APRENDERAPROGRAMAR.COM

OPERADORES ARITMÉTICOS EN PHP: OPERADORES BÁSICOS Y RESTO (MÓDULO). INCREMENTO Y DECREMENTO. (CU00819B)

Sección: Cursos
Categoría: Tutorial básico del programador web: PHP desde cero

Fecha revisión: 2029

Resumen: Entrega nº19 del Tutorial básico "PHP desde cero".

Autor: Enrique González Gutiérrez

OPERADORES ARITMÉTICOS BÁSICOS EN PHP

En PHP disponemos de los operadores habituales en los distintos lenguajes de programación. Estos operadores nos permiten realizar operaciones aritméticas: suma, resta, multiplicación, división, etc. así como obtener el módulo o resto de una división entre dos enteros.

Nombre	Ejemplo	Resultado	**Ejemplo** con $a = 8 y $b = 4
Suma	$a + $b	El resultado de la suma.	12
Resta	$a - $b	El resultado de la resta.	4
Multiplicación	$a * $b	El resultado de la multiplicación.	32
División	$a / $b	El resultado de la división.	2
Resto o módulo	$a % $b	El resto de la división de $a entre $b [(1)]	0

[(1)]Nota: Los números se convierten a enteros antes de efectuar la operación. Es decir, 9 % 4.5 da como resultado 1 y no 0 porque calcula el resto de 9 entre 4, no de 9 entre 4.5

El operador resto o módulo es un operador útil en algunos procesos repetitivos en programación. Fíjate en los valores que toma cuando van progresando los valores que toma una variable. En el ejemplo que mostramos a continuación sirve para contar hasta dos y empezar de nuevo repetitivamente.

$a	$a % 3
1	1
2	2
3	0
4	1
5	2
6	0
7	1
8	2

Destacar que el operador % es de uso exclusivo entre enteros. 7%3 devuelve 1 ya que el resto de dividir 7 entre 3 es 1. 8%2 devuelve 0 ya que el resto de dividir 8 entre 2 es cero. Al valor obtenido lo denominamos módulo (en otros lenguajes en vez del símbolo % se usa la palabra clave *mod*) y a este operador a veces se le denomina "operador módulo".

Aunque en otros lenguajes existe un operador de exponenciación para calcular potencias, en PHP no es así. Para calcular una potencia podemos hacer varias cosas:

a) Recurrir a multiplicar n veces el término. Por ejemplo min^3 lo podemos calcular como min*min*min. Obviamente esto no es práctico para potencias de exponentes grandes.

b) Usar un bucle que dé lugar a la repetición de la operación multiplicación n veces. Estas opciones las comentaremos más adelante.

c) Usar herramientas propias del lenguaje que permiten realizar esta operación. Esta opción es la más sencilla. Basta con escribir pow(base, exponente) para que PHP realice el cálculo de la potencia. Por ejemplo pow(2, 3) devuelve dos elevado a 3 que resulta 8.

Las operaciones con operadores siguen un **orden de prelación o de precedencia** que determinan el orden con el que se ejecutan. Con los operadores matemáticos la multiplicación y división tienen precedencia sobre la suma y la resta. Si existen expresiones con varios operadores del mismo nivel, la operación se ejecuta de izquierda a derecha. Para evitar resultados no deseados, en casos donde pueda existir duda se recomienda el uso de paréntesis para dejar claro con qué orden deben ejecutarse las operaciones. Por ejemplo, si dudas si la expresión 3 * a / 7 + 2 se ejecutará en el orden que tú deseas, especifica el orden deseado utilizando paréntesis: por ejemplo 3 * ((a / 7) + 2).

OPERADORES DE INCREMENTO Y DECREMENTO

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

++ y -- son sólo válidos para variables numéricas y sirven para incrementar una unidad el valor de la variable. Dependiendo de dónde se coloquen (antes o después de la variable) el resultado del cálculo puede diferir debido al momento en que se ejecuta la adición de la unidad.

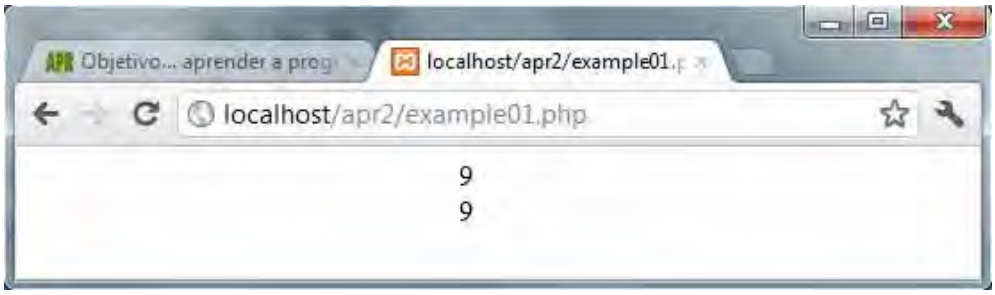
Tener en cuenta que ++, --, +=, -= y *= son expresiones que siempre se aplican sobre variables. Por ejemplo no es válido escribir 2++ porque 2 no es una variable. Todas estas operaciones pueden sustituirse por otra equivalente más evidente. Muchos programadores prefieren no usar estos

operadores porque hacen menos legible el código. A otros programadores les gusta usarlos porque les ahorra escribir. Nosotros preferimos no usarlos, pero es cierto que los puedes encontrar cuando tengas que revisar el código escrito por otra persona.

