UCH UNIVERSIDAD DE CIENCIAS Y HUMANIDADES

PROGRAMACIÓN II

SEMANA 01
COMENZANDO CON PHP

Eric Gustavo Coronel Castillo
ecoronel@uch.edu.pe
gcoronelc.blogspot.com

Comenzando con PHP

En esta semana aprenderás los fundamentos del lenguaje PHP, que es la base para todo lo que desarrollaras con PHP.

Índice

1 Insertando Código PHP

Los scripts basados en PHP están insertados en el código HTML, y para esto tenemos tres formas, que se describen a continuación.

1.1 Caso 1

Esta primera forma sólo está disponible si se han habilitado las etiquetas cortas. Esto se puede hacer habilitando la directiva de configuración short_open_tag en el archivo de configuración de PHP.

Sintaxis

```
<?

   // Aquí se inserta el script PHP

?>
```

Por defecto la directiva short_open_tag está deshabilitada. Si desea usar PHP en conjunto con XML se recomienda que se mantenga deshabilitada, de modo que pueda usar <?xml ?> en forma directa.

- Valor por defecto de la directiva short_open_tag:

```
short_open_tag = Off
```

- Para habilitarla debe establecerla a On:

```
short_open_tag = On
```

Esta directiva afecta también shorthand <?=expression?>, la cual es idéntica a <? echo(expression) ?>. El uso de este atajo requiere que short_open_tag se encuentre habilitado.

> **Nota:**
>
> En general, se recomienda que la directiva **short_open_tag** se mantenga deshabilitada; además, muchos servicios de hosting la mantienen deshabilitada.

Ejemplo 1:

Este ejemplo ilustra el uso de etiquetas cortas para insertar código PHP, recuerde que debe habilitar la directiva short_open_tag.

La sentencia header(...) es para configurar la salida en el navegador en formato **UTF-8** para que se puedan apreciar los acentos.

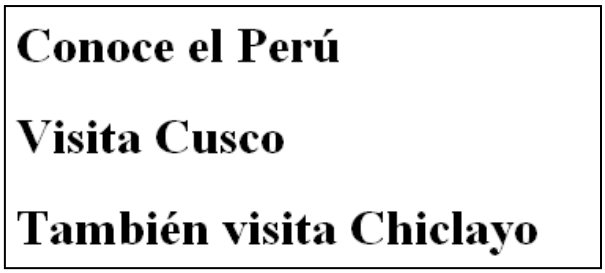
Archivo: prog2\sem01\ejm01.php

```
<?
   header('Content-Type: text/html; charset=utf-8' );
   echo "<h1>Conoce el Perú</h1>";
   echo "<h1>Visita Cusco</h1>";

   $destino2 = "Chiclayo";
?>
<h1>También visita <?=$destino2?></h1>
```

Para probar el ejemplo, en el campo dirección del navegador anote la siguiente URL:

```
http://localhost/prog2/sem01/ejm01.php
```

El resultado es el siguiente:



1.2 Caso 2

También denominadas etiquetas largas, es la forma estándar de insertar código PHP, no es necesario hacer configuración alguna para que funcione, cualquier instalación de PHP la soporta.

Sintaxis

```
<?php

   // Aquí se inserta el script PHP

?>
```

Ejemplo 2

Este ejemplo ilustra el uso de etiquetas largas para insertar código PHP.

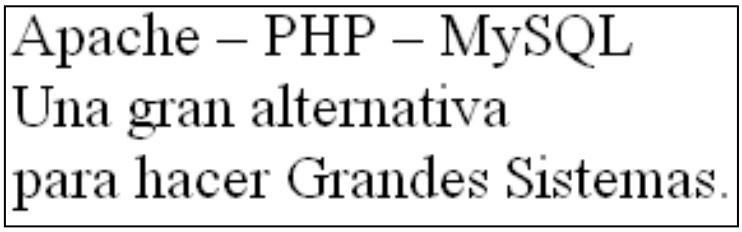
Archivo: prog2\sem01\ejm02.php

```
<?php
   header('Content-Type: text/html; charset=utf-8' );
   echo "Apache – PHP – MySQL<br>";
   echo "Una gran alternativa<br>";
   $msg = "para hacer Grandes Sistemas.";
?>
<?php echo($msg) ?>
```

Para probar el ejemplo, en el campo dirección del navegador anote la siguiente URL:

```
http://localhost/prog2/sem01/ejm02.php
```

El resultado es el siguiente:



1.3 Caso 3

La etiqueta script también permite insertar código PHP, pero tiene el inconveniente que no es muy práctica, por lo tanto su uso en aplicaciones reales es prácticamente nulo, solo se menciona por cultura informática.

