Pregunta 1. Sobre la utilización de tablas responded a las siguientes preguntas:

1.1 ¿Qué atributos utilizaremos para fusionar celdas verticalmente y horizontalmente? Poned un ejemplo de código que muestre su uso.

1.2 Explicad el uso de cada uno de los elementos siguientes: `thead`, `tbody` y `tfoot`.

1.3 ¿Para qué resulta útil el uso del atributo `scope`? ¿Qué valores puede tener este atributo? ¿En qué elemento de tabla podemos utilizarlo?

Atributos para fusionar celdas vertical y horizontalmente

Para fusionar celdas tenemos que utilizar los atributos **`colspan`** y **`rowspan`**, que permiten la fusión de celdas horizontal y verticalmente, respectivamente.

Por ejemplo, en esta tabla hay contenidos que ocupan más de una celda en horizontal (en color rojo) y otras que ocupan más de una celda en vertical (en color azul)

Turistas extranjeros. Gasto declarado. España, septiembre del 2018

Gasto	Valor	Variación interanual		
		absoluta	% mes	% acumulado
Total (M€)	9.543	66,7	0,7	2,5
Media por persona (€)	1.069	2,0	0,2	2,5
Media diaria por persona (€)	147	4,0	3,0	6,0
Fuente: INE. Encuesta de gasto turístico. Los datos son provisionales desde enero del 2018.				

El código HTML que le correspondería es el siguiente:

```
<table>
      <caption> Turistas extranjeros. Gasto declarado. España, septiembre del
2018</caption>
      <thead>
            <tr>
                  <th rowspan="2" scope="col">Gasto</th>
                  <th rowspan="2" scope="col">Valor</th>
                  <th colspan="3" scope="colgroup">Variación interanual</th>
            </tr>
            <tr>
                  <th scope="col">absoluta</th>
                  <th scope="col">% mes</th>
                  <th scope="col">% acumulado</th>
            </tr>
      </thead>
      <tbody>
            <tr>
                  <th scope="row">Total (M€)</th>
                  <td>9.543</td>
                  <td>66,7</td>
                  <td>0,7</td>
                  <td>2,5</td>
            </tr
            <tr>
                  <th scope="row">Media por persona (€)</th>
                  <td>1.069</td>
```

```
                    <td>2,0</td>
                    <td>0,2</td>
                    <td>2,5</td>
              </tr>
              <tr>
                    <th scope="row">Media diaria por persona (€)</th>
                    <td>147</td>
                    <td>4,0</td>
                    <td>3,0</td>
                    <td>6,0</td>
              </tr>
        </tbody>
        <tfoot>
              <tr>
                    <td colspan="5"> Fuente: INE. Encuesta de gasto turístico. Los datos son
provisionales desde enero del 2018</td>
              </tr>
        </tfoot>
</table>
```

Uso de `thead` , `tbody` y `tfoot`

Los elementos `thead`, `tbody` y `tfoot` nos permiten diseñar tablas estructuradas.

- `thead` constituye el encabezamiento de la tabla. Se inserta después `caption` de la tabla.
- `tbody` constituye el cuerpo de la tabla.
- `tfoot` constituye el pie de la tabla.

En la tabla del apartado anterior hemos estructurado la tabla mediante estos tres elementos (en color verde).

Utilidad del atributo `scope`, sus valores y elemento de tabla donde lo utilizamos

Para los usuarios videntes, es fácil establecer visualmente cuáles son las asociaciones que hay entre los datos que ocupan las columnas y las filas de una tabla:

Así, en esta tabla:

Tipología	**Valor**	**% sobre total**
Turistas	2.005,3	73,4
Excursionistas	727,5	26,6

Leemos que:

- "Tipología" es la celda que encabeza los datos "Turistas" y "Excursionistas"
- "Valor" es la celda que encabeza los datos "2.005,3" y "727,5"
- ...
- "Turistas" es la celda que encabeza los datos "2.005,3" y "73.4"
- ...

Las relaciones entre datos y sus encabezamientos son muy evidentes.

Para hacer que estas relaciones sean también evidentes a los lectores de pantalla que tienen que leer la tabla a sus usuarios, es necesario utilizar las celdas de encabezamiento, `<th>`, y, si los encabezamientos lo requieren, añadiremos el atributo `scope` que puede tener cuatro valores posibles: `row`, `col`, `rowgroup`, `colgroup`.

