

Manual de PHP

Que es PHP?

PHP (acrónimo de "PHP: Hypertext Preprocessor") es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor.

Hay que entender primero como funciona la solicitud de páginas en un navegador para comenzar a programar en PHP.

Comunicación entre el cliente y el servidor sin PHP:

1. Tipeamos en la barra del navegador la dirección y el archivo a solicitar.
2. El web browser (navegador) envía el mensaje a través de Internet a la computadora, por ejemplo `www.lanacion.com/pagina1.htm` solicitando la página (archivo) `pagina1.htm`
3. El web server (servidor web, que puede ser el Apache, IIS, etc.) que es un programa que se ejecuta en la máquina `www.lanacion.com`, recibe el mensaje y lee el archivo solicitado desde el disco duro.
4. El servidor web envía el archivo solicitado por el navegador tal cual está en el disco duro.
5. El navegador muestra en pantalla el archivo que envió el servidor web.

Este proceso siempre es el mismo cuando hablamos de páginas estáticas (páginas que no cambian), cualquiera sea el cliente que solicita la página el contenido siempre será el mismo. La única forma que el contenido del archivo cambie es que el administrador de ese sitio web edite el contenido del archivo `pagina1.htm` y haga modificaciones.

Comunicación entre el cliente y el servidor con PHP:

1. Tipeamos en la barra del navegador la dirección y el archivo a solicitar.
2. El web browser (navegador) envía el mensaje a través de Internet a la computadora llamada `www.lanacion.com` solicitando la página (archivo) `pagina1.php`
3. El web server (servidor web, que puede ser el Apache, IIS, etc.), recibe el mensaje y al ver que la extension es "php" solicita al interprete de PHP (que es otro programa que se ejecuta en el servidor web) que le envíe el archivo.
4. El intérprete PHP lee desde el disco el archivo `pagina1.php`
5. El intérprete PHP ejecuta los comandos contenidos en el archivo y eventualmente se comunica con un gestor de base de datos (ejemplos de ellos pueden ser MySQL, Oracle, Informix, SQL Server, etc.)
6. Luego de ejecutar el programa contenido en el archivo envía éste al servidor web.
7. El servidor web envía la página al cliente que la había solicitado.
8. El navegador muestra en pantalla el archivo que envió el servidor web.

Para aquellas personas que conocen otros lenguajes de programación (C - Java - C# - Pascal etc.) la salida de los resultados normalmente es la pantalla, en PHP la salida es la página HTML (luego ésta viaja por internet y llega al cliente que la solicitó)

Programa "Hola Mundo" en PHP

Para agregar un programa PHP dentro de una página HTML debemos por un lado al crear el archivo definirlo con extensión php (a diferencia de las páginas estáticas que tienen extensión htm o html) y dentro del contenido de la página, encerrar el programa entre los símbolos `<?php [aqui el programa PHP] ?>`.

El comando de PHP para imprimir dentro de la página se llama echo. Nuestro programa "Hola Mundo" será entonces:

```
<html>
<head></head>
<body>
<?php
    echo "Hola Mundo";
?>
</body>
</html>
```

Es decir que la página que se generará al ejecutarse el programa será:

```
<html>
<head></head>
<body>
Hola Mundo
</body>
</html>
```

Como podemos ver, es muy poco útil este programita, ya que el resultado de la ejecución de este programa PHP será siempre el mismo, es decir mostrar el texto "Hola Mundo".

pagina1.php

```
<html>
<head> </head>
<body>
<?php
    echo "Hola Mundo";
    echo "<br>";
    echo "como estas?";
?>
</body>
</html>
```



Confeccione un programa que muestre una serie de mensajes en la página empleando el comando echo. Tenga en cuenta que cuando utiliza el comando echo el mensaje se debe encerrar entre comillas dobles (como veremos más adelante también podrá encerrarse entre simples comillas).

Toda instrucción finaliza con punto y coma.

Un programita más útil que "Hola Mundo"

Un problema sencillo que se nos puede presentar y que no se puede resolver empleando solo HTML es que una página esté disponible sólo los 10 primeros días del mes. Mostraremos un cartel que diga que el sitio se encuentra disponible si la fecha es menor o igual a 10, en caso contrario mostraremos un mensaje de sitio fuera de servicio.

Para obtener la fecha del servidor web debemos llamar a la función date y requerir sólo el día:

```
$dia=date("d");
```

A las variables en PHP se les antecede el caracter \$. Si a la función date le pasamos el string "d" retornará sólo el día (si queremos la fecha completa: **\$fecha=date("Y:m:d")**)

Para verificar si la variable \$dia es menor o igual a 10, debemos emplear la instrucción if, similar a otros lenguajes.

Entonces la página con el programa queda de la siguiente forma:

```
<html>
<head></head>
<body>
<?php
    $dia=date("d");
    if ($dia<=10)
    {
        echo "sitio activo";
    }
    else
    {
        echo "sitio fuera de servicio";
    }
?>
</body>
</html>
```

Los nombres de variables son sensibles a mayúsculas y minúsculas, por lo que si la escribimos en minúscula inicialmente debemos respetar en el resto del programa. En cambio las instrucciones del lenguaje PHP no son sensibles por lo que si deseamos escribir IF o if, las dos formas estarán bien. Los que venimos de otros lenguajes como C, C++, Java tenemos por costumbre escribir las palabras claves en minúsculas, pero esto es solo por costumbre.

La condición del if debe ir obligatoriamente entre paréntesis. Los operadores relacionales disponibles son:

```
> Mayor
>= Mayor o igual
< Menor
<= Menor o igual
== Igual
!= Distinto
```

Si la condición se verifica verdadera se ejecuta el primer bloque encerrado entre llaves, en caso de verificarse falsa la condición se ejecuta el bloque entre llaves que le sigue al else.

pagina1.php

```
<html>
<head></head>
<body>

<?php
$dia=date("d");
if ($dia<=10)
{
    echo "sitio activo";
}
else
{
    echo "sitio fuera de servicio";
}
?>
```



Sabiendo que la función **rand** nos retorna un valor aleatorio entre un rango de dos enteros:

```
$num=rand(1,100);
```

En la variable **\$num** se almacena un valor entero que la computadora genera en forma aleatoria entre 1 y 100.

Hacer un programa que lo muestre por pantalla al valor generado. Mostrar además si es menor o igual a 50 o si es mayor.

Para imprimir el contenido de una variable también utilizamos el comando echo:

```
echo $num;
```

Tipos de variables

Los nombres de variables comienzan con el signo \$ y son sensibles a mayúsculas y minúsculas (no así las palabras claves del lenguaje).

En PHP no es necesario definir el tipo antes de utilizarla, las mismas se crean en el momento de emplearlas. Las variables se declaran cuando se le asigna un valor, por ejemplo:

```
$dia = 24; //Se declara una variable de tipo integer.
```

```
$sueldo = 758.43; //Se declara una variable de tipo double.
```

```
$nombre = "juan"; //Se declara una variable de tipo string.
```

```
$exite = true; //Se declara una variable boolean.
```

También podemos hacer notar que para disponer comentarios de línea debemos utilizar dos caracteres //

Para la impresión de variables utilizaremos inicialmente el comando **echo**. Un programa completo que inicializa y muestra el contenido de cuatro variables de distinto tipo es:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$dia = 24; //Se declara una variable de tipo integer.
$sueldo = 758.43; //Se declara una variable de tipo double.
$nombre = "juan"; //Se declara una variable de tipo string.
$exite = true; //Se declara una variable boolean.
echo "Variable entera:";
echo $dia;
echo "<br>";
echo "Variable double:";
echo $sueldo;
echo "<br>";
echo "Variable string:";
echo $nombre;
echo "<br>";
echo "Variable boolean:";
echo $exite;
?>
</body>
</html>
```

Hemos utilizado un comando echo para mostrar los mensajes, otro el contenido de variables y finalmente otro para imprimir marcas HTML. Este proceso puede acortarse un poco pero para que sea más claro inicialmente tomaremos el camino largo de hacer la impresión de un dato con cada comando echo.

pagina1.php

```
<html>
<head></head>
<body>
<?php
$dia1=15;
$dia2=30;
echo "El valor de la primer variable es:";
echo $dia1;
echo "<br>";
echo "El valor de la segunda variable es:";
echo $dia2;

?>
</body>
</html>
```



Definir una variable de cada tipo: integer, double, string y boolean. Luego imprimirlas en la página, una por línea.

Variables de tipo string

Una variable de este tipo puede almacenar una serie de caracteres.

```
$cadena1="Hola";  
$cadena2="Mundo";  
echo $cadena1." ".$cadena2;
```

Para concatenar string empleamos el operador .

Tengamos en cuenta que el comando echo de más arriba lo podemos hacer más largo de la siguiente forma:

```
echo $cadena1;  
echo " ";  
echo $cadena2;
```

A medida que uno haga ejercicios podremos resumir en un solo comando echo la salida de múltiples variables.

Cuando una cadena encerrada entre comillas dobles contiene una variable en su interior, ésta se trata como tal, por lo tanto se utilizará su contenido para el almacenamiento.

```
$dia=10;  
$fecha="Hoy es $dia";  
echo $fecha;
```

En pantalla se muestra: Hoy es 10

Es decir, en la cadena, se sustituye el nombre de la variable \$dia, con el contenido de la misma.

Una cadena se puede definir con las comillas simples (pero es importante tener en cuenta que no se sustituyen las variables si empleamos comillas simples):

```
$nombre='juan carlos';
```

pagina1.php

```
<html>  
<head>  
<title>Problema</title>  
</head>  
<body>  
  
<?php  
$cadena1="diego";
```

```
$cadena2="juan";  
$cadena3="ana";  
$todo=$cadena1.$cadena2.$cadena3."<br>";  
echo $todo;  
$edad1=24;  
echo $cadena1." tiene $edad1 de edad";  
  
?>  
</body>  
</html>
```

```
diegojuanana  
diego tiene 24 de edad
```



Definir tres variables enteras. Luego definir un string que incorpore dichas variables y las sustituya en tiempo de ejecución.

Recordar que una variable se sustituye cuando el string está encerrado por comillas dobles:

```
$precio=90;  
echo "La computadora tiene un precio de $precio";
```

Estructura condicional (if)

Cuando se pretende que el programa, una vez llegado a un cierto punto, tome un camino concreto en determinados casos y otro diferente si las condiciones de ejecución difieren, se utiliza el conjunto de instrucciones:

if, else y elseif. La estructura base de este tipo de instrucciones es la siguiente:

```
if (Condición)  
{  
    Instrucción 1;  
    Instrucción 2;  
}  
else  
{  
    Instrucción A;  
    Instrucción B;  
}
```

Cuando la ejecución llega al punto donde se encuentra la instrucción if, el programa verificará el cumplimiento o no de la condición. Si la condición es verdadera se ejecutarán las instrucciones 1 y 2, de lo contrario, se ejecutarán las instrucciones A y B. En los casos en que las condiciones sean varias, se pueden utilizar los if de un modo denominado anidado o anillado, como se indica de la manera siguiente:

```
if (Condicion 1)  
{
```

```
Instrucción 1;
Instrucción 2;
}
else
{
    if (Condicion 2)
    {
        Instrucción A;
        Instrucción B;
    }
    else
    {
        Instrucción X;
        Instrucción Z;
    }
}
```

De este modo se pueden introducir tantas condiciones como se quiera dentro de la condición principal. Una variante de este sistema es utilizando la sentencia `elseif`, que permite en una sola línea introducir una condición adicional:

```
if (Condicion 1)
{
    Instrucción 1;
    Instrucción 2;
}
elseif (Condicion 2)
{
    Instrucción A;
    Instrucción B;
}
else
{
    Instrucción X;
    Instrucción Z;
}
```

Para las condiciones tener en cuenta que disponemos de los siguientes operadores:

`==` para ver si una variable es igual a otra.

`!=` distinto.

`>=` mayor o igual.

`>` mayor.

`<=` menor o igual

`<` menor

La mejor forma de entender esta estructura condicional es por medio de ejemplos. El primero que nos plantearemos es generar un valor aleatorio (es decir lo elige la máquina al azar, como extraer una bolilla de un bolillero de lotería) comprendido entre 1 y 10. Luego mostraremos un mensaje si es menor o igual a 5 o si es mayor a 5.

El programa completo es:

```
<html>
<head>
<title>Problema</title>
</head>
```



```
<body>
<?php
$valor=rand(1,10);
echo "El valor sorteado es $valor<br>";
if ($valor<=5)
{
    echo "Es menor o igual a 5";
}
else
{
    echo "Es mayor a 5";
}
?>
</body>
</html>
```

Es importante recordar que siempre la condición del if debe ir entre paréntesis. Si la condición del if se verifica verdadera (es decir el número sorteado es menor o igual a 5) ejecuta el primer bloque que se encuentra entre llaves. En caso de verificarse falsa la condición del if se ejecuta el bloque entre llaves que se encuentra después del else.

El valor aleatorio lo generamos llamando a la función random pasándole el valor mínimo y máximo:

```
$valor=rand(1,10);
```

Imprimimos el valor generado a los efectos de controlar el resultado:

```
echo "El valor sorteado es $valor<br>";
```

En el primer problema tenemos solo dos caminos posibles, el valor es menor o igual a cinco o es mayor. En un segundo ejemplo mostraremos como disponer una estructura condicional if anidada.

El problema es el siguiente: Generar un valor aleatorio entre 1 y 100. Luego mostrar si tiene 1,2 o 3 dígitos.

Como podemos observar estamos en presencia de un problema que tiene tres caminos posibles. El valor puede tener 1 dígito, 2 dígitos o 3 dígitos. Si pensamos un poco podremos identificar que para que tenga un dígito debe generarse un valor entre 1 y 9, para que tenga dos dígitos deberá estar comprendido entre 10 y 90, y finalmente para tener 3 dígitos deberá ser el valor 100.

La página con el código respectivo es:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$valor=rand(1,100);
echo "El valor sorteado es $valor<br>";
if ($valor<=9)
{
    echo "Tiene un dígito";
}
else
```

```
{
    if ($valor<100)
    {
        echo "Tiene 2 dígitos";
    }
    else
    {
        echo "Tiene 3 dígitos";
    }
}
?>
</body>
</html>
```

Es importante notar como la segunda estructura if se encuentra contenida entre las llaves del else del primer if. Es decir que si el valor aleatorio es menor o igual a 9 se ejecuta el bloque del verdadero del primer if y no se ejecuta por lo tanto el if anidado en el else. Por el contrario si la condición del primer if se verifica false se ejecuta el bloque del else del primer if, la misma contiene una estructura if con sus bloques del verdadero y false.

A modo ilustrativo el mismo ejemplo resuelto con la estructura if/elseif será:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$valor=100;//rand(1,100);
echo "El valor sorteado es $valor<br>";
if ($valor<=9)
{
    echo "Tiene un dígito";
}
elseif ($valor<100)
{
    echo "Tiene 2 dígitos";
}
else
{
    echo "Tiene 3 dígitos";
}
?>
</body>
</html>
```

Si uno tiene mucha práctica con otros lenguajes de programación esta estructura de if anidados le será mas adecuada, en caso que haya empezado a programar hace poco será conveniente que elija y practique la primera forma.

pagina1.php

```
html>
<head>
<title>Problema</title>
</head>
<body>

<?php
```

```
//Generar un valor aleatorio entre 1 y 100. Luego mostrar  
//si tiene 1,2 o 3 dígitos.
```

```
$valor=rand(1,100);  
echo "El valor sorteado es $valor<br>";  
if ($valor<=9)  
{  
    echo "Tiene un dígito";  
}  
else  
{  
    if ($valor<100)  
    {  
        echo "Tiene 2 dígitos";  
    }  
    else  
    {  
        echo "Tiene 3 dígitos";  
    }  
}  
?>
```

```
</body>  
</html>
```

El valor sorteado es 1
Tiene un dígito

Generar un valor aleatorio entre 1 y 5. Luego imprimir en castellano el número (Ej. si se genera el 3 luego mostrar en la página el string "tres"). Para ver si una variable es igual a cierto valor debemos plantear una condición similar a:



```
if ($valor==3)  
{  
    //algoritmo  
}
```

Estructuras repetitivas (for - while - do/while)

Las estructuras repetitivas son similares al lenguaje C.
Estructura for:

```
for([Inicialización de la variable];[Condición];[Incremento o decremento de la variable])  
{  
    [Instrucciones];  
}
```

El primer ejemplo que haremos es mostrar en la página los números del 1 al 100:

```
html>  
<head>  
<title>Problema</title>  
</head>
```

```
<body>
<?php
for ($f=1; $f<=100; $f++)
{
    echo $f;
    echo "<br>";
}
?>
</body>
</html>
```

Quien no ha visto la estructura for en otro lenguaje pasamos a explicarla:

```
for ($f=1; $f<=100; $f++)
{
    echo $f;
    echo "<br>";
}
```

El primer argumento del for es la inicialización de una variable, en este caso se inicializa la variable \$f con el valor 1. Este primer argumento del for se ejecuta solo una vez. Luego se ejecuta el segundo argumento que es la condición. Si la misma se verifica como verdadera se ejecuta todo el bloque comprendido entre las llaves de apertura y cerrado. Luego de haberse ejecutado el bloque repetitivo se ejecuta el tercer argumento del for que es el incremento de la variable, en este caso \$f++ incrementa el contenido de la variable \$f en 1 (también podemos poner en lugar de \$f++ la asignación \$f=\$f+1).

Luego del incremento de la variable se ejecuta nuevamente la condición del for (segundo argumento), de validarse nuevamente verdadero pasa a ejecutar el bloque repetitivo. Este ciclo se repite hasta que la condición del for se verifica false.

La segunda estructura repetitiva es:

```
while (condición)
{
    [Instrucciones];
}
```

Esta estructura está en casi todos los lenguajes. El bloque se repite mientras la condición del while sea verdadera.

La condición del while se verifica antes de ingresar al bloque a repetir. Si la misma se verifica falsa la primera vez no se ejecutará el bloque.

Veamos un ejemplo: Generar un valor aleatorio entre 1 y 100, luego imprimir en la página desde 1 hasta el valor generado (de uno en uno):

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$valor=rand(1,100);
$inicio=1;
while ($inicio<=$valor)
```

```
{
    echo $inicio;
    echo "<br>";
    $inicio++;
}
?>
</body>
</html>
```

La variable \$inicio tiene el valor 1 antes de ingresar al while. Cada vez que se ejecuta una vez el bloque del while se incrementa \$inicio en uno. Cuando \$inicio supere la variable aleatoria \$valor finalizará la estructura repetitiva y pasará a ejecutarse la instrucción inmediatamente siguiente a la llave de cerrado.

Es importante notar que luego de la condición del while NO disponemos PUNTO y COMA.

Por último tenemos en el lenguaje una estructura repetitiva similar al while llamada do/while, donde la condición se verifica luego de ejecutarse el bloque repetitivo.

```
do
{
    [Instrucciones];
} while (condición);
```

Queda como tarea intentar hacer una página empleando esta estructura. Tener en cuenta que al final de la línea del while SI LLEVA punto y coma.