Sintaxis

```
<script language=”php”>

   // Aquí se inserta el script PHP

</script>
```

Ejemplo 3

Este ejemplo ilustra el uso de la etiqueta script para insertar código PHP.

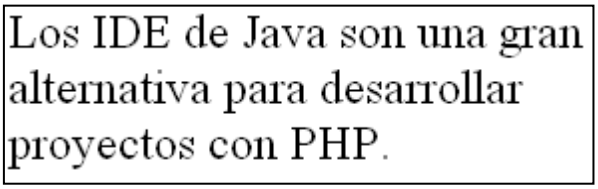
Archivo: prog02\sem01\ejm03.php

```
<script language="PHP">
   header('Content-Type: text/html; charset=utf-8' );
   echo( "Los IDE de Java son una gran<br>");
   echo("alternativa para desarrollar<br>");
   echo("proyectos con PHP.");
</script>
```

Para probar el ejemplo, en el campo dirección del navegador anote la siguiente URL:

```
http://localhost/prog02/sem01/ejm03.php
```

El resultado es el siguiente:



1.4 Caso 4

Para los que todavía recuerdan ASP 3.0, esta forma sólo está disponible si se han habilitado las etiquetas tipo ASP. Esto se puede hacer habilitando la directiva de configuración asp_tags en el archivo de configuración de PHP.

Sintaxis

```
<%

   // Aquí se inserta el script PHP

%>
```

Ejemplo 4

Este ejemplo ilustra el uso de etiquetas tipo ASP, recuerde que debe habilitar la directiva asp_tags.

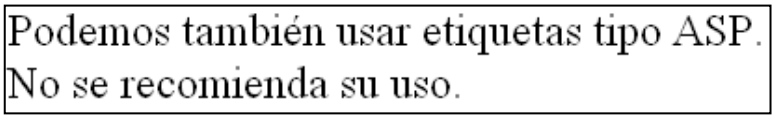
Archivo: prog2\sem01\ejm04.php

```
<%
   header('Content-Type: text/html; charset=utf-8' );
   echo("Podemos también usar etiquetas tipo ASP.<br>");
   echo("No se recomienda su uso.");
%>
```

Para probar el ejemplo, en el campo dirección del navegador anote la siguiente URL:

```
http://localhost/php100/prog2/sem01/ejm04.php
```

El resultado es el siguiente:



2 Instrucciones Básicas

2.1 Separación de Instrucciones

Las instrucciones se separan igual que en C o PERL, terminando cada sentencia con un punto y coma.

La etiqueta de cierre (?>) también implica el fin de la sentencia, así tenemos que el Ejemplo 5 es equivalente al Ejemplo 6.

Ejemplo 5

Archivo: prog2\sem01\ejm05.php

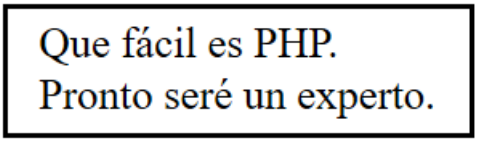
```
<?php
   header('Content-Type: text/html; charset=utf-8' );
   echo("Que fácil es PHP.<br>");
   echo("Pronto seré un experto.");
?>
```

Ejemplo 6

Archivo: prog2\sem01\ejm06.php

```
<?php
   header('Content-Type: text/html; charset=utf-8' );
   echo("Que fácil es PHP.<br>");
   echo("Pronto seré un experto.")
?>
```

En ambos ejemplos el resultado es:



2.2 Combinando PHP y HTML

Es posible combinar el código PHP y HTML para producir bloques HTML que serán parte de un condicional o un bucle, tal como se ilustra en los siguientes ejemplos.

Ejemplo 7

Archivo: prog2\sem01\ejm07.php

```
<?php
header('Content-Type: text/html; charset=utf-8' );
srand((double)microtime()*1000000);
$nota = rand(0,20);
echo("<h1>Nota: $nota</h1>");
?>
<?php if($nota<14) { ?>
   <h1>Estas Desaprobado.</h1>
<?php } else { ?>
   <h1>Felicitaciones Aprobaste.</h1>
<?php } ?>
```

En este ejemplo el HTML está dentro de un condicional, por lo tanto su ejecución depende en este caso del valor que toma la variable **$nota**. Su resultado podría ser así:

Ejemplo 8

Archivo: prog2\sem01\ejm08.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<table width="200" border="1">
   <tr>
      <th align="center">Número</th>
      <th align="center">Cuadrado</th>
   </tr>
   <?php for($k=1;$k<=5;$k++){ ?>
      <tr>
         <td align="center"><?php echo( $k ); ?></td>
         <td align="center"><?php echo( $k * $k ); ?></td>
      </tr>
   <?php } ?>
</table>
```

En este ejemplo el código HTML está dentro de un bucle PHP, por lo tanto se repite tantas veces como ejecuciones tenga el bucle; además dentro del código HTML tenemos también código PHP. Note que la segunda fila de la tabla es la que se encuentra dentro del bucle. El resultado es el siguiente:

Número	Cuadrado
1	1
2	4
3	9
4	16
5	25

2.3 Comentarios

PHP soporta comentarios tipo C, C++ y Shell de Unix.

Sintaxis 1

Comentario multilínea:

```
/*
   Inicio del comentario
   El comentario continúa en esta línea
   El comentario termina en esta línea
*/
```

Sintaxis 2

Comentario en línea – Caso 1

```
Sentencia; // Comentario en línea
```

Sintaxis 3

Comentario en línea – Caso 2

```
Sentencia; # Comentario en línea tipo Shell de Unix
```

Ejemplo 9

Archivo: prog2\sem01\ejm09.php

```
<?php
   /*
    * Este ejemplo ilustra el uso de Comentarios
    * Como podemos observar son tres los tipos
   */
   header('Content-Type: text/html; charset=utf-8' );
   echo("Ejemplos de Comentarios<br>");
   echo("PHP is Powerfull Campeón<br>"); // Mensaje Ganador
   echo("Perú Campeón"); # Esperanza de todos los peruanos
?>
```

El resultado es el siguiente:

2.4 Variables

Las variables en PHP no necesitan ser declaradas, podemos decir que en PHP las variables son débilmente tipadas.

Toda variable debe tener un nombre al que se le debe anteponer el símbolo $, además el nombre debe empezar con una letra.

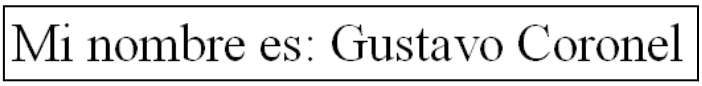
El ámbito de una variable es global a nivel del archivo actual y los archivos incluidos; dentro de una función son locales a la función.

Ejemplo 10

Archivo: prog2\sem01\ejm10.php

```
<?php
   header('Content-Type: text/html; charset=utf-8' );
   $nombre = "Gustavo Coronel";
   echo( "Mi nombre es: " . $nombre );
?>
```

En este ejemplo estamos creando la variable nombre, y luego estamos imprimiendo su contenido. El resultado es el siguiente:

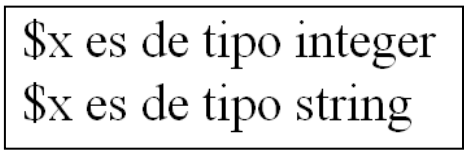


Ejemplo 11

Archivo: prog2\sem01\ejm11.php

```
1  <?php
2     header('Content-Type: text/html; charset=utf-8' );
3     $x = 20; # $x es un entero
4     echo("\$x es de tipo " . gettype($x) . "<br>");
5     $x = "Viva el Perú"; # $x es un cadena
6     echo("\$x es de tipo " . gettype($x) . "<br>");
7  ?>
```

En éste ejemplo, la variable $x cambia de tipo de dato, en la línea 3 es de tipo entero (integer) y en la línea 5 es de tipo cadena (string). El resultado es el siguiente:



3 Tipos de Datos

3.1 Enteros

Los enteros se pueden especificar usando una de las siguientes sintaxis:

```
$a = 4546; # número decimal

$a = -467; # un número negativo

$a = 0352; # número octal (equivalente a 234 decimal)

$a = 0xA5; # número hexadecimal (equivalente a 65 decimal)
```

Ejemplo 12

Archivo: prog2\sem01\ejm12.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Verdana">
<?php
   $a1 = 4546; # número decimal
   $a2 = -467; # un número negativo
   $a3 = 0352; # número octal (equivalente a 234 decimal)
   $a4 = 0xA5; # número hexadecimal (equivalente a 65 decimal)
   echo("a1 -> " . $a1 . "<br>");
   echo("a2 -> " . $a2 . "<br>");
   echo("a3 -> " . $a3 . "<br>");
   echo("a4 -> " . $a4 . "<br>");
?>
</font>
```

El resultado es el siguiente:

```
a1 -> 4546
a2 -> -467
a3 -> 234
a4 -> 165
```

3.2 Números de Punto Flotante

Los números en punto flotante (double) se pueden especificar utilizando cualquiera de las siguientes sintaxis:

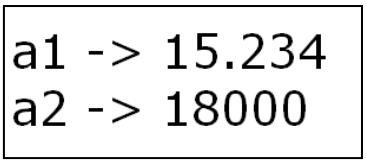
```
$a = 15.234;

$a = 1.8e4;
```

Ejemplo 13

Archivo: prog2\sem01\ejm13.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Verdana">
<?php
   $a1 = 15.234;
   $a2 = 1.8e4;
   echo("a1 -> " . $a1 . "<br>");
   echo("a2 -> " . $a2);
?>
</font>
```

El resultado es el siguiente:



3.3 Cadenas

Las cadenas se especifican usando como delimitadores la comilla simple (') o la comilla doble (").

Ejemplo 14

Archivo: prog2\sem01\ejm14.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   $cad = "\"Esto es una cadena\"";
   echo $cad;
?>
</font>
```

El resultado es el siguiente:

"Esto es una cadena"

El carácter de barra invertida (\) se puede utilizar para indicar caracteres especiales según el siguiente cuadro:

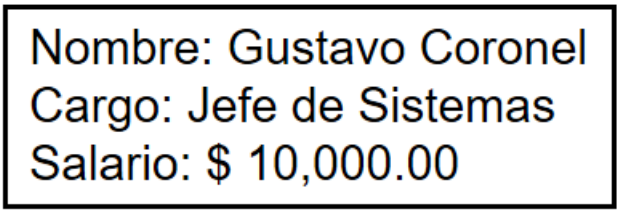
secuencia	significado
\n	Nueva línea
\r	Retorno de carro
\t	Tabulación horizontal
\\	Barra invertida
\$	Signo del dólar
\"	Comillas dobles

Ejemplo 15

Archivo: prog2\sem01\ejm15.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   $nombre = "Gustavo Coronel";
   $cargo = "Jefe de Sistemas";
   $salario = 10000.00;
   echo("Nombre: " . $nombre . "<br>");
   echo("Cargo: " . $cargo . "<br>");
   echo("Salario: \$ " . number_format($salario, 2, '.', ','));
?>
</font>
```

El resultado es el siguiente:



Cuando una cadena está limitada por comilla doble (“), las variables que están dentro de la cadena se expanden, esto quiere decir que son reemplazadas por su valor, tal como se ilustra en el siguiente ejemplo.

Ejemplo 16

Archivo: prog2\sem01\ejm16.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Verdana">
<?php
   $a = 5;
   $b = 9;
   $c = $a * $b;
   echo("Variables dentro de<br>");
   echo("una cadena se expanden<br>");
   echo("\$a = $a<br>");
   echo("\$b = $b<br>");
   echo("\$c = $c<br>");
?>
</font>
```

El resultado es el siguiente:

Variables dentro de
una cadena se expanden
$a = 5
$b = 9
$c = 45

También podemos delimitar las cadenas con comillas simples, pero en este caso las únicas secuencias permitidas son la doble barra invertida (\\) y la comilla simple (\').

Igualmente, se puede insertar código HTML como parte de la cadena, tal como se ilustra en el siguiente ejemplo.

Ejemplo 17

Archivo: prog2\sem01\ejm17.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<?php

   echo("<font face=\"Verdana\">\n");
   echo("<H1>Software Libre</H1>\n");
   echo("<H2>Es la Alternativa</H2>\n");
   echo("</font>");

?>
```

El resultado es el siguiente:

Si revisamos el código HTML que llega al navegador, obtenemos lo siguiente:

```
   <font face="Verdana">
   <H1>Software Libre</H1>
   <H2>Es la Alternativa</H2>
   </font>
```

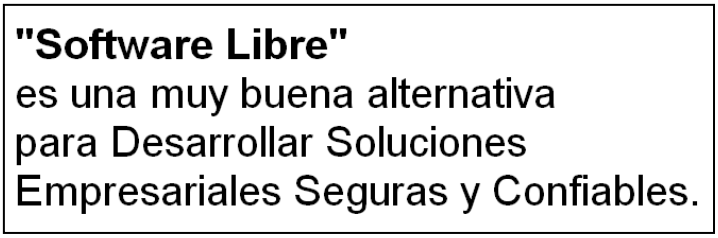
Para concatenar dos cadenas se utiliza el operador punto (.). A continuación tenemos un ejemplo ilustrativo.

Ejemplo 18

Archivo: prog2\sem01\ejm18.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   $s1 = "<b>\"Software Libre\"</b><br>";
   $s2 = "es una muy buena alternativa<br>";
   $s3 = "para Desarrollar Soluciones<br>";
   $s4 = "Empresariales Seguras y Confiables";
   echo($s1.$s2.$s3.$s4.".");
?>
</font>
```

El resultado es el siguiente:



3.4 Averiguar el Tipo de una Variable

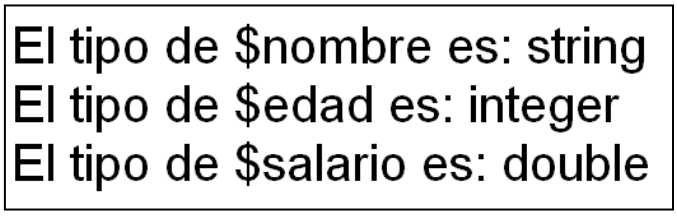
Una forma de determinar el tipo de dato de una variable es utilizando la función gettype(), tal como se ilustra en el siguiente ejemplo.

