Integrantes:

Barroso Bollero Agustin - 52818
Caracchi Victoria - 53482
Ponce Lautaro - 52898
Reschini Enrico - 52973

Ejercitación 1

1- HTML (HyperText Markup Language o, en español Lenguaje de marcas de hipertexto) es un lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones. Este lenguaje permite a los desarrolladores definir la estructura básica de una página web y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros. Fue creado por Tim Berners-Lee en 1991, como parte del desarrollo de la World Wide Web. Desde su creación, HTML pasó por varias versiones de sí mismo:

1. HTML 1.0 (1991): La primera versión de HTML fue lanzada en 1993, era una versión muy básica, que incluía soporte para textos, hipervínculos y la estructura básica de documentos.

2. HTML 2.0: Esta versión, lanzada en 1995, estandarizó las características básicas del HTML y agregó algunas nuevas, como soporte para formularios y scripts.

3. HTML 3.2: Se lanzó en 1997 e Incluyó características adicionales como tablas frames y hojas de estilo en línea.

4. HTML 4.0: Fue lanzado en 1997 y agregó soporte para hojas de estilo externas, capas y marcos en línea.

5.HTML 4.01: Lanzado en 1999 introdujo un modelo de contenido más estructurado, soporte mejorado para scripts, y estilos en cascada (CSS), mejorando la versión anterior.

6. XHTML 1.0 (Extensible Hypertext Markup Language) : Se lanzó en el año 2000 Una reescritura de HTML 4.01 en XML, diseñada para ser más estricta y compatible con XML, dando mayor modularidad y estructura al lenguaje.

7. HTML5 : Es la quinta y más reciente versión significativa de HTML, la cual se lanzó en 2014. Introduce nuevas características como soporte para audio y video sin necesidad de plugins, nuevos elementos semánticos, y APIs para aplicaciones web. HTML5 también se centra en mejorar la interoperabilidad entre navegadores y dispositivos.

Como dijimos anteriormente, la última versión de HTML es HTML5. Este estándar se sigue manteniendo y evolucionando, con actualizaciones continuas para adaptarse a las

nuevas tecnologías web. Además esta versión ha sido diseñada para ser compatible con una amplia gama de dispositivos y permite a los desarrolladores crear aplicaciones web más ricas e interactivas.

2- El W3C recomienda seguir una serie de principios básicos para la creación de documentos HTML. Estos principios incluyen:

1. Compatibilidad con versiones anteriores: Las tecnologías utilizadas deben ser compatibles con versiones anteriores, asegurando que los documentos HTML funcionen en una amplia gama de navegadores y dispositivos, tanto antiguos como modernos.
2. Especificaciones detalladas: Las especificaciones deben ser lo suficientemente detalladas para permitir una implementación completa y consistente sin la necesidad de ingeniería inversa. Esto asegura que los documentos HTML se muestren de manera coherente en diferentes navegadores y plataformas.
3. Interoperabilidad: Se deben seguir criterios claros de conformidad para los navegadores y otros agentes de usuario, con el objetivo de mejorar la interoperabilidad, es decir, la capacidad de diferentes sistemas para intercambiar y utilizar información de manera eficaz.
4. Adaptabilidad: HTML debe poder adaptarse a diferentes tipos de dispositivos, incluyendo PCs, teléfonos móviles, dispositivos de mano, entre otros, para garantizar que la información sea accesible desde cualquier dispositivo.
5. Claridad y simplicidad: Las etiquetas HTML deben ser claras y descriptivas, facilitando la lectura y el mantenimiento del código por parte de los desarrolladores.

3- En las especificaciones de HTML, un elemento o atributo se considera desaprobado cuando se ha quedado anticuado por la presencia de nuevas estructuras. Los agentes de usuario deberían seguir dando soporte a este tipo de elementos por razones de compatibilidad. Estos elementos podrían considerarse obsoletos en un futuro. Un ejemplo podría estar relacionado con los atributos de presentación de HTML, los cuales han sido desaprobados cuando existen alternativas con hojas de estilo.

Por otro lado, un elemento o atributo obsoleto es aquél para el cual no hay garantía de soporte por parte de un agente de usuario. Los elementos obsoletos han dejado de estar definidos en la especificación, pero se enumeran por motivos históricos en la sección de cambios del manual.

