¿Qué es un Problema?

- Un **problema** es un conjunto de cuestiones o situaciones que se plantean para ser resueltas.

  En *Informática*: se busca la solución utilizando computadoras, mediante un programa (buscando el mejor resultado en tiempo y forma).

¿Qué es un programador?

- **Programador** es aquella persona que crea o desarrolla programas para solucionar problemas.

¿Qué es un Algoritmo?

- El conjunto de instrucciones que especifican la secuencia de operaciones a realizar para resolver un sistema específico o clase de problema se denomina ***algoritmo***.
  En otras palabras, *un algoritmo* es una fórmula para la resolución de un problema.

Características de un Algoritmo

1. Debe ser **correcto:** responder a lo que me piden y resolver el problema
2. **Eficiente** en cuanto a recursos y tiempo
3. **Claro**
4. **Flexible:** poder adaptarse a pequeños cambios de lógica
5. **Preciso** e indicar el orden de realización de cada paso
6. **Estar definido:** si se sigue un algoritmo dos veces, se debe obtener el mismo resultado cada vez
7. **Ser finito:** se debe terminar el algún momento; debe tener un número finito de pasos
8. **Fiable ó confiable:** estar acorde a lo anterior en cuanto a propuesta de solución

La definición de un algoritmo debe describir tres partes:
entrada, proceso y salida.

¿Qué es un Programa?

- Un ***programa*** se escribe en un lenguaje de programación, y a la actividad de expresar un algoritmo en forma de programa se le denomina programación.
  Un programa consta de una secuencia de instrucciones, cada una de las cuales especifica las operaciones que debe realizar la computadora.

Pseudocódigo

- **Pseudocódigo** (o falso lenguaje) es comúnmente utilizado por los programadores para omitir secciones de código o para dar una explicación del paradigma que tomó el mismo programador para hacer su código, esto quiere decir, que el Pseudocódigo no es programable sino que facilita la programación.

Variables

- Cuando representamos datos (numéricos o alfanuméricos) debemos darles un nombre. Una **variable** es un nombre que representa el valor de un dato.
- En esencia, una **variable** es una zona o posición de memoria en la computadora donde se almacena información.
- En un pseudocódigo y también en un programa se pueden crear tantas variables como querramos.

Variables

- *A* ← *50*
  Variable tipo numérica *A* cuyo valor es 50

- *Ciudad* ← *"Asunción“*
  Variable alfanumérica o de tipo carácter *Ciudad*, cuyo valor es "Asunción“

- *X* ← *C + B*
  Variable numérica *X* cuyo valor es la suma de los valores de las variables numéricas *C* y *B*. Esta es una variable calculada

Variables

- Ten en cuenta que las operaciones que se pueden realizar con dos o más variables exigen que éstas sean del mismo tipo. No podemos "sumar", por ejemplo una variable alfanumérica a otra numérica y viceversa como por ejemplo:

  FechaNueva ← "1 de Junio de 1.971" + 5

  ** Esto NO se puede hacer **

Ejemplo

- Verificar si un número es mayor a 10

Inicio Programa

Leer numero

Si numero mayor a 10 *entonces*

Mostrar “El número es mayor a 10”

Sino

Mostrar “El número no es mayor a 10”

Fin Si

Fin Programa

Estructura "*Si*"

Si *condición* **entonces**

...

(esta parte se ejecuta SOLO si la condición se cumple)

...

Sino

...

(esta parte se ejecuta SOLO si la condición NO se cumple)

...

Fin Si

Ejemplo estructura "***Si***"

Inicio Programa

Leer *edad*

Si *edad > 18* **entonces**

Mostrar *"Mayor de edad"*

Sino

Mostrar *"Menor de edad"*

Fin Si

Fin Programa

Estructura "***Para***" o "***Desde***"

Para *variable = Vi* **hasta** *Vf [incremento]*

...

instrucción o instrucciones

...

Fin Para

Donde:

- variable: variable índice
- *Vi:* valor inicial de la variable índice
- *Vf:* valor final de la variable índice
- *[incremento]:* el número que se incrementa (o decrementa) a la variable índice en cada iteración del bucle, si se omite es 1.

Ejemplo estructura "***Para***" o "***Desde***"

Inicio Programa

Para *i = 1* **hasta** *10 incremento 1*

Mostrar *i*

Fin Para

Fin Programa

Estructura "***Mientras***"

Mientras *condición*

...

instrucción 1

instrucción 2

instrucción 3

...

Fin Mientras

Lo primero que la computadora hace es examinar la condición

- Si la *condición* **se cumple**:
  Se ejecutan la instrucción 1, instrucción 2, instrucción 3, etc.
  Las estará repitiendo hasta que la *condición* no se cumpla, entonces se sale del ciclo y se siguen ejecutando la o las instrucciones que vienen a continuación y están fuera del bucle.
- Si la *condición* NO se cumple:
  No entrará en el ciclo. Se ejecutan las instrucciones que vienen después del bucle o bien terminará el programa.

Ejemplo estructura "***Mientras***"

Inicio Programa

edad ← 15

Mientras *edad < 18*

Mostrar *"Es menor de edad"*

Leer *edad*

Fin Mientras

Mostrar "Es mayor de edad"

Fin Programa

Estructura "***Repetir***"

Repetir

…

instrucción 1

instrucción 2

instrucción 3

…

hasta *condición*

Instrucción x

La estructura ***Repetir*** cumple la misma función que la estructura ***Mientras***. La diferencia está en que la estructura ***Mientras*** comprueba la condición al inicio, y la estructura ***Repetir*** lo hace al final; por lo que esta se ejecuta al menos una vez.

Estructura "***Repetir***"

Lo que la computadora hace al ejecutar la estructura ***Repetir*** es:

1. Se ejecuta la *instrucción 1*, *instrucción 2, instrucción 3*, …
2. Se evalúa la *condición*.
   - Si la *condición* NO se cumple, entonces se vuelve a repetir el ciclo y se ejecutan *instrucción 1, instrucción 2, instrucción 3*, … .
   - Si la *condición* se cumple, se sale de ciclo y se ejecutan las instrucciones que vienen después del bucle (*instrucción x*) o bien termina el programa.

Ejemplo estructura "***Repetir***"

Inicio Programa

edad ← 15

Repetir

Mostrar *"Es menor de edad"*

Leer *edad*

hasta *edad < 18*

Fin Programa

Diferencias entre estructuras **Mientras** y **Repetir**

MIENTRAS	REPETIR
Comprobación de la *condición* al **inicio** (antes de entrar al bucle)	Comprobación de la *condición* al **final** (después de haber ingresado una vez al bucle)
Las instrucciones del cuerpo del bucle se ejecutan en forma repetitiva si la condición es **verdadera**.	Las instrucciones del cuerpo del bucle se ejecutan si la condición es **falsa**.
Las acciones del bucle se pueden ejecutar cero o más veces.	Las acciones del bucle se ejecutan por lo menos una vez.

