T.C.
MİLLÎ EĞİTİM BAKANLIĞI

MEGEP

(MESLEKİ EĞİTİM VE ÖĞRETİM SİSTEMİNİN
GÜÇLENDİRİLMESİ PROJESİ)

ENDÜSTRİYEL OTOMASYON TEKNOLOJİLERİ

PROGRAMLAMA

ANKARA 2007

Milli Eğitim Bakanlığı tarafından geliştirilen modüller;

- Talim ve Terbiye Kurulu Başkanlığının 02.06.2006 tarih ve 269 sayılı Kararı ile onaylanan, Mesleki ve Teknik Eğitim Okul ve Kurumlarında kademeli olarak yaygınlaştırılan 42 alan ve 192 dala ait çerçeve öğretim programlarında amaçlanan mesleki yeterlikleri kazandırmaya yönelik geliştirilmiş öğretim materyalleridir (Ders Notlarıdır).
- Modüller, bireylere mesleki yeterlik kazandırmak ve bireysel öğrenmeye rehberlik etmek amacıyla öğrenme materyali olarak hazırlanmış, denenmek ve geliştirilmek üzere Mesleki ve Teknik Eğitim Okul ve Kurumlarında uygulanmaya başlanmıştır.
- Modüller teknolojik gelişmelere paralel olarak, amaçlanan yeterliği kazandırmak koşulu ile eğitim öğretim sırasında geliştirilebilir ve yapılması önerilen değişiklikler Bakanlıkta ilgili birime bildirilir.
- Örgün ve yaygın eğitim kurumları, işletmeler ve kendi kendine mesleki yeterlik kazanmak isteyen bireyler modüllere internet üzerinden ulaşılabilirler.
- Basılmış modüller, eğitim kurumlarında öğrencilere ücretsiz olarak dağıtılır.
- Modüller hiçbir şekilde ticari amaçla kullanılamaz ve ücret karşılığında satılamaz.

İÇİNDEKİLER

AÇIKLAMALAR

KOD	**481BB0084**
ALAN	**Endüstriyel Otomasyon Teknolojileri**
DAL/MESLEK	**Alan Ortak**
MODÜLÜN ADI	**Programlama**
MODÜLÜN TANIMI	Betik dilinin temel kullanım becerilerinin kazanıldığı öğretim materyalidir.
SÜRE	40/32
ÖN KOŞUL	-
YETERLİK	Betik dilinde programlama yapmak
MODÜLÜN AMACI	**Genel Amaç** Betik dili ile programlama işlemini doğru olarak yapabileceksiniz. **Amaçlar** 1. Bilgisayar işletim sisteminde WEB sunucusunun kurulumunu hatasız olarak yapabileceksiniz. 2. Betik dili ile sunucu taraflı programlarda döngü ve koşulları hatasız bir şekilde kullanabileceksiniz. 3. Betik dili ile sunucu taraflı programlarda oturum nesnesini hatasız bir şekilde kullanabileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	**Ortam:** Bilgisayar laboratuvarı **Donanım:** Bilgisayar, hub, işletim sistemi
ÖLÇME VE DEĞERLENDİRME	Her faaliyetin sonunda ölçme soruları ile öğrenme düzeyinizi ölçeceksiniz. Araştırmalarla grup çalışmaları ve bireysel çalışmalarla öğretmen rehberliğinde ölçme ve değerlendirmeyi gerçekleştirebileceksiniz.

GİRİŞ

Sevgili Öğrenci,

İnternette uygulamalar yapmak gün geçtikçe kolaylaşmaktadır. Bu yüzden, özellikle aktif sayfaların yapımı ve kullanımı uygulamaların daha fazla yaygınlaşmasını sağlamaktadır. Bu modülümüzde biz aktif sayfalar yapabilmek için gerekli olan PHP betik dilini anlatmaya çalışacağız.

Bu modül içerisinde sırasıyla betik dilinin temellerini ve genel kullanım yöntemlerini öğreneceksiniz.

ÖĞRENME FAALİYETİ–1

AMAÇ

Bilgisayar işletim sisteminde web sunucusunun kurulumunu hatasız olarak yapabileceksiniz.