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>

<?php
//Mostramos los números de los días del 1 a la fecha actual;
$día=date("d");
$inicio=1;
while($inicio<=$día)
{
    echo $inicio."<br>";
    $inicio++;
}
?>
</body>
</html>
```

1
2
3
4
5



Mostrar la tabla de multiplicar del 2. Emplear el for, luego el while y por último el do/while.

La estructura for permite incrementar una variable de 2 en 2:

```
for ($f=2; $f<=20; $f=$f+2)
```

Envío de datos de un FORMULARIO (controles text y submit)

El proceso para el manejo de FORMULARIOS requiere generalmente dos páginas, una que implementa el formulario y otra que procesa los datos cargados en el formulario. La estructura mínima de un formulario es la siguiente: para la entrada de un nombre de persona, un objeto text y un botón para el envío del dato al servidor:

```
<html>
  <head>
    <title>Formulario de entrada del dato</title>
  </head>
  <body>
    <form method="post" action="pagina2.php">
      Ingrese su nombre:
      <input type="text" name="nombre">
      <br>
      <input type="submit" value="confirmar">
    </form>
  </body>
</html>
```

Esta página está completamente codificada en HTML, es decir un formulario contiene marcas HTML puras.

La marca `<form>` y `</form>` nos permite definir un formulario en la página. La marca FORM tiene dos propiedades que debemos inicializar obligatoriamente: `action` y `method`. La propiedad `action` indica el nombre del archivo que recibirá los datos ingresados por el operador en el formulario y que serán enviados al servidor cuando se presione el botón (submit). La propiedad `method` indica como se organizan esos datos para enviarlos al servidor, pudiendo ser mediante los métodos `post` o `get` (normalmente los datos de un formulario se envían mediante el método `post`).

Para crear un cuadro de texto para el ingreso del nombre debemos definir un objeto de tipo "text" y darle un nombre:

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

La propiedad `type` nos permite definir el tipo de control y con la propiedad `name` indicamos el nombre del control.

Por último todo formulario tiene un botón de tipo submit:

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

También utilizamos la marca `input` pero en la propiedad `type` indicamos que se trata de un botón de envío de datos. En la propiedad `value` indicamos el texto que queremos que aparezca en el botón.

Ahora necesitamos una página con un pequeño programa en PHP que procese los datos ingresados en el formulario:

```
<html>
  <head>
    <title>Captura de datos del form</title>
```

```
</head>
<body>
<?php
    echo "El nombre ingresado es:";
    echo $_REQUEST['nombre'];
    ?>
</body>
</html>
```

Para acceder al dato en PHP se cuenta con 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 problema sólo mostramos por pantalla el valor ingresado en la página anterior:

```
echo $_REQUEST['nombre'];
```

pagina1.php

```
<html>
<head>
<title>Formulario de entrada del dato</title>
</head>
<body>

<form method="post" action="pagina2.php">
Ingrese su nombre:
<input type="text" name="nombre">
<br>
<input type="submit" value="confirmar">
</form>

</body>
</html>
```

pagina2.php

```
<html>
<head>
<title>Captura de datos del form</title>
</head>
<body>

<?php
    echo "El nombre ingresado es:";
    echo $_REQUEST['nombre'];
    ?>

</body>
</html>
```

Ingrese su nombre:

El nombre ingresado es:Paul

Confeccionar un formulario que solicite la carga de un nombre de persona y su edad, luego mostrar en otra página si es mayor de edad (si la edad es mayor o igual a 18)

FORMULARIO (control radio)

Para analizar este control dispondremos un ejemplo: Implementar un formulario que solicite la carga de dos enteros, uno en cada text. Disponer dos controles de tipo radio que nos permitan seleccionar si queremos sumar o restar los dos valores ingresados:

```
<html>
  <head>
    <title>Problema</title>
  </head>
  <body>
    <form action="pagina2.php"
      method="post">
      Ingrese primer valor:
      <input type="text" name="valor1">
      <br>
      Ingrese segundo valor:
      <input type="text" name="valor2">
      <br>
      <input type="radio" name="radio1" value="suma">sumar
      <br>
      <input type="radio" name="radio1" value="resta">restar
      <br>
      <input type="submit" name="operar">
    </form>
  </body>
</html>
```

Es importante notar que se trata nuevamente de un archivo HTML puro, que no tiene código PHP.

La entrada de los dos números se efectúa en dos controles
<input type="text" name="valor1"> <input type="text" name="valor2"> Es importante notar que cada text tiene un name DIFERENTE.

Para seleccionar el tipo de operación a efectuar disponemos dos controles de tipo radio:

<input type="radio" name="radio1" value="suma">sumar

<input type="radio" name="radio1" value="resta">restar Es

importante notar que los dos controles tienen el MISMO nombre. Esto es necesario para que el navegador sepa que los dos controles están relacionados (recordar que cuando uno selecciona un radio se debe deseleccionar el otro)
Desde la otra página accederemos al value del control seleccionado.

Por último disponemos un control de tipo submit para el envío de los datos del formulario. El código de la página que procesa el formulario, llamada: "pagina2.php" (la que indicamos en la marca FORM del formulario) es:

```
<html>
  <head>
    <title>Problema</title>
```



```
</head>
<body>
<?php
if ($_REQUEST['radio1']=="suma")
{
    $suma=$_REQUEST['valor1'] + $_REQUEST['valor2'];
    echo "La suma es:".$suma;
}
else
{
    if ($_REQUEST['radio1']=="resta")
    {
        $resta=$_REQUEST['valor1'] - $_REQUEST['valor2'];
        echo "La resta es:".$resta;
    }
}
?>
</body>
</html>
```

El vector asociativo `$_REQUEST` tiene tres componentes: `$_REQUEST['radio1']`, `$_REQUEST['valor1']` y `$_REQUEST['valor2']`. En la componente `$_REQUEST['radio1']` almacena la cadena "suma" o "resta" según cual se seleccionó en el formulario.

Con dos `if` verificamos cual operación está seleccionada y procedemos a efectuarla:

```
if ($_REQUEST['radio1']=="suma")
{
    $suma=$_REQUEST['valor1'] + $_REQUEST['valor2'];
    echo "La suma es:".$suma;
}
.
```

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>

<form action="pagina2.php" method="post">
Ingrese primer valor:
<input type="text" name="valor1">
<br>
Ingrese segundo valor:
<input type="text" name="valor2">
<br>
<input type="radio" name="radio1" value="suma">sumar
<br>
<input type="radio" name="radio1" value="resta">restar
<br>
<input type="submit" name="operar">
</form>

</body>
</html>
```

pagina2.php

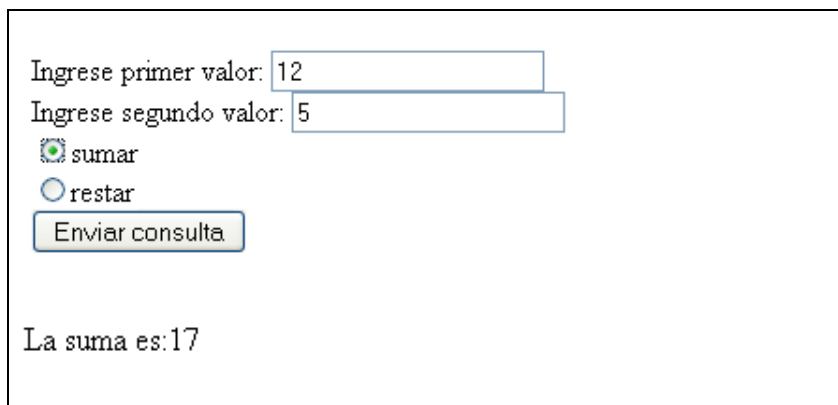
```
<html>
<head>
```

```
<title>Problema</title>
</head>
<body>

<?php
if ($_REQUEST['radio1']=="suma")
{
    $suma=$_REQUEST['valor1'] + $_REQUEST['valor2'];
    echo "La suma es:".$suma;
}
else
{
    if ($_REQUEST['radio1']=="resta")
    {
        $resta=$_REQUEST['valor1'] - $_REQUEST['valor2'];
        echo "La resta es:".$resta;
    }
}

?>

</body>
</html>
```



The screenshot shows a web form with the following elements:

- Two text input fields for "Ingrese primer valor:" and "Ingrese segundo valor:". The first field contains the value "12" and the second contains "5".
- Two radio buttons: "sumar" (selected) and "restar".
- A button labeled "Enviar consulta".
- A text output area at the bottom that displays "La suma es:17".

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: 1-no tiene estudios, 2-estudios primarios, 3-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.

FORMULARIO (control checkbox)

Para analizar este control utilizaremos prácticamente el mismo ejemplo que con el objeto radio:

Implementar un formulario que solicite la carga de dos enteros, uno en cada text. Disponer

dos controles de tipo checkbox que nos permitan seleccionar si queremos sumar y/o restar los valores ingresados.

El formulario html tiene el siguiente código:

```
<head>
  <title>Problema</title>
</head>
<body>
<form action="pagina2.php"
  method="post">
  Ingrese primer valor:
  <input type="text" name="valor1">
  <br>
  Ingrese segundo valor:
  <input type="text" name="valor2">
  <br>
  <input type="checkbox" name="check1">sumar
  <br>
  <input type="checkbox" name="check2">restar
  <br>
  <input type="submit" name="operar">
</form>
</body>
</html>
```

Lo nuevo en este problema son los dos controles de tipo checkbox:

```
<input type="checkbox" name="check1">sumar
<br>
<input type="checkbox" name="check2">restar
<br>
```

Es importante notar que cada checkbox tiene un nombre distinto.

Ahora veamos el código de la página que procesa el formulario:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
  if (isset($_REQUEST['check1']))
  {
    $suma=$_REQUEST['valor1'] + $_REQUEST['valor2'];
    echo "La suma es:".$suma."<br>";
  }
  if (isset($_REQUEST['check2']))
  {
    $resta=$_REQUEST['valor1'] - $_REQUEST['valor2'];
    echo "La resta es:".$resta;
  }
  ?>
</body>
</html>
```

Si el checkbox no está seleccionado en el formulario no se crea una entrada en el vector asociativo `$_REQUEST`, para saber si existe una determinada componente en un vector se emplea la función `isset`, si retorna `true` significa que existe y por lo tanto el checkbox está seleccionado.

Disponemos dos if a la misma altura ya que los dos controles de tipo checkbox podrían estar seleccionados.

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>

<form action="pagina2.php" method="post">
Ingrese primer valor:
<input type="text" name="valor1">
<br>
Ingrese segundo valor:
<input type="text" name="valor2">
<br>
<input type="checkbox" name="check1">sumar
<br>
<input type="checkbox" name="check2">restar
<br>
<input type="submit" name="operar">
</form>

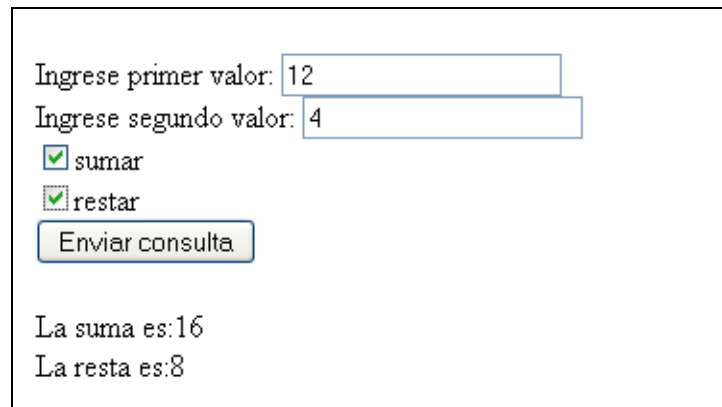
</body>
</html>
```

pagina2.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>

<?php
if (isset($_REQUEST['check1']))
{
    $suma=$_REQUEST['valor1'] + $_REQUEST['valor2'];
    echo "La suma es:". $suma."<br>";
}
if (isset($_REQUEST['check2']))
{
    $resta=$_REQUEST['valor1'] - $_REQUEST['valor2'];
    echo "La resta es:". $resta;
}
?>

</body>
</html>
```



The screenshot shows a web form with the following elements:

- Input field for "Ingrese primer valor:" containing the number 12.
- Input field for "Ingrese segundo valor:" containing the number 4.
- Two radio buttons for selecting an operation: "sumar" (checked) and "restar".
- A button labeled "Enviar consulta".
- Output text: "La suma es:16" and "La resta es:8".

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.

FORMULARIO (control select)

Implementar un formulario que solicite la carga de dos enteros, uno en cada "text". Disponer un control de tipo select que nos permita seleccionar si queremos sumar o restar los dos valores ingresados:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php"
method="post">
  Ingrese primer valor:
  <input type="text" name="valor1">
  <br>
  Ingrese segundo valor:
  <input type="text" name="valor2">
  <br>
  <select name="operacion">
    <option value="suma">sumar</option>
    <option value="resta">restar</option>
  </select>
  <br>
  <input type="submit" name="operar">
</form>
</body>
</html>
```

Lo nuevo que aparece en este formulario es el control de tipo select.

```
<select name="operacion">
  <option value="suma">sumar</option>
  <option value="resta">restar</option>
</select>
```

Cada opción tiene un valor. El seleccionado es el que se enviará a la página que procesa el formulario. El texto que aparece dentro del control es el que disponemos entre las marcas option.

Ahora la página que captura los datos ingresados en el formulario es:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
    if ($_REQUEST['operacion']==suma)
    {
        $suma=$_REQUEST['valor1'] + $_REQUEST['valor2'];
        echo "La suma es:".$suma;
    }
    else
    {
        if ($_REQUEST['operacion']==resta)
        {
            $resta=$_REQUEST['valor1'] - $_REQUEST['valor2'];
            echo "La resta es:".$resta;
        }
    }
?>
</body>
</html>
```

El vector asociativo `$_REQUEST` almacena en la componente del control select el valor de la opción seleccionada.

Con una serie de if verificamos el valor seleccionado:

```
if ($_REQUEST['operacion']==suma)
...

```

Sólo se puede seleccionar un elemento de un control select (más adelante veremos como seleccionar varios elementos en forma simultánea)

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>

<form action="pagina2.php" method="post">
Ingrese primer valor:
<input type="text" name="valor1">
<br>
Ingrese segundo valor:
<input type="text" name="valor2">
<br>
<select name="operacion">
<option value="suma">sumar</option>
<option value="resta">restar</option>
</select>
<br>
<input type="submit" name="operar">
</form>
```

```
</body>
</html>
```

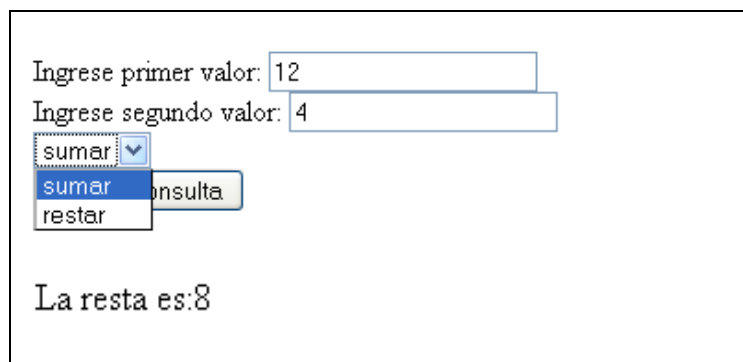
pagina2.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>

<?php
if ($_REQUEST['operacion']==suma)
{
    $suma=$_REQUEST['valor1'] + $_REQUEST['valor2'];
    echo "La suma es:".$suma;
}
else
{
    if ($_REQUEST['operacion']==resta)
    {
        $resta=$_REQUEST['valor1'] - $_REQUEST['valor2'];
        echo "La resta es:".$resta;
    }
}

?>

</body>
</html>
```



Confeccionar un formulario que solicite el ingreso del nombre de una persona y un combo de selección (en este último permitir la selección de los ingresos anuales 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).

FORMULARIO (control textarea)

El control "textarea" se diferencia del "text" en que permite el ingreso de muchas líneas. Lo probaremos implementando un problema que permita ingresar el curriculum de una persona.

```
<html>
  <head>
    <title>Problema</title>
  </head>
  <body>
    <form action="pagina2.php" method="post">
      Ingrese nombre:<input type="text" name="nombre"><br>
      Ingrese su curriculum:<br>
      <textarea name="curriculum"></textarea>
      <br>
      <input type="submit" value="Confirmar">
    </form>
  </body>
</html>
```

La sintaxis de este control es bastante diferente a la del control text:

```
<textarea name="curriculum"></textarea>
```

Si queremos que aparezca inicializado con texto debemos disponerlo en:

```
<textarea name="curriculum">Hola Mundo</textarea>
```

La página PHP que procesa los dos datos ingresados en el formulario es:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
  echo "El nombre ingresado:".$_REQUEST['nombre'];
  echo "<br>";
  echo "El curriculum:".$_REQUEST['curriculum'];
  ?>
</body>
</html>
```

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
  <form action="pagina2.php" method="post">
    Ingrese nombre:<input type="text" name="nombre"><br>
    Ingrese su curriculum:<br>
    <textarea name="curriculum"></textarea>
    <br>
    <input type="submit" value="Confirmar">
  </form>

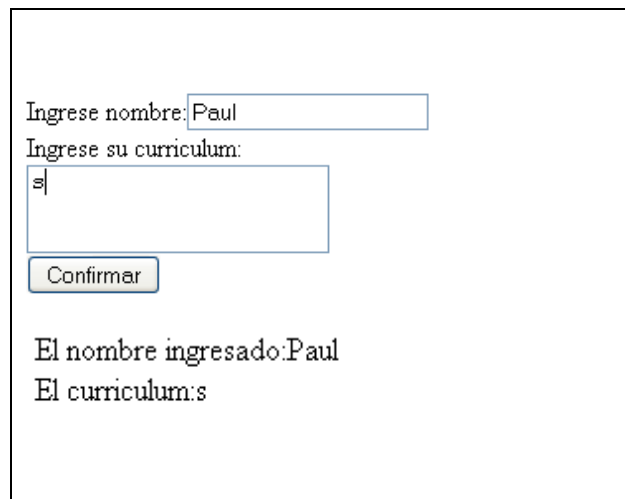
  </body>
</html>
```

pagina2.php


```
<html>
<head>
<title>Problema</title>
</head>
<body>

<?php
echo "El nombre ingresado:".$_REQUEST['nombre'];
echo "<br>";
echo "El curriculum:".$_REQUEST['curriculum'];
?>

</body>
</html>
```

The image shows a web browser window. At the top, there is a form with two input fields. The first field is labeled 'Ingrese nombre:' and contains the text 'Paul'. The second field is labeled 'Ingrese su curriculum:' and contains the text 's'. Below the second field is a button labeled 'Confirmar'. Below the form, the output of the PHP script is displayed: 'El nombre ingresado:Paul' and 'El curriculum:s'.

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. Ej. de un contrato puede ser: 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.

Vectores (tradicionales)

Un Array es una colección de valores. Los array pueden ser unidimensionales (vectores), bidimensionales (matrices) y multidimensionales (más de dos dimensiones) Los arrays se utilizan ampliamente en el lenguaje PHP. Se utiliza el delimitador [] para acceder a los diferentes elementos del vector.

Se lo puede crear al vuelo, sin tener que declararlo:

```
$dias[0]=31;  
$dias[1]=28;
```

Luego de estas dos líneas, tenemos creado un vector de dos elementos, a los cuales accedemos por un subíndice que comienza a numerarse desde cero.

```
echo $dias[0]; //31  
echo $dias[1]; //28
```

El vector, como podemos ver, puede ir creciendo en forma dinámica, es decir que si ahora hacemos:

```
$dias[2]=31;
```

el vector tiene 3 componentes.

También podemos obviar el subíndice cuando asignamos los valores:

```
$dias[]=31;  
$dias[]=28;  
$dias[]=31;
```

Automáticamente comienza a numerarse desde cero.

Si necesitamos conocer el tamaño del vector en cualquier momento podemos llamar a la función count.

```
echo count($dias); //3
```

Si queremos imprimir todos los elementos en la página podemos hacer:

```
<?php  
$nombres[]="juan";  
$nombres[]="pedro";  
$nombres[]="ana";  
for($f=0;$f<count($nombres);$f++)  
{  
    echo $nombres[$f];  
    echo "<br>";  
}  
?>
```

La función sizeof(<nombre del vector>) es equivalente a count

Otra forma de inicializar un vector es definirlo e inicializarlo simultáneamente:

```
$edades=array("menores", "jovenes", "adultos");
```

Estamos definiendo el vector edades con tres componentes, numeradas automáticamente de cero a dos.

pagina1.php

```
<html>  
<head>  
<title>Problema</title>  
</head>  
<body>  
  
<?php  
$nombres[]="juan";  
$nombres[]="pedro";  
$nombres[]="ana";  
for($f=0;$f<count($nombres);$f++)  
{  
    echo $nombres[$f];  
}
```

```
        echo "<br>";
    }
?>
```

```
</body>
</html>
```

```
juan
pedro
ana
```

Definir un vector con los nombres de los días de la semana. Luego imprimir el primero y el último elemento del vector.

Creación de un archivo de texto.

Una actividad fundamental es poder registrar información en el servidor (no como hemos estado haciendo hasta el momento generando sólo una página con los datos cargados). Para la registración de datos en el servidor disponemos de dos herramientas que se complementan en muchos casos (archivos de texto y bases de datos). En este apartado veremos como crear un archivo de texto y añadir datos al mismo.

Lo presentaremos al tema resolviendo un problema: Implementación de un libro de visitas. Para resolver este problema plantearemos dos páginas, un formulario para realizar la carga del nombre del visitante y sus comentarios (disponemos un objeto de tipo "text" y otro de tipo "textarea"):

```
<html>
<head>
<title>Problema</title>
</head>
<body> <form action="pagina2.php" method="post">
Ingrese su nombre:
<input type="text" name="nombre">
<br>
Comentarios:
<br>
<textarea name="comentarios" rows="10" cols="40">
</textarea>
<br>
<input type="submit" value="Registrar">
</form>
</body>
</html>
```

Este formulario es similar a los planteados en problemas anteriores, sólo le hemos agregado al control textarea, las propiedades rows y cols que dimensionan el mismo en la pantalla:

```
<textarea name="comentarios" rows="10" cols="40">
</textarea>
```

Veamos ahora la página (pagina2.php) que graba los datos cargados en el formulario en un archivo:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
    $ar=fopen("datos.txt","a") or
        die("Problemas en la creacion");
    fputs($ar,$_REQUEST['nombre']);
    fputs($ar,"\n");
    fputs($ar,$_REQUEST['comentarios']);
    fputs($ar,"\n");
    fputs($ar,"-----");
    fputs($ar,"\n");
    fclose($ar);
    echo "Los datos se cargaron correctamente.";
?>
</body>
</html>
```

En primer lugar creamos o abrimos el archivo de texto "datos.txt". El segundo parámetro de la función fopen indica la forma de apertura de archivo "a" (lo crea o si ya existe el archivo lo abre para añadir datos al final), "w" (crea el archivo de texto, si existe borra su contenido) y la última forma de apertura del archivo es "r" (abre el archivo para su lectura). Como en este problema nos interesa que el archivo vaya creciendo con los datos que aportan los visitantes al sitio lo abrimos para añadir, parámetro "a". La función retorna una referencia al archivo, la almacenamos en una variable. Si el archivo no se puede abrir, se ejecuta la instrucción que se encuentra luego del operador "or" en nuestro caso llamamos a la función die que finaliza la ejecución del programita PHP mostrando como mensaje el texto que le pasamos a dicha función.

```
$ar=fopen("datos.txt","a") or
    die("Problemas en la creacion");
```

Para la grabación de datos utilizamos la función fputs que tiene dos parámetros: la referencia al archivo donde grabamos y el string a grabar.

```
fputs($ar,$_REQUEST['nombre']);
fputs($ar,"\n");
```

Para el salto de línea en el archivo de texto, usamos los caracteres \n. De esta forma cuando leamos el archivo de texto lo haremos línea a línea. Cuando dejamos de trabajar con el archivo llamamos a la función fclose.

