Programación de código embebido en páginas web (PHP)

Corregido

Nombre:

Joaquín Soroya

Paseo por la playa

Contenido

1 Desarrollo de código embebido en lenguajes de marcas (PHP)

- ¿A qué se refiere el término "código embebido o incrustado" en lenguajes de marcas?.

El código embebido o incrustado se refiere a la disposición del código de un lenguaje de "scripting" tal como Javascript o PHP intercalada dentro del contenido de un archivo HTML.

La ejecución de dicho código se realiza durante el proceso de interpretación del contenido HTML y se realiza en orden secuencial desde el comienzo del archivo hasta el final.

- Pega un ejemplo de archivo HTML con código embebido (puede ser javascript o PHP).

Busca un ejemplo de archivo HTML

- La actuación de un servidor web es distinta cuando un cliente le pide un archivo con extensión ".html" y ".php". Explica dicha diferencia.

EL servidor tiene un comportamiento distinto dependiendo de la extensión del archivo solicitado:

Si se solicita un archivo con extensión ".html", el servidor simplemente sirve dicho archivo al cliente para que lo procese en el navegador correspondiente y muestre el contenido al usuario.

Si se solicita un archivo con extensión ".php", el servidor establece un entorno de ejecución con el motor Zend (entorno de ejecución de PHP) y ejecuta el código PHP embebido

- ¿Qué etiqueta indica en un archivo HTML que se va a introducir código PHP?.

La pareja de etiquetas son:

<?php

?>

- Fíjate en las figuras de la sección "Esquemas de diferentes peticiones de páginas WEB" del capítulo "Introducción". Explica las diferencias entre los tres esquemas.

Figura 1:

En este escenario el navegador realiza su petición de recurso al servidor web. El servidor web identifica el recurso solicitado y se lo sirve al navegador. El archivo HTML enviado por el servidor llega al cliente y el navegador lo interpreta mostrándolo al usuario.

Figura 2:

En este escenario se realiza la petición por parte del navegador y el servidor web le entrega el archivo HTML. Dicho archivo contiene código embebido escrito en javascript que es interpretado en el equipo donde corre el navegador. El resultado de la ejecución del código es finalmente mostrado al usuario.

Figura 3:

En este escenario se realiza una petición por parte del navegador y el recurso solicitado contiene código embebido escrito en PHP. Dicho código es ejecutado por el entorno de ejecución de PHP que reside en el equipo del servidor y produce un archivo html con código embebido en Javascript. Dicho archivo llega al navegador donde se interpreta el código HTML y se ejecuta el código Javascript.

2 Instalación del entorno de desarrollo Netbeans 7.2

❖ Haz tick en cada una de las siguientes acciones:

Acción de Configuración	Check
Instalación Netbeans 7.2	X
Instalación Netbeans paquete PHP	X
Copia de XAMPP en D:/2DAW	X
Ejecución del script setup_xampp.bat	X
Cambio de PHP.ini para la ejecución de XDEBUG: xdebug.remote_enable = on zend	X
Asegurarse que el camino a PHP es correcto Options-PHP-General (PHP5 Interpreter)	X

❖ Lista las versiones de los componentes del paquete de XAMPP instalado en tu PC

- Apache 2.4.3
- MySQL 5.5.27
- PHP 5.4.7
- phpMyAdmin 3.5.2.2
- FileZilla FTP Server 0.9.41
- Tomcat 7.0.30 (with mod_proxy_ajp as connector)
- Strawberry Perl 5.16.1.1 Portable
- XAMPP Control Panel 3.1.0 (from hackattack142)

- Crea un proyecto PHP llamado "Factorial". Utiliza todos los valores por defecto a excepción del campo "Run As" en la sección "Run Configuration". Selecciona el valor"PHP Built-in Web Server".

Crea el proyecto en Netbeans

- Añade un archivo fuente al proyecto llamado factorial.php. Copia el código que aparece a continuación:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
  <html>
    <head>
            <meta http-equiv="Content-Type" content="text/html; charset=UTF-
8">
                    <title>NetBeans PHP debugging sample</title>
            </head>
<body>
    <?php
    $m=5;
    $n=10;
      $sum_of_factorials = calculate_sum_of_factorials ($m, $n);
      echo "The sum of factorials of the entered integers is " .
$sum_of_factorials;

        function calculate_sum_of_factorials ($argument1, $argument2) {
            $factorial1 = calculate_factorial ($argument1);
            $factorial2 = calculate_factorial ($argument2);
            $result = calculate_sum ($factorial1, $factorial2);
            return $result;
            }

      function calculate_factorial ($argument) {
            $factorial_result = 1;
            for ($i=1; $i<=$argument; $i++) {
                    $factorial_result = $factorial_result*$i;
            }
                    return $factorial_result;
            }

        function calculate_sum ($argument1, $argument2) {
                    return $argument1 + $argument2;
            }
?>
  </body>
</html>
```

Ejecuta este código y pega el resultado.

The sum of factorials of the entered integers is 3628920

- Inicia una sesión de depuración en netbeans para la ejecución de dicho archivo. Pega una imagen de la pantalla mientras realizas la ejecución paso a paso del código:

Pega un pantallazo que muestre el modo de depuración

3 Mi primer programa PHP

- php11.pdf. Escribe un programa php que muestre en pantalla el mensaje: Hola a todos y todas, utilizando la función echo.

