Guía Práctica de Ejercicios Nº 2

En nuestra segunda práctica seguiremos viendo los conceptos básicos relacionados con el lenguaje PHP. Comenzando con el manejo de arreglos, el uso de funciones y la utilización de cadenas (strings), entre otros. Aprenderemos a utilizar muchas funciones para el manejo de arreglos y cadenas en PHP, tales como: mostrar el contenido de una arreglo, diferencias entre arreglos, pasar a mayúsculas una cadena, obtener el largo de un strings, entre tantas.

Objetivo: Profundizar y comprender el funcionamiento básico de la Web y PHP. También se pretende entender y emplear arreglos en el lenguaje; luego lograr un manejo de las funciones dentro del lenguaje y por último el tratamiento de cadenas (strings).

Introducción: En nuestra actualidad vemos que el desarrollo de Internet ha sido inminente y con ello las aplicaciones Web, por lo tanto, se hace indispensable el uso de un lenguaje que permita desarrollar aplicaciones Web como PHP, entre otros. Teniendo en cuenta la situación anterior es que veremos la como desarrollar aplicaciones Web que se ejecutarán del lado del servidor utilizando uno de los mejores lenguajes del ambiente del Software Libre.

Construcciones (palabras clave) y funciones que estudiaremos en esta unidad.

Funciones con las que trabajaremos.

array() nos permite crear un arreglo.

sort() nos brinda la posibilidad de ordenar un arreglo que le pasamos como parámetro, modificando al mismo a nivel de claves.

asort() nos brinda la posibilidad de ordenar un arreglo que le pasamos como parámetro permitiendo mantener la asociación de las claves que contenga el arreglo.

rsort() nos brinda la posibilidad de ordenar un arreglo que le pasamos como parámetro (igual a sort()), pero ordenando los elementos de forma descendente.

arsort() nos brinda la posibilidad de ordenar un arreglo que le pasamos como parámetro (igual a asort()), pero ordenando los elementos de forma descendente.

ksort() ordena un arreglo teniendo en cuenta las claves del mismo.

array_diff() nos permite calcula la diferencia entre matrices.

array_intersect() calcula la intersección de arreglos.

printf() nos permite mostrar en pantalla una cadena que se recibe como parámetro pero que además permite darle formato a la misma.

sprintf() idéntica a printf() pero con la diferencia de que sprintf() no muestro los resultados por pantalla sino que se almacenarán en una variable.

strtoupper() función que nos permite convertir una cadena de texto que se recibe como parámetro a mayúsculas.

strtolower() permite convertir una cadena de texto que se recibe como parámetro a minúsculas.

trim() elimina los espacios en blanco al inicio y al final de una cadena que se recibe como parámetro.

ltrim() elimina los espacios en blanco que se encuentran al comienzo de una cadena de texto que se pasa como parámetro.

rtrim() permite eliminar los espacios en blanco que se encuentran al final de una cadena de texto que se pasa como parámetro.

chop() trabaja de forma idéntica a rtrim() y chop() es un alias del mismo.

strlen() devuelve la longitud de una cadena de texto que se le pasa como parámetro.

count() permite obtener la longitud de una cadena de texto, de arreglos y objetos.

substr() permite obtener una porción de texto de una cadena que se le pasa como parámetro a la misma.

substr_replace() permite encontrar y reemplazar una porción de texto de una cadena que se pasa como parámetro.

ereg() nos permite buscar una cadena de texto dentro de otra, devolviendo verdadero si encontró coincidencia, o falso, en caso de no encontrar coincidencia.

eregi() trabaja de igual forma que ereg() con la diferencia de que no es sensitiva a mayúsculas y minúsculas.

str_replace() devuelve un string del cual se reemplazaron las distintas coincidencias que se encontraron.

str_ireplace() devuelve un string del cual se reemplazaron las distintas coincidencias que se encontraron, y no es sensitiva a mayúsculas y minúsculas.

split() permite separar (dividir) una cadena de texto indicándole el carácter separador como parámetro.

explode() permite separar (dividir) una cadena de texto indicándole el carácter separador como parámetro.

implode() permite juntar los elementos de un arreglo en una cadena de texto.

Ejercicios

Preparar y verificar nuestro entorno de desarrollo PHP

Antes de comenzar con la práctica, asegúrese de tener instalado su entorno de desarrollo.

1) - Realice un script PHP que permita mostrar las características de interprete PHP utilizando la función phpinfo().

