Materia: Tecnologías para la Web.
Tema: PHP.

1. Introducción

- PHP o PHP: Hypertext Preprocessor, se ha convertido en uno de los lenguajes de script del lado del servidor más populares para la creación de páginas web dinámicas.
- PHP es de código abierto y de plataforma independiente, sin embargo existen implantaciones para todos los principales sistemas operativos UNIX, Linux, Mac y Windows. PHP también soporta un gran número de bases de datos.

2. Un Simple Programa PHP

- El código PHP se anida directamente en documentos HTML5 y es interpretado en el servidor.
- Los nombres del archivo script de PHP terminan con `.php`.
- En PHP, se inserta el código entre los delimitadores de secuencias de comandos `<?` php y `?>`. El código PHP se puede colocar en cualquier etiqueta en HTML5, siempre y cuando el código esté encerrado en estos delimitadores.
- Las variables son precedidas por un `$` y se crean la primera vez que se les encuentra.
- Las declaraciones de PHP terminan con un punto y coma (`;`).
- Los comentarios de una sola línea que comienzan con dos barras inclinadas (`//`). El texto a la derecha del delimitador es ignorado por el intérprete. Los comentarios multilínea empiezan con el delimitador `/ *` y terminan con el delimitador `*/`.
- Cuando una variable se encuentra dentro de una cadena entre comillas dobles (`""`), PHP interpola la variable e inserta el valor de la variable donde el nombre de la variable aparece en la cadena.
- Todas las operaciones que requieren la interpolación PHP se ejecutan en el servidor antes de que el documento HTML5 se envíe al cliente.

Tipo	Descripción
`int, integer`	Número enteros (por ejemplo, números sin punto decimal)
`float, double, real`	Números reales (por ejemplo, números con punto decimal)
`string`	Texto encerrado entre comillas sencillas (`''`) o dobles (" "). Nota: El uso de comillas dobles permite a PHP reconocer secuencias de escape.
`bool, boolean`	`true` o `false`.
`array`	Grupos de elementos.
`object`	Grupo con datos asociados y métodos.
`resource`	Una fuente externa, usualmente información de una base de datos.
`NULL`	Sin valor.

EJEMPLO 1.
Archivo primero.php:

```
<!-- Un programa sencillo en PHP. -->
<!DOCTYPE html>
<html>
      <?php
            $name = "Beatles"; // declaration and initialization
      ?><!-- end PHP script -->
      <head>
            <meta charset = "utf-8">
            <title>Simple PHP document</title>
      </head>
      <body>
            <!-- print variable name's value -->
            <h1><?php print( "Bienvenido a PHP, $name!" ); ?></h1>
      </body>
```

```
</html>
```

3. Conversión entre Tipos de Datos

- Las variables de PHP que se escriben sueltas pueden contener diferentes tipos de datos en diferentes momentos.
- Las conversiones de tipos se realizan utilizando la función `settype`. Esta función toma dos argumentos: una variable cuyo tipo se va a cambiar y el nuevo tipo de la variable.
- Las variables se convierten automáticamente al tipo del valor que tienen asignados.
- Función `gettype` devuelve el tipo actual de su argumento.
- La llamada a la función `settype` puede resultar en la pérdida de datos. Por ejemplo, los datos `double` se truncan cuando se convierten en enteros.
- Cuando se convierte de una cadena a un número, PHP utiliza el valor del número que aparece al principio de la cadena. Si no aparece ningún número al principio, la cadena se evalúa como `0`.
- Otra opción para la conversión entre tipos es la conversión `casting`. La conversión `casting` no cambia el contenido, más bien crea una copia temporal del valor de una variable en la memoria.
- El operador de concatenación (`. .`) combina varias cadenas.
- Una división de enunciados de impresión a través de múltiples líneas imprime todos los datos que están encerrados en sus paréntesis.