Ejercicio

Realizar un programa para un estacionamiento de autos, que nos permita:

- Inicialmente, ingresar el numero de lugares que tiene la cochera.
- Luego, que nos permita, reiteradamente:
  - Si ingresa un auto, ingresamos **1**.
  - Si sale un auto, ingresamos **-1**.
  - Si queremos finalizar el programa, ingresamos **0**.

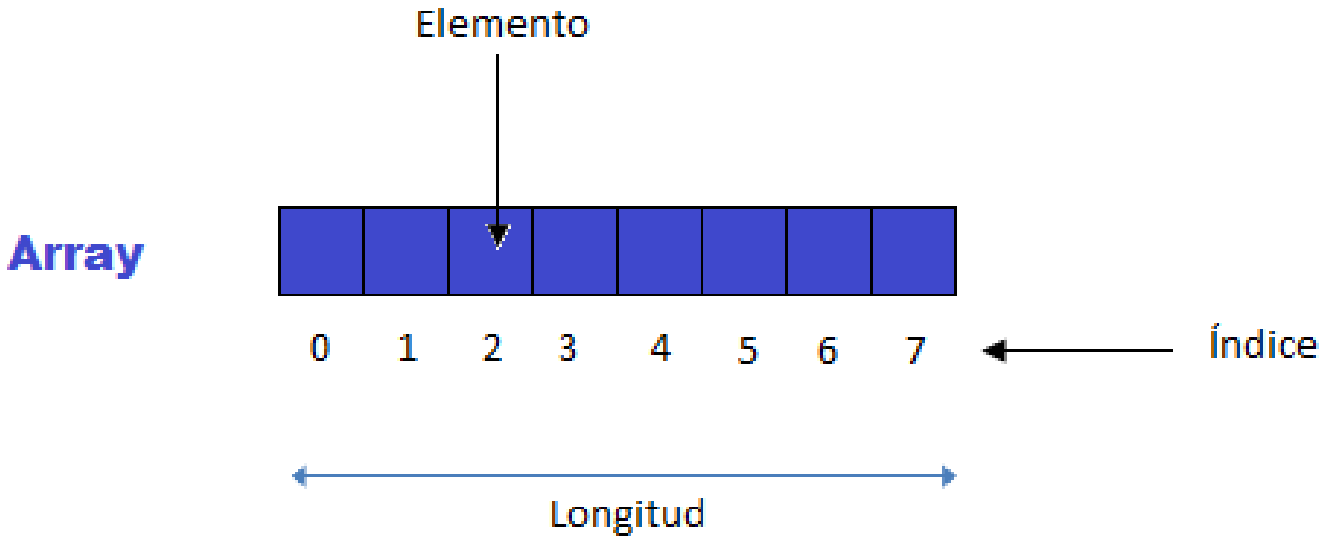
Cada vez que ingresa o sale un auto, que nos muestre un mensaje indicando "*quedan lugares disponibles en la cochera*", y si los hay, que nos indique la cantidad de lugares disponibles.

Array

- Un **array**, es un tipo de dato estructurado que permite almacenar un conjunto de datos homogéneo, es decir, todos del mismo tipo y relacionados.
  Cada uno de los elementos que componen un vector (o array) pueden ser de tipo simple (como *caracteres, entero o real*), o de tipo compuesto o estructurado (como son *vectores, estructuras, listas*).

Array

- A los datos almacenados en un *array* se los denomina **elementos**.
- Al número de elementos de un *array* se les denomina **tamaño o rango del vector**.
- Para acceder a los elementos individuales de un *array* se emplea un **índice** (es un número entero no negativo que indica la posición del elemento dentro del *array*)



Array

- Para declarar un *array*:

  dimensionar *variable*[4]

  dimensionar *variable*[4,4]

- Para inicializar un array:

  variable[1] ← 15

  variable[3] ← “Hola que tal?”

Ejercicio #1

Un colegio desea saber qué porcentaje de niños y de niñas hay en el curso actual.

Diseñar un algoritmo para este propósito (recuerda que para calcular el porcentaje puedes hacer una regla de 3).

Ejercicio #2

Dadas dos variables numéricas *A* y *B*, que el usuario debe ingresar, se pide realizar un algoritmo que intercambie los valores de ambas variables y muestre el valor final de las mismas.

Ejercicio #3

Hacer un seudocódigo que imprima el mayor y el menor número de una serie de cinco números que vamos introduciendo por teclado.

Ejercicio #4

Crear un *arrray* unidimensional de 20 elementos.

Que permita ingresar 20 nombres de personas, y luego los imprima uno a uno.

Ejercicio #5

Crear un *arrray* que almacene 5 números y los ordene de manera ascendente, y luego los imprima.

Ejercicio #6

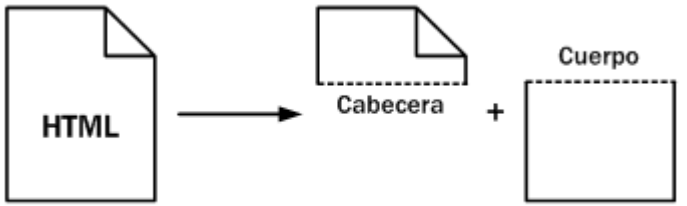
Hacer un programa que lea las calificaciones de un alumno en 10 exámenes, las almacene en un vector y calcule e imprima su promedio, la nota más alta y la nota más baja.

HTML

- **HTML** es un lenguaje de etiquetas, y las páginas web habituales están formadas por cientos o miles de pares de etiquetas.
- Más concretamente, **HTML** es el *lenguaje* con el que se *"escriben"* la mayoría de páginas web.
- Para poder crear una página **HTML** se requiere un simple editor de texto y un navegador de internet (Chrome, IExplorer, FireFox, Safari etc.)

Estructura de una página **HTML**

- Las páginas **HTML** se dividen en dos partes: la *cabecera* y el *cuerpo*.
- La *cabecera* incluye información sobre la propia página, como por ejemplo su título y su idioma.
- El *cuerpo* de la página incluye todos sus contenidos, como párrafos de texto e imágenes.



