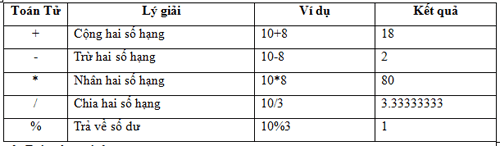
**A- Toán tử trong PHP:**

**1- Toán tử gán:**

Chúng ta đã từng tiếp xúc với toán tử này bởi việc khởi tạo 1 biến. Nó gồm ký tự đơn =. Toán tử gán lấy giá trị của toán hạng bên phải gán nó vào toán hạng bên trái.

**2- Toán tử số học:**

Là dạng phép tính giản đơn cộng, trừ, nhân, chia trong số học. Ngoài ra còn có phép chia lấy dư (%). Được sử dụng để lấy ra đơn vị dư của 1 phép toán.



**3- Toán tử so sánh:**

Là toán tử được sử dụng để thực hiện các phép toán so sánh giữa hai số hạng. Chi tiết, xem bảng bên dưới.



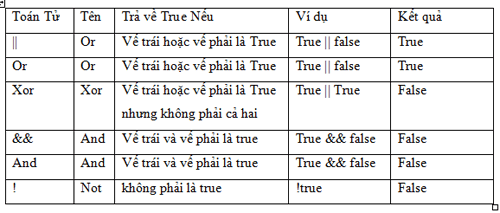
**4- Toán tử logic:**

Toán tử logic là các tổ hợp các giá trị boolean.

Ví dụ: toán tử or trở về true nếu toán tử trái hoặc toán tử phải là true.

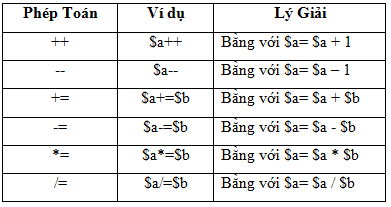
True || false à true.

Ta có bảng các toán tử như sau:



**5- Toán tử kết hợp:**

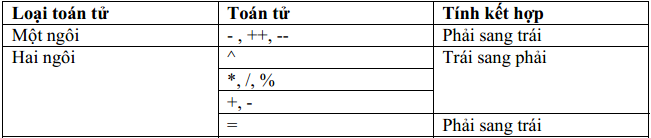
Khi tạo mã PHP, chúng ta sẽ thường nhận thấy cần phải tăng hoặc giảm lượng biến một số nguyên nào đó. Bạn sẽ thường thực hiện điều này khi chúng ta đếm 1 giá trị nào đó trong vòng lặp.



**4. Độ ưu tiên các toán tử**

Độ ưu tiên các toán tử thiết lập thứ tự ưu tiên tính toán của một biểu thức. Tóm lại độ ưu tiên trong PHP đề cập đến thứ tự các phép tính mà PHP sẽ biên dịch trước. Các toán tử và biểu thức trong php có sự liên hệ lẫn nhau, toán tử kết hợp toán hạng tạo thành biểu thức.

Bảng thứ tự ưu tiên của các toán tử số học.



Những toán tử nằm cùng một hàng có cùng độ ưu tiên và cấp độ ưu tiên đi từ trên xuống dưới. Việc tính toán biểu thức số học sẽ được tính toán từ trái qua phải và ưu tiên trong ngoặc trước kết hợp với độ ưu tiên trong bảng (như trong tính toán thường thì nhân chia trước, cộng trừ sau ưu tiên trong ngoặc).

