Universidad Tecnológica Nacional

Ingeniería en Sistemas de Información

Cátedra: Entornos Gráficos

Práctica Nro 4: PHP

Alumno:

Agustín Yurescia – Agustin_yurescia@hotmail.com - 42683

EJERCICIO 1

En el siguiente código identificar:

```
<?php
function doble($i) {
   return $i*2;
}
$a = TRUE;
$b = "xyz";
$c = 'xyz';
$d = 12;
echo gettype($a);
echo gettype($b);
echo gettype($c);
echo gettype($d);
if (is_int($d)) {
   $d += 4;
}
if (is_string($a)) {
   echo "Cadena: $a";
}
$d = $a ? ++$d : $d*3;
$f = doble($d++);
$g = $f += 10;
echo $a, $b, $c, $d, $f , $g;
?>
```

- las variables y su tipo
- los operadores
- las funciones y sus parámetros
- las estructuras de control
- cuál es la salida por pantalla

- Variables:

VARIABLE	TIPO
$a	Boolean
$b	String
$c	String
$d	String
$f	Integer
$g	Integer

- Operadores**:** +, ++, *, *=, ?:, +=

- Funciones:
    - doble(parámetro1)
    - gettype(parámetro1)
    - is_int(parámetro1)
    - is_string(parámetro1)
- Estructuras de control: if
- Salida por pantalla: booleanstringstringinteger1xyzxyz184444

EJERCICIO 2

Indicar si los siguientes códigos son equivalentes:

```
<?php
$i = 1;
while ($i <= 10) {
print $i++;
}
?>
```

```
<?php
$i = 1;
 while ($i <= 10):
    print $i;
    $i++;
 endwhile;
?>
```

```
<?php
 $i = 0;
 do {
    print ++$i;
 } while ($i<10);
?>
```

Los tres códigos anteriores son similares, devuelven los números del 1 al 10. Los dos primeros lo realizan a través de estructuras de control WHILE, mientras que el último lo hace a través de DO WHILE.

```
<?php
for ($i = 1; $i <= 10; $i++) {
    print $i;
 }
?>
```

```
<?php
for ($i = 1; $i <= 10; print $i, $i++) ;
?>
```

```
<?php
for ($i = 1; ;$i++) {
    if ($i > 10) {
        break;
    }
    print $i;
 }
?>
```

```
<?php
 $i = 1;
 for (;;) {
    if ($i > 10) {
        break;
    }
    print $i;
    $i++;
 }
?>
```

Los cuatro códigos anteriores son similares, devuelven también los números del 1 al 10. Todos lo hacen con variantes de la estructura de control FOR.

```php
<?php
…
…
if ($i == 0) {
   print "i equals 0";
} elseif ($i == 1) {
   print "i equals 1";
} elseif ($i == 2) {
   print "i equals 2";
}
?>
```

```php
<?php
…
…
switch ($i) {
   case 0:
      print "i equals 0";
      break;
   case 1:
      print "i equals 1";
      break;
   case 2:
      print "i equals 2";
      break;
}
?>
```

Los dos códigos anteriores son similares, devuelven “i equals 0” si la variable i es igual a 0, “i equals 1” si i es igual a 1 y “i equals 2” si es igual a 2. El primero lo hace utilizando ELSEIF y el segundo utilizando SWITCH.

EJERCICIO 3

Explicar para qué se utiliza el siguiente código:

- El primer código mostrado se utiliza para crear una tabla HTML de cinco filas y dos columnas donde cada celda está formada por un espacio vacío.
- El segundo código mostrado se utiliza para recibir el campo “age” a través de un formulario HTML por el método post y a partir del valor del mismo mostrar “Mayor de edad” si age >= 21 o “Menor de edad” si age > 21.

EJERCICIO 4

Si el archivo datos.php contiene el código que sigue:

```php
<?php
$color = 'blanco';
$flor = 'clavel';
?>
```

Indicar las salidas que produce el siguiente código. Justificar.

```php
<?php
echo "El $flor $color \n";
include 'datos.php';
echo " El $flor $color";
?>
```

Las salidas el código son las siguientes:

- En primer lugar, devuelve una advertencia de que las variables no se encuentran definidas.
- En segundo lugar, muestra “El clavel blanco”.

Lo anterior sucede porque primero se desea mostrar un mensaje utilizando las variables $flor y $color antes del include 'datos.php'.

