Öğr. Gör. Serkan AKSU
http://www.serkanaksu.net

PHP'de Program Denetimi

Belli durumlarda örneğin değişkenlerin aldığı değerlere veya sayfaya yapılan ziyaretlere göre PHP programının nasıl davranacağına karar vermemiz gerekir. Bu yönlendirmeleri, **program denetimi deyimleri** ile yaparız.

Bu amaçla, **if** ve **switch** gibi karar verme deyimleri, **for**, **while**, **do..while** gibi döngü deyimleri kullanılır.

if Deyimi

Kelime anlamı **eğer** olan **if** deyimi ile programımızı karar almaya zorlarız ve bu şartın doğru veya yanlış olması durumunda ne yapacağına karar veririz. **if** deyimi şöyle yazılır:

```
<?php
    $degisken = 9;
    if($degisken < 10) {
        echo "değişken 10'dan küçük";
    }else if($degisken >= 10 && $degisken < 20) {
        echo "değişken 10 ile 20 arasında";
    }else {
        echo "değişken 20'den büyük veya eşit";
    }
?>
```

PHP, **if** ifadesinin doğru olması halinde, ifadeye ait ilk blok parantezinin içindeki komutları icra eder; bu şartlar doğru değilse ve **else if** deyimi doğru ise bu blok icra edilir.

```
<?php
    if( $parola == "" ) {
        echo "Sitemize girmek için parola yazmanız gerekir.<br>";
        echo "Lütfen parolayı yazın! <br>";
    }
?>
```

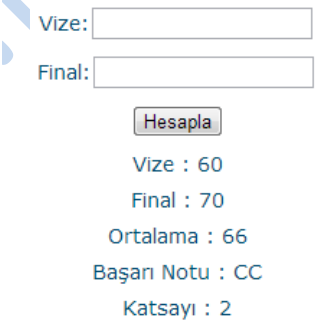
Buradaki **if** deyimi **$parola** değişkeninin boş alfanümerik olması halinde ziyaretçiyi uyaracak ve görevi bitecektir. Ziyaretçi bir parola yazmışsa, daha sonraki komutlar bu parolanın doğru olup olmadığını sınayabiliriz.

switch Deyimi

PHP'de program akışını yönlendirmekte kullandığımız bir diğer deyim, **switch** deyimidir. Adı anahtar anlamına gelen **switch** deyimi, verilen bir değişkenin değerinin sıraladığımız koşullardan hangisine uygun olduğunu sınar ve o koşula ilişkin komutları icra eder. PHP'nin yaptığı işi bitirdikten sonra **switch** deyiminin dışına çıkmasını sağlayan, **break** komutu vardır. **switch** deyimi yukarıdaki örnekte gösterilmiştir.

nothesapla.php

```
12  <body>
13  <form action="nothesapla.php" method="get">
14  <div>Vize:<input type="text" name="txtVize" value="" /></div>
15  <div>Final:<input type="text" name="txtFinal" value="" /></div>
16  <div><input type="submit" value="Hesapla" /></div>
17  </form>
18  <?php
19      @$vize = $_GET["txtVize"];
20      @$final = $_GET["txtFinal"];
21      if($vize >= 0);else $vize = 0;
22      if($final >= 0);else $vize = 0;
23
24      $ort = $vize * 0.4 + $final * 0.6;
25      if($ort >= 90 && $ort <= 100) $bn = "AA";
26      if($ort >= 85 && $ort < 90) $bn = "BA";
27      if($ort >= 80 && $ort < 85) $bn = "BB";
28      if($ort >= 75 && $ort < 80) $bn = "CB";
29      if($ort >= 65 && $ort < 75) $bn = "CC";
30      if($ort >= 58 && $ort < 65) $bn = "DC";
31      if($ort >= 50 && $ort < 58) $bn = "DD";
32      if($ort >= 00 && $ort < 50) $bn = "FF";
33
34      switch($bn)
35      {
36          case "AA" : $ks = 4.0; break;
37          case "BA" : $ks = 3.5; break;
38          case "BB" : $ks = 3.0; break;
39          case "CB" : $ks = 2.5; break;
40          case "CC" : $ks = 2.0; break;
41          case "DC" : $ks = 1.5; break;
42          case "DD" : $ks = 1.0; break;
43          case "FF" : $ks = 0.0; break;
44      }
45      echo "<div>Vize : ".$vize."</div>";
46      echo "<div>Final : ".$final."</div>";
47      echo "<div>Ortalama : ".$ort."</div>";
48      echo "<div>Başarı Notu : ".$bn."</div>";
49      echo "<div>Katsayı : ".$ks."</div>";
50  ?>
51  </body>
```



Döngüler

Bazen programımızın bir koşul gerçekleşinceye kadar belli deyimleri tekrar tekrar çalıştırmasını isteyebiliriz. Örneğin, bir tablonun başından sonuna kadar kayıtları belli formatta sayfaya yazdırmak isteyebiliriz veya bir dizinin elemanlarını tek tek işlememiz gerekebilir. Bu gibi durumlarda **döngü deyimleri** kullanırız.

