiquetas de enlace se utilizan en HTML para crear hipervínculos que permiten a los usuarios navegar entre diferentes páginas web. La etiqueta principal para crear un enlace es la etiqueta **<a>**

```
<!DOCTYPE html>
<html>
<head>
	<title>Ejemplo de etiquetas de enlace</title>
</head>
<body>
	<h2>Enlace a otra página web</h2>
	<p>Visita <a href="https://www.ejemplo.com" target="_blank">ejemplo.com</a> para obtener más
información.</p>
</body>
</html>
```

ETIQUETAS DE ENLACE

```
<!DOCTYPE html>
<html>
<head>
	<title>Ejemplo de etiquetas de enlace</title>
</head>
<body>
	<h2>Enlace a otra página en el mismo sitio web</h2>
	<p>Visita <a href="pagina2.html">Página 2</a> para obtener más información.</p>
</body>
</html>
```

ETIQUETAS DE ENLACE

Las etiquetas de formulario se utilizan en HTML para crear formularios que permiten a los usuarios enviar información al servidor web. Las etiquetas principales para crear formularios son la etiqueta **<form>** y la etiqueta **<input>**, pero también hay otras etiquetas que se utilizan comúnmente en los formularios, como la etiqueta **<label>**, la etiqueta **<select>** y la etiqueta **<option>**.

ETIQUETAS DE ENLACE

```
<!DOCTYPE html>
<html>
<head>
    <title>Ejemplo de etiquetas de formulario</title>
</head>
<body>
    <h2>Formulario de ejemplo</h2>
    <form>
        <label for="nombre">Nombre:</label>
        <input type="text" id="nombre" name="nombre">
        <input type="submit" value="Enviar">
    </form>
</body>
</html>
```

HTML

ETIQUETAS DE ENLACE

```
<!DOCTYPE html>
<html>
<head>
	<title>Ejemplo de etiquetas de formulario</title>
</head>
<body>
	<h2>Formulario de ejemplo</h2>
	<form>
		<label for="nombre">Nombre:</label>
		<input type="text" id="nombre" name="nombre"><br><br>
		<label for="edad">Edad:</label>
		<select id="edad" name="edad">
			<option value="18-24">18-24</option>
			<option value="25-34">25-34</option>
			<option value="35-44">35-44</option>
			<option value="45-54">45-54</option>
			<option value="55+">55+</option>
		</select><br><br>
		<input type="submit" value="Enviar">
	</form>
</body>
</html>
```

ETIQUETA: input

- Input significa entrada… y se utiliza para la entrada de datos por parte del usuario. Miremos este ejemplo:

- Donde ingresamos los datos de búsqueda...es un elemento input de HTML !

```
<input type="text" name="busqueda">
```

ETIQUETA: input

- La etiqueta input NO TIENE ETIQUETA DE CIERRE, porque es simplemente un espacio para ingresar datos.
- Puede tener varios atributos, los más frecuentes son:
  - Type (tipo) : indica el tipo de datos que se va a ingresar. Hay muchos! Algunos son:
    - type="text" : para ingresar texto
    - type="password" : oculta los caracteres y se utiliza para las contraseñas
    - type="submit" : para enviar los datos de un formulario
    - type="number" : para ingresar datos numéricos
  - Name (nombre) : cada input debe tener un nombre, en general se lo llama como el dato que van a recibir
  - Value (valor) : el valor inicial del input. Si queremos que se cargue un valor por defecto:

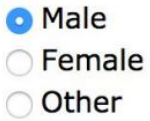
```
<input type="text" name="nombre" value="John">
```

En pantalla:

John

ETIQUETA: radio buttons

- Algo que también es muy frecuente de ver es este estilo de input:



- Para lograr esto deberíamos tener:

```
<input type="radio" name="gender" value="male" checked> Male<br>
<input type="radio" name="gender" value="female"> Female<br>
<input type="radio" name="gender" value="other"> Other
```