Hay que tener muy presente que el archivo se almacena en el servidor y no en la máquina de la persona que está navegando. Es decir, no vaya al explorador de archivos para ver donde se almacenó "datos.txt", tenga en cuenta que está en la máquina donde se ejecutó el script de PHP. Luego veremos como leer el contenido del archivo y mostrarlo en otra página del sitio.

pagina1.php

```
<html>
```

```
<head>
<title>Problema</title>
</head>
<body>

<form action="pagina2.php" method="post">
Ingrese su nombre:
<input type="text" name="nombre">
<br>
Comentarios:
<br>
<textarea name="comentarios" rows="10" cols="40">
</textarea>
<br>
<input type="submit" value="Registrar">
</form>

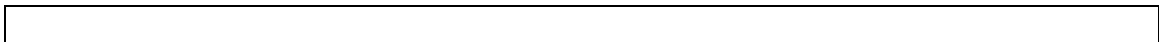
</body>
</html>
```

pagina2.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>

<?php
$ar=fopen("datos.txt","a") or
    die("Problemas en la creacion");
fputs($ar,$_REQUEST['nombre']);
fputs($ar,"\n");
fputs($ar,$_REQUEST['comentarios']);
fputs($ar,"\n");
fputs($ar,"-----");
fputs($ar,"\n");
fclose($ar);
echo "Los datos se cargaron correctamente.";
?>

</body>
</html>
```



Ingrese su nombre:

Comentarios:

p|

Los datos se cargaron correctamente.

Confeccionar un programita en PHP que permita hacer el pedido de pizzas via internet.
El formulario debe ser:

Nombre:[.....]

Direccion:[.....]

Jamon y queso:[x]

Cantidad[...]

Napolitana:[x]

Cantidad[...]

Muzzarella:[x]

Cantidad[...]

[Confirmar]

Para el ingreso del nombre, dirección y cantidad de pizzas de cada tipo disponer objetos de la clase "text".
Disponer tres objetos de tipo "check" para seleccionar los tipos de pizzas.
Por último disponer un botón para el envío de datos: "submit".
Grabar en un archivo de texto cada pedido, separados por una línea de puntos (obligatoriamente dar el nombre del archivo de texto como "datos.txt", esto es para que no se llene mi disco duro de archivos, es decir de ahora en más siempre que cree un archivo de texto debe llamarse datos.txt)

Lectura de un archivo de texto.

Para la lectura de un archivo de texto contamos con la función fgets. Además debemos habrir el archivo para lectura.

Para mostrar por pantalla el contenido del archivo "datos.txt" creado en el punto anterior tenemos el siguiente programa:

```
<html>
```

```
<head>
<title>Problema</title>
</head>
<body>
<?php
    $ar=fopen("datos.txt","r") or
        die("No se pudo abrir el archivo");
    while (!feof($ar))
    {
        $linea=fgets($ar);
        $lineasalto=nl2br($linea);
        echo $lineasalto;
    }
    fclose($ar);
?>
</body>
</html>
```

Lo primero que debemos identificar es la forma de apertura del archivo:

```
$ar=fopen("datos.txt","r") or
die("No se pudo abrir el archivo");
```

El segundo parámetro de fopen es "r" es decir read (apertura para lectura), si el archivo no existe por ejemplo se ejecuta la función die que finaliza el programa mostrando el string correspondiente.

La función feof retorna true si se ha llegado al final del archivo en caso contrario retorna false. Para que se impriman todas las líneas del archivo se plantea una estructura repetitiva que se ejecuta mientras no se llegue al final de archivo:

```
while (!feof($ar))
```

Dentro de la estructura repetitiva leemos una línea completa del archivo de texto con la función fgets:

```
$linea=fgets($ar);
```

La variable \$linea contiene una línea completa del archivo de texto, inclusive el salto de línea (\\n)

Como el navegador no hace un salto de línea con este caracter, debemos convertir dicho caracter a la marca
 propia de HTML. La función que realiza esta actividad se llama nl2br (new line to br)

El resultado se almacena en una nueva variable que es la que realmente imprimimos:

```
$lineasalto=nl2br($linea);
echo $lineasalto;
```

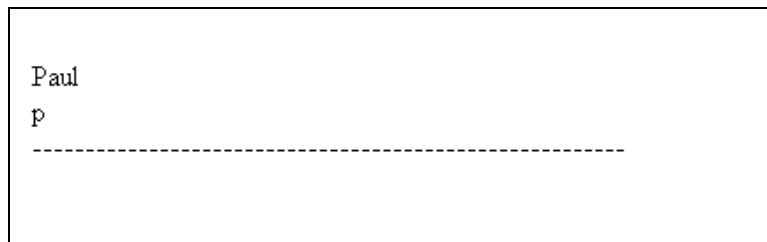
pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>

<?php
$ar=fopen("datos.txt","r") or
    die("No se pudo abrir el archivo");
while (!feof($ar))
{
    $linea=fgets($ar);
```

```
$lineasalto=nl2br($linea);  
echo $lineasalto;  
}  
fclose($ar);  
?>
```

```
</body>  
</html>
```



Confeccionar un programa que muestre el archivo de pedido de pizzas via internet del punto anterior. (obligatoriamente dar el nombre del archivo de texto como "datos.txt", esto es para que no se llene mi disco duro de archivos, es decir de ahora en más siempre que cree un archivo de texto debe llamarse datos.txt)

Vectores (asociativos)

Este tipo de vectores no es común a otros lenguajes, pero en PHP son de uso indispensable en distintas situaciones (ya lo empleamos cuando recuperamos información de un formulario accediendo al vector `$_REQUEST` que crea PHP en forma automática, cuando accedamos a datos de una base de datos también lo emplearemos etc.) Un vector asociativo permite acceder a un elemento del vector por medio de un subíndice de tipo string.

Inicialmente uno piensa que esto nos complica las cosas, como veremos más adelante la realidad nos demuestra lo contrario. Como ejemplo, consideremos que deseamos guardar en un vector el DNI, nombre y dirección de una persona. Empleando un vector con subíndice entero la solución sería:

```
<?php  
$registro[]="20456322";  
$registro[]="Martinez Pablo";  
$registro[]="Colon 134";  
?>
```

De esta forma debemos recordar que cuando deseamos mostrar el nombre de la persona debemos acceder al subíndice 1. Esto es sencillo si tenemos un vector con tres elementos, pero que sucede si debemos almacenar 20 datos relacionados en un vector?

Un vector asociativo se define de la siguiente forma:

```
<?php  
$registro['dni']="20456322";  
$registro['nombre']="Martinez Pablo";  
$registro['direccion']="Colon 134";
```



```
echo $registro['nombre'];  
?>
```

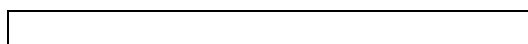
Ahora vemos que para imprimir el nombre de la persona no debemos recordar una posición dentro de un vector sino un nombre de clave. Esto se hace indispensable cuando administramos un conjunto de datos grandes. En un vector asociativo toda componente está asociada a una clave.

Otras formas de crear un vector asociativo:

```
<?php  
$registro=array('dni'=>'20456322',  
                'nombre'=>'Martinez Pablo',  
                'direccion'=>'Colon 134');  
echo $registro['dni'];  
?>
```

pagina1.php

```
<html>  
<head>  
<title>Problema</title>  
</head>  
<body>  
  
<?php  
//Almacenar en un vector asociativo la cantidad de dias que tiene  
//cada mes del año. Luego accederlo por su nombre como subíndice.  
$mes['enero']=31;  
$mes['febrero']=29;  
$mes['marzo']=31;  
$mes['abril']=30;  
$mes['mayo']=31;  
$mes['junio']=30;  
$mes['julio']=31;  
$mes['agosto']=31;  
$mes['septiembre']=30;  
$mes['octubre']=31;  
$mes['noviembre']=30;  
$mes['diciembre']=31;  
  
echo "Enero tiene:".$mes['enero']."<br>";  
echo "Agosto tiene:".$mes['agosto']."<br>";  
echo "Septiembre tiene:".$mes['septiembre']."<br>";  
  
?>  
  
</body>  
</html>
```



Enero tiene:31 Agosto tiene:31 Septiembre tiene:30
--

Crear un vector asociativo que almacene las claves de acceso de 5 usuarios de un sistema. Acceder a cada componente por su nombre. Imprimir una componente del vector.

Funciones en PHP

La sintaxis para la definición de una función en PHP es:

```
function [nombre de la función]([parámetros])  
{  
[algoritmo]  
}
```

Implementaremos una función que muestre un mensaje centrado en pantalla, y la llamaremos posteriormente dos veces:

```
<html>  
<head>  
<title>Problema</title>  
</head>  
<body>  
<?php  
function mensajecentrado($men)  
{  
    echo "<table width=\"100%\" border=\"1\">";  
    echo "<tr><td align=\"center\">";  
    echo $men;  
    echo "</tr></td>";  
    echo "</table>";  
}  
  
mensajecentrado("Primer recuadro");  
echo "<br>";  
mensajecentrado("Segundo recuadro");  
?>  
</body>  
</html>
```

Para mostrar el texto centrado en un recuadro utilizamos la marca table de HTML. Definimos las propiedades border con 1, para que sea visible y el ancho de 100% para que ocupe todo el navegador. La tabla tiene una fila a la que definimos con la marca tr (table row) y un solo dato en esa fila mediante la marca td (table data). Para que el texto dentro de la tabla salga centrado, inicializamos la propiedad align de la marca td.

Si vemos la función, notamos que lo más trabajoso es definir todas las marcas HTML para crear la tabla. Es importante notar que en PHP para introducir las dobles comillas dentro de un string debemos antecederle el carácter '\\'; para introducir el caracter '\\' debemos tipear \\\.

Las llamadas a la función las hacemos por medio de su nombre y pasándole el único parámetro que requiere:

```
mensajecentrado("Primer recuadro");  
echo "<br>";  
mensajecentrado("Segundo recuadro");
```

Las funciones nos permiten tener un programa más ordenado y facilitan la reutilización del código. Más adelante veremos como hacer archivos con rutinas comunes a muchas páginas.

Una función puede retornar un dato, supongamos que necesitamos una función que nos retorne el promedio de dos valores, el código sería:

```
<html>  
<head>  
<title>Problema</title>  
</head>  
<body>  
  
<?php  
function retornarpromedio($valor1,$valor2)  
{  
    $pro=$valor1/$valor2;  
    return $pro;  
}  
  
$v1=100;  
$v2=50;  
$p=retornarpromedio($v1,$v2);  
echo $p;  
?>  
  
</body>  
</html>
```

Cuando una función retorna un dato debemos emplear la palabra clave return seguida del valor que devuelve.

En la llamada a la función el valor retornado se almacena generalmente en una variable:

```
$p=retornarpromedio($v1,$v2);
```

Si queremos que retorne más de un dato debemos emplear parámetros por referencia. Supongamos que necesitamos ahora que una función nos retorne el cuadrado y cubo de un número:

```
<html>  
<head>  
<title>Problema</title>  
</head>  
<body>  
<?php  
function cuadradocubo($valor,&$cuad,&$cub)  
{  
    $cuad=$valor*$valor;  
    $cub=$valor*$valor*$valor;  
}  
cuadradocubo(2,$c1,$c2);  
echo "El cuadrado de 2 es:".$c1."<br>";  
echo "El cubo de 2 es:".$c2;  
?>  
</body>  
</html>
```

Es decir, cuando le antecedemos el carácter ampersand al parámetro, es por referencia. El objetivo es asignarle cierto valor al parámetro y posteriormente el dato quedará almacenado en la variable que le pasamos a la función.

```
function cuadradocubo ($valor, &$cuad, &$cub)
{
    $cuad=$valor*$valor;
    $cub=$valor*$valor*$valor;
}
```

El parámetro \$cuad se almacena en la variable \$c1 y el parámetro \$cub se almacena en \$c2. Es la forma más adecuada de modificar variables dentro de una función.

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>

<?php
function mensajecentrado($men)
{
    echo "<table width=\"100%\" border=\"1\">";
    echo "<tr><td align=\"center\">";
    echo $men;
    echo "<tr><td>";
    echo "</table>";
}

mensajecentrado("Primer recuadro");
echo "<br>";
mensajecentrado("Segundo recuadro");

?>

</body>
</html>
```

Primer recuadro
Segundo recuadro

Confeccionar un formulario que solicite la carga del nombre de usuario y su clave en dos oportunidades. En la página que se procesan los datos del formulario implementar una función que imprima un mensaje si las dos claves ingresadas son distintas.

Base de datos (MySQL)

Uno de los empleos principales de PHP es el acceso a una base de datos en el servidor. Las operaciones básicas se hacen empleando como lenguaje el SQL. PHP implementa distintas funciones según la base de datos a emplear. Existen funciones actualmente para acceder a las siguientes servidores de bases de datos:

- MySQL
- Microsoft SQL Server
- Oracle
- PostgreSQL
- SysBase
- FrontBase
- Informix
- InterBase
- Ingres
- mSQL
- dBase
- SQLite

El más empleado en la actualidad en la web es el gestor de base de datos MySQL (debido que cuando se lo emplea sin fines de lucro se puede emplear el software en forma gratuita).

Yo, en este servidor, he creado una base de datos llamada phpfacil (por razones de seguridad no he permitido la creación de nuevas bases de datos, luego usted podrá instalar en su equipo el mysql y dispondrá de todos los permisos para la creación de bases de datos).

Nombre de la base de datos: phpfacil

También he creado una tabla llamada: alumnos

La estructura de la misma es:

```
CREATE TABLE alumnos (  
  codigo int(11) NOT NULL auto_increment,  
  nombre varchar(40) default NULL,  
  mail varchar(50) default NULL,  
  codigocurso int(11) default NULL,  
  PRIMARY KEY (`codigo`)  
)
```

La tabla almacenará datos de alumnos que desarrollarán cursos de programación en PHP, ASP y JSP.

El código del alumno es de tipo numérico (int) y al indicar que es auto_increment se generará automáticamente por el gestor de base de datos.

Los campos nombre y mail son de tipo varchar (podemos almacenar cualquier caracter) y por último el campo codigocurso representa el curso a tomar por el alumno (1=PHP, 2=ASP y 3=JSP)

El campo clave de esta tabla es el código de alumno (es decir no podemos tener dos alumnos con el mismo código, no así el nombre del alumno que puede eventualmente repetirse)

INSERT (Alta de registros en una tabla)

Luego de crear una base de datos y sus tablas (como dijimos, en este servidor vamos a trabajar con una base de datos ya creada: phpfacil, que contiene la tabla alumnos), veremos como agregar registros.

Para añadir datos en la tabla empleamos el comando SQL llamado insert. Necesitamos dos páginas para este proceso, una será el formulario de carga de datos y la siguiente será la que efectúe la inserción en la tabla.

Formulario de carga de datos:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<h1>Alta de Alumnos</h1>
<form action="pagina2.php" method="post">
Ingrese nombre:
<input type="text" name="nombre"><br>
Ingrese mail:
<input type="text" name="mail"><br>
Seleccione el curso:
<select name="codigocurso">
<option value="1">PHP</option>
<option value="2">ASP</option>
<option value="3">JSP</option>
</select>
<br>
<input type="submit" value="Registrar">
</form>
</body>
</html>
```

El formulario es bastante similar a los que venimos desarrollando en puntos anteriores, tal vez lo distinto es cómo emplearemos el control "select" del curso a desarrollar:

```
<select name="codigocurso">
<option value="1">PHP</option>
<option value="2">ASP</option>
<option value="3">JSP</option>
</select>
```

Cada opción tiene su respectivo valor (en este caso los números 1,2 y 3) y los textos a mostrar PHP, ASP y JSP. El dato que se envía a la otra página es el código de curso (esto debido a que definimos la propiedad value).

Ahora veremos como realizar la registración de los datos cargados en el formulario, en la tabla alumnos de la base de datos phpfacil:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80")
or die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
die("Problemas en la seleccion de la base de datos");
mysql_query("insert into alumnos(nombre,mail,codigocurso) values
```

```
        ('$_REQUEST[nombre]', '$_REQUEST[mail]', $_REQUEST[codigocurso])",
        $conexion) or die("Problemas en el select".mysql_error());
mysql_close($conexion);
echo "El alumno fue dado de alta.";
?>
</body>
</html>
```

Veamos los pasos para efectuar el alta en la tabla alumnos:

```
$conexion=mysql_connect("localhost","root","z80")
or die("Problemas en la conexion");
```

La función `mysql_connect` se conecta a una base de datos de tipo MySQL, el primer parámetro es la dirección donde se encuentra el gestor de base de datos (en este caso en el mismo servidor por lo que indicamos esto con "localhost"), el segundo parámetro es el nombre de usuario de la base de datos ("root" en nuestro caso, que es el usuario por defecto que crea MySQL para el administrador) y por último la clave del usuario (cuando yo instalé MySQL definí como clave la cadena "z80"). En caso de hacer algún error en la llamada a la función la misma retorna false por lo que se ejecuta la instrucción seguida del operador `or`, en nuestro caso llamamos a la función `die` que detiene la ejecución del programa y muestra el mensaje por pantalla.

Paso seguido se selecciona una base de datos (ya que el gestor de base de datos puede administrar varias bases de datos):

```
mysql_select_db("phpfacil",$conexion) or
die("Problemas en la seleccion de la base de datos");
```

A esta función le indicamos como primer parámetro el nombre de la base de datos con la que trabajaremos y como segundo parámetro la referencia que retornó la función `mysql_connect`.

El paso más importante es la codificación del comando SQL `insert`:

```
mysql_query("insert into alumnos(nombre,mail,codigocurso) values
('$_REQUEST[nombre]', '$_REQUEST[mail]', $_REQUEST[codigocurso])",
$conexion) or die("Problemas en el select".mysql_error());
```

La sintaxis del comando `insert` es bastante sencilla, indicamos el nombre de la tabla y los campos de la tabla a cargar. Luego debemos indicar en el mismo orden los valores a cargar en cada campo (dichos valores los rescatamos del formulario anterior). Los campos de tipo `varchar` SQL requiere que encerremos entre comillas simples, esto sucede para el nombre y el mail; en cambio, para el `codigocurso` esto no debe ser así. Otra cosa a tener en cuenta es que los subíndices de los vectores asociativos no deben ir entre simples comillas ya que se encuentran dentro de un string, sino se producirá un error. En caso que MySQL detecte un error, retorna false esta función, por lo que se ejecuta la instrucción posterior al `or`, es decir la función `die` que mostrará el error generado por MySQL llamando a la función `mysql_error()`. Por último cerramos la conexión con la base de datos y mostramos un mensaje indicando que la carga se efectuó en forma correcta. Tener en cuenta que el campo código se generó en forma automática.

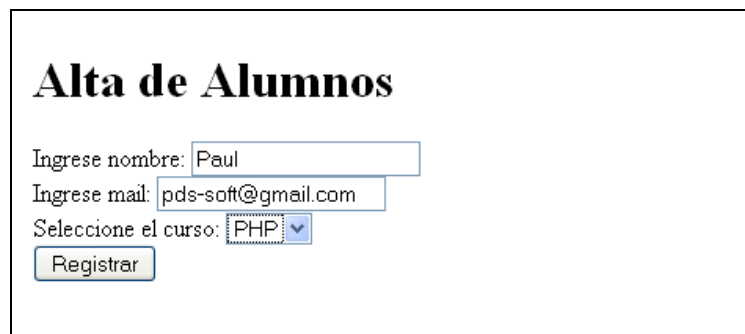
pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
```

```
<body>
<h1>Alta de Alumnos</h1>
<form action="pagina2.php" method="post">
Ingrese nombre:
<input type="text" name="nombre"><br>
Ingrese mail:
<input type="text" name="mail"><br>
Seleccione el curso:
<select name="codigocurso">
  <option value="1">PHP</option>
  <option value="2">ASP</option>
  <option value="3">JSP</option>
</select>
<br>
<input type="submit" value="Registrar">
</form>
</body>
</html>
```

pagina2.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
  die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
  die("Problemas en la seleccion de la base de datos");
mysql_query("insert into alumnos(nombre,mail,codigocurso) values
('$ _REQUEST[nombre]','$ _REQUEST[mail]','$ _REQUEST[codigocurso]'",
$conexion) or
  die("Problemas en el select".mysql_error());
mysql_close($conexion);
echo "El alumno fue dado de alta.";
?>
</body>
</html>
```



Alta de Alumnos

Ingrese nombre: Paul

Ingrese mail: pds-soft@gmail.com

Seleccione el curso: PHP

Registrar

El alumno fue dado de alta.

Ya existe una tabla llamada: cursos en la base de datos: phpfacil. Efectuar el alta de la tabla.

La tabla cursos tiene la siguiente estructura:

```
CREATE TABLE `cursos` (  
  `codigo` int(11) NOT NULL auto_increment,  
  `nombrecur` varchar(40) NOT NULL,  
  PRIMARY KEY (`codigo`)  
)
```

Es decir que, por teclado sólo debemos cargar el nombre del curso (nombrecur).

Listado (selección de registros de una tabla)

Ahora veremos como recuperar los datos almacenados en la tabla alumnos de la base de datos phpfacil.

El programa que muestra los registros en una página es:

```
<html>  
<head>  
<title>Problema</title>  
</head>  
<body>  
<?php  
$conexion=mysql_connect("localhost","root","z80")  
  or die("Problemas en la conexion");  
mysql_select_db("phpfacil",$conexion)  
  or die("Problemas en la selección de la base de datos");  
$registros=mysql_query("select codigo,nombre, mail, codigocurso  
                        from alumnos",$conexion) or  
  die("Problemas en el select:".mysql_error());  
while ($reg=mysql_fetch_array($registros))  
{  
  echo "Codigo:".$reg['codigo']."<br>";  
  echo "Nombre:".$reg['nombre']."<br>";  
  echo "Mail:".$reg['mail']."<br>";  
  echo "Curso:";  
  switch ($reg['codigocurso']) {  
    case 1:echo "PHP";  
      break;  
    case 2:echo "ASP";  
      break;  
    case 3:echo "JSP";  
      break;  
  }  
  echo "<br>";  
  echo "<hr>";  
}  
mysql_close($conexion);  
?>  
</body>  
</html>
```

La primer parte es similar a lo visto hasta ahora, es decir nos conectamos a la base de datos y seleccionamos la base de datos phpfacil.

El comando SQL que nos permite recuperar datos de tablas se llama SELECT. Indicamos los campos a rescatar de la tabla y luego de la palabra clave from indicamos el nombre de la tabla:

```
$registros=mysql_query("select codigo,nombre, mail, codigocurso  
                        from alumnos",$conexion) or  
    die("Problemas en el select:".mysql_error());
```

En caso de haber codificado incorrectamente, el comando SQL select la función mysql_query retorna false, por lo que se ejecuta el comando siguiente al operador or, es decir la función die.

Si el comando SQL es correcto, en la variable \$registros se almacena una referencia a los datos rescatados de la tabla alumnos. Ahora debemos ir mostrando registro a registro los datos extraídos:

