EĞİTİM İÇERİĞİ

İçindekiler

HTML 5
İnternet 5
IP (Internet Protocol) Numarası 5
Alan Adı (Domain) 5
Hosting (Barındırma) 5
Htmle'e Giriş 5
Basit Html Etiketleri 6
Başlıklar 6
H1 ile yapılan başlık 7
H2 ile yapılan başlık 7
H3 ile yapılan başlık 7
Paragraflar, Satır boşlukları ve Yatay Çizgiler 7
Metinler 7
Renkler 8
Linkler 9
Resim Ekleme 10
Listeler 10
i. Sıralı Liste(Ordered List) 11
ii. Sırasız Liste(Unordered List) 12
iii. Tanımlama Listeleri (Definition List) 12
Tablolar 13
Formlar 14
Listeleme 15
CSS 17
CSS'e Giriş 17
CSS Kuralları 17
Biçimlendirme Sınıfları ve Kimlikler 18
Sınıflar 18
Kimlikler 18
Pseudo Sınıfları 18
Arkaplan Özellikleri 19

Metin Özellikleri 20
Font Özellikleri 21
Kenarlık Özellikleri 22
Margin ve Padding Özellikleri 23
Liste Özellikleri 23
PHP 25
PHP Nedir? 25
PHP Çalışma Prensibi 25
PHP ile neler yapılabilir? 26
İlk PHP Sayfası 26
PHP'de Açıklamalar 27
Değişkenler 28
Operatörler 33
Aritmetik İşlem Operatörleri 33
Atama Operatörleri 34
Karşılaştırma Operatörleri 36
Mantıksal Operatörler 36
Değişken Tipleri 37
Tamsayı Değişkenler (Integer) 37
Ondalıklı Sayı Değişkenleri (float) 37
Metin Değişkenleri (String) 37
Boolean Değişkenler 39
Sabitler Değişkenler 39
Dizi Değişkenler 40
Çok Boyutlu Diziler 44
if...else Kontrol Yapısı 49
switch case Kontrol Yapısı 53
for Döngüsü 58
for Döngüsüyle İlgili Örnekler 61
while Döngüsü 66
do-while Döngüsü 70
foreach Döngüsü 71
Fonksiyonlar 73
Fonksiyon Tanımlama 74
Fonksiyon Örnekleri 76

Varsayılan Değer Alan Fonksiyonlar 79
Global Değişkenli Fonksiyonlar 82
Statik (static) Değişkenli Fonksiyonlar 85
PHP'de Kullanılan Hazır Fonksiyonlar 87
Dizi Fonksiyonları 87
1) count($dizi) 88
2) in_array($deger, $dizi) 88
3) array_search($deger, $dizi) 89
4) sort($dizi) 89
5) rsort($dizi) 90
6) asort($dizi) 91
7) arsort($dizi) 91
8) array_merge($dizi1,$dizi2,$dizi3,...) 92
9) array_keys($dizi) 93
10) array_push($dizi,$deger1,$deger2,$deger3...) 93
Matematiksel Fonksiyonlar 93
1) pow($x,$y) 94
2) sqrt($x) 94
3) abs($x) 95
4) base_convert($x,$taban1,$taban2) 95
5) fmod($x,$y) 95
6) round($x,$ondalik) 96
7) floor($x) 96
8) ceil($x) 97
9) rand(min,mak) 97
10) deg2rad($x) 98
11) sin($x), cos($x), tan($x) 98
12) pi() 98
13) exp($x) 99
String Fonksiyonları 99
1) strlen($metin) 99
2) chr($sayi) 100
3) explode($ayirici,$metin) 100
4) implode($ayirici,$dizi) 101
5) str_split($metin,$sayi) 102
6) ltrim($metin), rtrim($metin), trim($metin) 102

7) substr($metin,$baslangic,$uzunluk) 103
8) strtolower($metin), strtoupper($metin) 104
9) ucfirst($metin), ucwords($metin) 104
10) str_replace($kaynak,$hedef,$metin) 104
11) nl2br($metin) 106
12) md5($metin), sha1($metin) 106
PHP SMTP Sınıfı: PHPMailer 106
Mail Göndermek 106
Mail ile dosya göndermek 107
Birden fazla kişiye gönderme 107
Yanıt adresini değiştirme 108
GET Metodu 108
POST Metodu 113
Dosya Yükleme (File Upload) 122
PDO (*PHP Data Objects*) Nedir? 129
Veritabanı Bağlantısı Nasıl Yapılır? 129
exec() Metodu 129
query() Metodu 130
Prepare() Metodu 130
bindParam() Metodu 131
execute() Metodu 131
bindColumn() Metodu 132
Select (Seçme) İşlemi 132
Select (Seçme) İşlemi Çoklu 133
Insert (Ekleme) İşlemi 133
Delete (Silme) İşlemi 133
Update (Güncelleme) İşlemi 133
errorInfo() ile Hataları Yazdırma 134
Veritabanı bağlantısı sonlandırma 134
Güvenlik 134
Kaynakça 134

HTML

İnternet

İnternet, dünya üzerindeki birçok bilgisayar sistemini TCP/IP (Transmission Control Protocol/Internet Protocol) protokolü ile birbirine bağlayan ve her geçen gün daha fazla büyüyen bir iletişim ağıdır.

IP (Internet Protocol) Numarası

IP (Internet Protocol), bilgisayarların birbirleri ile iletişim kurmalarını sağlayan standart bir protokoldür. İnternet Protokol adresi, TCP/IP standartını kullanan bir ağda bulunan cihazların birbirleriyle iletişim kurmak ve veri alışverişinde bulunmak için kullandıkları benzeri olmayan bir numaradır. IP numarası, iletilen bilginin doğru adrese gönderilmesini veya doğru adresten alınmasını sağlar. İki bilgisayar iletişim kurdukları zaman birbirlerini bulmak için IP adresini kullanırlar.

Alan Adı (Domain)

Domain; IP adresi diye tabir edilen, bilgisayarların birbirini görmesini sağlayan nümerik sisteminin daha kolaylaştırılmış ve rahatça girilebilmesi için kelimelerle ifade edilen halidir.

Hosting (Barındırma)

Hosting, hazırlanan web sitelerinin internet ortamında yayımlanmasını sağlayan hizmet türüdür. Bu hizmet, hosting firmaları tarafından belirli süreliğine sağlanır. Server (sunucu) bilgisayarlar, hosting amacıyla kullanılan gelişmiş donanımsal özelliklere sahip olan ve birçok kullanıcıya aynı anda hizmet veren bilgisayarlardır.

Htmle'e Giriş

HTML, HyperText Markup Language, **tarayıcı (browser)**'lardan görebileceğimiz (Internet Explorer, Netscape gibi) internet dokümanlarını yaratmaya yarayan bir işaretleme dilidir. İnternet üzerindeki tüm sayfaların kaynağı HTML'dir. Tarayıcı olmadan HTML kodları birşey ifade etmez.

HTML dokümanları herhangi bir yazı editörü ile düzenlenip *.htm, *.HTML, *.sHTML gibi uzantılarla kaydedilir. Bunun için **notepad, pico, wordpad** gibi editörler yeterlidir. Bunların yanında Frontpage, Dreamweaver, Homesite gibi bu iş için hazırlanmış ve kodlamayı kolaylaştıran programlar da vardır.

Her HTML dokümanı **<HTML>** ile başlar ve **</HTML>** ile biter. <> şeklinde görülen komutlara **etiket (tag)** adı verilir. Bir HTML dokümanı iki ana kısımdan oluşmaktadır:<head> . . . </head> etiketi arasında yer alan **Başlık** bölümü; <body> . . .

</body> etiketleri arasında yer alan **Gövde** Bölümü. **Başlık** bölümü içine **<title>. . . </title>** etiketi konulur. **title,** tarayıcımızın en üstünde gözükür ve oluşturulan sayfanın başlığıdır.

Örnek:

```
<html>
<head>
<title>Title</title>
</head>
<body>
Merhaba Dünyali!
.....
.....
</body>
</html>
```

Eğer hazırlanan sayfa Türkçe içerikli ise Türkçe karakterlerin (ç, ğ, ı, , ş, ö, ü) düzgün görünmesi için **Başlık** kısmı içerisine aşağıdaki meta etiketi eklenir.

<meta http-equiv="Content-Type" content="text/HTML; charset=iso-8859-9">

title etiketi ile başlığı belirttikten sonra **Başlık** bölümü kapatılır ve ana kısım olan **Gövde'** ye geçilir. Sayfamızda görüntülemek istediğimiz bütün bilgileri **Gövde** kısmına koyarız.

Herhangi bir web sayfasının HTML kodunu görmek istediğimiz zaman, farenin sağ tuşuna tıklayıp **view source/kaynağı görüntüle** seçeneğini seçeriz.

Bir etiketin nasıl davranması gerektiği hakkında bilgi içeren etiketin içindeki metne **Değişken** denir. Aşağıdaki örnekte body etiketinin içindeki **bgcolor** ifadesi değişkendir ve arkaplan rengini belirler. Çoğu etiket isteğe bağlı olarak ya da gerektiği için bağımsız değişkenleri kabul eder.

<body bgcolor="yellow">

Çift taraflı etiketler içiçe geçebilirler ama dikkat edilmesi gereken nokta en son açılan etiketin en önce kapatılmasıdır. Aşağıda yanlış ve doğru uyguluma örneği verilmiştir.

Bu bir yanlış örnektir.
Doğrusu budur.

 ve <hr> gibi tek taraflı etiketler ,
 ve <hr /> şeklinde yazılmalıdır.

Basit Html Etiketleri

Başlıklar:

6 çeşit başlık vardır:<h1>, <h2>, . . . <h6>. Bunlardan en büyüğü <h1>, en küçüğü ise <h6>'dır.

HTML etiketi	Etiketin Web Sayfasındaki görüntüsü
H1 ile yapılan başlık H2 ile yapılan başlık H3 ile yapılan başlık H4 ile yapılan başlık H5 ile yapılan başlık H6 ile yapılan başlık	H1 ile yapılan başlık **H2 ile yapılan başlık** H3 ile yapılan başlık *H4 ile yapılan başlık* H5 ile yapılan başlık H6 ile yapılan başlık

Paragraflar, Satır boşlukları ve Yatay Çizgiler

Paragraflar etiketiyle oluşturulur. Paragraf metni ile arasında yer alır. Örneğin:

Bu bir paragraftır.

Paragraf başlatmadan bir satır boşluğu yaratmak için
 etiketi kullanılır. Örneğin:

Paragrafın ilk satırı.
Paragrafın ikinci satırı.

Sayfaya (aşağıdaki gibi) yatay çizgi eklemek için <hr> etiketini kullanırız.

HTML etiketi	Etiketin Web Sayfasındaki görüntüsü
Birinci paragraf İkinci paragraf Üçüncü paragraf Alt satır <hr />	Birinci paragraf Ikinci paragraf Üçüncü paragraf Alt Satır ---

Metinler

. .

Color yazı karakterinin rengini, **face** yazi tipini (arial, verdana, tahoma gibi), **size** da boyutunu belirler. **Size** değişkeninde kullanılan rakam 1'den 7'ye kadardır.

 etiketi HTML 4. 0 'da desteklenmemektedir ama yine de taracıyıcılar etiketi ile yapılmış biçimlendirmeyi destekler. Hala pek çok web sayfasında etiketi

kullanılmaktadır. Başlangıç seviyesi için etiketi kullanmak yeterliyken ilerki seviyeler için stilleri kullanmanız tavsiye edilir. Font etiketi ile birlikte aşağıdaki tabloda verilen etiketleri kullanarak sayfanızdaki metinleri biçimlendirebilirsiniz.

Etiket	Açıklama	Uygulama
. .	Kalın biçimlendirme	**Metin**
. .	İtalik biçimlendirme	*Metin*
<u>. . </u>	Altı çizgili biçimlendirme	Metin
^{. .}	Üst simge	X^2
_{. .}	Alt simge	H_2
<code>. . </code>	Bilgisayar kodu biçimi	Metin
<blockquote>. . </blockquote>	Alinti biçimi	Metin

Not: Metin biçimlerken başladığınız etiketi kapatmayı unutmayın.

Renkler

HTML dokümanlarında renkler ya İngilizce isimleriyle, ya da 16'lık sayı düzenindeki "hexadecimal" değerleriyle belirtilir.

En sık kullanılan ve hemen hemen bütün tarayıcıların desteklediği 16 renk aşağıdakilerdir:

Renk	Renk adi:	Renk	Renk adi:
	aqua		black
	blue		fuchsia
	gray		green
	lime		maroon
	navy		olive
	purple		red
	silver		teal
	white		yellow

Örnek :

<body bgcolor="green">

Bu örnekte sayfanın arkaplan rengi yeşil olur.

<body bgcolor="#008000"> aynı sonucu verir.

Örnek :

 Font rengi "indigo" oldu. Etiketi kapattığımız için tekrar normale döndü.

Linkler

Linkler **<a>. . . ** etiketi içinde, **href=""** komutuyla belirtilir.

Örnek:

ODTÜ Ana Sayfası

ODTÜ Ana Sayfası

Bu örnekte bir de target değişkeni verilmiştir. Adresi verilen web sayfasının başka bir pencerede açılmasını isterseniz **target="_blank"** komutunu eklemeniz gerekir. Aynı pencerede açılmasını isterseniz ilgili değişkeni target="_top" şeklinde yazabilirsiniz. Hiç bir şey yazılmazsa da sayfa aynı pencerede açılır.

title değişkenini kullanarak da fare linkin üzerine geldiğinde gözükecek link açıklamasını ekleyebilirsiniz. Aşağıdaki örnekte fareyle linkin üzerine gelip beklerseniz Orta Doğu Teknik Üniversitesi yazısını göreceksiniz.

Mustafa Dalcı

Mustafa Dalcı

Eğer hazırladığınız dokümanlar arasında bir bağlantı kurmak istiyorsanıiz, <a> etiketini aşağıdaki gibi kullanmalısınız.

Önceki sayfa

Bir e-posta adresine link vermek istiyorsanız:

E-posta atmak için tıklayın.

E-posta atmak için tıklayın.

Bu linke tıklandığında bilgisayarda kurulu varsayılan e-posta okuma programı açılır ve gönderilecek adres bölümünde etiket içinde belirtilen adres yazar.

Aynı doküman içinde bağlantı kurmak istersek tutturucu noktaları **(anchor points)** kullanırız. Tutturucu HTML sayfasında bir yer işaretidir. İsmi ile bir bölgeyi belirler ve bu tutturucuya bir link verebilirsiniz. Tutturucuların kullanımı aşağıdaki gibidir.

Linkler başlığına aşağıdaki gibi bir tutturucu belirleriz.

```
<a name="link">Linkler</a>
```

Sayfanın herhangi bir bölümünde yukarıda belirlediğimiz tutturucuya aşağıdaki gibi link veririz

```
<a href="#link">Linklere git</a>
```

Resim Ekleme

Resim ekleme:

```
<img src="resim.jpg" />
```

şeklinde olur. Dikkat etmemiz gereken kulanacağımız resmin dizin yapısıdır.

Bağımsız Değişkenler :

```
<img src="" width="" height="" border="" alt="" />
```

alt: Resme açıklama vermemizi sağlar. Fareyi resmin üstüne getirdiğimizde, **alt** değişkeninde yazılan açıklama ekranda çıkar. Eğer resim açılmazsa, onun yerine açıklama görünür.

src : Resim dosyasının kaynağını belirtir.

** Eğer web sayfasının arka planında bir imajın çıkması istenirse:

```
<body background="resim.jpg">
```
şeklinde yazılır.

Resimleri linke dönüştürmek için <a> etiketi ile <img> etiketini içiçe kullanırız.

```
<a href="http://www.metu.edu.tr" target="_blank"><img src="odtu.jpg" /></a>
```

Not: Resimlerin width ve height değişkenlerini belirtmeniz sayfanızın yüklenirken şeklinin bozulmasını önler.

Listeler

Üç çeşit liste vardır:

i. Sıralı Liste(Ordered List):

Örnek:

HTML etiketi	Etiketin Web Sayfasındaki görüntüsü
<ol type="1">Çerezler: Kavurga Çekirdek Mısır Cips <ol type="a">Içkiler: Bira Votka Şarap Viski Rakı 	Çerezler: 1. Kavurga 2. Çekirdek 3. Mısır 4. Cips Içkiler: a. Bira b. Votka c. Şarap d. Viski e. Rakı

Örnekte görüldüğü gibi **type** değişkeni sıralı listenin türünü belirler. Type değişkeni için aşağıdaki değerlerden biri kullanılabilir:

type: {1, a, A, I, i}

Listeyi start değişkeni ile istediğimiz sayıdan baslatabiliriz:

Örnek:

HTML etiketi	Etiketin Web Sayfasındaki görüntüsü
<ol start="199">Çerezler: Kavurga Çekirdek Mısır Cips 	Çerezler: 199. Kavurga 200. Çekirdek 201. Mısır 202. Cips

ii. Sırasız Liste(Unordered List):

Örnek:

HTML etiketi	Etiketin Web Sayfasındaki görüntüsü
<ul type="circle">Çerezler: - Kavurga - Çekirdek - Mısır - Cips <ul type="disc">Içkiler: - Bira - Votka - Şarap - Viski - Rakı 	Çerezler: ○ Kavurga ○ Çekirdek ○ Mısır ○ Cips Içkiler: • Bira • Votka • Şarap • Viski • Rakı

Benzer şekilde burada da **type** değişkeni sırasız listenin işaretini belirler. Type değişkeni için aşağıdaki değerlerden biri kullanılabilir:

type:{square, disc, circle}

iii. Tanımlama Listeleri (Definition List):

Örnek:

HTML etiketi	Etiketin Web Sayfasındaki görüntüsü
<dl> <dt>Karbonhidrat ve ben <dd>En çok bol karbonhidratlı yemekleri severim, özellikle de makarna ve türevlerini. Lazanya favorimdir. Pizza ve mantıya da bayılırım. </dd></dt> <dt>Sebze ve ben <dd>Sebzeyle aram pek iyi değildir ama taze fasulye oldukça lezziz bir yiyecektir. Onun dışında dolma, sarma da güzeldir. </dd></dt>	Karbonhidrat ve ben En çok bol karbonhidratlı yemekleri severim, özellikle de makarna ve türevlerini. Lazanya favorimdir. Pizza ve mantıya da bayılırım. Sebze ve ben

<dt>Et ve ben <dd>Et seven bir insanimdir. Her çesit kebabi afiyetle yerim. Kırmızı olsun, beyaz olsun, hemen hemen bütün etleri yerim. Balık seçerim ama. </dd></dt> </dl>	Sebzeyle aram pek iyi değildir ama taze fasulye oldukça lezziz bir yiyecektir. Onun dışında dolma, sarma da güzeldir. Et ve ben Et seven bir insanımdır. Her çesit kebabı afiyetle yerim. Kırmızı olsun, beyaz olsun, hemen hemen bütün etleri yerim. Balık seçerim ama.

Tablolar

Tablolar *<table>. . . . </table>* etiketleri arasında oluşturulur. *<table>* etiketinden sonra daima *<tr>* gelir. Her satır tanımlandığında *<tr>*, her hücre tanımlandığında da *<td>* etiketi kullanılır.

Örnek :

```
<table border="1">
<tr><td>1. hücre</td><td>2. hücre</td><td>3. hücre</td></tr>
<tr><td>4. hücre</td><td>5. hücre</td><td>6. hücre</td></tr>
</table>
```

1. hücre	2. hücre	3. hücre
4. hücre	5. hücre	6. hücre

Bağımsız Değişkenler:

```
<table border="" cellpadding="" cellspacing="" width="" height="" bgcolor="" align="" valign="">
<td height="" width="" bgcolor="" align="" valign="" colspan="" rowspan="">
```

border : Çerçevenin kalınlığını belirler. *border="0"* dersek tabloda çerçeve bulunmaz, bu miktarı arttırdıkça çerçevenin kalınlığı da artar.

cellpadding , cellspacing : Hücre elemanlarının sınırlara olan uzaklığı **cellpadding**, satır ve sütunların uzaklığı ise **cellspacing** değişkeni ile belirtilir.

bgcolor: *<table bgcolor="red">* şeklinde kullanarak bütün tablo ya da *<td bgcolor= "red">* şeklinde sadece tek bir hücre renklendirilir.

align : Hücredeki elemanın yatay konumunu belirler ve "right, left, center" opsiyonları ile kullanılır.

valign : Hücre elemanının düşey konumunu belirler ve opsiyonları "top, bottom, middle"dır.

colspan , rowspan : Aynı satırdaki hücreleri birleştirmek için **colspan**, aynı sütundaki hücreleri birleştirmek için de **rowspan** değişkeni kullanılır. Birleştirilen hücreye ait *<td>. . </td>* etiketi silinir.

Örnek :

```
<table border="1">
<tr><td>1. hücre</td><td>2. hücre</td><td>3. hücre</td></tr>
<tr><td rowspan="2">4. hücre</td><td colspan="2">5. hücre</td></tr>
<tr><td>6. hücre</td><td>7. hücre</td></tr>
</table>
```

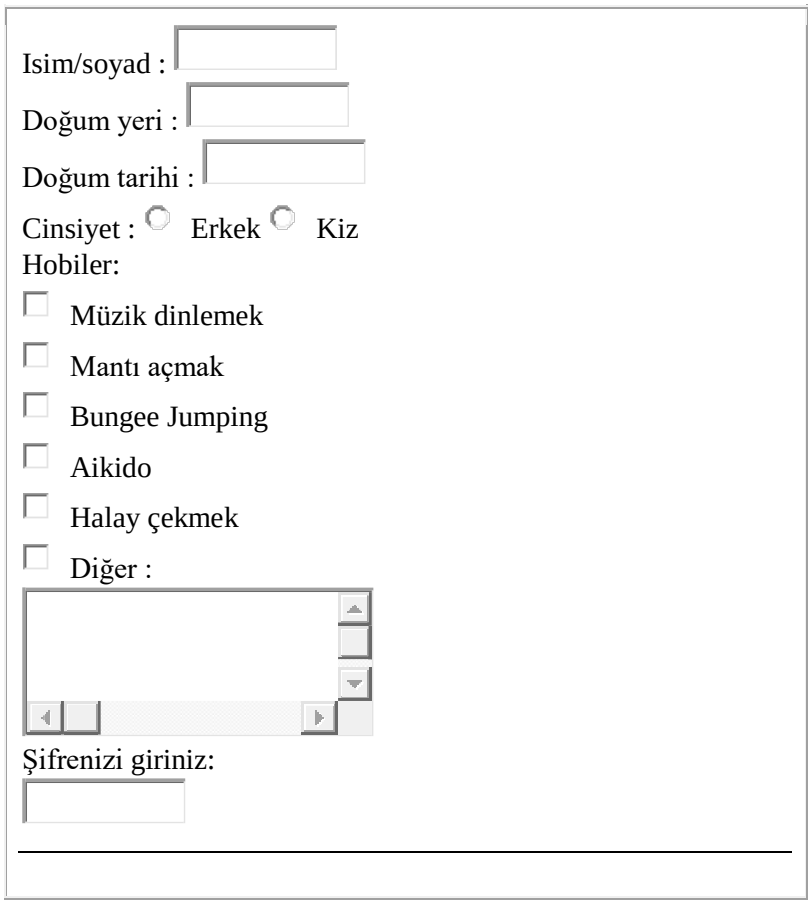
<table>
<tr><td>1. hücre</td><td>2. hücre</td><td>3. hücre</td></tr>
<tr><td rowspan="2">4. hücre</td><td colspan="2">5. hücre</td></tr>
<tr><td>6. hücre</td><td>7. hücre</td></tr>
</table>

Formlar

Formlar **<form>. . . </form>** etiketleri arasında oluşturulur. Form bilgilerini **action** değişkeninin içine yazdığınız dosyaya veri olarak gönderebilirsiniz. Formlar sayesinde anket ve geribildirim uygulamaları hazırlayabilirsiniz.