- Notemos que los 3 input tienen el mismo nombre pero diferentes valores. Esto es necesario para que el mecanismo de selección pueda funcionar.
- Checked es un atributo, que indica cual de los inputs debe estar marcado por defecto
- **
** no es más que un salto de línea se puede utilizar con todas las etiquetas

ETIQUETA: chekbox

- A menudo tenemos que seleccionar varias opciones de una lista:

  ☐ I have a bike
  ☐ I have a car

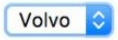
- Usamos los checkboxes cuando las opciones no son excluyentes entre si:

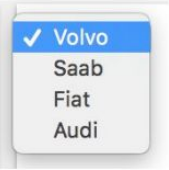
```
<input type="checkbox" name="vehicle1" value="Bike"> I have a bike<br>
<input type="checkbox" name="vehicle2" value="Car"> I have a car
```

- Notemos que a diferencia de los radio buttons, el atributo nombre debe ser distinto para cada input.
- También podemos marcar una opción por defecto con el atributo **checked**

ETIQUETA: dropdown menu

- Un dropdown menu es una lista desplegable:






```
<select name="cars">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi">Audi</option>
</select>
```

- La etiqueta <option> define un elemento de la lista que puede ser seleccionado
- Por defecto, el primer elemento de la lista siempre es seleccionado, si necesitamos que algún otro elemento figure por defecto entonces agregamos el atributo **selected** a la opción que queramos.

ETIQUETA: button

- Cuando necesitamos un botón para nuestra página web, podemos utilizar:

```
<button type="button">Click Me!</button>
```

- Y por pantalla veríamos: Click Me!

- También podemos encontrar botones con esta etiqueta:

```
<input type="submit" value="Submit">
```

- Y por pantalla veríamos: Submit

- Este tipo de botón (submit) suele usarse en formularios… más adelante veremos de que se trata

ETIQUETA: form

- Un formulario está compuesto por otros elementos: inputs, botones, botones de radio, checkboxes, dropdowns, botón para subir archivos… todo depende de que datos necesitamos recibir :)

En el editor:

```
<form action="/action_page.php">
  First name:<br>
  <input type="text" name="firstname" value="Mickey"><br>
  Last name:<br>
  <input type="text" name="lastname" value="Mouse"><br><br>
  <input type="submit" value="Submit">
</form>
```

Los datos de este formulario serán enviados a otro archivo. En este caso el archivo está escrito en el lenguaje de programación PHP (Lenguaje de Backend)

En la pantalla:

First name:

Mickey

Last name:

Mouse

Submit

ETIQUETAS DE TABLA

Las etiquetas de tabla en HTML se utilizan para crear tablas que pueden contener datos y presentarlos de forma estructurada. Las etiquetas principales para crear tablas son la etiqueta **<table>,** la etiqueta **<tr> -table row-** y la etiqueta **<td> -table data-**, pero también hay otras etiquetas que se utilizan comúnmente en las tablas, como la etiqueta **<th> -table head-.**

HTML

ETIQUETAS DE TABLA

```
<!DOCTYPE html>
<html>
<head>
        <title>Ejemplo de etiquetas de tabla</title>
</head>
<body>
        <h2>Tabla de estudiantes</h2>
        <table>
                <tr>
                        <th>Nombre</th>
                        <th>Edad</th>
                        <th>Curso</th>
                </tr>
                <tr>
                        <td>Pablo</td>
                        <td>21</td>
                        <td>Matemáticas</td>
                </tr>
                <tr>
                        <td>Lucía</td>
                        <td>25</td>
                        <td>Historia</td>
                </tr>
        </table>
</body>
</html>
```

EJERCICIOS

1. Crear una página web con una barra de navegación, un título, un párrafo de texto y una imagen. La imagen debe estar centrada en la página y debe tener un tamaño de 300 píxeles de ancho y 200 píxeles de alto.
2. Crear una lista de compras utilizando etiquetas de lista. La lista debe incluir cinco elementos: pan, leche, huevos, queso y fruta. Los elementos deben estar numerados y presentados en una lista ordenada.
3. Crear un enlace a otra página. El enlace debe tener el texto "Haz clic aquí para ir a la página de Google" y debe llevar al usuario a la página de inicio de Google cuando se hace clic en él.