Estructura de una página **HTML**

A continuación se muestra el código HTML de una página web muy sencilla:

```
<html>

<head>
   <title>El primer documento HTML</title>
</head>

<body>
   <p>
        El lenguaje HTML es <strong>tan sencillo</strong> que prácticamente se
        entiende sin estudiar el significado de sus etiquetas principales.
   </p>
</body>

</html>
```

Primer documento **HTML**

Si quieres probar este primer ejemplo, debes hacer lo siguiente:

1. Abre un editor de archivos de texto y crea un archivo nuevo.
2. Copia el código HTML mostrado anteriormente y pégalo tal cual en el archivo que has creado.
3. Guarda el archivo con el nombre que quieras, pero con la extensión ***.html***

Para que el ejemplo anterior funcione correctamente, es imprescindible que utilices un editor de texto sin formato (ej: Bloc de notas, Wordpad, EmEditor, UltraEdit, Notepad++, etc.) pero **NO** puedes utilizar un procesador de textos como Word o Open Office.

Después de crear el archivo con el contenido *HTML*, ya se puede abrir con cualquier navegador haciendo doble clic sobre el archivo.

Estructura interna **HTML**

Es importante conocer las tres etiquetas principales de un documento HTML (***<html>***, ***<head>***, ***<body>***):

- ***<html>***: indica el comienzo y el final de un documento HTML. Ninguna etiqueta o contenido puede colocarse antes o después de la etiqueta *<html>.*
  En el interior de la etiqueta ***<html>*** se definen la cabecera y el cuerpo del documento HTML y todo lo que se coloque fuera de la etiqueta ***<html>*** se ignora.

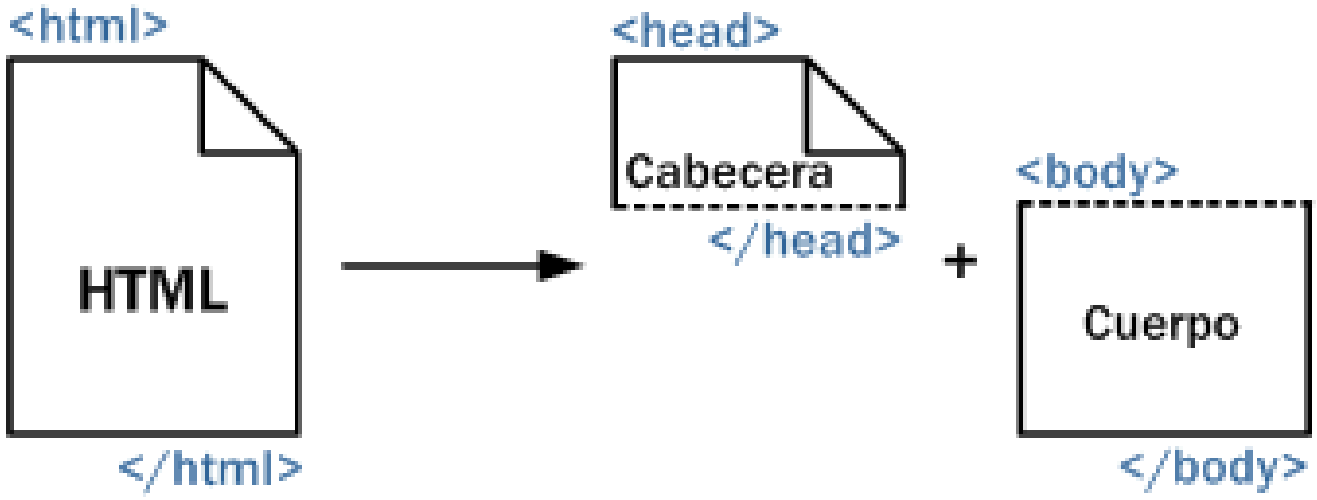
Estructura interna **HTML**

- ***<head>:*** delimita la parte de la cabecera del documento.
  La cabecera contiene información sobre el propio documento HTML, como por ejemplo su título y el idioma de la página. Los contenidos indicados en la cabecera no son visibles para el usuario, con la excepción de la etiqueta ***<title>***, que se utiliza para indicar el título del documento y que los navegadores lo visualizan en la parte superior izquierda de la ventana del navegador

Estructura interna **HTML**

- ***<body>***: delimita el cuerpo del documento HTML.
  El cuerpo encierra todos los contenidos que se muestran al usuario (párrafos de texto, imágenes, tablas). En general, el ***<body>*** de un documento contiene cientos de etiquetas HTML, mientras que el ***<head>*** no contiene más que unas pocas.

Estructura interna **HTML**



Etiquetas principales

- HTML define 91 etiquetas que los diseñadores pueden utilizar para marcar los diferentes elementos que componen una página:

  a, abbr, acronym, address, applet, area, b, base, basefont, bdo, big, blockquote, body, br, button, caption, center, cite, code, col, colgroup, dd, del, dfn, dir, div, dl, dt, em, fieldset, font, form, frame, frameset, h1, h2, h3, h4, h5, h6, head, hr, html, i, iframe, img, input, ins, isindex, kbd, label, legend, li, link, map, menu, meta, noframes, noscript, object, ol, optgroup, option, p, param, pre, q, s, samp, script, select, small, span, strike, strong, style, sub, sup, table, tbody, td, textarea, tfoot, th, thead, title, tr, tt, u, ul, var.

- De todas las etiquetas disponibles, las siguientes se consideran obsoletas y no se pueden utilizar: ***applet***, ***basefont***, ***center***, ***dir***, ***font***, ***isindex***, ***menu***, ***s***, ***strike***, ***u***.

Salto de línea ***
***

- Todo el texto que disponemos en el cuerpo de la página aparece en la misma línea, no importa si cuando tipeamos la página disponemos cada palabra en una línea distinta (es decir un navegador no tiene en cuenta la tecla ENTER).
- Para indicarle al navegador que queremos que continúe en la próxima línea debemos hacerlo con el elemento HTML **
**.
- Cuando aparece la marca **
** el navegador continúa con el texto en la línea siguiente. Es uno de los pocos elementos HTML que no tiene marca de cerrado como habíamos visto hasta ahora.

Párrafo *** ... ***

- Un párrafo es una oración o conjunto de oraciones referentes a un mismo tema. Todo lo que encerremos entre las marcas ** ** y ** ** aparecerá separado por un espacio con respecto al próximo párrafo.
- Dentro de un párrafo puede haber saltos de línea **
**.
- Para recordar el nombre de este elemento HTML: ** ** viene de la palabra *paragraph*.

Problema: Confeccione una página que muestre en un párrafo datos referentes a sus estudios y en otro párrafo su nombre y mail.