Ejemplo 19

Archivo: prog2\sem01\ejm19.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php

   $nombre = "Claudia";
   $edad = 25;
   $salario = 3500.00;
   echo("El tipo de \$nombre es: " . gettype($nombre) . "<br>");
   echo("El tipo de \$edad es: " . gettype($edad) . "<br>");
   echo("El tipo de \$salario es: " . gettype($salario));

?>
</font>
```

El resultado es el siguiente:



3.5 Valor Lógico (Boolean)

Este es el tipo más simple. Un boolean expresa un valor de verdad. Puede ser TRUE o FALSE.

Para especificar un literal boolean se usa las palabras reservadas TRUE o FALSE. Ambas son insensibles a mayúsculas y minúsculas.

Para convertir explícitamente un valor a boolean, se debe hacer casting entre tipos de datos usando bool o boolean. Sin embargo, en la mayoría de casos no es necesario hacer casting, ya que un valor será convertido automáticamente si un operador, función o estructura de control requiere un argumento tipo boolean.

Cuando se realizan conversiones a boolean, los siguientes valores son considerados FALSE:

- El literal boolean FALSE
- El integer 0 (cero)
- El float 0.0 (cero)
- El valor string vacío, y el string "0"
- Un array con cero elementos
- Un object con cero variables miembro (sólo en PHP 4)
- El tipo especial NULL (incluyendo variables no definidas)
- Objetos SimpleXML creados desde etiquetas vacías

Cualquier otro valor es considerado TRUE.

Ejemplo 20

Archivo: prog2\sem01\ejm20.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   $cad = "<b>Alianza Campeón</b><br>";
   echo($cad);
   if ($cad){
      $estado = "Es verdadero";
   }else{
      $estado = "Es falso";
   }
   echo($estado);
?>
</font>
```

En este ejemplo la variable **$cad** no está vacía, por lo tanto se interpreta como verdadera. El resultado es el siguiente:

3.6 Forzado de tipos

El forzado de tipos se conoce como casting y en PHP el funcionamiento es similar a como funciona en C, el nombre del tipo deseado se escribe entre paréntesis antes de la variable a la que se pretende forzar.

```
   $a = 10;   // $a es un entero
   $b = (double) $a;   // $b es un doble
```

Los casting permitidos son:

Casting	Descripción
(int), (integer)	Fuerza a entero (integer)
(real), (double), (float)	Fuerza a doble (double)
(string)	Fuerza a cadena (string)
(array)	Fuerza a array (array)
(object)	Fuerza a objeto (object)

Las tabulaciones y espacios se permiten dentro de los paréntesis, así que los siguientes ejemplos son funcionalmente equivalentes:

```
$a = (int) $b;
$a = ( int ) $b;
```

Puede no ser obvio que ocurrirá cuando se hace casting entre ciertos tipos. Por ejemplo, lo siguiente debería ser tenido en cuenta.

Cuando se hace casting de un escalar o una variable de cadena a un arreglo, la variable se convertirá en el primer elemento del arreglo.

Ejemplo 21

Archivo: prog2\sem01\ejm21.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   $cad = "Si lo intentas lo lograrás";
   $arr = (array) $cad;
   echo($arr[0]);
?>
</font>
```

El resultado es el siguiente:

Si lo intentas lo lograrás

Cuando se hace casting de una variable escalar o de una cadena a un objeto, la variable se convertirá en un atributo del objeto; el nombre del atributo será scalar.

Ejemplo 22

Archivo: prog2\sem01\ejm22.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   $msg = "<b>¡Es libre el que vive según su elección!</b>";
   $obj = (object) $msg;
   echo($obj->scalar);
?>
</font>
```

El resultado es el siguiente:

¡Es libre el que vive según su elección!

4 Impresión en el Navegador

4.1 Instrucción: echo

Imprime una o más cadenas. El uso de paréntesis es opcional.