CSS

CSS es un lenguaje que facilita instrucciones que podemos usar para asignar estilos a los elementos HTML, como colores, tipos de letra, tamaños, etc.

Los estilos se deben definir con CSS y luego asignar a los elementos hasta que logramos el diseño visual que queremos para nuestra página.

Por razones de compatibilidad, los navegadores asignan estilos por defecto a algunos elementos HTML.

CSS

```
<!DOCTYPE html>
<html lang="es">
	<head>
		<title>Este texto es el título del documento</title>
		<meta charset="utf-8">
	</head>
	<body>
		<main>
			<section>
				<p style="font-size: 20px;">Mi Texto</p>
			</section>
		</main>
	</body>
</html>
```

CSS

```
<!DOCTYPE html>
<html lang="es">
	<head>
		<title>Este texto es el título del documento</title>
			<meta charset="utf-8">
			<style>
				p {
					font-size: 20px;
				}
			</style>
	</head>
	<body>
		<main>
			<section>
				<p>Mi texto</p>
			</section>
		</main>
	</body>
</html>
```

SINTAXIS DE CSS

- El "selector" especifica a qué elemento HTML se aplicará la regla de estilo.
- Las "propiedades" son los atributos de estilo que se aplicarán al elemento seleccionado.
- Los "valores" son los valores asignados a las propiedades para especificar el estilo deseado.

```
selector {
propiedad: valor;
}
```

```
p {
color: blue;
}
```

SELECTORES CSS

Selector	Ejemplo	Tutorial de aprendizaje de CSS
Selector de tipo	`h1 { }`	Los tipos de selectores
Selector universal	`* { }`	El selector universal
Selector de clase	`.box { }`	Los selectores de clase
Selector de ID	`#unique { }`	Los selectores de ID
Selector de atributo	`a[title] { }`	Los selectores de atributo
Pseudoclase	`p:first-child { }`	Las pseudoclases
Pseudoelemento	`p::first-line { }`	Los pseudoelementos
Operadores de combinación descendentes	`article p`	Operadores de combinación descendentes
Operador de combinación de elementos hijo	`article > p`	Operadores de combinación de elementos hijo
Operador de combinación de elementos hermanos adyacentes	`h1 + p`	Hermanos adyacentes
Operador de combinación general de elementos hermanos	`h1 ~ p`	Hermanos generales

SELECTORES DE ATRIBUTOS

- **Selector de presencia:** se utiliza para seleccionar elementos que contienen ciertos tipos de contenido. El selector de presencia se escribe utilizando dos puntos (::) seguidos del nombre del contenido que se desea seleccionar. Por ejemplo, el selector ::before seleccionará cualquier elemento que tenga un pseudo-elemento ::before asociado.

PSEUDOCLASES Y PSEUDOELEMENTOS

¿Qué es una pseudoclase?

Una pseudoclase es un selector que marca los elementos que están en un estado específico, por ejemplo, los que son el primer elemento de su tipo, o aquellos por los que el cursor les pasa por encima. Tienden a actuar como si hubieras aplicado una clase en una parte determinada del documento y, a menudo, ayudan a reducir el exceso de clases y proporcionan un marcado más flexible y fácil de mantener.

Las pseudoclases son palabras clave que comienzan con dos puntos:

:pseudo-class-name

:hover: solo interviene si el usuario pasa el cursor sobre un elemento, normalmente un enlace.

:focus: solo interviene si el usuario selecciona el elemento con los controles del teclado.

PSEUDOCLASES Y PSEUDOELEMENTOS

¿Qué es un pseudoelemento?

Los pseudoelementos se comportan de manera similar. Sin embargo, actúan como si hubieras añadido un elemento HTML totalmente nuevo en el marcado, en lugar de haber aplicado una clase nueva a los elementos presentes. Los pseudoelementos empiezan con un doble signo de dos puntos ::.