Títulos
<h1> <h2> <h3> <h4> <h5> <h6>

- Otros elementos HTML muy utilizados son para indicar los títulos, contamos con los elementos:
  <h1>…</h1>
  <h2>…</h2>
  <h3>…</h3>
  <h4>…</h4>
  <h5>…</h5>
  <h6>…</h6>

Títulos
<h1> <h2> <h3> <h4> <h5> <h6>

- El título de mayor nivel es ***<h1>***, es decir el que tienen normalmente una fuente de mayor tamaño.
- Según la importancia del título utilizaremos alguno de estos elementos HTML.
- Los buscadores que indexan contenido (Google, Bing, Yahoo etc.) hacen incapié en los títulos para identifica los temas que tratan las páginas.
- No hay que confundir el título de la página que va en la sección del *head* con el elemento *title*.
- Cada título aparece siempre en una línea distinta, no importa si lo tipeamos seguido en el archivo.
- ***<h1>*** viene de la palabra heading, que significa título.

Títulos
<h1> <h2> <h3> <h4> <h5> <h6>

- **Problema:** Confeccionar el titular de un periódico con un título de nivel 1. Luego definir dos títulos de segundo nivel con los textos (Noticias políticas y Noticias deportivas), en cada una de estas secciones definir dos titulares de tercer nivel con un párrafo cada una.

Énfasis
****** y ******

- Enfatizar algo significa realzar la importancia de una cosa, por ejemplo una palabra o conjunto de palabras.
- Así como tenemos seis niveles de títulos para enfatizar un bloque contamos con dos elementos que son (****** ******).
- El elemento de mayor fuerza de énfasis es **strong** (negrita) y le sigue **em** (cursiva).
- Para recordar el nombre de estos elementos HTML:
  ****** viene de *empathize* que significa énfasis.
  ****** significa fuerte.

Problema: Confeccionar una página que muestre la definición de tres palabras. Aplicar el elemento ***strong*** a cada palabra previo a su definición. Luego agregar el elemento "***em***" a una o a un conjunto de palabras dentro de la definición.

Hipervínculos ***<a>***

- El elemento más importante que tiene una página de internet es el *hipervínculo*, estos nos permiten acceder a otras páginas (las mismas pueden estar o no en el mismo sitio web).
- La etiqueta de hipervínculo es:

```
<a href="pagina2.html">Noticias</a>
```

Hipervínculos ***<a>***

- Como vemos, se trata de otro elemento HTML que tiene comienzo y fin de etiqueta.
- Lo nuevo que aparece en este elemento es el concepto de una ***propiedad***. Una propiedad se incorpora en el comienzo de una marca y tiene un nombre y un valor.
- El valor de la propiedad debe ir entre comillas dobles.
- Toda propiedad toma el valor que se encuentra seguidamente del caracter "="
- La propiedad ***href*** del elemento "***a***" hace referencia a la página que debe mostrar el navegador si el visitante hace clic sobre el hipervínculo.

Hipervínculos ***<a>***

```
<html>
  <head>
   <title>Título de la página 1</title>
  </head>

  <body>
   <a href=“pagina2.html">Noticias</a>
  </body>
</html>
```

El valor de la propiedad ***href*** en este caso es *pagina2.html* (es otro archivo HTML que debe encontrarse en nuestro sitio y en el mismo directorio)

Hipervínculos ***<a>***

- Las **URL absolutas** incluyen todas las partes de la URL (protocolo, servidor y ruta) por lo que no se necesita más información para obtener el recurso enlazado.
  Ejemplo:
  http://www.ejemplo.com/ruta1/ruta2/pagina2.html

- Las **URL relativas** prescinden de algunas partes de las URL para hacerlas más breves.
  Las URL relativas se construyen a partir de las URL absolutas y prescinden de la parte del protocolo, del nombre del servidor e incluso de parte o toda la ruta del recurso enlazado.
  Ejemplo:
  /ruta1/ruta2/pagina2.html

Hipervínculos ***<a>***

Problema: Confeccionar una página principal con dos hipervínculos a las páginas *pagina2.html* y *pagina3.html*
Luego en las dos páginas secundarias, poner hipervínculos a la página principal.

Listas

- En ocasiones, es posible agrupar determinadas palabras o frases en un conjunto de elementos que tienen más significado de forma conjunta.

- El lenguaje HTML define tres tipos diferentes de listas para agrupar los elementos: **listas no ordenadas** (se trata de una colección simple de elementos en la que no importa su orden)**, listas ordenadas** (similar a la anterior, pero los elementos están numerados y por tanto, importa su orden) y **listas de definición** (un conjunto de términos y definiciones similar a un diccionario)**.**

Listas no ordenadas ***
***

- Las listas no ordenadas son las más sencillas y las que más se utilizan.
- Una lista no ordenada es un conjunto de elementos relacionados entre sí pero para los que no se indica un orden o secuencia determinados.
- La etiqueta ***<ul>*** encierra todos los elementos de la lista y la etiqueta ***<li>*** cada uno de sus elementos.
- Para recordar el nombre de estos elementos HTML:
  <ul> viene de las palabras *unordered list*
  <li> viene de las palabras *list item*

Listas no ordenadas ***
***

```
<html>
  <head> <title>Ejemplo de etiqueta UL</title> </head>
<body>
 <h1>Menú</h1>
<ul>
  <li>Inicio</li>
  <li>Noticias</li>
  <li>Artículos</li>
</ul>
</body>
</html>
```

Listas ordenadas ***
***

- Las listas ordenadas son casi idénticas a las listas no ordenadas, salvo que en este caso los elementos relacionados se muestran siguiendo un orden determinado.
- La lista ordenada se define mediante la etiqueta ***<ol>***.
- Los elementos de la lista se definen mediante la etiqueta ***<li>***, la misma que se utiliza en las listas no ordenadas.

Listas ordenadas ***
***

```
<html>
  <head> <title>Ejemplo de etiqueta OL</title> </head>
<body>
 <h1>Instrucciones</h1>
<ol>
  <li>Enchufar correctamente</li>
  <li>Comprobar conexiones</li>
  <li>Encender el aparato</li>
</ol>
</body>
</html>
```

Imágenes ***<img>***

- Para insertar una imagen dentro de una página debemos utilizar el elemento HTML **<img>**, la misma no tiene una etiqueta de finalización (similar a la marca **
)**.
- Generalmente, la imagen se encuentra en el mismo servidor donde se almacenan nuestras páginas HTML. Los formatos clásicos son los archivos con extensiones *gif, jpg y png*.
- La sintaxis de esta marca es:

```
<img src=“imagen.png" alt=“Imagen de Mi Sitio" />
```