EJEMPLO 2.
Archivo datos.php:

```
<!-- Conversión de tipos de datos. -->
<!DOCTYPE html>
<html>
	<head>
		<meta charset = "utf-8">
		<title>Conversión de tipos de datos</title>
		<style type = "text/css">
			p	{ margin: 0; }
			.head { margin-top: 10px; font-weight: bold; }
			.space	{ margin-top: 10px; }
		</style>
	</head>
	<body>
		<?php
			// declarar un string, double and integer
			$testString = "3.5 seconds";
			$testDouble = 79.2;
			$testInteger = 12;
		?><!-- fin del script PHP -->
		<!-- imprime cada valor y tipo de la variable -->
		<p class = "head">Valores originales:</p>
		<?php
			print( "<p>$testString is a(n) ".gettype( $testString )
				."</p>" );
			print( "<p>$testDouble is a(n) ".gettype( $testDouble )
				."</p>" );
			print( "<p>$testInteger is a(n) ".gettype( $testInteger )
				."</p>" );
		?><!-- fin del script PHP -->
		<p class = "head">Conversi&oacute;n a otros tipos de datos:</p>
```

```
            <?php
                  // llama a la función settype para convertir la variable
                  // testString a diferentes tipos de datos
                  print( "<p>$testString " );
                  settype( $testString, "double" );
                  print( " as a double is $testString</p>" );
                  print( "<p>$testString " );
                  settype( $testString, "integer" );
                  print( " as an integer is $testString</p>" );
                  settype( $testString, "string" );
                  print( "<p class = 'space'>Conversi&oacute;n inversa a string resulta en
                        $testString</p>" );
                  // usar el tipo casting para convertir las variables a tipos diferentes
                  $data = "98.6 degrees";
                  print( "<p class = 'space'>Antes de la conversi&oacute;n: $data es un ".
                        gettype( $data )."</p>" );
                  print( "<p class = 'space'>Usando el tipo casting:</p>
                        <p>as a double: ".(double) $data."</p>".
                        "<p>as an integer: ".(integer) $data."</p>";
                  print( "<p class = 'space'>Despu&eacute;s de la conversi&oacute;n: $data
is a ".
                  gettype( $data )."</p>" );
            ?><!-- end PHP script -->
      </body>
</html>
```

4. Operadores Aritméticos

- La función `define` crea una constante nombrada. Toma dos argumentos: el nombre y el valor de la constante. Un tercer argumento opcional acepta un valor booleano que especifica si la constante es sensible a mayúsculas y/o minúsculas en forma predeterminada.
- Las variables no inicializadas tienen valores indefinidos. En un contexto numérico, un valor indefinido se evalúa con `0`. En un el caso de una cadena, se evalúa como "`undef`").
- Las palabras clave no pueden utilizarse como nombres de función, método de clase o espacio de nombres.

Palabras clave de PHP				
abstract	and	array	as	break
case	catch	class	clone	const
continue	declare	default	do	else
elseif	enddeclare	endfor	endforeach	endif
endswitch	endwhile	extends	final	for
foreach	function	global	goto	if
implements	interface	instanceof	namespace	new
or	private	protected	public	static
switch	throw	try	use	var
while	xor			

Operador	Tipo	Asociatividad
new clone	constructor copia un objeto	ninguna
[ ]	subíndice	izquierda a derecha
++ --	incremento decremento	ninguna
~ - @ (tipo)	no bit negativo unario control de error conversión cast	derecha a izquierda
instanceof		ninguna
!	negación	derecha a izquierda
* / %	multiplicación división módulo	izquierda a derecha
+ - .	suma resta concatenación	izquierda a derecha
<< >>	corrimiento a la izquierda de bits corrimiento a la derecha de bits	izquierda a derecha
< > <= >=	menor que mayor que menor o igual que mayor o igual que	ninguna
== != === !==	igual diferente idéntico no idéntico	ninguna
&	AND de bits	izquierda a derecha
^	XOR de bits	izquierda a derecha
|	OR de bits	izquierda a derecha
&&	AND lógico	izquierda a derecha
||	OR lógico	izquierda a derecha
? :	condicional ternario	izquierda a derecha