::pseudo-element-name

```
p::first-line {
  font-size: 20px;
}
```

COMBINADORES

Selectores descendientes: Los selectores descendientes son una categoría de selectores en CSS que se utilizan para seleccionar elementos secundarios dentro de un elemento principal. Un selector descendiente selecciona un elemento secundario que está contenido dentro de un elemento primario específico.

```
div span {
  color: blue;
}
```

```
div.contenedor p
span {
  color: red;
}
```

COMBINADORES

Selectores de combinación de elementos hijos: El selector de combinación de elementos hijo es un símbolo de «mayor que» (>), que selecciona solo cuando los selectores identifican elementos que son hijos directos. Los elementos descendientes que se encuentran más abajo en la jerarquía no se seleccionan.

Por ejemplo, para seleccionar solo los elementos <p> que son hijos directos de elementos <article>:

article > p

CASCADA Y HERENCIA

Cascada: En un primer nivel de simplicidad, la cascada en las hojas de estilo significa que el orden de las reglas importa en CSS: cuando dos reglas tienen la misma especificidad, se aplica la que aparece en último lugar en el CSS.

Herencia: algunos valores de las propiedades CSS que se han establecido para los elementos padre los heredan los elementos hijo, pero otros no.

Por ejemplo, si para un elemento se establece el color (color) y el tipo de letra (font-family), cada elemento que se encuentre dentro de él también se mostrará de ese color y con ese tipo de letra, a menos que les se haya aplicado un color y un tipo de letra diferentes directamente.

CASCADA Y HERENCIA

Especificidad: La especificidad es el modo que tiene el navegador de decidir qué regla se aplica si diversas reglas tienen selectores diferentes pero podrían aplicarse a un mismo elemento. Básicamente, la especificidad mide cuán específica es la selección de un selector:

- Un selector de elemento es menos específico (selecciona todos los elementos de aquel tipo que aparecen en la página) por lo que presenta una puntuación más baja en especificidad.
- Un selector de clase es más específico (selecciona solo los elementos de una página que tienen un valor de atributo class dado), y por tanto recibe una puntuación mayor.

This is my heading.

Interactive editor

```
.main-heading {
    color: red;
}

h1 {
    color: blue;
}
```

```
<h1 class="main-heading">This is my heading.</h1>
```

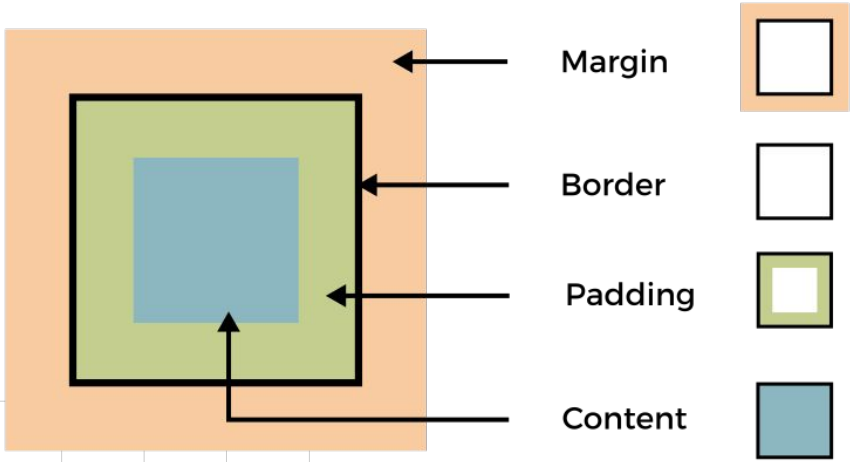
MODELO DE CAJA

El modelo de cajas es un concepto fundamental en CSS que describe cómo se representan los elementos HTML en una página web. Cada elemento HTML se representa como una "caja" rectangular, que incluye el contenido del elemento, el relleno, el borde y el margen.