Örnek:

```
<form name="kimlik" action="gonder.php" method="get">
Isim/soyad : <input type="text" size="20" name="isim"><br>
Doğum yeri : <input type="text" size="20" name="dogumyer" ><br>
Doğum tarihi : <input type="text" size="10" name="dogumtarih" ><br>
Cinsiyet : <input type="radio" name="cins" value="erkek" > Erkek <input type="radio" name="cins"
value="kiz"> Kiz<br>
Hobiler:<br>
<input type="checkbox" name="muzik" >Müzik dinlemek<br>
<input type="checkbox" name="manti" >Manti açmak<br>
<input type="checkbox" name="bungee" > Bungee Jumping<br>
<input type="checkbox" name="aikido">Aikido<br>
<input type="checkbox" name="halay">Halay çekmek<br>
<input type="checkbox" name="diger">Diğer :<br>
<textarea rows="4" cols="30" name="diger"></textarea><br>
Şifrenizi giriniz:<br>
<input type="Password" size="15"><br>
<input type="submit" value="GÖNDER"> <input type="reset" value="SIL">
</form>
```



action: Formun gönderileceği adresi belirtir.
method="get": Formdaki bilgiler başka bir dosyaya gönderilecekse kullanılır.
method="post": Formdaki bilgiler bir adrese postalanacaksa kullanılır.
type="text" : Tek satırlık bir metin alanı açar.
size="" : Bu metin alanının boyutunu belirler.
type="checkbox" : Çok seçenekli bir sorunun birden fazla yanıtını almamızı sağlar.
type="radio" : Tek seçenekli bir sorunun tek yanıtı alınır.
type="submit" : formu *action*'la belirtilen dosyaya yönlendiren bir buton yaratır.
type="reset" : Bu butona basınca form boş hale gelir
type="password" : Bir password alanı olusturur. Buraya girilen her karakter * şeklinde görünür.
<textarea rows="" cols=""> :*type="text"* gibi tek satırlı değil de çok satırlı bir metin alanı istiyorsak bu etiketi kullaniriz. cols metin alanının uzunluğunu, rows ise yüksekliğini pixel cinsinden verir.

Listeleme:

Select ve **option** etiketlerini kullanarak seçimlik liste (menü) oluşturabiliriz. **Option** etiketi ile belirtilen her bir değer listenin bir elamanını oluşturur ve fareyle seçilen bu elemanlardan biri **select** etiketinde belirtilen değişkenin değeri haline gelir.

Örnek:

```
<select name="otolar">
<option>Alfa Romeo</option>
<option>BMW</option>
<option>Peugeot</option>
<option>Renault</option>
<option>Seat</option>
<option>Lada</option>
</select>
```

CSS

CSS'e Giriş

Basamaklı stil sayfaları (cascading style sheets, CSS) stil tabanlı biçimlendirmeleri belirlerken kullanılan koddur. Web sayfası ya da sayfa kümelerinin biçimlerini standartlaştırmaya yardımcı olur. Stil sayfasına etiketler için belirlenen stiller yazılarak HTML sayfasındaki etiketler biçimlendirilebilir.

Stil Sayfaları; Katıştırılmış (embedded) ve Harici (external) Stil sayfaları olmak üzere ikiye ayrılmıştır.

Katıştırılmış Stil Sayfaları HTML sayfasının Başlık bölümünün içine <style>..</style> arasına yazılır ve sadece o belgedeki biçimlendirmeyi sağlar. Örneğin;

```
<html>
<head>
<style>
p{
color:red
}
</style>
</head>
```

Harici Stil Sayfaları .css uzantısı ile kaydedilmiş dosyalardır ve HTML sayfasının head bölümünden aşağıdaki gibi link verilerek ulaşılır. Dosya ismini stil.css varsayalım.

```
<link rel="stylesheet" href="stil.css" type="text/css">
```

CSS Kuralları

CSS'nin 3 bölümden oluşan bir yazım mantığı vardır: Her CSS kuralı bir *selector (seçici)* ve bir *tanımlama* bölümüne sahiptir. Tanımlama bölümü bir *özellik* ve bir *değer*den meydana gelir.

selector { **özellik** : **değer**}

Örnek olarak:

```
body {color: red } veya p {font-size: 10px} div{font-color: blue}
```

Eğer bir etiket için birden fazla stil kuralı belirlemek istenirse, kurallar noktalı virgülle ayrılır:

```
body {color: red ; margin: 0px ;font-size: 10px}
```

Birkaç etikete aynı stili vermek için gruplama yapılabilir. Aşağıdaki örnekte tüm başlıklar gruplanarak kırmızı olmuştur.

```
h1,h2,h3,h4,h5,h6 {color:red }
```

Biçimlendirme Sınıfları ve Kimlikler

Sınıflar

İstenen etiketleri isteğe bağlı oluşturulan stillerle biçimlendirmek için **sınıf**lar belirlenir. Sınıflar T ürkçe karakter içermemek şartıyla istenen ismi alabilir. CSS belgesinde sınıfı tanımlamak için sınıf isminin önüne nokta konur. Örnek olarak belirlenen **sagayasla** sınıfı ile class="sagayasla" değişkenine sahip tüm etiketlerin içindeki metinler sağa yaslanmış olacaktır.

```
.sagayasla {text-align: right }
```

Aşağıda ise h4 ve p etiketlerine class="sağayasla" değişkeni eklendiğinde başlığın ve paragrafın sağa yaslanmış olduğu görülebilir.

HTML etiketi	Etiketin Web Sayfasındaki görüntüsü
<h4 class="sagayasla">Sağa yaslanmış başlık</h4> Sağa yaslanmış paragraf	*Saga yaslanmış başlık* Saga yaslanmış paragraf

Kimlikler

Kimliklerin sınıflardan farkı, sınıflar birden fazla etikete atanabilirken kimliklerin sayfada sadece bir kez kullanılabilmesidir. CSS belgesinde kimliği tanımlamak için önüne # konur. Aşağıda örnek olarak **altbolum** kimliği oluşturulmuştur.

```
#altbolum{color:blue; text-align: center }
```

Aşağıda sayfanın alt kısmı için oluşturulan bölüme id="altbolum" değişkenini eklendiğinde metnin mavi ve ortalanmış olduğu görülebilir.

HTML etiketi	Etiketin Web Sayfasındaki görüntüsü
Her hakkı saklıdır.Bilgi için xxx@xxx.com </h4>	Her hakki saklıdır.Bilgi için xxx@xxx.com

Pseudo Sınıfları

Pseudo sınıfları bazı etiketlere özel efektler eklemek istendiğinde kullanılır. En sık kullanımı, link vermek için kullanılan <a> etiketinde görülebilir. Yazım mantığı şöyledir.

selector:pseudo sınıfı { **özellik** : **değer**}

selector.sınıf:pseudo sınıfı { **özellik** : **değer**}

Linkler renklendirmek istenildiğinde, aşağıdaki stil kodlarının .css uzantılı dosyaya veya Başlık kısmındaki <style>..</style> etiketleri arasına konulması gerekir

```
a:link {color: navy}     /* ziyaret edilmemiş link rengi*/
a:visited {color: green}  /* ziyaret edilmiş link rengi*/
a:hover {color: red}   /* fare üstüne geldiğindeki renk */
a:active {color: yellow}   /* seçilmiş link */
```

Eğer linkler için bir sınıf oluşturmak istenirse (aşağıdaki örnekte **siyah** sınıfı oluşturuluyor);

```
a.siyah:link {color:black}
a.siyah:visited {color:black}
a.siyah:hover {color:gray}
a.siyah:active {color:black}
```

kodlarını kullanılır. İstenen linke bu sınıfı atamak için link etiketine(<a>) class="siyah" değişkeni eklenir.

Arkaplan Özellikleri

Özellik	Açıklama	Aldığı Değerler	Örnek Kullanım

background-color	Etiketin arkaplanına renk atar.	Renk değerleri	body{
			background-color: red
			}

background-image	Etiketin arkaplanına resim atar.	Resim dosyası adresi	table{
			background-image:
			url(resim.jpg)
			}

background-repeat	Etiketin arkaplanına eklenen resmin tekrar edip etmeyeceğinibelirler.	repeat,
		repeat-x,
		repeat-y,
		no-repeat
			body{
			background-image:
			url(resim.jpg);
			background-repeat:
			repeat
			}

background-attachment	Arkaplana eklenen resmin sayfa ile kaymasını veya sabit kalmasını sağlar.	scroll,
		fixed
			body{
			background-image:
			url(resim.jpg);
			background-
			attachment: scroll
			}

background-position	background-image ile belirlenen resmin başlangıç noktasını belirler.	top left
		top center
		top right
		center left
		center center
		center right
		bottom left
		bottom center
		bottom right
		x-% y-%
		x-poz y-poz
			body{
			background-image:
			url(resim.jpg);
			background-
			attachment: scroll
			background-image:
			top left
			}
			
			body{
			background-image:
			url(resim.jpg);
			background-

			attachment: scroll background-image: 10% 20% }

Metin Özellikleri

Özellik	Açıklama	Aldığı Değerler	Örnek Kullanım
color	Metnin rengini belirler.	Renk değerleri	p{ color: red}

direction	Metnin yönünü belirler.	ltr,(soldan sağa)
		rtl (sağdan sola)
			.sagdansola {
			direction:rtl
			}

letter-spacing	Harfler arasındaki boşluk değerini belirler.	normal *uzunluk*	p{letter-spacing: normal } p.bosluk {letter- spacing: 5 px }
text-align	Metnin etiketin içindeki hizasını belirler.	left right center justify	p.solayasla{text- align: left }
text-decoration	Bu özellik metinlere özel işaretler koymamızı sağlar.	none underline overline line-through blink	.alticiz{text- decoration: underline }
text-indent	Metni ilk satırda sola kaydırmak için kullanılır.	*uzunluk* *%*	p{text-indent: 10 px} p{text-indent: 10 %}
text-transform	Metnin büyük-küçük harf çevrimi için kullanılır.	none capitalize (ilk harfler büyük) uppercase (hepsi büyük) lowercase (hepsi küçük)	.ilkharfbuyuk { text- transform: capitalize}
word-spacing	Kelimeler arasındaki boşluk değerini belirler.	normal *uzunluk*	span{word-spacing: normal} div{word-spacing: 10px}

Font Özellikleri

Özellik	Açıklama	Aldığı Değerler	Örnek Kullanım
font-family	Metinlerin kullanılacağı font ailesini belirlemek için kullanılır.	font aile ismi (herhangi bir font ailesi ismi kullanılabilir.) soysal aile ismi	p {font-family : Verdana, Arial, Helvetica, sans-serif; }
font-size	Fontun boyutunu belirler.	xx-small x-small small medium large x-large xx-large smaller larger *uzunluk* *%*	p{font-size:xx-small} div{font-size:10 px} p.buyukfont{font-size:150%}
font-style	Fontun biçimlendirmesini belirler.	normal italic oblique	.italik{font-style:italic}
font-weight	Fontun kalınlık incelik durumunu belirler.	normal bold bolder lighter 100 200 300 400 500 600 700 800 900	.kalinyaz {font-weight: bold}

Kenarlık Özellikleri

Özellik	Açıklama	Aldığı Değerler	Örnek Kullanım
border-style	Kenarlık stilini belirlememizi sağlar	none hidden dotted (noktalı) dashed (kesik çizgili) solid (kesiksiz) double (çift çizgi) groove (yivli) ridge (çıkıntılı) inset (içe doğru) outset (dışa doğru)	table{border-style: solid}
border-top-style, border-right-style, border-bottom-style, border-left-style	border-style özelliğinin her kenara ayrı ayrı atamasını yapabilmek için kullanılır.	border-style ile aynı değerleri alır.	td {border-top-style: none}
border-width	Kenarlık kalınlığı için kullanılır.	thin medium thick *uzunluk*	td {border-width:thin}
border-top-width, border-right-width, border-bottom-width, border-left-width	border-width özelliğinin her kenara ayrı ayrı atamasını yapabilmek için kullanılır.	border-width ile aynı değerleri alır.	.ustkenar{border-top-width: 2px}
border-color	Kenarlık rengini belirlemek için kullanılır.	Renk değerleri	table.yesilrenk {border-color: #00FF66}
border-top-color, border-right-color, border-bottom-color, border-left-color	border-color özelliğinin her kenara ayrı ayrı atamasını yapabilmek için kullanılır.	Renk değerleri	.sagtaraf{border-right-color: #CCFF00}

Margin ve Padding Özellikleri

Özellik	Açıklama	Aldığı Değerler	Örnek Kullanım
margin	Etiketin etrafındaki boşluk olarak tanımlanır.	auto *uzunluk* *%*	p {margin:auto} td {margin:10px} table {margin:10%}
margin-top, margin-right, margin-bottom, margin-left	margin özelliğinin her kenara ayrı ayrı atamasını yapabilmek için kullanılır.	margin ile aynı değerleri alır.	td{margin-top:10px}
padding	Padding içerik ile kenarlık arasındaki boşluk olarak tanımlanır.	*uzunluk* *%*	table {padding:10px} td {padding:10%}
padding-top, padding-right, padding-bottom, padding-left	Arkaplana eklenen resmin sayfa ile kaymasını veya sabit kalmasını sağlar.	padding ile aynı değerleri alır.	td {padding-top:10px}

Liste Özellikleri

Özellik	Açıklama	Aldığı Değerler	Örnek Kullanım
list-style-type	Listedeki işareti belirler.	none disc circle square decimal decimal-leading-zero lower-roman upper-roman lower-alpha upper-alpha lower-greek lower-latin upper-latin hebrew armenian	ul {list-style-type:disc} ol{list-style-type:upper-roman }

		georgian cjk-ideographic hiragana katakana hiragana-iroha katakana-iroha	
list-style-image	Listede işaretin yerine resim koymak için kullanılır.	Resim dosyası adresi	ul {list-style-image: url(liste.jpg)}
list-style-position	Liste işaretinin yerini belirler.	inside outside	ul{list-style-position: nside }

PHP

PHP Nedir?

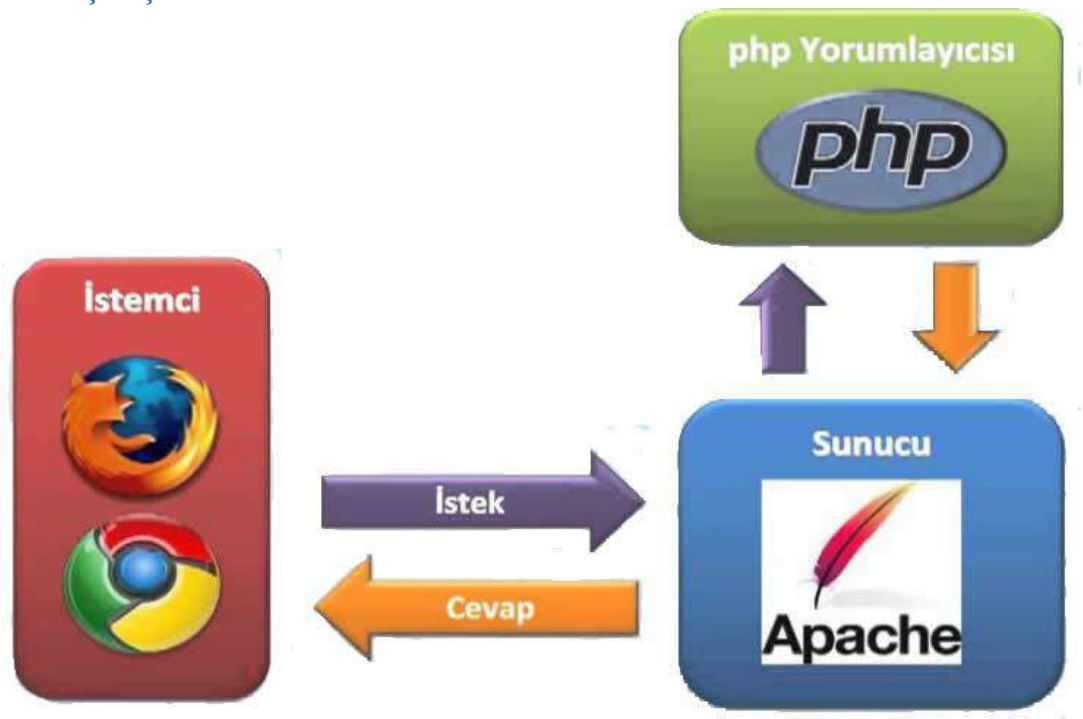
- PHP Sunucu taraflı bir betik dilidir.
- Betik dilinde olaylara cevap olarak işlem yapılır.

Server (sunucu): Bilgisayar ağlarında, bir programı veya kaynağı farklı kullanıcılara / sistemlere paylaştıran/dağıtan donanım veya yazılımlardır.

Client (istemci): Bir ağ üzerinde, sunucu bilgisayarlardan hizmet alan kullanıcı bilgisayarlarıdır.

PHP, HTML ile kullanılmak üzere tasarlanmıştır ve HTML içine gömülebilir. PHP dosyaları sunucuda .php uzantısıyla saklanır.

PHP Çalışma Prensibi



HTML sayfaları sabit iken , PHP sayfaları dinamik yapıdadır. Php sayfalarda sayfa içeriği kullanıcıya veya veri tabanında kayıtlı olan veriye bağlı olarak değişebilir.

- Php, açık kaynak ve ücretsiz bir yazılımdır.
- Farklı platformlarda geliştirilebilir,dağıtılabilir ve kullanılabilir.
- Güçlü,dayanıklı ve ölçeklenebilir bir dildir.
- Kaynak açısından zengindir.
- Php'nin resmi web sayfası www.php.net

PHP ile neler yapılabilir?

- İçerik yönetim sistemleri (**WordPress**, Drupal, …)
- Dinamik sayfalar
- Veri depolama sistemleri (Senduit, Imageshack.us, …)
- Anlık bilgi paylaşımı (**Facebook**, …)
- E-ticaret sistemleri (ShopPhp, …)
- Otomasyon yazılımları
- Resim oluşturma ve düzenleme
- Online oyunlar (**Travian**, Ogame, …)
- Forumlar, makaleler, sözlükler (**Wikipedia**, **Vbulletin**, MyBB, …)
- İçerik bulup derleyen botlar
- Web tabanlı robotlar

İlk PHP Sayfası

Php kodlamak için **Notepad++, Sublime Text** gibi editörler kullanılabilir. PHP kodlarını HTML kodlarından ayırmak için;

```
<?php
  ..........
?>
```

ayraçları kullanılır ve her PHP komutundan sonra **noktalı virgül (;)** kullanılır.

Aşağıda **index.php** olarak kaydettiğimiz sayfamızın kodlarını görüyorsunuz.

```
1  <html>
2  <body>
3  <?php
4  echo "Merhaba Dünya!";
5  ?>
6  </body>
7  </html>
```

Hazırladığımız bu sayfayı çalıştırmak için tarayıcımızın adres çubuğuna **http://localhost/index.php** yazmak yeterlidir.
Not: Eğer yazılan adreste sayfanın adı belirtilmemişse varsayılan olarak sunucu **index.php**'yi arar. Varsa çalıştırır. Burada sayfamızın adı **index.php** olduğundan adres sadece **http://localhost/** şeklinde de yazılabilirdi. Aksi halde sayfanın adının da yazılması gerekmektedir.

Çalıştırılan sayfanın ekran görüntüsüne ve kullanıcıya gönderilen kaynak koduna dikkat ediniz.

Bilgisayarımıza kurulu olan Apache, PHP ve MySQL hakkındaki tüm bilgileri ekranda görmek için **phpinfo();** fonksiyonu kullanılır. Bu fonksiyon ile versiyon bilgileri, dizin bilgileri ve diğer tüm yapılandırma bilgileri ekranda görüntülenebilir. Aşağıdaki kodu yazıp kaydederek çalıştırdığınızda sonucu sizler de ekranda görebilirsiniz.

```
1	<html>
2	<body>
3	<?php
4	phpinfo();
5	?>
6	</body>
7	</html>
```

PHP'de Açıklamalar

PHP kodu yazarken yazdığınız kodlar ile ilgili yada yaptığınız iş ile ilgili olarak yazmak istediğiniz açıklamalar veya hatırlatıcı notlar olabilir. Unutmayınız ki iyi kodlanmış bir sayfada açıklama satırlarını programcılar her zaman kullanmaktadır.

Tek bir satırdaki açıklamalar için # yada // , birden fazla satırı açıklama satırı yapmak için **/*** ve ***/** kullanılır. Unutmayın açıklamalar çalıştırılmaz, programda göz ardı edilir. Onlar sadece hatırlatıcı yada açıklayıcı notlar için kullanılır.

```php
<?php
echo "Merhaba Dünya!"; # buradan sonrası açıklama satırıdır
$tarih="01.05.2010"; // satırın sadece bu kısmı açıklama satırıdır

# bu satırın tamamı açıklama satırıdır
// bu satırın da tamamı açıklama satırıdır

/*
Bu kısımda ise birden
fazla satır
açıklama satırı olarak
tanımlanmıştır
*/

echo "<br><b>Tarih</b>: "; /* bu araya açıklama ekledik */ echo $tarih;

// echo $sonuc;  <- kodun çalışmasını istemediğimiz için açıklama satırı yaptık.
?>
```

Değişkenler

Bilindiği üzere değişkenler o an için değer saklamak için kullanılırlar. Bu değerler metinler, sayılar ya da diziler olabilir ve program akışı içerisinde istediğiniz kadar kullanıp değerini değiştirebilirsiniz.
PHP'de kullanılan değişkenlerin önünde **$** işareti kullanılır. Yeni başlayanlar için bu işareti unutmamalarını aksi halde programın istediğiniz gibi çalışmayacağını hatırlatırız.

Aşağıda tanımlanmış birkaç değişken ve ona atanmış farklı değerler görmektesiniz.

```php
<?php
$metin="Merhaba Dünya!";
$sayi=27;
?>
```

Görüldüğü üzere birkaç değişken tanımlayıp onlara istediğimiz değerleri atadık. Öncesinde bunun ne tür bir değer saklayacağını söylemedik. Çünkü PHP, değişkene atanan değere göre

türünü otomatik olarak kendisi ayarlamaktadır. Böyle bir işlemi kuralcı bir dilde yaptığınızda, kullandığınız değişkenin türünü belirtmediğinizden size hata verecektir. Bu yönüyle php bize oldukça esneklik kazandırmaktadır.

Bir değişkenin adını belirlerken aşağıdaki kurallara dikkat etmelisiniz.

- Bir değişkenin adı harfle yada _ ile başlamalıdır.
- Değişken isimlerinde boşluk bırakılmamalıdır. Boşluğun yerine _ işareti kullanabilirsiniz.
- Bir değişkenin isminde a-z, A-Z, 0-9 ve _ karakterlerinin dışında başka bir karakter kullanamazsınız.

Not: PHP dilinde büyük küçük harf ayrımı olduğunda **$adi** ile **$Adi** değişkenlerinin aynı olmadığını unutmayınız.

```
1 <?php
2 $metin="Merhaba Dünya!";
3 $Metin="Hello World!";
4 $adi_soyadi="Ali KAVAK";
5 $sayi_1=27;
6 $sayi_2=15;
7 echo $Metin; // sayfaya Hello World! Yazar.
8 ?>
```

Değişkenlerle İlgili Örnekler

Örnek 1: Tanımlı iki sayının toplamını bulup farklı şekillerde ekrana yazdıralım.

```
1 <?php
2 $sayi1=23;
3 $sayi2=42;
4 $sonuc=$sayi1+$sayi2;
5
   echo $sonuc; // ekrana 65 yazar.“
6 echo "<br>";
7 echo $sayi1+$sayi2; // ekrana 65 yazar
8 echo "<br>";
9 echo "Sonuc=",$sonuc; // ekrana Sonuc=65 yazar
10 echo "<br>";
11 echo "Sonuc=".$sonuc; // ekrana Sonuc=65 yazar
```

```
12echo "<br>";
13echo "Sonuc=$sonuc"; // ekrana Sonuc=65 yazar
14echo "<br>";
15echo "Sonuc=",$sayi1+$sayi2; // ekrana Sonuc=65 yazar
16echo "<br>";
17echo "Sonuc=".($sayi1+$sayi2); // ekrana Sonuc=65 yazar
18echo "<br>";
19echo "Sonuc=$sayi1+$sayi2"; // ekrana Sonuc=23+42 yazar. Çünkü aritmetik işlemler çift tırnak içinde yapılamaz. + operatör olarak değil, ekran yazılması gereken bir karakter olarak algılanır.
20echo "<br>";
21echo "$sayi1+$sayi2=",$sayi1+$sayi2; // ekrana 23+42=65 yazar
22?>
23
```

echo fonksiyonunda kullanılan virgül (,) ile nokta (.) arasındaki farkı tekrar hatırlatmak gerekirse:

Virgül (,) parametreleri ayırmak için kullanılır.

Nokta (.) değişkenleri birleştirip tek bir değere dönüştürmek için kullanılır.

Aritmetiksel işlemler ise çift tırnak içinde gerçekleşmez. Zira aritmetiksel operatörler (+, -, *, /) ekrana yazılması gereken karakterler olarak algılanır.