Sintaxis

```
echo (string arg1, string [argn]…)
```

Ejemplo 23

Archivo: prog2\sem01\ejm23.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   echo "<b>Hola Mundo</b><br>";
   echo "Esto se extiende
   por varias líneas. <br>También puedes
   insertar código HTML";
?>
</font>
```

El resultado es el siguiente:

Hola Mundo
Esto se extiende por varias líneas.
También puedes insertar código HTML

4.2 Función: print()

Imprime una cadena.

Sintaxis

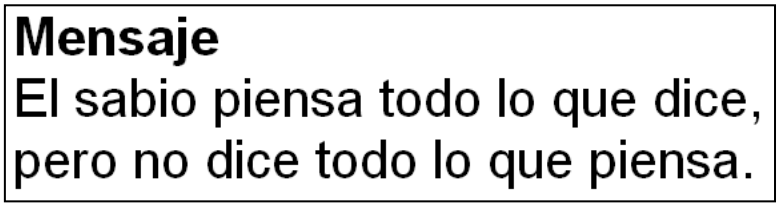
```
print (string arg)
```

Ejemplo 24

Archivo: prog2\sem01\ejm24.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   print("<b>Mensaje</b><br>");
   $cad = "El sabio piensa todo lo que dice,<br>";
   $cad = $cad . "pero no dice todo lo que piensa.";
   print($cad);
?>
</font>
```

El resultado es el siguiente:



4.3 Función: printf()

Imprime una cadena con formato.

Sintaxis

```
int printf (string formato [, mixed args…])
```

El formato debe tener el siguiente patrón:

```
%[carácter_de_relleno][ancho][.precisión]tipo
```

Los tipos posibles se especifican en la siguiente tabla:

Tipo	Descripción
b	El argumento es tratado como un entero y presentado como un número binario.
c	El argumento es tratado como un entero, y presentado como el caracter con dicho valor ASCII.
d	El argumento es tratado como un entero y presentado como un número decimal.
f	El argumento es tratado como un doble y presentado como un número de coma flotante
o	El argumento es tratado como un entero, y presentado como un número octal.
s	El argumento es tratado como una cadena y es presentado como tal.
x	El argumento es tratado como un entero y presentado como un número hexadecimal (con minúsculas).
X	El argumento es tratado como un entero y presentado como un número hexadecimal (con mayúsculas).

Ejemplo 25

Archivo: prog2\sem01\ejm25.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   $A = 10;
   $B = 15.5;
   $C = "MySQL la BD más rápida";
   printf("El valor de A es: %d<br>", $A);
   printf("Ahora relleno con ceros: %03d<br>", $A);
   printf("El valor de B es: %f<br>", $B);
   printf("Ahora relleno con ceros: %01.2f<br>", $B);
   printf("El valor de C es: %s<br>", $C);
?>
</font>
```

El resultado es el siguiente:

El valor de A es: 10
Ahora relleno con ceros: 010
El valor de B es: 15.500000
Ahora relleno con ceros: 15.50
El valor de C es: MySQL la BD más rápida

5 Constantes

PHP define varias constantes y proporciona un mecanismo para definir más en tiempo de ejecución. Las constantes son como las variables, salvo por dos circunstancias, que las constantes deben ser definidas usando la función *define()* y no pueden ser redefinidas con otro valor.

Las constantes especiales __FILE__ y __LINE__ son una excepción, ya que actualmente no lo son. Las constantes son sensibles a mayúsculas, por convención, los identificadores de constantes suelen declararse en mayúsculas.

Las constantes sólo pueden contener valores escalares: boolean, integer, float y string.

Ejemplo 26

Archivo: prog2\sem01\ejm26.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   define("PI",3.141516);
   $radio = 5;
   $area = PI * $radio * $radio;
   echo("PI: " . PI . "<br>");
   echo("Radio: $radio<br>");
   echo("Area: $area");
?>
</font>
```

El resultado es el siguiente:

PI: 3.141516
Radio: 5
Area: 78.5379

PHP ofrece un gran número de constantes predefinidas a cualquier programa en ejecución. Muchas de estas constantes, sin embargo, son creadas por diferentes extensiones, y solo estarán presentes si dichas extensiones están disponibles, bien por carga dinámica o porque han sido compiladas. Entre estas constantes podemos mencionar las siguientes:

Constante	Descripción
__FILE__	El nombre del archivo (programa.php) que está siendo interpretado actualmente. Si se usa dentro de un archivo que ha sido incluido o requerido, entonces se da el nombre del archivo incluido, y no el nombre del archivo padre.
__LINE__	El número de línea dentro del archivo que está siendo interpretado. Si se usa dentro de un archivo incluido o requerido, entonces se da la posición dentro del archivo incluido.
PHP_VERSION	La cadena que representa la versión del analizador de PHP en uso.
PHP_OS	El nombre del sistema operativo en el cuál se ejecuta el analizador PHP.
TRUE	Valor verdadero.
FALSE	Valor falso.
E_ERROR	Denota un error distinto de un error de interpretación del cual no es posible recuperarse.
E_WARNING	Denota una condición donde PHP reconoce que hay algo erróneo, pero continuará de todas formas; pueden ser capturados por el propio programa de PHP.
E_PARSE	El intérprete encontró sintaxis inválida en el programa PHP. La recuperación no es posible.
E_NOTICE	Ocurrió algo que pudo ser o no un error. La ejecución continúa.