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

Utilizar los conceptos básicos del lenguaje (Arreglos)

2) - Realice un script que defina un arreglo de nombre $dias_semana y al cual debemos asignar cada uno de los días de la misma comenzando por el Domingo. Recuerde utilizar etiquetas estándares.

```
<?php
   $dias_semana = array('Domingo', 'Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes',
'Sábado');
?>
```

3) - Edite el script anterior y agregue una nueva sentencia que muestre el contenido del arreglo con la construcción foreach.

```
<?php
   $dias_semana = array('Domingo', 'Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes',
'Sábado');
   foreach ($dias_semana as $un_dia) {
      echo $un_dia."<br />";
   }
?>
```

4) - Edite el script del punto 3 y agregue la función print_r() para ver el contenido y la estructura del arreglo $dias_semana.

```
<?php
   $dias_semana = array('Domingo', 'Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes',
'Sábado');
```

```
   foreach ($dias_semana as $un_dia) {
      echo $un_dia."<br />";
   }
   print_r($dias_semana);
?>
```

5) - Edite el script del punto 4 y ordene el arreglo de forma ascendente y muestre en pantalla el contenido del mismo.

```
<?php
   $dias_semana = array('Domingo', 'Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes',
'Sábado');
   foreach ($dias_semana as $un_dia) {
      echo $un_dia."<br />";
   }
   print_r($dias_semana);
   echo "<br />";
   sort($dias_semana);
   foreach ($dias_semana as $un_dia) {
      echo $un_dia."<br />";
   }
?>
```

6) - Edite el script del punto 4 y ordene el arreglo de forma descendiente y muestre en pantalla el contenido del mismo.

```
<?php
   $dias_semana = array('Domingo', 'Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes',
'Sábado');
   foreach ($dias_semana as $un_dia) {
      echo $un_dia."<br />";
   }
   print_r($dias_semana);
   echo "<br />";
   rsort($dias_semana);
   foreach ($dias_semana as $un_dia) {
```

```
    echo $un_dia."<br />";
  }
?>
```

7) - Realice un nuevo script que defina un arreglo asociativo de nombre $persona y al cual debemos asignar los siguiente valores y mostrarlos en pantalla:

Clave: tipo_documento, Valor: DNI
Clave: numero_documento, Valor: 30.122.122
Clave: apellidos, Valor: Garcia
Clave: nombres, Valor: Juan
Clave: domicilio, Valor: San Martín 123
Clave: telefono, Valor: 4222324
Clave: edad, Valor: 25

```
<?php
  $persona = array(
    'tipo_documento' => 'DNI',
    'numero_documento' => '30122122',
    'apellidos' => 'Garcia',
    'nombres' => 'Juan',
    'domicilio' => 'San Martín 123',
    'telefono' => '4222324',
    'edad' => 25);
  foreach ($persona as $dato_persona) {
    echo $dato_persona."<br />";
  }
?>
```

Utilizar funciones en PHP

8) - Escriba un script PHP que defina cuatro funciones que realicen las operaciones matemáticas de suma, resta, multiplicación y división. Las funciones deberán esperar dos parámetros pasados por valor y devolver el resultado de la operación. Dentro del mismo script deberán utilizar cada una de las funciones y mostrar el resultado por pantalla.

```
<?php
function suma($valor1, $valor2) {
  $resultado = $valor1 + $valor2;
```

```
    return $resultado;
}
function resta($valor1, $valor2) {
    $resultado = $valor1 - $valor2;
    return $resultado;
}
function multiplicacion($valor1, $valor2) {
    $resultado = $valor1 * $valor2;
    return $resultado;
}
function division($valor1, $valor2) {
    $resultado = $valor1 / $valor2;
    return $resultado;
}
$a = 2;
$b = 4;
echo "La suma de $a y $b es: ".suma($a, $b)."<br />";
echo "La resta de $a y $b es: ".resta($a, $b)."<br />";
echo "La multiplicación de $a y $b es: ".multiplicacion($a, $b)."<br />";
echo "La división de $a y $b es: ".division($a, $b)."<br />";
?>
```

9) - Editar el script PHP del punto anterior y agregue una función que permita incrementar el valor de una variable pasada como parámetro por referencia. Dentro del mismo script deberá utilizar la función y mostrar el resultado por pantalla.

```
<?php
function suma($valor1, $valor2) {
    $resultado = $valor1 + $valor2;
    return $resultado;
}
function resta($valor1, $valor2) {
    $resultado = $valor1 - $valor2;
```

```
    return $resultado;
}
function multiplicacion($valor1, $valor2) {
    $resultado = $valor1 * $valor2;
    return $resultado;
}
function division($valor1, $valor2) {
    $resultado = $valor1 / $valor2;
    return $resultado;
}
function incrementar($valor){
    $valor++;
    return $valor;
}
$a = 2;
$b = 4;
echo "La suma de $a y $b es: ".suma($a, $b)."<br />";
echo "La resta de $a y $b es: ".resta($a, $b)."<br />";
echo "La multiplicación de $a y $b es: ".multiplicacion($a, $b)."<br />";
echo "La división de $a y $b es: ".division($a, $b)."<br />";
echo "El incremento de $a es: ".incrementar($a);
?>
```

Utilizar y manejar cadenas en PHP

10) – Escriba un script PHP que utilice la función date('Y-m-d') para obtener la fecha actual y utilizando la función explode() cambie el formato de la fecha a dd/mm/yyyy, muestre la fecha con formato por pantalla.

```
<?php
$fecha = date('Y-m-d');
$arreglo_fecha = explode("-", $fecha);
$fecha_nueva = $arreglo_fecha[2]."/".$arreglo_fecha[1]."/".$arreglo_fecha[0];
```

```
echo $fecha_nueva;

?>
```