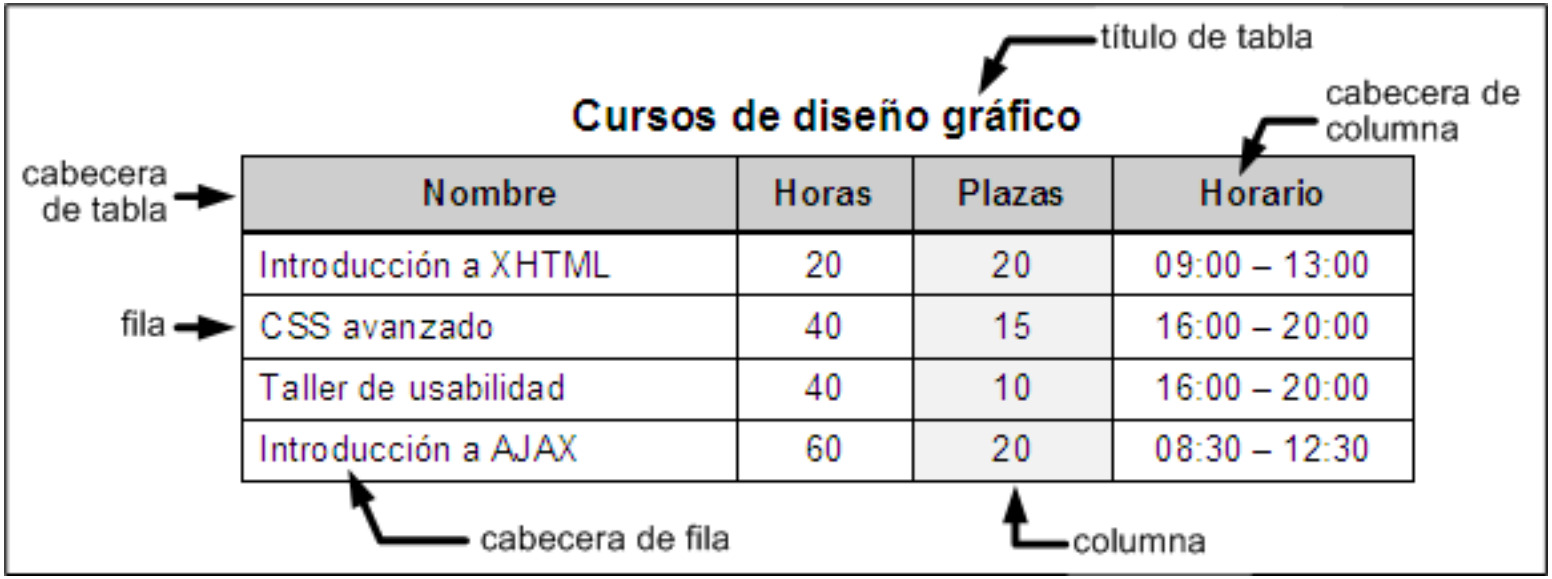
Imágenes ***<img>***

- Como ***<img>*** es una etiqueta vacía, no tiene etiqueta de cierre, sin embargo debe llevar los caracteres ***/>*** al final de la etiqueta.

Problema: Desarrollar una página que muestre dos imagenes cualquiera (obtenidas de Google). Disponer un título a cada imagen.

Tablas

- Las tablas en HTML utilizan los mismos conceptos de filas, columnas, cabeceras y títulos que los que se utilizan en cualquier otro entorno de publicación de documentos:



Nombre	Horas	Plazas	Horario
Introducción a XHTML	20	20	09:00 – 13:00
CSS avanzado	40	15	16:00 – 20:00
Taller de usabilidad	40	10	16:00 – 20:00
Introducción a AJAX	60	20	08:30 – 12:30

Tablas

Las tablas más sencillas de HTML se definen con tres etiquetas: ***<table>*** para crear la tabla, ***<tr>*** para crear cada fila y ***<td>*** para crear cada columna.

Ejemplo:

```
<html>
 <head>
  <title>Ejemplo de tabla sencilla</title>
 </head>

 <body>
  <h1>Listado de cursos</h1>
  <table>
   <tr>
    <td><strong>Curso</strong></td>
    <td><strong>Horas</strong></td>
    <td><strong>Horario</strong></td>
   </tr>
   <tr>
    <td>CSS</td>
    <td>20</td>
    <td>16:00 - 20:00</td>
   </tr>
   <tr>
    <td>HTML</td>
    <td>20</td>
    <td>16:00 - 20:00</td>
   </tr>
  </table>
 </body>
</html>
```

Tablas

Problema: Confeccionar una tabla que muestre en la primer columna los nombre de distintos empleados de una compañía y en la segunda el sueldo bruto (la compañía tiene 4 empleados)

Comentarios <!-- -->

- Un comentario es un texto que solo le interesa a la persona que desarrolló la página, el navegador ignora todo el contenido que se encuentra dentro de esta etiqueta.
- Otro uso muy habitual cuando estamos desarrollando la página si queremos deshabilitar una parte del código podemos encerrarla entre los caracteres de comentarios.
- La sintaxis para definir un comentario es:
  <!– *Aquí va el comentario* -->

Formularios ***<form> </form>***

- Un formulario permite que el visitante al sitio cargue datos y sean enviados al servidor.
- Es el medio ideal para registrar comentarios del visitante sobre el sitio, solicitar productos, sacar turnos etc.
- Para crear un formulario debemos utilizar el elemento ***form***, que tiene etiqueta de comienzo y fin. Dentro de la etiqueta ***form*** veremos otros elementos para crear botones, editores de línea, cuadros de chequeo, radios de selección etc.

Formularios *<form> </form>*

```
<form action="http://www.mipagina.php" method="post">
   Escribe tu nombre:  <input type="text" name="nombre" value="" />
   <br/>
   <input type="submit" value="Enviar" />
</form>
```

Formularios ***<form> </form>***

- La mayoría de formularios utilizan sólo los atributos ***action*** y ***method***.
- El atributo ***action*** indica la *URL* o dirección de la página que se encarga de procesar los datos introducidos por los usuarios. Esta página también se encarga de generar la respuesta que muestra el navegador.
- El atributo ***method*** establece la forma en la que se envían los datos del formulario al servidor.
  Los dos valores que se utilizan en los formularios son GET y POST.
  De esta forma, casi todos los formularios incluyen el atributo *method="get"* o el atributo *method="post"*.

Formularios ***<form> </form>***

- Al margen de otras diferencias técnicas, el método *POST* permite el envío de mucha más información que el método *GET*.
  En general, el método *GET* admite como máximo el envío de unos 500 bytes de información. La otra gran limitación del método *GET* es que no permite el envío de archivos adjuntos con el formulario.
  Además, los datos enviados mediante *GET* se ven en la barra de direcciones del navegador (se añaden al final de la URL de la página), mientras que los datos enviados mediante *POST* no se pueden ver tan fácilmente.