4- DTD (Document Type Definition) define la estructura y las reglas de un documento HTML. Especifica qué elementos y atributos son permitidos. En HTML 4.01, los posibles DTDs son:

a. Transitional: Permite el uso de elementos y atributos obsoletos por compatibilidad con versiones anteriores.
b. Strict: Excluye elementos y atributos obsoletos, promoviendo un código más limpio y acorde con los estándares.
c. Frameset: Permite la definición de marcos (frames), usados comúnmente para dividir la ventana del navegador en secciones.

5- Los metadatos son elementos HTML que tienen como función describir la página web que los contiene, siendo estos opcionales. Estos son usados por los motores de búsqueda para hacer un análisis de dicho sitio web en cuanto a qué tipo de contenido se ofrece y cuál es la temática que allí se maneja. En HTML, se especifican dentro del elemento <head> usando la etiqueta <meta>.

Ejercitación 2

1. Este segmento del código se trata de un comentario en HTML, no es obligatorio y no tiene ningún efecto visual en nuestra página y puede ser colocado en cualquier parte de nuestra página.
2. Este segmento se coloca en el *body* de nuestro sitio, no produce ningún efecto visual y el elemento que tiene es “id” y su valor es “bloque1”, el cual no es obligatorio.
3. Este segmento de código se coloca en el “body” de nuestra página, lo que hace es poner la imagen que se encuentre en la dirección indicada en “src”. Dentro de las etiquetas que no son obligatorias se encuentran: “alt“, “id“, “name“, “width“, “height“ y “longdesc“, donde sus valores son respectivamente “lugar imagen“, “im1“, “im1“, “32“, “32“ y “detalles.html“ y la única etiqueta obligatoria que tiene es “src” que se encuentra sin valor alguno.
4. <meta name="keywords" lang="es" content="casa, compra, venta, alquiler " />
   <meta http-equiv="expires" content="16-Sep-2019 7:49 PM" />
   Este segmento se coloca en la cabecera (head) de nuestro html. Estas etiquetas se utilizan para incluir información (metadatos) de referencia sobre la página.
   En el primer caso mediante el atributo name definimos el nombre del metadato, en nuestro caso, “keywords” (palabras claves) estas palabras le dan a los motores de búsqueda más información acerca del contenido de la página, con el atributo lang definimos el lenguaje (language) y con content (atributo) definimos cada una de estas palabras. En el segundo caso el atributo http-equiv=”expires” nos indica que el documento html expirará en la fecha indicada en el atributo content.
   Es obligatorio tener un atributo name o charset o http-equiv y si se está trabajando con atributos name o http-equiv necesitamos su correspondiente atributo content para saber el contenido de los mismos.
5. Este segmento de código se trata de un elemento ‘<a>’, el cual define un hipervínculo. Por lo general, se colocan en la sección “body” del documento. Esto crea un enlace de texto visible en la página que dice “Resumen HTML” (contenido de la etiqueta obligatorio). De esta forma, cuando se hace click sobre él, te envía hacia la URL especificada en el atributo ‘href’. La etiqueta ‘<a>’ es usada para crear enlaces en HTML y es obligatoria para la definición de los mismos. Luego, el atributo ‘href’ establece la URL a la que redirige el enlace cuando se clickea y también es obligatoria. Tenemos además los atributos ‘type’, ‘hreflang’, ‘charset’ y ‘rel’, los cuáles no son obligatorios. Estos indican el tipo MIME del recurso enlazado (en este caso, text/html indica que el recurso es una página HTML.), el idioma del recurso de destino (‘es’ indica

que el documento enlazado está en español), especifican el conjunto de caracteres del recurso enlazado (utf-8 en este caso) y definen la relación del documento enlazado con el documento actual (en este caso el valor es ‘help’) respectivamente.

