Facultad Regional Rosario
Universidad Tecnológica Nacional

ENTORNOS GRAFICOS

PRÁCTICA N°4 PHP:

Variables, tipos, operadores, expresiones, estructuras de control PHP: arrays, funciones

Integrantes:

- ***Navarro, Franco***
- ***Romero, Joan***
- ***Santolari, Matías***

Ejercicio 1: En el siguiente código identificar:

• las variables y su tipo

$a Boolean $b String $c String $d Integer

• los operadores

* Binario ; + Binario ; ++ Unario ; ?: Terciario

• las funciones y sus parámetro

Función: doble Parámetro: $i,$d++ ;

getttype $a,$b,$c,$d ;

is_string $a is_int $d

• las estructuras de control

1- if (is_int($d)) { $d += 4; }

2- if (is_string($a)) { echo "Cadena: $a"; }

• cuál es la salida por pantalla

Boolean

String

String

Integer

TRUE xyz xyz 17 34 44

```php
<?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;
?>
```

Ejercicio 2: Indicar si los siguientes códigos son equivalentes.

a)

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

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

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

Ejemplo 1 el parámetro $i inicializa en 1, muestra i y luego incrementa, así hasta que i sea igual a 10.

Ejemplo 2 el parámetro $i inicializa en 1, muestra i y luego incrementa, así hasta que i sea igual a 10.

Ejemplo 3 el parámetro $i inicializa en 0, $i se incrementa y luego se muestra, así mientras i sea menor a 10 al final de la iteración.

Por lo tanto son equivalentes

b)

```
<?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++;
 }
?>
```

Son equivalentes, todos muestran números del 1 al 10.

Ya que en todos el valor inicial es 1, la condición del for siempre es i<=10 y siempre se incrementa de a 1.

c)

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

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

Si son equivalentes, la sentencia switch es similar a una serie de sentencias IF en la misma expresión.

Ejercicio 3: Explicar para qué se utiliza el siguiente código:

a)

```
<html>
<head><title>Documento 1</title></head>
<body>
<?php
   echo "<table width = 90% border = '1' >";
   $row = 5;
   $col = 2;
   for ($r = 1; $r <= $row; $r++) {
         echo "<tr>";
         for ($c = 1; $c <= $col;$c++) {
               echo "<td> </td>\n";
         }     echo "</tr>\n";
   }
   echo "</table>\n";
?>
</body></html>
```

El siguiente código se utiliza para crear una tabla, donde el primer FOR genere una fila y dentro del otro FOR genera la columnas correspondientes con la información.

b)

```
<html>
<head><title>Documento 2</title></head>
<body>
<?php
if (!isset($_POST['submit'])) {
?>
   <form action="<?php echo $_SERVER['PHP_SELF']; ?>" method="post">
   Edad: <input name="age" size="2">
   <input type="submit" name="submit" value="Ir">
   </form>
<?php
   }
else {
   $age = $_POST['age'];
   if ($age >= 21) {
      echo 'Mayor de edad';
      }
   else {
      echo 'Menor de edad';
   }
}
?>
</body></html>
```

El primer if se utiliza para corroborar que los datos del formulario estén cargados. Como el isset está negado, devolverá false si es que los datos están cargados en el formulario, por lo tanto dentro del else se corroborará con otro if si es mayor de edad o no.

Ejercicio 4:

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

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

Indicar las salidas que produce el siguiente código. Justificar.

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

Lo que sucede es que en la primera línea de salida solamente se muestra "el" ya que el archivo datos.php que tiene las variables no se encuentra cargado, una vez cargado el archivo datos.php en la siguiente línea, permite a la próxima línea de salida que se encuentra debajo de esta a actuar correctamente y dar como salida "el clavel blanco".

Ejercicio 5:

Analizar el siguiente ejemplo: Contador de visitas a una página web

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 la misma carpeta, crear el archivo de texto **contador.dat**, con el valor inicial del contador y con permisos de lectura y escritura.

Cada vez que se ingresa en la página visitas.php esta ejecuta el código de contador.php que abre el archivo contador.dat lee la información en él y la modifica sumándole una unidad al contador dentro del archivo. Finalmente cierra contador.dat y muestra la cantidad de visitas recién actualizado.

PHP: arrays, funciones

Ejercicio 1: Indicar si los siguientes códigos son equivalentes.

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

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

Si son equivalentes, el primero define un array entero donde de último elemento no tiene clave por la tanto se le asigna la posición 0. Mientras que en el segundo se define el mismo array pero a una clave a la vez, cuando se define el ultimo no se especifica la clave y se le asigna un 4. Por lo tanto son equivalentes.

Ejercicio 2: En cada caso, indicar las salidas correspondientes:

a)

```php
<?php
$matriz = array("x" => "bar", 12 => true);
echo $matriz["x"];
echo $matriz[12];
?>
```

Su salida es de bar1.

b)

```php
<?php
$matriz = array("unamatriz" => array(6 => 5, 13 => 9, "a" => 42));

echo $matriz["unamatriz"][6];
echo $matriz["unamatriz"][13];
echo $matriz["unamatriz"]["a"];
?>
```

devuelve:

5

9

42

c)

```
<?php
$matriz = array(5 => 1, 12 => 2);
$matriz[] = 56;
$matriz["x"] = 42;  unset($matriz[5]);  unset($matriz);
?>
```

No devuelve nada,solo se asignan valores a los elementos del array

Ejercicio 3: En cada caso, indicar las salidas correspondientes:

a)

```
<?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]";
?>
```

La salida de este código es "Has entrado en esta página a las 18 horas, con 54 minutos y 11 segundos, del 7/6/2021".

b)

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

devuelve: "5+6=11"

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;
}
```

Script:

```php
<?php
 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, limite de caracteres<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, caracter no permito>";
 return false;
 }
 }
```

```php
 echo $nombre_usuario . " es válido<br>";
 return true;
};
comprobar_nombre_usuario('jo');
?>
```

Verifica los límites de longitud y que se utilicen los caracteres permitidos del nombre de usuario..