EJEMPLO 3.
Archivo operadores.php:

```
<!-- Uso de los operadores aritméticos. -->
<!DOCTYPE html>
<html>
	<head>
		<meta charset = "utf-8">
		<style type = "text/css">
			p { margin: 0; }
		</style>
		<title>Uso de los operadores aritm&eacute;ticos</title>
	</head>
	<body>
		<?php
			$a = 5;
			print( "<p>El valore de la variable a es $a</p>" );
			// se define la constante VALUE
			define( "VALUE", 5 );
```

```
                // agrega la constante VALUE a la variable $a
                $a = $a + VALUE;
                print("<p>La variable a despu&eacute;s de a&ntilde;adir la constante VALUE
es $a</p>");
                // multiplica la variable $a por 2
                $a *= 2;
                print( "<p>Multiplicar la variable a por 2 resulta $a</p>" );
                // prueba si la variable $a es menor que 50
                if ( $a < 50 )
                      print( "<p>La variable a es menor que 50</p>" );
                // add 40 to variable $a
                $a += 40;
                print( "<p>La variable a despu&eacute;s de agregar 40 es $a</p>" );
                // prueba si la variable $a es 50 o menor
                if ( $a < 51 )
                      print( "<p>La variable a es todav&iacute;a 50 o menor</p>" );
                elseif ( $a < 101 ) // $a >= 51 and <= 100
                      print( "<p>La variable a est&aacute; ahora entre 50 y 100,
                            inclusive</p>" );
                else // $a > 100
                      print( "<p>La variable a es ahora mayor a 100</p>" );
                // imprime una variable sin inicializar
                print( "<p>Uso de una variable antes de inicializarla:
                      $nothing</p>" ); // nothing se evalúa a ""
                // agregar la constante VALUE a una variable sin inicializar
                $test = $num + VALUE; // num se evalúa a 0
                print( "<p>Una variable no inicializada m&aacute;s una constant
                      VALUE resulta $test</p>" );
                // agrega un string a un integer
                $str = "3 dollars";
                $a += $str;
                print( "<p>Agregar un string a la variable a resulta $a</p>" );
          ?><!-- fin del script PHP -->
      </body>
</html>
```

5. Inicialización y Manejo de Arreglos

- PHP proporciona la capacidad de almacenar datos en arreglos. Las matrices se dividen en elementos que se comportan como variables individuales. Los nombres de arreglos, al igual que otras variables, comienzan con el símbolo `$`.
- Los elementos individuales se acceden siguiendo el nombre de la variable del arreglo con un índice entre corchetes (`[]`).
- Si se asigna un valor a un arreglo que no existe, se crea el arreglo. Del mismo modo, la asignación de un valor a un elemento en el que se omite el índice añade un nuevo elemento al final del arreglo.
- La función `count` devuelve el número total de elementos del arreglo.

- La función `array` crea un arreglo que contiene los argumentos que se le pasan. El primer elemento de la lista de argumentos se almacena como el primer elemento de la matriz (índice `0`), el segundo elemento se almacena como el segundo elemento del arreglo y así sucesivamente.
- Los arreglos con índices numéricos se denominan arreglos asociativos. Se puede crear un arreglo asociativo mediante el operador =>, donde el valor a la izquierda del operador es el índice del arreglo y el valor a la derecha es el valor del elemento.
- PHP proporciona funciones para iterar a través de los elementos de un arreglo. Cada arreglo tiene un apuntador interno incluido, que apunta al elemento del arreglo al que actualmente se está haciendo referencia. La función `reset` establece el apuntador interno al primer elemento de matriz. La función `key` devuelve el índice del elemento de referencia actualmente referenciado por el apuntador interno, y la función `next` mueve el apuntador interno al siguiente elemento.
- La instrucción `foreach`, diseñada para la iteración a través de los arreglos, se inicia con el arreglo por recorrer, seguido de la palabra clave `as`, seguido de dos variables: a la primera se le asigna el índice del elemento y a la segunda se le asigna el valor de ese índice. (Si sólo una variable aparece después de `as`, se asigna el valor del elemento de matriz).