Ejemplo 27

Archivo: prog2\sem01\ejm27.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   function msgError($file, $line, $message) {
      echo("<b>ERROR</b><br>");
      echo("<b>Archivo:</b> $file<br>");
      echo("<b>Linea:</b> $line<br>");
      echo("<b>Mensaje:</b> $message");
   }

   msgError(__FILE__,__LINE__, "Algo está mal!!!");
?>
</font>
```

El resultado es el siguiente:

ERROR
Archivo: C:\wamp\www\prog2\sem01\ejm27.php
Linea: 15
Mensaje: Algo está mal!!!

Ejemplo 28

Archivo: prog2\sem01\ejm28.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
   <body text="Navy">
      <table border="2" width="400">
         <tr>
            <th>Constante</th>
            <th>Valor</th>
         </tr>
         <tr>
            <td>PHP_VERSION</td>
            <td><?php echo( PHP_VERSION ) ?></td>
         </tr>
         <tr>
            <td>PHP_OS</td>
            <td><?php echo( PHP_OS ) ?></td>
         </tr>
      </table>
   </body>
</font>
```

El resultado es el siguiente:

Constante	Valor
PHP_VERSION	5.5.12
PHP_OS	WINNT

6 Expresiones y Operadores

Sin duda alguna, las expresiones constituyen la base de todo lenguaje, estas se construyen en base a operadores.

6.1 Operadores Aritméticos

Recordemos la aritmética básica de colegio.

Ejemplo	Nombre	Resultado
$a + $b	Adición	Suma de $a y $b.
$a - $b	Substracción	Diferencia entre $a y $b.
$a * $b	Multiplicación	Producto de $a por $b.
$a / $b	División	Cociente de $a entre $b.
$a % $b	Módulo	Resto de $a dividido entre $b.

6.2 Operadores de Asignación

Estos operadores permiten asignar un valor a una variable.

Operador	Ejemplo	Equivalente
=	$a = 7;	
+=	$a += 3;	$a = $a + 3;
-=	$a -= 3;	$a = $a - 3;
*=	$a *= 3;	$a = $a * 3;
/=	$a /= 3;	$a = $a / 3;
%=	$a %= 3;	$a = $a % 3;

Ejemplo 29

Archivo: prog2\sem01\ejm29.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   $a = 10;
   $b = 40;
   echo("\$a = $a<br>");
   echo("\$b = $b<br>");
   $b += $a;
   echo("El nuevo valor de \$b es $b");
?>
</font>
```

El resultado es el siguiente:

$a = 10
$b = 40
El nuevo valor de $b es 50

6.3 Operadores de Comparación

Permiten comparar dos valores; el resultado es un valor de tipo boolean, TRUE o FALSE.

Ejemplo	Nombre	Resultado
$a == $b	Igualdad	Cierto si $a es igual a $b.
$a === $b	Identidad	Cierto si $a es igual a $b y si son del mismo tipo (sólo PHP4 ó superior)
$a != $b	Desigualdad	Cierto si $a no es igual a $b.
$a < $b	Menor que	Cierto si $a es estrictamente menor que $b.
$a > $b	Mayor que	Cierto si $a es estrictamente mayor que $b.
$a <= $b	Menor o igual que	Cierto si $a es menor o igual que $b.
$a >= $b	Mayor o igual que	Cierto si $a es mayor o igual que $b.

Otro operador condicional es el operador "?:" (o ternario), que funciona como en C y otros muchos lenguajes.

```
(expr1) ? (expr2) : (expr3);
```

La expresión toma el valor **expr2** si **expr1** se evalúa como verdadera, y toma el valor **expr3** si **expr1** se evalúa como falso.

6.4 Operador de Ejecución

PHP soporta un operador de ejecución: el apóstrofe invertido (`). ¡No son apóstrofes normales! PHP intentará ejecutar la instrucción contenida dentro de los apóstrofes invertidos como si fuera un comando del shell; y su salida devuelta como el valor de esta expresión.

Ejemplo 30

Archivo: prog2\sem01\ejm30.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   $output = `dir ejm*`;
   echo("<pre>$output</pre>");
?>
</font>
```