**Ví dụ:**$t = -8 \* 4 – 3

Bước 1: tính -8 trước vì đây là oán tử một ngôi cố độ ưu tiên cao nhất. Kết quả = -8  
Bước 2: -8 \*4 vì phép nhân có độ ưu tiên cao hơn phép -. Kết quả = -32  
Bước 3**:** -32 – 3: vì đây là phép cuối cùng, ko cần phải so sánh với phép tính khác nữa. Kết quả = -35

### Độ ưu tiên toán tử luận lý

Độ ưu tiên theo thứ tự như sau: NOT -> AND -> OR

**Ví dụ**: Tính độ ưu tiên sau ( 7 > 5 &&  !(-5 > 1) || 10 == 10 ) (1)

Bước 1: trong biểu thức  này có một phép toán NOT đó là  !(-5 > 1) nên ta tính trước phép này. Trong PHP cũng như các ngôn ngữ lập trình khác biểu thức !(biểu thức)cùng ý nghĩa với biểu thức (biểu thức) == false nên biểu thức ở trên ta biến đổi thành ( (-5 > 1 ) == false ). Biểu thức này trả về giá trị TRUE vì (-5 > 1) là sai.

Bước 2: Lấy kết quả bước 1 ta viết lại biểu thức (1) như sau: ((7>5) && true || 10 == 10) Theo độ ưu tiên thì ta tính phép AND trước tức là tính ((7 > 5) && true) trước. Phép tính này trả về TRUE bởi vì (7 > 5) = true suy ra true && true => true

Bước 3: Bước này lấy kết quả ở bước 2 ta ráp vào thì biểu thức (1) như sau: (true  || 10 == 10). Phép OR sẽ trả về TRUE nếu một trong 2 biểu thức có giá trị true => biểu thức (1) là biểu thức có giá trị TRUE.

String có những loại toán tử nào

* Toán tử nối:

<?php  
$a = "Hello ";  
$b = $a . "World!"; // now $b contains "Hello World!"  
  
$a = "Hello ";  
$a .= "World!";     // now $a contains "Hello World!"  
?>

echo "thr"."ee";           //prints the string "three"  
echo "twe" . "lve";        //prints the string "twelve"  
echo 1 . 2;                //prints the string "12"  
echo 1.2;                  //prints the number 1.2  
echo 1+2;                  //prints the number 3

**array có những loại toán tử nào**

| **Array Operators** | | |
| --- | --- | --- |
| **Example** | **Name** | **Result** |
| $a + $b | Union | Union of *$a* and *$b*. |
| $a == $b | Equality | **TRUE** if *$a* and *$b* have the same key/value pairs. |
| $a === $b | Identity | **TRUE** if *$a* and *$b* have the same key/value pairs in the same order and of the same types. |
| $a != $b | Inequality | **TRUE** if *$a* is not equal to *$b*. |
| $a <> $b | Inequality | **TRUE** if *$a* is not equal to *$b*. |
| $a !== $b | Non-identity | **TRUE** if *$a* is not identical to *$b*. |

<?php  
$a = array("a" => "apple", "b" => "banana");  
$b = array("a" => "pear", "b" => "strawberry", "c" => "cherry");  
  
$c = $a + $b; // Union of $a and $b  
echo "Union of \$a and \$b: \n";  
var\_dump($c);  
  
$c = $b + $a; // Union of $b and $a  
echo "Union of \$b and \$a: \n";  
var\_dump($c);  
  
$a += $b; // Union of $a += $b is $a and $b  
echo "Union of \$a += \$b: \n";  
var\_dump($a);  
?>

Result:

Union of $a and $b:

array(3) {

["a"]=>

string(5) "apple"

["b"]=>

string(6) "banana"

["c"]=>

string(6) "cherry"

}

Union of $b and $a:

array(3) {

["a"]=>

string(4) "pear"

["b"]=>

string(10) "strawberry"

["c"]=>

string(6) "cherry"

}

Union of $a += $b:

array(3) {

'a' =>

string(5) "apple"

'b' =>

string(6) "banana"

'c' =>

string(6) "cherry"

}

<?php  
$a = array("apple", "banana");  
$b = array(1 => "banana", "0" => "apple");  
  
var\_dump($a == $b); // bool(true)  
var\_dump($a === $b); // bool(false)  
?>

<?php  
$a = array (0 => "apple", 1 => "banana");  
$b = array (1 => "banana", 0 => "apple");  
  
var\_dump($a === $b); // prints bool(false) as well  
  
$b = array ("0" => "apple", "1" => "banana");  
  
var\_dump($a === $b); // prints bool(true)  
?>