EJEMPLO 4.
Archivo arreglos.php:

```
<!-- Manejo de Arreglos. -->
<!DOCTYPE html>
<html>
	<head>
		<meta charset = "utf-8">
		<title>Manejo de Arreglos</title>
		<style type = "text/css">
			p { margin: 0; }
			.head { margin-top: 10px; font-weight: bold; }
		</style>
	</head>
	<body>
		<?php
			// se crea el primer arreglo
			print( "<p class = 'head'>Creando el primer arreglo</p>" );
			$first[ 0 ] = "cero";
			$first[ 1 ] = "uno";
			$first[ 2 ] = "dos";
			$first[] = "tres";
			// imprime el índice y valor de cada elemento
			for ( $i = 0; $i < count( $first ); ++$i )
				print( "Elemento $i es $first[$i]</p>" );
			print( "<p class = 'head'>Creando el segundo arreglo</p>" );
			// se invoca a la función array para crear el segundo arreglo
			$second = array( "cero", "uno", "dos", "tres" );

			for ( $i = 0; $i < count( $second ); ++$i )
				print( "Elemento $i es $second[$i]</p>" );

			print( "<p class = 'head'>Creando el tercer arreglo</p>" );
			// asigna valores a la entradas usando índices no numéricos
			$third[ "Paty" ] = 21;
			$third[ "Diana" ] = 18;
			$third[ "Ale" ] = 23;

			// itera en los elementos del arreglo e imprime cada
			// nombre y valor del elemento
			for ( reset( $third ); $element = key( $third ); next( $third ) )
				print( "<p>$element es $third[$element]</p>" );
```

```
                print( "<p class = 'head'>Creando el cuarto arreglo</p>" );

                // invoca a la función array para crear el cuarto arreglo usando
                // índices string
                $fourth = array(
                      "Enero" => "primer", "Febrero" => "segundo",
                      "Marzo" => "tercer", "Abril" => "cuarto",
                      "Mayo" => "quinto", "Junio" => "sexto",
                      "Julio" => "s&eacute;ptimo", "Agosto" => "octavo",
                      "Septiembre" => "noveno", "Octubre" => "d&eacute;cimo",
                      "Noviembre" => "onceavo","Diciembre" => "doceavo" );
                // imprime el nombre y valor de cada
                foreach ( $fourth as $element => $value )
                      print( "<p>$element es el $value mes</p>" );
          ?><!-- fin del script PHP -->
    </body>
</html>
```

6. Comparación de Cadenas

Muchas de las tareas de procesamiento de cadena se pueden realizar mediante los operadores de igualdad y relacionales. La función `strcmp` compara dos cadenas. La función devuelve `-1` si la primera cadena alfabéticamente precede a la segunda cadena, `0` si las cadenas son iguales, y `1` si la primera cadena le sigue a la segunda en orden alfabético.