ARAŞTIRMA

Bu öğrenme faaliyetinden önce aşağıdaki hazırlıkları yapmalısınız.

- HTML komutları ve web sayfası yapımı konularını araştırınız.
- PHP betik dili versiyonları ve arasındaki farkları araştırınız.

1. WEB SİSTEMİNİN ANA HATLARI

1.1. Web Nedir

WWW, World Wide Web'in kısaltılmış halidir ve "Tüm dünyayı saran ağ" anlamına gelmektedir. Mekanizma, CERN (Conseil Europeen pour la Recherche Nucleaire) de bir bilim adamı olan Mr.Tim Berners-Lee tarafından 1989 yılında da dokümanların kolayca paylaşılabilmesi önerisiyle ortaya çıkmıştır. WWW'nin internette tüm insanların kullanımına açılması ile de hızla bütün dünyaya yayılmıştır.

Web, temel olarak Sunucu/İstemci modeli üzerine kurulmuştur. Bu sistem internetteki bütün bilgilere Hyper Text kullanarak ulaşmamızı sağlar. Web aşağıdaki üç yapıyı içerir.

HTML (**H**yper **T**ext **M**arkup **L**anguage)
HTTP (**H**yper **T**ext **T**ransfer **P**rotocol)
URL (**U**niversal **R**esource **L**ocator)

İlk olarak, HTML Web'in çekirdeğini oluşturan SGML (Standart Generalized Markup Language) temeli baz alınarak oluşturulmuş Hyper Text'i tanımlayan etiket yapısında bir dildir. Web Hyper Text'i çeşitli bilgileri birleştirme amacı ile kullanır. Hyper Text'i tanımlamak amacı ile de, HTML dili oluşturulmuştur. HTML, diğer HTML dokümanları ve diğer kaynaklar ile ilgili link bilgilerini içerir. Ayrıca HTML, dokümanın formatını da belirtir.

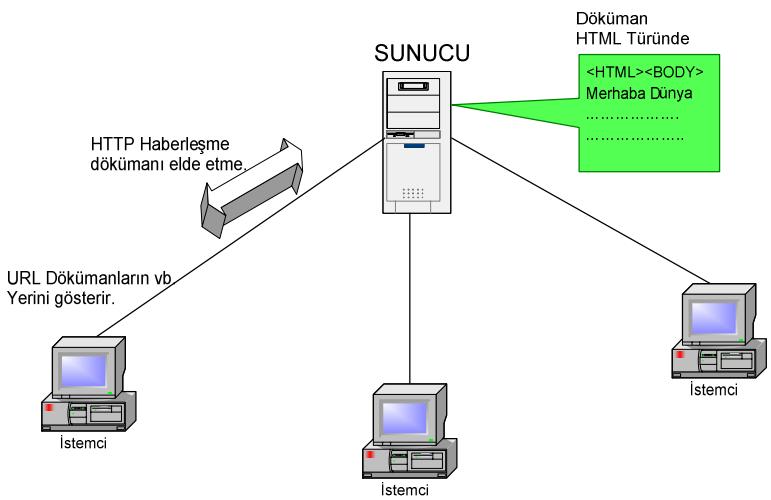
HTTP (Hyper Text Transfer Protocol) Web'de Sunucu/İstemci arasında kullanılan ve dosyalara veya başka bilgisayardaki verilere ulaşmak amacı ile geliştirilmiş bir protokoldür. Web tarayıcısında adres bilgisini girerken "http://" şeklinde kullanılır.

Url, internetteki kaynaklara isim isim nasıl ulaşılacağına karar veren yapıdır. Bu kaynaklar HTML dökümanı, resim, animasyon ya da program olabilir. Her kaynağın yeri vardır. Önemli olan bu kaynakların yerinin belirlenmesidir. Bu amaçla URL bu kaynaklara ulaşım bilgilerini içerir. URL, URI (Universal Recourse Idendifier "Evrensel Kaynak Belirteci ") olarak adlandırılır.

Örnek: URL *http://www.jica.com/index.html*

www.jica.com. Domain adına sahip makinadaki index.html adlı HTML türündeki dosya.