Una parte del resultado se ilustra a continuación:

```
 El volumen de la unidad C es SYSTEM
 El número de serie del volumen es: DAE1-5300

 Directorio de C:\wamp\www\prog2\sem01

21/07/2016  06:18 a.m.               411 ejm01.php
21/07/2016  06:18 a.m.               405 ejm02.php
21/07/2016  06:21 a.m.               422 ejm03.php
21/07/2016  06:23 a.m.               415 ejm04.php
14/08/2016  04:52 p.m.               220 ejm05.php
14/08/2016  04:55 p.m.               281 ejm06.php
21/07/2016  05:37 p.m.               370 ejm07.php
14/08/2016  10:09 p.m.               441 ejm08.php
21/07/2016  08:25 p.m.               415 ejm09.php
21/07/2016  05:14 p.m.               220 ejm10.php
21/07/2016  11:30 p.m.               324 ejm11.php
22/07/2016  01:03 a.m.               554 ejm12.php
22/07/2016  01:21 a.m.               316 ejm13.php
21/07/2016  11:56 p.m.               272 ejm14.php
14/08/2016  09:18 p.m.               454 ejm15.php
```

6.5 Operadores de Incremento/Decremento

PHP soporta los operadores de pre y post incremento y decremento, similar C.

Ejemplo	Nombre	Efecto
++$a	Pre-incremento	Incrementa $a en uno y después devuelve $a.
$a++	Post-incremento	Devuelve $a y después incrementa $a en uno.
--$a	Pre-decremento	Decrementa $a en uno y después devuelve $a.
$a--	Post-decremento	Devuelve $a y después decrementa $a en uno.

Ejemplo 31

Archivo: prog2\sem01\ejm31.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   $a = 15;
   echo( "\$a = " . ++$a . "<br>" );
   echo( "\$a = " . $a++ . "<br>" );
   echo( "\$a = " . $a );
?>
</font>
```

El resultado es el siguiente:

$a = 16
$a = 16
$a = 17

6.6 Operadores Lógicos

Permiten construir expresiones lógicas compuestas.

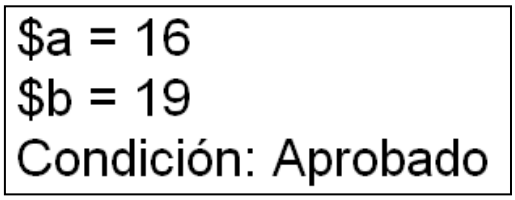
Ejemplo	Nombre	Resultado
$a and $b	Y	Verdadero si $a y $b son verdaderos.
$a or $b	O	Verdadero si $a o $b es verdadero.
$a xor $b	O exclusiva	Verdadero si $a es verdadero o $b es verdadero, pero no ambos a la vez.
! $a	Negación	Verdadero si $a es falso.
$a && $b	Y	Verdadero si $a y $b son verdaderos.
$a || $b	O	Verdadero si $a o $b es verdadero

Ejemplo 32

Archivo: prog2\sem01\ejm32.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   $a = rand(0,20);
   $b = rand(0,20);
   if($a>14 and $b>14){
      $cond = "Condición: Aprobado";
   }
   else{
      $cond = "Condición: Desaprobado";
   }
   echo( "\$a = $a<br>" );
   echo( "\$b = $b<br>" );
   echo( $cond );
?>
</font>
```

El resultado es el siguiente:



6.7 Operadores de Cadena

6.7.1 Operadores de Concatenación

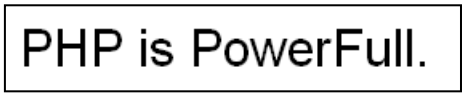
Para concatenar dos cadenas utilizamos el operador Punto (.).

Ejemplo 33

Archivo: prog2\sem01\ejm33.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Helvetica">
<?php
   $cad1 = "PHP is ";
   $cad2 = "PowerFull.";
   echo( $cad1 . $cad2 );
?>
</font>
```

El resultado es el siguiente:



6.7.2 Operador de Concatenación y Asignación

Este es el operador punto e igual (.=), agrega a una cadena, otra cadena.

Ejemplo 34

Archivo prog2\sem01\ejm34.php

```
<?php header('Content-Type: text/html; charset=utf-8' ); ?>
<font face="Arial">
<?php
   $cad = "Este es el equipo: ";
   $cad .= "Gustavo, Sergio, y Ricardo";
   echo( $cad );
?>
</font>
```

El resultado es el siguiente:

7 Práctica de Laboratorio

7.1 Revisar la instalación de PHP

Desarrollo el siguiente programa:

Programa: info.php

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

Luego de ejecutar el programa info.php proceda a revisar la configuración de PHP.

7.2 Generar Tabla de Multiplicar

Tomando como base el ejemplo 8, desarrolle un programa que genere la tabla de multiplicar del número **N**.

El valor de **N** debe ser generado aleatoriamente.

7.3 División de Dos Números

Desarrolle un programa que permita encontrar el cociente y el residuo de una división entera.