Formularios ***<form> </form>***

- El ejemplo más común de formulario con método *GET* es el de los buscadores. Si realizas una búsqueda con tu buscador favorito, verás que las palabras que has introducido en tu búsqueda aparecen como parte de la URL de la página de resultados.

Elementos de formularios

- Los elementos de formulario como botones y cuadros de texto también se denominan "campos de formulario" y "controles de formulario".
- La mayoría de controles se crean con la etiqueta ***<input>***, por lo que su definición formal y su lista de atributos es muy extensa:

Elementos de formularios

- `type = "text | password | checkbox | radio | submit | reset | file | hidden | image | button"` - Indica el tipo de control que se incluye en el formulario
- `name = "texto"` - Asigna un nombre al control (es imprescindible para que el servidor pueda procesar el formulario)
- `value = "texto"` - Valor inicial del control
- `size = "unidad_de_medida"` - Tamaño inicial del control (para los campos de texto y de password se refiere al número de caracteres, en el resto de controles se refiere a su tamaño en píxel)
- `maxlength = "numero"` - Máximo número de caracteres para los controles de texto y de password
- `checked = "checked"` - Para los controles checkbox y radiobutton permite indicar qué opción aparece preseleccionada
- `disabled = "disabled"` - El control aparece deshabilitado y su valor no se envía al servidor junto con el resto de datos
- `readonly = "readonly"` - El contenido del control no se puede modificar
- `src = "url"` - Para el control que permite crear botones con imágenes, indica la URL de la imagen que se emplea como botón de formulario
- `alt = "texto"` - Descripción del control

Cuadro de texto

- Se trata del elemento más utilizado en los formularios. En el caso más sencillo, se muestra un cuadro de texto vacío en el que el usuario puede escribir cualquier texto:

- A continuación se muestra el código HTML correspondiente al ejemplo anterior:

```
Nombre <br/>
<input type="text" name="nombre" value="" />
```

Cuadro de texto (**name**)

- El atributo ***type*** diferencia a cada uno de los diez controles que se pueden crear con la etiqueta <*input*>. Para los cuadros de texto, su valor es **text**.
- El atributo ***name*** es el más importante en los campos del formulario. De hecho, si un campo no incluye el atributo ***name***, sus datos no se envían al servidor.
  El valor que se indica en el atributo ***name*** es el nombre que utiliza la aplicación del servidor para obtener el valor de este campo de formulario.
- No se deben utilizar caracteres problemáticos en programación (espacios en blanco, acentos y caracteres como **ñ** o **ç**)

Cuadro de texto (**name**)

- Cuando el usuario pulsa el botón de envío del formulario, el navegador envía los datos a una aplicación del servidor para que procese la información y genere una respuesta adecuada.
- En el servidor, la aplicación que procesa los datos debe obtener en primer lugar toda la información introducida por el usuario. Para ello, utiliza el valor del atributo ***name*** para obtener los datos de cada control del formulario.

Cuadro de texto (**value**)

- El atributo ***value*** se emplea para establecer el valor inicial del cuadro de texto.
- Si se crea un formulario para insertar datos, los cuadros de texto deberían estar vacíos. Por lo tanto, o no se añade el atributo ***value*** o se incluye con un valor vacío ***value***=""".
- Si por el contrario se crea un formulario para modificar datos, lo lógico es que se muestren inicialmente los datos guardados en el sistema. En este caso, el atributo ***value*** incluirá el valor que se desea mostrar:

```
<input type=“text” name=“nombre” value=“Juan Perez" />
```

Cuadro de texto (**size**)

- Si no se especifica un tamaño, el navegador muestra el cuadro de texto con un tamaño predeterminado. El atributo ***size*** permite establecer el tamaño, en caracteres, con el que se muestra el cuadro de texto. Su uso es imprescindible en muchos formularios, en los que algunos campos como la dirección deben mostrar más caracteres de lo normal (***<input size="100"*** ...) y otros campos como el código postal deben mostrar menos caracteres de lo normal (***<input size="5"***...).

```
<input type=“text” name=“nombre” size=“50" />
```

Cuadro de texto (**maxlenght**)

- Además de controlar el tamaño con el que se muestra un cuadro de texto, también se puede limitar el tamaño del texto introducido.
- El atributo ***maxlengthh*** permite establecer el máximo número de caracteres que el usuario puede introducir en un cuadro de texto.
- Su uso es imprescindible para campos como el código postal, el DNI y cualquier otro dato con formato predefinido y limitado.

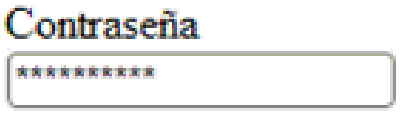
```
<input type=“text” name=“dni” maxlenght=“9" />
```

Cuadro de texto (**readonly & disabled**)

- El atributo **readonly** permite que el usuario pueda ver los contenidos del cuadro de texto pero no pueda modificarlos.
- El atributo **disabled** deshabilita un cuadro de texto de forma que el usuario no pueda modificarlo y además, el navegador no envía sus datos al servidor.

Cuadro de contraseña

- La única diferencia entre este control y el cuadro de texto normal es que el texto que el usuario escribe en un cuadro de contraseña no se ve en la pantalla. En su lugar, los navegadores ocultan el texto utilizando asteriscos o círculos, por lo que es ideal para escribir contraseñas y otros datos sensibles.

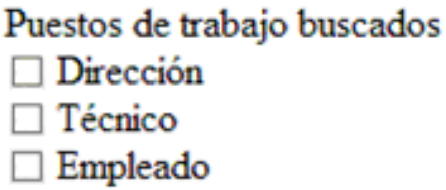



```
Contraseña <br/>
<input type=“password” name=“contrasena” value="" />
```

- Cambiando el valor del atributo **type** por ***password*** se transforma el cuadro de texto normal en un cuadro de contraseña. Todos los demás atributos se utilizan de la misma forma y tienen el mismo significado

Checkbox

- Los **checkbox** o "casillas de verificación" son controles de formulario que permiten al usuario seleccionar y deseleccionar opciones individualmente. Aunque en ocasiones se muestran varios **checkbox** juntos, cada uno de ellos es completamente independiente del resto. Por este motivo, se utilizan cuando el usuario puede activar y desactivar varias opciones relacionadas pero no excluyentes.




```
Puestos de trabajo buscados <br/>
<input name="puesto_directivo" type="checkbox" value="direccion"/> Dirección
<input name="puesto_tecnico" type="checkbox" value="tecnico"/> Técnico
<input name="puesto_empleado" type="checkbox" value="empleado"/> Empleado
```