```
while ($reg=mysql_fetch_array($registros))
```

Para rescatar registro a registro los datos obtenidos por el select debemos llamar a la función mysql_fetch_array. Esta función retorna un vector asociativo con los datos del registro rescatado, o false en caso de no haber más registros. Es decir que si retorna un registro se almacena en el vector \$reg y la condición del while se valida como verdadero y pasa a ejecutarse el bloque del while:

```
{  
    echo "Codigo:".$reg['codigo']."<br>";  
    echo "Nombre:".$reg['nombre']."<br>";  
    echo "Mail:".$reg['mail']."<br>";  
    echo "Curso:";  
    switch ($reg['codigocurso']) {  
        case 1:echo "PHP";  
            break;  
        case 2:echo "ASP";  
            break;  
        case 3:echo "JSP";  
            break;  
    }  
    echo "<br>";  
    echo "<hr>";  
}
```

El bloque del while muestra el contenido del registro rescatado por la función mysql_fetch_array. Como vemos, para rescatar cada campo accedemos mediante el vector asociativo \$reg indicando como subíndice un campo indicado en el select:\$reg['codigo'] Cada vez que llamamos a la función mysql_fetch_array nos retorna el siguiente registro. Cuando debemos mostrar el curso mediante la instrucción switch, analizamos si tiene un 1,2 ó 3 y procedemos a mostrar el nombre del curso. Para separar cada alumno en la página HTML llamamos disponemos la marca "<hr>"

pagina1.php

```
<html>  
<head>  
<title>Problema</title>  
</head>  
<body>  
  
<?php  
$conexion=mysql_connect("localhost","root","z80") or  
    die("Problemas en la conexion");  
mysql_select_db("phpfacil",$conexion) or
```

```
die("Problemas en la selección de la base de datos");
$registros=mysql_query("select codigo,nombre, mail, codigocurso
                        from alumnos",$conexion) or
die("Problemas en el select:".mysql_error());
while ($reg=mysql_fetch_array($registros))
{
    echo "Codigo:".$reg['codigo']."<br>";
    echo "Nombre:".$reg['nombre']."<br>";
    echo "Mail:".$reg['mail']."<br>";
    echo "Curso:";
    switch ($reg['codigocurso']) {
        case 1:echo "PHP";
            break;
        case 2:echo "ASP";
            break;
        case 3:echo "JSP";
            break;
    }
    echo "<br>";
    echo "<hr>";
}
mysql_close($conexion);
?>

</body>
</html>
```

```
Codigo:55571
Nombre:Gato
Mail:oso@gmail.com
Curso:PHP


---


Codigo:55572
Nombre:Rocky
Mail:carlita@gmail.com
Curso:


---


Codigo:55573
Nombre:nombre
Mail:vjhvk
Curso:ASP


---


Codigo:55574
Nombre:nkljklñhñfklhn
Mail:dcjgfdchgfgf@
Curso:PHP
```

Confeccionar un programa que recupere los datos de la tabla cursos de la base de datos phpfacil.

Consulta (selección de registros de una tabla)

El proceso de consulta de datos de una tabla es similar al del listado, la diferencia es que se muestra sólo aquel que cumple la condición por la que buscamos. Haremos un programa que nos permita consultar los datos de un alumno ingresando su mail para su búsqueda. Tengamos en cuenta que no puede haber dos alumnos con el mismo mail, por lo que la consulta nos puede arrojar uno o ningún registro. Debemos codificar un formulario para el ingreso del mail a consultar:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese el mail del alumno a consultar:
<input type="text" name="mail">
<br>
<input type="submit" value="buscar">
</form>
</body>
</html>
```

Por otro lado tenemos el archivo "pagina2.php" que se encarga de buscar el mail ingresado en el formulario:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select codigo,nombre, codigocurso
                        from          alumnos                      where
mail='$_REQUEST[mail]'", $conexion) or
    die("Problemas en el select:".mysql_error());
if ($reg=mysql_fetch_array($registros))
{
    echo "Nombre:".$reg['nombre']."<br>";
    echo "Curso:";
    switch ($reg['codigocurso']) {
        case 1:echo "PHP";
            break;
        case 2:echo "ASP";
            break;
        case 3:echo "JSP";
            break;
    }
}
else
{
    echo "No existe un alumno con ese mail.";
}
mysql_close($conexion);
?>
</body>
```

</html>

Lo más importante está en el comando select:

```
$registros=mysql_query("select codigo,nombre, codigocurso
                        from          alumnos                      where
mail='$_REQUEST[mail]'", $conexion) or
die("Problemas en el select:".mysql_error());
```

Acá es donde con la clausula where seleccionamos sólo el registro que cumple con la condición que el mail sea igual al que ingresamos. Como sólo puede haber un registro que cumpla la condición, llamamos a la función mysql_fetch_array en un if:

```
if ($reg=mysql_fetch_array($registros))
```

En caso de retornar un vector asociativo la condición del if se verifica como verdadera y pasa a mostrar los datos, en caso de retornar false se ejecuta el else.

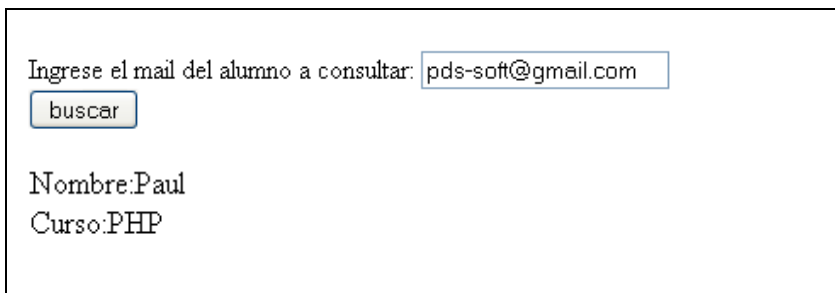
pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese el mail del alumno a consultar:
<input type="text" name="mail">
<br>
<input type="submit" value="buscar">
</form>
</body>
</html>
```

pagina2.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
die("Problemas en la selección de la base de datos");
$registros=mysql_query("select codigo,nombre, codigocurso
                        from alumnos where mail='$_REQUEST[mail]'", $conexion) or
die("Problemas en el select:".mysql_error());
if ($reg=mysql_fetch_array($registros))
{
echo "Nombre:".$reg['nombre']."<br>";
echo "Curso:";
switch ($reg['codigocurso']) {
case 1:echo "PHP";
```

```
        break;
    case 2:echo "ASP";
        break;
    case 3:echo "JSP";
        break;
    }
}
else
{
    echo "No existe un alumno con ese mail.";
}
mysql_close($conexion);
?>
</body>
</html>
```

A screenshot of a web browser displaying a search form. The form has a label "Ingrese el mail del alumno a consultar:" followed by a text input field containing "pds-soft@gmail.com". Below the input field is a button labeled "buscar". Underneath the button, the results are displayed as "Nombre:Paul" and "Curso:PHP".

Ingrese el mail del alumno a consultar:

Nombre:Paul
Curso:PHP

Confeccionar un programa que permita ingresar el nombre de un alumno en un formulario, luego mostrar los datos del mismo (tener en cuenta que puede haber más de un alumno con el mismo nombre).

DELETE (Baja de un registro en una tabla)

El objetivo de este punto es el borrado de un registro de una tabla. Para ello, implementaremos un algoritmo que solicite ingresar el mail de un alumno y posteriormente efectue su borrado.

La primera página es idéntica a la consulta, ya que debemos implementar un formulario que solicite la carga del mail del alumno:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese el mail del alumno a borrar:
<input type="text" name="mail">
<br>
<input type="submit" value="buscar">
</form>
</body>
</html>
```

Por otro lado tenemos el archivo "pagina2.php" que se encarga de buscar el mail ingresado en el formulario y en caso que exista se procede a borrarlo:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select codigo from alumnos
                        where mail='".$_REQUEST[mail]'", $conexion) or
    die("Problemas en el select:".mysql_error());
if ($reg=mysql_fetch_array($registros))
{
    mysql_query("delete          from          alumnos          where
mail='".$_REQUEST[mail]'", $conexion) or
    die("Problemas en el select:".mysql_error());
    echo "Se efectuó el borrado del alumno con dicho mail.";
}
else
{
    echo "No existe un alumno con ese mail.";
}
mysql_close($conexion);
?>
</body>
</html>
```

En esta segunda página efectuamos dos llamadas a la función `mysql_query`, una para consultar si existe el mail ingresado y otra para efectuar el borrado del registro respectivo.

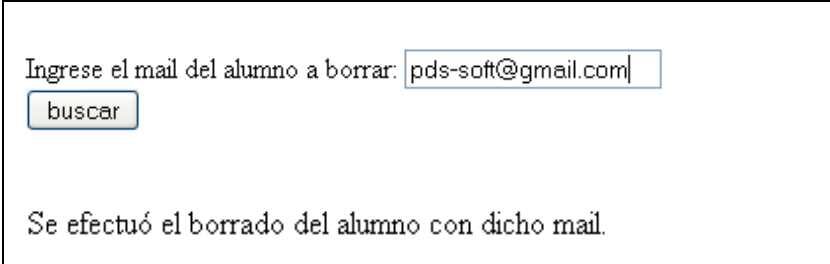
pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese el mail del alumno a borrar:
<input type="text" name="mail">
<br>
<input type="submit" value="buscar">
</form>
</body>
</html>
```

pagina2.php

```
<html>
<head>
<title>Problema</title>
</head>
```

```
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
die("Problemas en la selección de la base de datos");
$registros=mysql_query("select codigo from alumnos
                        where mail='$_REQUEST[mail]'", $conexion) or
die("Problemas en el select:".mysql_error());
if ($reg=mysql_fetch_array($registros))
{
    mysql_query("delete from alumnos where mail='$_REQUEST[mail]'", $conexion)
or
    die("Problemas en el select:".mysql_error());
    echo "Se efectuó el borrado del alumno con dicho mail.";
}
else
{
    echo "No existe un alumno con ese mail.";
}
mysql_close($conexion);
?>
</body>
</html>
```

A screenshot of a web browser window showing a form. The form has a label "Ingrese el mail del alumno a borrar:" followed by a text input field containing "pds-soft@gmail.com". Below the input field is a button labeled "buscar". Below the button, the text "Se efectuó el borrado del alumno con dicho mail." is displayed.

Ingrese el mail del alumno a borrar:

Se efectuó el borrado del alumno con dicho mail.

Confeccionar un programa que permita ingresar el nombre de un curso por teclado y posteriormente efectúe el borrado de dicho registro en la tabla cursos. Mostrar un mensaje si no existe el curso.

DELETE (Baja de todos los registros de una tabla)

Para borrar todos los registros de una tabla debemos llamar al comando delete de SQL sin disponer la cláusula where:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
die("Problemas en la conexion");
```



```
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
mysql_query("delete from alumnos",$conexion) or
    die("Problemas en el select:".mysql_error());
echo "Se efectuó el borrado de todos los alumnos.";
mysql_close($conexion);
?>
</body>
</html>
```

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
mysql_query("delete from alumnos",$conexion) or
    die("Problemas en el select:".mysql_error());
echo "Se efectuó el borrado de todos los alumnos.";
mysql_close($conexion);
?>
</body>
</html>
```

Se efectuó el borrado de todos los alumnos.

Efectuar el borrado de todos los registros de la tabla cursos.

UPDATE (Modificación de un registro de una tabla)

De las actividades con tablas esta es la más larga. Vamos a resolverlo implementando tres páginas, la primera un formulario de consulta del mail de un alumno, la segunda otro formulario que nos permita cargar su mail modificado y la última registrará el cambio en la tabla.

El formulario de consulta del mail del alumno es similar a problemas anteriores:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese el mail del alumno:
```

```
<input type="text" name="mail"><br>
<input type="submit" value="buscar">
</form>
</body>
</html>
```

La segunda página es la más interesante y con conceptos nuevos:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select * from alumnos
                        where mail='$_REQUEST[mail]'", $conexion) or
    die("Problemas en el select:".mysql_error());
if ($reg=mysql_fetch_array($registros))
{
    ?>
    <form action="pagina3.php" method="post">
    Ingrese nuevo mail:
    <input type="text" name="mailnuevo" value="<?php echo $reg['mail'] ?>">
    <br>
    <input type="hidden" name="mailviejo" value="<?php
    echo $reg['mail'] ?>">
    <input type="submit" value="Modificar">
    </form>
<?php
}
else
    echo "No existe alumno con dicho mail";
?>
</body>
</html>
```

Lo primero que podemos observar es que si el if se verifica verdadero se ejecuta un bloque que contiene código HTML:

```
if ($reg=mysql_fetch_array($registros))
{
    ?>
    <form action="pagina3.php" method="post">
    Ingrese nuevo mail:
    <input type="text" name="mailnuevo" value="<?php
    echo $reg['mail'] ?>">
    <br>
    <input type="hidden" name="mailviejo" value="<?php echo $reg['mail']
    ?>">
    <input type="submit" value="Modificar">
    </form>
<?php
}
}
```

Es decir que podemos disponer bloques de PHP dispersos dentro de la página. Otro concepto importante es como enviar el mail del primer formulario a la tercer página, esto se logra con los controles de tipo "hidden", este tipo de control no se visualiza en el formulario pero se envía al presionar el botón submit.

Si queremos que el control text se inicialice con el mail ingresado en el formulario anterior debemos cargar la propiedad value con dicho valor:

```
<input type="text" name="mailnuevo" value="<?php echo $reg['mail'] ?>">
```

Por último la pagina3.php es la que efectúa la modificación de la tabla propiamente dicha. Con el mail ingresado en la pagina1.php, el mail modificado en la pagina2.php se efectúa el update.

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("update alumnos
                        set mail='$_REQUEST[mailnuevo]'
                        where mail='$_REQUEST[mailviejo]'", $conexion) or
    die("Problemas en el select:".mysql_error());
    echo "El mail fue modificado con exito";
?>
</body>
</html>
```

Tengamos en cuenta que el segundo formulario nos envía dos datos: \$_REQUEST[mailnuevo] y \$_REQUEST[mailviejo].

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese el mail del alumno:
<input type="text" name="mail"><br>
<input type="submit" value="buscar">
</form>
</body>
</html>
```

pagina2.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>

<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
```

```
die("Problemas en la selección de la base de datos");
$registros=mysql_query("select * from alumnos
                        where mail='$_REQUEST[mail]'", $conexion) or
die("Problemas en el select:".mysql_error());
if ($reg=mysql_fetch_array($registros))
{
?>

<form action="pagina3.php" method="post">
Ingrese nuevo mail:
<input type="text" name="mailnuevo" value="<?php echo $reg['mail'] ?>">
<br>
<input type="hidden" name="mailviejo" value="<?php echo $reg['mail'] ?>">
<input type="submit" value="Modificar">
</form>

<?php
}
else
    echo "No existe alumno con dicho mail";
?>
</body>
</html>
```

pagina3.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
die("Problemas en la conexion");
mysql_select_db("phpfacil", $conexion) or
die("Problemas en la selección de la base de datos");
$registros=mysql_query("update alumnos
                        set mail='$_REQUEST[mailnuevo]'
                        where mail='$_REQUEST[mailviejo]'", $conexion) or
die("Problemas en el select:".mysql_error());
echo "El mail fue modificado con exito";
?>
</body>
</html>
```

Ingrese el mail del alumno: pds-soft@gmail.com

buscar

No existe alumno con dicho mail

Efectuar la modificación del nombre del curso de la tabla "cursos". Para la búsqueda ingresar el código de curso.

INSERT (y consulta de otra tabla)

Ahora vamos a ver como resolver el problema del alta de un alumno seleccionando el curso de la tabla "cursos". Es decir, el formulario de carga de datos no es HTML puro ya que debemos cargar el control "select" con los datos de la tabla cursos. El código por lo tanto queda de la siguiente forma:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese nombre:
<input type="text" name="nombre"><br>
Ingrese mail:
<input type="text" name="mail"><br>
Seleccione el curso:
<select name="codigocurso">
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select codigo,nombrecur from cursos",$conexion)
or
    die("Problemas en el select:".mysql_error());
while ($reg=mysql_fetch_array($registros))
{
    echo "<option value=\"".$reg[codigo].\">".$reg[nombrecur]</option>";
}
?>
</select>
<br>
<input type="submit" value="Registrar">
</form>
</body>
</html>
```

El algoritmo es similar a cuando trabajamos con una tabla, pero el control "select" lo cargamos con los datos de la tabla "cursos":

```
while ($reg=mysql_fetch_array($registros))
{
```

```
    echo "<option value=\"\$reg[codigo]\">\$reg[nombrecur]</option>";
}
```

Dentro del while generamos todas las opciones que contiene el "select" imprimiendo el campo nombrecur y asociando el campo codigo a la propiedad value(que es en definitiva el código que necesitamos rescatar en la otra página)

La página que efectúa el insert es exactamente la misma que vimos anteriormente:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la seleccion de la base de datos");
mysql_query("insert into alumnos(nombre,mail,codigocurso) values
    ('$_REQUEST[nombre]', '$_REQUEST[mail]',
    $_REQUEST[codigocurso])", $conexion) or
    die("Problemas en el select".mysql_error());
mysql_close($conexion);
echo "El alumno fue dado de alta.";
?>
</body>
</html>
```

pagina1.php.

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese nombre:
<input type="text" name="nombre"><br>
Ingrese mail:
<input type="text" name="mail"><br>
Seleccione el curso:
<select name="codigocurso">
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select codigo,nombrecur from cursos",$conexion) or
    die("Problemas en el select:".mysql_error());
while ($reg=mysql_fetch_array($registros))
{
    echo "<option value=\"\$reg[codigo]\">\$reg[nombrecur]</option>";
}
?>
</select>
```

```
<br>
<input type="submit" value="Registrar">
</form>
</body>
</html>
```

pagina2.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la seleccion de la base de datos");
mysql_query("insert into alumnos(nombre,mail,codigocurso) values
('$ _REQUEST[nombre]','$ _REQUEST[mail]','$ _REQUEST[codigocurso]'",
$conexion) or
    die("Problemas en el select".mysql_error());
mysql_close($conexion);
echo "El alumno fue dado de alta.";
?>
</body>
</html>
```

Confeccionar el alta de la tabla alumnos empleando controles de tipo "radio" para la selección del curso.

Listado (selección de registros de varias tablas - INNER JOIN)

Ahora veremos como imprimir todos los alumnos inscriptos a los cursos junto al nombre del curso donde está inscripto. Los datos se encuentran en las tablas "alumnos" y "cursos". Debemos aparear el código de curso de la tabla "alumnos" con el código de la tabla "cursos". El código del programa que hace esto es el siguiente:

```
<html>
```

```
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select
                        alu.codigo
                        nombre,mail,codigocurso,
                        nombrecur
                        from alumnos as alu
                        inner join cursos as cur on
                        cur.codigo=alu.codigocurso",
                        $conexion) or
    die("Problemas en el select:".mysql_error());
while ($reg=mysql_fetch_array($registros))
{
    echo "Codigo:".$reg['codigo']."<br>";
    echo "Nombre:".$reg['nombre']."<br>";
    echo "Mail:".$reg['mail']."<br>";
    echo "Curso:".$reg['nombrecur']."<br>";
    echo "<hr>";
}
mysql_close($conexion);
?>
</body>
</html>
```

Hay varias cosas nuevas cuya sintaxis necesitamos analizar, la primera es como hacer el apareo con la tabla cursos:

```
inner join cursos as cur on cur.codigo=alu.codigocurso",
```

Luego de las palabras claves inner join, indicamos la tabla que necesitamos aparear, podemos crear un alias de una palabra mediante la palabra clave as. En el resto de la consulta, en vez de indicar el nombre de la tabla, hacemos referencia al alias(generalmente un nombre más corto). Seguidamente de la palabra clave on, indicamos los campos por los que apareamos las tablas, en nuestro caso el código de la tabla cursos con el código de la tabla alumnos. Otro punto a tener en cuenta es indicar en el select qué campos debemos rescatar de las tablas, es decir, indicarle a qué tabla pertenece en el caso que tengan el mismo nombre:

```
$registros=mysql_query("select
                        alu.codigo
                        nombre,mail,codigocurso,
                        nombrecur
                        from alumnos as alu
```

En este caso rescatamos el código del alumno (y no el código de curso). Debemos crear un alias si dos tablas tienen el mismo nombre de campo para evitar confusión; como ocurre aquí con el campo código de las tablas alumnos y cursos. También creamos un alias para la tabla alumnos. Cuando rescatamos los datos y los mostramos en la página, hacemos referencia al alias del campo:

```
while ($reg=mysql_fetch_array($registros))
{
    echo "Codigo:".$reg['codigo']."<br>";
    echo "Nombre:".$reg['nombre']."<br>";
    echo "Mail:".$reg['mail']."<br>";
    echo "Curso:".$reg['nombrecur']."<br>";
    echo "<hr>";
}
```


pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select alu.codigo as codigo,nombre,mail,codigocurso,
    nombrecur from alumnos as alu
    inner join cursos as cur on cur.codigo=alu.codigocurso",
    $conexion) or
    die("Problemas en el select:".mysql_error());
while ($reg=mysql_fetch_array($registros))
{
    echo "Codigo:".$reg['codigo']."<br>";
    echo "Nombre:".$reg['nombre']."<br>";
    echo "Mail:".$reg['mail']."<br>";
    echo "Curso:".$reg['nombrecur']."<br>";
    echo "<hr>";
}
mysql_close($conexion);
?>
</body>
</html>
```

Codigo:55587
Nombre:Paul
Mail:pds-soft@gmail.com
Curso:php avanzado

Confeccionar un programa que permita ingresar el código de un alumno y nos muestre su nombre, mail y nombre del curso en el cual está inscripto. Hacer un formulario donde se ingrese el código de alumno y otra página donde se muestren los datos respectivos. Mostrar un mensaje si no existe el código de alumno ingresado.

Función count de SQL.

Cuando necesitamos saber la cantidad de registros que cumplen una condición podemos utilizar la función count, por ejemplo si deseamos conocer la cantidad de alumnos que hay en la tabla "alumnos" la codificación será la siguiente:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select count(*) as cantidad
                        from alumnos",$conexion) or
    die("Problemas en el select:".mysql_error());
$reg=mysql_fetch_array($registros);
echo "La cantidad de alumnos inscriptos son :".$reg['cantidad'];
?>
</body>
</html>
```

En la sentencia select en vez de indicar los campos de la tabla, colocamos la llamada a la función count pasando como parámetro un asterisco y creando un alias para su posterior recuperación e impresión del dato:

```
$registros=mysql_query("select count(*) as cantidad
                        from alumnos",$conexion)
```

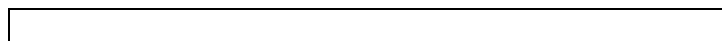
El select no tiene cláusula where ya que debemos contar todos los alumnos y no los de algún curso en particular.