```
<?php
    echo "Hola a todos y todas";
?>
```

- php11.pdf. Escribe un programa php que muestre en pantalla el mensaje: Hola a "todos" y "todas". Fíjate que las comillas deben aparecer en el mensaje.

```
<?php
    echo "Hola a \"todos\" y \"todas\"";
?>
```

- php29.pdf. ¿Qué diferencias existen entre la función echo() y la función print()?.

La principal diferencia es que print devuelve el valor 1 y echo no devuelve ningún valor.

Además echo es más rápido que print ya que no devuelve nada

La función print toma como parámetro un solo argumento mientras que echo puede tomar un número indefinido de valores separados por comas.

- php29.pdf. Explica el uso de la función printf. Escribe un programa que muestre 4 ejemplos de salida con formato.

La función printf permite realizar una presentación de información con formato. Incluye multiples posibilidades para definir el relleno, alineaciones, anchos ..etc.

Busca 4 ejemplos similares a los que aparecen en la documentación.

- php12.pdf. ¿Cómo se pueden añadir comentarios a un programa PHP?. Escribe un ejemplo de cada posibilidad.

Se puede comentar una única línea con los caracteres // y #.

También se puede hacer un comentario varia líneas utilizando los caracteres /* y */ a los extremos del comentario.

4 Constantes y variables

- php13.pdf. ¿Qué es una constante?.

Una constante almacena un valor que no va a variar durante la ejecución de programa.

- php13.pdf. Escribe un programa que defina constantes para los números pi y número aureo (con 4 decimales) y luego los muestre en pantalla siguiendo el esquema del ejemplo 8.

```
<?php
        define("PI", 3.1415);
        define("NUM_AUREO", 1.1680);
        echo "El valor de la constante PI es: ", PI, "<br>";
        echo "El valor del numero AUREO es: ", NUM_AUREO;
    ?>
```

- php14.pdf ¿Qué es una variable?.

Una variable es un espacio en memoria destinado a almacenar un valor y que puede ser consultado o manipulado dentro de las instrucciones de un programa.

- php15.pdf Escribe un pequeño programa que muestre el valor de las constantes predefinidas PHP_OS y PHP_VERSION.

```
<?php
        echo "Datos OS: ", PHP_OS, "<br>";
        echo "Datos PHP: ", PHP_VERSION;
 ?>
```

❖ php14.pdf ¿Cómo se nombran a las variables?.

Todos los nombres de variables deben comenzar con el carácter $ y el primer carácter después del $ debe ser una letra.

❖ php14.pdf Explica cómo funciona en PHP el ámbito de las variables. ¿Qué son las variables globales y superglobales?

Las variables definidas en un script (que no sea dentro de una función) pueden utilizarse en cualquier parte de ese script.

Para acceder a esas variables dentro de una función se han de definir previamente como globales con la instrucción "global". A excepción de las variables superglobales que pueden usarse en cualquier lugar, incluso dentro de las funciones de usuario sin necesidad de declararlas como tal.

❖ php14.pdf. Ejecuta el script del ejemplo 9. Copia el resultado y explícalo con detalle.

Ejecuta el script y explícalo como aparece en la documentación

❖ php16.pdf. Lista y describe las variables superglobales

Consulta la documentación.

- php16.pdf. ¿Qué diferencia existe entre las variables superglobales y las variables predefinidas?

Las variables predefinidas incluyen a todas las variables superglobales y a un grupo de variables que no son accesibles desde cualquier lugar de nuestro script. Esas variables son:

$php_errormsg, $http_response_header, $argc, $argv y $HTTP_RAW_POST_DATA

- php16.pdf. Explica 5 variables de servidor y 5 variables de entorno que te hayan parecido interesantes.

Consulta la documentación y escribe sobre las variables escogidas

- php16.pdf. ¿Para qué sirven las variables $_POST y las variables $_GET?.

Estas variables superglobales permiten al script PHP acceder a los valores en forma de parámetros enviados desde el navegador en mensajes HTTP de tipo GET y POST.

- php16.pdf. ¿Qué aportan las variables $GLOBALS?.

La variable superglobal $GLOBALS recoge en un array asociativo todas las variables globales del script en curso.

- php17.pdf. ¿Qué es una variable estática?.

Una variable estática conserva su valor entre distintas ejecuciones de la misma función dentro de un script.

❖ php17.pdf. Ejecuta el script del ejemplo 11. Copia el resultado y explícalo con detalle.

Ejecuta el script y explícalo como aparece en la documentación

❖ php18.pdf. ¿Qué tipos de datos puede almacenar una variable?.

Una variable puede almacenar valores enteros, decimales, cadenas de caracteres y valores lógicos o booleanos.

❖ php18.pdf. Escribe un programa que utilice una variable para cada tipo posible y posteriormente se muestre el tipo de cada una de las variables en pantalla.