HTTP protokolü kullanılmış



Şekil 1.1: Web'in mekanizması

1.2. Http Nedir

Http ve Url webde anahtar roldedir. HTT alıcının isteği doğrultusunda kaynakları bu alıcıya iletmekle yükümlüdür. Bununla beraber, Web servis birleştirici olarak görevlidir. HTTP servis uygulamaları için iletişim protokolü rolündedir.

HTTP, birkaç talimatın birleşiminden meydana gelmektedir. HTTP'de sadelik en önemli özelliktir. Bu nedenle hızla popüler hale gelmiştir. HTTP'nin sadeliği nedeni ile birleşik anlaşmaya gerek yoktur. Sunucu ve istemci, her ikisi için de işlem oldukça azdır. HTTP minimum haberleşmede aşağıdaki yapıyı takip eder.

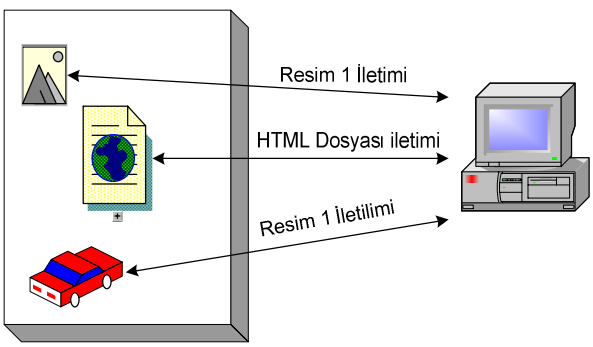
- İstemciden sunucuya istekte bulunmak (İSTEK).
- İşlem sonucunu istemciye iletmek (YANIT).

Bu yapı, cevabı bir döngüde elde edebilmek için kolay ve anlaşılabilir bir yapıdır. Bu yavaş bir yapıdır. Örneğin bir HTML dokümanı web tarayıcı tarafından çağrıldığında bu döngü sadece bir defa işlenmiş olur. İşlem sadece bununla bitmiş olur. Bu oldukça kısa ve özlü bir yapıdır. Bununla birlikte linkler ile sayfaya eklenmiş kaynaklar da bu işlemleri tekrarlattıracağından işlemler yavaş olacaktır, bu da HTML'nin etkisizliğini göstermektedir. Bunun nedeni linkli durumdaki her kaynak için bu döngünün tekrarlanmasıdır. Kısacası tüm verilerin ulaşımı tamamlandığında aynı anda web tarayıcıda tüm verileri göstermiş olur.

Başka bir deyişle, içerisinde bol miktarda resim içeren web sayfasının web tarayıcısında yavaş görünmesinin sebepleri aşağıdakilerdir;

- Resimlerin ağdan bilgisayara ulaşması zaman alır.
- Resimlerin fazlalığı ve http'nin bağlantı yapısı ve her resmi tek tek yüklemesidir.

HTTP kaynakların gönderildiği ya da alındığı bir protokoldür. HTML'nin görüntülenmesi http'nin görevi değildir. Bu web tarayıcısı ile HTML arasında bir konudur. HTTP birçok çeşit kaynağı HTML ile birlikte gönderir.