La llamada a la función mysql_fetch_array se hace sin estructura condicional o repetitiva ya que sabemos que nos retornará un único registro (en realidad, un registro que tiene una sola columna llamada cantidad, en caso de estar vacía la tabla alumnos, se almacena cero en el alias cantidad):

```
$reg=mysql_fetch_array($registros);
```

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select count(*) as cantidad from
                        alumnos",$conexion) or
    die("Problemas en el select:".mysql_error());
$reg=mysql_fetch_array($registros);
echo "La cantidad de alumnos inscriptos son :".$reg['cantidad'];
?>
</body>
</html>
```



La cantidad de alumnos inscriptos son :1

Confeccionar un programa que muestre por pantalla los nombres de todos los cursos y al final la cantidad total de cursos.

UPDATE (Modificación de un registro trabajando con dos tablas)

Ahora complicaremos un poco la modificación de un registro consultando dos tablas. Supongamos que un alumno desea cambiarse de curso, es decir, se inscribió en uno y quiere borrarse de ese e inscribirse en otro diferente. Debemos mostrar en un "select" el curso actual en el que está inscripto y los otros cursos disponibles en la tabla "cursos". Para resolver este problema tenemos que plantear tres páginas, una donde ingresemos el mail del alumno, la segunda donde se pueda cambiar el curso y por última una que permita modificar la tabla "alumnos" con el nuevo curso seleccionado.

La primer página de ingreso del mail es:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese el mail del alumno:
<input type="text" name="mail"><br>
<input type="submit" value="buscar">
</form>
</body>
</html>
```

La segunda página y la más importante en cuanto a novedades es la siguiente:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select * from alumnos
                        where mail='".$_REQUEST[mail].'", $conexion) or
    die("Problemas en el select:".mysql_error());
if ($regalu=mysql_fetch_array($registros))
{
    ?>
<form action="pagina3.php" method="post">
<input type="hidden" name="mailviejo" value="<?php
    echo $regalu['mail'] ?>">
<select name="codigocurso">
<?php
    $registros=mysql_query("select * from cursos",$conexion) or
        die("Problemas en el select:".mysql_error());
    while ($reg=mysql_fetch_array($registros))
```

```
{
    if ($regalu['codigocurso']==$reg['codigo'])
        echo "selected>$reg[nombrecur]</option>";
    else
        echo "<option value=\"\$reg[codigo]\">$reg[nombrecur]</option>";
    }
?>
</select>
<br>
<input type="submit" value="Modificar">
</form>
<?php
}
else
    echo "No existe alumno con dicho mail";
?>
</body>
</html>
```

La primera consulta de la tabla alumnos es para verificar si existe un alumno con el mail ingresado por teclado:

```
$registros=mysql_query("select * from alumnos
                        where mail='$_REQUEST[mail]'", $conexion) or
    die("Problemas en el select:".mysql_error());
if ($regalu=mysql_fetch_array($registros))
{
```

En caso de existir un alumno con dicho mail, el if se verifica verdadero y pasamos a poblar el control select con los distintos cursos que contiene la tabla "cursos":

```
$registros=mysql_query("select * from cursos", $conexion) or
    die("Problemas en el select:".mysql_error());
while ($reg=mysql_fetch_array($registros))
{
    if ($regalu['codigocurso']==$reg['codigo'])
        echo "selected>$reg[nombrecur]</option>";
    else
        echo "<option value=\"\$reg[codigo]\">$reg[nombrecur]</option>";
    }
?>
</select>
```

Para que aparezca seleccionado el curso actual debemos inicializar lo marca option con el texto selected. Es decir que el if dentro del while se verifica verdadero una solo vez.

Por último la tercer página procede a registrar el cambio en la tabla "alumnos":

```
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil", $conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("update alumnos
                        set codigocurso=$_REQUEST[codigocurso]
                        where mail='$_REQUEST[mailviejo]'", $conexion) or
    die("Problemas en el select:".mysql_error());
echo "El curso fue modificado con éxito";
?>
</body>
```

</html>

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese el mail del alumno:
<input type="text" name="mail"><br>
<input type="submit" value="buscar">
</form>
</body>
</html>
```

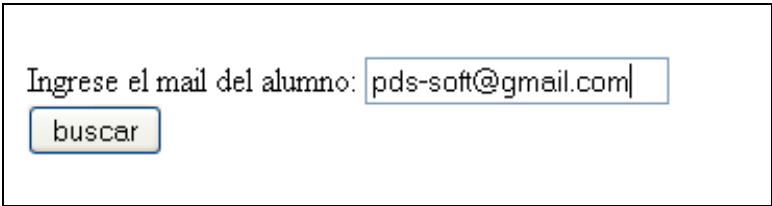
pagina2.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select * from alumnos
                        where mail='$_REQUEST[mail]'", $conexion) or
    die("Problemas en el select:".mysql_error());
if ($regalu=mysql_fetch_array($registros))
{
    ?>
    <form action="pagina3.php" method="post">
    <input type="hidden" name="mailviejo" value="<?php echo $regalu['mail'] ?>">
    <select name="codigocurso">
    <?php
    $registros=mysql_query("select * from cursos", $conexion) or
        die("Problemas en el select:".mysql_error());
    while ($reg=mysql_fetch_array($registros))
    {
        if ($regalu['codigocurso']==$reg['codigo'])
            echo "<option value=\"\$reg[codigo]\" selected>$reg[nombrecur]</option>";
        else
            echo "<option value=\"\$reg[codigo]\">$reg[nombrecur]</option>";
    }
    ?>
    </select>
```

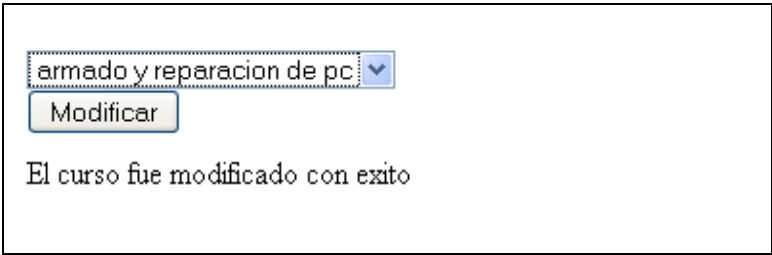
```
<br>
<input type="submit" value="Modificar">
</form>
<?php
}
else
    echo "No existe alumno con dicho mail";
?>
</body>
</html>
```

pagina3.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("update alumnos
                        set codigocurso=$_REQUEST[codigocurso]
                        where mail='$_REQUEST[mailviejo]'",$conexion) or
    die("Problemas en el select:".mysql_error());
echo "El curso fue modificado con exito";
?>
</body>
</html>
```



Ingresa el mail del alumno:



El curso fue modificado con exito

Confeccionar la modificación del mail, nombre y curso de la tabla "alumnos". Ingresar por teclado el código de alumno para su búsqueda.

Cláusula Group By de SQL.

Entre las muchas posibilidades que nos brinda SQL, una es agrupar registros y obtener información resumida de tablas. En nuestro problema, un listado interesante sería mostrar la cantidad de alumnos inscriptos por curso. Para resolver de una manera sencilla esta situación, SQL nos permite agrupar los registros de la tabla "alumnos" por la columna "codigocurso" y contar la cantidad de registros que hay por cada código de curso igual. El programa que nos permite resolver este problema es el siguiente:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select count(alu.codigo) as cantidad,
                        nombrecur from alumnos as alu
                        inner join cursos as cur on
                        cur.codigo=alu.codigocurso
                        group by alu.codigocurso", $conexion) or
    die("Problemas en el select:".mysql_error());
while ($reg=mysql_fetch_array($registros))
{
    echo "Nombre del curso:".$reg['nombrecur']."<br>";
    echo "Cantidad de inscriptos:".$reg['cantidad']."<br>";
    echo "<hr>";
}
mysql_close($conexion);
?>
</body>
</html>
```

Hay varias partes importantes en este código; primero, en el select indicamos que cuente la cantidad de registros de la tabla "alumnos":

```
"select count(alu.codigo) as cantidad,
                        nombrecur from alumnos as alu
```

Pero, como veremos más adelante, en 'cantidad' no se almacena la cantidad total de registros de la tabla "alumnos" debido a que más adelante empleamos la cláusula group by. Como necesitamos rescatar el nombre del curso hacemos el apareo con la tabla "cursos":

```
inner join cursos as cur on cur.codigo=alu.codigocurso
```

Por último en la sentencia select de SQL disponemos la cláusula group by:

```
group by alu.codigocurso"
```

Con esta cláusula se hace un corte de control por cada grupo de registros que tienen el mismo código de curso almacenado.

Luego mostramos el nombre de curso con la cantidad de inscriptos para dicho curso:

```
while ($reg=mysql_fetch_array($registros))
{
    echo "Nombre del curso:".$reg['nombrecur']."<br>";
    echo "Cantidad de inscriptos:".$reg['cantidad']."<br>";
    echo "<hr>";
}
```

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select count(alu.codigo) as
    cantidad,nombrecur from alumnos as alu
    inner join cursos as cur on cur.codigo=alu.codigocurso
    group by alu.codigocurso",
    $conexion) or
    die("Problemas en el select:".mysql_error());
while ($reg=mysql_fetch_array($registros))
{
    echo "Nombre del curso:".$reg['nombrecur']."<br>";
    echo "Cantidad de inscriptos:".$reg['cantidad']."<br>";
    echo "<hr>";
}
mysql_close($conexion);
?>
</body>
</html>
```

Nombre del curso:armado y reparacion de pc
Cantidad de inscriptos:1

Confeccionar un programa que muestre el nombre del curso, la cantidad de inscriptos y todos los inscriptos a dicho curso. Repetir esto con todos los cursos. Es decir, en la página debe aparecer algo parecido a:

Nombre	del	curso:PHP
Cantidad	de	inscriptos:3
Nombres:	Martinez Luis - Rodriguez Pablo - Gonzalez Ana	

Nombre	del	curso:JSP
Cantidad	de	inscriptos:2
Nombres:	Hernandez Hector - Roca Marta	

Parámetros en un hipervínculo.

Hasta ahora hemos visto como enviar datos de una página a otra mediante formularios. Otra forma muy utilizada que complementa la anterior es como parámetro en un hipervínculo.

Confeccionaremos una página que muestre tres hipervínculos, cada uno tiene por objetivo mostrar en una página las tablas de multiplicar del 2, del 3 o la del 5. La primer página es un archivo HTML puro, ya que sólo disponemos las marcas de hipervínculos:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<A href="pagina2.php?tabla=2">Tabla del 2</A> <br>
<A href="pagina2.php?tabla=3">Tabla del 3</A> <br>
<A href="pagina2.php?tabla=5">Tabla del 5</A>
</body>
</html>
```

La sintaxis para pasar parámetros en un hipervínculo es:

```
<A href="pagina2.php?tabla=2">Tabla del 2</A> <br>
```

Es decir, luego del caracter "?" indicamos el nombre del parámetro y seguidamente el valor del mismo.

La página que rescata el valor pasado como parámetro es la siguiente:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
echo "Listado de la tabla del $_REQUEST[tabla] <br>";
for($f=1;$f<=10;$f++)
{
$valores=$f*$_REQUEST['tabla'];
echo $valores."-";
}
?>
</body>
</html>
```

Es decir que con el mismo vector asociativo \$_REQUEST recuperamos los datos enviados por parámetro en la llamada a la página.

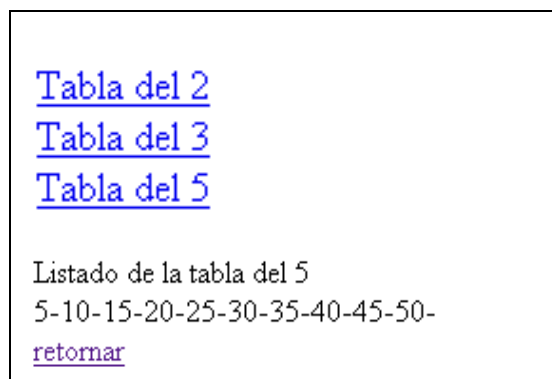
pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<A href="pagina2.php?tabla=2">Tabla del 2</A> <br>
<A href="pagina2.php?tabla=3">Tabla del 3</A> <br>
<A href="pagina2.php?tabla=5">Tabla del 5</A>
</body>
```

</html>

pagina2.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
echo "Listado de la tabla del $_REQUEST[tapa] <br>";
for($f=1;$f<=10;$f++)
{
    $valor=$f*$_REQUEST['tapa'];
    echo $valor."-";
}
?>
<br>
<a href="pagina1.php">retornar</a>
</body>
</html>
```



Confeccionar un programa que muestre una página con todos los nombres de la tabla "cursos". Los nombres deben aparecer como hipervínculos a otra página que deberá mostrar todos los alumnos inscriptos a dicho curso. Como parámetro en el hipervínculo, pasar el código de curso.

Paginación de registros.

En situaciones en las cuales una consulta retorna muy muchos datos, en vez de enviarlos todos al navegador, se puede enviar un conjunto limitado de registros. Luego, mediante hipervínculos, ver el resto de datos. Por ejemplo, cuando hacemos búsquedas con el servidor google, generalmente no nos retorna todas las direcciones donde se encuentran los resultados buscados, nos retorna páginas con 10 enlaces por página (pensemos el tiempo de transferencia si nos retornara 1.000.000 de enlaces). Bueno, ahora resolvamos con el lenguaje PHP este problema de paginación:

```
<?php
```

```
if (isset($_REQUEST['pos']))
    $inicio=$_REQUEST['pos'];
else
    $inicio=0;
?>
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select alu.codigo as
                        codigo,nombre,mail,codigocurso,
                        nombrecur from alumnos as alu
                        inner join cursos as cur on
cur.codigo=alu.codigocurso
                        limit $inicio,2", $conexion) or
    die("Problemas en el select:".mysql_error());
$impresos=0;
while ($reg=mysql_fetch_array($registros))
{
    $impresos++;
    echo "Codigo:".$reg['codigo']."<br>";
    echo "Nombre:".$reg['nombre']."<br>";
    echo "Mail:".$reg['mail']."<br>";
    echo "Curso:".$reg['nombrecur']."<br>";
    echo "<hr>";
}
mysql_close($conexion);
if ($inicio==0)
    echo "anteriores ";
else
{
    $anterior=$inicio-2;
    echo "<a href=\"paginal.php?pos=$anterior\">Anteriores </a>";
}
if ($impresos==2)
{
    $proximo=$inicio+2;
    echo "<a href=\"paginal.php?pos=$proximo\">Siguientes</a>";
}
else
    echo "siguientes";
?>
</body>
</html>
```

Hay muchas cosas importantes en este ejemplo, lo primero que vemos es el bloque que rescata a partir de qué registro ir mostrando:

```
if (isset($_REQUEST['pos']))
    $inicio=$_REQUEST['pos'];
else
    $inicio=0;
?>
```

La función `isset` retorna verdadero si existe la variable que le pasamos como parámetro, en este caso le estamos pasando la componente `pos` del vector asociativo `$_REQUEST`. Cuando llamamos por primera vez a esta página, lo hacemos : `paginal.php` sin parámetros,

por lo que el if se verifica como falso. Es decir la variable \$inicio se carga con el valor 0.

Otro concepto importante es la cláusula limit que es propiedad del gestor MySQL. Mediante esta cláusula limitamos la cantidad de registros que retorna el select. El primer valor del limit indica a partir de cual registro y el segundo la cantidad de registros. Es decir si un select sin limit retorna 100 registro, luego utilizando por ejemplo la sintaxis limit 50,25 nos retornará, de esa lista de 100 registros, a partir del registro de la posición 50, 25 registros.

En nuestro problema indicamos que retorne desde valor que tenga la variable \$inicio y como cantidad 2 (páginas con 2 registros):

```
$registros=mysql_query("select alu.codigo as
                        codigo,nombre,mail,codigocurso,
                        nombrecur from alumnos as alu
                        inner      join      cursos      as      cur      on
cur.codigo=alu.codigocurso
                        limit $inicio,2", $conexion) or
```

Seguidamente mostramos todos los registros retornados y además los contamos:

```
$impresos=0;
while ($reg=mysql_fetch_array($registros))
{
    $impresos++;
    echo "Codigo:".$reg['codigo']."<br>";
    echo "Nombre:".$reg['nombre']."<br>";
    echo "Mail:".$reg['mail']."<br>";
    echo "Curso:".$reg['nombrecur']."<br>";
    echo "<hr>";
}
```

Ahora vemos dónde dispondremos los hipervínculos, hacia adelante o atrás:

```
if ($inicio==0)
    echo "anteriores ";
else
{
    $anterior=$inicio-2;
    echo "<a href=\"paginal.php?pos=$anterior\">Anteriores </a>";
}
```

Si la variable \$inicio vale 0 significa que no hay registros antes de éste, por lo que sólo mostramos un texto "anteriores". En caso que la variable \$inicio sea distinta de 0, se ejecuta el else, donde disponemos un hipervínculo con la misma página e inicializando el parámetro pos con el valor de \$inicio menos 2.

Si el contador \$impresos tiene el valor 2 significa que posiblemente hay más registros por mostrar y debemos disponer un hipervínculo con la misma página pero inicializando el parámetro pos con el valor de \$inicio más 2:

```
if ($impresos==2)
{
    $proximo=$inicio+2;
    echo "<a href=\"paginal.php?pos=$proximo\">Siguientes</a>";
}
else
    echo "siguientes";
```

paginal.php

```
<?php
if (isset($_REQUEST['pos']))
    $inicio=$_REQUEST['pos'];
```

```
else
    $inicio=0;
?>
<html>
<head>
<title>Problema</title>
</head>
<body>

<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select alu.codigo as
    codigo,nombre,mail,codigocurso,
    nombrecur from alumnos as alu
    inner join cursos as cur on cur.codigo=alu.codigocurso
    limit $inicio,2",
    $conexion) or
    die("Problemas en el select:".mysql_error());
$impresos=0;
while ($reg=mysql_fetch_array($registros))
{
    $impresos++;
    echo "Codigo:".$reg['codigo']."<br>";
    echo "Nombre:".$reg['nombre']."<br>";
    echo "Mail:".$reg['mail']."<br>";
    echo "Curso:".$reg['nombrecur']."<br>";
    echo "<hr>";
}
mysql_close($conexion);
if ($inicio==0)
    echo "anteriores ";
else
{
    $anterior=$inicio-2;
    echo "<a href=\"pagina1.php?pos=$anterior\">Anteriores </a>";
}
if ($impresos==2)
{
    $proximo=$inicio+2;
    echo "<a href=\"pagina1.php?pos=$proximo\">Siguietes</a>";
}
else
    echo "siguietes";
?>

</body>
</html>
```

Codigo:55587
Nombre:Paul
Mail:pds-soft@gmail.com
Curso:armado y reparacion de pc

[anteriores](#) [siguientes](#)

Confeccionar un programa que muestre los registros de la tabla "cursos" con páginas de 3 registros.

Subir un archivo al servidor (Upload)

Una actividad común en un sitio es poder almacenar un archivo en el servidor, más comunmente conocido como upload. Se necesita en muchas ocasiones este algoritmo, por ejemplo para subir fotos, documentos, programas, etc. Se requieren dos páginas, una de ellas, un formulario donde seleccionamos el archivo a enviar y otra página donde se graba el archivo en el servidor. El formulario para hacer la selección del archivo es:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post" enctype="multipart/form-data">
Seleccione el archivo:
<input type="file" name="foto"><br>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

Veamos los puntos que tenemos que respetar cuando efectuamos el upload de archivos:

```
<form action="pagina2.php" method="post" enctype="multipart/form-data">
```

Se define una nueva propiedad para la marca form, con esta indicamos que dentro del formulario de carga se envían archivos. Hay que tener mucho cuidado cuando tipeamos esta propiedad, si nos equivocamos en algún carácter el proceso de upload no funcionará. El control HTML para la selección del archivo se llama "file":

```
<input type="file" name="foto"><br>
```

Automáticamente aparecerá el botón dentro de la página para poder navegar en nuestro disco duro para la selección del archivo (por defecto PHP está configurado para poder cargar archivos de hasta 2 Mb, de todos modos, a este valor lo podemos modificar).