```
<?php
    $entero = 5;
    $decimal = 5.7;
    $cadena = "palabra";
    $logico = True;
    echo 'El tipo de la variable $entero es: ', gettype($entero), "<br>";
    echo 'El tipo de la ariable $decimal es: ', gettype($decimal), "<br>";
    echo 'El tipo de la variable $cadena es: ', gettype($cadena), "<br>";
    echo 'El tipo de la variable $logico es: ', gettype($logico);
?>
```

- php18.pdf. En qué consiste el forzado de tipos. Escribe un programa que fuerze el tipo Integer (entero) para el valor 6.5, el tipo String(cadena) para el valor 23454.

Una variable asume su tipo cuando se le asigna un valor. Esta asignación por defecto puede ser modificada por el programador mediante el forzado de tipos. Para realizar esta operación se hace explícito el tipo que debe asumir la variable.

```
<?php

   $var1 = 6.5;

   $var2 = "23456";

   echo 'El tipo de la variable $var1 es: ', gettype($var1), " y su valor es: ", $var1, "<br>";

  $var1 = (integer)$var1;

   echo 'El tipo de la variable $var1 es: ', gettype($var1), " y su valor es: ", $var1, "<br>";

   echo 'El tipo de la variable $var2 es: ', gettype($var2), " y su valor es: ", $var2, "<br>";

  $var2 = (integer)$var2;

   echo 'El tipo de la variable $var2 es: ', gettype($var2), " y su valor es: ", $var2, "<br>";



?>
```

- php18.pdf. ¿Se puede sumar una variable de tipo cadena con una variable de tipo entero?.

Si, el lenguaje tiene mecanismos para que se puedan convertir los tipos de las variables de tal manera que se puedan llevar a cabo la operación requerida. En este caso se forzará un cambio de tipo de string a entero.

- php44.pdf. Escribe un programa que muestre ejemplos de las funciones disponibles para obtener información sobre las variables. (is_<tipo>)

Consultar la documentación y escribir funciones similares

5 Operaciones aritméticas y operadores de incremento

- php20.pdf. Escribe un programa que defina dos variables numéricas y realice todas las operaciones aritméticas soportadas por PHP mostrando todos los resultados en la ventana de salida.

```
<?php
    $a = 25;
    $b = 5;
     echo "$a + $b = ", $a+$b, "<br>";
    echo "$a - $b = ", $a-$b, "<br>";
    echo "$a * $b = ", $a*$b, "<br>";
    echo "$a / $b = ", $a/$b, "<br>";
    echo "$a^$b = ", pow($a, $b), "<br>";
    echo "$a % $b (modulo de a entre b) = ", $a%$b, "<br>";
    echo "raiz cuadrada de $a = ", sqrt($a), "<br>";
    echo "Potencia inversa $b de $a = ", pow($a, 1/$b);
?>
```

- php23.pdf. Escribe un programa que genere números aleatorios entre el 1 y el 150.

```
<?php
    $min = 1;
    $max = 150;
    $n = 15;
    echo "Numeros aleatorios:", "<br>";
    for ($i = 0; $i < $n; $i++) {
        echo mt_rand($min, $max), "<br>";
    }
?>
```

- php36.pdf. ¿Qué diferencia existe entre los operadores pre-incremento y operadores post-incremento?.

Los operadores de pre-incremento realizan el incremento antes de ejecutar la instrucción sin embargo los operadores de post-incremento lo hacen después de ejecutar la instrucción.

- php36.pdf. Escribe un programa con un ejemplo de todos los operadores de incremento.

Consulta la documentación y escribe instrucciones similares

6 Operadores de comparación y lógicos

- php34.pdf. ¿Qué diferencia existe entre el operador "==" y el operador "===". Escribe un programa que muestre dicha diferencia.

El operador "==" compara los valores de las variables incluso si son distinto tipo forzando el tipo de alguno de ellos.

El operador "===" compara los valores y los tipos de las variables de tal manera que deben coincidir en el valor y el tipo.

- php34.pdf. Escribe un programa que utilice cada uno de los operadores de comparación y por cada operador muestre un ejemplo en el que devuelva los valores posibles, es decir 1 y NUL.

Consulta la documentación y escribe instrucciones similares

- php35.pdf. Escribe una condición lógica que sólo la cumplan los números mayores de 100 y menores de 200.