Ejemplo: Dada esta tabla,

Turistas extranjeros. Gasto declarado. España, septiembre del 2018

Gasto	**Valor**	**Variación interanual**		
		absoluta	**% mes**	**% acumulado**
Total (M€)	9.543	66,7	0,7	2,5
Media por persona (€)	1.069	2,0	0,2	2,5
Media diaria por persona (€)	147	4,0	3,0	6,0
Fuente: INE. Encuesta de gasto turístico. Los datos son provisionales desde enero del 2018.				

El código HTML que le correspondería es el siguiente:

```
<table>
     <caption>Turistas extranjeros. Gasto declarado. España, septiembre del
2018</caption>
     <thead>
          <tr>
               <th rowspan="2" scope="col">Gasto</th>
               <th rowspan="2" scope="col">Valor</th>
               <th colspan="3" scope="colgroup">Variación interanual</th>
          </tr>
          <tr>
               <th scope="col">absoluta</th>
               <th scope="col">% mes</th>
               <th scope="col">% acumulado</th>
          </tr>
     </thead>
     <tbody>
          <tr>
               <th scope="row">Total (M€)</th>
               <td>9.543</td>
               <td>66,7</td>
               <td>0,7</td>
               <td>2,5</td>
          </tr
          <tr>
               <th scope="row">Media por persona (€)</th>
               <td>1.069</td>
               <td>2,0</td>
               <td>0,2</td>
               <td>2,5</td>
          </tr>
          <tr>
               <th scope="row">Media diaria por persona (€)</th>
               <td>147</td>
               <td>4,0</td>
               <td>3,0</td>
               <td>6,0</td>
          </tr>
     </tbody>
     <tfoot>
          <tr>
               <td colspan="5">Fuente: INE. Encuesta de gasto turístico. Los datos son
provisionales desde enero del 2018.</td>
          </tr>
     </tfoot>
</table>
```

Pregunta 2. Sobre la utilización de formularios responded a las siguientes preguntas:

2.1 Explicad qué características tienen los siguientes tipos de `input` y poned ejemplos que indiquen en qué casos resulta útil su utilización:

- `radio`
- `checkbox`
- `password`
- `number`

2.2 Explicad para qué sirve y cómo debe utilizarse la etiqueta `label` dentro de un formulario.

Características del tipo de `input`: `radio, checkbox, password, number`

`radio`

- Los controles de formulario de tipo **`radio`** se utilizan cuando el usuario únicamente puede seleccionar una opción entre unas cuantas y que, además, son excluyentes.
- El atributo `name` permite definir el grupo de botones de radio de entre los cuales tendrán que escoger uno solo. El valor de este atributo tiene que ser el mismo para todos los botones del mismo grupo.
- El atributo `value` determina el valor del botón de tipo **`radio`** seleccionado que se enviará al script para su procesamiento.

Ejemplo, en un formulario de reserva de alojamiento queremos que los usuarios puedan escoger una (solo una) opción de pago:

```
<fieldset>
      <legend>Tipo de pago para el depósito de reserva</legend>
            <ul>
              <li>
                <input id="pagoPorVisa" name="tipoPago" type="radio" value="Visa">
                <label for=" pagoPorVisa ">Visa</label>
              </li>
              <li>
                <input id="pagoPorMastercard" name="tipoPago" type="radio"
value="Mastercard">
                <label for="pagoPorMastercard">Mastercard</label>
              </li>
              <li>
                <input id="pagoPorPaypal" name="tipoPago" type="radio" value="Paypal">
                <label for="pagoPorPaypal">Paypal</label>
              </li>
            </ul>
</fieldset>
```

checkbox

- Los controles de formulario de tipo **checkbox** se utilizan cuando el usuario tiene diversas opciones a escoger y, además, puede escoger ninguna, una, muchas o todas.
- El atributo `name` no ha de tener el mismo valor en todo el grupo ya que interesa que el usuario pueda hacer una selección múltiple.
- El atributo `value` determina el valor del botón de tipo **checkbox** seleccionado que se enviará al script para su procesamiento.