Örnek 2: Tanımlı olan ad ve soyad değerlerini ekranda yan yana gösterelim.

```
1 <?php
2 $adi="Fuat";
3 $soyadi="Ocak";
4 $adisoyadi_1=$adi." ".$soyadi; // ad , boşluk ve soyad birleştirip tek değer olarak atanıyor
5 $adisoyadi_2="$adi $soyadi"; // string içerisinde ad ve soyad kullanılıyor
6
7 // aşağıdaki kodların herbiri ekrana Fuat Ocak yazar
8 echo $adi." ".$soyadi; // echo ya tek parametre veriliyor
9 echo "<br>";
10 echo "$adi $soyadi"; // echo ya tek parametre veriliyor
10 echo "<br>";
```

```
11 echo $adisoyadi_1;
12 echo "<br>";
13 echo $adisoyadi_2;
14 echo "<br>";
15 echo $adi," ",$soyadi; // echo ya 3 parametre veriliyor
16 ?>
17
```

Örnek 3: Tanımlı olan 3 sayıyı toplam değişkenine katalım.

```
1  <?php
2  $toplam=11;
3  $sayi1=5;
4  $sayi2=7;
5  $sayi3=14;
6  $toplam+=$sayi1; // toplam değişenine sayi1 katılıyor
7  $toplam+=$sayi2; // toplam değişenine sayi2 katılıyor
8  $toplam+=$sayi3; // toplam değişenine sayi3 katılıyor
9  echo "Toplam=$toplam"; // ekrana Toplam=37 yazar
10 ?>
```

Örnek 4: Farklı türdeki tanımlı değişkenleri birleştirip sayfaya yazalım.

```
1 <?php
2 $boyut=7;
3 $renk="green";
4 $acilis="<marquee><i>";
5 $kapanis="</i></marquee>";
6 $metin="Nasipse gelir hintten yemenden nasip değilse ne gelir elden!";
7
8 $tam_metin="<font size='$boyut' color='$renk'>".$acilis.$metin.$kapanis."</font>";
```

```
9 echo $tam_metin;
10?>
```

Bu kodu çalıştırdığımızda ekranda kayan bir yazı görmekteyiz. Burada şunu görmekteyiz: echo ile sayfaya yazdırılan değerler aslında sayfanın kaynak koduna yazdırılıyor. Bunun anlamı şudur: Eğer echo ile sayfaya html kodları yazdırılırsa bunlar tarayıcı tarafından yorumlanarak ekranda gösterilir.

Dikkat edilmesi gereken bir diğer nokta ise çift tırnaklar içinde yazılan html parametrelerine atanan değerler tek tırnak içinde yazılmaktadır. Zira bu şekilde tırnakların karışması engellenmiş oluyor. Bunu sağlamanın öteki yolu ise tek tırnaklar yerine önüne \ koymak şartıyla çift tırnak kullanmaktır. Aşağıdaki koda dikkat ediniz.

```
1$tam_metin="<font size=\"$boyut\" color=\"$renk\">".$acilis.$metin.$kapanis."</font>";
```

Örnek 5: Yarıçapı tanımlanmış bir dairenin alanını ve çevresini sabit tanımlı pi değeri ile hesaplayıp ekrana yazdıralım.

```
1 <?php
2 define("pi",3);
3 $r=4;
4 $alan=pi*$r*$r;
5 $cevre=2*pi*$r;
6 echo "<strong>Alan=</strong>$alan<br>";
7 echo "<strong>Çevre=</strong>$cevre";
8 ?>
```

Dikkat edilirse sabiti kullanırken önüne $ koymadık.

Örnek 6: Tanımlı olan sayı büyüklüğünde yine tanımlı olan başlığı ekranda gösterelim.

```php
<?php
$sayi=3;
$metin="Bir musibet bin nasihattan iyidir.";
echo "<h$sayi>$metin</h$sayi>";
$sayi=5;
echo "<h$sayi>$metin</h$sayi>";
$sayi=1;
echo "<h$sayi>$metin</h$sayi>";
?>
```

Yukarıdaki kodları çalıştırdıktan sonra sayfanın kaynak koduna dikkat ediniz.

Eğer echo ile kaynak koda yazdırdığınız ifadelerin kaynak kodda alt alta görünmesini istiyorsanız yazdırılan değerin sonun \n (new line-yeni satır) ifadesini ekleyiniz. Aynı örneği aşağıdaki gibi çalıştırıp sayfa kaynağına bakınız.

```php
<?php
$sayi=3;
$metin="Bir musibet bin nasihattan iyidir.";
echo "<h$sayi>$metin</h$sayi>\n";
$sayi=5;
echo "<h$sayi>$metin</h$sayi>\n";
$sayi=1;
echo "<h$sayi>$metin</h$sayi>\n";
?>
```

Operatörler

Aritmetik İşlem Operatörleri

Matematiksel hesaplamaları yapmak için kullanılan operatörlerdir.

Operatör	Açıklama

+	Toplama
-	Çıkarma
*	Çarpma
/	Bölme
%	Mod Alma
++	Bir Arttırma
--	Bir Azaltma

% operatörü bir sayının başka bir sayıya bölümünden kalanı vermektedir.

++ operatörü bir değişkenin değeri 1 arttırmakta, -- ise 1 azaltmaktadır. Ancak bunun kullanımına dikkat ediniz. Zira $a++ ile ++$a farklı sonuçlar verebilir. Çünkü $a++ ifadesi önce $a değişkenini kullan sonra 1 arttır anlamına gelmektedir. ++$a ifadesi ise önce 1 arttır sonra kullan anlamına gelmektedir.

```
1   <?php
2   $a=15;
3   $b=6;
4
5   $c=$a++;   // $c=15 değerini alır sonra $a=16 olur
6   $d=++$a;   // $a=16+1=17 oldu ve $d=17 değerini aldı.
7
8   $e=$a%$b;   // $a=17 nin $b=6 ya bölümünden kalanı hesaplar. $e=5 değerini alır
9
10  $f=$b--; // $f=6 değerini alır sonra $b=6-1=5 olur
11
12  $g=--$b; // $b=5-1=4 oldu ve $g=4 değerini aldı.
13
14  $h=++$a-$b++; // $a=17+1=18 oldu, $h=18-4=14 değerini aldı ve $b=4+1=5 oldu
15
16  $i=($a--)+(--$b); // $b=5-1=4 oldu, $i=18+4=22 değerini aldı ve $a=18-1=17 oldu
17
18  echo "a=$a b=$b c=$c d=$d e=$e f=$f g=$g h=$h i=$i";
19  ?>
```

Atama Operatörleri

Operatör	Örnek Kullanım	Açık Kullanımı
=	$a=3;	$a=3;
+=	$a+=$b;	$a=$a+$b;
-=	$a-=$b;	$a=$a-$b;
*=	$a*=$b;	$a=$a*$b;
/=	$a/=$b;	$a=$a/$b;
.=	$a.=$b;	$a=$a.$b;
%=	$a%=$b;	$a=$a%$b;

```
1   <?php
2   $a=15;
3   $b=6;
4
5   $a+=$b; // $a=15+6=21
6   $a-=$b; // $a=21-6=15
7   $a*=$b; // $a=15*6=90
8   $a/=$b; // $a=90/6=15
9   $a.=$b; // $a=15.6=156
10  $a%=$b; // $a=156%6=0
11  echo "a=$a b=$b";
12  ?>
```

PHP'de nokta (.) değişkenleri birleştirip tek bir değer oluşturmak için kullanılır. Virgül (,) ise bir fonksiyona gönderilen parametreleri ayırmak için kullanılır. Nokta (.) PHP'de ondalıklı sayılarda ondalık kısmını ayırmak için de kullanılır.

```php
<?php
$adi="Ahmet";
$soyadi="Erkişi";

// $adi ve $soyadi değişkenleri arasına boşluk eklenerek birleştirilip tek bir değer oluşturuluyor
bu değer $adi_soyadi değişkenine atanıyor.
$adi_soyadi=$adi." ".$soyadi;

$boyu=187;
$kilosu=89.5; // Ondalıklı bir sayı ataması yapılıyor

// Burada echo fonksiyonuna ekrana yazması için iki parametre verilmiştir. $adi_soyadi ve <br>
echo $adi_soyadi,"<br>";

// Burada echo fonksiyonuna tek parametre verilmiştir. Çünkü tüm değerler birleştirilip tek bir
değer olarak echo fonksiyonuna gönderiliyor.
echo "Boyu: ".$boyu."<br>Kilosu: ".$kilosu;
?>
```

Karşılaştırma Operatörleri

Operatör	Açıklama
==	Eşittir
<	Küçüktür
>	Büyüktür
<=	Küçük Eşittir
>=	Büyük Eşittir
!=	Eşit Değildir
<>	Eşit Değildir (Farklıdır)

Mantıksal Operatörler

Operatör	Açıklama

&&	ve
||	veya
!	Değil

Not: **&&** yerine **and** sözcüğü, || yerine **or** sözcüğü de kullanılabilir.

Değişken Tipleri

Tamsayı Değişkenler (Integer)

Bilinen negatif ve pozitif tamsayılardır. Z={..., -2, -1, 0, 1, 2, ... }

```
1  <?php
2  $a=5;
3  $b=-24;
4  $c=160;
5  ?>
```

Ondalıklı Sayı Değişkenleri (float)

Ondalık kısmı bulunan negatif ve pozitif sayılardır. Ondalıklı kısmı ayırmak için nokta (.) kullanılır.

```
1  <?php
2  $a=5.3;
3  $b=-2.4;
4  $c=1.60;
5  ?>
```

Metin Değişkenleri (String)

Karakterlerden oluşan değerleri saklayan değişkenlerdir. Karakter olarak a-z, A-Z, 0-9 ve diğer simgeler kullanılabilir. String değişkenlere atanacak değerler çift tırnak (" ") arasında yazılır.

```
1  <?php
2  $adi="Ali";
3  $soyadi="KAVAK";
4  $sifre="Kxt87?_[/";
5  ?>
```

String ifadeleri kullanırken çift tırnak içerisinde PHP için özel anlama sahip ifadeler varsa bunlar yorumlanır ve bunların karşılıkları kullanılır. Örneğin $ işareti bir değişkenin varlığını söyler. Eğer çift tırnak içerisinde $ ile başlayan bir ifade varsa bu değişken olarak algılanır ve yerine bunun değeri getirilir.

```
1  <?php
2  $gun=15;
3  $ay="Mart";
4  $yil=2010;
5  $tarih="Bugün $gun $ay $yil dur. ";
6  echo $tarih; // Ekrana Bugün 15 Mart 2010 dur. yazacaktır
7  $c=160;
8  ?>
```

Eğer string ifadelerin içindeki özel anlamı olan ifadeleri devre dışı bırakmak isterseniz bu ifadelerin önüne \ koymalısınız. (Ekrana \ yazmak isterseniz \\ kullanabilirsiniz)

```
1  <?php
2  $gun=15;
3  $ay="Mart";
4  $yil=2010;
5  echo "\$gun=$gun \$ay=$ay \$yil=$yil"; // Ekrana $gun=15 $ay=Mart $yil=2010 yazacaktır
6  $c=160;
7  ?>
```

Bir başka yöntem ise string değişken içerisindeki tüm özel ifadeleri devre dışı bırakmaktır. Bunun için çift tırnak yerine tek tırnak kullanılır. Eğer string ifade tek tırnaklar arasına yazılırsa içerisindeki hiçbir özel ifadeye bakılmaz olduğu gibi ekrana yazılır.

```
1  <?php
2  $gun=15;
3  $ay="Mart";
4  $yil=2010;
5  $tarih='Bugün $gun $ay $yil dur.';
6  echo $tarih; // Ekrana Bugün $gun $ay $yil dur yazacaktır
7  $c=160;
8  ?>
```

Aşağıdaki örneği inceleyebilirsiniz.

```php
<?php
$ali="Ben gelmiyorum!";
echo "Ali o kadar kızdı ki \"Ben gelmiyorum!\" diye bağırdı.";
echo "<br>";

$yol="c:\\windows";
echo $yol;
echo "<br>";

$tag='<font color="red" size="7">Nasıl Ama!</font>';
echo $tag;
echo "<br>";

$ders="PHP'de değişkenlerin önüne \$ işareti konulur. Örnek: \$sayi=15;";
echo $ders;
echo "<br>";
?>
```

Boolean Değişkenler

Yalnızca true yada false değerini alan değişkenlerdir. Aslında sıfırdan farklı her değer **true**, geriye kalan 0 yada null ise **false** olarak bilinir.

```php
<?php
$a=true;
$b=false;
$c=1; // $c=true
$d=0; //$d=false
?>
```

Sabitler Değişkenler

Değeri değiştirilemeyen değişkenlerdir. Bunlar önceden tanımlanır ve program akışında sadece kullanılırlar. Değerleri değiştirlemez. Bir sabit **define(değişkenadı, değeri);** şeklinde tanımlanır. Aşağıdaki örneği inceleyiniz.

```php
<?php
define("pi",3.14); // sabitler kullanılırken önüne $ işareti konulmaz ve adı çift tırnak içinde
tanımlanır
echo pi; // ekrana 3.14 yazar.
?>
```

Açıklamada da yazıldığı gibi sabit değişkenler tanımlanırken ve kullanılırken önüne $ işareti konulmaz. Aşağıda sabitlerin kullanımı ile ilgili olarak yapılan birkaç hata ve açıklaması vardır.

```php
<?php
define("pi",3.14); // Doğru bir sabit tanımlaması yapılmıştır
echo $pi; // Hata: Sabitin önüne $ konulmuş
pi=3; // Hata: Sabitin değeri değiştirilmeye çalışılıyor
$alan=pi*2*2; // Burada herhangi bir hata yoktur. Alan hesaplaması için sabit kullanılmıştır.
?>
```

Birkaç farklı sabit tanımlaması da aşağıda verilmiştir.

```php
<?php
define("kullaniciadi","mustix"); // string bir sabit tanımlanmıştır
define("edit",true); // boolean bir sabit tanımlanmıştır
define("yil",2010); // integer bir sabit tanımlanmıştır
define("katsayi",0.013); // float bir sabit tanımlanmıştır.

echo kullaniciadi." ".edit." ".yil." ".katsayi." "; // ekrana mustix 1 2010 0.013 yazar
?>
```

Dizi Değişkenler

Diziler, içerisinde bir çok değer saklayabilen değişkenler kümesi olarak tanımlanabilir. Örneğin 30 kişilik bir sınıftaki öğrencilerin adlarını saklamak üzere 30 tane değişken tanımlamaktansa 30 elemanlı bir dizi tanımlamak daha kolaydır.

Dizideki bir değer ulaşmak için indis'ler kullanılır. Aşağıdaki örneği inceleyiniz.

```php
<?php
$meyveler[0]="Elma";
$meyveler[1]="Armut";
$meyveler[2]="İncir";
echo $meyveler[1]; // ekrana Armut yazar.
?>
```

Buradaki 0, 1 ve 2 indislerdir. Elma, Armut ve İncir ise değerlerdir.

Örnek: Günleri saklayan bir dizi tanımlayalım.

```php
<?php
$gunler[0]="Pazartesi";
$gunler[1]="Salı";
$gunler[2]="Çarşamba";
$gunler[3]="Perşembe";
$gunler[4]="Cuma";
$gunler[5]="Cumartesi";
$gunler[6]="Pazar";
?>
```

Burada indis tanımlamaları çok esnektir. Biz burada Pazartesi için 0 kullandık. Sizler Pazartesi için 1 kullanabilirsiniz. Burada yapılan iş aslında diziye yeni değerler eklemektir.

Bazen bir dizide kaçtane değer olduğunu, kullanılan enson indisin kaç olduğunu bilmediğimiz durumlar olur. Ya da indisi yazmak istemediğiniz durumlar da olabilir. Bu tür durumlarda diziye ekleme yapmak için şu yöntemi kullanabilirsiniz.

```
<?php
$gunler[]="Pazartesi";
$gunler[]="Salı";
$gunler[]="Çarşamba";
$gunler[]="Perşembe";
$gunler[]="Cuma";
$gunler[]="Cumartesi";
$gunler[]="Pazar";
echo $gunler[3]; // ekrana Perşembe yazar.
?>
```

Dizilerde indis olarak sayılar kullanmak zorunda değilsiniz. İndis olarak string ifadeler de kullanılabilir. Biz buna değerleri etiketlemek diyoruz. Bir örnekle açıklamak gerekirse. Gerçek hayatta okuldaki odalar numaralandırılmaz. Odalara isimler verilir. Örneğin müdür odasının kapısına Müdür Odası etiketini, öğretmenler odasına Öğretmenler Odası etiketini asarlar. Yani odalar etiketlenir. Benzer mantıkla bir dizideki değerler de etiketlenebilir. Aşağıdaki örneği inceleyiniz.

```
<?php
$personel["güvenlik"]="Ali Kavak";
$personel["sekreter"]="Ayşe Eren";
$personel["müdür"]="Ahmet Ergün";
$personel["hizmetli"]="Ercan Kır";
echo $personel["müdür"]; // ekrana Ahmet Ergün yazar.
?>
```

Bir dizideki değerleri yukarıdaki örneklerde olduğu gibi tek tek tanımlamak zorunda değilsiniz. Bu değerleri toplu bir şekilde tanımlamak için şu yapıyı kullanmalısınız.

```
$gunler=array("Pazartesi","Salı","Çarşamba","Perşembe","Cuma");
```

Bu tür bir tanımlamada dizinin ilk değerinin indisi 0, diğerleri sırasıya 1, 2, ... şeklinde devam eder. Ancak siz böyle bir tanımlama yaparken indisin 0 dan başlamasını istemeyebilirsiniz. Bu tür durumlarda her değerin indisi belirtebilirsiniz.

```
$gunler=array(1=>"Pazartesi",2=>"Salı",3=>"Çarşamba",4=>"Perşembe",5=>"Cuma");
```

Eğer indisler sıralı bir şekilde ard arda gidecekse işlemi biraz daha kısaltabiliriz. Bunun işin ilk değere bir indis verilir. Diğerlerine verilmediği zaman sıradaki indis değerini alırlar.

```
1  $gunler=array(1=>"Pazartesi","Salı","Çarşamba","Perşembe","Cuma");
```

Bu tanımlama görüldüğü gibi Pazartesi 1 indisini alırken sırasıyla Salı 2, Çarşamba 3 indislerini alacaktır. Bu işlem bu şekilde devem edecektir.

Aşağıda farklı şekilde tanımlanmış diziler bulunmaktadır. İnceleyiniz.

```
1  <?php
2  $dersler[1]="Matematik";
3  $dersler[2]="Fizik";
4  $dersler[3]="Geometri";
5  echo $dersler[2]; // ekrana Fizik yazar
6  echo "<br>";
7  //----------------------------
8  $takim[]="Ali";
9  $takim[]="Ömer";
10 $takim[]="Erkan";
11 $takim[]="Serkan";
12 $takim[]="Emre";
13 echo $takim[3]; // ekrana Serkan yazar
14 echo "<br>";
15 //----------------------------
16 $gorevler["pazartesi"]="Faturalar ödenecek";
17 $gorevler["çarşamba"]="Alışveriş yapılacak";
18 $gorevler["cumartesi"]="Temizlik yapılacak";
19 echo $gorevler["çarşamba"]; // ekrana Alışveriş yapılacak yazar.
20 echo "<br>";
21 //----------------------------
22 $yaz=array("Haziran","Temmuz","Ağustos");
23 echo $yaz[1]; // ekrana Temmuz yazar
24 echo "<br>";
25 //----------------------------
26 $bahar=array(1=>"Mart","Nisan","Mayıs");
27 echo $bahar[3]; // ekrana Mayıs yazar
28 echo "<br>";
29 //----------------------------
30 $sorular=array(15=>"Ram Nedir?",65=>"CPU ne anlama gelir?",18=>"1MB=?KB");
31 echo $sorular[18]; // ekrana 1MB=?KB yazar.
32 echo "<br>";
33 //----------------------------
34 $futbol=array("kaleci"=>"Can","defans"=>"Ercan","orta"=>"Ali","forvet"=>"şükrü");
35 echo $futbol["defans"]; // ekrana Ercan yazar
36 ?>
```

Örnek: Aşağıdaki tabloyu dizi olarak tanımlayalım.

Linux	
KDE	Pardus

GNOME	Fedora
XFCE	Xubuntu

```
1  <?php
2  $linux=array("KDE"=>"Pardus","GNOME"=>"Fedora","XFCE"=>"Xubuntu");
3  ?>
```

Örnek: Aşağıdaki tabloyu dizi olarak tanımlayalım.

Diller	
1	PHP
2	PTYHON
3	C

```
1  <?php
2  $diller=array(1=>"PHP","PYTHON","C");
3  ?>
```

Örnek: Aşağıdaki tabloyu dizi olarak tanımlayalım.

Kazananlar
Ayşe
Oya
Meral
Canan

```
1  <?php
2  $kazananlar=array("Ayşe","Oya","Meral","Canan");
3  ?>
```

Çok Boyutlu Diziler

Dizi içindeki diziler olarak düşünülebilir. Yine bir örnekle açıklamak gerekirse bir otelin katları bir dizi olarak düşünülürse katlardaki odalar da bir dizi olarak düşünülebilir. Dolayısıyla burada dizi içindeki diziler söz konusudur. Başka bir örnek olarak şu verilebilir: Sınıftaki öğrenciler bir dizi, onları notları ise yine bir dizi olarak dününülürse yine karşımıza dizi içindeki diziler çıkmaktadır.

Çok boyutlu dizilerde her boyutun indisi için [ ] kullanılır. Aşağıdaki örneği inceleyebilirsiniz.

Otel		
Kat1	Oda1	Ali
	Oda2	Ahmet
	Oda3	Ömer
Kat2	Oda1	Hasan
	Oda2	Kenan
	Oda3	Mert
Kat3	Oda1	Ayşe
	Oda2	Fatma
	Oda3	Hatice

```
1   <?php
2   $otel["kat1"]["oda1"]="Ali";
3   $otel["kat1"]["oda2"]="Ahmet";
4   $otel["kat1"]["oda2"]="Ömer";
5
6   $otel["kat2"]["oda1"]="Hasan";
7   $otel["kat2"]["oda2"]="Kenan";
8   $otel["kat2"]["oda3"]="Mert";
9
10  $otel["kat3"]["oda1"]="Ayşe";
11  $otel["kat3"]["oda2"]="Fatma";
12  $otel["kat3"]["oda3"]="Hatice";
13
14  echo $otel["kat2"]["oda3"]; // ekrana Mert yazar
15  ?>
```

Görüldüğü gibi katlar dizinin birinci boyutu, odalar ise ikinci boyutudur. Aynı diziyi şimdi farklı bir şekilde tanımlayalım.

```
<?php
$otel=array(
 "kat1"=>array("oda1"=>"Ali","oda2"=>"Ahmet","oda3"=>"Ömer"),
 "kat2"=>array("oda1"=>"Hasan","oda2"=>"Kenan","oda3"=>"Mert"),
 "kat3"=>array("oda1"=>"Ayşe","oda2"=>"Fatma","oda3"=>"Hatice")
);

echo $otel["kat2"]["oda3"]; // ekrana Mert yazar
?>
```

Not: En son tanımlanan dizinin sonunda virgül (,) yoktur. Dikkat ediniz. Çünkü virgül dizi elemanlarını ayırır. En son elamandan sonra elaman olmadığından virgül konmaz.

Bir sınıftaki öğrencilerin matematik dersinden almış oldukları 3 notu öğrenci numaralarına göre saklayan bir dizi tanımlayalım.

Matematik		
155	1	52
	2	64
	3	35
225	1	98
	2	90
	3	70
302	1	15
	2	45
	3	63

```php
<?php
$matematik[155][1]=52;
$matematik[155][2]=64;
$matematik[155][3]=35;

$matematik[225][1]=98;
$matematik[225][2]=90;
$matematik[225][3]=70;

$matematik[302][1]=15;
$matematik[302][2]=45;
$matematik[302][3]=63;
?>
```

Burada birinci boyut öğrenci numaraları (155, 225 ve 302), ikinci boyut ise sınavlar (1, 2 ve 3) olarak düşünülmüştür. Bunlara atanan sayılar ise değerler yani öğrencilerin almış oldukları notlar (52, 64, 35, ...) olarak düşünülebilir. Aynı diziyi toplu bir şekilde tanımlamak istersek aşağıdaki gibi yapabiliriz.

```php
<?php
$matematik=array(
 155=>array(1=>52,2=>64,3=>35),
 225=>array(1=>98,2=>90,3=>70),
 302=>array(1=>15,2=>45,3=>63)
);
?>
```

Örnek: Aşağıdaki tabloda 3 farklı sınıfın 4'er öğrencisinin gösterildiğini kabul ederek bunu bir dizi olarak tanımlayalım.