```
(($num > 100) && ($num < 200))
```

- php35.pdf. Escribe una condición que sólo la cumplan los números menores de 100 o mayores de 200.

```
(($num < 100) || ($num > 200))
```

- php35.pdf. Escribe una condición que sólo la cumplan las cadenas de longitud 20 y que empiecen por la palabra "Un".

```
((strlen($cadena) ==20 )&&(substr($cadena,0,2) == "Un"))
```

- php35.pdf. Dadas dos variables numéricas escribe una condición lógica que se cumpla simplemente si uno de los valores es superior a 100 y el otro es superior a 500.

```
(($num1 > 100)&&($num2 > 500))||(($num1 > 500)&&($num2 > 100))
```

- php35.pdf. Escribe una condición que sólo la cumplan los años bisiestos.

```
(($año % 4 == 0 && $año % 100 != 0) || ($año % 400 == 0))
```

7 Cadenas

- php24.pdf. Escribe un programa que utilice la sintaxis de documento incrustado para asignar a una variable cadena el texto:

  "En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor.".

```
<?php
  $cadena = <<< Texto
En un lugar de la mancha, de cuyo nombre no quiero acordarme, no ha mucho que vivia un hidalgo de los de lanza en astillero rocin flaco, adarga antigua y galgo corredor.
Texto;
    echo $cadena;
?>
```

- php25.pdf. Escribe un programa que dadas dos variables de tipo cadena con los valores "Hola" y "mundo", asigne a una tercera variable la concatenación de las dos primeras, mostrando todos los valores en la ventana de salida.

```
<?php
  $cadena1 = "Hola";
  $cadena2 = "mundo";
  $cadena12 = $cadena1.$cadena2;
   echo $cadena12;
?>
```

- php25.pdf. Escribe un programa que dadas dos variables de tipo cadena con los valores "Hola" y "mundo", asigne a la primera variable la concatenación de ambas, mostrando todos los valores en la ventana de salida.

```
<?php
  $cadena1 = "Hola";
  $cadena2 = "mundo";
  $cadena1 .= $cadena2;
   echo $cadena1;
?>
```

- php30.pdf. Escribe un programa que utilice el primer programa de esta sección y realice las siguientes operaciones sobre la cadena:
  - Devuelve la longitud (número de caracteres) del texto
  - Pasa el texto a mayúsculas
  - Pasa el texto a minúsculas
  - Hace que todas las palabras empiecen por mayúscula
  - Devuelve el texto menos los 18 primeros caracteres
  - Devuelve el texto menos los 10 últimos caracteres
  - Devuelve el texto invertido
  - Devuelve el texto repetido 3 veces
  - Devuelve el texto con la palabra "no" sustituida por la palabra "si"
  - Devuelve el texto con un guión insertado cada 10 caracteres.
  - Devuelve las 5 primeras palabras en líneas distintas y el resto del texto en una última línea.
  - Devuelve el texto encriptado
  - Aplica el algoritmo md5 al texto para generar una huella digital

```
<?php
  $cadena = <<< Texto
En un lugar de la mancha, de cuyo nombre no quiero acordarme
no ha mucho que vivia un hidalgo de los de lanza en astillero
rocin flaco, adarga antigua y galgo corredor.
Texto;
  echo "Numero de caracteres del texto:", "<br>", strlen($cadena),
"<br>";
  echo "Texto en mayusculas:", "<br>", strtoupper($cadena), "<br>";
  echo "Texto en minusculas:", "<br>", strtolower($cadena), "<br>";
  echo "Primera letra de todas las palabras convertida a mayuscula:",
"<br>", ucwords($cadena), "<br>";
  echo "Primeros 18 caracteres del texto:", "<br>", substr($cadena, 17,
strlen($cadena)-17), "<br>";
  echo "Ultimos 10 caracteres del texto:", "<br>", substr($cadena, 0,
strlen($cadena)-10), "<br>";
  echo "Texto al reves:", "<br>", strrev($cadena), "<br>";
  echo "Repeticion del texto 3 veces:", "<br>", str_repeat($cadena, 3),
"<br>";
  echo "Reemplazo de los 'no' por 'si':", "<br>", str_replace("no", "si",
$cadena), "<br>";
  echo "Introduccion de un guion cada 10 caracteres:", "<br>",
chunk_split($cadena, 10, "-"), "<br>";
  $trozos = explode(" ", $cadena, 6);
  echo "Primeras cinco palabras en lineas distintas y el resto en una
linea:", "<br>";
  for ($i = 0; $i < sizeof($trozos); $i++) {
     echo $trozos[$i], "<br>";
  }
  echo "<br>";
  echo "Texto encriptado:", "<br>", crypt($cadena), "<br>";
  echo "Texto encriptado con md5:", "<br>", md5($cadena), "<br>";
?>
```

8 Arrays

- php26.pdf. ¿Qué es una array?.

> Un array es una estructura de datos que permite agrupar valores en posiciones distintas de la estructura. La manera de acceder a los valores de dichas variables es mediante índices numéricos o simbólicos.

- php26.pdf. Escribe un programa que cree un array escalar con 5 colores y después lo muestre en la ventana de salida.