EJEMPLO 5.
Archivo compara.php:

```
<!-- Uso de los operadores de comparación de cadenas. -->
<!DOCTYPE html>
<html>
    <head>
        <meta charset = "utf-8">
        <title>Comparaci&oacute;n de Cadenas</title>
        <style type = "text/css">
            p { margin: 0; }
        </style>
    </head>
    <body>
        <?php
            // se crea un arreglo de frutas
            $fruits = array( "apio", "naranja", "banana" );

            // iterar sobre cada elemento del arreglo
```

```
                for ( $i = 0; $i < count( $fruits ); ++$i ){
                // invoca a la función strcmp para comparar el elemento del arreglo
                // con la cadena "banana"
                if( strcmp( $fruits[ $i ], "banana" ) < 0 )
                      print( "<p>" . $fruits[ $i ] . " es menor que banana " );
                elseif ( strcmp( $fruits[ $i ], "banana" ) > 0 )
                      print( "<p>" . $fruits[ $i ] . " es mayor que banana ");
                else
                      print( "<p>" . $fruits[ $i ] . " es igual que banana " );

                // uso de los operadores relacionales para comparar cada elemento
                // con la cadena "apio"
                if ( $fruits[ $i ] < "apio" )
                      print( "y menor que apio!</p>" );
                elseif ( $fruits[ $i ] > "apio" )
                      print( "y mayor que apio!</p>" );
                elseif ( $fruits[ $i ] == "apio" )
                      print( "e igual que apio!</p>" );
                } // fin del for
          ?><!-- fin del script PHP -->
      </body>
</html>
```

7. Procesamiento de Cadenas con Expresiones Regulares

- Una expresión regular es una serie de caracteres que se utilizan para las plantillas de comparación de patrones en cadenas, archivos de texto y bases de datos.
- La función `preg_match` utiliza expresiones regulares para buscar una cadena con un patrón especificado. Si se encuentra un patrón, devuelve la longitud de la cadena coincidente.
- Cualquier cosa entre comillas simples en un enunciado de impresión no se interpola (a menos que las comillas simples se aniden en una cadena literal entre comillas dobles).
- La función `preg_match` recibe el patrón por buscar de una expresión regular y la cadena a buscar.
- Las expresiones regulares pueden incluir metacaracteres que especifiquen patrones. Por ejemplo, el símbolo del metacaracter caret (`^`) que coincida con el comienzo de una cadena, y el signo de pesos (`$`) que coincida con el final de una cadena. El metacaracter punto (`.`) que coincida con cualquier carácter excepto los saltos de línea.
- Las expresiones de corchete son listas de caracteres encerrados entre corchetes (`[]`) que coincidan con cualquier carácter individual de la lista. Los rangos pueden ser especificados por el suministro del principio y el final del rango separados por un guion (`-`).
- Los cuantificadores se utilizan en expresiones regulares para indicar con qué frecuencia un carácter, o conjunto de caracteres en particular, puede aparecer en una comparación.
- El tercer argumento opcional de la función `preg_match` es un arreglo que almacena coincidencias por cada enunciado en paréntesis de la expresión regular. El primer elemento almacena la cadena coincidente para todo el patrón, y el resto de elementos están indexados de izquierda a derecha.
- Para buscar varias instancias de un patrón, llamadas múltiples a `preg_match`, y eliminar instancias coincidentes antes de llamar a la función de nuevo, se utiliza la función `preg_replace`.
- Las clases `Character`, o conjuntos de caracteres específicos, están encerrados por los delimitadores `[:` y `:]`. Cuando esta expresión se coloca en otro conjunto de corchetes, es una expresión regular que coincide con todos los caracteres en la clase.
- Una expresión entre paréntesis que contiene dos o más clases `caracter` adyacentes entre los delimitadores de clase representa los conjuntos de caracteres combinados.

- La función `preg_replace` toma tres argumentos: el patrón por comparar, una cadena para reemplazar la cadena coincidente y la cadena a buscar. Se devuelve la cadena modificada.