Okul	
Birinci sınıf	Ebru
	Fuat
	Gülüzar
	Emre
İkinci sınıf	Zeynep
	Haşim

	Merve
	Nihat
Üçüncü sınıf	Raşit
	Gizem
	Onur
	Ayşe

```
1   <?php
2   $okul[0][]="Ebru";
3   $okul[0][]="Fuat";
4   $okul[0][]="Gülüzar";
5   $okul[0][]="Emre";
6
7   $okul[1][]="Zeynep";
8   $okul[1][]="Haşim";
9   $okul[1][]="Merve";
10  $okul[1][]="Nihat";
11
12  $okul[2][]="Raşit";
13  $okul[2][]="Gizem";
14  $okul[2][]="Onur";
15  $okul[2][]="Ayşe";
16
17  echo $okul[1][1]; // ekrana Haşim yazar
18  ?>
```

Yukarıdaki tanımlamada dikkat ederseniz öğrenciler için indis belirtilmemiştir. Dolayısıyla öğrenciler dizinin sonuna eklenecektir. Böylece ikinci boyutun indisleri ise 0,1,2 ve 3 olacaktır. İkinci boyuttaki 0; birinci öğrenci olarak düşünülürse, 1,2 ve 3 ise sırasıyla diğer öğrenciler olarak düşünülebilir. Bu tamamen sizin tasarrufunuzda olan bir anlamlandırmadır. Aynı diziyi faklı bir şekilde tanımlayalım.

```
1   <?php
2   $okul[]=array("Ebru","Fuat","Gülüzar","Emre");
3   $okul[]=array("Zeynep","Haşim","Merve","Nihat");
4   $okul[]=array("Raşit","Gizem","Onur","Ayşe");
5
6   echo $okul[1][2]; // ekrana Merve yazar
7   ?>
```

Dikkat ederseniz burada hiçbir indis belirtilmemiştir. Dolayısıyla her dizi elemanı dizinin sonuna eklenecektir. Böylece her iki boyutun indisleri 0 dan başlayacaktır. Eğer indislerin 1 den başlmasını isterseniz şöyle bir tanımlama yapabilirsiniz.

```
1  <?php
2  $okul[1]=array(1=>"Ebru","Fuat","Gülüzar","Emre");
3  $okul[]=array(1=>"Zeynep","Haşim","Merve","Nihat");
4  $okul[]=array(1=>"Raşit","Gizem","Onur","Ayşe");
5
6  echo $okul[1][2]; // ekrana Merve yazar
7  ?>
```

Şimdi de aynı diziyi iki farklı şekilde daha tanımlayalım.

Aşağıdaki tanımlamada her iki boyutun indisleri 0 dan başlar.

```
1  <?php
2  $okul=array(
3   array("Ebru","Fuat","Gülüzar","Emre"),
4   array("Zeynep","Haşim","Merve","Nihat"),
5   array("Raşit","Gizem","Onur","Ayşe")
6  );
7
8  echo $okul[1][2]; // ekrana Merve yazar
9  ?>
```

Aşağıdaki tanımlamada ise her iki boyutun indisleri 1 den başlar.

```
1  <?php
2  $okul=array(
3   1=>array(1=>"Ebru","Fuat","Gülüzar","Emre"),
4   array(1=>"Zeynep","Haşim","Merve","Nihat"),
5   array(1=>"Raşit","Gizem","Onur","Ayşe")
6  );
7
8  echo $okul[1][2]; // ekrana Fuat yazar
9  ?>
```

Dizilerdeki boyut sayısını daha da artırabilirsiniz. O zaman dizi içindeki dizilerin içindeki dizilerden bahsetmiş olursunuz ki bunlarında mantığı yukarıda anlatılan iki boyutlu diziler gibidir. İhtiyaç duyulduğunda bunlar da kullanılabilir. Örneğin 3 boyutlu bir dizi olarak, okuldaki sınıflarda bulunan öğrencilerin coğrafya dersinden aldıkları 2 farklı notu saklayan bir dizi düşünülebilir.

```php
<?php
// birinci sınıf
$cografya[0][156][]=74; // birinci sınıfın 156 nolu öğrencisinin birinci coğrafya notudur.
$cografya[0][156][]=68; // birinci sınıfın 156 nolu öğrencisinin ikinci coğrafya notudur.
$cografya[0][174][]=18; // birinci sınıfın 174 nolu öğrencisinin birinci coğrafya notudur.
$cografya[0][174][]=23; // birinci sınıfın 174 nolu öğrencisinin ikinci coğrafya notudur.

// ikinci sınıf
$cografya[1][223][]=56; // ikinci sınıfın 223 nolu öğrencisinin birinci coğrafya notudur.
$cografya[1][223][]=61; // ikinci sınıfın 223 nolu öğrencisinin ikinci coğrafya notudur.
$cografya[1][254][]=41; // ikinci sınıfın 254 nolu öğrencisinin birinci coğrafya notudur.
$cografya[1][254][]=63; // ikinci sınıfın 254 nolu öğrencisinin ikinci coğrafya notudur.

// üçüncü sınıf
$cografya[2][415][]=33; // üçüncü sınıfın 415 nolu öğrencisinin birinci coğrafya notudur.
$cografya[2][415][]=51; // üçüncü sınıfın 415 nolu öğrencisinin ikinci coğrafya notudur.
$cografya[2][521][]=89; // üçüncü sınıfın 521 nolu öğrencisinin birinci coğrafya notudur.
$cografya[2][521][]=96; // üçüncü sınıfın 521 nolu öğrencisinin ikinci coğrafya notudur.

echo $cografya[1][254][1]; // ekrana 63 yazar. ikinci sınıfın 254 nolu öğrencisinin ikinci
coğrafya notudur.
?>
```

Aynı diziyi toplu bir şekilde tanımlayalım.

```php
<?php
$cografya=array(
  array( // birinci sınıf
       156=>array(74,68), // birinci öğrenci
       174=>array(18,23) // ikinci öğrenci
  ),
  array( // ikinci sınıf
       223=>array(56,61), // birinci öğrenci
       254=>array(41,63) // ikinci öğrenci
  ),
  array( //üçüncü sınıf
       415=>array(33,51), // birinci öğrenci
       521=>array(89,96) // ikinci öğrenci
  )
);

echo $cografya[1][254][1]; // ekrana 63 yazar. ikinci sınıfın 254 nolu öğrencisinin ikinci
coğrafya notudur.
?>
```

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.

```
1  <?php
2  $kullanici_adi="hayrix";
3  $sifre="muric";
4  if ($kullanici_adi=="mustix" and $sifre=="muric")
5    echo "Giriş Başarılı";
6  else
7    echo "Kullanıcı adı yada şifre yanlış!";
8  ?>
```

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.

```
1  <?php
2  $sayi=7;
3  if ($sayi%2==1) // sayının 2'ye bölümünden kalan 1 ise
4    $sayi++;
5  echo "Sayı=$sayi"; // ekrana Sayı=8 yazar
6  ?>
```

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.

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

Aynı örneği if yapısını açıkça yazarak yapalım.

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

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

```php
<?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
<?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
<?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
<?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
<?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
<?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
<?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
<?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
<?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";
?>
```

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.

```
1   <html>
2   <body>
3   <?php
4   for($i=1;$i<=10;$i++){
5     echo "Merhaba Dünya";
6     echo "<br>";
7   }
8   ?>
9   </body>
10  </html>
```

İlk örnek olduğundan örneği açıklayalım: 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.

```
1   <?php
2   for ($i=1;$i<=10;$i++)
3     echo "$i<br>";
4   ?>
```

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

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

```
1   <?php
2   for ($i=1;;$i++){
3     if ($i>10)
4       break; // döngüyü kırmak için kullanılır.
5     echo "$i<br>"; // Dikkat: Bu komut if yapısına bağlı değildir. {} parantezleri yok.
6   }
7   ?>
```

```
<?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
<?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
<?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
<?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
<?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
<?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
$aylar=array("Ocak","Şubat","Mart","Nisan","Mayıs","Haziran","Temmuz","Ağustos","Eylül","
Ekim","Kasım","Aralık");
echo "<select name='aylar'>";
for ($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.

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

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

```
<?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
$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
$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
$sayac=1;
while($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
<?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.

```
1  do{
2  // döngü içinde çalışacak kodlar
3  }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.

```
1  <?php
2  do{
3     $sayi=rand(1,100);
4  }while($sayi!=45);
5
6  echo "Bulunan sayı: $sayi";
7  ?>
```

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

```
1   <?php
2   $toplam=0;
3   $sayac=0;
4   do{
5      $sayi=rand(1,100);
6      $toplam+=$sayi;
7      $sayac++;
8      echo ($sayac==1)?"$sayi":"+$sayi";
9   }while($sayac<10);
10
11  echo "=$toplam";
12  ?>
```

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
$personel=array("Müdür"=>"Nihat","Yardımcı"=>"Uğur","Memur"=>"Haşim","Hizmetli"=>"Uf
uk");
foreach($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
<?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.

Fonksiyonlar

Fonksiyonlar, belirli bir işi yapması için önceden hazırlanmış olan program parçalarıdır. Bir kere tanımlanırlar daha sonra ihtiyaç duyulan yerlerde çağrılıp kullanılırlar. Bu açıdan fonksiyonlar programlama dillerinin ayrılmaz parçalarıdır.

PHP'de önceden tanımlı birçok fonksiyonun yanında kullanıcılar da istedikleri fonksiyonları tanımlayıp kullanabilirler. Örneğin sürekli olarak yaptığınız bir hesaplama için farklı sayfalarda hep aynı kodları yazmak ya da kopyalayıp yapıştırmak hem zaman alıcı bir işlemdir hem de gereksiz bir kodlamadır. Bunun yerine fonksiyon, bir sayfada tanımlanır daha sonra ihtiyaç duyulan yerlerde çağrılıp kullanılır. Dolayısıyla tekrar tekrar yazılması gerekmez.

Şu ana kadar kullandığımız echo() ve rand() komutları da birer fonksiyondur. Bu şekilde PHP'de tanımlanmış birçok kullanışlı fonksiyonlar vardır. Örneğin biz, bir diziyi sıralamak için iç içe döngüler kullanıp karmaşık kodlar yazarak bu işi ancak yapabilmişken bunu sort() fonksiyonu ile kolayca yapabiliyoruz. Başka bir örnek vermek gerekirse; diyelim ki bir sayının istediğimiz bir üssünü hesaplamak istersek bunun için uzun uzun kodlar yazmamız gerekecektir. Ancak pow() fonksiyonu bunu zaten kendi içinde bizim için hesaplamaktadır. Bize düşen sadece pow($sayi,$us) yazmaktır.

Bir fonksiyon yapısına göre;

- Dışarıdan parametre alıp, hesaplama yapıp geriye değer gönderebilir

- Dışarıdan parametre alıp, hesaplama yapıp geriye bir değer göndermeyebilir
- Dışarıdan parametre almadan hesaplama yapıp geriye değer gönderebilir
- Dışarıdan parametre almadan hesaplama yapıp geriye değer göndermeyebilir

Fonksiyonlar, tanımlamasına göre yukarıdakilerden birine uyacak şekilde çalışırlar. Ancak genel bir fonksiyon tanımlaması şu şekilde yapılır.

```
1  function fonksiyon_adi($parametre1, $parametre2, ...){
2    // fonksiyonun yapacağı iş ya da hesaplamalar
3    return $deger;
4  }
```

Yukarıdaki tanımlamayı özetle açıklamak gerekirse;

- Fonksiyon tanımı **function** sözcüğüyle başlar.
- Fonksiyonun adı, sayı ile başlayamaz, türkçeye özgü karakterler ve boşluk barındıramaz. Başına $ işareti konmaz.
- Parametreler yazılabilir ya da yazılmayabilir (isteğe bağlı)
- **return** ile geriye değer gönderilebilir ya da gönderilmeyebilir (isteğe bağlı)
- **return** ifadesi fonksiyonun bittiği yerdir. **return**'den sonra yazılan kodlar çalışmaz, göz ardı edilir.
- Bir fonksiyonda bir tane **return** olabilir. (Ancak if yapısında ya da switch case yapısında birden fazla return kullanılabilir. Zaten durumlardan sadece bir sağlanacağı için **return**'lerden sadece biri çalışacaktır)

Bir fonksiyonun yapısında her tür php kodu kullanılabilir. Hatta fonksiyon kendi içerisinde farklı fonksiyonları çağırıp kullanabilir.

Fonksiyon Tanımlama

Kullanıcı tanımlı fonksiyonlara çok basit bir örnekle başlayalım. Bu örnekte ekrana sadece Merhaba Dünya! yazan bir fonksiyon tanımlayıp çağıralım.

```
1   <html>
2   <body>
3   <?php
4   function yaz(){
5     echo "Merhaba Dünya!";
6   }
7
8   yaz();
9   ?>
10  </body>
11  </html>
```

Yazdığımız örnekte fonksiyonumuzun adı **yaz**'dır. Görüldüğü üzere hiçbir parametre almadığı gibi geriye de herhangi bir değer göndermiyor. Daha sonra tanımladığımız fonksiyonu **yaz();** diyerek çağırıyoruz. Hiçbir parametre almasa dahi () parantezlerini kullanmamız gerekir.

Şimdi örneğimizi biraz geliştirelim. Bu kez fonksiyonuz dışarıdan aldığı metni ekrana yazsın.

```
1   <html>
2   <body>
3   <?php
4   function yaz($metin){
5     echo $metin,"<br>";
6   }
7
8   yaz("Fazla laf yalansız olmaz!");
9
10  $gunun_sozu="Bu kadar cehalet ancak tahsille olur!";
11  yaz($gunun_sozu);
12  ?>
13  </body>
14  </html>
```

Örnek: Bu kez aldığı iki sayıyı toplayarak geriye sonucu gönderen bir fonksiyon tanımlayalım ve çağıralım.

```
1   <?php
2   function topla($sayi1,$sayi2){
3     $toplam=$sayi1+$sayi2;
4     return $toplam;
5   }
6
7   $sonuc=topla(45,14);
8   echo $sonuc,"<br>";
9
10  $s1=45;
11  $s2=14;
12  $sonuc=topla($s1,$s2);
13  echo $sonuc,"<br>";
14
15  echo topla(45,14),"<br>";
16  echo topla($s1,$s2),"<br>";
17  ?>
```

Eğer bir fonksiyon return ile geriye bir değer gönderiyorsa o fonksiyon kullanılırken bir değişkene atanmalı ki; bu değişken fonksiyondan dönen değeri tutacaktır. Daha sonra bunu istediğiniz yerde kullanabilirsiniz. Burada biz, $sonuc isimli bir değişken kullandık. Diğer bir yöntem ise dönen değeri başka bir fonksiyona kullanması için vermektir. Yukarıdaki örnekte, echo topla(45,14),"
"; diyerek topla() fonksiyonundan dönen değeri doğrudan ekrana yazması için echo() fonksiyonuna verdik.

Örnek: Aldığı iki sayıyı, aldığı operatöre göre işleme alıp sonucu gönderen fonksiyonu tanımlayıp çağıralım.

```
1   <?php
2   function hesapla($sayi1,$sayi2,$islem){
3     switch ($islem){
4       case "+": $sonuc=$sayi1+$sayi2; break;
5       case "-": $sonuc=$sayi1-$sayi2; break;
6       case "*": $sonuc=$sayi1*$sayi2; break;
7       case "/": $sonuc=$sayi1/$sayi2; break;
8       default: $sonuc="Geçersiz işlem";
9     }
10    return $sonuc;
11  }
12
13  $sonuc=hesapla(6,4,"*");
14  echo $sonuc,"<br>";
15
16  $s1=42;
17  $s2=3;
18  echo hesapla($s1,$s2,"+"),"<br>";
19  echo hesapla($s1,$s2,"-"),"<br>";
20  echo hesapla($s1,$s2,"*"),"<br>";
21  echo hesapla($s1,$s2,"/"),"<br>";
22  ?>
```

Örnek: Aldığı metni aldığı sayı kadar ekrana yazan fonksiyonu tanımlayıp çağıralım.

```
1   <?php
2   function yaz($metin,$sayi){
3     for ($i=1;$i<=$sayi;$i++)
4       echo $metin,"<br>";
5   }
6
7   yaz("Bu metni on kere yaz bakalım.",10);
8
9   $txt="Yazdır yazdır nereye kadar?";
10  $number=20;
11  yaz($txt,$number);
12  ?>
```

Fonksiyon Örnekleri

Örnek: Aldığı iki notun ortalamasını gönderen fonksiyonu tanımlayıp çağıralım.

```php
<?php
function ortalama($not1,$not2){
  return ($not1+$not2)/2;
}

$y1=64;
$y2=88;
echo "<b>Not1</b>=$y1 <b>Not2</b>=$y2 <b>Ortalama</b>=",ortalama($y1,$y2);
?>
```

Örnek: Aldığı satır ve sütun sayısına göre her sütunun genişliği 80px olacak şekilde bir tablo oluşturan fonksiyonu tanımlayıp çağıralım.

```php
<?php
function tablo($satir,$sutun){
  echo "<table border=\"1\" width=\"",$sutun*80,"\">";
  for($i=1;$i<=$satir;$i++){
    echo "<tr>";
    for($j=1;$j<=$sutun;$j++)
      echo "<td> </td>";
    echo "</tr>";
  }
  echo "</table>";
}

tablo(5,8);
echo "<br>";
$row=9;
$col=12;
tablo($row,$col);
?>
```

Örnek: Aldığı sayıya karşılık gelen ay adını gönderen fonksiyonu tanımlayıp çağıralım.

```php
<?php
function ay_yaz($sayi){

$aylar=array("Ocak","Şubat","Mart","Nisan","Mayıs","Haziran","Temmuz","Ağustos","Eylül","
Ekim","Kasım","Aralık");
  return $aylar[$sayi-1];
}

$gun=30;
$ay=ay_yaz(3);
$yil=2010;
echo "<strong>Tarih</strong>: $gun $ay $yil";
?>
```

Örnek: Aldığı diziyi ekranda sırasız liste şeklinde gösteren fonksiyonu yazıp çağıralım.

```php
<?php
function liste($elemanlar){
  echo "<ul>";
  foreach($elemanlar as $eleman)
    echo "<li>$eleman</li>";
  echo "</ul>";
}

$ic_donanim=array("İşlemci","Anakart","RAM","Ekran Kartı","Sabitdisk");
liste($ic_donanim);

liste(array("Klavye","Fare","Ekran","Yazıcı","Kasa","Hoparlör"));
?>
```

Örnek: Aldığı dizinin değerlerini açılır listeye ekleyen fonksiyonu yazıp çağıralım.

```php
<?php
function acilir_liste($secenekler){
  echo "<select>";
  foreach($secenekler as $secenek)
    echo "<option>$secenek</option>";
  echo "</select>";
}

$merkezler=array("Ankara","İstanbul","İzmir","Adana","Erzurum");
acilir_liste($merkezler);
?>
```

Örnek: Aldığı 3 basamaklı sayının birler, onlar ve yüzler basamağındaki rakamları bir dizi olarak geriye gönderen fonksiyonu yazıp çağıralım.

```php
<?php
function basamaklar($sayi){
  $birler=$sayi%10;
  $onlar=(($sayi-$birler)/10)%10;
  $yuzler=(($sayi-$birler)-$onlar*10)/100;
  return array("birler"=>$birler,"onlar"=>$onlar,"yüzler"=>$yuzler);
}

$sayimiz=854;
$donen_dizi=basamaklar($sayimiz);
echo "Sayımız=$sayimiz<br>";
echo "Birler=",$donen_dizi["birler"],"<br>";
echo "Onlar=",$donen_dizi["onlar"],"<br>";
echo "Yüzler=",$donen_dizi["yüzler"],"<br>";
?>
```

Örnek: Çağrıldığında içerisinde 1 ile 100 arasında 10 sayı bulunan bir dizi gönderen fonksiyonu tanımlayıp çağıralım.

```php
<?php
function secilenler(){
  for($i=1;$i<=10;$i++)
    $dizi[]=rand(1,100);
  return $dizi;
}

$donen_dizi=secilenler();
foreach($donen_dizi as $deger)
echo $deger," ";
?>
```

Örnek: Aldığı dizideki sayıların en büyüğünü, en küçüğünü ve ortalamasını gönderen fonksiyonu yazıp çağıralım.

```php
<?php
function isle($sayilar){
  $eb=$sayilar[0];
  $ek=$sayilar[0];
  $toplam=0;
  $sayac=0;
  foreach($sayilar as $deger){
    $ek=($deger<$ek)?$deger:$ek;
    $eb=($deger>$eb)?$deger:$eb;
    $toplam+=$deger;
    $sayac++;
  }
  return array("ek"=>$ek,"eb"=>$eb,"ort"=>$toplam/$sayac);
}

$degerler=array(43,6,15,41,97,65,84,68,45,64,94,72,56);
$sonuc=isle($degerler);
echo "En küçük sayı=",$sonuc["ek"],"<br>";
echo "En büyük sayı=",$sonuc["eb"],"<br>";
echo "Ortalama=",$sonuc["ort"],"<br>";
?>
```

Varsayılan Değer Alan Fonksiyonlar

Tanımlı olan bir fonksiyonu çağırırken tanımlamasında belirtilen sayıda ve tipte parametre göndermek gerekmektdir. Eğer fonksiyonun beklediği parametreler gelmezse fonksiyon hata verecektir. Bu durumda fonksiyonun hatasız çalışması için parametrelere varsayılan değer ataması yapılır.

Eğer fonksiyonun beklediği parametre gelmezse varsayılan değeri, gelirse gelen değerler kullanılır. Aşağıdaki örneği inceleyiniz.

```php
<?php
function ortalama($not1,$not2,$not3){
  $ort=($not1+$not2+$not3)/3;
  return $ort;
}

echo "Ortalama=",ortalama(60,90); // hata verecektir. fonksiyon 3 parametre istiyor
?>
```

Örnekte gönderilen notların ortalamasını bulmak istiyoruz. Fonksiyon tanımlmasına bakıldığında 3 tane parametre yani not beklediği görüldüğü halde 2 tane not gönderilerek çağırılmış. Bundan dolayı hata oluşmaktadır. Hatayı engellemek için 3. not için varsayılan değer ataması yapalım.

```php
<?php
function ortalama($not1,$not2,$not3="yok"){
  if ($not3=="yok")
    $ort=($not1+$not2)/2;
  else
    $ort=($not1+$not2+$not3)/3;
  return $ort;
}

echo "Ortalama=",ortalama(60,90); // hatasız çalışacaktır.
echo "<br>";
echo "Ortalama=",ortalama(60,90,30); // hatasız çalışacaktır.
?>
```

Eğer 3. not gönderilmezse "yok" olarak kabul edilir ve iki notun ortalaması, değilse 3. not gönderilmiş demektir. Bu durumda da üç notun ortalaması hesaplanarak gönderiliyor.

Örnek: Aldığı metni aldığı renkte ekrana yazdıralım. Eğer renk belirtilmezse siyah renkte yazdırılsın.

```php
<?php
function yaz($metin,$renk="#000000"){
  echo "<font color=\"$renk\">$metin</font>";
}

echo yaz("Bir musibet bin nasihatttan iyidir!","blue");
echo "<br>";
echo yaz("Çalışan demir ışıldar!"); // renk gönderilmediğinden varsayılan olarak siyah yazılır
?>
```

Örnek: Dairenin alanını hesaplayan fonksiyonu yazalım. Eğer pi değeri belirtilmezse 3.14 kabul edilsin.

```php
<?php
function daire($r,$pi="3.14"){
  $sonuc=$pi*$r*$r;
  return $sonuc;
}

echo "Alan=",daire(4,3); // ekrana Alan=48 yazar. $pi=3 kullanıldı
echo "<br>";
echo "Alan=",daire(4); // ekrana Alan=50.24 yazar. $pi=3.14 kullanıldı
?>
```

Örnek: Aldığı sayının belirtilen yüzdesini hesaplayan fonksiyonu tanımlayalım. Eğer yüzdesi belirtilmezse sayının kendisi gönderilsin.