El modelo de cajas en CSS se compone de cuatro partes:

1. **Contenido:** es la parte del elemento HTML que contiene el texto, las imágenes u otros elementos del contenido.
2. **Relleno:** es un espacio opcional que se coloca entre el contenido y el borde. El relleno ayuda a separar el contenido del borde y proporciona espacio adicional dentro de la caja.
3. **Borde:** es una línea que rodea el contenido y el relleno. El borde se utiliza para resaltar la caja y proporcionar un espacio visualmente separado del contenido.
4. **Margen:** es un espacio opcional que se coloca entre el borde y el elemento adyacente. Los márgenes se utilizan para separar las cajas entre sí y proporcionar espacio adicional alrededor de la caja.

MODELO DE CAJA



- El **borde (border)**, en negro, es el límite que separa el interior del exterior del elemento.
- El **márgen (margin)**, en naranja, es la parte exterior del elemento, por fuera del borde.
- El **relleno (padding)**, en verde, es la parte interior del elemento, entre el contenido y el borde.
- El **contenido**, en azul, es la parte interior del elemento, excluyendo el relleno.

MODELO DE CAJA

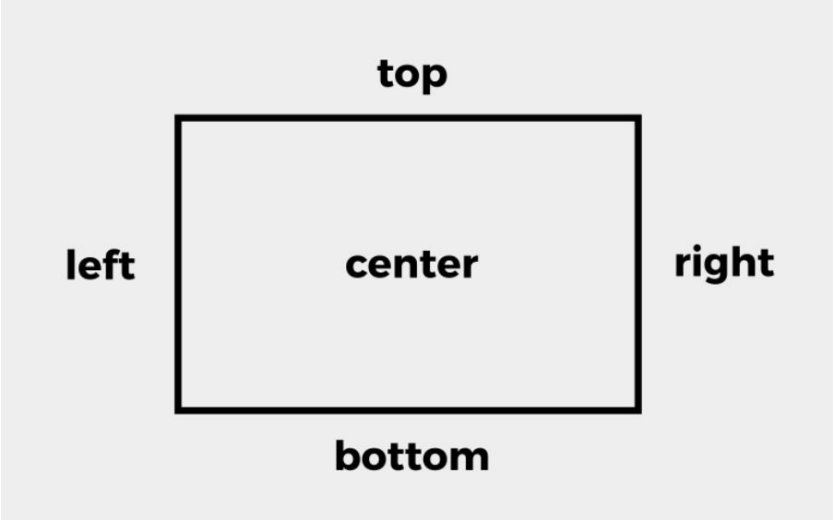
Dimensiones (ancho y alto): Para dar tamaños específicos a los diferentes elementos de un documento HTML, necesitaremos asignarles valores a las propiedades width (ancho) y height (alto)

Propiedad	Valor	Significado
width	auto | SIZE	Tamaño de ancho de un elemento.
height	auto | SIZE	Tamaño de alto de un elemento.

Propiedad	Valor	Significado
max-width	none | SIZE	Ancho máximo que puede ocupar un elemento.
min-width	0 | SIZE	Ancho mínimo que puede ocupar un elemento.
max-height	none | SIZE	Alto máximo que puede ocupar un elemento.
min-height	0 | SIZE	Alto mínimo que puede ocupar un elemento.

MODELO DE CAJA

Zonas de un elemento:



EJERCICIOS

1. Crear un menú de navegación con CSS: Puedes crear un menú de navegación horizontal o vertical con CSS. Utiliza la propiedad display para definir la orientación del menú y utiliza la propiedad background-color para establecer el color de fondo. También puedes utilizar pseudo-clases como :hover para cambiar el estilo de los elementos del menú cuando el usuario pasa el cursor sobre ellos.
2. Crear un botón con CSS: Puedes crear un botón con CSS utilizando la propiedad border-radius para darle una forma redondeada, la propiedad background-color para establecer el color de fondo y la propiedad padding para agregar espacio dentro del botón. También puedes utilizar pseudo-clases como :hover para cambiar el estilo del botón cuando el usuario pasa el cursor sobre él.
3. Crea tres cajas con texto lorem ipsum, una de ellas con borde rojo alineada en el centro de la página, otra de ellas debajo de la primera a la izquierda con fondo verde y la tercera en el lado derecho de la página con texto en negrita y borde amarillo.