while döngüsü

Bir değişkenin içeriğinin belirli bir şartı karşılaması veya karşılamaması halinde icra edilir. Burada dikkat edeceğimiz nokta, programın icrası sırasında değişkenin içeriğinin veya koşulun değişmesinin sağlanmasıdır. Aksi taktirde programımız sonsuz döngüye girer ve muhtemelen çöker. **while** döngüsü şöyle yazılır:

```
while (koşul) {
        Koşul doğru ise yapılacak işlere ilişkin komutlar
}
```

Bu yöntemin yaygın kullanıldığı alan, bir sayaçla yaptırılan işlerdir.

do..while

while döngüsü farkettiğiniz gibi, ileri sürdüğümüz şartı, işi yapmadan önce sınar; ve bu şart ortadan kalkmamışsa (yani henüz **doğru/true** ise) yapacağı işi yapar; başka bir deyişle **while** döngüsünün yapacağı iş hiç yapılmayabilir. Fakat sınamanın iş yapıldıktan sonra yerine getirildiği bir şekli de vardır: **do..while** Bu döngü ise şöyle yazılır:

```
do {
        Koşul doğru ise yapılacak işlere ilişkin komutlar
}
while (koşul);
```

Burada gördüğünüz gibi **do..while** döngüsü en az bir kere icra edilir; çünkü şartın sınanması yapılacak işe ilişkin komutlardan sonra gelmektedir. Bu döngünün aradığı şartın döngünün yaptığı işlerin sonucu veya kod bölümünde bizim tarafımızdan gerçekleştirilmesi için gerekli komutların bulunmasına dikkat etmelisiniz. Yoksa, bu döngü de sonsuz kere döner! Yukarıda örneği, bu yöntemle yazalım:

```php
<?php
    $sayac = 1;
    do {
        echo "<font size = $sayac >";
        echo "<p>Döngü ile font büyüklüğü!</p>";
        echo "</font>";
        $sayac++;
    }while ( $sayac <= 7 ) ;     //sayac 7 olunca döngü bitecek
?>
```

Bu döngüyü yazarken, **while** satırının sorunda **noktalı virgül (;)** bulunduğuna dikkat edin.

for döngüsü

PHP'de Web programlarımız, döngünün belirli bir sayıda olmasını ve mesela bu sayının bizim istediğimiz basamaklarda artmasını gerektiriyorsa, döngüyü **for** deyimiyle kurabiliriz. Ayrıca **for** deyimi sayaç gibi şartın yerine gelmesini sağlayacak arttırma veya eksiltme işlemlerini kendisi yapacağı için, sonsuz döngüye girme tehlikesi de hemen hemen yoktur. Bu döngü şöyle yazılır:

```
1  <html>
2  <head>
3  <meta htt- -~~iv="Content-Type" content="text/html; charset=UTF-8" />
4  <title>Pi.. ___iler</title>
5  </head>
6  <body>
7  <?php
8      echo "<h1>ÇARPIM TABLOSU</h1>";
9      echo "<table width='400' border='1'>";
10     $i=1;
11     while($i<10)
12     {
13         echo "<tr>";
14         for($j=1;$j<10;$j++)
15         {
16             $c=$i*$j;
17             echo "<td align='center'>".$c."</td>";
18         }
19         echo "</tr>";
20         $i++;
21     }
22     echo "</table>";
23 ?>
24 </body>
25 </html>
```

Programımız, **while** döngüsü içindeki iken, sonsuz döngüden kurtulabilmek için, **$i** değişkeninin değerini arttırıyor.

Döngüyü sona erdirmek için: break

Programlarda belli koşuların sağlanması halinde döngünün durdurulmasını gerekebilir.

```
<?php
    $limit = 5;
    for($sayac = 1; $sayac <= 10 ; $sayac++ ) {
        if ($sayac == $limit )break;
        echo "<br>Sayaç :".$sayac;
    }
?>
```

Yukarıdaki örnekte, **$ayac** değişkeninin değeri **$limit**'e ulaştığında döngü kesilir ve program döngüden sonraki satırdan itibaren devam eder.

Döngüyü sürdürmek için: continue

Kimi zaman da döngünün sadece belirli bir durumda kendisinden beklenen işi yapmamakla birlikte böyle tümüyle kesilmesini de gerektirmez. Belli bir koşul sağlandığında döngünün başına gidilir ve döngüye devam edilir.

```
<?php
    $atla = 5;
    for($sayac = 1; $sayac <= 10 ; $sayac++ ) {
        if ($sayac == $atla )continue;
        echo "<br>Sayaç :".$sayac;
    }
?>
```

Bu örnekte **$sayaç** değeri **$atla** değerine eşit olduğu zaman döngü başa gider ve devam eder.