```php
<?php
function yuzdesi($sayi,$oran="100"){
  return ($sayi*$oran)/100;
}

echo "60'ın %12'si = ",yuzdesi(60,12);
echo "<br>";
echo "70'in %100'ü = ",yuzdesi(70);
?>
```

Örnek: Aldığı sayının aldığı üssünü bulan fonksiyonu tanımlayalım. Eğer üs belirtilmezse karesini hesaplasın.

```php
<?php
function us_al($sayi,$üs=2){
  $sonuc=1;
  for($i=1;$i<=$üs;$i++)
    $sonuc*=$sayi;
  return $sonuc;
}

echo "5<sup>3</sup>=",us_al(5,3);
echo "<br>";
echo "4<sup>5</sup>=",us_al(4,5);
echo "<br>";
echo "4<sup>2</sup>=",us_al(4); // üs 2 kabul edildi
?>
```

Örnek: Aldığı notun 5 üzerinden karşılığını sayı olarak yada yazı olarak gönderen fonksiyonu tanımlayalım.

```php
<?php
function karne($notu,$bicimi="sayi"){
  $sayi=array(0,1,2,3,4,5);
  $yazi=array("Başarısız","Kalır","Geçer","Orta","İyi","Pekiyi");
  switch ($notu){
    case ($notu>=85 and $notu<=100): $sonuc=($bicimi=="sayi")?$sayi[5]:$yazi[5]; break;
    case ($notu>=70 and $notu<=84): $sonuc=($bicimi=="sayi")?$sayi[4]:$yazi[4]; break;
    case ($notu>=55 and $notu<=69): $sonuc=($bicimi=="sayi")?$sayi[3]:$yazi[3]; break;
    case ($notu>=45 and $notu<=54): $sonuc=($bicimi=="sayi")?$sayi[2]:$yazi[2]; break;
    case ($notu>=25 and $notu<=44): $sonuc=($bicimi=="sayi")?$sayi[1]:$yazi[1]; break;
    case ($notu>=0 and $notu<=24): $sonuc=($bicimi=="sayi")?$sayi[0]:$yazi[0]; break;
    default: $sonuc="Hata: Geçersiz not!";
  }
  return $sonuc;
}

echo "Karne notunuz = ".karne(62)." ( ".karne(62,"yazi")." ) ";
echo "<br>";
echo "Karne notunuz = ".karne(-5)." ( ".karne(-5,"yazi")." ) ";
?>
```

Yukarıdaki örneğimizde notun biçimi belirtilmezse sayı karşılığı, belirtilirse yazı karşılığı gönderilir.

Global Değişkenli Fonksiyonlar

Bir fonksiyon içinde tanımlı olan değişkenler sadece o fonksiyon içinde geçerlidir. Dolayısıyla fonksiyon içindeki değişkenleri fonksiyon dışında kullanamazsınız. Bunu yapabilmek için return ile kullanmak istediğimiz değişkenleri göndeririz. Aynı şekilde fonksiyon dışındaki değişkenleri de fonksiyon içinde kullanamazsınız. Eğer bir değişkeni global olarak tanımlarsanız o değişkeni sayfa içerisindeki her yerde kullanabilirsiniz. Aşağıdaki örneği inceleyiniz.

```php
<?php
$sayi=15;

function iki_kat($sayi){
  $sayi=$sayi*2; // $sayi değişkeninin değeri iki kat arttırılıyor
  return $sayi;
}

echo iki_kat($sayi); // ekrana 30 yazar.
echo "<br>";
echo $sayi; // ekrana 15 yazar.
?>
```

$sayi değişkeninin değeri iki kat arttırıldığı halde echo $sayi; komutu ekrana 15 yazar. Çünkü fonksiyon içindeki $sayi değişkeni ile fonksiyon dışındaki $sayi değişkeni farklıdır. İsimleri aynı olsa dahi tanımlandığı yerler itibariyle farklı olduklarından iki ayrı değişken olarak değerlendirilirler. Aynı örneği global değişken tanımlamasını kullanarak yapalım.

```php
<?php
$sayi=15;

function iki_kat($sayi){
   global $sayi;
   $sayi=$sayi*2; // $sayi değişkeninin değeri iki kat arttırılıyor
   return $sayi;
}

echo iki_kat($sayi); // ekrana 30 yazar.
echo "<br>";
echo $sayi; // ekrana 30 yazar.
?>
```

Görüldüğü gibi $sayi değişkeni global olarak tanımlanmıştır. Bundan dolayı fonksiyon bu değişkeni tanır değerini 2 kat arttırmaktadır. Dolayısıyla fonksiyon dışındaki $sayi değişkeni ile fonksiyon içindeki $sayi değişkeni aynı değişkendir.

Örnek: Aldığı sayıları sayfada tanımlı olan $toplam değişkenine ekleyen fonksiyonu tanımlayalım.

```php
<?php
$toplam=30;
echo "Toplam=",$toplam; // Toplam=30
echo "<br>";

function ekle($sayi1,$sayi2){
   global $toplam;
   $toplam+=$sayi1+$sayi2;
}

ekle(10,15);
echo "Toplam=",$toplam; // Toplam=55
?>
```

Örnek: Aldığı kullanıcı adı ve şifre bilgilerini tanımlı değerlerle karşılaştıran fonksiyonu tanımlayalım.

```php
<?php
$kullanici_adi="muyo";
$sifre="muric";

function giris($user,$password){
  global $kullanici_adi;
  global $sifre;
  if ($user==$kullanici_adi and $password==$sifre)
    $login=true;
  else
    $login=false;
  return $login;
}

if (giris("hayr","h1845")==true)
  echo "Giriş başarılı";
else
  echo "Hatalı kullanıcı yada şifre!";

echo "<br>";

if (giris("muyo","muric")==true)
  echo "Giriş başarılı";
else
  echo "Hatalı kullanıcı yada şifre!";
?>
```

Örnek: 1 ile 100 arasında,10 dan az sayıda rastgele ürettiği sayıları $sayılar dizisine ekleyen fonksiyonu tanımlayalım.

```php
<?php
$sayilar=array();

function uret(){
  global $sayac;
  global $sayilar;
  $tane=rand(1,10); // üretilecek sayı sayısı
  for($i=1;$i<=$tane;$i++)
    $sayilar[]=rand(1,100); // üretilen sayı diziye ekleniyor.
}

function yaz(){
  global $sayilar;
  foreach($sayilar as $indis=>$deger)
    echo $indis+1,") ",$deger,"<br>";
}
uret();
yaz();

echo "<hr>";

uret(); // mevcut değerlere ekleme yapıyor
yaz();
?>
```

Örneğin ekran çıktısında çizginin üstündeki değerlerin çizginin altında da olduğuna dikkat ediniz. Bu da mevcut değerlere ekleme yapıldığını gösternektedir. Çünkü $sayilar dizisi global olarak fonksiyonlarda kullanılmıştır. Yani her iki fonksiyon da aynı dizi üzerinde değişiklik yapmaktadır.

Statik (static) Değişkenli Fonksiyonlar

Bir fonksiyonun çalışması bittiğinde içerisinde tanımlı olan değişkenler yok edilir yani ölürler. Fonksiyonun bir sonraki kullanımında tanımlı olan değişkenlerin eski değerleri kaybolmuş olur. Eğer fonksiyonun ikinci kez ya da daha sonraki kullanımlarında içerisindeki değişkenlerin eski değerlerinin kaybolmasını istemiyorsanız o değişkenleri statik (static) olarak tanımlamanız gerekmektedir. Bu sayede değişkenlerin eski değerlerine ulaşabilirsiniz. Aşağıdaki örneği inceleyiniz.

```
1   <?php
2   function say(){
3     $sayac=0;
4     $sayac++;
5     return $sayac;
6   }
7
8   echo "Sayaç=",say(); // ekrana Sayaç=1 yazar
9   echo "<br>";
10  echo "Sayaç=",say(); // ekrana yine Sayaç=1 yazar
11  echo "<br>";
12  echo "Sayaç=",say(); // ekrana yine Sayaç=1 yazar
13  ?>
```

Aynı örneği statik değişken kullanarak yapalım.

```
1   <?php
2   function say(){
3     static $sayac=0;
4     $sayac++;
5     return $sayac;
6   }
7
8   echo "Sayaç=",say(); // ekrana Sayaç=1 yazar
9   echo "<br>";
10  echo "Sayaç=",say(); // ekrana Sayaç=2 yazar
11  echo "<br>";
12  echo "Sayaç=",say(); // ekrana Sayaç=3 yazar
13  echo "<br>";
14  ?>
```

Görüldüğü gibi fonksiyon çağırıldığında $sayac değişkeninin eski değeri kaybolmayarak tekrar tekrar kullanılıyor.

Örnek: Çağırıldığında diziye, dizide olmayan 1 ile 100 arasındaki bir sayıyı ekleyen fonksiyonu tanımlayalım.

```php
<?php
function yaz($dizi){
  foreach($dizi as $deger)
    echo $deger," ";
}

function ekle(){
  static $sayilar=array();
  $varmi=true; // döngüye girmek için kullandık
  while($varmi==true){
    $varmi=false;
    $sayi=rand(1,100);
    foreach($sayilar as $deger)
      if ($sayi==$deger)
        $varmi=true;
  }
  $sayilar[]=$sayi;
  return $sayilar;
}

// ekle() fonksiyonu ile diziye bir sayı ekleniyor ve dönen dizi yaz() fonksiyonuna veriliyor
// her defasında diziye dizide olmayan bir sayı eklendiğine dikkat ediniz
// dizi static olduğundan eski değerleri korunuyor
yaz(ekle());
echo "<hr>";

yaz(ekle());
echo "<hr>";

yaz(ekle());
echo "<hr>";

yaz(ekle());
echo "<hr>";

yaz(ekle());
echo "<hr>";

yaz(ekle());
echo "<hr>";
?>
```

Örnek: Aldığı metni $mesaj değişkeninin eski değerine ekleyen fonksiyonu tanımlayalım.

```php
<?php
function hata($metin){
  static $mesaj="<b><u>Oluşan Hatalar:</u></b><br>";
  $mesaj.=$metin."<br>";
  return $mesaj;
}

$tam_metin=hata("Veritabanı bağlantısı yapılamadı!");
$tam_metin=hata("Tabloda ilgili kayıtlara ulaşılamadı!");
$tam_metin=hata("Listeleme yapılamadı!");
// tüm metinler $mesaj değişkenine eklendi
echo $tam_metin;
?>
```

Örnek: Aldığı dizinin değerlerini $diziler adındaki diziye ekleyen fonksiyonu tanımlayalım.

```php
<?php
function yaz($dizi){
  foreach($dizi as $deger)
    echo $deger," ";
}

function ekle($dizi){
  static $diziler=array();
  foreach($dizi as $deger)
    $diziler[]=$deger;
  return $diziler;
}

$dizi1=array(5,4,9);
$dizi2=array(1,3,11,14);
$dizi3=array(0,6,7,2,17);

$tumu=ekle($dizi1);
yaz($tumu);
echo "<hr>";

$tumu=ekle($dizi2);
yaz($tumu);
echo "<hr>";

$tumu=ekle($dizi3);
yaz($tumu);
echo "<hr>";
?>
```

PHP'de Kullanılan Hazır Fonksiyonlar

Dizi Fonksiyonları

count()

in_array()

array_search()

sort()

rsort()

asort()

arsort()

array_merge()

array_keys()

array_push()

1) count($dizi)

Dizinin eleman sayısını verir. Bu fonksiyondan dönen değeri başka bir fonksiyona yada değişkene atamak gerekir. Aşağıdaki örneği inceleyiniz.

```
1  <?php
2  $takim=array("Esra","Gülizar","Büşra","Cangül","Ebru","Seda");
3  echo "Takımdaki kişi sayısı: ",count($takim);
4  // ekrana Takımdaki kişi sayısı: 6 yazar
5  ?>
```

Örnek: Dizideki sayıların ortalamasını bulalım.

```
1  <?php
2  $sayilar=array(3,12,17,4,24);
3  $toplam=0;
4  for ($i=0;$i<=count($sayilar)-1;$i++)
5     $toplam+=$sayilar[$i];
6  $ortalama=$toplam/count($sayilar);
7  echo "Ortalama=",$ortalama; // Ortalama=12
8  ?>
```

2) in_array($deger, $dizi)

Bir değerin dizide olup olmadığını bulur. Varsa true, yoksa false değerini döndürür.

```php
<?php
$guzergah=array("İstanbul","Kocaeli","Sakarya","Düzce","Bolu","Ankara");
$varmi=in_array("Bolu",$guzergah);
if($varmi)
  echo "Ankara'ya giderken Bolu'dan geçilir.";
else
  echo "Ankara istikametinde Bolu yoktur.";
?>
```

Örnek: 1 ile 20 arasında birbirinden farklı rastgele 5 sayıdan oluşan bir dizi üretelim.

```php
<?php
$sayilar=array();

for($i=1;$i<=5;$i++){
  do{
    $sayi=rand(1,20);
  }while(in_array($sayi,$sayilar)==true);
  $sayilar[]=$sayi;
}

foreach($sayilar as $deger)
  echo $deger." ";
?>
```

3) array_search($deger, $dizi)

Belirtilen değeri dizide arar. Bulursa değerin indisini gönderir. Yoksa false döner.

```php
<?php
$kadro=array("Patron"=>"Ali","Müdür"=>"Emin","Müd. Yrd."=>"Kaan","Hizmetli"=>"Emre");
$adi="Emin";
$varmi=array_search($adi,$kadro);
if($varmi)
  echo "Adı: ".$adi."<br>Konumu: ".$varmi;
else
  echo $adi." kadroda yoktur";
?>
```

4) sort($dizi)

Verilen diziyi sıralar. İndisler sıralanmayacağından sıralama sonucu, indislere karşılık gelen degerler değişir.

```php
<?php
$meyveler=array("elma","ayva","portakal","mandalina","muz");
foreach($meyveler as $indis=>$deger)
  echo $indis."=>".$deger."<br>";

echo "<hr>";

sort($meyveler);
foreach($meyveler as $indis=>$deger)
  echo $indis."=>".$deger."<br>";
?>
```

Örnek: Sayı bulunan bir diziyi sıralayalım.

```php
<?php
$sayilar=array(34,65,12,45,124,656,2,3446,3);
foreach($sayilar as $indis=>$deger)
  echo $indis."=>".$deger."<br>";

echo "<hr>";

sort($sayilar);
foreach($sayilar as $indis=>$deger)
  echo $indis."=>".$deger."<br>";
?>
```

5) rsort($dizi)

Diziyi tersten (reverse) sıralamak için kullanılır. sort() fonksiyonu diziyi A..Z şeklinde sıralarken, rsort() Z..A şeklinde sıralar. Yine burada da indisler sıralanmayacağında indislere karşılık gelen değerler değişecektir.

```php
<?php
$meyveler=array("elma","ayva","portakal","mandalina","muz");
print_r($meyveler); // print_r dizileri indisleriyle beraber ekrana yazar.
echo "<hr>";

rsort($meyveler);
print_r($meyveler);
?>
```

Örnek:

```php
<?php
$sayilar=array(34,65,12,45,124,656,2,3446,3);
print_r($sayilar); // print_r dizileri indisleriyle beraber ekrana yazar.

echo "<hr>";

rsort($sayilar);
print_r($sayilar);
?>
```

6) asort($dizi)

Dizideki değerleri artan bir şekilde sıralar. Sıralama sonucunda orijinal indisler korunur. Yani sıralamadan önceki değerlerin indisleri ile sıralamadan sonraki değerlerin indisleri aynı olur. Değerlerle beraber indisler de yer değiştirir. Unutmayın ki sıralama değere göre yapılır.

```
<?php
$hayvanat=array("deve","köpek","ceylan","aslan","zebra");
print_r($hayvanat); // sıralamadan önceki indislere dikkat ediniz.
// [0]=>deve [1]=>köpek [2]=>ceylan [3]=>aslan [4]=>zebra

echo "<hr>";
asort($hayvanat);
print_r($hayvanat); // sıralamadan sonraki indislere dikkat!
// [3]=>aslan [2]=>ceylan [0]=>deve [1]=>köpek [4]=>zebra
?>
```

Örnek: Öğrenci numaralarına göre verilen notları sıralayalım.

```
<?php
$geometri=array(165=>45,850=>32,141=>74,295=>95,208=>51,302=>17);
print_r($geometri);

echo "<hr>";
asort($geometri);
print_r($geometri);
// [302]=>17 [850]=>32 [165]=>45 [208]=>51 [141]=>74 [295]=>95
?>
```

Bu örnekte öğrencilerin notları küçükten büyüğe doğru sıralanırken indis olarak kullanılan öğrenci numaraları da aynı değeri işaret ederek notların karışması önlenmiş oluyor.

7) arsort($dizi)

Bu fonksiyon dizi değerlerini azalan sıralamaktadır. Sıralama sonucunda asort() fonksiyonunda olduğu gibi orijinal indisler korunmaktadır. Sıralamadan önceki indislerin işaret ettiği değerler ile sıralamadan sonradaki işarete ettiği değerler aynıdır. Yani değerlerle beraber indisler de yer değiştirir.

```
<?php
$secenekler=array("a"=>"RAM","b"=>"USB Disk","c"=>"Harddisk","d"=>"L1 Cache");
arsort($secenekler);
print_r($secenekler);
// [b]=>USB Disk [a]=>RAM [d]=>L1 Cache [c]=>Harddisk
// sıralama sonrası değerlerin azalan sıralandığına dikkat ediniz.
// Aynı zamanda indislerin korunduğuna da dikkat ediniz.
?>
```

8) array_merge($dizi1,$dizi2,$dizi3,...)

İki veya daha fazla diziyi birleştirerek tek bir dizi oluşturur. Bu fonksiyondan geriye birleştirilmiş bir dizi döner. Dolayısıyla dönen değeri bir başka değişkene atamamız gerekmetedir. Aşağıdaki örneği inceleyiniz.

Örneğimizde farklı sınıflardan seçilen öğrenciler ile oluşturulan bir basketbol takımının öğrencilerini saklayan bir dizi oluşturmaya çalışacağız.

```
1   <?php
2   $bilisim10a=array("emre","kaan","hüseyin");
3   $bilisim10b=array("hasan","mehmet");
4   $anadolubilisim=array("can","mert","raşit");
5   $basketboltakimi=array_merge($bilisim10a,$bilisim10b,$anadolubilisim);
6
7   echo "Okulumuzun Basketbol Takımı<hr>";
8   foreach($basketboltakimi as $indis=>$deger)
9     echo $indis+1,"-",$deger,"<br>";
10  ?>
```

Örnek: Rasgele üretilmiş birbirinden faklı sayılardan oluşan 3 faklı diziyi tek bir dizide birleştirelim. Bunları üç farklı başlık altında seçilmiş olan farklı sorular olduğunu düşünebilirsiniz. Bu soruları birleştirerek bir sınav hazırlandığını da düşünebilirsiniz.

```
1   <?php
2   function sec($min,$mak,$adet){
3     $secim=array();
4     for ($i=1;$i<=$adet;$i++){
5       while(in_array($sayi=rand($min,$mak),$secim))
6         $sayi=rand($min,$mak);
7       $secim[]=$sayi;
8     }
9     return $secim;
10  }
11
12  function yaz($dizi,$baslik){
13    echo $baslik,": ";
14    foreach($dizi as $deger)
15      echo $deger," ";
16    echo "<br>";
17  }
18
19  $mat=sec(1,100,15);
20  $geo=sec(101,200,5);
21  $tur=sec(201,300,20);
22
23  yaz($mat,"Matematik");
24  yaz($geo,"Geometri");
25  yaz($tur,"Türkçe");
26
27  echo "<hr>";
```

```
28  $sinav=array_merge($mat,$geo,$tur);
29  sort($sinav);
30  yaz($sinav,"Sınavda soruları");
31  ?>
```

9) array_keys($dizi)

Dizide kullanılan indisleri bir dizi olarak gönderir. Başka bir ifadeyle dizideki indisleri kullanarak bir dizi oluşturur ve gönderir.

Örneğimizde Takıma seçilen öğrencilerin numaralarını alıp ekrana yazalım.

```
1  <?php
2  $takim=array(165=>"Can",84=>"Onur",98=>"Aziz",174=>"Şaban",61=>"Harun");
3  $ogrenciler=array_keys($takim); // dizinin indislerinden bir dizi oluşturuluyor
4
5  echo "Takıma seçilen öğrencilerin numaraları: ";
6  foreach($ogrenciler as $deger)
7     echo $deger," ";
8  ?>
```

10) array_push($dizi,$deger1,$deger2,$deger3...)

Belirtilen dizinin sonuna bir veya daha fazla değer eklemek için kullanılır. Örneğimizde yolcu listesine 3 kişi daha ekleyelim.

```
1   <?php
2   $yolcu_listesi=array("Elif","Sena","Ufuk","Sercan","Merve");
3   echo "<u>Eski Yolcu Listesi</u>:<br>";
4   foreach($yolcu_listesi as $deger)
5      echo $deger,"<br>";
6
7   echo "<hr>";
8   array_push($yolcu_listesi,"Serkan","Duygu","Haşim");
9   echo "<u>Yeni Yolcu Listesi</u>:<br>";
10  foreach($yolcu_listesi as $deger)
11     echo $deger,"<br>";
12  ?>
```

Matematiksel Fonksiyonlar

- pow()
- sqrt()
- abs()
- base_convert()
- fmod()
- round()
- floor()

ceil()

rand()

deg2rad()

sin()

cos()

tan()

pi()

exp()

1) pow($x,$y)

Üslü sayıları hesaplamak için kullanılır. Burada $x üzeri $y işleminin sonucunu verir. Bu fonksiyondan dönen değer bir başka değişkene atanmalı yada kullanılmak üzere birbaşka fonksiyona verilmelidir.

```
1   <?php
2   $x=3;
3   $y=4;
4   $sonuc=pow($x,$y);
5   echo "$x<sup>$y</sup>=$sonuc<br>";
6
7   echo "3<sup>2</sup>=",pow(3,2),"<br>";
8   echo "6<sup>2</sup>=",pow(6,2),"<br>";
9   echo "4<sup>3</sup>=",pow(4,3),"<br>";
10  echo "5<sup>4</sup>=",pow(5,4),"<br>";
11  echo "81<sup>0.5</sup>=",pow(81,0.5),"<br>";
12  echo "8<sup>4.5</sup>=",pow(8,4.5),"<br>";
13  ?>
```

2) sqrt($x)

Belirtilen $x sayının karekökünü gönderir. Aynı işlemi yukarıdaki pow() fonksiyonuna üs olarak 0.5 vererek de yapabilirsiniz. Bilindiği üzere köklü sayılar aslında üssü 1/2 yani 0.5 olan sayılardır. Aşağıdaki örnekleri inceleyiniz.

```
1   <?php
2   $x=25;
3   $sonuc=sqrt($x);
4   echo "$x sayısının karekökü=$sonuc <br>";
5
6   echo "64'ün karekökü=",sqrt(64),"<br>";
7   echo "81'in karekökü=",sqrt(81),"<br>";
8   echo "144'ün karekökü=",sqrt(144),"<br>";
9   echo "16'nın karekökü=",sqrt(16),"<br>";
10  echo "2'nin karekökü=",sqrt(2),"<br>";
11  ?>
```

3) abs($x)

Verilen $x sayının mutlak değerini verir. Bilindiği üzere mutlak değer, sayının sayı ekseninde sıfıra olan uzaklığıdır. Uzaklık negatif olamayacağından mutlak değerin sonucu her zaman verilen sayının pozitif olanıdır.

```
<?php
$x=-5;
$y=5;
echo "$x sayısının mutlak değeri |$x|=",abs($x),"<br>";
echo "$y sayısının mutlak değeri |$y|=",abs($y),"<br>";
echo "|$x-$y|=",abs($x-$y),"<br>";

echo "|-3|=",abs(-3),"<br>";
echo "|12|=",abs(12),"<br>";
echo "|-6.1|=",abs(-6.1),"<br>";
echo "|12.4|=",abs(12.4),"<br>";
echo "|22|=",abs(22),"<br>";
?>
```

4) base_convert($x,$taban1,$taban2)

Verilen $x sayısını belirtilen taban1 sayı sisteminden yine belirtilen taban2 sayı sistemine dönüştürür. Burada verilen sayının taban1 sisteminde olmasına dikkat ediniz. Aşağıdaki örnekleri inceleyiniz.