Şekil 1.2: Web elemanlarının iletimi

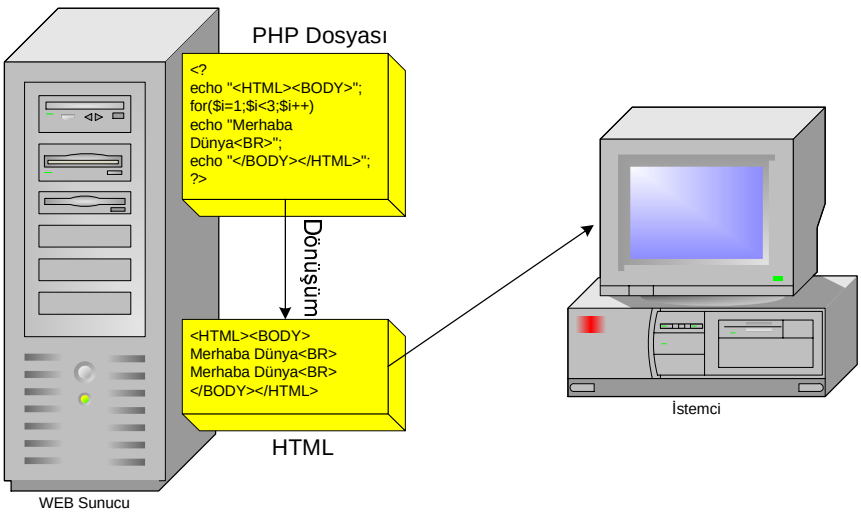
1.3. Betik Dilinin Ana Hatları

1.3.1. Php Nedir

PHP, web'de uygulamalar yazmak amacı ile geliştirilmiş bir Script (Betik) dilidir. PHP açık kaynak kodlu ve kullanımı herhangi bir ücrete tabi olmayan, kolay anlaşılır bir dildir.

PHP, 1995 yılında ilk olarak web'de kullanılmak için PHP/FI dili olarak ortaya çıkmıştır. Açık kodlu script dillerinin azlığı nedeni ile hızla yaygınlaşmıştır. PHP/FI Mr. Rasmus Lerdor tarafından geliştirilmiştir. Bununla birlikte daha sonraları bir grup programcı tarafından daha da geliştirilmiştir. PHP3 ile birlikte nesne yönelimli bir dil haline gelmiştir. Günümüzde PHP4 ve PHP5 versiyonları kullanılmaya başlanmıştır.

PHP web sunucu üzerinde çalıştırılan (sunucu tabanlı) programdır. Herhangi bir istemciden istek geldiği anda web sunucu PHP kodlarını derleyerek HTML koda çevirir, ardından istekte bulunan bilgisayara iletilir. Aşağıdaki şekil bu yapıyı anlatmaktadır.



Şekil 1.3: PHP nin istemciye iletilmesi

1.3.2. Php'nin Özellikleri

PHP'nin kısa sürede yaygınlaşmasının nedenleri şunlardır.

- **Yazım Düzeninin Kolay Olması**

Komut yapısı C ve Perl dillerine benzer. Böylelikle C ve Perl bilen birisi PHP'yi de kullanabilir. Ayrıca programlama deneyimi olmayan biri dahi programı anlayabilir.

- **Nesne Yönelimli Olması**

Büyük boyutlu uygulamaları sadece programlama dilinin yazım düzeninin kolay olması yeterli değildir. PHP kolaydır fakat sınıfları kullanabilen bir nesne yönelimli dildir ve nesne yönelimi sayesinde iyi programlar yapmak mümkündür. Dahası birçok çeşit sınıf ve kütüphane eklenebilir.

- **Derlemeye Gerek Olmaması**

Çalıştırılmadan önce C ve Java dillerindeki gibi derlemeye ihtiyaç yoktur. Komut hataları ve çalışma anı hataları doğrudan olarak tarayıcıda gösterilir. Ayrıca hatalar metin düzenleme programlarında düzeltilebilir.

- **Ek Kolaylıklarının Olması**

PHP' de birçok ek kolaylıklar kullanılabilir. Özellikle veri tabanı ile ilgili birçok ek kolaylıklar vardır.

- **Yüksek Performanslı Apache Modülünün Olması**

1.3.3. Uygulama Sunucusu Yapımı

PHP programlama yapmak amacı ile bilgisayarınıza Apache web sunucu kurmalısınız. Biz bu amaçla Linux'un Fedora sürümünü kullanacağız. Fedora'da Apache web sunucu PHP modülü ile birlikte gelmektedir. Ayrıca Ek Çalışma bölümünde Windows kurulu bilgisayarınızda da PHP modülü bulunan web sunucu kurulumu anlatılacaktır.

1.3.3.1. Fedora Core Kurulumu

Not: Bu uygulamada Fedora Core 3 sürümünü kullanılacaktır. Kurulum CD'leri internetten ücretsiz olarak indirebilir. Fedora'nın yeni sürümlerinde önemli bir farklılık yoktur. Bu nedenle Fedora'nın diğer üst sürümleri de kullanabilir.

Öncelikle bilgisayarınızda mutlaka bir Ethernet kartının bulunması gerekir. Bunun amacı uygulamaları ağa bağlı diğer bilgisayarlardan izleyebilmektir. Kurulum sırasında Ip numarası olarak bulunduğunuz ağa uygun bir Ip numarası belirlenebilir.