6. El siguiente segmento de código se coloca en el “body“ de la página, mediante la etiqueta table se produce una tabla con filas y columnas. Sus elementos son “tr” “th” y “td”, es esencial que tenga al menos uno de cada uno de estos elementos. “th” define el encabezado de la tabla, “tr” divide las filas y las columnas y “td” define las celdas de la tabla. Caption es otro elemento, que con su atributo align top (alinear arriba) nos permite agregar una descripción de la tabla arriba de la misma. El atributo scope en los elementos th (headers) permite aclarar si vamos a definir un header de una fila o de una columna, dependiendo si le damos el valor “row” o “col”. En este caso contamos con las etiquetas width y summary, “width”, nos permite definir el ancho de nuestra tabla, en este caso su valor es “200” el cual es tomado en píxeles, por otro lado, la etiqueta “summary” nos permite poner un resumen lo cual hace que nuestra pagina sea mas accesible.

Ejercitación 3

1.

Mediante la etiqueta <a>, la cual es común en todos los ejemplos, creamos hipervínculos.

```
<a href="http://www.google.com.ar">Click aquí para ir a Google</a>
```

En este primer caso creamos un hipervínculo sobre el texto “Click aquí para ir a Google” que al tocarlo nos llevará a Google en la misma pestaña

```
<a href="http://www.google.com.ar" target="_blank">Click aquí para ir a Google</a>
```

A diferencia del anterior, este elemento cuenta con la etiqueta *target=”_blank”* la cual hace que al presionar el texto nos dirija al Google en una nueva pestaña

```
<a href="http://www. google.com.ar" type="text/html" hreflang="es" charset="utf-8" rel="help">
```

En este caso, el elemento se encuentra incompleto ya que no tiene etiqueta de cierre y tampoco podemos acceder al hipervínculo. Por otro lado, cuenta con varios atributos que ayudan a la comprensión del elemento tales como type=”text/html” que nos dice que el contenido del enlace va a ser de tipo html, hreflang=”es” que nos indica que el idioma del contenido del enlace esta en idioma español, charset=”utf-8” que indica que la codificación del enlace es utf-8 y por último, el atributo rel=”help” que nos indica que el enlace ofrece información de ayuda para el usuario.

```
<a href="#">Click aquí para ir a Google</a>
```

En este caso, pese a que el hipervínculo diga que nos redireccionará a google nos está direccionando al principio de la página en la que nos encontramos,debido a que en lugar de colocar la url de google colocamos “**#**” en el atributo href.

`<a name="arriba" id="arriba"></a>` con este elemento definimos una sección la cual se llama arriba dentro de la misma página, por lo tanto luego con:

```
<a href="#arriba">Click aquí para volver arriba</>
```

Nos estamos dirigiendo a la sección de la página previamente creada.

2. Párrafos e imágenes.

```
<p><img src="im1.jpg" alt="imagen1" /><a href="http://www.google.com.ar">Click aquí</a></p>
```

En este caso, se muestra una imagen seguida de un enlace de texto. Los dos elementos están incluidos en el mismo párrafo ya que están encerrados por la misma etiqueta <p>, pero funcionan de manera independiente.En cuanto a su visualización, aparece la imagen que se encuentre en la dirección im1.jpg junto a un enlace de texto que dice "Click aquí". Solo el texto "Click aquí" es clickeable y lleva al usuario a `http://www.google.com.ar`. Hacer clic en la imagen no genera ninguna acción, ya que no está envuelta en la etiqueta de enlace.

```
<p><a href="http://www.google.com.ar"><img src="im1.jpg" alt="imagen1" /></a> Click aquí</p>
```

En este caso, la imagen está dentro de la etiqueta de enlace. Esto significa que la imagen misma es un elemento clickeable que nos llevará al destino especificado en el href del enlace `http://www.google.com.ar` mientras que el texto "Click aquí" está fuera del enlace, por lo que no es clickeable.

```
<p><a href="http://www.google.com.ar"><img src="im1.jpg" alt="imagen1" />Click aquí</a></p>
```

Aquí, tanto la imagen como el texto "Click aquí" están dentro de la etiqueta de enlace <a>. Esto significa que ambos, la imagen y el texto, forman un bloque clickeable. Si hacemos clic en la imagen o en el texto, nos redirigirá al enlace especificado en el atributo href `http://www.google.com.ar`.

```
<p><a href="http://www.google.com.ar"><img src="im1.jpg" alt="imagen1" /></a> <a href="http://www.google.com.ar">Click aquí</a></p>
```

En este último caso, tenemos dos enlaces distintos, uno para la imagen y otro para el texto. Ambos enlaces son funcionalmente iguales ya que nos llevan al mismo lugar a pesar de estar en dos etiquetas separadas. Desde una perspectiva de usabilidad, los usuarios pueden no ver inmediatamente que la imagen y el texto están relacionados, lo que puede generar una pequeña confusión.