```
<?php
  $colores[0] = "rojo";
  $colores[1] = "verde";
  $colores[2] = "azul";
  $colores[3] = "amarillo";
  $colores[4] = "negro";

  for ($i = 0; $i < sizeof($colores); $i++) {
     echo "El color de la posicion $i es: ", $colores[$i], "<br>";
  }
?>
```

- php26.pdf. Ejecuta el script del ejemplo 19. Copia el resultado y explícalo con detalle.

> Ejecuta el script y explícalo como aparece en la documentación

❖ php27.pdf. Escribe un programa que rellene un array bidimensional con los resultados de una pequeña liguilla de tres equipos de fútbol: Real Madrid, Manchester Utd y AC Milan. Los resultados tiene el formato "golescasa-golesvisitante" Con dicha tabla el programa deberá descubrir el partido en el que se marcaron más goles. Pista. Utiliza la función de string "explode".

```
<?php
$resultados = array (array (null, "2-3", "3-1"),
                array ("3-4", null, "0-0"),
                array ("2-0", "6-0", null));
$equipos = array ("Real Madrid", "Manchester Utd", "AC Milan");
$numgoles = 0;
$equipolocal;
$equipovisitante;

for ($i = 0; $i < sizeof($resultados); $i++) {
  for ($j = 0; $j < sizeof($resultados[$i]); $j++) {
     if ($i!=$j) {
        $goles = explode("-", $resultados[$i][$j]);
        if (array_sum($goles) > $numgoles)
        {
              $numgoles = array_sum($goles);
              $equipolocal = $i;
              $equipovisitante = $j;
        }
     }
  }
}
   echo "Maximo numero de goles: ", $numgoles, "<br>";
   echo "Partido: ", $equipos[$equipolocal], " - ",
$equipos[$equipovisitante];
?>
```

- php44.pdf. Escribe un programa que dado un array de enteros, muestre por pantalla el tamaño de dicho array.

```
<?php
   $enteros = array (2,6,45,234);
   echo "El array tiene: ", sizeof($enteros), "enteros";
?>
```

- php45.pdf. Ejecuta el script del ejemplo 60. Copia el resultado y explícalo con detalle.

Ejecuta el script y explícalo como aparece en la documentación

❖ php47.pdf. Escribe un programa que dado un array numérico devuelva dicho array ordenado de forma ascendente y de forma descendente.

```
<?php
   $enteros=array(4,7,9,1,34,12,83,195,33,0,156,234,22);
   echo "Array original:", "<br>";
   foreach ($enteros as $entero) {
      echo $entero;
      echo ", ";
   }
   echo "<br>";

   echo "Array ordenado de forma ascendente:", "<br>";
   sort($enteros);
   foreach ($enteros as $entero) {
      echo $entero;
      echo ", ";
   }
   echo "<br>";

   echo "Array ordenado de forma descendente:", "<br>";
   rsort($enteros);
   foreach ($enteros as $entero) {
      echo $entero;
      echo ", ";
   }
?>
```

❖ php48.pdf. Ejecuta el script del ejemplo 63. Copia el resultado y explícalo con detalle.

Ejecuta el script y explícalo como aparece en la documentación

9 Flujo de control condicional (if)

- php37.pdf. Escribe un programa que dadas dos variables numéricas, muestre por pantalla el valor mayor de ambas.

```
<?php
   $num1 = 8;
   $num2 = 5;
   if ($num1 > $num2)
      $max = $num1;
   else if ($num1 < $num2)
      $max = $num2;
        else
      $max = $num1;
   echo "El valor maximo es ", $max;
?>
```

- php37.pdf. Escribe un programa que dada una variable numérica, muestre por pantalla si es un número par o impar.

```
<?php
  $num = 24;
   if ($num%2 == 0)
      echo "El numero $num es par";
   else
      echo "El numero $num es impar";
?>
```

❖ php37.pdf. Escribe un programa que dadas tres variables numéricas, muestre por pantalla el valor mayor de ambas.

