if...else Kontrol Yapısı

Belirli kodları belirli şartlara bağlı olarak çalıştırmak için kullanılır. Basit bir ifadeyle eğer böyleyse şöyle yap, şöyleyse böyle yap diyebilmek için kullanılır. Kullanımı şu şekildedir.

```
if(koşul1){
  // koşul1 sağlanırsa çalışacak kodlar
}
elseif(koşul2){
  // koşul2 sağlanırsa çalışacak kodlar
}
elseif(koşul3){
  // koşul3 sağlanırsa çalışacak kodlar
}
.
.
.
else{
  // yukarıdaki hiçbir koşul sağlanmazsa çalışacak kodlar
}
```

Kontol yapısı if ile başlar isteğe bağlı olarak diğer şartları belirtmek için elseif ile, hiçbir şart sağlanmazsa else ile devam eder. Dikkat edilirse else ifadesinde herhangi bir koşul belirtmedik. Dikkat edilmesi gereken diğer husus ise if, elseif ve else ifadelerinden sonra noktalı virgül (;) konulmamasıdır.

Eğer if, elseif veya else'den sonra çalışması gereken kod bir tane ise { } parantezlerini kullanmak da gerekmez.

Aşağıdaki örneği inceleyiniz.

```
<?php
$a=15;
$b=9;
if($a>$b)
   echo "a değişkeni b'den büyüktür.";
elseif($a<$b)
   echo "a değişkeni b'den küçüktür.";
else
   echo "a değişkeni b'ye eşittir.";
?>
```

Görüldüğü gibi if, elseif ve else'den sonra birer komut olduğundan { } parantezlerini kullanmadık.

Örnek: Tanımlı olan sayı pozitif ise sayının kendisini ve karesini, değilse *Sayı pozitif değil* mesajını ekranda gösterelim.

```
<?php
$sayi=-3;
if($sayi>0){
   echo "Sayı=$sayi<br>";
   echo "Karesi=",$sayi*$sayi;
```

```
}
else
   echo "Sayı pozitif değil";
?>
```

Görüldüğü üzere if'den sonra çalışmasını istediğimiz iki tane komut olduğundan { } parantezlerini kullandık. else'den sonra ise tek komut olduğundan { } kullanmadık. Ayrıca tek bir koşula göre işlem yapacağımız için elseif kısmını da kullanmadık.

Örnek: Tanımlı olan kullanıcı adının ve şifrenin sırasıyla *mustix* ve *muric* olup olmadığını kontrol eden php kodunu yazalım.

```
<?php
$kullanici_adi="hayrix";
$sifre="muric";
if ($kullanici_adi=="mustix" and $sifre=="muric")
   echo "Giriş Başarılı";
else
   echo "Kullanıcı adı yada şifre yanlış!";
?>
```

Kod çalıştığında ekrana *Kullanıcı adı yada şifre yanlış!* mesajını yazacaktır. Çünkü if içerisinde and (ve) mantıksal operatörü ile her iki şartın sağlanması gerektiği söylendiği halde koşullardan biri sağlanmadığı için else kısmı çalışacaktır.

Örnek: Tanımlı olan sayı tek ise sayıyı bir arttıran php kodunu yazalım.

```
<?php
$sayi=7;
if ($sayi%2==1) // sayının 2'ye bölümünden kalan 1 ise
   $sayi++;
echo "Sayı=$sayi"; // ekrana Sayı=8 yazar
?>
```

Duruma göre kontrol yapısının sadece if kısmını da kullanabilirsiniz. else kısmı olmak zorunda değil. Bir şeye daha dikkat ediniz. { } parantezleri kullanılmadığından if altında sadece $sayi++; komutu çalışmaktadır. echo komutu her halükarda çalışacaktır. Sayıyı değiştirerek farklı sonuçları gözlemleyebilirsiniz.

Şimdi kontrol yapısının farklı bir kullanımına bakalım. Bu kullanım sadece bir değişkene atanacak değeri belirlerken ya da fonksiyona gönderilecek değeri belirlerken kullanılan bir yapıdır. Kullanımı şöyledir:

$degisken=(koşul)?koşul sağlanırsa:koşul sağlanmazsa;

Örneğimizde eğer tanımlı olan sayı tek ise sayının küpünü, değil ise karesini hesaplayıp ekranda yazdıralım.

```
<?php
$sayi=5;
$sonuc=($sayi%2==1)?$sayi*$sayi*$sayi:$sayi*$sayi;
echo "Sayı=$sayi<br>Sonuç=$sonuc";
?>
```

Aynı örneği if yapısını açıkça yazarak yapalım.

```
<?php
$sayi=5;
if($sayi%2==1)
   $sonuc=$sayi*$sayi*$sayi;
else
   $sonuc=$sayi*$sayi;

echo "Sayı=$sayi<br>Sonuç=$sonuc";
?>
```

Örnek: Eğer tanımlı olan boolean tipindeki değişken true ise false, false ise true yapalım.

```
<?php
$durum=false;
$durum=($durum==true)?false:true;
echo $durum;
?>
```

Yapılan işi bir cümle olarak söylemek gerekirse şöyle denilebilir. Eğer $durum değişkeni true ise false yap, değilse true yap.
Bir değişkenin true olup olmadığı şöyle de kontrol edilebilir.

```
<?php
$durum=false;
$durum=($durum)?false:true;
echo $durum;
?>
```