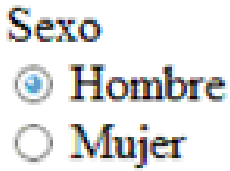
Checkbox

- El valor del atributo ***type*** para estos controles de formulario es **checkbox**. Como se muestra en el ejemplo anterior, el texto que se encuentra al lado de cada checkbox no se puede establecer mediante ningún atributo, por lo que es necesario añadirlo manualmente fuera del control del formulario. Si no se añade un texto al lado de la etiqueta *<input />* del checkbox, el usuario sólo ve un pequeño cuadrado sin ninguna información relativa a la finalidad de ese checkbox.
- El valor del atributo ***value***, junto con el valor del atributo ***name***, es la información que llega al servidor cuando el usuario envía el formulario.
- Si se quiere mostrar un checkbox seleccionado por defecto, se utiliza el atributo ***checked***. Si el valor del atributo es ***checked***, el checkbox se muestra seleccionado. En cualquier otro caso, el checkbox permanece sin seleccionar.

```
<input type="checkbox" checked=“ckecked”/> Checkbox seleccionado por defecto
```

Radiobutton

- Los controles de tipo **radiobutton** son similares a los controles de tipo **checkbox**, pero presentan una diferencia muy importante: son mutuamente excluyentes.
- Los **radiobutton** se utilizan cuando el usuario solamente puede escoger una opción entre las distintas opciones relacionadas que se le presentan. Cada vez que se selecciona una opción, automáticamente se deselecciona la otra opción que estaba seleccionaba.




```
Sexo <br/>
<input type="radio" name="sexo" value="hombre" checked="ckecked"/> Hombre
<input type="radio" name="sexo" value="mujer"/> Mujer
```

Radiobutton

- El valor del atributo ***type*** para estos controles de formulario es **radio**.
- El atributo ***name*** se emplea para indicar los *radiobutton* que están relacionados. Por lo tanto, cuando varios *radiobutton* tienen el mismo valor en su atributo ***name***, el navegador sabe que están relacionados y puede deseleccionar una opción del grupo de *radiobutton* cuando se seleccione otra opción.

Botón de envío de formulario

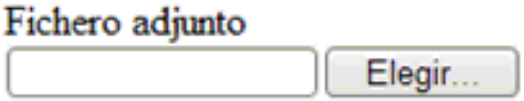
- La mayoría de formularios dispone de un botón para enviar al servidor los datos introducidos por el usuario:


```
<input type=“submit” name=“buscar" value=“Buscar"/>
```

- El valor del atributo ***type*** para este control de formulario es ***submit***.
  El navegador se encarga de enviar automáticamente los datos cuando el usuario hace clic sobre este tipo de botón.
- El valor del atributo ***value*** es el texto que muestra el botón. Si no se establece el atributo ***value***, el navegador muestra el texto predefinido Enviar consulta.

Archivos adjuntos

- Los formularios también permiten adjuntar archivos para subirlos al servidor. Aunque desde el punto de vista de HTML y del navegador no existe ninguna limitación sobre el número, tipo o tamaño total de los archivos que se pueden adjuntar, todos los servidores añaden restricciones por motivos de seguridad.




```
<input type="file" name="adjunto" />
```

Archivos adjuntos

- El valor del atributo ***type*** para este control de formulario es ***file***. El navegador se encarga de mostrar un cuadro de texto donde aparece el nombre del archivo seleccionado y un botón que permite navegar por los directorios y archivos del ordenador del usuario.
- Si se incluye un control para adjuntar archivos, es obligatorio añadir el atributo ***enctype*** en la etiqueta **<form>** del formulario. El valor del atributo ***enctype*** debe ser ***multipart/form-data***, por lo que la etiqueta **<form>** de los formularios que permiten adjuntar archivos siempre es:

```
<form action="index.php" method="post" enctype="multipart/form-data"/>
. . .
</form>
```

Campos ocultos

- Los campos ocultos se emplean para añadir información oculta en el formulario:

*Los campos ocultos
no se ven en pantalla

```
<input type="hidden" name="identificado" value="10195" />
```

- El valor del atributo ***type*** para este control de formulario es ***hidden***. Los campos ocultos NO se muestran por pantalla, de forma que el usuario desconoce que el formulario los incluye. Normalmente los campos ocultos se utilizan para incluir información que necesita el servidor pero que no es necesario o no es posible que la establezca el usuario.

Área de Texto

- Las áreas de texto son útiles cuando se debe introducir una gran cantidad de texto, ya que es mucho más cómodo de introducir que en un campo de texto normal

```
<textarea name="descripcion" rows="5" cols="10"> </textarea>
```

- Los atributos más utilizados en las etiquetas **<textarea>** son los que controlan su anchura y altura.
- La anchura del área de texto se controla mediante el atributo ***cols***, que indica las columnas o número de caracteres que se podrán escribir como máximo en cada fila.
- La altura del área de texto se controla mediante ***rows***, que indica directamente las filas de texto que serán visibles.