3.

```
<ul>
  <li>xxx</li>
  <li>yyy</li>
  <li>zzz</li>
</ul>
```

De esta manera creamos una lista no ordenada, es decir, sus elementos se enumeran con puntos.

```
<ol>
      <li>xxx</li>
      <li>yyy</li>
      <li>zzz</li>
</ol>
```

También se crea una lista, pero en este caso es ordenada, por lo tanto enumera utilizando números.

```
<ol>
<li>xxx</li>
</ol>
<ol>
<li value="2">yyy</li>
</ol>
<ol>
<li
value="3">zzz</li>
</ol>
```

En este caso creamos 3 listas ordenadas, en las cuales, mediante el atributo value utilizado en cada elemento li nos permite especificar el número de enumeración, de no tener este atributo, todos los elementos tendrán el número 1 debido a que son el primer elemento de la lista a la que pertenecen.

```
<blockquote>
<p>1. xxx<br />
2. yyy<br />
3. zzz</p>
</blockquote>
```

Al utilizar la etiqueta blockquote hacemos referencia a que el contenido de la misma es una cita de otra fuente, permitiéndonos así utilizar el mismo formato cada vez que necesitemos realizar una cita. Luego tenemos un párrafo en el cual se arma una lista utilizando la etiqueta br, que agrega un salto de línea.

4. En este caso ambos códigos utilizan la etiqueta <table></table>, la cual se utiliza para representar datos en dos o más dimensiones. Se crean tablas similares de tres filas y

dos columnas, pero presentan diferencias relacionadas a la manera en que se alinea y formatea el contenido.

```
<table border="1" width="300">
<tr>
<th>Columna 1</th>
<th>Columna 2</th>
</tr>
<tr>
<td>Celda 1</td>
<td>Celda 2</td>
</tr>
<tr>
<td>Celda 3</td>
<td>Celda 4</td>
</tr>
</table>
```

Entre las principales diferencias entre los códigos se encuentra el centrado y la negrita en los encabezados. En este primer código, los encabezados están centrados y en negrita automáticamente por el uso de la etiqueta <th>. Es decir, vemos como el texto de los encabezados “Columna 1” y “Columna 2” están escritos en una sola línea utilizando la etiqueta <th>, alineándose automáticamente al centro de la celda. En este código se usa <th> para los encabezados, lo cual es semánticamente correcto para identificar celdas de encabezado.

```
<table border="1" width="300">
<tr>
<td><div align="center"><strong>Colum
na1</strong></div></td>
<td><div align="center"><strong>Columna
2</strong></div></td>
</tr>
<tr>
<td>Celda 1</td>
<td>Celda 2</td>
</tr>
<tr>
<td>Celda 3</td>
```

```
<td>Celda 4</td>
</tr>
</table>
```

En este segundo código, se utiliza para centrar y poner en negrita los textos. La combinación de estas etiquetas puede llevar a un comportamiento inesperado, como por ejemplo el salto de línea dentro del texto, lo que afecta la legibilidad y el diseño. De esta forma, observamos que en el segundo código se realizan 2 saltos de línea al momento de escribir el texto de los encabezados. Esto visualmente se observa como un espacio, por lo que en este caso en la tabla uno de los encabezados aparece como “Colum na1” y el otro “Columna 2” pero por el hecho de que antes de escribir el “2” se hace un salto de línea.