Ejemplo, en un formulario de reserva de alojamiento queremos que los usuarios puedan escoger algunas opciones extra (más de una):

```
<fieldset>
      <legend>Opciones extra</legend>
            <ul>
              <li>
                <input id="cbDesayuno" type="checkbox" value="Desayuno">
                <label for="cbDesayuno">Desayuno</label>
              </li>
              <li>
                <input id="cbAlmuerzo" type="checkbox" value="Almuerzo">
                <label for="cbAlmuerzo">Almuerzo</label>
              </li>
              <li>
                <input id="cbCena" type="checkbox" value="Cena">
                <label for="cbCena">Cena</label>
              </li>
              <li>
                <input id="cbDiarios" type="checkbox" value="Diarios">
                <label for="cbDiarios">Diarios</label>
              </li>
              <li>
                <input id="cbExtraParking" type="checkbox" value="Plaza de aparcamiento
extra">
                <label for="cbExtraParking">Plaza de aparacamiento extra</label>
              </li>
            </ul>
</fieldset>
```

password

Cuando un usuario tiene que hacer una conexión a un espacio reservado, resulta aconsejable que introduzca su contraseña sin que se muestren los caracteres que escribe. En este caso, en el elemento `input`, el atributo `type` toma el valor `password`.

Ejemplo, en un formulario de acceso a la zona de usuarios del sitio es necesario introducir el correo electrónico y la contraseña:

```
<fieldset>
      <legend>Datos de usuario</legend>
            <ul>
                <li>
                      <label for="correo">Correo electrónico</label>
                      <input type="email" id="correo" name="email" value="" required>
                </li>
                <li>
                      <label for="contrasenya">Contraseña</label>
                      <input type="password" id="contrasenya" name="password" value=""
required>
                 </li>
```

```
            </ul>
</fieldset>
```

number

- Cuando un usuario tiene que hacer una entrada de datos que únicamente admitirá datos numéricos, en el elemento `input`, el atributo `type` toma el valor `number`.
- El interés de usar un campo de tipo numérico es la posibilidad que se da de utilizar un botón que aparece a la derecha del campo y que permite incrementar o reducir el valor.
- En los campos de tipo numérico podemos utilizar los atributos `min`, para especificar el valor mínimo admitido, `max` para especificar el valor máximo admitido y `step` para indicar el valor de incremento con el que actuará el botón a la derecha del campo que hemos descrito en el punto anterior.

Ejemplo, en un formulario de reserva de alojamiento queremos que los usuarios puedan indicar el número de personas para quienes se realiza la reserva:

```
<fieldset>
      <legend>Datos de la reserva</legend>
            <ul>
              <li>
                <label for="numeroAdultos">Número de adultos</label>
                <input id="numeroAdultos" type="number" value="" required>
              </li>
              <li>
                <label for="numeroNinos">Número de niños</label>
                <input id="numeroNinos" type="number" value="" required/>
              </li>
            </ul>
</fieldset>
```

Uso de la etiqueta `label`

La utilización del elemento `label` es importante para identificar los controles de un formulario. Sus funciones son las siguientes:

- Describen cuál es el propósito del control de formulario.
- Cuando clicamos en la etiqueta, los navegadores permiten activar el control que identifican.
- Los lectores de pantalla pueden referirse de manera correcta al control del formulario que la etiqueta identifica.

La asociación entre la etiqueta y su control puede hacerse de manera **explícita** o **implícita**.

Asociación explícita

La etiqueta y el control se asocian mediante el valor del atributo `for` de `label` que ha de coincidir con el valor del identificador (`id`) del control:

```
<label for="correo">Dirección de correo electrónico</label>
<input type="correo" name="correo" id="correo" value="">
```

Asociación implícita

En este tipo de asociación `label` contendrá el control:

```
<label>
     <input type="text" name="codigo" value="">
</label>
```

Desde el punto de vista de la accesibilidad y la usabilidad del formulario, es preferible la asociación explícita.

Recomendamos la lectura de este artículo sobre el uso de `label`:

https://www.w3.org/WAI/tutorials/forms/labels/

Pregunta 3. Sobre el modelo de caja (CSS box model) responded a las siguientes preguntas:

3.1 ¿Qué elementos componen la estructura de una caja?

3.2 ¿Qué diferencia hay entre `display: block`, `display: inline` y `display: inline-block`?

3.3 ¿Para qué se utiliza la propiedad CSS`box-sizing`? ¿Qué valores puede tener esta propiedad?

Elementos que componen la estructura de una caja

Los elementos de un documento HTML son representados por el navegador como cajas rectangulares de acuerdo con lo que se conoce como modelo de caja (*box model*). El CSS se encarga de determinar las diversas propiedades de estas cajas.