La segunda página es:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
```

```
copy($_FILES['foto']['tmp_name'], $_FILES['foto']['name']);
echo "La foto se registro en el servidor.<br>";
$nom=$_FILES['foto']['name'];
echo "<img src=\"\$nom\">";
?>
</body>
</html>
```

Cuando se ejecuta esta página, ya está almacenado en el servidor el archivo, en una carpeta temporal. Ahora nos hace falta copiar el mismo a la carpeta donde se encuentra nuestra página (en definitiva nuestro sitio de internet). Para esto llamamos a la función copy:

```
copy($_FILES['foto']['tmp_name'], $_FILES['foto']['name']);
```

La matriz \$_FILES almacena el nombre del archivo almacenado en el directorio temporal (\$_FILES['foto']['tmp_name']) y el nombre del archivo originario (\$_FILES['foto']['name'])

Por último mostramos en la página el archivo que se almacenó en el servidor:

```
$nom=$_FILES['foto']['name'];
echo "<img src=\"\$nom\">";
```

Otras cosas interesantes que tiene la matriz \$_FILES:
\$_FILES['foto']['name'] El nombre original del fichero en la máquina cliente.
\$_FILES['foto']['type'] El tipo de archivo (si el navegador lo proporciona). Un ejemplo podría ser "image/gif".
\$_FILES['foto']['size'] El tamaño en bytes del fichero recibido.
\$_FILES['foto']['tmp_name'] El nombre del archivo temporal que se utiliza para almacenar en el servidor el archivo recibido.

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post" enctype="multipart/form-data">
Seleccione el archivo:
<input type="file" name="foto"><br>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

pagina2.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>

<?php
copy($_FILES['foto']['tmp_name'], $_FILES['foto']['name']);
echo "La foto se registro en el servidor.<br>";
```

```
$nom=$_FILES['foto']['name'];  
echo "<img src=\"\$nom\">";  
?>
```

```
</body>  
</html>
```

Confeccionar un programa que permita hacer multiples upload con una página (por ejemplo que permita seleccionar hasta 3 archivos)

Creación y lectura de una cookie

El protocolo HTTP es desconectado. Esto significa que cada vez que solicitamos una página a un servidor representa una conexión distinta. Una cookie es una pequeña cantidad de datos almacenada por el navegador del usuario cuando solicita una página a un servidor. El que envía que se genere la cookie es el servidor.

Una cookie consta de un nombre, un valor, una fecha de expiración y un servidor. Una cookie está limitada a 4KB.

Luego que una cookie es creada sólo el sitio que la creó puede leerla. Luego de creada una cookie, cada vez que el navegador del usuario visita el sitio, se envía dicha cookie. Otra cosa importante que hay que tener en cuenta es que el usuario del browser puede configurar el mismo para no permitir la creación de cookies, lo que significa que el uso de cookies debe hacerse con moderación y cuando la situación lo requiera. De todos modos, el 95% de los navegadores están configurados para permitir la creación de cookies. Para la creación de una cookie desde PHP debemos llamar a la función setcookie. Los parámetros de esta función son: setcookie(<nombre de la cookie>, <valor de la cookie>, <fecha de expiración>, <carpeta del servidor>)

Con un problema sencillo entenderemos el uso de esta función. Supongamos que queremos que los usuarios que entran a nuestro sitio puedan configurar con qué color de fondo de página quiere que aparezca cada vez que ingresa al sitio. Al color seleccionado por el visitante lo almacenaremos en una cookie. En caso que no exista el color, por defecto es blanco.

La primera página mostrará un formulario con tres controles de tipo radio para la selección del color. También esta página verificará si existe la cookie creada, en caso afirmativo fijará el fondo de la página con el valor de la cookie. Tengamos en cuenta que la primera vez que ejecutemos este programa la página es de color blanco, luego variará según el color seleccionado en el formulario. El código de la primera página es:

```
<html>  
<head>  
<title>Problema</title>  
</head>  
<body>  
<?php if (isset($_COOKIE['color'])) echo " bgcolor=\"$_COOKIE[color]\""  
>  
>  
<form action="pagina2.php" method="post">
```



```
Seleccione de que color desea que sea la página de ahora en más:<br>
<input type="radio" value="rojo" name="radio">Rojo<br>
<input type="radio" value="verde" name="radio">Verde<br>
<input type="radio" value="azul" name="radio">Azul<br>
<input type="submit" value="Crear cookie">
</form>
</body>
</html>
```

El formulario no varía en nada respecto a otros vistos. Lo más importante es el bloque PHP que verifica si ya existe la cookie en el navegador del cliente. Es importante entender que la primera vez que ejecutemos esta página la cookie no existe, por lo que el if se verifica falso:

```
<body
<?php if (isset($_COOKIE['color'])) echo " bgcolor=\"$_COOKIE[color]\""
?>
>
```

El vector asociativo `$_COOKIE` almacena todas las cookies creadas por el visitante. Si es la primera vez que petitionamos esta página, el vector `$_COOKIE` no tendrá elementos. Es decir que la marca body no tiene inicializada la propiedad bgcolor.

La segunda página es la que crea la cookie propiamente dicha:

```
<?php
if ($_REQUEST['radio']=="rojo")
    setcookie("color", "#ff0000", time()+60*60*24*365, "/");
elseif ($_REQUEST['radio']=="verde")
    setcookie("color", "#00ff00", time()+60*60*24*365, "/");
elseif ($_REQUEST['radio']=="azul")
    setcookie("color", "#0000ff", time()+60*60*24*365, "/");
?>
<html>
<head>
<title>Problema</title>
</head>
<body>
Se creó la cookie.
<br>
<a href="paginal.php">Ir a la otra página</a>
</body>
</html>
```

La llamada a la función `setcookie` debe hacerse antes de imprimir cualquier marca HTML, de lo contrario no funcionará. Como podemos observar, la creación de la cookie se hace llamando a la función `setcookie`:

```
<?php
if ($_REQUEST['radio']=="rojo")
    setcookie("color", "#ff0000", time()+60*60*24*365, "/");
elseif ($_REQUEST['radio']=="verde")
    setcookie("color", "#00ff00", time()+60*60*24*365, "/");
elseif ($_REQUEST['radio']=="azul")
    setcookie("color", "#0000ff", time()+60*60*24*365, "/");
?>
```

El nombre de la cookie se llama "color" y el valor que almacenamos depende de qué control de tipo radio esté seleccionado en la página anterior. La fecha de expiración de la cookie la calculamos fácilmente llamando a la función `time()` que nos retorna la fecha actual en segundos y le sumamos el producto $60 \times 60 \times 24 \times 365$ (60 segundos * 60 minutos * 24 horas * 365 días) es decir que la cookie existirá en la máquina del visitante hasta el año

próximo.

Cuando indicamos como directorio la sintaxis "/" significa que la cookie se crea a nivel del sitio y con cualquier petición a dicho sitio, el navegador enviará la cookie al servidor. Por último dispusimos en esta página un hipervínculo a la página anterior, para ver que, de ahora en más, cada vez que ejecutemos la pagina1.php, el color de fondo de la misma dependerá del valor de la cookie registrada.

pagina1.php

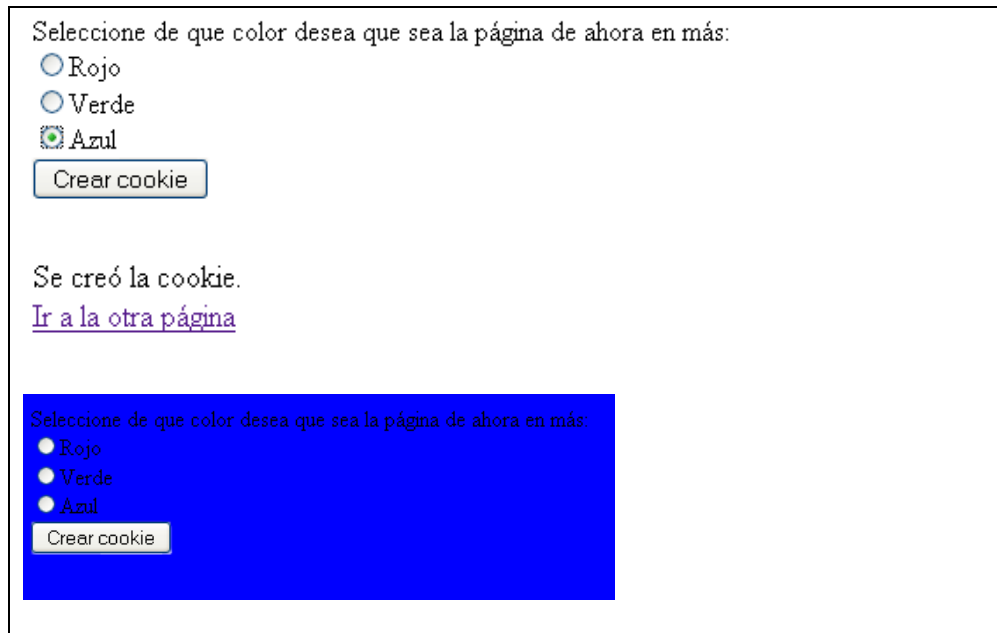
```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php if (isset($_COOKIE['color'])) echo " bgcolor=\"$_COOKIE[color]\"" ?>
>
<form action="pagina2.php" method="post">
Seleccione de que color desea que sea la página de ahora en más:<br>
<input type="radio" value="rojo" name="radio">Rojo<br>
<input type="radio" value="verde" name="radio">Verde<br>
<input type="radio" value="azul" name="radio">Azul<br>
<input type="submit" value="Crear cookie">
</form>
</body>
</html>
```

pagina2.php

```
<?php
if ($_REQUEST['radio']=="rojo")
    setcookie("color", "#ff0000", time()+60*60*24*365, "/");
elseif ($_REQUEST['radio']=="verde")
    setcookie("color", "#00ff00", time()+60*60*24*365, "/");
elseif ($_REQUEST['radio']=="azul")
    setcookie("color", "#0000ff", time()+60*60*24*365, "/");
?>
<html>
<head>
<title>Problema</title>
</head>
<body>

Se creó la cookie.
<br>
<a href="pagina1.php">Ir a la otra página</a>

</body>
</html>
```



Seleccione de que color desea que sea la página de ahora en más:

☐ Rojo

☐ Verde

☒ Azul

Se creó la cookie.

[Ir a la otra página](#)

Seleccione de que color desea que sea la página de ahora en más:

☐ Rojo

☐ Verde

☐ Azul

Crear una cookie que almacene el nombre del visitante al sitio y cada vez que ingresemos al sitio nos de la bienvenida imprimiendo nuestro nombre. Para cargar el nombre crear un formulario con un control de tipo text.

Borrado de una cookie

Para borrar una cookie se debe llamar a la función setcookie con una fecha anterior a la actual.

Haremos un algoritmo muy común a muchos sitios que administran webmail. Recordaremos en una cookie el mail ingresado por el operador, el código fuente de la primera página es la siguiente:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingresa su mail:
<input type="text" name="mailusuario"
value="<?php if (isset($_COOKIE['mail'])) echo $_COOKIE['mail'];?>">
<br>
<input type="radio" name="opcion" value="recordar">
Recordar en esta computadora el mail ingresado.
<br>
<input type="radio" name="opcion" value="norecordar">
No recordar.
<br>
<input type="submit" value="confirmar">
</form>
</body>
</html>
```

Es decir, disponemos un control text y como valor de la misma verificamos si el cliente tiene ya una cookie guardada en su máquina, en caso afirmativo la mostramos dentro del

text modificando la propiedad value.
Mediante dos controles de tipo radio daremos la posibilidad al cliente que el navegador recuerde o no el mail ingresado. Como es sabido, la primera vez que accedamos a la página no existe la cookie llamada mail.

La segunda página:

```
<?php
if ($_REQUEST['opcion']=="recordar")
    setcookie("mail",$_REQUEST['mailusuario'],time()+(60*60*24*365),"/");
elseif ($_REQUEST['opcion']=="norecordar")
    setcookie("mail","",time()-1000,"/");
?>
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
if ($_REQUEST['opcion']=="recordar")
    echo "cookie creada";
elseif ($_REQUEST['opcion']=="norecordar")
    echo "cookie eliminada";
?>
<br>
<a href="paginal.php">Ir a la otra página</a>
</body>
</html>
```

En esta página verificamos cuál control radio está seleccionado, si es el primero, creamos la cookie mail con una vida de 1 año:

```
setcookie("mail",$_REQUEST['mailusuario'],time()+(60*60*24*365),"/");
```

En caso que esté seleccionado el segundo radio, creamos la misma cookie pero con una fecha de caducidad inferior a la actual (con esto estamos virtualmente eliminando la cookie):

```
setcookie("mail","",time()-1000,"/");
```

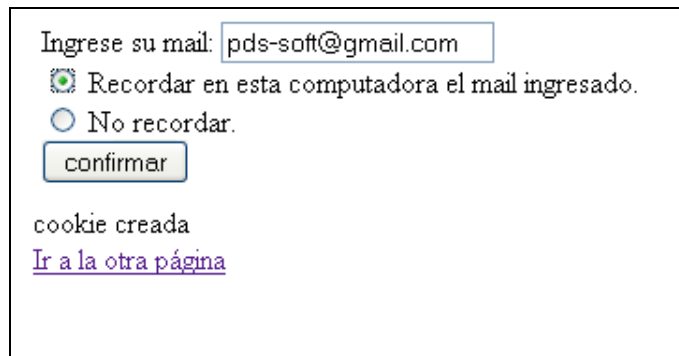
paginal.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese su mail:
<input type="text" name="mailusuario"
value="<?php if (isset($_COOKIE['mail'])) echo $_COOKIE['mail'];?>">
<br>
<input type="radio" name="opcion" value="recordar">
Recordar en esta computadora el mail ingresado.
<br>
<input type="radio" name="opcion" value="norecordar">
No recordar.
<br>
```

```
<input type="submit" value="confirmar">
</form>
</body>
</html>
```

pagina2.php

```
<?php
if ($_REQUEST['opcion']=="recordar")
    setcookie("mail",$_REQUEST['mailusuario'],time()+(60*60*24*365),"/");
elseif ($_REQUEST['opcion']=="norecordar")
    setcookie("mail","",time()-1000,"/");
?>
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
if ($_REQUEST['opcion']=="recordar")
    echo "cookie creada";
elseif ($_REQUEST['opcion']=="norecordar")
    echo "cookie eliminada";
?>
<br>
<a href="pagina1.php">Ir a la otra página</a>
</body>
</html>
```



Confeccionar una página que simule ser la de un periódico. La misma debe permitir configurar qué tipo de titular deseamos que aparezca al visitarla, pudiendo ser: Noticia política, Noticia económica o Noticia deportiva. Mediante tres objetos de tipo radio, permitir seleccionar qué titular debe mostrar el periódico. Almacenar en una cookie el tipo de titular que desea ver el cliente. La primera vez que visita el sitio deben aparecer los tres titulares.

Cookie de sesión

Para crear una cookie que sólo tenga existencia mientras no cerremos la ventana del navegador, pasaremos como fecha de expiración de la cookie, el valor cero. Una vez que la instancia del navegador se cierra, dicha cookie desaparecerá. Este tipo de cookie puede ser muy útil para validar un usuario en un conjunto de páginas, si previamente ingresó correctamente su nombre de usuario y clave. Es decir, una vez validado el usuario, se verifica en páginas sucesivas si existe la cookie. Una vez que el usuario cierra el navegador, no hay posibilidad de solicitar las páginas recorridas sin previa validación nuevamente de clave y usuario. Entonces la sintaxis es:

```
setcookie(,0)
```

Veamos un pequeño ejemplo para crear y verificar si existe una cookie de sesión. La primera página es:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
if (isset($_COOKIE['usuario']))
echo "Cookie de sesión creada. Su valor es $_COOKIE[usuario]";
else
echo "No existe cookie de sesión";
?>
<br>
<a href="pagina2.php">Crear cookie de sesión</a>
</body>
</html>
```

La segunda página es la que crea la cookie de sesión:

```
<?php
setcookie("usuario","diego",0);
?>
<html>
<head>
<title>Problema</title>
</head>
<body>
Cookie de sesión creada.<br>
<a href="pagina1.php">Retornar a la página anterior.</a>
</body>
</html>
```

Si ejecutamos este programa y creamos la cookie de sesión, la misma existirá mientras no cerremos el navegador. Probemos luego cerrando completamente el navegador y veremos qué contiene la cookie de sesión.

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
if (isset($_COOKIE['usuario']))
```

```
    echo "Cookie de sesión creada. Su valor es $_COOKIE[usuario]";
else
    echo "No existe cookie de sesión";
?>
<br>
<a href="pagina2.php">Crear cookie de sesión</a>
</body>
</html>
```

pagina2.php

```
<?php
setcookie("usuario","diego",0);
?>
<html>
<head>
<title>Problema</title>
</head>
<body>
Cookie de sesión creada.<br>
<a href="pagina1.php">Retornar a la página anterior.</a>
</body>
</html>
```

No existe cookie de sesión
[Crear cookie de sesión](#)

Cookie de sesión creada.
[Retornar a la página anterior.](#)

Cookie de sesión creada. Su valor es diego
[Crear cookie de sesión](#)

Variables de sesión (\$_SESSION)

Es otro método para hacer que variables estén disponibles en múltiples páginas sin tener que pasarlas como parámetro. A diferencia de las cookies, las variables de sesión se almacenan en el servidor y tienen un tiempo limitado de existencia. Para identificar al usuario que generó las variables de sesión, el servidor genera una clave única que es enviada al navegador y almacenada en una cookie. Luego, cada vez que el navegador solicita otra página al mismo sitio, envía esta cookie (clave única) con la cual el servidor identifica de qué navegador proviene la petición y puede rescatar de un archivo de texto las variables de sesión que se han creado. Cuando han pasado 20 minutos sin peticiones por parte de un cliente (navegador) las variables de sesión son eliminadas automáticamente (se puede configurar el entorno de PHP para variar este tiempo).

Una variable de sesión es más segura que una cookie ya que se almacena en el servidor. Otra ventaja es que no tiene que estar enviándose continuamente como sucede con las cookies. Otra ventaja de emplear una variable de sesión en lugar de una cookie es que cuando el navegador del cliente está configurado para desactivar las cookies las variables de sesión, tienen forma de funcionar (enviando la clave como parámetro en cada hipervínculo).

Como desventaja podemos decir que ocupa espacio en el servidor.

Haremos un problema muy sencillo, cargaremos en un formulario el nombre de usuario y clave de un cliente, en la segunda página crearemos dos variables de sesión y en una tercera página recuperaremos los valores almacenados en las variables de sesión. La primera página es un formulario HTML puro:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese nombre de usuario:
<input type="text" name="campousuario"><br>
Ingrese clave:
<input type="password" name="campoclave"><br>
<input type="submit" value="confirmar">
</form>
</body>
</html>
```

Lo que podemos recalcar es que, cuando en un cuadro de texto queremos ingresar una clave y no queremos que aparezcan los caracteres tipeados en pantalla, debemos utilizar el control de tipo password:

```
<input type="password" name="campoclave">
```

La segunda página es donde creamos e inicializamos las dos variables de sesión:

```
<?php
session_start();
$_SESSION['usuario']=$_REQUEST['campousuario'];
$_SESSION['clave']=$_REQUEST['campoclave'];
?>
<html>
<head>
<title>Problema</title>
</head>
<body>
Se almacenaron dos variables de sesión.<br><br>
<a href="pagina3.php">Ir a la tercer página donde se recuperarán
las variables de sesión</a>
</body>
</html>
```

Cuando creamos o accedemos al contenido de variables de sesión debemos llamar a la función `session_start()` antes de cualquier salida de marcas HTML. Para almacenar los valores en las variables de sesión lo hacemos:

```
$_SESSION['usuario']=$_REQUEST['campousuario'];
$_SESSION['clave']=$_REQUEST['campoclave'];
```

Es decir, tenemos el vector asociativo `$_SESSION` que almacena las variables de sesión. Por último, esta página tiene un hipervínculo a la tercera página.

La última página de este ejemplo tiene por objetivo acceder a las variables de sesión:

```
<?php
session_start();
?>
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
echo "Nombre de usuario recuperado de la variable de
sesión:".$_SESSION['usuario'];
echo "<br><br>";
echo "La clave recuperada de la variable de sesión:".$_SESSION['clave'];
?>
</body>
</html>
```

De nuevo vemos que la primera línea de esta página es la llamada a la función `session_start()` que, entre otras cosas, rescata de un archivo de texto las variables de sesión creadas para ese usuario (recordemos que desde el navegador todas las veces retorna una cookie con la clave que generó PHP la primera vez que llamamos a una página del sitio). Para mostrar las variables de sesión, las accedemos por medio del vector asociativo `$_SESSION`:

```
echo "Nombre de usuario recuperado de la variable de
sesión:".$_SESSION['usuario'];
echo "<br><br>";
echo "La clave recuperada de la variable de sesión:".$_SESSION['clave'];
```

Tengamos en cuenta que en cualquier otra página del sitio tenemos acceso a las variables de sesión sólo con llamar inicialmente a la función `session_start()`.

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese nombre de usuario:
<input type="text" name="campousuario"><br>
Ingrese clave:
<input type="password" name="campoclave"><br>
<input type="submit" value="confirmar">
</form>
</body>
</html>
```

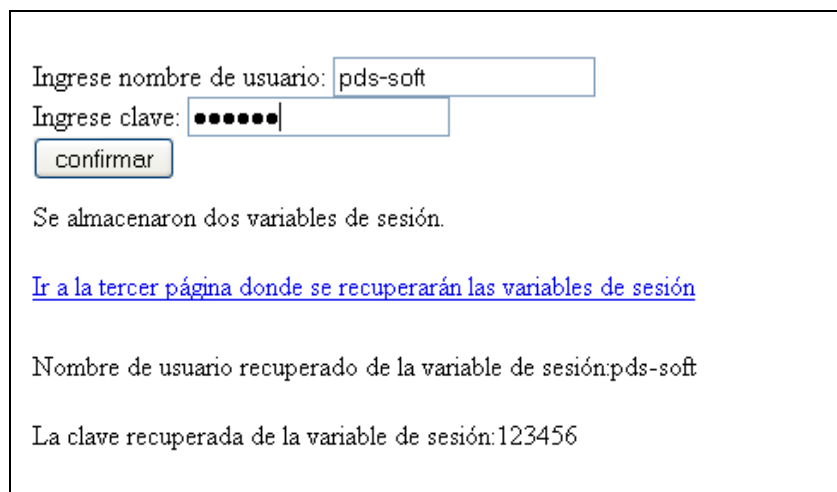
pagina2.php

```
<?php
session_start();
$_SESSION['usuario']=$_REQUEST['campousuario'];
$_SESSION['clave']=$_REQUEST['campoclave'];
```

```
?>
<html>
<head>
<title>Problema</title>
</head>
<body>
Se almacenaron dos variables de sesión.<br><br>
<a href="pagina3.php">Ir a la tercer página donde se recuperarán las variables de
sesión</a>
</body>
</html>
```

pagina3.php

```
<?php
session_start();
?>
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
echo "Nombre de usuario recuperado de la variable de
sesión:".$_SESSION['usuario'];
echo "<br><br>";
echo "La clave recuperada de la variable de sesión:".$_SESSION['clave'];
?>
</body>
</html>
```



The screenshot shows a web page with a login form. The form has two input fields: "Ingrese nombre de usuario:" with the value "pds-soft" and "Ingrese clave:" with masked characters "••••••". Below the inputs is a "confirmar" button. The page content below the form states: "Se almacenaron dos variables de sesión." followed by a blue hyperlink "Ir a la tercer página donde se recuperarán las variables de sesión". At the bottom, it displays the retrieved session data: "Nombre de usuario recuperado de la variable de sesión:pds-soft" and "La clave recuperada de la variable de sesión:123456".