```
<?php
echo "Onluk sistemdeki 27 sayısının ikilik sistemdeki karşılığı ",base_convert(27,10,2),"
'dir.<br>";
echo "(27)<sub>10</sub>=(",base_convert(27,10,2),")<sub>2</sub><br>";
echo "(101101)<sub>2</sub>=(",base_convert(101101,2,10),")<sub>10</sub><br>";
echo "(425)<sub>8</sub>=(",base_convert(425,8,2),")<sub>2</sub><br>";
echo "(8a4b)<sub>16</sub>=(",base_convert('8a4b',16,10),")<sub>10</sub><br>";
echo "(A37334)<sub>16</sub>=(",base_convert('A37334',16,2),")<sub>2</sub><br>";
echo "(10101110011)<sub>2</sub>=(",base_convert(10101110011,2,8),")<sub>8</sub><br>";
echo "(36213)<sub>4</sub>=(",base_convert(36213,4,10),")<sub>10</sub><br>";
?>
```

5) fmod($x,$y)

Verilen $x sayısının $y sayısına bölümünden kalanı vermektedir. % operatörü ile aynı işlevi görmektedir. Örneğin 15 sayısının 4'e bölümünden kalan 3'dür.

```php
<?php
$x=15;
$y=4;
$kalan=fmod($x,$y);
echo "$x sayısının $y sayısına bölümünden kalan = $kalan <br>";
echo "$x%$y=",$x%$y,"<br>";
echo "54%8=",fmod(54,8),"<br>";
echo "54%8=",54%8,"<br>";

echo "32%4=",fmod(32,4),"<br>";
echo "17%10=",fmod(17,10),"<br>";
echo "49%5=",fmod(49,5),"<br>";
echo "$x sayısı ",(fmod($x,2)==0)?"çifttir":"tektir","<br>";
echo "$y sayısı ",(fmod($y,2)==0)?"çifttir":"tektir","<br>";
?>
```

6) round($x,$ondalik)

Verilen $x sayısını belirtilen sayıda ondalik kalacak şekilde en yakın sayıya yuvarlar. Eğer ondalık belirtilmezse en yakın tam sayıya yuvarlar.

```php
<?php
$x=14.354;
$sonuc=round($x,1); // bir tane ondalık kalacak şekilde yuvarla
echo "Sonuc=$sonuc <br>";

$sonuc=round($x,2); // iki tane ondalık kalacak şekilde yuvarla
echo "Sonuc=$sonuc <br>";

$sonuc=round($x); // hiç ondalık kalmayacak şekilde yuvarla
echo "Sonuc=$sonuc <br>";

echo round(14.68),"<br>"; // ekrana 15 yazar
echo round(14.12),"<br>"; // ekrana 14 yazar
echo round(14.68,1),"<br>"; // ekrana 14.7 yazar
echo round(14.19,1),"<br>"; // ekrana 14.2 yazar
echo round(-14.6),"<br>"; // ekrana -15 yazar
?>
```

7) floor($x)

Verilen $x sayısını her zaman aşağıya tam sayı olacak şekilde yuvarlar. Dikkat ediniz: Bu fonksiyon yakın olan sayıya değil her zaman aşağıya yuvarlar. Yani kendisinden küçük en yakın tam sayıya yuvarlar.

```php
<?php
$x=14.354;
$sonuc=floor($x);
echo "Sonuc=$sonuc <br>";// ekrana Sonuc=14 yazar.

echo floor(14.68),"<br>"; // ekrana 14 yazar
echo floor(14.12),"<br>"; // ekrana 14 yazar
echo floor(14.99),"<br>"; // ekrana 14 yazar
echo floor(27.041),"<br>"; // ekrana 27 yazar
echo floor(-14.9),"<br>"; // ekrana -15 yazar
echo floor(-14.1),"<br>"; // ekrana -15 yazar
?>
```

8) ceil($x)

Verilen $x sayısını her zaman yukarıya tam sayı olacak şekilde yuvarlar. Başka bir ifadeyle kendisinden büyük en yakın tam sayıya yuvarlar.

```php
<?php
$x=14.354;
$sonuc=ceil($x);
echo "Sonuc=$sonuc <br>";// ekrana Sonuc=15 yazar.

echo ceil(14.68),"<br>"; // ekrana 15 yazar
echo ceil(14.12),"<br>"; // ekrana 15 yazar
echo ceil(14.99),"<br>"; // ekrana 15 yazar
echo ceil(27.041),"<br>"; // ekrana 28 yazar
echo ceil(-14.9),"<br>"; // ekrana -14 yazar
echo ceil(-14.1),"<br>"; // ekrana -14 yazar
?>
```

9) rand(min,mak)

Verilen aralıkta rastgele bir tamsayı üretir.

```php
<?php
$sayi=rand(1,10);
echo $sayi,"<br>";
echo rand(20,25),"<br>";
echo rand(160,200),"<br>";

echo "<hr>";
$sayilar=array();
$sayilar[]=rand(1,100);
$sayilar[]=rand(90,100);
$sayilar[]=rand(1,5);
$sayilar[]=rand(1000,10000);
$sayilar[]=rand(500,510);
foreach($sayilar as $deger)
   echo $deger," ";
?>
```

10) deg2rad($x)

Derece cinsinden verilen $x açısının radyan karşılığını verir. Aşağıda anlatılan sin(), cos() ve tan() gibi fonksiyonlar açıyı radyan cinsinden istemektedir.

```
<?php
$x=30;
$radx=deg2rad($x);
echo "$x açısının radyan karşılığı = ",$radx,"<br>";

echo "90 derecenin radyan karşılığı = ",deg2rad(90),"<br>";
echo "60 derecenin radyan karşılığı = ",deg2rad(60),"<br>";
echo "180 derecenin radyan karşılığı = ",deg2rad(180),"<br>";
echo "360 derecenin radyan karşılığı = ",deg2rad(360),"<br>";
?>
```

11) sin($x), cos($x), tan($x)

Radyan cinsinden verilen $x açısının sırasıyla sinüs, kosinüs ve tanjant değerlerini verir. Dolayısıyla derece cinsindeki açıyı önce radyana çevirmek gerekmektedir.

```
<?php
$x=30;
$radx=deg2rad($x); // derece radyana çevriliyor
echo "sin($x)=",sin($radx),"<br>";
echo "cos($x)=",cos($radx),"<br>";
echo "tan($x)=",tan($radx),"<br>";

$x=60;
$radx=deg2rad($x); // derece radyana çevriliyor
echo "sin($x)=",sin($radx),"<br>";
echo "cos($x)=",cos($radx),"<br>";
echo "tan($x)=",tan($radx),"<br>";
?>
```

12) pi()

Bu fonksiyon pi sayısını göndermektedir. Yaklaşık olarak 3 ya da 3.14 olarak kullandığımız pi sayısının yerine, pi() fonksiyonunun gönderdiği değeri kullanmak daha doğru sonuç verecektir.

```
<?php
$pi=pi();
echo $pi,"<br>";
echo pi(),"<br>";

echo "yarıçapı 3cm olan dairenin çevresi = ",2*pi()*3," cm'dir.<br>";
echo "yarıçapı 3cm olan dairenin alanı = ",pi()*pow(3,2)," cm<sup>2</sup> 'dir.<br>";
?>
```

13) exp($x)

Matematikte yaklaşık 2.718281828459 olarak bilinen e sayısının kuvvetlerini hesaplar ve gönderir. Buradaki $x sayısı e sayısının üssü olmaktadır.

```
1  <?php
2  echo "e=",exp(1),"<br>";
3  echo "e<sup>2</sup>=",exp(2),"<br>";
4  echo "e<sup>3</sup>=",exp(3),"<br>";
5  echo "e<sup>4</sup>=",exp(4),"<br>";
6  ?>
```

String Fonksiyonları

- strlen()
- chr()
- explode()
- implode()
- str_split()
- ltrim()
- rtrim()
- trim()
- substr()
- strtolower()
- strtoupper()
- ucfirst()
- ucwords()
- str_replace()
- nl2br()
- md5()
- sha1()

1) strlen($metin)

Verilen metnin karakter sayısını yani uzunluğunu verir.

```php
<?php
$metin="İlmin bereketi güzel ameldir.";
$uzunluk=strlen($metin);
echo "$metin > cümlesinin uzunluğu = $uzunluk";
?>
```

2) chr($sayi)

Verilen ascii koda karşılık gelen karakteri verir. Bilindiği üzere klavye üzerindeki her karakterin 255'e kadar olan bir ascii kodu vardır.

```php
<?php
echo "Ascii kodu 65 olan karakter ",chr(65)," 'dır.<br>";
echo "<u>Aşağıdaki kelime ascii kodlar ile yazılmıştır.</u><br>";
echo chr(109),chr(101),chr(114),chr(104),chr(97),chr(98),chr(97),"<br>";

echo "<table border='1'>";
echo "<tr>";
echo "<th>Ascii Kodu</th>";
echo "<th>Karekter</th>";
echo "</tr>";
for ($i=0;$i<=255;$i++){
   echo "<tr>";
   echo "<td>",$i,"</th>";
   echo "<td>",chr($i),"  </td>";
   echo "</tr>";
}
echo "</table>";
?>
```

3) explode($ayirici,$metin)

Verilen metni belirtilen ayırıcı karaktere göre parçalar ve bir dizi olarak gönderir. Örneğin bir cümledeki kelimeleri bir diziye aktaralım. Buradaki ayırıcımız boşluk karakteri olacaktır.

```php
<?php
$metin="Kişi bilmediğinin düşmanıdır.";
$dizi=explode(" ",$metin); // cümlemiz boşluklardan bölünecek
print_r($dizi); // her kelime dizide bir değer olacaktır
?>
```

Örnek: Veritabanına 2010-04-25 şeklinde kaydedilen tarihi ekrana 25 Nisan 2010 şeklinde yazdıralım.

```php
<?php
$tarih="2010-04-25";
// - ile ayrılan her sayı diziye atanacak
$sayilar=explode("-",$tarih); // 0=>2010 1=>04 2=>25
$aylar=array("Ocak","Şubat","Mart","Nisan","Mayıs","Haziran","Temmuz","Ağustos","Eylül","
Ekim","Kasım","Aralık");
echo $sayilar[2]," ",$aylar[$sayilar[1]-1]," ",$sayilar[0];
?>
```

Örnek: Bir paragrafta kaç kelime ve kaç cümle olduğunu bulalım.

```php
<?php
$paragraf="Bir zamanlar, her şeyden sürekli şikayet eden, her gün hayatının
ne kadar berbat olduğundan yakınan bir kız vardı. Hayat, ona göre çok kötüydü
ve sürekli savaşmaktan, mücadele etmekten yorulmuştu.
Genç kızın bu yakınmaları karşısında, mesleği aşçılık olan babası ona bir hayat
dersi vermeye niyetlendi. Bir gün onu mutfağa götürdü. Üç ayrı cezveyi suyla
doldurdu ve ateşin üzerine koydu. Cezvelerdeki sular kaynamaya başlayınca,
bir cezveye bir patates, diğerine bir yumurta ve sonuncusuna da kahve
çekirdeklerini koydu. Daha sonra kızına tek kelime etmeden, beklemeye başladı.
Kızı da hiçbir şey anlamadığı bu faaliyeti seyrediyor ve sonunda karşılaşacağı
şeyi görmeyi bekliyordu. Ama o kadar sabırsızdı ki, sızlanmaya ve daha ne kadar
bekleyeceklerini sormaya başladı.";

$kelimeler=explode(" ",$paragraf);
$kelime_sayisi=count($kelimeler);
$cumleler=explode(".",$paragraf);
$cumle_sayisi=count($cumleler)-1;

echo "<hr>";
echo "$paragraf <br><br>";
echo "Yukarıdaki paragrafta $kelime_sayisi kelime ve $cumle_sayisi cümle vardır.";

echo "<hr>";
foreach($kelimeler as $indis=>$deger)
  echo $indis+1,"=>",$deger,"<br>";

echo "<hr>";
foreach($cumleler as $indis=>$deger)
  echo $indis+1,"=>",$deger,"<br>";
?>
```

4) implode($ayirici,$dizi)

Bu fonksiyon explode() fonksiyonunun aksine bir dizideki değerleri belirtilen ayırıcıyla birleştirerek biraraya getirir. Örneğimizde bir cümleyi oluşturan kelimelerin bulunduğu diziyi birleştirip cümlemizi kuralım.

```php
<?php
$kelimeler=array("Sükut","yalan","söylemekten","ve","başkalarını","çekiştirmekten","herhalde",
"evladır");
$cumle=implode(" ",$kelimeler);
echo $cumle,"!";
?>
```

Örnek: Ayları arasına – koyarak birleştirip ekrana yazdıralım.

```php
<?php
$aylar=array("Ocak","Şubat","Mart","Nisan","Mayıs","Haziran","Temmuz","Ağustos","Eylül","
Ekim","Kasım","Aralık");
$metin=implode("-",$aylar);
echo $metin;
?>
```

5) str_split($metin,$sayi)

Verilen metni verilen sayı kadar karakter gruplarına bölerek bir diziye dönüştürür. Eğer sayı belirtilmezse, metni harf harf bölerek bir diziye atar.

```php
<?php
$metin="Fazlı BÜNYAMİN";
$harfler=str_split($metin); // sayı belirtilmediğinden harf harf böler
foreach($harfler as $harf)
  echo "<h1>",$harf,"</h1>";
?>
```

Örnek: Verilen örnek bir iban kodunu dörtlü gruplara bölerek ekrana yazdıralım.

```php
<?php
$iban="TR680004600000222990022302";    //http://tr.wikipedia.org/wiki/IBANOrnek_IBAN
$grup=str_split($iban,4);
foreach($grup as $deger)
  echo $deger," ";
?>
```

6) ltrim($metin), rtrim($metin), trim($metin)

Verilen metnin sırasıyla solundaki(left-ltrim), sağındaki(right-rtrim) ve her iki tarafındaki(trim) gereksiz boşlukları temizler. Örneğin bir formdan gelen metnin soluna yada sağına gereksiz boşluklar bırakılmışsa bunların temizlenmesi gerekmektedir. Aslında bu tür basit işlemler için web sunucu meşgul edilmek istenmez. Bundan dolayı bu tür kontroller için istemci taraflı çalışan javascript kodları kullanılır.

```php
<?php
$metin1="        Zonguldak";
$metin2="Zonguldak    ";
$metin3="        Zonguldak        ";
echo $metin1,"<br>\n";
echo $metin2,"<br>\n";
echo $metin3,"<br>\n";

echo "<hr>\n"; // görünürde bir faklılık yok gibi olsa da kaynak koda dikkat ediniz.

$metin1=ltrim($metin1); // solundaki boşluklar temizlendi.
$metin2=rtrim($metin2); // sağındaki boşluklar temizlendi.
$metin3=trim($metin3); // her iki tarafındaki boşluklar temizlendi.

echo $metin1,"<br>\n";
echo $metin2,"<br>\n";
echo $metin3,"<br>\n";
?>
```

Verdiğimiz örnekte boşlukları temizlemeden önceki ve sonraki ekran çıktıları aynı görünebilir. Ancak kaynak koddaki farklılıklara dikkat ediniz. Örneğin bir kullanıcı adının başına yada sonuna boşluk bırakılarak yazılması üye girişinde hataya sebep olacaktır. Çünkü kontroller görünüre göre değil esas bilgiye göre yapılır. Bundan dolayı gereksiz boşlukların temizlenmesinde fayda vardır.

7) substr($metin,$baslangic,$uzunluk)

Bir metnin belirtilen başlangıç konumundan itibaren istenilen uzunluktaki bir parçasını gönderir. Yani metinlerin belirli bir bölümünü almak için kullanılır.

```php
<?php
$metin="Fırsat karınca yürüyüşü ile gelir, yıldırım hızı ile gider.";
$parca1=substr($metin,7,7); // karınca
$parca2=substr($metin,24,3); // ile
$parca3=substr($metin,53,6); // gider.
echo $parca1," ",$parca2," ",$parca3;
?>
```

Unutmayın ilk karakterin konumu sıfır(0)'dır.

Örnek: Bir dosyanın uzantısını bulalım.

```php
<?php
$dosyaadi="e-kitap.pdf";
$uzunluk=strlen($dosyaadi);
$uzanti=substr($dosyaadi,$uzunluk-3,3);
echo $uzanti;
?>
```

8) strtolower($metin), strtoupper($metin)

Verilen metni sırasıyla küçük harfe ve büyükharfe dönüştürür.

```
1  <?php
2  $metin="PARAYI VEREN DÜDÜĞÜ ÇALAR!";
3  echo strtolower($metin),"<br>"; // küçük harfe dönüştürülüyor
4
5  $metin="Pilavdan dönenin kaşığı kırılsın!";
6  echo strtoupper($metin);// büyük harfe dönüştürülüyor
7  ?>
```

Bu örnekte bazı türkçe karakterlerin düzgün görüntülenmediğini farketmişsinizdir. Bu sorunu çözmek için mb_strtolower() ve mb_strtoupper() fonksiyonlarını kullanabilirsiniz. Bu iki fonksiyona türkçenin karakter kodu verilerek düzgün sonuçlar alınabilir.

```
1  <?php
2  $metin="PARAYI VEREN DÜDÜĞÜ ÇALAR!";
3  echo mb_strtolower($metin,"iso-8859-9"),"<br>"; // küçük harfe dönüştürülüyor
4
5  $metin="Pilavdan dönenin kaşığı kırılsın!";
6  echo mb_strtoupper($metin,"iso-8859-9");// büyük harfe dönüştürülüyor
7  ?>
```

9) ucfirst($metin), ucwords($metin)

Bu iki fonksiyon sırasıyla verilen metnin ilk karakterini (ucfirst) ve metindeki her kelimenin ilk karakterini(ucwords) büyük harfe dönüştürür.

```
1  <?php
2  $metin="parayı veren düdüğü çalar!";
3  echo ucfirst($metin),"<br>"; // ilk karakter olan p, P yapılıyor.
4
5  $metin="Pilavdan dönenin kaşığı kırılsın!";
6  echo ucwords($metin);// her kelimenin ilk karekteri büyük harfe dönüştürülüyor
7  ?>
```

10) str_replace($kaynak,$hedef,$metin)

Verilen metindeki kaynak karakterlerin yerine hedef karakterileri koyarak değiştirme yapar. Bu fonksiyon küçük/büyük harf duyarlıdır.

```php
<?php
$metin="PC üzerindeki RAM, olmazsa olmaz bileşenlerdendir.";
$metin=str_replace("PC","Bilgisayar",$metin); // PC yerine Bilgisayar yazılıyor
$metin=str_replace("RAM","bellek",$metin); // RAM yerine bellek yazılıyor

echo $metin; // Bilgisayar üzerindeki bellek, olmazsa olmaz bileşenlerdendir.
?>
```

Örnek: Bu örneğimizde bir dizideki bir değeri başka bir değerle değiştirelim.

```php
<?php
$araclar=array("Kamyon","Otobüs","Araba","Kamyonet","Araba");
$araclar=str_replace("Araba","Otomobil",$araclar);
print_r($araclar);
?>
```

Örnek: Sınav tarihlerinin yazılı bulunduğu metindeki ingilizce günleri türkçe günlerle değiştirelim.

```php
<?php
$sinavlar="Birinci sınav: 26.04.2010 Monday, İkinci sınav: 14.05.2010 Friday, Üçüncü sınav:
09.06.2010 Wednesday";
$kaynak=array("Monday","Tuesday","Wednesday","Thursday","Friday","Saturday","Sunday");
$hedef=array("Pazartesi","Salı","Çarşamba","Perşembe","Cuma","Cumartesi","Pazar");
$sinavlar=str_replace($kaynak,$hedef,$sinavlar);
echo $sinavlar;
?>
```

Burada belirtilen metinde varsa Monday yerine Pazartesi, Tuesday yerine Salı, ... yazılacaktır. Aslında bu işlem ile birçok farklı kelime değiştirilmektedir. Aranan değerler bir dizi şeklinde, buna karşılık gelecek değerler de aynı sıralamada farklı bir dizide belirtilmiştir.

Örnek: Bir sınıftaki öğrencilerin aldığı notların yazı karşılıklarını yazdıralım. Öğrenci numaraları indis olsun.

```php
<?php
$bilisim10b=array(94=>4,184=>2,941=>5,641=>3,201=>2,745=>5);
print_r($bilisim10b);

echo "<br>";
$kaynak=array(0,1,2,3,4,5);
$hedef=array("Başarısız","Kalır","Geçer","Orta","İyi","Pekiyi");
$bilisim10b=str_replace($kaynak,$hedef,$bilisim10b);
print_r($bilisim10b);
?>
```

Bu kez üzerinde çalışılan bir metin değil bir dizidir.($bilisim10b)

11) nl2br($metin)

String ifadelerde kullanılan ve alt satıra geçmeyi sağlayan \n yerine kaynak kodda
 yazmak için kullanılır.

```
1  <?php
2  $mesaj="Hata oluştu!\nHata Kodu: 101\nHata sebeplerini ortadan kaldırıp tekrar deneyiniz.\n";
3  echo $mesaj;
4
5  echo "<hr>\n";
6  echo nl2br($mesaj);
7  ?>
```

Sayfanın kaynak koduna bakınız. \n kaynak kodda bir alt satıra geçerken sayfada bir alt satıra geçmemektedir. nl2br() fonksiyonu ile \n yerine
 yazılarak hem kaynak kodda hem de sayfada bir alt satıra geçilmektedir.

12) md5($metin), sha1($metin)

Sırasıyla aldığı metne karşılık gelen karmaşık md5 ve sha1 kodlarını verir. Genelde kaydedilecek şifreleri şifrelemek için kullanılır.

```
1   <?php
2   $parola="muyo";
3   echo "Parola: $parola<br>";
4   echo "md5 kodu: ",md5($parola),"<br>";
5   echo "sha1 kodu: ",sha1($parola),"<br>";
6
7   if(md5($parola)=="b4e13754952fd48da1b830a615082dfd")
8     echo "şifre doğru<br>";
9   else
10    echo "Şifre yanlış<br>";
11
12  if(sha1($parola)=="0c368701bf596cc80a9815ec7c9774be43a0e573")
13    echo "şifre doğru<br>";
14  else
15    echo "Şifre yanlış<br>";
16  ?>
```

PHP SMTP Sınıfı: PHPMailer

Kullanımı basit ve mail olaylarıyla alakalı istediğiniz her şeyi yapmak mümkün (dosya göndermek, birden fazla alıcı eklemek, yanıt adresini değiştirmek vs.)

Bu sınıfa şu adresten ulaşıp https://github.com/PHPMailer/PHPMailer/releases güncel sürümünü indirebilirsiniz.

Mail Göndermek

PHPMailer sınıfı ile SMTP mail göndermek için “class.phpmailer.php” ve “class.smtp.php” dosyalarını alıp aşağıdaki kod bloğunu kullanmanız yeterli. İlgili yerlere kendi sunucunuzun host, port ve kullanıcı adı şifre bilgilerinizi girin.

```
include 'class.phpmailer.php';
$mail = new PHPMailer();
$mail->IsSMTP();
$mail->SMTPAuth = true;
$mail->Host = 'smtp.sitem.com';
$mail->Port = 587;
$mail->Username = 'benim@adresim.com';
$mail->Password = 'sifre';
$mail->SetFrom($mail->Username, 'Benim Adım');
$mail->AddAddress('alici@adresi.com', 'Alıcının Adı');
$mail->CharSet = 'UTF-8';
$mail->Subject = 'Mail Başlığı';
$mail->MsgHTML('Mailin içeriği!');
if($mail->Send()) {
   echo 'Mail gönderildi!';
} else {
   echo 'Mail gönderilirken bir hata oluştu: ' . $mail->ErrorInfo;
}
```

Olay bu kadar düzenli ve basit. Eğer yukarıda da bahsettiğim gibi kendi sunucunuzu kullanmak yerine varolan Gmail hesabınız ile bu olayı kullanmak istiyorsanız yukarıdaki host ve port bölümü aşağıdaki gibi değiştirin:

```
$mail->Host = 'smtp.gmail.com';
$mail->Port = 587;
$mail->SMTPSecure = 'tls';
```

Mail ile dosya göndermek

Göndereceğiniz mailin içerisine dosya eklemek için AddAttachment metodunu kullanacağız. Örneğin bir resim göndermek için:

```
$mail->AddAttachment('img/gonderilecek_resim.jpg');
```

Birden fazla kişiye gönderme

Birden fazla alıcı eklemek için ekstra bir şeye ihtiyaç yok. Yukarıda alıcı adresi tanımladığımız AddAddress metodunu tekrar kullanabiliriz. Ama eğer CC ya da BCC eklemek istiyorsanız onun için de AddCC ya da AddBCC metodlarını kullanmanız yeterli.