Ekranda true için 1, false için bir şey görünmeyecektir. Açık bir şekilde yazacak olursak:

```
<?php
$durum=false;

if($durum)
   $durum=false;
else
   $durum=true;

echo $durum;
?>
```

Dikkat ederseniz if içerisindeki değişken herhangi bir şeyle karşılaştırılmıyor. Sanki burada bir koşul yokmuş gibi geliyor. Burada aslında sorulan şudur:

if ($durum)->Eğer $durum değişkeninde bir değer varsa yada $durum değişkeni true ise

Not: Bir değişkenin değeri sıfırdan farklı ise o değişken true kabul edilir. Aşağıdaki örneği inceleyiniz.

```
<?php
$sayi=-5;
```

```
if ($sayi)
   echo "sayi değişkeni true'dur. Yada sıfırdan farklı bir değere sahiptir.";
else
   echo "sayi değişkeni false'dur. Yada sıfırdır. Yada boştur.";

echo "<br>";

$adi="";
if ($adi)
   echo "adi değişkeni true'dur. Yada sıfırdan farklı bir değere sahiptir.";
else
   echo "adi değişkeni false'dur. Yada sıfırdır. Yada boştur.";

echo "<br>";

$durum=false;
if ($durum)
   echo "durum değişkeni true'dur. Yada sıfırdan farklı bir değere sahiptir.";
else
   echo "durum değişkeni false'dur. Yada sıfırdır. Yada boştur.";
?>
```

Örnek: Eğer $sonuc değişkeninde bir değer varsa ekrana *Sonuç hesaplandı*, yoksa *Sonuç hesaplanamadı* ifadelerini yazdıralım.

```
<?php
$sonuc="";
echo ($sonuc)?"Sonuç hesapladı.":"Sonuç hesaplanamadı.";
?>
```

Burada kontrol yapısından dönen değer doğrudan echo fonksiyonuna verilmiştir.

switch case Kontrol Yapısı

Seçilen değişkenin değerinin belirli durumlara uyup uymadığını kontrol eden yapıdır. Değişkenin değeri, belirtilen birçok durumdan hangisine uyuyorsa o durum altındaki komutlar çalışır. Bunu, ben if ile de yapabilirim diye düşünebilirsiniz. Doğru düşünüyorsunuz. switch ile yaptığınız işi if yapısı ile de yapabilirsiniz. Burada switch yapısının avantajı daha düzenli kod yazmamızı sağlamasıdır. Kullanımı şu şekildedir.

```
switch ($degisken){
case durum1:
   // durum1 sağlanırsa çalışacak kodlar
   break;
case durum2:
   // durum2 sağlanırsa çalışacak kodlar
   break;
.
.
.
default:
   // hiçbir durum sağlanmazsa çalışacak kodlar
}
```

Seçilen değişkenin değeri hangi duruma uyarsa o durum altındaki komutlar çalışır. **break** ifadesi, durum sağlanırsa başka durumlara bakılmasını engellemek için kullanılır. Eğer break kullanmazsanız durumlardan biri sağlandığında, ondan sonraki tüm durumlar da çalıştırılır. Bu genelde istenmez. Son olarak; **default** ifadesinden sonra break kullanılmadığına dikkat ediniz. Çünkü default'tan sonra başka durum olmadığından break kullanmaya gerek yoktur. Aşağıdaki örneği inceleyiniz.

```
<?php
$sayi=2;

switch ($sayi){
   case 0: echo "Sayı 0'dır."; break;
   case 1: echo "Sayı 1'dir."; break;
   case 2: echo "Sayı 2'dir."; break;
   default: echo "Sayı 0,1 ve 2 değildir.";
}
?>
```

Burada $sayi değişkeninin değerlerine bakıyoruz. Değeri 2 olduğundan ekrana *Sayı 2'dir.* yazacaktır. echo ve break komutlarını aynı satırda buradaki gibi yazabilirsiniz. Şimdi bu işlemi if yapısı ile yapalım.

```
<?php
$sayi=2;

if($sayi==0)
   echo "Sayı 0'dır.";
elseif($sayi==1)
   echo "Sayı 1'dir.";
elseif($sayi==2)
   echo "Sayı 2'dir.";
else
   echo "Sayı 0,1 ve 2 değildir.";
?>
```

Görüldüğü üzere aynı işlemi rahatlıkla if ile de yapabilirsiniz. Tercih sizindir. Bazı durumlarda switch yapısı daha kolay olabilir.

Örnek: Beş üzerinden tanımlı notun yazı karşılığını ekrana yazdıralım.

```
<?php
$notu=2;

switch ($notu){
   case 0: echo "Başarısız"; break;
   case 1: echo "Zayıf"; break;
   case 2: echo "Geçer"; break;
   case 3: echo "Orta"; break;
   case 4: echo "İyi"; break;
   default: echo "Pekiyi";
}
?>
```

Örnek: Şimdi de sayının tek ya da çift olma durumunu ekrana yazdıralım.

```
<?php
$sayi=23;
switch ($sayi%2){ // sayının 2'ye bölümünden kalan
   case 0: echo "Sayı çifttir."; break;
   default: echo "Sayı tektir."; break;
}
?>
```

Örnek: Tanımlı olan işleme göre iki sayıyı işleme alıp sonucu ekrana yazalım.

```
<?php
$islem="*";
$sayi1=16;
$sayi2=4;

switch ($islem){
   case "+": $sonuc=$sayi1+$sayi2; break;
   case "-": $sonuc=$sayi1-$sayi2; break;
   case "*": $sonuc=$sayi1*$sayi2; break;
   default: $sonuc=$sayi1/$sayi2;
}

echo "Sonuç=$sonuc";
?>
```

Kontrol edeceğiniz değişken sayı olabileceği gibi buradaki gibi string bir ifade de olabilir. Aynı zamanda boolean, sabit ve dizi değişkenleri de burada kullanabilirsiniz.