```
<?php
   $num1 = 65;
   $num2 = 52;
   $num3 = 45;
      if ($num1 > $num2)
      if ($num1 > $num3)
         $max = $num1;
       else
          $max = $num3;
    else if ($num2 > $num3)
       $max = $num2;
       else
          $max = $num3;
    echo "El valor maximo es ", $max;
?>
```

❖ php37.pdf. Escribe un programa que dada una variable que contiene un carácter, muestre por pantalla si es vocal, consonante, dígito o cualquier otro carácter.

```
<?php
   $var = "k";
   $vocales = "aeiou";
   $consonantes = "bcdfghjklmnñpqrstvwxyz";
    if (strpos($vocales, $var) !== False)
         echo "La variable es una vocal";
     else if (strpos($consonantes, $var) !== False)
            echo "La variable es una consonante";
         else if (is_int ($var))
               echo "La variable es un digito";
                else
                   echo "La variable otro caracter";
?>
```

10 Flujo de control condicional (switch)

- php38.pdf. ¿Para qué sirve la sentencia "break"?.

La sentencia break determina el punto donde se terminará la ejecución del bloque o bloques de la instrucción switch. El flujo de control salta a la siguiente instrucción fuera de la instrucción switch.

- php38.pdf. Escribe un programa que dadas dos variables numéricas, una indica la cantidad y otra indica la divisa (1 – dólar, 2 – libra esterlina, 3 – rupia, 5 – Yuan), muestre en pantalla la conversión de dicha cantidad a euros.

```
<?php
   $cantidad = 34;
   $divisa = "yen";
   switch ($divisa) {
      case "dolar":
         echo $cantidad, " ", $divisa, " son: ", $cantidad*0.77, " euros";
         break;
      case "yen":
         echo $cantidad, " ", $divisa, " son: ", $cantidad*0.01, " euros";
         break;
      case "libra esterlina":
         echo $cantidad, " ", $divisa, " son: ", $cantidad*1.24, " euros";
         break;
      case "rupia":
         echo $cantidad, " ", $divisa, " son: ", $cantidad*0.014, " euros";
         break;
      default:
         echo "Error en el nombre de divisa";
   }
?>
```

❖ php38.pdf. Escribe un programa que dada una variable numérica que tome valores del 0 al 9, muestre por pantalla dicho número expresado con letras.

```
<?php
   $num = 7;
   switch ($num) {
      case 0:
         echo "cero";
         break;
      case 1:
         echo "uno";
         break;
      case 2:
         echo "dos";
         break;
      case 3:
         echo "tres";
         break;
      case 4:
         echo "cuatro";
         break;
      case 5:
         echo "cinco";
         break;
      case 6:
         echo "seis";
         break;
      case 7:
         echo "siete";
         break;
      case 8:
         echo "ocho";
         break;
      case 9:
         echo "nueve";
         break;
      default:
         echo "No es un numero del 0 al 9";
   }
?>
```

11 Flujo de control iterativo (while)

- php39.pdf. Escribe un programa que dada una variable entera, muestre en pantalla todos los valores desde el 1 hasta dicho valor.

```
<?php
$num = 34;
$indice = 1;
while ($indice <= $num) {
   echo $indice, "<br>";
   $indice++;
}
?>
```

- php39.pdf. Escribe un programa que dadas dos variables numéricas, una indica la base u otra el exponente de una potencia, calcule el resultado de dicha potencia.

```
<?php
$base = 5;
$exp = 3;
$i = 0;
$res = 1;

while($i < $exponente) {
   $res = $res * $base;
   $i++;
}
echo $res;
?>
```

- php39.pdf. Escribe un programa que dada una variable numérica, muestre en pantalla el factorial de ese número.

```
<?php
$num = 8;
$i = 1;
$res = 1;
while($num >= $i) {
   $res = $res * $i;
   $i++;
}
echo "El factorial de $num es: $res";
?>
```

- php39.pdf. Escribe un programa que dado un array numérico, muestre en pantalla el valor máximo de dicho array.

```
<?php
$numeros = array(15,45,34,67,55,38,95);
$i = 0;
$max = 0;
while(sizeof($numeros) > $i) {
   if($numeros[$i] > $max) {
      $max = $numeros[$i];
   }
   $i++;
}
echo "El numero maximo es: $max";
?>
```

12 Flujo de control iterativo (do .. while)

- php40.pdf. ¿Cuál es la diferencia entre un bucle "while" y un bucle "do .. while"?.

En el bucle "do .. while" se comprueba la condición después de ejecutar el bloque de instrucciones del bucle. Por tanto el número mínimo de iteraciones es 1 y no 0 como ocurre con el bucle "while".

- php40.pdf. Escribe un programa que dada una variable numérica que indica el número de tiradas de un dado, simule el lanzamiento de dichas tiradas y muestre por pantalla las veces que ha obtenido cada número.