```
$mail->AddAddress('baska@biri.com', 'Başkası');
$mail->AddCC('baska@iki.com', 'Diğeri');
$mail->AddBCC('baska@ucu.com', 'Öteki');
```

Yeri gelmişken CC ile BCC arasından da bahsetmek gerekirse; CC: “Carbon Copy” yani bir kopyasını da bu şekilde eklenmiş adreslere gönderir. BCC: “Blend Carbon Copy” CC ile aynı tek farkı buraya yazılan adresler diğer gönderilen kişilerde gözükmez.

Yanıt adresini değiştirme

Eğer gönderdiğiniz adrese değilde başka bir adrese cevap yazılmasını istiyorsanız AddReplyTometodu ile cevap atılmasını istediğiniz mail adresini yazmanız yeterli.

```
1       $mail->AddReplyTo('cevaplar@buraya.com', 'Cevapçı');
```

GET Metodu

<form>...</form> etiketinin **action** özelliğinde belirtilen sayfada **GET** metoduyla yani URL'ye eklenerek gönderilen değişkeleri almak için **$_GET** dizisi kullanılır.

$_GET dizisinde indis olarak değişken adları ya da form adları kullanılır.

Örneğin; **http://www.phpdefteri.com/index.php?konu=selam&mesaj=Merhaba** şeklinde GET ile gönderilen **konu** ve **mesaj** bilgilerini şu şekilde alırız.

```
<?php
$gelen_konu=$_GET["konu"];
$gelen_mesaj=$_GET["mesaj"];
?>
```

Örnek olarak basit bir uygulama yapalım.

Örnek: GET metodu ile gönderilen ad ve soyad bilgilerini alıp ekrana yazdıralım.

```
1    <!--index.php-->
2    <html>
3    <body>
4      <form name="form1" method="get" action="yaz.php">
5        Adı: <input type="text" name="adi"><br>
6        Soyadı: <input type="text" name="soyadi"><br>
7        <input type="submit" name="gönder" value="Gönder">
8      </form>
9    </body>
10   </html>
```

```
1   <!--yaz.php-->
2   <html>
3   <body>
4   <?php
5   echo "<b>Adınız:</b> ",$_GET["adi"],"<br>";
6   echo "<b>Soyadınız:</b> ",$_GET["soyadi"],"<br>";
7   ?>
8   </body>
9   </html>
```

Burada iki sayfamız vardır.

İlk sayfamız olan **index.php** sayfasında formlarımız var. Bu formlara girilen bilgilerin **yaz.php** sayfasına gönderileceği **action="yaz.php"** ile belirtilmiştir. **method="get"** ifadesiyle bilgilerin GET metodu ile gönderileceği belirtilmiştir. Unutmayınız form yapılarında çoğu zaman POST metodu kullanılır. GET metodu pek kullanılmaz. Burada sadece GET metodu ile bilgi göndermeye ve almaya örnek vermekteyiz.

Bilindiği gibi GET metodu <a>... etiketi ile oluşturulan linkler ile bilgi göndermede kullanılır.

İkinci sayfamız olan **yaz.php** sayfasında formlardan gelen bilgiler alınıyor. Daha sonra sayfaya yazdırılması için **echo** fonksiyonuna veriliyor. Bu bilgileri değişkenlere atayarak da işlem yapabilirsiniz.

Not: **index.php** ile **yaz.php** sayfalarının aynı dizinde olduğu kabul edilerek **action** kısmında **yaz.php** sayfasının yolu belirtilmemiştir. Aksi halde dosyanın tam yolu yada göreceli yolu yazılmalıdır.

Örnek: Rastgele üretilen iki sayıyı **yaz.php** sayfasına bir link ile (GET metodu ile) gönderip ekrana yazdıralım.

```
1   <!--index.php-->
2   <html>
3   <body>
4   <?php
5   $s1=rand(1,100);
6   $s2=rand(1,100);
7   echo "<b>Bulunan sayılar:</b><br>";
8   echo "Sayı 1=$s1<br>Sayı 2=$s2<br>";
9   echo "<a href='yaz.php?sayi1=$s1&sayi2=$s2'>Bu linke tıklayarak sayıları
10  gönderebilirsiniz</a>";
11  ?>
12  </body>
    </html>
```

```
1   <!--yaz.php-->
2   <html>
3   <body>
4   <?php
5   echo "<b>Sayı 1:</b> ",$_GET["sayi1"],"<br>";
6   echo "<b>Sayı 2:</b> ",$_GET["sayi2"],"<br>";
7   ?>
8   </body>
9   </html>
```

Bu örnek gerçek bir GET metoduna güzel bir örnektir.

index.php sayfasını yenileyerek her seferinde farklı sayılar gönderebilirsiniz.

Örnek: Tablodaki numaralara ve isimlere tıklandığında bu bilgileri **yaz.php** sayfasına gönderip ekrana yazdıralım.

```
<!--index.php-->
<html>
<body>
  <table border="1">
    <tr>
      <td>Numarası</td><td>Adı</td>
    </tr>
    <tr>
      <td><a href="yaz.php?no=15">15</a></td>
      <td><a href="yaz.php?ad=Ayşe">Ayşe</a></td>
    </tr>
    <tr>
      <td><a href="yaz.php?no=17">17</a></td>
      <td><a href="yaz.php?ad=Osman">Osman</a></td>
    </tr>
    <tr>
      <td><a href="yaz.php?no=29">29</a></td>
      <td><a href="yaz.php?ad=Mustafa">Mustafa</a></td>
    </tr>
</table>
</body>
</html>
```

```
<!--yaz.php-->
<html>
<body>
<?php
echo "<b>Numarası</b>: ",$_GET["no"],"<br>";
echo "<b>Adı</b>: ",$_GET["ad"],"<br>";
?>
</body>
</html>
```

Burada tıklanan linke göre sadece **no** yada **ad** gönderilmektedir. Buna göre **yaz.php** sayfasında duruma göre sadece gelen bilgiyi ekrana yazdırmak için **yaz.php**sayfası şu şekilde düzenlenebilir.

```
<!--yaz.php-->
<html>
<body>
<?php
if(@$_GET["no"]) // eğer no geldi ise
   echo "<b>Numarası</b>: ",$_GET["no"],"<br>";
elseif(@$_GET["ad"]) // eğer ad geldi ise
   echo "<b>Adı</b>: ",$_GET["ad"],"<br>";
?>
</body>
</html>
```

Burada **if(@$_GET["no"])** ifadesi; *GET metodu ile gelen **no** bilgisi varsa* anlamına gelmektedir. Başındaki @ işareti ise uyarı mesajına engel olmaktadır. Dikkat ediniz hata değil, uyarı mesajına engel olur. Eğer GET ile **no** bilgisi gelmiyorsa, gelmeyen yani olmayan bir değişkeni **if** yapısı kontrol ederken uyarı verir. Bu uyarı mesajını gizlemek için @ işaretini kullanıyoruz.

Örnek: Öğrenci bilgilerinin bulunduğu dizideki öğrenci adlarını sayfaya yazdıralım. Daha sonra bu adlara tıklandığında **yaz.php** sayfasına öğrencinin numarasını gönderip ekrana yazdıralım.

```
<!--index.php-->
<html>
<body>
<?php
$liste=array(74=>"Onur",64=>"Samet",85=>"Can",16=>"Raşit",80=>"Mert",14=>"Aziz");
foreach($liste as $indis=>$deger)
   echo "$indis <a href='yaz.php?no=$indis'>$deger</a><br>";
?>
</body>
</html>
```

```
<!--yaz.php-->
<html>
<body>
<?php
echo "<b>Gelen Numara</b>=",$_GET["no"];
?>
</body>
</html>
```

Örnek: Dizideki kişileri tablo şeklinde ekrana yazdıralım. Tabloda **sil** ve **düzelt** şeklinde tanımlanan iki link ile bu kişilerin numaralarını ve işlem adını **islem.php** sayfasına gönderip ekrana yazdıralım.

```
<!--index.php-->
<html>
<body>
<?php
$uyeler=array(14=>"Ömer",22=>"Erkin",35=>"Zekiye",41=>"Vildan",85=>"Yalçın",114=>"R
abia");
echo "<table border='1'>";
echo "<tr>";
echo "<td>Numarası</td>";
echo "<td>Adı</td>";
echo "<td colspan='2'>İşlem</td>";
echo "</tr>";
foreach($uyeler as $indis=>$deger){
   echo "<tr>";
   echo "<td>$indis</td>";
   echo "<td>$deger</td>";
   echo "<td><a href='islem.php?no=$indis&islem=duzelt'>Düzelt</a></td>";
   echo "<td><a href='islem.php?no=$indis&islem=sil'>Sil</a></td>";
```

```
    echo "</tr>";
}
echo "</table>";
?>
</body>
</html>
```

```
<!--islem.php-->
<html>
<body>
<?php
$no=$_GET["no"];
if($_GET["islem"]=="duzelt")
    echo "$no nolu kayıt düzeltilecektir.";
elseif($_GET["islem"]=="sil")
    echo "$no nolu kayıt silinecektir!";
?>
</body>
</html>
```

Örnek: Bu örneğimizde GET metodu ile sayfanın kendisine bilgi göndereceğiz. Bunun için hedef olarak bulunduğunuz sayfanın adını yazacağız.

Örneğimizde ekranda bulunan numaralardan birine tıklandığında buna karşılık gelen resmi ekranda göstereceğiz. Bu resimlerin adları bir dizide saklıdır.

```
<!--index.php-->
<html>
<body>
Kullanılan resimleri indirmek için <a
href="http://docs.google.com/uc?id=0B4VfeLzXn5MgZmU1ZjQ4MjYtMWQ0ZC00MGI3LTk
zM2MtNWVjOTQyNzE3ZDI0&export=download&hl=en_US">tıklayınız.</a><br>
<?php
$resimler=array(1=>"gok.jpg","mavi.jpg","park.jpg","sari.jpg","yesil.jpg");
foreach($resimler as $indis=>$deger)
    echo "<a href='index.php?resim_no=$indis'>$indis</a> ";

echo "<br>";
if($_GET){ // GET olayı varsa anlamındadır.
    echo "<img src='resimler/".$resimler[$_GET["resim_no"]]."'>"; // GET olayı varsa ilgili
resmi gösterir
    echo "<br><b>".$resimler[$_GET["resim_no"]]."</b>"; // resmin adını yazar
}
else
    echo "<img src='resimler/".$resimler[1]."'>"; // GET olayı yoksa ilk resmi gösterir
    echo "<br><b>".$resimler[1]."</b>"; // ilk resmin adını yazar
?>
</body>
</html>
```

Burada, sayfa ilk yüklendiğinde GET olayı olmadığından dizideki ilk resim gösteriliyor. Eğer GET olayı varsa GET ile gelen **resim_no** 'ya göre ilgili resim ve adı erkran gösteriliyor.

Buradaki **if($_GET)** ifdadesi GET olayı varsa anlamına gelmektedir.

POST Metodu

Formlardan **POST** metodu ile gönderilen bilgileri almak için **$_POST** dizisi kullanılır. Burada indis olarak formların isimleri yani name değerleri kullanılır. Konuyu bir örnekle açıklayalım.

Örnek: POST metoduyla formlardan gönderilen ad ve soyad bilgilerini alıp ekrana yazdıralım.

(Örneğimizde **Metin Kutusu + Gönder Düğmesi** kullanılmıştır.)

```
1   <!--index.php-->
2   <html>
3   <body>
4     <form name="form1" method="post" action="yaz.php">
5       Adı: <input type="text" name="adi"><br>
6       Soyadı: <input type="text" name="soyadi"><br>
7       <input type="submit" name="gönder" value="Gönder">
8     </form>
9   </body>
10  </html>
```

```
1   <!--yaz.php-->
2   <html>
3   <body>
4   <?php
5   echo "<b>Adınız:</b> ",$_POST["adi"],"<br>";
6   echo "<b>Soyadınız:</b> ",$_POST["soyadi"],"<br>";
7   ?>
8   </body>
9   </html>
```

Buradaki birinci sayfamızda (**index.php**) formlar vardır. Bu formlara girilen bilgilerin POST metoduyla yaz.php sayfasına gönderilmektedir. İkinci sayfamızda (**yaz.php**) ise bu formlardan gelen bilgiler alınarak ekrana yazılmaktadır. Bu bilgiler burada olduğu gibi doğrudan echo fonksiyonuna verilebileceği gibi değişkenlere atanarak da işlem yapılabilir.

Örnek: Formlara girilen iki sayının toplamını topla.php sayfasında ekrana yazdıralım.

(Örneğimizde **Metin Kutusu + Gönder Düğmesi** kullanılmıştır.)

```
<!--index.php-->
<html>
<body>
  <form name="form1" method="post" action="topla.php">
    Birinci Sayı: <input type="text" name="sayi1"><br>
    İkinci Sayı: <input type="text" name="sayi2"><br>
    <input type="submit" name="topla" value="topla">
  </form>
</body>
</html>
```

```
<!--topla.php-->
<html>
<body>
<?php
$s1=$_POST["sayi1"];
$s2=$_POST["sayi2"];
$toplam=$s1+$s2;
echo "Sayıların toplamı: $s1+$s2=$toplam";
?>
</body>
</html>
```

Örnek: Kullanıcının girdiği iki sayıyı yine kullanıcının açılır listeden seçtiği işleme göre **hesapla.php** sayfasında işleme alıp sonucu ekranda gösterelim.

(Örneğimizde **Metin Kutusu + Açılır Liste + Gönder Düğmesi** kullanılmıştır.)

```
<!--index.php-->
<html>
<body>
  <form name="form1" method="post" action="hesapla.php">
    Birinci Sayı: <input type="text" name="sayi1"><br>
    İkinci Sayı: <input type="text" name="sayi2"><br>
    İşleminizi seçiniz:
    <select name="islem">
      <option value="+">Topla</option>
      <option value="-">Çıkar</option>
      <option value="*">Çarp</option>
      <option value="/">Böl</option>
    </select>
    <input type="submit" name="hesapla" value="Hesapla">
  </form>
</body>
</html>
```

```
<!--hesapla.php-->
<html>
<body>
<?php
$s1=$_POST["sayi1"];
$s2=$_POST["sayi2"];
$islem=$_POST["islem"];

// Bu kısımda switch-case yapısı da kullanılabilirdi.
if($islem=="+")
   $sonuc=$s1+$s2;
elseif($islem=="-")
   $sonuc=$s1-$s2;
elseif($islem=="*")
   $sonuc=$s1*$s2;
else
   $sonuc=$s1/$s2;

echo "İşlemin Sonucu:<br>$s1$islem$s2=$sonuc";
?>
</body>
</html>
```

Örnek: Girilen sayının karesini, küpünü, karekökünü yada faktoriyelini **hesapla.php** sayfasında bulalım.

(Örneğimizde **Metin Kutusu + Radyo Buton + Gönder Düğmesi** kullanılmıştır.)

```
<!--index.php-->
<html>
<body>
   <form name="form1" method="post" action="hesapla.php">
      Sayınızı Giriniz: <input type="text" name="sayi"><br>
      İşleminizi seçiniz:<br>
      <input type="radio" name="islem" value="kare" checked>Karesi<br>
      <input type="radio" name="islem" value="kup">Küpü<br>
      <input type="radio" name="islem" value="karekok">Karekökü<br>
      <input type="radio" name="islem" value="faktoriyel">Faktoriyeli<br>
      <input type="submit" name="Hesapla" value="Hesapla">
   </form>
</body>
</html>
```

```
<!--hesapla.php-->
<html>
<body>
<?php
$sayi=$_POST["sayi"];
$islem=$_POST["islem"];

if($islem=="kare"){
  $sonuc=pow($sayi,2);
  echo "$sayi<sup>2</sup>=$sonuc";
}
elseif($islem=="kup"){
  $sonuc=pow($sayi,3);
  echo "$sayi<sup>3</sup>=$sonuc";
}
elseif($islem=="karekok"){
  $sonuc=sqrt($sayi);
  echo "karekök($sayi)=$sonuc";
}
else{ // faktoriyel hesaplanıyor
  $sonuc=1;
  for($i=1;$i<=$sayi;$i++)
    $sonuc*=$i;
  echo "$sayi!=$sonuc";
}
?>
</body>
</html>
```

Örnek: Kullanıcının girmiş olduğu kullanıcı adı ve şifreye göre **giris.php** sayfasında giriş izni veren yada vermeyen php kodunu yazalım.

(Örneğimizde **Metin Kutusu + Şifre Kutusu + Gönder Düğmesi** kullanılmıştır.)

```
<!--index.php-->
<html>
<body>
  <form name="form1" method="post" action="giris.php">
    Kullanıcı Adınız: <input type="text" name="k_adi"><br>
    Şifreniz:<input type="password" name="sifre"><br>
    <input type="submit" name="giris" value="Giriş">
  </form>
</body>
</html>
```

```
<!--giris.php-->
<html>
<body>
<?php
// İstenen kullanıcı adını ve şifreyi sabit değişkenler olarak tanımlayalım.
define("username","muyo");
define("password","muric");

if($_POST["k_adi"]==username and $_POST["sifre"]==password)
   echo "Giriş izni verildi. Hoşgeldiniz.";
else{
   echo "Kullanıcı adını yada şifreyi yanlış girdiniz. Tekrar deneyiniz!<br>";
   echo "<a href='index.php'>Geri</a>";
}
?>
</body>
</html>
```

Örnek: Girilen metni seçilen biçimlerde ekranda gösteren **yaz.php** sayfasını hazırlayalım.

(Örneğimizde **Metin Kutusu** + **Fieldset** + **Açılır Liste** + **Radyo Buton** + **Onay Kutusu** + **Gönder Düğmesi** kullanılmıştır.)

```
<!--index.php-->
<html>
<body>
  <form name="form1" method="post" action="yaz.php">
    Metni Giriniz: <input type="text" name="metin"><br>

    <fieldset>
    <legend>Yazı stilini seçiniz:</legend>
      Yazı Tipi:
      <select name="tip">
        <option value="Times New Roman">Times New Roman</option>
        <option value="Arial">Arial</option>
        <option value="Verdana">Verdana</option>
        <option value="Tahoma">Tahoma</option>
      </select><br>
      Yazı Boyutu:
      <select name="boyut">
        <option value="1">1</option>
        <option value="2">2</option>
        <option value="3" checked>3</option>
        <option value="4">4</option>
        <option value="5">5</option>
        <option value="6">6</option>
        <option value="7">7</option>
      </select>
    </fieldset>

    <fieldset>
    <legend>Yazı rengini seçiniz:</legend>
      <input type="radio" name="renk" value="red">Kırmızı<br>
      <input type="radio" name="renk" value="blue" checked>Mavi<br>
      <input type="radio" name="renk" value="green">Yeşil<br>
      <input type="radio" name="renk" value="brown">Kahverengi
    </fieldset>

    <fieldset>
    <legend>Yazı biçiminiz seçiniz:</legend>
      <input type="checkbox" name="kalin" value="b">Kalın<br>
      <input type="checkbox" name="egik" value="i">Eğik<br>
      <input type="checkbox" name="alticizili" value="u">Altı çizili
    </fieldset>

    <input type="submit" name="yaz" value="Yaz">
  </form>
</body>
</html>
```

```
<!--yaz.php-->
<html>
<body>
<?php
$metin=$_POST["metin"];
$tip=$_POST["tip"];
$boyut=$_POST["boyut"];
$renk=$_POST["renk"];

$etiket_acilisi="<font face='$tip' size='$boyut' color='$renk'>";
$etiket_kapanisi="</font>";

if(@$_POST["kalin"]){ // kalın seçili ise
  $etiket_acilisi.="<b>"; // sonuna ekler
  $etiket_kapanisi="</b>".$etiket_kapanisi; // başına ekler
}

if(@$_POST["egik"]){ // egik seçili ise
  $etiket_acilisi.="<i>"; // sonuna ekler
  $etiket_kapanisi="</i>".$etiket_kapanisi; // başına ekler
}

if(@$_POST["alticizili"]){ // altı çizili seçili ise
  $etiket_acilisi.="<u>"; // sonuna ekler
  $etiket_kapanisi="</u>".$etiket_kapanisi; // başına ekler
}

echo $etiket_acilisi.$metin.$etiket_kapanisi;
?>
</body>
</html>
```

Örnek: Gizli form elemanı kullanarak sayı tahmin sayfası hazırlayalım. Girilen sayıyı aynı sayfaya POST edelim.

(Örneğimizde **Metin Kutusu + Gizli Form + Gönder Düğmesi** kullanılmıştır.)

```
<!--index.php-->
<html>
<body>

<?php
if(!$_POST){// POST olayı yoksa yani sayfa ilk kez açılıyorsa
  $sayi=rand(1,100); // bulunması gereken sayı rastgele üretiliyor
  $mesaj="Bir sayı tuttuk bilin bakalım kaç!";
}
else{ // POST olayı varsa yani sayı tamin edildiyse
  $tahmin=$_POST["tahmin"];
  $sayi=$_POST["sayi"];
  if($tahmin>$sayi)
    $mesaj="Daha küçük bir sayı giriniz.";
  elseif($tahmin<$sayi)
    $mesaj="Daha BÜYÜK bir sayı giriniz.";
  else{
    $mesaj="Tebrikler bildiniz!";
    $mesaj.="<br><a href='index.php'>Tekrar denemek için tıklayınız.</a>";
  }
}
?>

  <form name="form1" method="post" action="index.php">
    Tahmin: <input type="text" name="tahmin"><br>
    <input type="hidden" name="sayi" value="<?php echo $sayi;?>">
    <input type="submit" name="tahminet" value="Tahmin Et">
  </form>

<?php
echo $mesaj;
?>
</body>
</html>
```

Bu örnekte tahmin edilmesi gereken sayı POST olayı olmadığı zaman yani sayfa ilk kez açıldığında üretiliyor. Daha sonra bu sayı gizli form elemanı ile her POST olayında gönderiliyor. Aksi halde bilinmesi gereken sayı POST olayından sonra kaybolur.

Ayrıca burada gizli formun value kısmına bilinmesi gereken sayıyı PHP kod bloğu açarak echo ile yazdırıyoruz. Formun altında da kullanıcıyı yönlendirici mesajı yine PHP kod bloğunu açıp echo ile yazdırıyoruz.

Örnek: Kullanıcıdan yorum alan ve ekrana yazan php sayfalarını hazırlayalım.

(Örneğimizde **Metin Kutusu + Metin Alanı + Gönder Düğmesi** kullanılmıştır.)

```
<!--index.php-->
<html>
<body>
  <form name="form1" method="post" action="yorum.php">
    <table border="0">
      <tr>
        <td align="right" valign="top">Adınız Soyadınız:</td>
        <td><input type="text" name="adsoyad"></td>
      </tr>
      <tr>
        <td align="right" valign="top">Başlık:</td>
        <td><input type="text" name="baslik"></td>
      </tr>
      <tr>
        <td align="right" valign="top">Tarih:</td>
        <td><input type="text" name="tarih" value="<?php echo date("d.m.Y G:i");?>"
readonly></td>
      </tr>
      <tr>
        <td align="right" valign="top">Yorum:</td>
        <td><textarea name="yorum" rows="8" cols="30"></textarea></td>
      </tr>
      <tr>
        <td> </td>
        <td><input type="submit" name="gonder" value="Gönder"></td>
      </tr>
    </table>
  </form>
</body>
</html>
```

```
<!--yorum.php-->
<html>
<head>
  <style type="text/css">
  <!--

  .main{
    background-color:#EDE;
    border:#F6F thin solid;
    color:#000;
    width:600px
  }

  .adsoyad {
    width:493px;
    background-color:#C39;
    border:#600 thin solid;
    font-weight: bold;
    color:#FFF;
    line-height:30px;
    padding-left:5px;
  }

```

```
24      .div1{
25        width:500px;
26      }
27
28      .baslik{
29        float:left;
30        width:340px;
31        background-color:#F9C;
32        border:#C09 thin solid;
33        line-height:30px;
34        color:#FFF;
35        padding-left:5px;
36      }
37
38      .tarih{
39        float:right;
40        width:146px;
41        background-color:#F9C;
42        border:#C09 thin solid;
43        line-height:30px;
44        color:#FFF;
45        padding-left:5px;
46      }
47
48      .yorum{
49        clear:both;
50        padding: 5px;
51      }
52
53      -->
54      </style>
55  </head>
56  <body>
57  <?php
58  echo "<div class=\"main\">";
59  echo "<div class=\"adsoyad\">".$_POST["adsoyad"]."</div>";
60  echo "<div class=\"div1\">";
61  echo "<div class=\"baslik\">".$_POST["baslik"]."</div>";
62  echo "<div class=\"tarih\">".$_POST["tarih"]."</div>";
63  echo "</div>";
64  echo "<div class=\"yorum\">".$_POST["yorum"]."</div>";
65  echo "</div>";
66  ?>
67  </body>
68  </html>
```

Dosya Yükleme (File Upload)

PHP ile dosya yüklemek oldukça kolaydır. Dosya yükleme formunda metot olarak **POST** metodu kullanılır ve **<form>...</form>** etiketlerine **enctype="multipart/form-data"** özelliği eklenir.

Yüklenecek dosyayı seçmek için **<input type="file" name="form_adi">** formu kullanılır.

Aşağıda örnek bir dosya yükleme formu vardır. İnceleyiniz.