Örnek: Tanımlı olan kullanıcıya hoş geldiniz mesajı yazdıralım.

```
<?php
$kullanici="Büşra";
$cinsiyet="bayan";

switch ($cinsiyet){
   case "bay": echo "Hoşgeldiniz $kullanici Bey"; break;
   default: echo "Hoşgeldiniz $kullanici Hanım";
}
?>
```

Örnek: Tanımlı olan mevsime göre ayları ekrana yazdıralım.

```
<?php
$mevsim="ilkbahar";

switch ($mevsim){
   case "ilkbahar": echo "Mart Nisan Mayıs"; break;
   case "yaz": echo "Haziran Temmuz Ağustos"; break;
   case "sonbahar": echo "Eylül Ekim Kasım"; break;
   default: echo "Aralık Ocak Şubat";
}
?>
```

Bu kez değişik bir örnek yapalım. Pek kullanılmasa da durum kısmında aşağıdaki gibi bir karşılaştırma da yapabilirsiniz.

Örnek: Sayının negatif, pozitif ya da sıfır olma durumunu ekrana yazdıralım.

```
<?php
$sayi=0;

switch ($sayi){
   case 0: echo "Sayı sıfırdır.";break;
   case $sayi>0: echo "Sayı pozitiftir.";break;
   default: echo "Sayı negatiftir.";
}
?>
```

Not: Burada önce sayının sıfır olma durumunu kontrol ediniz. Aksi halde hatalı sonuç alırsınız. Sıfırdan farklı sayılarla çalışırken istediğiniz sırada durumları ifade edebilirsiniz.

Örnek: 100 üzerinden tanımlanmış bir notun 5 üzerinden karşılığını ekrana yazdıralım.

```
<?php
$notu=64;

switch ($notu){
   case 85<=$notu: echo "5-Pekiyi"; break;
   case 70<=$notu: echo "4-İyi"; break;
   case 55<=$notu: echo "3-Orta"; break;
   case 45<=$notu: echo "2-Geçer"; break;
   case 25<=$notu: echo "1-Zayıf"; break;
   default: echo "Başarısız";
}
?>
```

Örnek: Tanımlı olan 3 sayıdan en büyüğünü bulalım.

```
<?php
$sayi1=10;
$sayi2=7;
$sayi3=15;

switch (true){
   case ($sayi1>=$sayi2 and $sayi1>=$sayi3): $eb=$sayi1; break;
   case ($sayi2>=$sayi1 and $sayi2>=$sayi3): $eb=$sayi2; break;
   default: $eb=$sayi3;
}

echo "En büyük sayı: $eb";
?>
```

Görüldüğü gibi durum kısmında mantıksal operatörler de kullanabilirsiniz. Bunun yanında true kısmına bakarsanız burada biraz farklı bir işlem yapılmıştır. Zira kontrol edilen bir değişken değil tam tersine durumdur. Yani durumlardan hangisinin true olduğunu kontrol ettik. Normalde hep bir değişkenin hangi duruma uyduğunu kontrol ediyorduk.

Örnek: Tanımlı olan sayıya karşılık gelen günü ekrana yazdıralım.

```
<?php
$gun=4;
switch ($gun){
   case 1: echo "Pazartesi";break;
   case 2: echo "Salı";break;
   case 3: echo "Çarşamba";break;
   case 4: echo "Perşembe";break;
   case 5: echo "Cuma";break;
   case 6: echo "Cumartesi";break;
   default: echo "Pazar";
}
?>
```

Örnek: Hafta içi çalışanların maaşına 50 TL, hafta sonu çalışanların maaşına 70 TL ekleyelim.

```
<?php
$maas=500;
$gun="salı";

switch ($gun){
   case "pazartesi":
   case "salı":
   case "çarşamba":
   case "perşembe":
   case "cuma":$maas+=50;break;
   case "cumartesi":
   default: $maas+=70;
}

echo "Maaş=$maas";
?>
```

Oldukça ilginç bir kullanım. Biraz açıklamaya çalışalım. Bir durumdan sonra komut yazmak zorunda değilsiniz. Bu, sonraki duruma geçmek için kullanılır. Hafta içindeki günlerden biri ise pazartesi, salı, çarşamba, perşembe ya da cuma olması fark etmiyor. Eğer bu durumlardan birine girilirse break olmadığı için cuma'ya kadar gidilir ve cuma durumunda 50 TL eklenir. Sonra break ile hafta sonuna geçilmesi engelleniyor. Aynı mantıkla cumartesi için break kullanılmayarak pazar gününe gidiliyor ve 70 TL ekleniyor. Diyelim ki; günlerden salı olsun. Salı durumuna girildi break olmadığı için cuma'ya kadar gidildi ve 50 TL eklendi ve break ile çıkıldı. Diyelim ki cumartesi; break olmadığı için pazar'a gidilir ve 70 TL eklenir. Tabiki bu bizim algoritmamızdır. Bilindiği gibi sonuca giden yol tek değildir. Siz de kendi çözümünüzü geliştirebilirsiniz.

for Döngüsü

İstenilen komutları istenilen sayıda çalıştırmak için kullanılır. Döngüyü kontrol etmek için döngü sayacı olarak adlandırılan bir değişken kullanılır. Bu genelde $i değişkenidir. Biz de burada döngü değişkeni olarak $i değişkenini kullanacağız.