EJEMPLO

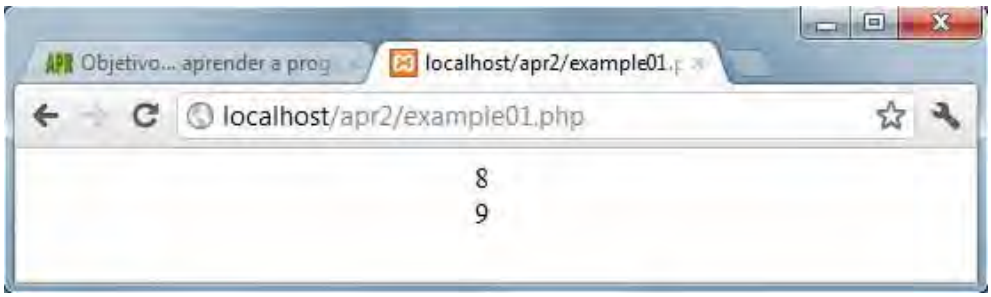
Escribe este código en un editor como Notepad++ y guárdalo con un nombre de archivo como ejemplo1.php. A continuación, sube el fichero al servidor y visualiza el resultado.

```php
<?php
$a = 8;
echo ++$a;
echo "<br />";
echo $a;
?>
```



Escribe este otro código y guárdalo con un nombre de archivo como ejemplo2.php. A continuación, sube el fichero al servidor y visualiza el resultado.

```php
<?php
$a = 8;
echo $a++;
echo "<br />";
echo $a;
?>
```



En los dos ejemplos anteriores podemos observar claramente la diferencia entre el pre-incremento y el post-incremento. Lo mismo ocurre con el pre-decremento y post-decremento.

OPERADORES DE ASIGNACIÓN

Con el uso de los operadores de asignación, podremos simplificar (escribir abreviadamente) algunas expresiones de asignación. No te recomendamos que utilices expresiones abreviadas durante el aprendizaje básico de php. No obstante, es adecuado conocer el significado de estas expresiones por si te enfrentas a tener que interpretar código escrito por otras personas.

Nombre	Ejemplo	Resultado
Suma	$a += $b;	$a = $a + $b;
Resta	$a -= $b;	$a = $a - $b;
Multiplicación	$a *= $b;	$a = $a * $b;
División	$a /= $b;	$a = $a / $b;
Resto o módulo	$a %= $b;	$a = $a % $b;

Los operadores +=, -= y *= son formas abreviadas de escribir operaciones habituales. Tener en cuenta que ++, --, +=, -= y *= son expresiones que siempre se aplican sobre variables.

EJERCICIO 1

Crea un código PHP donde crees las variables $primerNumero y $segundoNumero y asigna valor 8 al primer número y 5 al segundo número:

a) El resto de dividir el primer número entre 5.

b) El resultado de dividir el primer número entre el segundo.

c) El resultado de sumar los dos números.

Para comprobar si tu código y respuestas son correctas puedes consultar en los foros aprenderaprogramar.com.

EJERCICIO 2

Crea un código PHP donde crees las variables $a y $b y usando los operadores adecuados haz que se muestren los siguientes mensajes por pantalla. Nota: para mantener los valores iniciales de las variables y poder volver a emplearlos usa variables auxiliares, por ejemplo $inicio = $a; te permitirá mantener en

%inicio el valor original de $a y volver a recuperar el valor inicial de $a antes de hacer una nueva operación.

Mensajes a mostrar por pantalla:

Operadores de incremento

Valores iniciales: a = 4, b = 2
Operador ++ (anterior): ++a * b == 10
(Ahora el valor de a es: 5)
Operador ++ (posterior): a++ * b == 8
(Ahora el valor de a es: 5)
Operador -- (anterior): --a * b == 6
(Ahora el valor de a es: 3)
Operador -- (posterior): a-- * b == 8
(Ahora el valor de a es: 3)

Operadores de asignación compuestos

Valores iniciales: a = 4, b = 2
Asignación compuesta de suma: a += b equivale a = a + b
(Ahora el valor de a es: 6)
Asignación compuesta de resta: a -= b equivale a = a - b
(Ahora el valor de a es: 2)
Asignación compuesta de multiplicación: a *= b equivale a = a * b
(Ahora el valor de a es: 8)
Asignación compuesta de división: a /= b equivale a = a / b
(Ahora el valor de a es: 2)
Asignación compuesta de módulo: a %= b equivale a = a % b
(Ahora el valor de a es: 0)

Para comprobar si tu código y respuestas son correctas puedes consultar en los foros aprenderaprogramar.com.

Próxima entrega: CU00820B
Acceso al curso completo en aprenderaprogramar.com -- > Cursos, o en la dirección siguiente:
http://www.aprenderaprogramar.com/index.php?option=com_content&view=category&id=70&Itemid=193