5.

```
<table width="200">
<caption>
Título
</caption>
<tr>
<td
bgcolor="#dddddd"> </td>
<td
bgcolor="#dddddd"> </td>
<td
bgcolor="#dddddd"> </td>
</tr>
<tr>
<td
bgcolor="#dddddd"> </td>
<td bgcolor="#dddddd">
 </td>
<td
bgcolor="#dddddd"> </td>
</tr>
</table>
```

```
<table width="200">
<tr>
<td colspan="3"><div
align="center">Título</div></td
>
</tr>
<tr>
<td
bgcolor="#dddddd"> </td>
<td
bgcolor="#dddddd"> </td>
<td
bgcolor="#dddddd"> </td>
</tr>
<tr>
<td
bgcolor="#dddddd"> </td>
<td
bgcolor="#dddddd"> </td>
<td
bgcolor="#dddddd"> </td>
</tr>
</table>
```

Visualmente, las 2 tablas son iguales, la diferencia está en la manera en la que se agrega el título en el código, en el primer caso, agrega el título con la etiqueta caption, la cual va siempre

inmediatamente después de la definición de la tabla y por defecto se alinea en el arriba de la misma, alineada en el centro. En cambio en el segundo ejemplo crea una fila (con tr) en la cual con el atributo colspan=”3” define que dentro de la fila habrá 3 columnas y luego alinea el título, el cual se encuentra dentro de un div, en el medio de la fila.

6.

```
<table width="200">
<tr>
<td colspan="3"><div
align="center">Título</div></td
>
</tr>
<tr>
<td rowspan="2"
bgcolor="#dddddd"> </td>
<td
bgcolor="#dddddd"> </td>
<td
bgcolor="#dddddd"> </td>
</tr>
<tr>
<td
bgcolor="#dddddd"> </td>
<td
bgcolor="#dddddd"> </td>
</tr>
</table>
```

```
<table width="200">
<tr>
<td colspan="3"><div
align="center">Título</div></td
>
</tr>
<tr>
<td colspan="2"
bgcolor="#dddddd"> </td>
<td
bgcolor="#dddddd"> </td>
</tr>
<tr>
<td
bgcolor="#dddddd"> </td>
<td
bgcolor="#dddddd"> </td>
<td
bgcolor="#dddddd"> </td>
</tr>
</table>
```

Para entender la diferencia entre estas dos tablas vamos a pensarlas como dos tablas iguales donde cada una tiene 3 columnas y 2 filas, enumerando a cada una de las celdas en sentido izquierda, derecha y arriba, abajo. Con esto, podrías decir que la diferencia entre ambas tablas es que la primera celda de la primera tabla está compuesta por las celdas 1 y 4 mientras que la primera celda de la segunda tabla está compuesta por las celdas 1 y 2.

7.

```
<table width="200"
border="1">
<tr>
<td colspan="3"><div
align="center">Título</div>
</td>
</tr>
<tr>
<td
```

```
<table width="200" border="1"
cellpadding="0"
cellspacing="0">
<tr>
<td colspan="2"><div
align="center">Título</div></
td>
</tr>
<tr>
```

colspan="2"rowspan="2"> </td> <td> </td> </tr> <tr> <td width="50%"> </td> </tr> </table>	<td rowspan="2"> </td> <td> </td> </tr> <tr> <td width="50%"> </td> </tr> </table>

La principal diferencia entre las dos tablas radica en la estructura de las celdas y el espaciado. La Tabla 1 utiliza `colspan="3"` para una celda de título que abarca tres columnas y combina `colspan="2"` con `rowspan="2"` en una celda grande que abarca dos columnas y dos filas, lo que resulta en una apariencia más espaciosa y equilibrada; en contraste, la Tabla 2 emplea `colspan="2"` para el título, y solo `rowspan="2"` en una celda, creando una estructura más compacta sin espacio adicional entre celdas debido a `cellpadding="0"` y `cellspacing="0"`, lo que hace que el contenido esté más ajustado al borde de las celdas.

8.

```
<form id="form1" name="form1" action="procesar.php" method="post"
target="_blank">
<fieldset>
<legend>LOGIN</legend>
Usuario: <input type="text" id="usu1" name="usu1" value="xxx"
/><br />
Clave: <input type="password" id="clave1" name="clave1"
value="xxx" />
</fieldset>
<input type="submit" id="boton1" name="boton1" value="Enviar" />
</form>
```

```
<form id="form2" name="form2" action="" method="get"
target="_blank">
LOGIN<br />
<label>Usuario: <input type="text" id="usu2" name="usu2"
/></label><br />
<label>Clave: <input type="text" id="clave2" name="clave2"
/></label><br />
<input type="submit" id="boton2" name="boton2" value="Enviar" />
</form>
```

```
<form id="form3" name="form3" action="mailto:xx@xx.com”
enctype=text/plain method="p
```

```
ost" target="_blank">
<fieldset>
<legend>LOGIN</legend>
Usuario: <input type="text" id="usu3" name="usu3" /><br />
Clave: <input type="password" id="clave3" name="clave3" />
</fieldset>
<input type="reset" id="boton3" name="boton3" value=
"Enviar" />
</form>
```