Döngü değişkenleri döngünün sınırlarını belirlemek için kullanılır. Ayrıca döngü altında çalışan komutların bir kısmı olarak da kullanılabilir. Örneğin bir dizinin indisi yada ekrana yazılan mesajın bir bölümü olabilirler. Eğer döngünün sınırlar iyi belirlenmezse döngü; sonsuz döngüye, diğer bir ifadeyle kısır döngüye girebilir. Bu durumda tarayıcımız yanıt vermeyebilir.

for döngüsünün kullanımı şu şekildedir.

```
<?php
for(/*başlangıç*/ ; /*koşul*/ ; /*artış miktarı*/){
   // döngü altında çalışacak komutlar
}
?>
```

Başlangıç: Döngünün kaçtan balayacağını belirtmek için kullanılır.

Koşul: Döngü için belirtilen koşuldur. Bu koşul sağlandığı sürece döngü döner.

Artış miktarı: Döngü değişkeninin kaçar kaçar artacağını belirtmek için kullanılır.

Örneğin 1'den 10'a kadar dönen ve ekrana Merhaba Dünya yazan bir döngü kuralım.

```
<html>
<body>
<?php
for($i=1;$i<=10;$i++){
   echo "Merhaba Dünya";
   echo "<br>";
}
?>
</body>
</html>
```

İlk örnek olduğundan örneği açıklayılım: Döngü değişkeni olarak $i kullanılmıştır. **$i=1** tanımlamasıyla döngünün 1'den başlayacağı belirtilmiştir. **$i<=10** koşulu ile döngünün, **$i**'nin 10'dan küçük yada 10'a eşit olduğu sürece döneceği belirtilmiştir. **$i++** ifadesi ile de her döngüden sonra **$i**'nin 1 arttırılacağı belirtilmiştir.

Örneğimizin ekran çıktısı ise aşağıdaki gibidir.

```
Merhaba Dünya
Merhaba Dünya
Merhaba Dünya
Merhaba Dünya
Merhaba Dünya
Merhaba Dünya
```

```
Merhaba Dünya
Merhaba Dünya
Merhaba Dünya
Merhaba Dünya
```

Örnek: Ekranda alt alta 1'den 10' kadar olan sayıları yazdıralım.

```
<?php
for ($i=1;$i<=10;$i++)
   echo "$i<br>";
?>
```

Not: Burada **for** altında çalışan tek komut olduğundan { } parantezlerini kullanmadık.

Aynı örneği iki farklı şekilde yazalım. Buradaki kullanım şekillerine dikkat ediniz.

```
<?php
for ($i=1;;$i++){
   if ($i>10)
      break; // döngüyü kırmak için kullanılır.
   echo "$i<br>"; // Dikkat: Bu komut if yapısına bağlı değildir. {}
parantezleri yok.
}
?>
```

```
<?php
$i=1;
for (;;){
   if ($i>10)
      break;
   echo "$i<br>";
   $i++;
}
?>
```

Örnek: Şimdiki örneğimizde 1 ile 100 arasındaki sayılardan 5'e bölünebilenlerin toplamını bulalım.

```
<?php
$toplam=0;
for ($i=1;$i<=100;$i++)
   if ($i%5==0) // $i'nin 5'e bölümünden kalan 0 ise
      $toplam=$toplam+$i;
echo "Toplam=$toplam";
?>
```

Burada şunu hatırlatmakta fayda var. for altında iki komut görünüyor ve { } parantezlerini kullamamız gerekiyor gibi gelebilir. Ancak if yapısı $toplam=$toplam+$i; komutunu, for döngüsü de if yapısını

tutmaktadır. Dolayısıyla { } parantezlerini kullanmak gerekmez. Basit bir ifadeyle { } parantezleri kullanılmadığında for, ilk noktalı virgüle (;) kadar olan komutları çalıştırır.

Aynı örneği farklı bir şekilde tekrar yapalım.

```
<?php
$toplam=0;
for ($i=1;$i<=100;$i++)
   $toplam+=($i%5==0)?$i:0;
echo "Toplam=$toplam";
?>
```

Burada ($i%5==0)?$i:0 komutundan koşula göre ya $i yada 0 döner.

Aslında hiç if yapısı kullanmadan sadece döngü değişkenini beşer beşer arttırarak da aynı işlemi yapabilirsiniz. Ancak o zaman $i sıfırdan başlatılmalıdır.

```
<?php
$toplam=0;
for ($i=0;$i<=100;$i+=5)
   $toplam+=$i;
echo "Toplam=$toplam";
?>
```

Yukarıdaki örnekte de görüldüğü üzere döngü değişkenini istediğiniz kadar arttırabilirsiniz.

Örnek: 7'den başlayarak üçer üçer 50'ye kadar yazdıralım.

```
<?php
for ($i=7;$i<=50;$i+=3){
   echo "$i<br>";
}
?>
```

Örnek: Bu örnekte ise döngümüz toplam 1000'den büyük olduğunda dursun. Döngü değişkeni de yedişer yedişer artsın.

```
<?php
$toplam=0;
for ($i=1;;$i+=7){
   $toplam+=$i;
   if ($toplam>1000)
      break;
}
echo "Toplam=$toplam";
?>
```

Gördüğünüz gibi döngüyü sonlandırmak için; koşul, döngü değişkenine bağlı değildir.