JavaScript

JavaScript es un lenguaje de programación que se usa para procesar información y manipular documentos.

Al igual que cualquier otro lenguaje de programación, JavaScript provee instrucciones que se ejecutan de forma secuencial para indicarle al sistema lo que queremos que haga (realizar una operación aritmética, asignar un nuevo valor a un elemento, etc.).

Cuando el navegador encuentra este tipo de código en nuestro documento, ejecuta las instrucciones al momento y cualquier cambio realizado en el documento se muestra en pantalla.

JavaScript

```
<!DOCTYPE html>
<html lang="es">
	<head>
		<meta charset="utf-8">
		<title>JavaScript</title>
	</head>
	<body>
		<section>
			<p onclick="alert('Hizo clic!')">Clic aquí</p>
			<p>No puede hacer clic aquí</p>
		</section>
	</body>
</html>
```

JavaScript

Siguiendo el mismo enfoque que CSS, el código JavaScript se puede incorporar al documento mediante tres técnicas diferentes: el código se puede insertar en un elemento por medio de atributos (En línea), incorporar al documento como contenido del elemento <script> o cargar desde un archivo externo.

SINTAXIS DE JAVASCRIPT

- Normas básicas:
  - 1. No se tienen en cuenta los espacios en blanco y las nuevas líneas (Javascript los ignora)
  - 2. Se distinguen mayúsculas de minúsculas. Si se llegaran a intercambiar las mayúsculas y minúsculas el script no funcionaría.
  - 3. No se define el tipo de las variables. Al crear una variable no se define de que tipo es (por ejemplo, no se define que 3 es un número entero)
  - 4. No es necesario terminar cada sentencia con un punto y coma (;), en Javascript no es obligatorio.
  - 5. Se pueden incluir comentarios.

VARIABLES EN JAVASCRIPT

var: Es la forma tradicional de declarar una variable en JavaScript. Una variable declarada con var puede ser accedida dentro de cualquier alcance en una función, incluso antes de su declaración. Además, su valor puede ser reasignado sin restricciones.

let: Se introdujo en ES6 como una alternativa a var. Las variables declaradas con let están limitadas al alcance del bloque en el que se declaran, lo que significa que no se pueden acceder fuera de ese bloque. Además, no se pueden declarar más de una vez en el mismo alcance.

const: También se introdujo en ES6 como una forma de declarar variables que no cambian de valor. Una vez que se le asigna un valor a una variable const, no se puede reasignar. Además, las variables const también están limitadas al alcance del bloque en el que se declaran.

¿Qué es el Scope?

El ámbito (Scope) se refiere a los lugares dentro de nuestro código en donde las variables están disponibles para su uso.

Cuando una variable tiene un **ámbito global**, significa que está disponible en cualquier lugar de tu programa.

Cuando una variable tiene un **ámbito local**, significa que está disponible solo dentro de la función donde fue creada.

TIPOS DE DATOS

Datos primitivos de JavaScript

Existen seis tipos de datos primitivos, es decir, que no son un objeto. Hay que tener cuidado con este concepto porque JavaScript en muchas situaciones convierte automáticamente los datos primitivos en objetos equivalentes a excepción de null y undefined.

Undefined: representa una variable que no ha sido declarada o a la cual no se le ha asignado un valor.
Boolean: representa un valor lógico y puede tener dos valores, ya sean true o false.
Number: permite representar y manipular valores numéricos como «37» o «-9.25».
String: representa datos textuales (cadenas de caracteres).
BigInt: representa valores numéricos que son demasiado grandes para ser representados por el tipo de dato number.
Symbol: es un valor primitivo único e inmutable.

