পিএইচপি টিউটোরিয়াল

পর্ব-০৮

আনোয়ার হোসেন

অ্যারে অপারেটর

উদাহরণ	অপারেটরের নাম	ব্যাখ্যা
$x == $y	Union	$x ও $y দুটি অ্যারে যোগ করা হয়েছে।
$x === $y	Equality	TRUE হবে যদি $x ও $y অ্যারে দুটির একই key/value থাকে।
$x === $y	Identity	TRUE হবে যদি $x ও $y অ্যারে দুটির key/value একই ক্রমানুসারে থাকে এবং তাদের typeও একই হয়।
$x != $y	Inequality	TRUE হবে যদি $x অ্যারেটি $y-এর সমান না হয়।
$x <> $y	Inequality	TRUE হবে যদি $x ও $y সমান না হয়।

যেমন–

```
01. <?php
02. I$x = array(1 => "Dhaka", 2=> "Rangpur");
03. $y = array("two" => "Chittagong", 3 => "Sylhet", 4 => "Khulna");
04. echo '<pre>';
05. print_r($x + $y).'<br/>';
06. print_r($y + $x);
07. ?>
```

আউটপুট

```
    Array
(
    [1] => Dhaka
[2] => Rangpur
[two] => Chittagong
[3] => Sylhet
[4] => Khulna
)
Array
(
[two] => Chittagong
    [3] => Sylhet
    [4] => Khulna
[1] => Dhaka
    [2] => Rangpur
)
```

** + দিয়ে বাম দিকের অ্যারের সাথে ডান দিকের অ্যারে যোগ হয়ে যায়। তাই উদাহরণেও দেখুন আগে বাম দিকের সব আসছে, এরপর ডান দিকেরগুলো বসছে। (আরও বিস্তারিত দেখতে চাইলে var_dump($x+$y) দিয়ে দেখতে পারেন)।

```
01. <?php
02. $x = array(0 => "Dhaka", 1 => "Rangpur");
03. $y = array("Dhaka", "Rangpur");
04. var_dump($x == $y);
05. ?>
```

আউটপুট

true আসবে। কারণ, দুটি অ্যারেরই key/value সমান। যদিও দ্বিতীয় অ্যারেটির শুরু দেয়া নেই, পিএইচপি অটোমেটিকালি সেটা সেট করে নেয়। এই অ্যারে দুটি identicalও হবে। কেননা, এদের key/value-এর typeও একই। var_dump($x === $y) দিয়ে দেখুন true আসবে।

** যেকোনো একটি key কিংবা value-তে পরিবর্তন করে দিন, তাহলে আর identical হবে না। যেমন–

```
01. <?php
02. $x = array(0 => 6, 1 => "Rangpur");
03. $y = array(0 => "6", 1 => "Rangpur");
04. var_dump($x === $y);
05. ?>
```

আউটপুট

false আসবে। তবে var_dump($x == $y) আবার true আসবে। কারণ, দুটিরই key/value একই, শুধু type আলাদা।

ক্যালকুলেশনের সময় অপারেটরগুলোর অগ্রগণ্যতা সম্পর্কে স্বচ্ছ ধারণা থাকতে হয়। তা না হলে সঠিক আউটপুট বলতে পারবেন না। যেমন– 5 + 6 x 5-এর আউটপুট কত? 35 হবে, 55 হবে না। কারণ, আগে গুণনের কাজ হবে, এরপর যোগের কাজ হবে। যেহেতু যোগ (+) অপারেটররের চেয়ে গুণনের (x) অগ্রগণ্যতা বেশি। এভাবে যে অপারেটরের অগ্রগণ্যতা বেশি, ক্যালকুলেশনের সময় সেটার কাজ করতে হবে আগে (সরল অঙ্কের মতো)। যাই হোক, আপনি চাইলে প্রথম বন্ধনী ব্যবহার করে অগ্রগণ্যতা রুখে দিয়ে প্রয়োজনীয়টির আগে অপারেশন করে নিতে পারেন। যেমন– 5 + 6 x 5-এর আউটপুট 55 পেতে চাইলে (5 + 6) x 5 এভাবে দিয়ে করতে পারেন।

পিএইচপি ম্যানুয়ালে অপারেটর অগ্রগণ্যতার একটা তালিকা আছে। সবচেয়ে উপরেরটির অগ্রগণ্যতা সবচেয়ে বেশি। এভাবে এক এক করে নিচে যাবে। এই তালিকাটি মনে রাখা খুব জরুরি। যেকোনো পরীক্ষা দিতে গেলে বিশেষ করে ওডেস্ক বা জেন্ড সার্টিফিকেশন ইত্যাদিতে এগুলো থাকে।

increment/decrement অপারেটর

++$x Pre-increment $x এক বাড়ে, এরপর $x রিটার্ন হবে।

$x++ Post-increment আগে $x রিটার্ন হবে, এরপর $x এক বাড়বে।

—$x Pre-decrement $x এক কমবে, এরপর $x রিটার্ন করবে।

$x— Post-decrement আগে $x রিটার্ন করবে, এরপর $x এক করে কমবে। যেমন–

```
01. <?php
02. $x = 2;
03. echo $x++.'<br/>';
04. echo $x;
05. ?>
```

আউটপুট

```
2
3
```

দেখুন, $x++ করাতে এক বাড়ছে, তবে echo $x++;-এর আউটপুট 2 আসছে এবং এর পরের বার $x-এর মান 3 আসছে। অর্থাৎ পরে এক (1) বাড়ছে। এর নাম হচ্ছে post-increment।

```
01. <?php
02. $x = 2;
03. echo ++$x.'<br/>';
04. echo $x;
05. ?>
```

আউটপুট

```
3
3
```

এটাতে দেখুন, প্রথমবারেই আউটপুট 3 আসছে। এটা হলো pre-increment।

এভাবে pre/post decrement-এর পরীক্ষা করে দেখতে পারেন।

টেনারি অপারেটর দিয়ে সংক্ষেপে if...else-এর কাজ করা যায়। যেমন– নিচে একটি সাধারণ if..else-এর উদাহরণটি দেখুন।

```
01. <?php
02. $x = 2;
03. $y = 5;
04. if($x < $y){
05. echo '$x is small than $y';
06. }else{
07. echo '$x is small than $y';
08. }
09. ?>
```

এই কোডটুকুই টেনারি অপারেটর দিয়ে দেখাতে পারেন এভাবে–

```
01. <?php
02. $x = 2;
03. $y = 5;
04. echo ($x < $y) ? '$x is small than $y' : '$x is small than $y';
05. ?>
```

if-এর ভেতরের অংশটুকু দেয়ার পর প্রশ্নবোধক চিহ্ন (?), এরপর কোডব্লক এবং এরপর কোলন (:) দিয়ে else-এর মধ্যে যে কোডব্লক থাকে সেই কোড কজ

ফিডব্যাক : hossain.anower009@gmail.com