EJERCICIO 5:

contador.php

```
<?
// Archivo para acumular el numero de visitas
$archivo = "contador.dat";
// Abrir el archivo para lectura
$abrir = fopen($archivo, "r");
// Leer el contenido del archivo
$cont = fread($abrir, filesize($archivo));
// Cerrar el archivo
fclose($abrir);
// Abrir nuevamente el archivo para escritura
$abrir = fopen($archivo, "w");
// Agregar 1 visita
$cont = $cont + 1;
// Guardar la modificación
$guardar = fwrite($abrir, $cont);
// Cerrar el archivo
fclose($abrir);
// Mostrar el total de visitas
echo "<font face='arial' size='3'>Cantidad de visitas:".$cont."</font>";
?>
```

visitas.php

```
<!-- Página que va a contener al contador de visitas  -->
<html>
<head></head>
<body>
<? include("contador.php")?>
</body>
</html>
```

En este ejemplo, se ejecuta el archivo visitas.php, la cual ejecuta con el uso de include, al archivo contador.php, que dentro del mismo se abre, lee, escribe (incrementa el valor de la variable $cont), guarda y cierra el archivo contador.dat, el cual es el archivo contenedor del valor de la cantidad de visitas de la página web. Finalmente se muestra en la página web: Cantidad de visitas: numero_de_visitas.

ARRAYS Y FUNCIONES

EJERCICIO 1

Indicar si los siguientes códigos son equivalentes.

<?php $a = array('color' => 'rojo', 'sabor' => 'dulce', 'forma' => 'redonda', 'nombre' => 'manzana', 4); ?>	<?php $a['color'] = 'rojo'; $a['sabor'] = 'dulce'; $a['forma'] = 'redonda'; $a['nombre'] = 'manzana'; $a[] = 4; ?>

Los códigos anteriores son similares, ambos crean un array del tipo clave → valor (asociativo) con las claves color, sabor, forma, nombre con los valores rojos, dulce, redondada y manzana respectivamente.
Además, añaden el valor 4 al mismo, sin ninguna clave (array indexado).

EJERCICIO 2

En cada caso, indicar las salidas correspondientes:

CÓDIGO	SALIDA
<?php $matriz = array("x" => "bar", 12 => true); echo $matriz["x"]; echo $matriz[12]; ?>	Bar1
<?php $matriz = array("unamatriz" => array(6 => 5, 13 => 9, "a" => 42)); echo $matriz["unamatriz"][6]; echo $matriz["unamatriz"][13]; echo $matriz["unamatriz"]["a"]; ?>	5942
<?php $matriz = array(5 => 1, 12 => 2); $matriz[] = 56; $matriz["x"] = 42; unset($matriz[5]); unset($matriz); ?>	No hay salida

EJERCICIO 3

En cada caso, indicar las salidas correspondientes:

CÓDIGO	SALIDA
<?php $fun = getdate(); echo "Has entrado en esta pagina a las $fun[hours] horas, con $fun[minutes] minutos y $fun[seconds] segundos, del $fun[mday]/$fun[mon]/$fun[year]"; ?>	Has entrado en esta página a las 15 horas, con 25 minutos y 36 segundos, del 17/05/2020

<?php
function sumar($sumando1,$sumando2){
$suma=$sumando1+$sumando2;
echo $sumando1."+".$sumando2."=".$suma;
}
sumar(5,6);
?>
	5+6=11

EJERCICIO 4

Analizar la siguiente función, y escribir un script para probar su funcionamiento:

```
function comprobar_nombre_usuario($nombre_usuario){
  //compruebo que el tamaño del string sea válido.
  if (strlen($nombre_usuario)<3 || strlen($nombre_usuario)>20){
    echo $nombre_usuario . " no es válido<br>";
    return false;
  }

  //compruebo que los caracteres sean los permitidos
  $permitidos = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789-
_";
  for ($i=0; $i<strlen($nombre_usuario); $i++){
    if (strpos($permitidos, substr($nombre_usuario,$i,1))===false){
      echo $nombre_usuario . " no es válido<br>";
      return false;
    }
  }
  echo $nombre_usuario . " es válido<br>";
  return true;
}
```

La función comprobar_nombre_usuario se encarga de recibir un nombre de usuario como parámetro y comprobar que el mismo tenga una logitud mayor a tres y menor o igual que 20.

Luego comprueba que el mismo esté constituido por caracteres válidos verificando a través de un FOR que cada uno de los mismo se encuentre entre los caracteres de la variable $permitidos.

Script para comprobar su funcionamiento:

```
<?php
$nombre1 = “ja”
$nombre2 = “juan33”
$nombre3 = “juan$$”
if (comprobar_nombre_usuario($nombre1) == 0){
        echo “La función funciona”
else:
        echo “La función no funciona”
if (comprobar_nombre_usuario($nombre2) == 1){
        echo “La función funciona”
else:
        echo “La función no funciona”
if (comprobar_nombre_usuario($nombre3) == 0){
        echo “La función funciona”
else:
        echo “La función no funciona”
?>
```