TIPOS DE DATOS

Otros tipos de datos de JavaScript

Null: representa la ausencia intencional de cualquier valor, un valor nulo o «vacío».
Object: representa una colección de datos definidos y entidades más complejas.
Function: es una forma abreviada para funciones, aunque cada constructor de funciones se de

TYPEOF

El operador **`typeof`** es muy útil porque nos permite consultar fácilmente el tipo de datos que una variable contiene. Esto es importante en JavaScript porque se trata de un lenguaje escrito dinámicamente.
Esto significa que no necesitamos asignar un tipo de datos fijo a una variable cuando la creamos. Al no restringir las variables de esta manera, el tipo de datos que contienen puede cambiar durante la ejecución de un programa.

```
var x = 12345;
console.log(typeof x) // number
x = 'string';
console.log(typeof x) // string
x = { key: 'value' };
console.log(typeof x) // object
```

OPERADORES DE COMPARACIÓN

1. **== (igual a)**: compara dos valores para verificar si son iguales en valor. Si los valores son iguales, devuelve verdadero. De lo contrario, devuelve falso. Sin embargo, este operador no compara los tipos de datos, lo que puede llevar a resultados inesperados.
2. **=== (igualdad estricta):** compara dos valores para verificar si son iguales en valor y tipo de datos. Si los valores son iguales y del mismo tipo de datos, devuelve verdadero. De lo contrario, devuelve falso.
3. **!= (diferente a)**: compara dos valores para verificar si son diferentes en valor. Si los valores son diferentes, devuelve verdadero. De lo contrario, devuelve falso.
4. **!== (diferente estricto)**: compara dos valores para verificar si son diferentes en valor o tipo de datos. Si los valores son diferentes o del tipo de datos diferente, devuelve verdadero. De lo contrario, devuelve falso.
5. **> (mayor que):** compara dos valores para verificar si el primer valor es mayor que el segundo. Si el primer valor es mayor que el segundo, devuelve verdadero. De lo contrario, devuelve falso.
6. **< (menor que):** compara dos valores para verificar si el primer valor es menor que el segundo. Si el primer valor es menor que el segundo, devuelve verdadero. De lo contrario, devuelve falso.
7. **>= (mayor o igual que):** compara dos valores para verificar si el primer valor es mayor o igual que el segundo. Si el primer valor es mayor o igual que el segundo, devuelve verdadero. De lo contrario, devuelve falso.
8. **<= (menor o igual que)**: compara dos valores para verificar si el primer valor es menor o igual que el segundo. Si el primer valor es menor o igual que el segundo, devuelve verdadero. De lo contrario, devuelve falso.

OPERADORES DE COMPARACIÓN

1. **== (igual a)**: compara dos valores para verificar si son iguales en valor. Si los valores son iguales, devuelve verdadero. De lo contrario, devuelve falso. Sin embargo, este operador no compara los tipos de datos, lo que puede llevar a resultados inesperados.
2. **=== (igualdad estricta):** compara dos valores para verificar si son iguales en valor y tipo de datos. Si los valores son iguales y del mismo tipo de datos, devuelve verdadero. De lo contrario, devuelve falso.
3. **!= (diferente a)**: compara dos valores para verificar si son diferentes en valor. Si los valores son diferentes, devuelve verdadero. De lo contrario, devuelve falso.
4. **!== (diferente estricto)**: compara dos valores para verificar si son diferentes en valor o tipo de datos. Si los valores son diferentes o del tipo de datos diferente, devuelve verdadero. De lo contrario, devuelve falso.
5. **> (mayor que):** compara dos valores para verificar si el primer valor es mayor que el segundo. Si el primer valor es mayor que el segundo, devuelve verdadero. De lo contrario, devuelve falso.
6. **< (menor que):** compara dos valores para verificar si el primer valor es menor que el segundo. Si el primer valor es menor que el segundo, devuelve verdadero. De lo contrario, devuelve falso.
7. **>= (mayor o igual que):** compara dos valores para verificar si el primer valor es mayor o igual que el segundo. Si el primer valor es mayor o igual que el segundo, devuelve verdadero. De lo contrario, devuelve falso.
8. **<= (menor o igual que)**: compara dos valores para verificar si el primer valor es menor o igual que el segundo. Si el primer valor es menor o igual que el segundo, devuelve verdadero. De lo contrario, devuelve falso.