Cuantificador	Compara
`{n}`	Exactamente n veces
`{m, n}`	Entre `m` y `n` veces, inclusive
`{n, }`	`n` o más veces
`+`	Una o más veces (lo mismo que `{1, }`)
`*`	Cero o más veces (lo mismo que `{0, }`)
`?`	Cero o más veces (lo mismo que `{0, 1}`)

Clase `Character`	Descripción
`alnum`	Caracteres alfanuméricos (por ejemplo, letras [a-zA-Z] o dígitos [0-9])
`alpha`	Caracteres de palabras (por ejemplo, [a-zA-Z])
`digit`	Dígitos
`space`	Espacio en blanco
`lower`	Letras minúsculas
`upper`	Letras mayúsculas

EJEMPLO 6.
Archivo expresion.php:

```
<!-- Expresiones regulares -->
<!DOCTYPE html>
<html>
	<head>
		<meta charset = "utf-8">
		<title>Expresiones regulares</title>
		<style type = "text/css">
			p { margin: 0; }
		</style>
	</head>
	<body>
		<?php
			$search = "Now is the time";
			print( "<p>Prueba de la cadena: '$search'</p>" );
			// invoca a preg_match para encontrar el patrón 'Now' en la búsqueda
			if( preg_match( "/Now/", $search ) )
				print( "<p>'Now' se encontr&oacute;.</p>" );
			// busca el patrón 'Now' en el inicio de la cadena
			if( preg_match( "/^Now/", $search ) )
				print( "<p>'Now' encontrado al inicio de la l&iacute;nea.</p>" );
			// busca el patrón 'Now' al final de la cadena
			if( !preg_match( "/Now$/", $search ) )
				print( "<p>'Now' se encontr&oacute; al final de la l&iacute;nea.
</p>" );
			// busca cualquier palabra que termina con 'ow'
			if ( preg_match( "/\b([a-zA-Z]*ow)\b/i", $search, $match ) )
				print( "<p>Palabra encontrada terminando en 'ow': ".
					$match[ 1 ]. "</p>" );
			// busca cualquier palabra que inicie con 't'
			print( "<p>Palabras encontradas que incian con 't': " );
			while ( preg_match( "/\b(t[[:alpha:]]+)\b/", $search, $match ) ){
				print( $match[ 1 ]. " " );
				// elimina la primera ocurrencia de una palabra que inicia
				// con 't' para encontrar otras instancias en la cadena
```

```
                        $search = preg_replace("/" . $match[ 1 ] . "/", "", $search);
                    } // fin del while
                    print( "</p>" );
                ?><!-- fin del script PHP -->
        </body>
</html>
```

8. Procesamiento de Formularios y la Lógica de Negocio

- Los arreglos superglobales son arreglos asociativos predefinidos por PHP que mantienen las variables adquiridas de entrada del usuario, el ambiente o el servidor web y se puede acceder desde cualquier ámbito variable.
- Los arreglos `$_GET` y `$_POST` recuperan información enviada al servidor por solicitudes HTTP `get` y `post`, respectivamente.
- Un script ubicado en un servidor web puede acceder a los datos del formulario enviado a la secuencia de comandos como parte de una solicitud.
- La función `die` termina la ejecución del script. El argumento opcional de la función es una cadena por mostrar o un número entero por devolver mientras exista el script.

Nombre de variable	Descripción
`$_SERVER`	Datos acerca del servidor actualmente activo.
`$_ENV`	Datos acerca del ambiente del cliente.
`$_GET`	Datos enviados al servidor por una solicitud `get`.
`$_POST`	Datos enviados al servidor por una solicitud `post`.
`$_COOKIE`	Datos contenidos en cookies en la computadora del cliente.
`$GLOBALS`	Arreglo que contiene todas las variables globales.

EJEMPLO 7.

