

# পিএইচপি টিউটোরিয়াল

আনোয়ার হোসেন

পৃষ্ঠা-১০

সব ল্যাপ্সয়েজের মতো পিএইচপিতেও লুপ একই জিনিস। লুপ দিয়ে একটা কোড ব্লক যতবার ইচ্ছে এক্সিকিউট করানো যায়। পুনরাবৃত্তির কাজগুলো লুপ দিয়ে করা হয়। লুপে শর্ত দেয়া যায় এবং যতক্ষণ শর্ত না মিলবে, ততক্ষণ পুনরাবৃত্তি চলতেই থাকবে। ধরুন, ১ থেকে ১০০ পর্যন্ত দেখতে চান, তাহলে সেটা করতে পারেন মাত্র কয়েক লাইনেই, লুপ দিয়ে। ওয়েব অ্যাপ্লিকেশন তৈরির সময় এ ধরনের হাজার হাজার অবস্থা আসবে, যেখানে আপনাকে লুপ ব্যবহার করতে হবে। যেমন—

\* একটা ড্রপডাউন তৈরি করতে হবে, যেখানে ১-১২ পর্যন্ত থাকবে, যাতে ইউজার যেকোনো একটি সিলেক্ট করতে পারেন (তারিখ নির্বাচনের জন্য এ ধরনের ড্রপডাউন লাগতে পারে)। এখানে লুপ ব্যবহার করতে হবে।

\* ডাটাবেজে হাজার হাজার ডাটা আছে, কোয়েরি করে একটা লুপে ফেলে দিলেই একটার পর একটা ডাটা আসতে থাকবে ইত্যাদি।

পিএইচপির কয়েক ধরনের লুপ—

ফর লুপ (for loop), হোয়াইল লুপ (while loop), ফরইচ লুপ (foreach loop), ডু হোয়াইল লুপ (do...while loop)।

মূলত সবচেয়ে বেশি লাগে ফর লুপ, ফরইচ এবং এরপর হোয়াইল লুপ। ডু হোয়াইল লুপ কম লাগে।

## while loop

হোয়াইল লুপে প্রথমেই একটা শর্ত/এক্সপ্রেশন থাকে, যতক্ষণ শর্তটি পূরণ না হয়, ততক্ষণ হোয়াইল লুপ (while loop) একটা কোড ব্লককে execute করতে থাকে। যখন শর্ত মিলে যায়, তখন লুপ থেমে যায়।

## সঙ্কেত

```
while (condition){
    code to be executed;
}
```

যদি condition false হয়, তখন সেকেন্ড ব্র্যাকেটের ভেতর (কোড ব্লকটি) কোডটি এড়িয়ে যায় (ব্র্যাকেট না বলে curly braces বলা উচিত)। যখন শেষের ব্র্যাকেটটি আসবে তখন condition আবার চেক করবে, true হলে কোড আবার execute করবে। condition-এ যতবার দেয়া আছে, ততবার এভাবে চলবেই। যেমন—

```
<?php
    $i=1;
    while($i<6){
        echo 'A counter start from '. $i . '<br/>';
        $i++;
    }
?>
```

## আউটপুট

```
A counter start from 1
A counter start from 2
```

```
A counter start from 3
A counter start from 4
A counter start from 5
```

ব্যাখ্যা : প্রথমে \$i-এর মান ১ দিয়ে শুরু হয়েছে। এরপর while লুপের ভেতর শর্তটি (\$i<6) চেক করবে \$i-এর মান কত?

প্রথমবার \$i-এর মান ১-এর মানে ১ এখানে ৬-এর চেয়ে ছোট, তাই এখন কোড ব্লককে চুকবে এবং সেখানকার echo স্টেটমেন্ট বা যে কোডই থাকুক একবার এক্সিকিউট করবে এবং \$i++ - এর কারণে \$i-এর মান ১ বাড়াবে।

দ্বিতীয় বার, তাহলে এবার \$i-এর মান পাবে ২ এবং এটাও ৬-এর চেয়ে ছোট, ফলে শর্তটি (\$i<6) true রিটার্ন করবে এবং লুপটি আবার একবার কোড ব্লকটি এক্সিকিউট করবে। যেহেতু লুপের ভেতর \$i++ আছে, তাই আবার \$i-এর মান ১ বাড়াবে। সুতরাং \$i-এর মান এই লুপ শেষে হলো ৩। কেননা, প্রথমবার ১ বাড়িয়ে ২ হয়েছিল। এভাবে ৫ বার লুপটি পুনরাবৃত্তি হবে। যখন \$i-এর মান বাড়তে বাড়তে ৬ হয়ে যাবে, তখন শর্তটি (\$i<6) আর মিলবে না, কেননা ৬ তো ৬-এর ছোট নয় বরং সমান। সুতরাং শর্তটি false রিটার্ন করবে।