Azalan döngüler de yapabilirsiniz. Bunun için koşulu dikkatli yazmak gerekir. Aksi halde sonzuz döngüye girilebilir. Ayrıca döngü değişkeni azaltılmalıdır.

Örnek: Tanımlı olan metni küçükten büyüğe doğru başlık şeklinde yazdıralım.

```
<?php
for ($i=6;$i>=1;$i--)
   echo "<h$i>Her şakanın yarısı gerçektir.</h$i>";
?>
```

Örnek: 2010'dan 1920'ye kadar olan yılları açılır listeye ekleyelim.

```
<?php
echo "<select name='yillar'>";
for ($i=2010;$i>=1920;$i--)
   echo "<option value='$i'>$i</option>";
echo "</select>";
?>
```

for Döngüsüyle İlgili Örnekler

Örnek: Tanımlı olan sayının çarpım tablosunu ekrana yazdıralım.

```
<?php
$sayi=7;
for ($i=1;$i<=10;$i++)
   echo "$sayi x $i = ",$sayi*$i,"<br>";
?>
```

Not: Unutmayın artimetik işlemler çift tırnak içinde yapılmaz.

Örnek: Tanımlı olan sayının faktöriyelini bulalım.

```
<?php
$sayi=5;
$faktoriyel=1;
for($i=1;$i<=$sayi;$i++)
   $faktoriyel*=$i;
echo "$sayi!=$faktoriyel";
?>
```

Örnek: Tanımlı olan sayının tam bölenlerini bir diziye atayıp ekrana yazalım.

```
<?php
$sayi=120;
for($i=2;$i<=$sayi/2;$i++)
   if ($sayi%$i==0)
      $bolenler[]=$i;

echo "<u>$sayi sayısının tam bölenleri:</u><br>";
for ($i=0;@$bolenler[$i];$i++)
   echo $bolenler[$i],"<br>";
?>
```

Örnek: Ekrana küçükten büyüğe doğru tanımlı olan metni yazdıralım.

```
<?php
$mesaj="ilim ilim bilmektir ilim kendin bilmektir";
for ($i=1;$i<=7;$i++)
   echo "<font size='$i'>$mesaj</font><br>";
?>
```

şimdi de yukarıdaki mesajın devamını azalan döngü ile ekrana yazdıralım.

```
<?php
$mesaj="sen kendini bilmezsen ilim nice okumaktır";
for ($i=7;$i>=1;$i--)
   echo "<font size='$i'>$mesaj</font><br>";
?>
```

Burada koşula dikkat ediniz. Arttırma değil azaltma yapıldığına da dikkat ediniz.

Örnek: for döngüsü ile 5 satır 3 sütunlu bir tablo oluşturalım.

```
<?php
echo "<table border='1' width='200px'>";
for ($i=1;$i<=5;$i++){
   echo "<tr>";
   echo "<td> </td>";
   echo "<td> </td>";
   echo "<td> </td>";
   echo "</tr>";
}
echo "</table>";
?>
```

Örnek: for döngüsü ile 10 satır 2 sütünlu bir tablo oluşturalım. Ancak satırların renklerini farklı gösterelim.

```
<?php
echo "<table border='1' width='200px'>";
for ($i=1;$i<=10;$i++){
   echo "<tr bgcolor='",($i%2)?"#abda68":"#d0f896","'>";
   echo "<td> </td>";
   echo "<td> </td>";
   echo "</tr>";
}
echo "</table>";
?>
```

Örnek: 1'den 20'ye kadar olan sayıları ve karesini bir tablo içinde ekranda gösterelim.

```
<?php
echo "<table border='1' width='200px'>";
echo "<tr>";
echo "<th>Sayı</th>";
echo "<th>Karesi</th>";
echo "</tr>";
for ($i=1;$i<=20;$i++){
   echo "<tr>";
   echo "<td>$i</td>";
   echo "<td>",$i*$i,"</td>";
   echo "</tr>";
}
```

```
echo "</table>";
?>
```

Örnek: Açılır listeye il plaka numaralarını dolduralım.

```
<?php
echo "İl Seçiniz: <select name='iller'>";
for ($i=1;$i<=81;$i++){
   echo "<option value='$i'>$i</option>";
}
echo "</select>";
?>
```

Şimdi aynı örneği 10'dan küçük sayıların başına 0 koyarak yapalım.

```
<?php
echo "İl Seçiniz: <select name='iller'>";
for ($i=1;$i<=81;$i++){
   echo "<option value='$i'>",($i<10)?"0".$i:$i,"</option>";
}
echo "</select>";
?>
```

Örnek: Tanımlı olan dizinin elemanlarını ekranda alt alta yazdıralım.

```
<?php
$takim=array("Kaan","Gürkan","Abdullah","Emin","Burak");
for ($i=0;$i<=4;$i++)
   echo $i+1,") ",$takim[$i],"<br>";
?>
```

Örnek: 1 ile 100 arasındaki sayılardan 9'a tam bölünenleri bir diziye atayıp diziyi ekrana yazdıralım.

```
<?php
for ($i=1;$i<=100;$i++)
   if($i%9==0)
      $sayilar[]=$i;

for ($i=0;@$sayilar[$i];$i++)
   echo $sayilar[$i]," ";
?>
```

Burada 9'a tam bölünebilen sayılar diziye katılmaktadır. İkinci döngüdeki @$sayilar[$i] koşulu; $i indisli bir dizi elemanı varsa anlamındadır. Eğer yoksa uyarı vermesin diye başına @ işareti koyduk.