Anchura y altura

`width` y **`height`**: establecen la anchura y la altura del cuadro de contenido, que es el área en la que se muestra el contenido de la caja (textos, imágenes...).

Otras propiedades relacionadas con la anchura y la altura son **`min-width`**, **`max-width`**, **`min-height`** y **`max-height`**, que permiten establecer las anchuras y alturas máximas y mínimas de una caja.

Área de relleno

El área de relleno es el margen interno de una caja, la que encontramos entre el contenido y los bordes de la caja.

`padding-left`, **`padding-right`**, **`padding-top`** y **`padding-bottom`**: establecen este margen interno en cada uno de sus lados (izquierdo, derecho, superior e inferior, respectivamente). La propiedad **`padding`** es la propiedad abreviada que permite establecer el margen de los cuatro lados de manera simultánea.

Bordes

El borde de una caja se encuentra entre el borde exterior del relleno y el borde interno del margen. De manera predeterminada, el borde tiene una medida de 0 píxeles, lo que lo hace invisible; las propiedades que nos permiten modificarlo son:

- **`border`** permite establecer el grosor, el estilo y el color de los cuatro lados de la caja
- **`border-top`**, **`border-bottom`**, **`border-left`** y **`border-right`** permiten establecer las mismas características, pero únicamente en uno de sus lados (superior, inferior, izquierdo y derecho, respectivamente).
- **`border-width`** permite indicar el grosor de los bordes de los cuatro lados.

- **`border-style`** permite indicar el estilo de los bordes de los cuatro lados.
- **`border-color`** permite indicar el color de los bordes de los cuatro lados.
- **`border-top-width`**, **`border-top-style`**, **`border-top-color`**, **`border-bottom-width`**... permiten indicar las características concretas de uno de los lados de la caja.

Área de márgenes

El área de márgenes está delimitada por los bordes, rodea una caja CSS y la separa de sus vecinas.

`margin-left`, **`margin-right`**, **`margin-top`** y **`margin-bottom`**: establecen este margen en cada uno de sus lados (izquierdo, derecho, superior e inferior, respectivamente). La propiedad **`margin`** es la propiedad abreviada que permite establecer el margen de los cuatro lados de manera simultánea.

Valores de la propiedad `display: block`, `inline` y `inline-block`

Con la propiedad `display` podemos indicar si un elemento tiene que ser tratado como un elemento de bloque o un elemento de línea.

Los elementos de bloque ocupan todo el espacio disponible en su contenedor; los navegantes muestran un salto de línea antes y después del elemento. Son elementos de bloque: `<p>`, `<div>`, `<blockquote>`, `<ul>`...

Los elementos de línea ocupan solo el espacio necesario para mostrar su contenido. Son elementos de línea: `<cite>`, `<span>`, `<img>`...

La propiedad `display` de CSS permite cambiar este comportamiento por defecto de los elementos, de tal manera que un elemento de línea puede comportarse como un elemento de bloque y viceversa.

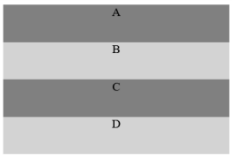
- El valor `block`: muestra el elemento como un elemento de bloque y adquiere la anchura (`width`) y altura (`height`) especificadas.
- El valor `inline`: muestra el elemento como un elemento de línea. Las propiedades `width` y `height` no tienen efecto.
- El valor `inline-block`: muestra el elemento como si se tratara de un elemento de línea, pero podemos aplicar valores de altura y anchura.

En el siguiente fragmento de código, cambiaremos el comportamiento del elemento `<span>` que, siendo un elemento de línea, se comportará como un elemento de bloque:

HTML
```
<span>A</span> <span>B</span> <span>C</span> <span>D</span>
```

CSS
```
span { height: 50px; width: 20%; margin-left: auto; margin-right: auto;
text-align: center; background-color: lightgrey; display: block;
}
span:nth-child(odd) { background-color: grey;}
```

La representación visual será la siguiente:



En el siguiente fragmento de código, cambiaremos el comportamiento del elemento `<div>` que, siendo un elemento de bloque es comportará como un elemento de línea.