```
1  <!--index.php-->
2  <html>
3  <body>
4    <form name="yukleme" method="post" action="yukle.php" enctype="multipart/form-data">
5      Yüklenecek Dosyayı seçiniz: <input type="file" name="dosya"><br>
6      <input type="submit" name="yukle" value="Yükle">
7    </form>
8  </body>
9  </html>
```

Dosya seçildikten sonra yüklemenin **yukle.php** sayfasında yapılacağı action özelliğinde görülmektedir.

Sunucuya yüklenen bütün dosyalar geçici olarak yüklenmektedir. Bunun sebebi dosyayı kalıcı olarak yüklemeden önce özelliklerine bakmaktır. Geçici olarak yüklenen dosyanın türüne, boyutuna, adına ve daha önce yüklenip yüklenmediğine bakarsınız, eğer tüm bunlar sizin istediğiniz gibiyse bu dosyayı kalıcı olarak sunucuda istediğiniz dizine kaydedersiniz. Kalıcı olarak kaydedilmeyen dosyalar ise zaman içinde sunucudan kendiliğinden silinecektir.

Şimdi geçici olarak yüklenen dosyanın yukle.php sayfasında ne gibi özelliklerine bakacağımıza ve bunu kalıcı olarak nasıl kaydedeceğimize bakalım.

Yüklenen dosyayla ilgili özelliklere ulaşmak için iki boyutlu $_FILES dizisi kullanılır. Bu dizinin birinci boyutunun indisi dosyanın seçildiği formun adı yani name değeridir (**name="dosya"**). İkinci boyutunun indisleri ise sabittir ve aşağıda verilmiştir.

$_FILES["dosya"]["name"] > "dosya" isimli form elemanından yüklenen dosyanın gerçek adıdır.

$_FILES["dosya"]["tmp_name"] > "dosya" isimli form elemanından yüklenen dosyanın geçici adıdır.

$_FILES["dosya"]["type"] > "dosya" isimli form elemanından yüklenen dosyanın türüdür.

$_FILES["dosya"]["size"] > "dosya" isimli form elemanından yüklenen dosyanın byte olarak boyutudur.

$_FILES["dosya"]["error"] > "dosya" isimli form elemanından yükleme yaparken oluşan hatadır.

Unutmayınız aynı anda birden fazla dosyayı upload edebilirsiniz. Şimdi **yukle.php** sayfasında yüklenen dosyanın özelliklerini ekrana yazdıralım.

```
<!--yukle.php-->
<html>
<body>
<?php
echo "<b>Yüklenen dosyanın adı:</b> ",$_FILES["dosya"]["name"],"<br>";
echo "<b>Yüklenen dosyanın geçici adı:</b> ",$_FILES["dosya"]["tmp_name"],"<br>";
echo "<b>Yüklenen dosyanın türü:</b> ",$_FILES["dosya"]["type"],"<br>";
echo "<b>Yüklenen dosyanın boyutu:</b> ",$_FILES["dosya"]["size"],"<br>";
echo "<b>Varsa yüklerken oluşan hata:</b> ",$_FILES["dosya"]["error"],"<br>";
?>
</body>
</html>
```

Yukarıdaki yukle.php sayfasında sadece, geçici olarak yüklenen dosyanın özellikleri ekrana yazdırılmıştır. Bu özellikleri kontrol ederek istenirse dosyayı kalıcı olarak kaydedebilirsiniz.

Bir dosyayı kalıcı olarak kaydetmek için **move_uploaded_file($p1, $p2);** fonksiyonu kullanılır. Bu fonksiyon iki parametre alır.

$p1 > Dosyanın geçici adıdır. Buraya **$_FILES["dosya"]["tmp_name"]** ifadesi yazılır.

$p2 > Dosyanın kalıcı yolu ve adıdır. Buraya ise istediğiniz bir yolu ve adı yazabilirsiniz.

Dosyaları kalıcı olarak kaydederken aşağıdaki sorunlardan birini yaşayabilirsiniz. Bunlara dikkat ediniz.

- Yüklenen dosyanın boyutunun çok büyük olması. Sunucuda php.ini dosyasında izin verilen boyuttan daha büyük bir dosya yüklemeye çalışmak.
- Dosyayı kalıcı olarak istenen dizine kaydederken kayıt izninin olmaması. Bunu, dizine yazma iznini vererek aşabilirsiniz.
- Dosyaya verilen kalıcı adın daha önce kullanılmış olması başka bir ifadeyle aynı dosyayı tekrar yüklemeye çalışmak. Bunu, dosyaya farklı bir ad vererek aşabilirsiniz. Yada baştan **file_exists($dosya_adi)** fonksiyonu ile dosyanın var olup olmadığına bakabilirsiniz.
- Dosyayı kaydederken dosya adında Türkçe'ye özgü karakterlerin ve geçersiz karakterlerin kullanılması (çÇğĞıİöÖşŞüÜ). Bu sorunu Türkçe karakterleri temizleyen bir fonksiyon tanımlayıp kullanarak aşabilirsiniz.

Örnek: Seçilen dosyayı kendi adıyla upload isimli dizine yükleyen php sayfalarını hazırlayalım. Web dizininde **upload** isimli bir dizin yoksa oluşturunuz ve yazma izni veriniz.

```
<!--index.php-->
<html>
<body>
  <form name="yukleme" method="post" action="yukle.php" enctype="multipart/form-data">
    Yüklenecek Dosyayı seçiniz: <input type="file" name="dosya"><br>
    <input type="submit" name="yukle" value="Yükle">
  </form>
</body>
</html>
```

```
<!--yukle.php-->
<html>
<body>
<?php
$gecici_ad=$_FILES["dosya"]["tmp_name"];
$kalici_yol_ad="upload/".$_FILES["dosya"]["name"]; // dosya kendi adıyla upload dizinine
kaydedilecek

if (move_uploaded_file($gecici_ad,$kalici_yol_ad)) // eğer dosya kaydedilirse
  echo "Dosya başarı ile yüklendi.";
else
  echo "Yükleme başarısız!";
?>
</body>
</html>
```

Bu örnekte dosya, hiçbir özelliğine bakmaksızın upload dizinine kaydediliyor.

Örnek: Daha önce yüklenmemiş olan **jpg** resimlerini kendi adlarıyla **upload** dizinine yükleyelim.

```
<!--index.php-->
<html>
<body>
  <form name="yukleme" method="post" action="yukle.php" enctype="multipart/form-data">
    Jpg resmi seçiniz: <input type="file" name="dosya"><br>
    <input type="submit" name="yukle" value="Yükle">
  </form>
</body>
</html>
```

```
<!--yukle.php-->
<html>
<body>
<?php
$gecici_ad=$_FILES["dosya"]["tmp_name"];
// dosya kendi adıyla upload dizinine kaydedilecek
$kalici_yol_ad="upload/".$_FILES["dosya"]["name"];

if (file_exists("upload/".$_FILES["dosya"]["name"])) // yüklenen dosya upload dizininde varsa
  echo "Seçtiğiniz dosya daha önce yüklenmiştir.";
else{
    if ($_FILES["dosya"]["type"]=="image/jpeg"){
      if (move_uploaded_file($gecici_ad,$kalici_yol_ad)) // eğer dosya kaydedilirse
        echo "Dosya başarı ile yüklendi.";
      else
        echo "Yükleme başarısız!";
    }
    else
      echo "Lütfen jpg resmi seçiniz.";
}
?>
```

```
</body>
</html>
```

Örnek: Aşağıdaki şartlar sağlanırsa dosyayı kendi adıyla **upload** dizinine yükleyelim.

-Hata oluşmazsa

-Dosya daha önce yüklenmemişse

-Dosya gif dosyası ise

```
<!--index.php-->
<html>
<body>
  <form name="yukleme" method="post" action="yukle.php" enctype="multipart/form-data">
    Gif resmi seçiniz: <input type="file" name="dosya"><br>
    <input type="submit" name="yukle" value="Yükle">
  </form>
</body>
</html>
```

```
<!--yukle.php-->
<html>
<body>
<?php
$gecici_ad=$_FILES["dosya"]["tmp_name"];
$kalici_yol_ad="upload/".$_FILES["dosya"]["name"]; // dosya kendi adıyla kaydedilecek

if ($_FILES["dosya"]["error"]) // hata oluştu ise
  echo "Hata : ",$_FILES["dosya"]["error"];
else{
    if (file_exists("upload/".$_FILES["dosya"]["name"])) // yüklenen dosya upload dizininde
varsa
      echo "Seçtiğiniz dosya daha önce yüklenmiştir.";
    else{
      if ($_FILES["dosya"]["type"]=="image/gif"){
        if (move_uploaded_file($gecici_ad,$kalici_yol_ad)) // eğer dosya kaydedilirse
          echo "Dosya başarı ile yüklendi.";
        else
          echo "Dosya yükleme başarısız!";
      }
      else
        echo "Lütfen gif resmi seçiniz.";
    }
}
?>
</body>
</html>
```

Örnek: Aşağıdaki şartlara göre kullanıcının seçtiği dosyayı kullanıcının yazdığı ad ile **upload** dizinine kaydeden php sayfalarını hazırlayalım.

-Hata oluşmazsa

-Dosya daha önce yüklenmemiş ise

-Dosya pdf dosyası ise

-Dosya boyutu 500 KB'dan küçük ise

```
<!--index.php-->
<html>
<body>
  <form name="yukleme" method="post" action="yukle.php" enctype="multipart/form-data">
    <table border="0">
      <tr>
        <td>Dosyanın Kalıcı Adı:</td>
        <td><input type="text" name="kalici_ad"></td>
      </tr>
      <tr>
        <td>PDF Dosyası Seçiniz:</td>
        <td><input type="file" name="dosya"></td>
      </tr>
      <tr>
        <td> </td>
        <td><input type="submit" name="yukle" value="Yükle"></td>
      </tr>
    </table>
  </form>
</body>
</html>
```

```
<!--yukle.php-->
<html>
<body>
<?php
$gecici_ad=$_FILES["dosya"]["tmp_name"];
$kalici_yol_ad="upload/".$_POST["kalici_ad"].".pdf";

if ($_FILES["dosya"]["error"]) // hata oluştu ise
  echo "Hata : ",$_FILES["dosya"]["error"];
else{
  if (file_exists($kalici_yol_ad)) // yüklenen dosya upload dizininde varsa
    echo "Yazdığınız ad ile bir dosya zaten kayıtlıdır.";
  else{
    if ($_FILES["dosya"]["size"]>500*1024)
      echo "500KB'dan küçük bir dosya seçiniz.";
    else{
      if ($_FILES["dosya"]["type"]=="application/pdf"){
        if (move_uploaded_file($gecici_ad,$kalici_yol_ad)) // eğer dosya kaydedilirse
          echo "Dosya başarı ile yüklendi.";
        else
          echo "Dosya yükleme başarısız.";
      }
      else
        echo "Lütfen PDF dosyası seçiniz.";
    }
```

```
26      }
27  }
28  ?>
29  </body>
30  </html>
```

Örnek: Şimdi tam bir dosya yükleme örneği verelim. Burada dosyalar kendi adlarıyla **upload** dizinine kaydedilecektir. Tabiki Türkçe'ye özgü karakterler temizlendikten sonra. Bunun için bir fonksiyon tanımlayıp kullanacağız. Ayrıca yükleme aynı sayfada yapılacaktır.

```
1   <!--index.php-->
2   <html>
3   <body>
4     <form name="yukleme" method="post" action="index.php" enctype="multipart/form-data">
5       <table border="0">
6         <tr>
7           <td>Dosya Seçiniz:</td>
8           <td><input type="file" name="dosya"></td>
9         </tr>
10        <tr>
11          <td> </td>
12          <td><input type="submit" name="yukle" value="Yükle"></td>
13        </tr>
14      </table>
15    </form>
16
17  <?php
18  function turkce($metin){
19    $aranan=array("ç","Ç","ğ","Ğ","ı","İ","ö","Ö","ş","Ş","ü","Ü"," ");
20    $yerine=array("c","c","g","g","i","i","o","o","s","s","u","u","_");
21    return str_replace($aranan,$yerine,$metin);
22  }
23
24  if($_POST){
25    $gecici_ad=$_FILES["dosya"]["tmp_name"];
26    $kalici_yol_ad="upload/".turkce($_FILES["dosya"]["name"]);
27
28    if ($_FILES["dosya"]["error"]) // hata oluştu ise
29      echo "<font color='green'>Hata : ",$_FILES["dosya"]["error"],"</font>";
30    else{
31      if (file_exists($kalici_yol_ad)) // yüklenen dosya upload dizininde varsa
32        echo "<font color='red'>Yazdığınız ad ile bir dosya zaten kayıtlıdır.</font>";
33      else{
34        if (move_uploaded_file($gecici_ad,$kalici_yol_ad)) // eğer dosya kaydedilirse
35          echo "<font color='green'>Dosya başarı ile yüklendi.</font>";
36        else
37          echo "<font color='red'>Dosya yükleme başarısız.</font>";
38      }
39    }
40  }
41  ?>
42  </body>
43  </html>
```

PDO (*PHP Data Objects*) Nedir?

PHP'deki veritabanlarına erişmek için hafif ve tutarlı bir arayüz tanımlar. Bu sınıf arayüz tanımı bulunan her veritabanı ile rahatlıkla çalışabilir.

Veritabanı Bağlantısı Nasıl Yapılır?

```
<?php
  ## Veritabanı bağlantısı
  $host = "localhost";
  $hesap = "root";
  $sifre = "";
  $vt_adi = "haber";
  try{
    $baglan = new PDO ("mysql:host=$host;dbname=$vt_adi;charset=utf8",$hesap,$sifre);
  }
  catch(PDOException $e){
    echo $e->getMessage();
  }

  ## Karakter seti ayarla
  $baglan->exec("SET CHARACTER SET utf8");
  $baglan->exec("SET NAMES utf8");
?>
```

exec() Metodu

SQL deyimini çalıştırır ve etkilenen satır sayısını döndürür. Ekleme, silme, güncelleme işlemlerinde kullanılır. Ancak bu SQL işlemleri **güvenli** değildir.

Örnek exec() kullanımı

```
<?php
  ## Veri Ekleyelim
  $haber_ekle = $baglan->exec("INSERT INTO haberler(haber_baslik,haber_detay) VALUES
('Deneme Haber Başlık','Deneme Haber Detay')");

  ## Etkilenen satır sayısını döndürelim
  echo $haber_ekle." haber eklendi.";
?>
```

Aynı işlemler **silme** ve **güncelleme** içinde geçerlidir.

ÖNEMLİ!

```
<?php
  $haber_sil = $baglan->exec("DELETE FROM haberler WHERE haber_id = '1'");
  if(!$haber_sil){
    //Yanlış Kullanım! Bu verinin silinmediği bilgisini vermez.
```

```
  }

  if($haber_sil == FALSE){
    //Doğru kullanım, veri silinmediğinde FALSE sonucu döner
  }
?>
```

query() Metodu

SQL sorugusunu çalıştırıp, tablodaki verileri bir dizi olarak geri döndürür.

$sonuc->query(SQL,sonuc biçimi) herhangi bir sonuç biçimi girilmezse varsayılan olarak PDO::FETCH_BOTH kullanılır.

Kullanılan sonuç biçimleri

PDO::FETCH_ASSOC = Sütun isimlerine göre bir değer döndürür.

PDO::FETCH_BOTH = Varsayılandır. Hem sütun isimlerine hemde indislere göre bir değer döndürür.

PDO::FETCH_BOUND = Sütun değerlerini **bindColumn()** ile PHP değişkenlerine aktarır ve TRUE değeri döner.

PDO::FETCH_NUM = Sütun numaralarına göre indisli bir dizi dönderir.

En çok kullanılabicek sonuç biçimlerini listeledim.

Prepare() Metodu

Prepare() çalıştırılmak üzere bir SQL sorgusun hazırlar. Prepare() metodu **bind_param()**, **execute()**, **bindColumn()**, **bindValue()** metdolarıyla birlikte çalışabilir. Prepare() metodu ile SQL sorgularınız son derece **güvenli** çalışmaktadır. Prepare() metodu ile iki türlü tanımalama yapılır. Biri ":" ile diğeri ise "?" ile tanımlama yapılır. Burada sadece fikir vermesi amacıyla bir örnek vereceğim. Detaylı örneği ise **execute()** metodunu kullanırken vereceğim.

? işareti ile kullanım

```
<?php
  $sorgu = $baglan->prepare("SELECT * FROM haberler WHERE haber_id = ?");
?>
```

: işareti ile kullanım

```
<?php
  $sorgu = $baglan->prepare("SELECT * FROM haberler WHERE haber_id = :id");
?>
```

bindParam() Metodu

bindParam() metodu **prepare()** metodu ile hazırlanan SQL sorgusunda ? veya :isim ile belirtilen parametrelerin hazırlanmasında kullanılır. Kısacası bir kayıt yapacağınız zaman bu verilerin tiplerini kendimiz belirleyebiliriz (string ifade, integer ifade vb.).

bindParam() metodunda kullanılan veri tipleri

PDO::PARAM_INT = SQL integer (sayısal) veri türünü ifade eder.

PDO::PARAM_STR = char, varchar gibi ifadelerdir.

PDO::PARAM_NULL = NULL (boş) bir değeri ifade eder.

: işareti için bir örnek;

```
<form action="" method="post">
  <input type="text" name="kullanici_adi" />
  <input type="password" name="sifre" />
  <button type="submit">Giriş Yap</button>
</form>
```

```
<?php
  if($_POST){
  $k_adi = trim($_POST["kullanici_adi"]);
  $k_sifre = trim($_POST["sifre"]);
  if($k_adi && $k_sifre){
    $sorgu = $baglan->prepare("SELECT * FROM uyeler WHERE ad = :kullanici_adi and sifre =
:password");
    $sorgu->bindParam(':kullanici_adi', $k_adi, PDO::PARAM_STR);
    $sorgu->bindParam(':password', $k_sifre, PDO::PARAM_STR);
    $sorgu->execute(); //Sorguyu çalıştır.
    if($sorgu->rowCount() > 0){ //rowCount() ile etkilenen satır sayısını bulduk.
      echo "giriş başarılı!";
    }
    else{
      echo "giriş başarısız!";
    }
  }
  else{
    echo "Boş alan bırakmayınız...";
  }
}
?>
```

execute() Metodu

prepare() sorgusu ile hazırlanan bir SQL sorgusunu çalıştırır. Aldığı ikinci bir parametre ilede **?** veya **:deger** parametrelerini değişkenlere bağlar.

execute() için birinci örnek;

```
<?php
  $sorgu = $baglan->prepare("SELECT * FROM haberler");
  $sorgu->execute();
?>
```

execute() için ikinci örnek;

```
<?php
  $kullanici_adi = trim($_POST["kullanici_adi"]);
  $sifre = trim($_POST["sifre"]);
  $sorgu = $baglan->prepare("SELECT * FROM uyeler WHERE kullanici_adi = ? && sifre = ?");
  $sorgu->execute(array($kullanici_adi,$sifre));
?>
```

execute() için üçüncü örnek;

```
<?php
  $kullanici_adi = trim($_POST["kullanici_adi"]);
  $sifre = trim($_POST["sifre"]);
  $sorgu = $baglan->prepare("SELECT * FROM uyeler WHERE kullanici_adi =:ad && sifre
=:password");
  $sorgu->execute(array(
    ':ad' => $kullanici_adi,
    ':password' => $sifre
  ));
?>
```

bindColumn() Metodu

Bu metod ile sütun başlıklarına istediğimiz **değişkenleri** atayarak kullanabiliriz.

bindColumn() örnek kullanımı;

```
<?php
  $haber_id = 10;
  $sorgu =  $baglan->prepare("SELECT * FROM haberler WHERE haber_id = ?");
  $sorgu->execute(array($haber_id));
  $sorgu->bindColumn('haber_baslik',$degiskenim);
  $sorgu->fetch(PDO::FETCH_BOUND);
  echo $degiskenim;
?>
```

Yukarıda kullandığımız **fetch()** metodu ile sonuçları farklı biçimlerde elde edebiliriz.

Select (Seçme) İşlemi

```
<?php
  $haberler = $baglan->query("SELECT * FROM haberler");
```

```
  $haber = $haberler->fetch(PDO::FETCH_ASSOC);
  echo $haber["haber_baslik"];
?>
```

Select (Seçme) İşlemi Çoklu

```
<?php
  $haberler = $baglan->query("SELECT * FROM haberler",PDO::FETCH_ASSOC);
  foreach($haberler as $haber_yaz){
    echo $haber_yaz["haber_baslik"]."<br />";
  }
?>
```

Insert (Ekleme) İşlemi

```
<?php
  $haber_baslik = trim($_POST["haber_baslik"]);
  $haber_detay = trim($_POST["haber_detay"]);
  $haber_ekle = $baglan->prepare("INSERT INTO haberler (haber_baslik,haber_detay)
VALUES(?,?)");
  $ekle = $haber_ekle->execute(array($haber_baslik,$haber_detay));
  if($ekle == TRUE){
    echo 'Haber başarılı bir şekilde eklendi';
  }
  else{
    echo 'Haber eklenirken bir hata oluştu.';
  }
?>
```

Delete (Silme) İşlemi

```
<?php
  $haber_id = trim($_GET["haber_id"]);;
  $haber_sil = $baglan->prepare("DELETE FROM haberler WHERE haber_id = ?");
  $sil = $haber_sil->execute(array($haber_id));
  if($sil == TRUE){
    echo "Haber başarılı bir şekilde silindi.";
  }
  else{
    echo "Haber silinirken bir hata oluştu.";
  }
?>
```

Update (Güncelleme) İşlemi

```
<?php
  $haber_baslik = trim($_POST["haber_baslik"]);
  $haber_detay = trim($_POST["haber_detay"]);
  $haber_id = trim($_GET["haber_id"]);
  $haber_guncelle =  $baglan->prepare("UPDATE haberler SET
  haber_baslik = ?,
  haber_detay = ? WHERE haber_id = ?");
  $guncelle = $haber_guncelle->execute(array($haber_baslik,$haber_detay,$haber_id));
  if($guncelle == TRUE){
```

```
    echo "Haber başarılı bir şekilde güncellendi.";
  }
  else{
    echo "Haber güncellenirken bir hata oluştu.";
  }
?>
```

errorInfo() ile Hataları Yazdırma

```
<?php
  $sql = $baglan->prepare('SELECT * FROM haberler');
  $sql->execute();
  if($sql->errorCode() == 0){
    //hata yoksa yapılacak işlemler
  }
  else{
    $hata = $sql->errorInfo();
    echo($hata[2]); // Hatayı ekrana yazdır.
  }
?>
```

Veritabanı bağlantısı sonlandırma

```
<?php
  $baglan = null;
?>
```

Güvenlik

Mysql'de dışarıdan gelen bilgileri ancak filitreleyerek veritabanına kaydediyor ya ya veritabanında değişiklikler yapabiliyorduk. Ancak **PDO**'da öyle bir sıkıntı yok. Örnek vermek gerekirse;

site.com/index.php?**uye_id=10** burada ID'si 10 olan üyenin bilgilerini **MYSQL** ile düzenlemek isteseydik büyük bir ihtimal ile **SQL injeksiyon** yerdik. Ancak PDO'da **prepare()** ile böyle bir durum söz konusu değildir.

Kaynakça

MEGEP Modülleri, HTML İle Basit Web İşlemleri

https://www.w3schools.com/

https://ig.cc.metu.edu.tr/html/index.php

https://ig.cc.metu.edu.tr/html/css.php

http://www.phpdefteri.com/tumdersler/1/giris.html

https://www.phpr.org/php-ile-smtp-mail-gonderme/

http://onurdeger.com.tr/konu/69/php-data-objects-pdo-kullanimi

Hazırlayan: İnci Bayramoğlu