Not: Bir komutun verdiği uyarıyı ekranda görmek istemiyorsanız o komutun başına @ işareti koymalısınız.

Örnek: Tanımlı olan dizinin elemanlarını açılır listeye ekleyelim.

```
<?php
$aylar=array("Ocak","Şubat","Mart","Nisan","Mayıs","Haziran","Temmuz","Ağus
tos","Eylül","Ekim","Kasım","Aralık");
echo "<select name='aylar'>";
for ($i=0;$i<=11;$i++)
   echo "<option value='",$i+1,"'>",$aylar[$i],"</option>";
echo "</select>";
?>
```

Örnek: Tanımlı olan notlar dizisindeki zayıf ve iyi notların sayısını ekranda gösterelim.

```
<?php
$notlar=array(64,52,32,96,15,77,81,25,78,55,65,40);
$zayif=0;
$iyi=0;
for ($i=0;@$notlar[$i];$i++){
   if ($notlar[$i]<45)
      $zayif++;
   else
      $iyi++;
}
echo "Zayıf not sayısı=$zayif <br> İyi not sayısı=$iyi";
?>
```

Size fikir vermesi açısından farklı bir çözümü de sizlerle paylaşalım.

```
<?php
$notlar=array(64,52,32,96,15,77,81,25,78,55,65,40);
$zayif=0;
$iyi=0;
for ($i=0;@$notlar[$i];$i++){
   $zayif+=($notlar[$i]<45)?1:0;
   $iyi+=($notlar[$i]>=45)?1:0;
}
echo "Zayıf not sayısı=$zayif <br> İyi not sayısı=$iyi";
?>
```

Örnek: Şimdi de iç içe döngülere bir örnek verelim. Örneğimizde 10'a kadar olan sayıların çarpım tablosunu ekrana yazdıralım.

```
<?php
for ($i=1;$i<=10;$i++){
   for ($j=0;$j<=10;$j++)
      echo "$i x $j = ",$i*$j,"<br>";
   echo "<br>";
}
?>
```

Örnek: Tanımlı olan notlar dizisindeki en büyük notu bulalım.

```
<?php
$notlar=array(64,52,32,96,15,77,81,25,78,55,65,40);
$enbuyuk=0;
for ($i=0;@$notlar[$i];$i++)
   $enbuyuk=($notlar[$i]>$enbuyuk)?$notlar[$i]:$enbuyuk;
echo "En büyük not=$enbuyuk";
?>
```

Örnek: Tanımlı olan notlar dizisindeki notları grafik olarak ekranda gösterelim. Bunun için dosyamınızın kayıtlı olduğu yerde cubuk.png resminin olduğunu kabul edelim.

```
<?php
$notlar=array(64,52,32,96,15,77,81,25,78,55,65,40);
for ($i=0;@$notlar[$i];$i++)
   echo "<img src='cubuk.png' width='20' height='",2*$notlar[$i],"'> ";
?>
```

Şimdi aynı grafiği tablo içinde gösterelim. Bu kez notları da altına yazalım

```php
<?php
$notlar=array(64,52,32,96,15,77,81,25,78,55,65,40);
echo "<table border='0'>";

echo "<tr>";
for ($i=0;@$notlar[$i];$i++)
   echo "<td valign='bottom'><img src='cubuk.png' width='20'
height='",2*$notlar[$i],"'></td>";
echo "</tr>";

echo "<tr>";
for ($i=0;@$notlar[$i];$i++)
   echo "<td>",$notlar[$i],"</td>";
echo "</tr>";

echo "</table>";
?>
```

Örnek: Bir dizideki notları küçükten büyüğe doğru sıralayalım.

```php
<?php
$notlar=array(64,52,32,96,15,77,81,25,78,55,65,40);
$degisiklik=true;
for(;$degisiklik;){
   $degisiklik=false;
   for ($i=1;@$notlar[$i];$i++){
      if ($notlar[$i]<$notlar[$i-1]){
         $gecici=$notlar[$i-1];
         $notlar[$i-1]=$notlar[$i];
         $notlar[$i]=$gecici;
         $degisiklik=true;
      }
   }
}

for ($i=0;@$notlar[$i];$i++)
   echo $notlar[$i]," ";
?>
```

while Döngüsü

Belirtilen koşul sağlandıkça dönen döngülerdir. Bu döngüye girmek için de koşulun sağlanmış olması gerekmektedir. Kullanımı şu şekildedir.

```
while(/*koşul*/){
   // koşul sağlanırsa çalışacak kodlar
}
```

İlk örneğimizde ekrana 1'den 10'a kadar olan sayıları alt alta yazdıralım.

```php
<?php
$sayi=1;
while($sayi<=10){
   echo "Sayı $sayi<br>";
   $sayi++;
```

```
}
?>
```

while için söylenen koşul **$sayi** değişkeninin 10'a eşit yada 10'dan küçük olma koşuludur. Dolayısıyla döngüye girebilmek için bu koşula uygun bir değeri başlangıçta $sayi=1; diyerek tanımlıyoruz. Daha sonra yapmasını istediğimiz kodu yazıp **$sayi** değişkenini 1 arttırdık. Döngü koşulu **$sayi** değişkenine bağlı çalışmaktadır. Eğer **$sayi** değişkenini 1 arttırmazsak değeri hep 1 olarak kalır ki bu döngünün sonsuz döngüye yani kısır döngüye girmesine sebep olur. Döngü her tekrarında koşulu kontrol eder. Koşul sağlanmadığı anda döngüden çıkar.