OPERADORES ARITMÉTICOS Y LÓGICOS

- + (suma): suma dos valores o concatena dos cadenas de texto.
- - (resta): resta un valor de otro valor.
- * (multiplicación): multiplica dos valores.
- / (división): divide un valor por otro valor.
- % (módulo): devuelve el resto de una división entera entre dos valores.
- ++ (incremento): aumenta el valor de una variable en uno.
- -- (decremento): disminuye el valor de una variable en uno.

- && (y lógico): devuelve verdadero si ambos operandos son verdaderos, de lo contrario, devuelve falso.
- || (o lógico): devuelve verdadero si alguno de los operandos es verdadero, de lo contrario, devuelve falso.
- ! (negación lógica): devuelve el valor opuesto al valor lógico del operando

OPERADOR TERNARIO: se utiliza para tomar una decisión basada en una condición. Si la condición es verdadera, se devuelve el primer valor. De lo contrario, se devuelve el segundo valor.

```
condition ? val1 : val2
```

¿QUÉ ES UNA FUNCIÓN?

En JavaScript, una función es un bloque de código que se puede llamar para realizar una tarea específica. Las funciones en JavaScript se utilizan para dividir el código en partes más pequeñas y reutilizables. Una vez definida una función, se puede llamar en cualquier momento del programa para realizar su tarea específica.

Las funciones en JavaScript se definen utilizando la palabra clave function, seguida de un nombre para la función, una lista de parámetros entre paréntesis (opcional) y un bloque de código entre llaves {} que contiene las instrucciones que deben realizarse cuando se llama la función.

```
function sumar(a, b) {

return a + b;

}
```

FUNCIÓN: PROMPT()

La función prompt() muestra un cuadro de diálogo en la pantalla que permite al usuario ingresar un valor. El cuadro de diálogo contiene un mensaje que se muestra al usuario, seguido de un campo de entrada de texto donde el usuario puede ingresar su respuesta. La función prompt() devuelve el valor ingresado por el usuario como una cadena de texto.

Por ejemplo, el siguiente código pide al usuario su nombre y luego lo muestra en la pantalla:

```
let nombre = prompt("¿Cuál es tu nombre?");

alert("Hola " + nombre + "!");
```

FUNCIÓN: ALERT()

La función alert() muestra un cuadro de diálogo en la pantalla que contiene un mensaje para el usuario. El cuadro de diálogo no contiene un campo de entrada de texto, solo un botón para cerrarlo. La función alert() es útil para mostrar mensajes de información, advertencia o error al usuario.

Por ejemplo, el siguiente código muestra un mensaje de advertencia al usuario si intenta ingresar un número negativo:

```
let num = prompt("Ingresa un número positivo:");
if (num < 0) {
  alert("El número debe ser positivo.");
}
```

EJERCICIOS

1. Crear un programa que pida al usuario su nombre y su edad, y luego muestre un mensaje que diga "Hola [nombre], tienes [edad] años". Para lograr esto, deberás utilizar la función prompt() para obtener la información del usuario y la función alert() para mostrar el mensaje.
2. Crear un programa que calcule el área de un círculo a partir del radio. Para lograr esto, deberás utilizar la fórmula $A = \pi r^2$, donde A es el área, π es la constante matemática pi (aproximadamente 3.14) y r es el radio. Para obtener el radio, puedes utilizar la función prompt(), y para mostrar el resultado, puedes utilizar la función alert().
3. Crear un programa que pida al usuario un número y determine si es par o impar. Para lograr esto, deberás utilizar el operador de módulo %, que devuelve el resto de la división de dos números. Si el resto de dividir el número ingresado por 2 es 0, entonces el número es par; de lo contrario, es impar. Puedes utilizar la función prompt() para obtener el número del usuario y la función alert() para mostrar el resultado.

CONSULTAS?

Muchas Gracias!