Listas desplegables

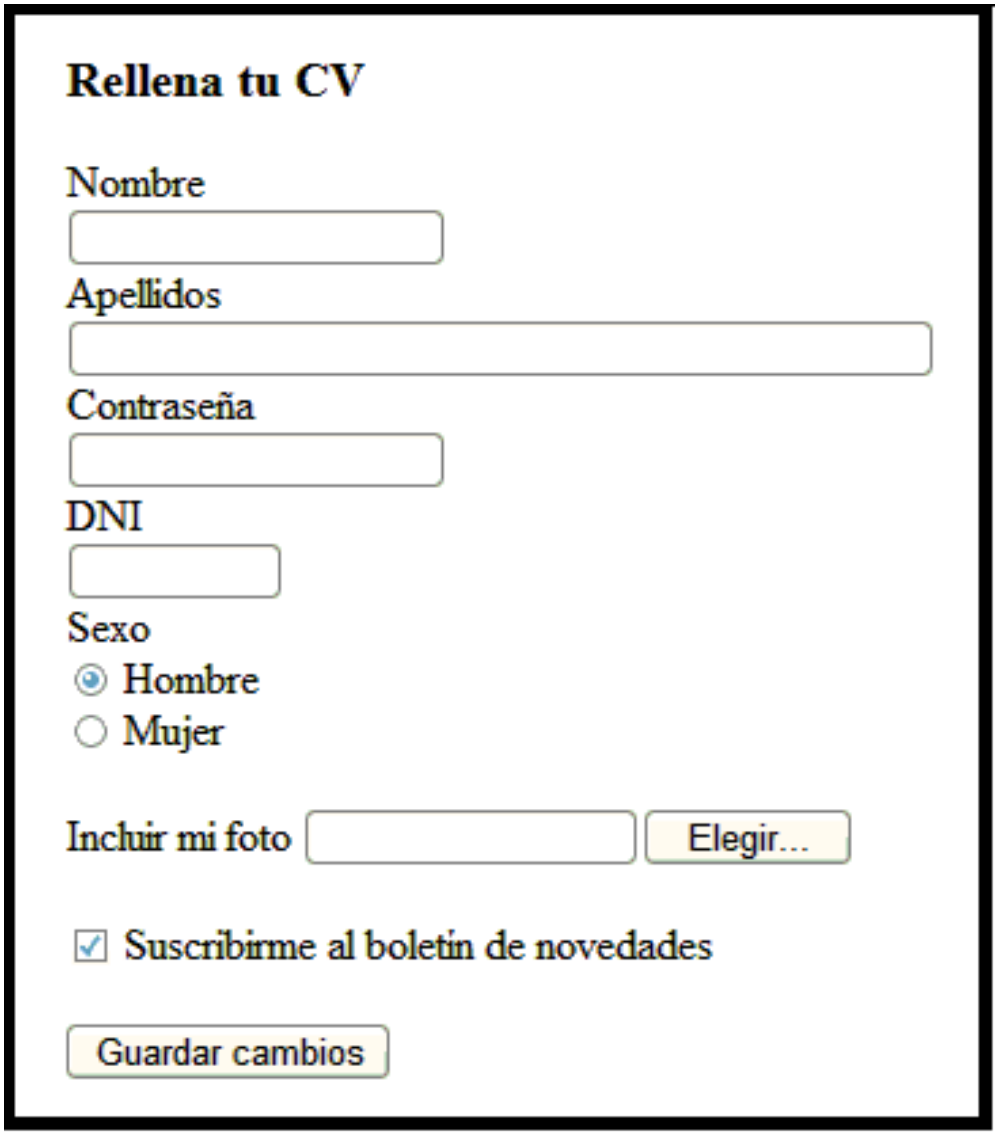
- La etiqueta **<select>** define la lista y encierra todas las opciones que muestra la lista.
- Cada una de las opciones de la lista se define mediante una etiqueta **<option>**.
- El atributo ***value*** de cada opción es obligatorio, ya que es el dato que se envía al servidor cuando el usuario envía el formulario.
- Para seleccionar por defecto una opción al mostrar la lista, se añade el atributo ***selected*** a la opción deseada.

```
<select name="color" >
   <option value="rojo" selected="selected"> Rojo</option>
   <option value="verde"> Verde </option>
   <option value="azul"> Azul </option>
   <option value="negro"> Negro </option>
</select>
```

Problema 1

Escribir el código HTML necesario para crear el formulario que se muestra en la siguiente imagen, teniendo en cuenta:

1. Elegir el método más adecuado para el formulario (GET o POST) y cualquier otro atributo necesario.
2. La aplicación que se encarga de procesar el formulario se encuentra en la raíz del servidor, carpeta "**php**" y archivo "**insertar_cv.php**" .
3. El nombre puede tener 30 caracteres como máximo, los apellidos 80 caracteres y la contraseña 10 caracteres como máximo.
4. Asignar los atributos adecuados al campo del DNI.
5. Por defecto, debe estar marcada la casilla de suscripción al boletín de novedades.



Formularios HTML + PHP

- Una actividad fundamental en PHP es la recolección de datos de un formulario HTML.
- El proceso para el manejo de FORMULARIOS requiere generalmente dos páginas, una que implementa el formulario (*HTML*) y otra que procesa los datos cargados en el formulario (*PHP*).

Formularios HTML + PHP

```
<form action="pagina2.php” method="post" />
 <input type="text" name="nombre" />
 <br>
 <input type="submit" value="Confirmar" />
</form>
```

- Para acceder a los datos ingresados en el formulario, existe un vector llamado **$_REQUEST** indicando como subíndice el nombre del cuadro de texto que definimos en el formulario (dicho nombre es sensible a mayúsculas y minúsculas)
  En nuestro ejemplo sería: **$_REQUEST[‘nombre’]** para acceder al dato ingresado en el cuadro de texto “*nombre*” del formulario.

Formularios HTML + PHP

- Si quisiéramos mostrar por pantalla el valor ingresado en el formulario:

  echo $_REQUEST[‘nombre’];

- También, en lugar de **$_REQUEST**, se puede utilizar **$_GET** o **$_POST** según el método (***method***) utilizado en el formulario.

  echo $_POST[‘nombre’];

Problema 2

- Solicitar que se ingrese por teclado el nombre de una persona y disponer tres controles de tipo ***radio*** que nos permitan seleccionar si la persona:
  - no tiene estudios
  - estudios primarios
  - estudios secundarios.

  En la página que procesa el formulario mostrar el nombre de la persona y un mensaje indicando el tipo de estudios que posee.

Solución

```
<?php
  echo $_REQUEST['nombre'];
  echo "<br>";
  if ($_REQUEST['radio1'] == "sin")  {
    echo "Sin estudios.";
  }
  if ($_REQUEST['radio1'] == "primario")  {
      echo "Estudios primarios.";
  }
  if ($_REQUEST['radio1'] == "secundario")  {
    echo "Estudios secundarios.";
  }
?>
```

Problema 3

- Confeccionar un formulario que solicite la carga del nombre de una persona y que permita seleccionar una serie de deportes que practica (futbol, basket, tennis, voley)
  Mostrar en la página que procesa el formulario la cantidad de deportes que practica.

Problema 4

- Confeccionar un formulario que solicite el ingreso del nombre de una persona y un control ***select*** (en este último permitir la selección de los ingresos mensuales de la persona: 1-1000, 1001-3000, >3000)
  En la página que procesa el formulario mostrar un mensaje si debe pagar impuestos a las ganancias (si supera 3000)

Problema 5

- Confeccionar una página que muestre un contrato dentro de un ***textarea***, disponer puntos suspensivos donde el operador debe ingresar un texto. La página que procesa el formulario sólo debe mostrar el contrato con las modificaciones que hizo el operador.

  Ejemplo de contrato:

En la ciudad de [.....], se acuerda entre la Empresa [.....] representada por el Sr. [.....] en su carácter de Apoderado, con domicilio en la calle [.....] y el Sr. [.....], futuro empleado con domicilio en [.....], celebrar el presente contrato a Plazo Fijo, de acuerdo a la normativa vigente de los artículos 90,92,93,94, 95 y concordantes de la Ley de Contrato de Trabajo N° 20.744.

Problema 6

Hacer un formulario que simule una calculadora simple.

El mismo deberá tener:

- 2 cuadros de texto para ingresar los valores a calcular
- 4 ***radiobuttons*** con las opciones “sumar”, “restar”, “multiplicar” y “dividir”
- 1 botón “Calcular” que muestre el resultado de la operación