Archivo formulario.html:

```
<!-- Formulario de HTML para capturar la entrada del usuario. -->
<!DOCTYPE html>
<html>
      <head>
            <meta charset = "utf-8">
            <title>Formulario de Ejemplo</title>
            <style type = "text/css">
                  label { width: 5em; float: left; }
            </style>
      </head>
      <body>
            <h1>Form de Registro</h1>
            <p>Por favor, llene todos los campos y clic en Registro.</p>

            <!-- envía los datos del formulario a form.php -->
            <form method = "post" action = "formulario.php">
                  <h2>Informaci&oacute; del Usuario</h2>
                  <!-- crea cuatro cajas de texto para la entrada del usuario -->
                  <div><label>Nombre:</label>
                        <input type = "text" name = "fname"></div>
                  <div><label>Apellido:</label>
                        <input type = "text" name = "lname"></div>
                  <div><label>Email:</label>
                        <input type = "text" name = "email"></div>
```

```
                  <div><label>Tel&eactue;fono:</label>
                        <input type = "text" name = "phone"
                              placeholder = "(555) 555-5555"></div>
                  </div>

                  <h2>Publicaciones</h2>
                  <p>De cu&aactue;l libro desea usted informaci&oacute;n?</p>

                  <!-- crea desplegable list conteniendo nombres de libros -->
                  <select name = "book">
                        <option>Internet and WWW How to Program</option>
                        <option>C++ How to Program</option>
                        <option>Java How to Program</option>
                        <option>Visual Basic How to Program</option>
                  </select>
                  <h2>Sistemas Operativos</h2>
                  <p>Cu&aactue;l sistema operativo utiliza usted?</p>

                  <!-- crea cinco radio buttons -->
                  <p><input type = "radio" name = "os" value = "Windows"
                              checked>Windows
                        <input type = "radio" name = "os" value = "Mac OS X">Mac OS X
                        <input type = "radio" name = "os" value = "Linux">Linux
                        <input type = "radio" name = "os" value = "Other">Other</p>

                  <!-- crea un botón submit -->
                  <p><input type = "submit" name = "submit" value = "Registro"></p>
            </form>
      </body>
</html>
```

Archivo formulario.php:

```
<!-- Procesa la información enviada desde formulario.html. -->
<!DOCTYPE html>
<html>
      <head>
            <meta charset = "utf-8">
            <title>Validaci&oacute;n del Formulaario</title>
            <style type = "text/css">
                  p     { margin: 0px; }
                  .error      { color: red }
                  p.head      { font-weight: bold; margin-top: 10px; }
            </style>
      </head>
      <body>
            <?php
                  // determina si el número telefónico es válido e imprime
                  // un mensaje de error si no lo es
                  if (!preg_match( "/^\([0-9]{3}\) [0-9]{3}-[0-9]{4}$/",
                        $_POST["phone"])){
                        print(   "<p   class   =   'error'>N&uacute;mero   telef&oacute;nico
inv&aacute;lido</p>
                              <p>Un  n&uacute;mero  telef&oacute;nico  v&aacute;lido  debe
tener la forma
                              (555) 555-5555</p><p>Clic en el bot&oacute;n Regresar,
                              ingresar un  n&uacute;mero telef&oacute;nico v&aacute;lido y
reenviar.</p>
```

```
                        <p>Gracias.</p></body></html>" );
                    die(); // terminate script execution
                }
            ?><!-- end PHP script -->
            <p>Hi <?php print( $_POST["fname"] ); ?>. Thank you for
                completing the survey. You have been added to the
                <?php print( $_POST["book"] ); ?>mailing list.</p>
            <p class = "head">The following information has been saved
                in our database:</p>
            <p>Name: <?php print( $_POST["fname"] );
                print( $_POST["lname"] ); ?></p>
            <p>Email: <?php print( "$email" ); ?></p>
            <p>Phone: <?php print( "$phone" ); ?></p>
            <p>OS: <?php print( $_POST["os"] ); ?></p>
            <p class = "head">This is only a sample form.
                You have not been added to a mailing list.</p>
      </body>
</html>
```

En el navegador se invoca al formulario.html, se ingresan los datos solicitados y se digita el botón Registro (Figura a). Si los datos son correctos, el formulario de PHP envía la imagen con los datos ingresados (Figura b).

Figura a.

Figura b.

Si el número telefónico es inválido se muestra la siguiente imagen (Figura c):

Figura c.

NOTA: Generar un reporte con formato de los ejercicios anteriores.