HTML
```
<div>A</div> <div>B</div> <div>C</div> <div>D</div>
```

CSS
```
div { height: 50px; width: 20%; margin-left: auto; margin-right: auto;   text-
align: center; background-color: lightgrey; display: inline;
}
div:nth-child(odd) { background-color: grey;}
```

La representación visual será la siguiente:

En el siguiente fragmento de código, cambiaremos el comportamiento del elemento `<div>` que, siendo un elemento de bloque, se comportará como un elemento de línea, pero admitirá las propiedades de altura y anchura.

HTML
```
<div>A</div> <div>B</div> <div>C</div> <div>D</dv>
```

CSS
```
div { height: 50px; width: 20%; margin-left: auto; margin-right: auto;   text-
align: center; background-color: lightgrey; display: inline-block;
}
div:nth-child(odd) { background-color: grey;}
```

La representación visual será la siguiente:

Propiedad box-sizing

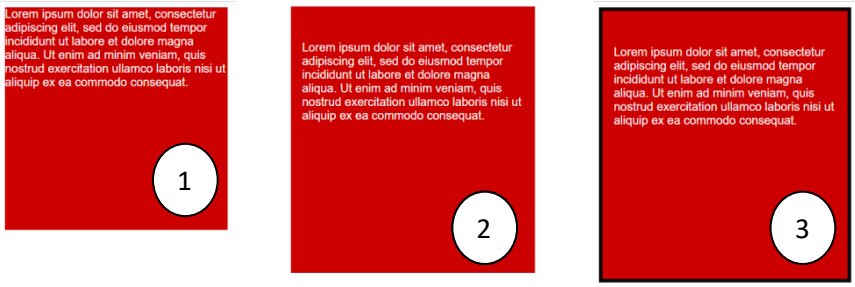
En el modelo tradicional de caja cuando indicamos que un elemento tiene una altura y una anchura y además un relleno y unos bordes, la anchura, o la altura,

real calculada por el navegador es la suma de todas esas propiedades. Así, por ejemplo, si tenemos un elemento con una altura y una anchura de 300 píxeles (1) que además tiene relleno y bordes la anchura y la altura calculadas resultarán ser la suma de todos esos valores:

```
.box{
      background-color: #cc0000;
      color: #fff;
      width: 300px;
      height: 300px;
      padding: 30px 15px;
      border: 5px solid #080808;
      margin-bottom: 30px;
}
```

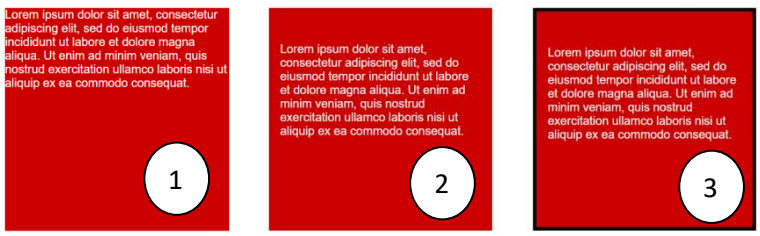
Al indicar que la caja tiene que tener un relleno de 30 píxeles por la parte superior e inferior y de 15 píxeles por la izquierda y la derecha, el espacio "real" ocupado por la caja se ha incrementado. Así la caja ha pasado a ocupar 300 + 15 + 15 = 330 píxeles de anchura y 300 + 30 +30 = 360 píxeles de altura (2).

Como además la caja tiene un borde de 5 píxeles por los cuatro lados, el espacio "real" ocupado por la caja se ha incrementado. Así la caja ha pasado a ocupar 330 + 5 + 5 = 340 píxeles de anchura y 360 + 5 + 5 = 370 píxeles de altura (3).



Cuando a la propiedad **`box-sizing`** le damos el valor **`content-box`**, estamos indicando que interesa que las cajas tienen que comportarse con el modelo tradicional de caja que hemos descrito en los párrafos anteriores.

La propiedad **`box-sizing`** con el valor **`border-box`** permite cambiar el modelo tradicional de caja descrito en el apartado anterior ya que la anchura y altura calculadas del elemento no se verán incrementadas por el relleno o por los bordes. Así, si al código CSS que hemos utilizado para representar las cajas anteriores, añadimos la propiedad **`box-sizing`** con valor **`border-box`**, vemos como la anchura y altura de las cajas se mantiene constante (300 píxeles).



```
.box{
      box-sizing: border-box;
      background-color: #cc0000;
      color: #fff;
      width: 300px;
      height: 300px;
      padding: 30px 15px;
      border: 5px solid #080808;
}
```