```
<?php
$num_tiradas = 18;
$i = 0;
$pos_val = 6;
$val_tiradas = array(0,0,0,0,0,0);
do{
   $num = mt_rand(1, $pos_val);
   $val_tiradas[$num-1]++;
   $i++;
}while ($i < $num_tiradas);
foreach ($val_tiradas  as $num => $veces)
   echo "El numero ", ($num+1), " ha salido $veces veces <br>";
?>
```

13 Flujo de control iterativo (for)

- php42.pdf. Escribe un programa que dada una variable numérica del 1 al 9. muestre en pantalla la tabla de multiplicar de dicho valor.

```
<?php
    $num = 8;
    $res = 0;
    echo "La tabla del $num: <br>";
    for ($i=1; $i<=10; $i++) {
        $res = $num * $i;
        echo "$num x $i = $res <br>";
    }
?>
```

- php42.pdf. Escribe un programa que dado un array numérico y una variable numérica, muestre en pantalla aquellos valores del array que superan en valor de la variable.

```
<?php
    $numeros = array(8, 23, 67, 33, 89, 1, 56);
    $num = 54;

    for ($i=0; $i<sizeof($numeros); $i++) {
        if ($numeros[$i] > $num) {
            echo "El numero $numeros[$i] es mayor que $num <br>";
        }
    }
?>
```

14 Flujo de control iterativo (foreach)

- php42.pdf ¿Cuál es la diferencia entre un bucle for y un bucle foreach?.

El bucle foreach es específico para el tratamiento de los arrays ya que está preparado para que cada iteración trabaje con un elemento de dicho array.

El bucle iterará tantas veces como elementos existen en el array.

Existe una sintaxis para que en cada iteración del bucle se tenga acceso tanto al valor como a su índice en el array.

- php42.pdf. Escribe un programa que dado una array de palabras, muestre en pantalla cada palabra en una línea diferente.

```
<?php
    $palabras = array("palabra1", "palabra2", "palabra3");
    foreach ($palabras as $palabra)
        echo "$palabra <br>";
?>
```

- php42.pdf. Escribe un programa que dado una array bidimensional de letras, muestre en pantalla el número de letras 'a' que hay en el array.

```
<?php
    $letraa = "a";
    $letras = array(array("a","b","c"), array ("d", "a", "f"), array
("g","a","h"));
    $numa = 0;

    foreach ($letras as $lista_letras)
       foreach ($lista_letras as $letra)
          if ($letra == $letraa)
             $numa ++;

    echo "Hay $numa as en el array"
?>
```

15 Funciones de usuario

- php49.pdf. Ejecuta el script del ejemplo 66. Copia el resultado y explícalo con detalle.

Ejecuta el script y explícalo como aparece en la documentación

- php49.pdf. Ejecuta el script del ejemplo 67. Copia el resultado y explícalo con detalle.

Ejecuta el script y explícalo como aparece en la documentación

- php49.pdf. ¿Qué diferencia existe entre el paso de parámetros por valor y por referencia?.

El paso de parámetros por valor se realiza pasando una copia del parámetro real al parámetro formal de la función. Por lo tanto las modificaciones al parámetro formal en la función de usuario no afectan a la variable utilizada como parámetro real.

Si el parámetro se pasa por referencia entonces el parámetro formal se convierte en una referencia o alias del parámetro real y por tanto los cambios que sufra dentro de la función serán también válidos sobre la variable utilizada como parámetro real.

❖ php50.pdf. Escribe un programa que define funciones de usuario para sumar, restar, multiplicar y dividir dos valores numéricos. Invoca todas las funciones de usuario desde el programa principal y muestra el resultado en pantalla.

```
<?php
$n1 = 8;
$n2 = 4;
$res;
$operacion = "multiplicacion";
$error=False;

function suma($num1, $num2) {
   return $num1 + $num2;
}
function resta($num1, $num2) {
   return $num1 - $num2;
}
function multiplicacion($num1, $num2) {
   return $num1 * $num2;
}
function division($num1, $num2) {
   return $num1 / $num2;
}

switch ($operacion) {
   case "suma":
     $res = suma($n1, $n2);
      break;
   case "resta":
     $res = resta($n1, $n2);
      break;
   case "multiplicacion":
     $res = multiplicacion($n1, $n2);
      break;
   case "dividir":
     $res = division($n1, $n2);
      break;
   default:
      echo ("Error en tipo de operacion.<br>");
      $error = True;

}

if (!$error) echo "El resultado de la operacion $operacion es: $res";

?>
```

- php50.pdf. Escribe un programa que define una función de usuario que encuentre los divisores de un número. Invoca dicha función de usuario desde el programa principal y muestra el resultado en pantalla.