Confeccionar un formulario que solicite ingresar el mail de un alumno. Si el mail existe en la tabla alumnos, rescatar su nombre y almacenarlo en una variable de sesión. Además disponer un hipervínculo a una tercera página que verifique si existe la variable de sesión y de la bienvenida al alumno, en caso contrario mostrar un mensaje indicando que no puede visitar esta página (para saber si una variable de sesión está definida llamamos a la función

```
isset)
if (isset($_SESSION['nombre'])) ....
```

Incluir un archivo externo (require_once)

Hasta hora hemos visto que un archivo contiene todo el código (HTML y PHP), pero el lenguaje PHP nos permite crear librerías de funciones u objetos que veremos más adelante. La idea fundamental de las librerías es agrupar funciones comunes a muchas páginas, no tener que tipearlas en cada archivo, lo que supone que cuando haya que hacer cambios en esas funciones las debemos localizar y modificar y dicha modificación afectará a todos los archivos donde se las utiliza.

La implementación de librerías nos permite hacer que un sistema sea más modular y facilita su mantenimiento. Para probar esta característica del lenguaje, implementaremos dos funciones en la página "pagina2.php" y llamaremos a dichas funciones en la "pagina1.php"

El código del archivo "pagina1.php" es:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
require_once("pagina2.php");
cabeceraPagina("Titulo principal de la página");
echo "<br><br><center>Este es el cuerpo de la página<br><br></center>";
piePagina("Pie de la página");
?>
</body>
</html>
```

Para incluir el contenido de otro archivo, debemos llamar a la función `require_once` pasando como parámetro el archivo a incluir:

```
require_once("pagina2.php");
```

Si el archivo no existe, se mostrará un error y no continuará la ejecución del programa PHP.

El programa continúa llamando a una función que no se encuentra implementada en este archivo sino que está en el archivo "pagina2.php":

```
cabeceraPagina("Titulo principal de la página");
```

Luego de mostrar otros textos en la página, llamamos a una segunda función que también está implementada en el archivo `pagina2.php`:

```
piePagina("Pie de la página");
```

Hay que tener en cuenta que en un proyecto real, las funciones que almacena el archivo "pagina2.php" pueden ser incluidas y utilizadas en muchas otras páginas.

El código fuente del archivo "pagina2.php" es:

```
<?php
function cabeceraPagina($tit)
{
    echo "<table width=\"100%\"><tr><td bgcolor=\"#ffff00\"
    align=\"center\">$tit</td></tr></table>";
}
```

```
function piePagina($tit)
{
    echo "<table width=\"50%\" align=\"center\"><tr><td
    bgcolor=\"#cccccc\">$tit</td></tr></table>";
}
?>
```

Normalmente las rutinas se tratan de archivos PHP puros, es decir que contienen sólo funciones y no marcas HTML fijas. Es importante el lugar donde llamamos a la función `require_once`, debe ser siempre antes de la llamada a las funciones que contiene la librería.

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
require_once("pagina2.php");
cabeceraPagina("Titulo principal de la página");
echo "<br><br><center>Este es el cuerpo de la página<br><br></center>";
piePagina("Pie de la página");
?>
</body>
</html>
```

pagina2.php

```
<?php
function cabeceraPagina($tit)
{
    echo "<table width=\"100%\"><tr><td
    bgcolor=\"#ffff00\"
    align=\"center\">$tit</td></tr></table>";
}

function piePagina($tit)
{
    echo "<table width=\"50%\"
    align=\"center\"><tr><td
    bgcolor=\"#cccccc\">$tit</td></tr></table>";
}
?>
```

Confeccionar una librería que contenga una función llamada `retornarConexion`, la misma debe llamar a las funciones `mysql_connect` y `mysql_select_db` y retornar la variable que generó la función `mysql_connect`. Tener en cuenta que a esta función la deben implementar en el archivo "pagina2.php". En el archivo "pagina1.php" incluir la librería que contiene la función `retornarConexion`. Luego imprimir todos los registros de la tabla `alumnos`.

Redireccionar a otra página (header)

Con PHP se puede implementar un pequeño programa que permita que cuando un usuario (navegador) solicita una página, la misma redireccione o otra página. Para probar el algoritmo implementaremos un formulario que solicite el ingreso por teclado de una dirección de internet. La segunda página redireccionará al sitio web cargado en el formulario. Es decir la segunda página nunca llega al cliente, sino que redirecciona a otra página.

El primer archivo no tiene nada de especial:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese dirección de sitio web (ej www.google.com):
<input type="text" name="direccion" size="30"><br>
<input type="submit" value="Redireccionar">
</form>
</body>
</html>
```

El segundo archivo "pagina2.php" es el que efectúa la redirección a otra página o sitio:

```
<?php
header("Location: http://$_REQUEST[direccion]");
?>
```

La llamada a la función header debe hacerse antes de cualquier salida HTML, sino, no funcionará. Debemos pasarle como parámetro un string con el texto Location y la dirección del sitio y/o página a recuperar.

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese dirección de sitio web (ej www.google.com):
<input type="text" name="direccion" size="30"><br>
<input type="submit" value="Redireccionar">
</form>
</body>
</html>
```

pagina2.php

```
<?php
header("Location: http://$_REQUEST[direccion]");
?>
```

Ingrese dirección de sitio web (ej www.google.com):

Confeccionar un programa que solicite el ingreso de una clave en un formulario. La segunda página debe verificar si ingresó el string "z80" y mostrar un mensaje de bienvenida, en caso contrario, esta página debe redireccionar a la primera página nuevamente.

Cuando tenemos que redireccionar a una página que está en el mismo sitio, sólo es necesario disponer su nombre:

```
<?php
if ($_REQUEST['direccion']<>"z80")
header("Location: paginal.php");
?>
```

Otra cosa interesante que podemos hacer es pasar como parámetro en la segunda página un código de error:

```
<?php
if ($_REQUEST['direccion']<>"z80")
header("Location: paginal.php?error=1");
?>
```

Y luego en la primera página mostrar un mensaje de error si es que la página recibe este parámetro:

```
<?php
if (isset($_REQUEST['error']))
echo "Ingreso clave incorrecta";
?>
```

Creación de imágenes dinámicas desde PHP.

Existe un conjunto de funciones que nos permite la creación de un archivo de imagen (jpg, png, gif, etc.) en el servidor y posterior envío al navegador que la solicitó. Es decir que, con PHP, no sólo podemos hacer páginas dinámicas sino también imágenes dinámicas. Veamos un ejemplo sencillo y útil donde aplicar la generación de una imagen dinámica. Casi todos hemos visto que los sitios nos obligan a ingresar un código verificador para registrarnos a un servicio de internet. Generalmente es un gráfico con una serie de números y letras poco legible (esto se hace para confirmar que quien está ingresando los datos se trata de un ser humano y no un programa de computadora camuflado como persona). Este tipo de problema se adapta muy bien para emplear la creación de imágenes dinámicas. Veamos el código que nos permite resolver este problema:

```
<?php
$ancho=100;
$alto=30;
$imagen=imageCreate($ancho,$alto);
$amarillo=ImageColorAllocate($imagen,255,255,0);
ImageFill($imagen,0,0,$amarillo);
$rojo=ImageColorAllocate($imagen,255,0,0);
$valoraleatorio=rand(100000,999999);
ImageString($imagen,5,25,5,$valoraleatorio,$rojo);
for ($c=0;$c<=5;$c++)
```

```
{
    $x1=rand(0,$ancho);
    $y1=rand(0,$alto);
    $x2=rand(0,$ancho);
    $y2=rand(0,$alto);
    ImageLine($imagen,$x1,$y1,$x2,$y2,$rojo);
}
Header ("Content-type: image/jpeg");
ImageJPEG ($imagen);
ImageDestroy($imagen);
?>
```

Lo primero que hay que tener en cuenta es que el archivo es PHP puro, es decir no tiene salidas HTML, esto debido a que es una imagen la que se genera y no un archivo HTML. Luego veremos que a esta imagen se la puede incorporar en un archivo HTML y ser parte dentro de una marca IMG. Lo primero que hacemos es llamar a la función `imageCreate` pasando como parámetros el ancho y el alto de la imagen a crear:

```
$ancho=100;
$alto=30;
$imagen=imageCreate($ancho,$alto);
```

La función `imageCreate` retorna una referencia a la imagen; la que utilizaremos en todas las otras funciones.

Seguidamente adquirimos una referencia a un color y rellenamos el fondo de la imagen con dicho color:

```
$amarillo=ImageColorAllocate($imagen,255,255,0);
ImageFill($imagen,0,0,$amarillo);
```

La función `ImageColorAllocate` tiene como parámetros la referencia a la imagen creada y los tres valores que indican la cantidad de rojo, verde y azul para la mezcla. La función `ImageFill` rellena con color a partir de las coordenadas que le pasamos en el segundo y tercer parámetro.

Hasta ahora tenemos un rectángulo de 100x30 pixeles de color amarillo. Para dibujar el código verificador tenemos:

```
$rojo=ImageColorAllocate($imagen,255,0,0);
$valoraleatorio=rand(100000,999999);
ImageString($imagen,5,25,5,$valoraleatorio,$rojo);
```

Recordemos que la función `rand` nos retorna un valor aleatorio comprendido entre los dos valores que le pasamos como parámetro. Seguidamente, con la función `ImageString` imprimimos el número generado de color rojo en las coordenadas 25,5 y con un tamaño de fuente 5 (valores posibles de fuente son de 1 a 5). Tenemos dibujado el código verificador (en nuestro caso es un número verificador, ya que no hemos incorporado caracteres) ahora para hacer más difícil la lectura del número incorporaremos una serie de segmentos que tapen en forma parcial el número aleatorio:

```
for ($c=0;$c<=5;$c++)
{
    $x1=rand(0,$ancho);
    $y1=rand(0,$alto);
    $x2=rand(0,$ancho);
    $y2=rand(0,$alto);
    ImageLine($imagen,$x1,$y1,$x2,$y2,$rojo);
}
```

Disponemos un `for` para que dibuje 6 líneas. Para graficar las líneas llamamos a la función `ImageLine` que tiene cuatro parámetros fundamentales que son las coordenadas de los dos puntos de origen y fin de la línea. Como sabemos el ancho y alto de la imagen y para que la línea se encuentre dentro de los límites del gráfico, disponemos como valor máximo los

valores almacenados en \$ancho y \$alto. Hasta aca la imagen se encuentra en memoria, debemos además indicar qué formato tendrá y enviarla al cliente que la solicitó:

```
Header ("Content-type: image/jpeg");  
ImageJPEG ($imagen);  
ImageDestroy($imagen);
```

También en este tipo de problemas requerimos la función header (recordemos que la empleamos en el redireccionamiento) Indicamos el tipo de archivo que recibirá el navegador. Llamamos en este caso a la función ImageJPEG (podríamos llamar también a ImagePNG, ImageGIF, etc.) que genera la información de la imagen propiamente dicha y por último llamamos a la función ImageDestroy para liberar los recursos ocupados en el servidor por este proceso de generación de la imagen.

pagina1.php

```
<?php  
$ancho=100;  
$alto=30;  
$imagen=imageCreate($ancho,$alto);  
$amarillo=ImageColorAllocate($imagen,255,255,0);  
ImageFill($imagen,0,0,$amarillo);  
$rojo=ImageColorAllocate($imagen,255,0,0);  
$valoraleatorio=rand(100000,999999);  
ImageString($imagen,5,25,5,$valoraleatorio,$rojo);  
for($c=0;$c<=5;$c++)  
{  
    $x1=rand(0,$ancho);  
    $y1=rand(0,$alto);  
    $x2=rand(0,$ancho);  
    $y2=rand(0,$alto);  
    ImageLine($imagen,$x1,$y1,$x2,$y2,$rojo);  
}  
Header ("Content-type: image/jpeg");  
ImageJPEG ($imagen);  
ImageDestroy($imagen);  
?>
```



Confeccionar una imagen dinámica que represente un botón. Utilizar las funciones vistas para imprimir el texto del botón y para rellenar regiones del gráfico, emplear la función imageFilledRectangle. ImageFilledRectangle crea un rectángulo relleno con color "col" en la imagen "imagen", comenzando con la coordenada superior izquierda (x1, y1) y finalizando en la coordenada inferior derecha (x2, y2). imagefilledrectangle (imagen,x1,y1,x2,y2,col)
Recordar que debe ser PHP puro, por lo que deberá borrar todo el código HTML que se encuentra fuera de las marcas:
<?php ?>

Agregar imágenes dinámicas en un archivo HTML

En un concepto anterior habíamos visto como crear un archivo gráfico en forma dinámica y el envío del mismo al navegador que lo solicitó. Ahora veremos como crear el archivo de la imagen y su posterior incorporación en una página HTML.

Continuaremos con el problema del dígito de verificación, pero ahora lo incorporaremos en un formulario donde el operador debe tipear el número que está viendo. En una tercera página verificaremos si ingresó el mismo valor que se generó en forma aleatoria.

La primera página "pagina1.php" es el formulario:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina3.php" method="post">
Dígitos verificadores:
<br>
Ingrese valor:
<input type="text" name="numero">
<br>
<input type="submit" value="Verificar">
</form>
</body>
</html>
```

Tengamos en cuenta que ésta es la página que se solicita inicialmente. Dentro de esta página, se incorpora una marca img para agregar una imagen a la página, pero la misma no es un archivo estático sino un archivo PHP que genera la imagen:

Dígito verificador:

Tengamos en cuenta que la página que procesa el valor ingresado y el valor generado en forma aleatoria es la tercera página:

```
<form action="pagina3.php" method="post">
```

El segundo archivo "pagina2.php" es la imagen propiamente dicha:

```
<?php
$ancho=100;
$alto=30;
$imagen=imageCreate($ancho,$alto);
$amarillo=ImageColorAllocate($imagen,255,255,0);
ImageFill($imagen,0,0,$amarillo);
$rojo=ImageColorAllocate($imagen,255,0,0);
$valoraleatorio=rand(100000,999999);
session_start();
$_SESSION['numeroaleatorio']=$valoraleatorio;
ImageString($imagen,5,25,5,$valoraleatorio,$rojo);
for($c=0;$c<=5;$c++)
{
    $x1=rand(0,$ancho);
    $y1=rand(0,$alto);
    $x2=rand(0,$ancho);
    $y2=rand(0,$alto);
    ImageLine($imagen,$x1,$y1,$x2,$y2,$rojo);
}
Header("Content-type: image/jpeg");
ImageJPEG($imagen);
ImageDestroy($imagen);
```

?>

El algoritmo es el mismo visto en un concepto anterior, la única salvedad es que necesitamos almacenar en una variable de sesión, el número aleatorio para poder compararlo en la página que procesa el formulario:

```
session_start();  
$_SESSION['numeroaleatorio']=$valoraleatorio;
```

Por último el tercer archivo "pagina3.php":

```
<?php  
session_start();  
?>  
<html>  
<head>  
<title>Problema</title>  
</head>  
<body>  
<?php  
if ($_SESSION['numeroaleatorio']==$_REQUEST['numero'])  
    echo "Ingresó el valor correcto";  
else  
    echo "Incorrecto";  
?>  
</body>  
</html>
```

Lo primero que hacemos es llamar a la función que rescata las variables de sesión:

```
<?php  
session_start();  
?>
```

Disponemos un if para verificar si el valor ingresado en el formulario es el mismo que el valor almacenado en la variable de sesión:

```
if ($_SESSION['numeroaleatorio']==$_REQUEST['numero'])  
    echo "Ingresó el valor correcto";  
else  
    echo "Incorrecto";
```

pagina1.php

```
<html>  
<head>  
<title>Problema</title>  
</head>  
<body>  
<form action="pagina3.php" method="post">  
    Dígitos verificadores:  
<br>  
    Ingrese valor:  
<input type="text" name="numero">  
<br>  
<input type="submit" value="Verificar">  
</form>  
</body>  
</html>
```

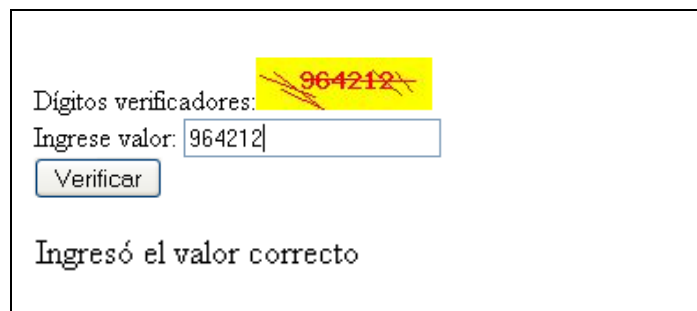
pagina2.php

```
<?php
```

```
$ancho=100;
$alto=30;
$imagen=imageCreate($ancho,$alto);
$amarillo=ImageColorAllocate($imagen,255,255,0);
ImageFill($imagen,0,0,$amarillo);
$rojo=ImageColorAllocate($imagen,255,0,0);
$valoraleatorio=rand(100000,999999);
session_start();
$_SESSION['numeroaleatorio']=$valoraleatorio;
ImageString($imagen,5,25,5,$valoraleatorio,$rojo);
for($c=0;$c<=5;$c++)
{
    $x1=rand(0,$ancho);
    $y1=rand(0,$alto);
    $x2=rand(0,$ancho);
    $y2=rand(0,$alto);
    ImageLine($imagen,$x1,$y1,$x2,$y2,$rojo);
}
Header ("Content-type: image/jpeg");
ImageJPEG ($imagen);
ImageDestroy($imagen);
?>
```

pagina3.php

```
<?php
session_start();
?>
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
if ($_SESSION['numeroaleatorio']==$_REQUEST['numero'])
    echo "Ingresó el valor correcto";
else
    echo "Incorrecto";
?>
</body>
</html>
```



The screenshot shows a web form titled "Problema". It contains a label "Dígitos verifcadores:" followed by a yellow box with the number "964212" and a red checkmark. Below this is a text input field labeled "Ingrese valor:" containing the number "964212". A "Verificar" button is positioned below the input field. At the bottom of the form, the text "Ingresó el valor correcto" is displayed.

Confeccionar un formulario que pida ingresar la dirección de un sitio de internet y mediante un control de tipo select permita dar un puntaje a la misma (cargar los valores de 0 a 5). Luego, en la segunda página, imprimir el nombre del sitio y un gráfico creado en forma dinámica con el puntaje obtenido (hacer un círculo por cada punto). Un poco de ayuda para este problema: En el archivo "pagina2.php" debemos disponer la marca HTML img para mostrar la imagen y debemos pasar a este archivo el valor que nos llegó del formulario, es decir, el puntaje seleccionado:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
echo "La direccion: $_REQUEST[direccion] tiene";
?>

</body>
</html>
```

Por último, queda como actividad, dibujar tantos círculos como indica el parámetro puntos: Recordemos que para recuperar el parámetro puntos lo accedemos mediante el vector asociativo `$_REQUEST`:

```
$_REQUEST['puntos']
```

Para dibujar los círculos rellenos debemos emplear la función:

```
imagefilledellipse ( <manejador de imagen>, <centro en x>,
                    <centro en y>, <ancho del elipse>,
                    <alto del elipse>,<manejador de color>)
```

Administración de fechas y horas (función date)

La función `date` retorna un string con una fecha y hora, o partes de ella según un string de formato que le pasamos como parámetro. Se obtiene la fecha y hora del servidor. Veamos como ejemplo la impresión de la fecha y hora actual del servidor:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
echo "La fecha de hoy es:";
$fecha=date("d/m/Y");
echo $fecha;
echo "<br>";
echo "La hora actual es:";
$hora=date("H:i:s");
echo $hora;
echo "<br>";
?>
<a href="pagina2.php">Siguiente problema</a>
</body>
</html>
```

Los caracteres de formato utilizados en las dos llamadas de la función date son:

d	día	del	mes	con	dos	dígitos	"01"	al	"31"
m	mes	con	dos	dígitos	"01"	al	"12"		
Y	año	con	cuatro	dígitos					

Para la hora, los caracteres que serán sustituidos son:

H	hora	con	dos	dígitos	"00"	a	"23"
i	minutos	con	dos	dígitos	"00"	a	"59"
s	segundos	con	dos	dígitos	"00"	a	"59"

Los otros caracteres que disponemos al llamar a la función date, son retornados sin cambios, en este caso la barra y los dos puntos.

Los caracteres de formato que veíamos son los más comunes, pero tenemos otras variantes posibles. Si queremos los días y meses sin el cero delante y el año con dos dígitos tenemos entonces:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
echo "La fecha de hoy es:";
$fecha=date("j/n/y");
echo $fecha;
echo "<br>";
?>
<a href="pagina3.php">Siguiente problema</a>
</body>
</html>
```

Los	caracteres	que	ahora	tenemos	son:	
j	día	del	mes	"1"	al	"31"
n	mes	"1"	al	"12"		
y	año	con	dos	dígitos		

Por último, a los otros caracteres de formato de la función date que nos pueden servir en alguna ocasión, los podemos ver en el siguiente ejemplo:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$dato=date("L");
if ($dato==1)
    echo "Año bisiesto";
else
    echo "Año no bisiesto";
echo "<br>";
echo "Día de la semana:";
$dato=date("w");
switch ($dato) {
    case 0: echo "domingo";
            break;
    case 1: echo "lunes";
            break;
```

```
case 2: echo "martes";
        break;
case 3: echo "miércoles";
        break;
case 4: echo "jueves";
        break;
case 5: echo "viernes";
        break;
case 6: echo "sábado";
        break;
}
?>
</body>
</html>
```

Los caracteres son:
L "1" or "0", según si el año es bisiesto o no
w día de la semana, en número, de "0" (domingo) a "6" (sábado)

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
echo "La fecha de hoy es:";
$fecha=date("d/m/Y");
echo $fecha;
echo "<br>";
echo "La hora actual es:";
$hora=date("H:i:s");
echo $hora;
echo "<br>";
?>
<a href="pagina2.php">Siguiete problema</a>
</body>
</html>
```

pagina2.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
echo "La fecha de hoy es:";
$fecha=date("j/n/y");
echo $fecha;
echo "<br>";
?>
```

```
<a href="pagina3.php">Siguiente problema</a>
</body>
</html>
```

pagina3.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$dato=date("L");
if ($dato==1)
    echo "Año bisiesto";
else
    echo "Año no bisiesto";
echo "<br>";
echo "Día de la semana:";
$dato=date("w");
switch ($dato) {
    case 0: echo "domingo";
        break;
    case 1: echo "lunes";
        break;
    case 2: echo "martes";
        break;
    case 3: echo "miércoles";
        break;
    case 4: echo "jueves";
        break;
    case 5: echo "viernes";
        break;
    case 6: echo "sábado";
        break;
}
?>
</body>
</html>
```

La fecha de hoy es:05/03/2009

La hora actual es:20:33:12

[Siguiente problema](#)

La fecha de hoy es:5/3/09

[Siguiente problema](#)

Año no bisiesto
Día de la semana:jueves

El archivo "pagina1.php" debe ser el formulario de ingreso de datos. El archivo "pagina2.php" debe registrar la información en el archivo de datos. Por último el archivo "pagina3.php" debe imprimir el archivo con todas las quejas registradas hasta el momento. Disponer un hipervínculo en el archivo "pagina1.php" para poder ver todas las quejas registradas.