এবার প্রথমে \$i = 6 দিলে আউটপুট কিছুই আসবে না। তখন (\$i<6) এই শর্তেও এর সাথে মিলবে না, তাই কোড execute হবে না। অর্থাৎ এখন \$i (বা ৬) ৬-এর চেয়ে ছোট নয়।

## do...while loop

এটা while লুপের মতোই। শুধু পার্থক্য এটুকু, শর্তটি শেষে যাচাই হয়, শুরুতে করার পরিবর্তে। অর্থাৎ কমপক্ষে একবার লুপটি ঘুরবেই।

## সঙ্কেত

```
do{
    code to be executed
}while(condition);
```

নিচের উদাহরণটির কোডটিও উপরের মতো একই আউটপুট দেবে, শুধু কাজ করবে ভিন্নভাবে।

```
<?php
    $i=1;
    do{
        echo 'A counter start from '. $i . '<br/>';
        $i++;
    }
    while ($i<6);
?>
```

এখানে যদি \$i=6 দেই, তাহলে কী হবে?

```
<?php
    $i=6;
    do{
        echo 'A counter start from '. $i . '<br/>';
        $i++;
    }
    while ($i<6);
?>
```

নিচের মতো আউটপুট আসবে। কারণ, প্রথমবার শর্ত যাচাই ছাড়াই লুপটি ঘুরবে।

```
A counter start from 6
```

দেখুন লুপটি প্রথমবার চেকিং ছাড়াই একবার ঘুরেছে। প্রথমবার ঘোরার পর চেক করেছে যে লুপটি repeat হবে কি না? কিন্তু while লুপে এটা প্রথমবারই চেক হয়।

For লুপ স্টেটমেন্ট ব্র্যাকেটের ভেতর তিনটি expression নেয়, যেগুলো সেমিকোলন দিয়ে বিভক্ত হয়। প্রথমটি assignment statement(loop control variable), প্রথমবার লুপটি পুনরাবৃত্তির আগেই একবার execution হয়। দ্বিতীয়টি Boolean expression, যেটা প্রতিবার পুনরাবৃত্তির আগেই একবার এর মান নির্ণীত (evaluate) হয়, এই মান true হলে পুনরাবৃত্তি চলবে, যদি false return করে তাহলে পুনরাবৃত্তি/লুপিং বন্ধ হয়ে যাবে। তৃতীয়টি দিয়ে loop control variable-এর মান বাড়াতে বা কমাতে ব্যবহার হয়। নিচের উদাহরণে লুপটি শুরু হয়েছে \$i=1 দিয়ে এবং চলবে যতক্ষণ \$i-এর মান ৫-এর চেয়ে ছোট বা সমান হয়। আর \$i-এর মান ১ করে বাড়বে।

```
<?php
for ($i=1; $i<=5; $i++){
    echo "The number is " . $i . "<br />";
}
?>
```

## আউটপুট

```
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
```

যেকোনো expression (expr1, expr2 অথবা expr3) ফাঁকা বা একটিতে একাধিক expression থাকতে পারে, যেগুলো কমা দিয়ে বিভক্ত হবে। যদি expr2 ফাঁকা থাকে, তাহলে এর default মান true নিয়ে নেয় এবং তখন লুপটি চলতেই থাকবে, কখনও থামবে না। তবে break স্টেটমেন্ট ব্যবহার করে থামানো যাবে। যেমন— আগের লুপটিই চাইলে নিচের মতো করে করতে পারেন।

```
<?php
for ($i=1; ; $i++){
    if($i> 5){
        break;
    }
    echo "The number is " . $i . "<br />";
}
?>
```

\*\* পিএইচপিতে ফর লুপে কোলন চিহ্ন সমর্থন করে। এটা বেশ উপকারী, বিশেষ করে যখন লুপের ভেতর এইচটিএমএল লিখতে হয়। যেমন—

```
<?php for ($i=1; $i<=5; $i++):?>
<p>The number is <?php echo $i;?></p>;
<?phpendifor; ?>
```

দেখুন কীভাবে কোলন চিহ্ন ব্যবহার হয়েছে দ্বিতীয় বন্ধনীর বদলে। দ্বিতীয় বন্ধনী দিলে যেমন সেটা শেষ করতে হয় “;” চিহ্ন দিয়ে, কোলন দিলে শেষ করতে হয় endifor; এভাবে

ফিডব্যাক : [hossain.anower009@gmail.com](mailto:hossain.anower009@gmail.com)