```
<?php
$num = 36;
$divisores = array();

function calculaDivisores($valor) {
   $divs = array();
   for($i =1;$i <= sqrt($valor);$i++)
{
    if($valor % $i == 0){
        $divs[] = $i;
         if ($valor != pow($i,2)) $divs[] = $valor / $i;
    }
}
   return $divs;
}

$divisores = calculaDivisores($num);
sort ($divisores);
echo "Los divisores de $num son: <br>";
foreach ($divisores as $divisor)
   echo "$divisor <br>";
?>
```

16 Funciones de fecha

- php51.pdf. Escribe un programa que muestre ejemplos de utilización de las funciones de fecha.

> Consulta la documentación para escribir funciones similares

17 Clases y objetos

- ¿Qué es una clase?. ¿De qué se compone?

Una clase es una construcción que se utiliza como un modelo (o plantilla) para crear objetos de ese tipo. El modelo describe el estado y el comportamiento que todos los objetos de la clase comparten.

El estado se representa mediante atributos o propiedades y el comportamiento lo definen sus métodos de la clase

- ¿Qué es un método constructor y un método destructor?

La clase se utiliza como plantilla para crear objetos o instancias de ella. Para la creación de objetos se utilizan los métodos constructores. Para destruir los objetos se utilizan los métodos destructores.

- ¿Qué diferencias existen entre los esquemas de visibilidad "public", "protected" y "private"?

Los elementos declarados como Public son accesibles tanto desde fuera como desde dentro de la clase.

Los elementos declarados como Private son accesibles sólo desde la misma clase donde fueron definidos.

Los elementos declarados como Protected son accesibles desde la misma clase donde fueron definidos y en sus subclases.

- ¿Qué es una jerarquía de clases?

Una jerarquía de clases muestra una serie de clases y sus interrelaciones de modo que se pueden establecer relaciones de generalización o de especialización entre ellas. La jerarquía de clases define la base para aplicar esquemas de herencia a objetos de una clase de la jerarquía y clases superiores a ella.

❖ ¿Qué es la herencia?

Como comentamos anteriormente, la herencia permite a objetos de clases inferiores en la jerarquía heredar propiedades y métodos de aquellas superclases de la jerarquía.

❖ ¿Qué es un objeto?

Un objeto es una instancia de una clase. La clase en sí es sólo una definición o descripción de un tipo de objetos mientras que los objetos existen y tienen vida dentro de los procesos.

❖ ¿Qué es el polimorfismo?

Dada una jerarquía de clases si todos los objetos de clases específicas cumplen el interfaz definido por la superclase, se podrá considerar a todos los objetos como instancias de la clase superior e invocarles los métodos de su interfaz. El polimorfismo permitirá que cada objeto invoque el método correspondiente a su verdadera entidad como instancia de la clase específica.

❖ Escribe un programa que defina una jerarquía de clases de figuras geométricas (cuadrado, rectángulo y círculo) con los métodos para el cálculo de área y perímetro. Crea objetos de dichas clases e introduce dichos objetos en un array de figuras geométricas. Después recorre el array invocando ambos métodos para cada uno de los objetos.

```php
<?php
abstract class Figura {
    abstract function area ();
    abstract function perimetro ();
}
class Cuadrado extends Figura {
   private $l=0;
   function __construct ($lado)
   {
      $this->l = $lado;
   }
   function area (){
      return (pow ($this->l, 2));    }

   function perimetro () {
      return (4 * $this->l);
   }
}

class Rectangulo extends Figura {
   private $a=0;
   private $b=0;

   function __construct ($alto, $ancho)
   {
      $this->a = $alto;
      $this->b = $ancho;
   }
   function area (){
      return ($this->a*$this->b);
   }
   function perimetro () {
      return ((2 * $this->a) + (2 * $this->b));
   }
}

class Circulo extends Figura {
   private $radio=0;

   function __construct ($r)
   {
      $this->radio = $r;
   }
   function area (){
      return (M_PI * pow($this->radio, 2));
   }
  function perimetro () {
      return (2 * M_PI * $this->radio);

   }
}
```

```
<?php include ("Figura.php") ?>

<?php

$cuadrado1 = new Cuadrado(3);
$cuadrado2 = new Cuadrado(4);
$rectangulo1 = new Rectangulo(2,3);
$circulo1 = new Circulo (4);

$formas=array ($cuadrado1, $cuadrado2, $rectangulo1, $circulo1);

foreach ($formas as $forma)
{
  echo "El area del objeto de clase ", get_class($forma), " es :",
$forma->area(), "<br>";
  echo "El perimetro del objeto de clase ", get_class($forma), " es :",
$forma->perimetro(), "<br>";
}
?>
```