Recordar que el archivo de texto obligatoriamente debe llamarse: "datos.txt".

Validación de una fecha ingresada por teclado (checkdate)

Si disponemos en forma separada del día, mes y año hay, una función que nos indica si se trata de una fecha válida:

```
boolean checkdate ( mes, dia, año)
```

Retorna verdadero si la fecha es válida, falso en caso contrario.

Implementemos un formulario que nos solicite el ingreso de una fecha:

```
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese una fecha (dd/mm/aaaa):
<input type="text" name="dia" size="2">
<input type="text" name="mes" size="2">
<input type="text" name="anio" size="4">
<br>
<input type="submit" value="validar">
</form>
</body>
</html>
```

Y la página que procesa la fecha ingresada es:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
if (checkdate($_REQUEST['mes'],$_REQUEST['dia'],$_REQUEST['anio']))
    echo "La fecha ingresada es correcta";
else
    echo "La fecha no es válida";
?>
</body>
</html>
```

En este programa llamamos a la función checkdate pasando los tres parámetros requeridos en el orden: mes, día y año. Podemos validar previamente si se han cargado valores numéricos en cada control "text", esto llamando a la función is_numeric(variable). Retorna true si la variable almacena un

número, falso en caso contrario.

El programa modificado quedará entonces:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
if (is_numeric($_REQUEST['dia']) &&
    is_numeric($_REQUEST['mes']) &&
    is_numeric($_REQUEST['anio']))
{
    if (checkdate($_REQUEST['mes'], $_REQUEST['dia'], $_REQUEST['anio']))
        echo "La fecha ingresada es correcta";
    else
        echo "La fecha no es válida";
}
else
    echo "La fecha no es válida";
?>
</body>
</html>
```

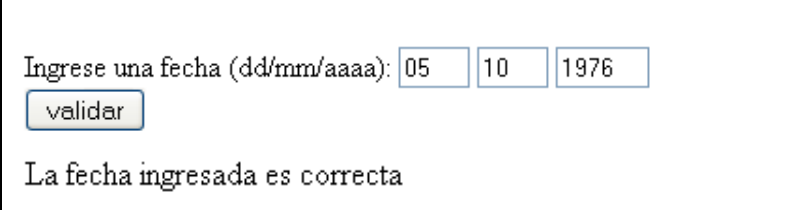
pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese una fecha (dd/mm/aaaa):
<input type="text" name="dia" size="2">
<input type="text" name="mes" size="2">
<input type="text" name="anio" size="4">
<br>
<input type="submit" value="validar">
</form>
</body>
</html>
```

pagina2.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
if (is_numeric($_REQUEST['dia']) &&
    is_numeric($_REQUEST['mes']) &&
    is_numeric($_REQUEST['anio']))
{
    if (checkdate($_REQUEST['mes'], $_REQUEST['dia'], $_REQUEST['anio']))
```

```
        echo "La fecha ingresada es correcta";
    else
        echo "La fecha no es válida";
    }
    else
        echo "La fecha no es válida";
    ?>
</body>
</html>
```

A screenshot of a web form within a rectangular border. The form contains the text "Ingrese una fecha (dd/mm/aaaa):" followed by three input fields containing the values "05", "10", and "1976". Below these fields is a button labeled "validar". At the bottom of the form, the text "La fecha ingresada es correcta" is displayed.

Confeccionar un formulario que solicite la carga de una fecha, disponer tres controles HTML de tipo "select" para elegir el día, el mes y el año. Validar si se trata de una fecha válida en la segunda página.

Carga de una fecha en una tabla de MySQL

La tabla alumnos tiene un campo que no habíamos nombrado llamado "fechanac" que es de tipo "date", es decir que permite almacenar una fecha. "fechanac" almacenará la fecha de nacimiento del alumno. Veremos cual es la estructura que debemos darle a la fecha para que el MySQL la tome como válida. Haremos el alta de la tabla alumnos que habíamos visto anteriormente añadiéndole la carga de la fecha de nacimiento. El primer formulario, prácticamente ya visto, es:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese nombre:
<input type="text" name="nombre"><br>
Ingrese mail:
<input type="text" name="mail"><br>
Ingrese la fecha de nacimiento (dd/mm/aaaa):
<input type="text" name="dia" size="2">
<input type="text" name="mes" size="2">
<input type="text" name="anio" size="4">
<br>
Seleccione el curso:
<select name="codigocurso">
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
```

```
die("Problemas en la selección de la base de datos");
$registros=mysql_query("select codigo,nombrecur from cursos",$conexion)
or
die("Problemas en el select:".mysql_error());
while ($reg=mysql_fetch_array($registros))
{
    echo "<option value=\"".$reg[codigo].\">$reg[nombrecur]</option>";
}
?>
</select>
<br>
<input type="submit" value="Registrar">
</form>
</body>
</html>
```

Lo único que podemos decir, es que agregamos tres controles de tipo "text" para el ingreso independiente del día, el mes y el año. El segundo formulario, y más importante, es el alta propiamente dicha en la tabla alumnos:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
die("Problemas en la seleccion de la base de datos");
$fechanacimiento=$_REQUEST['anio']."-".$_REQUEST['mes']."-".$_REQUEST['dia'];
mysql_query("insert into alumnos(nombre,mail,codigocurso,fechanac) values
('".$_REQUEST[nombre]."', '".$_REQUEST[mail]."',
".$_REQUEST[codigocurso]."', '".$_REQUEST[fechanacimiento]."', $conexion) or
die("Problemas en el select".mysql_error());
mysql_close($conexion);
echo "El alumno fue dado de alta.";
?>
<br>
<a href="pagina3.php">ver listado de alumnos</a>
</body>
</html>
```

Lo primero que hacemos es generar una string que contenga el año-mes-día, en ese orden y utilizando como separador el caracter "-":

```
$fechanacimiento=$_REQUEST['anio']."-".$_REQUEST['mes']."-".$_REQUEST['dia'];
```

Tenemos ahora en la variable \$fechanacimiento el valor de la fecha ingresada por teclado con el formato que requiere el MySQL, procedemos ahora a plantear el comando insert con todos los datos ingresados en el formulario:

```
mysql_query("insert into alumnos(nombre,mail,codigocurso,fechanac) values
('".$_REQUEST[nombre]."', '".$_REQUEST[mail]."',
".$_REQUEST[codigocurso]."', '".$_REQUEST[fechanacimiento]."', $conexion) or
die("Problemas en el select".mysql_error());
```

Por último, dispusimos un hipervínculo a una tercera página donde mostramos el contenido de la tabla alumnos, aquí podremos controlar los datos ingresados. No hemos validado la fecha ingresada, tarea obligatoria cuando implementemos un sitio real.

El último archivo contiene la página que imprime el contenido de la tabla alumnos, con el campo fechanac inclusive:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select alu.codigo as codigo,nombre,mail,
                        codigocurso,fechanac,nombrecur from alumnos as
alu
                        inner      join      cursos      as      cur      on
cur.codigo=alu.codigocurso",
                        $conexion) or
    die("Problemas en el select:".mysql_error());
while ($reg=mysql_fetch_array($registros))
{
    echo "Codigo:".$reg['codigo']."<br>";
    echo "Nombre:".$reg['nombre']."<br>";
    echo "Mail:".$reg['mail']."<br>";
    echo "Fecha de Nacimiento:".$reg['fechanac']."<br>";
    echo "Curso:".$reg['nombrecur']."<br>";
    echo "<hr>";
}
mysql_close($conexion);
?>
</body>
</html>
```

paginal.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<form action="pagina2.php" method="post">
Ingrese nombre:
<input type="text" name="nombre"><br>
Ingrese mail:
<input type="text" name="mail"><br>
Ingrese la fecha de nacimiento (dd/mm/aaaa):
<input type="text" name="dia" size="2">
<input type="text" name="mes" size="2">
<input type="text" name="anio" size="4">
<br>
Seleccione el curso:
<select name="codigocurso">
<?php
$conexion=mysql_connect("localhost","root","z80") or
    die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
```

```
$registros=mysql_query("select codigo,nombrecur from cursos",$conexion) or
die("Problemas en el select:".mysql_error());
while ($reg=mysql_fetch_array($registros))
{
    echo "<option value=\"\$reg[codigo]\">$reg[nombrecur]</option>";
}
?>
</select>
<br>
<input type="submit" value="Registrar">
</form>
</body>
</html>
```

pagina2.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
die("Problemas en la conexion");
mysql_select_db("phpfacil",$conexion) or
die("Problemas en la seleccion de la base de datos");
$fechanacimiento=$_REQUEST['anio']."-".$_REQUEST['mes']."-
".$_REQUEST['dia'];
mysql_query("insert into alumnos(nombre,mail,codigocurso,fechanac) values
('".$_REQUEST[nombre]','".$_REQUEST[mail]',
".$_REQUEST[codigocurso]','$fechanacimiento')", $conexion) or
die("Problemas en el select".mysql_error());
mysql_close($conexion);
echo "El alumno fue dado de alta.";
?>
<br>
<a href="pagina3.php">ver listado de alumnos</a>
</body>
</html>
```

pagina3.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","z80") or
die("Problemas en la conexion");
```

```
mysql_select_db("phpfacil",$conexion) or
    die("Problemas en la selección de la base de datos");
$registros=mysql_query("select          alu.codigo          as
codigo,nombre,mail,codigocurso,fechanac,
    nombrecur from alumnos as alu
    inner join cursos as cur on cur.codigo=alu.codigocurso",
    $conexion) or
    die("Problemas en el select:".mysql_error());
while ($reg=mysql_fetch_array($registros))
{
    echo "Codigo:". $reg['codigo']."<br>";
    echo "Nombre:". $reg['nombre']."<br>";
    echo "Mail:". $reg['mail']."<br>";
    echo "Fecha de Nacimiento:". $reg['fechanac']."<br>";
    echo "Curso:". $reg['nombrecur']."<br>";
    echo "<hr>";
}
mysql_close($conexion);
?>
</body>
</html>
```

Ingrese nombre:

Ingrese mail:

Ingrese la fecha de nacimiento (dd/mm/aaaa):

Seleccione el curso:

El alumno fue dado de alta.
[ver listado de alumnos](#)

```
Codigo:55587
Nombre:Paul
Mail:pds-soft@gmail.com
Fecha de Nacimiento:
Curso:armado y reparacion de pc
```

```
Codigo:55591
Nombre:Paul
Mail:pds-soft@gmail.com
Fecha de Nacimiento:1976-10-05
Curso:armado y reparacion de pc
```

Confeccionar un programa que permita efectuar el alta en la tabla alumnos. Para el ingreso de la fecha de nacimiento del alumno, disponer 3 controles de tipo "select" (en el día disponer valores entre 1 y 31, en el mes valores comprendidos entre 1 y 12 y por último en el año disponer valores comprendidos entre 1900 y 2006) Validar la fecha en la página que se efectúa el insert propiamente dicho. Recomendación: Plantear una estructura repetitiva de PHP para la creación del control "select" donde se selecciona el año de nacimiento.

Formateo de datos en una página (printf)

Hasta ahora siempre hemos impreso dentro de la página, utilizando el comando echo, pero en ocasiones que necesitamos mayor control sobre el formato de impresión, podemos utilizar la función printf.

La función printf requiere como primer parámetro una cadena de control donde se indica cómo deben imprimirse el resto de parámetros de la misma función. El siguiente ejemplo muestra el contenido de una variable entera con distintos formatos, lo mismo hacemos para una variable de tipo double:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$entero=255;
printf("Valor entero en formato decimal %d <br>", $entero);
printf("Valor entero en formato hexadecimal con letras minúsculas
%x<br>", $entero);
printf("Valor entero en formato hexadecimal con letras mayúsculas
%X<br>", $entero);
printf("Valor entero en formato binario %b<br>", $entero);
printf("Valor entero en formato octal %o<br>", $entero);
$letra=65;
printf("Valor entero como caracter ascii %c<br>", $letra);
echo "<br>";
$real=10.776;
```

```
printf("Impresion de un valor de tipo double %f <br>",$real);
printf("Impresion de un valor de tipo double indicando la cantidad de
decimales
a imprimir %0.2f <br>",$real);
?>
<br>
<A href="pagina2.php">Algunas utilidades de estas conversiones</A>
</body>
</html>
```

Como podemos ver, es posible imprimir el contenido de una variable entera en formato ASCII, decimal, hexadecimal, octal. Y con una variable de tipo double, la podemos restringir la cantidad de decimales que deben aparecer. La función printf sustituye todos los lugares en los cuales encuentra el caracter %, por el valor que le pasamos desde el segundo parámetro en adelante. Si queremos imprimir este caracter: %, con la función printf, debemos disponer dos: %%.

Podemos ver un uso relativamente seguido para cuando imprimamos valores de tipo double y necesitemos restringir a una determinada cantidad de decimales. Pero las otras conversiones, ¿nos servirán? Veamos una utilidad de la función printf formateando a tipo de dato hexadecimal:

```
<html>
<head>
<title>Problema</title>
</head>
<body bgcolor="<?php printf("#%X%X%X",150,150,0); ?>">
En esta página definimos el color de fondo indicando la cantidad de
rojo,verde
y azul, en formato decimal y solicitando a la función printf que haga la
conversión a hexadecimal. Recordemos que la propiedad bgcolor de la
marca body, lo requiere en hexadecimal.<br><br>
<a href="pagina3.php">último ejemplo</a>
</body>
</html>
```

La función printf puede formatear n datos en una única llamada, como ocurre en este caso:

```
<body bgcolor="<?php printf("#%X%X%X",150,150,0); ?>">
```

Cuando tenemos los tres valores en formato decimal, para crear un color, la función printf nos facilita la tarea de generar el color definitivo en hexadecimal.

Por último, con la función printf, podemos determinar el número de caracteres que va a ocupar o en su defecto se rellenará con ceros:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$dia=6;
$mes=5;
$anio=2006;
printf("%02d/%02d/%d",$dia,$mes,$anio);
?>
</body>
</html>
```


Con esto logramos que una fecha aparezca con el día y el mes siempre de dos dígitos.

Si deseamos rellenar con otro caracter debemos disponer una simple comilla y el caracter a rellenar:

```
$importe=170;  
printf("Valor:$%'x7d",$importe);
```

pagina1.php

```
<html>  
<head>  
<title>Problema</title>  
</head>  
<body>  
  
<?php  
$entero=255;  
printf("Valor entero en formato decimal %d <br>",$entero);  
printf("Valor entero en formato hexadecimal con letras minúsculas %x<br>",$entero);  
printf("Valor entero en formato hexadecimal con letras mayúsculas %X<br>",$entero);  
printf("Valor entero en formato binario %b<br>",$entero);  
printf("Valor entero en formato octal %o<br>",$entero);  
$letra=65;  
printf("Valor entero como caracter ascii %c<br>",$letra);  
echo "<br>";  
$real=10.776;  
printf("Impresion de un valor de tipo double %f <br>",$real);  
printf("Impresion de un valor de tipo double indicando la cantidad de decimales a imprimir %0.2f <br>",$real);  
?>  
<br>  
<A href="pagina2.php">Algunas utilidades de estas conversiones</A>  
</body>  
</html>
```

pagina2.php

```
<html>  
<head>  
<title>Problema</title>  
</head>  
<body bgcolor="<?php printf("#%X%X%X",150,150,0); ?>">  
Esta página definimos el color de fondo indicando la cantidad de rojo,verde y azul en formato decimal y solicitando a la función printf que haga la conversión a hexadecimal. Recordemos que la propiedad bgcolor de la marca body lo requiere en hexadecimal.<br><br>  
<a href="pagina3.php">último ejemplo</a>  
</body>  
</html>
```

pagina3.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
$dia=6;
$mes=5;
$anio=2006;
printf("%02d/%02d/%d",$dia,$mes,$anio);
?>
</body>
</html>
```

Valor entero en formato decimal 255
Valor entero en formato hexadecimal con letras minúsculas ff
Valor entero en formato hexadecimal con letras mayúsculas FF
Valor entero en formato binario 11111111
Valor entero en formato octal 377
Valor entero como caracter ascii A

Impresion de un valor de tipo double 10.776000
Impresion de un valor de tipo double indicando la cantidad de decimales a imprimir 10.78

[Algunas utilidades de estas conversiones](#)

Esta página definimos el color de fondo indicando la cantidad de rojo, verde y azul en formato decimal y solicitando a la función printf que haga la conversión a hexadecimal. Recordemos que la propiedad bgcolor de la marca body lo requiere en hexadecimal.

[último ejemplo](#)

06/05/2006

Confeccionar un formulario que solicite la carga del nombre de una persona, su mail y un importe que desea donar. En la página que procesa los datos ingresados, mostrar el importe a donar relleno con ceros, por ejemplo: \$0000170 dolares.

Formateo de datos y salida a un string (sprintf)

Así como vimos que la función printf nos permite tener un control más fino sobre cómo un dato debe imprimirse en una página, la función sprintf nos permite formatear la salida de un dato hacia un string y no a la página HTML. Tipos de conversión utilizadas por la función sprintf (tener en cuenta que son los mismos

caracteres de control que se aplican a la función printf):

%b	Formatea un entero como un número binario.
%d	Formatea un entero como un número decimal con signo.
%u	Formatea un entero como un número decimal sin signo.
%o	Formatea un entero como un número octal.
%x	Formatea un entero como un número hexadecimal en minúsculas.
%X	Formatea un entero como un número hexadecimal en mayúsculas.
%c	Formatea un entero como un carácter ASCII.
%f	Formatea un double con una cantidad de decimales.
%s	Formatea un string.

Un ejemplo utilizando la función sprintf:

```
<html>
<head>
<title>Problema</title>
</head>
<body>
<?php
function retornarColorHexa($rojo,$verde,$azul)
{
    $color=sprintf("#%02X%02X%02X",$rojo,$verde,$azul);
    return $color;
}
?>
<table>
<tr>
<td bgcolor="<?php echo retornarColorHexa(255,0,0)?>">Cuadro
rojo</td>
<td bgcolor="<?php echo retornarColorHexa(0,255,0)?>">Cuadro
verde</td>
<td bgcolor="<?php echo retornarColorHexa(0,0,255)?>">Cuadro
azul</td>
</tr>
</table>
</body>
</html>
```

Implementamos una función a la cual le enviamos 3 valores decimales y nos retorna un string que especifica un color en hexadecimal.

pagina1.php

```
<html>
<head>
<title>Problema</title>
</head>
<body>

<?php
function retornarColorHexa($rojo,$verde,$azul)
{
    $color=sprintf("#%02X%02X%02X",$rojo,$verde,$azul);
    return $color;
}
```

?>

<table>

<tr>

<td bgcolor="<?php echo retornarColorHexa(255,0,0)?>">Cuadro rojo</td>

<td bgcolor="<?php echo retornarColorHexa(0,255,0)?>">Cuadro verde</td>

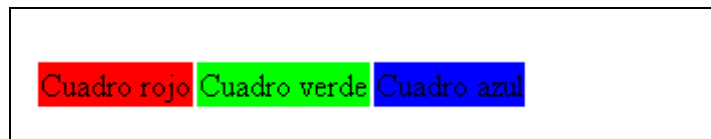
<td bgcolor="<?php echo retornarColorHexa(0,0,255)?>">Cuadro azul</td>

</tr>

</table>

</body>

</html>



Imprimir la tabla de caracteres ASCII. Implementar una función que le enviemos un entero y nos retorne el caracter ASCII de dicho valor.

<http://www.phpya.com.ar/>