Örnek: 1 ile 100 arasında rastgele üretilen sayı 45 olana kadar sayı üreten ve kaçıncı denemede bulunduğunu ekrana yazan php kodunu yazalım.

```
<?php
$sayi=1;
$sayac=0;
while($sayi!=45){
   $sayi=rand(1,100);
   $sayac++;
}
echo "$sayi sayısı $sayac denemede bulundu.";
?>
```

Buradaki koşulumuz $sayi değişkeninin 45'e eşit olmama koşuludur. Yani $sayi değişkeni 45 olmadığı sürece sayı üretmeye devam et diyoruz. Üretilen her sayıdan sonra sayacı 1 arttırarak deneme sayısını sayıyoruz. Başlangıçta $sayi değişkenine 45'den farklı bir sayı veriyoruz ki döngüye girebilelim.

Not: rand() fonksiyonu verilen aralıkta rastgele bir sayı üretir. rand(min,max)

Örnek: 1 ile 100 arasında rastgele üretilen sayıların toplamı 500'den büyük olduğunda duran döngüyü, toplam ile toplanan sayı sayısını ekrana yazan php kodunu yazalım.

```
<?php
$sayi=rand(1,100);
$sayac=1;
$toplam=$sayi;
while($toplam<=500){
   $sayi=rand(1,100);
   $sayac++;
   $toplam+=$sayi;
}
echo "Toplam=$toplam Toplanan sayı sayısı=$sayac";
?>
```

Burada $sayi değişeninin ilk değeri rastgele üretilen bir sayı olmalıdır. Dolayısıyla ilk sayı üretildiğinden $sayac değişkeninin ilk değeri de 1, $toplam değişkeni başka sayı olmadığından ilk sayıya eşit olmalıdır. Sonra toplam 500'den küçük olduğu sürece sayı üretmeye devam ediyoruz. Toplam 500'ü geçtiğinde döngüden çıkıp sonucu ekrana yazdırıyoruz.

Örnek: Ekrana 1'den 10'a kadar olan sayıların toplamını 1+2+3+4+5+6+7+8+9+10=55 şeklinde yazdıralım.

```php
<?php
$sayi=1;
$toplam=$sayi;
echo $sayi;
while($sayi<10){
   $sayi++;
   $toplam+=$sayi;
   echo "+$sayi";
}
echo "=$toplam";
?>
```

Örnek: Tanımlı olan dizinin tüm değerlerini ekrana yazdıralım.

```php
<?php
$kadro=array("elif","kaan","gözde","emine","hüseyin","emre");
$indis=0;
while(@$kadro[$indis]){
   echo $kadro[$indis]," ";
   $indis++;
}
?>
```

Buradaki koşul; dizide, belirtilen indise karşılık gelen bir değer varsa anlamına gelmektedir. Yani belirtilen indise karşılık dizide bir değer olduğu sürece bunları ekrana yaz diyoruz.

Örnek: 1 ile 100 arasında rastgele üretilen 10 tek sayıyı bir diziye atayıp ekrana yazdıralım.

```php
<?php
$sayac=1;
while($sayac<=10){
   $sayi=rand(1,100);
   if ($sayi%2==1){ // üretilen sayı tek ise
      $sayilar[]=$sayi;
      $sayac++;
   }
}

$indis=0;
while(@$sayilar[$indis]){
   echo $sayilar[$indis]," ";
   $indis++;
}
?>
```

Örnek: 1 ile 100 arasında rastgele üretilen sayı bir önceki sayıya eşit olduğunda duran sonsuz döngüyü yazalım.

```php
<?php
$onceki=rand(1,100);
echo $onceki," ";
while(true){ // while, koşul true olduğu sürece döner
   $yeni=rand(1,100);
```

```
    echo $yeni," ";
    if ($onceki==$yeni)
       break;
    $onceki=$yeni;
}
?>
```

while, yapısı itibariyle koşul sağlandığı sürece yani true olduğu sürece döner. while(true) diyerek döngünün devamlı dönmesi sağlanıyor. Yani sonsuz dön anlamına gelir. Bu tür döngülerden çıkmak için break komutu kullanılır. Aynı zamanda sıfırdan farklı her sayı true kabul edildiğinde ifade; while(1), while(5), while(-6), while("ufuk") şeklinde de yazılarak sonsuz döngüler kurulabilir.

Örnek: 1 ile 10 arasında birbirinden farklı üretilen 5 sayıyı bir diziye atayalım.

```
<?php
$sayi=rand(1,10);
$sayac=1;
$sayilar[]=$sayi;
while($sayac<5){
   $sayi=rand(1,10);

   $varmi=false; // her dizi elemanı üretilen sayı ile karşılaştırılacak
   $indis=0;
   while(@$sayilar[$indis]){
      if ($sayilar[$indis]==$sayi) //eğer üretilen sayı dizide varsa
         $varmi=true; // dizide var olarak işaretle
      $indis++;
   }

   if ($varmi==false){ // eğer üretilen sayı dizide yoksa
      $sayilar[]=$sayi; // diziye ekle
      $sayac++;
   }
}

$indis=0;
while(@$sayilar[$indis]){
   echo $sayilar[$indis]," ";
   $indis++;
}
?>
```

do-while Döngüsü

Belirtilen koşul sağlandığı sürece dönen döngülerdir. Bu döngü en az bir kere çalışır sonra koşula bakar. Koşul sağlandığı sürece de çalışmaya devam eder. While döngüsünde ise döngüye girebilmek için de koşulun sağlanmış olması gerekmekteydi. while ile do-while döngüsü arasındaki tek fark budur. Kullanımı şöyledir.

```
do{
// döngü içinde çalışacak kodlar
}while(/*koşul*/);
```

Burada, program akışına göre döngüye doğrudan girilir. Yani döngü her halükarda bir kere çalışır. Sonra koşula bakılır ve koşula göre döngü çalışmaya devam eder yada etmez.