Una de las principales diferencias entre estos 3 formularios se encuentra en el destino del formulario. En el formulario de id = “form1”, los datos se envían a “procesar.php” usando el método POST, lo que significa que dichos datos se envían de forma segura y no son visibles en la URL. Por otro lado, el “form2” no especifica un destino, por lo que enviaría los datos a la misma página. Además, usa el método GET, lo que hace que los datos SI sean visibles en la URL. Por último, el formulario de id = “form3” envía los datos a una dirección de correo electrónico especificada en el “action” utilizando el método POST. Incluye además `enctype="text/plain"` para enviar los datos como texto sin formato. Al hacer click en el botón, se abre el cliente de correo predeterminado del usuario, mientras que los resultados de los 2 primeros se abren en una nueva pestaña.

Otra de las diferencias se da en los campos de entrada. En el primer formulario los campos de usuario y contraseña tienen valores predeterminados (value="xxx") y el campo de contraseña está protegido (input type="password"). Por otra parte, el segundo formulario no posee valores predeterminados en sus campos y su campo contraseña no está protegido (es un input type="text"). Luego el formulario 3 es similar al primero en relación a la configuración de sus campos, pero éste no presenta valores predeterminados.

Por último podemos mencionar el tipo de botón. Los formularios 1 y 2 utilizan un botón de tipo submit para enviar el formulario, mientras que el formulario 3 usa un botón de tipo reset etiquetado incorrectamente como "Enviar", lo que en realidad restablece el formulario en lugar de enviarlo.

9.

```
<label>Botón 1
<button type="button" name="boton1" id="boton1">
<img src="logo.jpg" alt="Botón con imagen " width="30"
height="20" /><br />
<b>CLICK AQUÍ</b></button></label>
```

```
<label>Botón 2
<input type="button" name="boton2" id="boton2" value="CLICK
AQUÍ" />
</label>
```

La principal diferencia entre ambos botones es que el primero utiliza la etiqueta `button`, la cual nos permite personalizar más el mismo y hace que nuestra pagina sea mas accesible a los usuarios mientras que el segundo utiliza la etiqueta input con el atributo `type="button"` que solo nos permite ingresar texto plano a través del atributo `value="<texto>"`.

10.

```
<p><label><input type="radio" name="opcion" id="X" value="X"
/>X</label><br />
<label><input type="radio" name="opcion" id="Y" value="Y"
/>Y</label></p>
```

```
<p><label><input type="radio" name="opcion1" id="X" value="X"
/>X</label><br />
<label><input type="radio" name="opcion2" id="Y" value="Y"
/>Y</label></p>
```

En ambos casos obtenemos un botón de selección, la diferencia está en que en el primer ejemplo, debido a que ambos tienen el mismo nombre (atributo name) solo se puede elegir una opción mientras que en el segundo caso como el nombre es distinto se pueden seleccionar ambas opciones.

11.

```
<select name="lista">
<optgroup label="Caso 1">
<option>Mayo</option>
<option>Junio</option>
</optgroup>
<optgroup label="Caso 2">
<option>Mayo</option>
<option>Junio</option>
</optgroup>
</select>
```

```
<select name="lista[]"
multiple="multiple">
<optgroup label=" Caso 1">
<option>Mayo</option>
<option>Junio</option>
</optgroup>
<optgroup label=" Caso 2">
<option>Mayo</option>
<option>Junio</option>
</optgroup>
</select>
```

En este caso, hay una diferencia funcional y otra visual. En el primer caso, solo nos permite seleccionar una opción a la vez mientras que en el segundo, con la ayuda de botones del teclado como `shift` y `ctrl` y el mouse, podemos seleccionar varias opciones a la vez. Por otro lado, existe una diferencia visual ya que en el primer caso tenemos que desplegar las opciones para verlas mientras que en el segundo no es necesario.