Örnek: üretilen sayı 45 olana kadar sayı üretelim ve ekrana yazdıralım.

```
<?php
do{
   $sayi=rand(1,100);
}while($sayi!=45);

echo "Bulunan sayı: $sayi";
?>
```

Örnek: Rastgele üretilen 10 sayının toplamını ekranda 15+41+10+6+4+.....+54=541 şeklinde gösterelim.

```
<?php
$toplam=0;
$sayac=0;
do{
   $sayi=rand(1,100);
   $toplam+=$sayi;
   $sayac++;
   echo ($sayac==1)?"$sayi":"+$sayi";
}while($sayac<10);

echo "=$toplam";
?>
```

Örnek: Tanımlı olan metni küçükten büyüğe doğru yazdıralım.

```
<?php
$metin="Bilenler yapar bilmeyenler öğretir.";
$boyut=1;
do{
   echo "<p style='font-size:$boyut;'>$metin</p>";
   $boyut++;
}while($boyut<50);
?>
```

Örnek: En az 1 satırlı olmak şartıyla tanımlı olan sayı kadar satıra sahip 3 sütunlu bir tablo oluşturalım.

```
<?php
$satir=5; // $satir=0 olsa dahi tabloya en az 1 satır eklenecektir.
$sayac=0;
echo "<table border=\"1\">";
do{
    echo "<tr>";
    echo "<td width=\"50\"> </td>";
    echo "<td width=\"50\"> </td>";
    echo "<td width=\"50\"> </td>";
```

```
    echo "</tr>";
    $sayac++;
}while($sayac<$satir);
echo "</table>";
?>
```

foreach Döngüsü

Bu döngü dizilerde kullanılır. Dizinin elaman sayısını bilmeden değerlerini ve indislerini almak için kullanılır. Kullanımı şu şekildedir.

```
foreach ($dizi as $indis=>$deger){
   // çalışmasını istediğiniz kodlar
}
```

Bu yapıda indis kısmı isteğe bağlı olarak kullanılır.

Örnek olarak tanımlı olan dizinin sadece değerlerini ekrana yazdıralım.

```
<?php
$gunler=array("Pazartesi","Salı","Çarşamba","Perşembe","Cuma","Cumartesi","
Pazar");
foreach($gunler as $deger)
   echo $deger,"<br>";
?>
```

Şimdi aynı dizinin değerlerini indisleriyle beraber yazdıralım.

```
<?php
$gunler=array("Pazartesi","Salı","Çarşamba","Perşembe","Cuma","Cumartesi","
Pazar");
foreach($gunler as $indis=>$deger)
   echo $indis,"=>",$deger,"<br>";
?>
```

Örnek: Aşağıdaki dizinin değerlerini indisleriyle beraber ekrana yazalım.

```
<?php
$personel=array("Müdür"=>"Nihat","Yardımcı"=>"Uğur","Memur"=>"Haşim","Hizme
tli"=>"Ufuk");
foreach($personel as $gorev=>$isim)
   echo "<strong>$gorev</strong>: $isim<br>";
?>
```

Örnek: Tanımlı olan dizideki sayıların toplamını bulalım.

```
<?php
$sayilar=array(4,5,8,9,9,1,6,7,1,7,4,2,2);
$toplam=0;
```

```
foreach($sayilar as $sayi)
   $toplam+=$sayi;
echo "Toplam=$toplam";
?>
```

Örnek: 1 ile 100 arasında rastgele üretilen sayının dizide olup olmadığını bulalım.

```
<?php
$sayi=rand(1,100);
$sayilar=array(42,25,68,29,79,36,86,67,61,37,74,82,42);
$varmi=false;
foreach($sayilar as $deger){
   if ($sayi==$deger)
      $varmi=true;
}

if($varmi==false)
   echo "$sayi sayısı dizide yok (-)";
else
   echo "$sayi sayısı dizide var (+)";
?>
```

Örnek: Tanımlı olan öğrencinin dizide olup olmadığını bulalım.

```
<?php
$ogrenci="Esra";
$liste=array("Pınar","Esra","Zekiye","Ayşe","Burcu","Meral","Özge");
$varmi=false;
foreach($liste as $kisi){
   if ($ogrenci==$kisi){
      $varmi=true;
      echo "<strong><u>$kisi</u></strong> ";
      continue;
   }
   echo "$kisi ";
}

echo "<br>";
if($varmi==false)
   echo "$ogrenci listede yok (-)";
else
   echo "$ogrenci listede var (+)";
?>
```

Buradaki continue; ifadesi öğrenci bulunduğunda ekrana kalın ve altılı çizili olarak yazıldıktan sonra, echo "$kisi "; komutuyla tekrar ekrana yazılmaması için bir sonraki döngüye doğrudan geçmek için kullanıldı. Döngü içinde continue; ifadesinden sonraki komutlar çalışmaz ve doğrudan bir sonraki döngüye gidilir.