- **Kurulum Aşamaları**

  - Kurulum 1 CD si ile bilgisayarınızı başlatınız.

  - Kurulum başlangıcında, aşağıdaki mesaj çıkacaktır.

boot: *"Push the Enter key"* (CD'den açılış için enter Tuşuna basınız). Bu bölümü enter tuşuna basarak geçebiliriz.

  - Kuruluma başlamak için media denetleme bölümü atlanır. Bu bölümde kurulum CD'leri bozukluklara karşı denetlenecektir. Bazen ihtiyaç olmayabilir. Bu denetim uzun sürebilir.

  - Fedora Core Hoşgeldiniz *"Welcome to Fedora Core"* Sonraki *Bu bölüm"Next"* ile geçilir.

  - Dil Seçimi *"Language Selection"* İngilizce*"English(English)" veya "Türkçe"* seçilebilir.

  - Klavye düzeni seçimi *"Keyboard Configuration"* ABD İngilizce *"U.S. English" veya "Türkçe"*seçilir.

- Bu basamakta fare tipi seçilir.

➢ **Mouse Konfigürasyonu"Mouse Configuration", "Wheel Mouse (PS/2)"**

- Kurulum tipi seçimi yapmak için kurulacak paketler tespit edilir.

Kurulum Tipi "*Installation Type*" Özel tip "*Custom*"

- Otomatik disk bölümlendirme yapmak için, Disk Bölümlendirme "*Ayarları Disk Partitioning Setup"* Otomatik Bölümlendirme "*Automatically Partition*" seçilir.

- "*Automatically Partition*" Otomatik Bölümlendirme Sistemdeki Bütün Bölümlendirmeleri İptal Et "*Remove all partitions on this system*" seçilir.

- Aşağıdaki sürücüdeki bütün bölümler (TÜM VERİLER) silinecektir:

You have chosen to remove all partitions (ALL DATA) on the following drivers:

/dev/hda

Are you sure you want to do this? "Yes"

(Emin misiniz : "evet")

- Disk Ayarları *"Disk Setup"* sonraki "*Next*"

- Açılış Seçici Ayarları *"Boot Loader Configuration"* Sonraki *"Next"*

- Ağ Ayarları "Network Configuration"

Network Aygıtları "Network Devices"

Ayarla->>> Seçim iptal Konfigürasyonda DHCP kullan

"Edit" ->>> "Check off" Configure using DHCP

IP address 192.168.2.1**

Örneğin bilgisayar numarası 20 ise bu alana 120 giriniz.

Netmask 255.255.255.0

"OK"

<Host Adı> <*Host Name*>

"Elle" *"Manually"*

Örneğin; I.E öğrencileri: ie.isminiz.com

Aynı Domain ismi kullanılmamaya
özen gösterilmelidir.

<Diğer Ayarlar>
<Miscellaneous Settings>
Gateway 192.168.2.254
Primary DNS 192.168. 2.1**
Örneğin bilgisayar numaranız 20 ise 120 giriniz.

- Güvenlik Duvarı ayarları *"Firewall Configuration"*

Güvenlik Duvarı Aktif *"Enable firewall"*

Hangi servisler güvenlik duvarını geçebilsin?

"What services should be allowed to pass through the firewall?"

"WWW (HTTP)" "FTP" "SSH" "Telnet" "Mail (SMTP)"

Eğer tüm trafiği bir tek aygıttan geçirmek istiyorsanız seçim yapınız.

"If you would like to allow all traffic from a device, select it below."

"eth0"

- Ek Dil Ayarları "Additional Language Support"

İngilizce (ABD) "*English* (USA)" Japonca "*Japanese*" Türkçe "*Turkish*"

- Zaman Bölgesi Seçimi *"Time Zone Selection"*

Avrupa / İstanbul *"Europe/Istanbul"*

- Root şifresi ayarlama *"Set Root Password"*

Root Password *****
Confirm *****

Sadece kendinizin bildiği bir root şifresi kullanınız.

- Örneğin adım Mehmet.
- Doğum günüm şubat 20.
- Şifrem mehmet0220
- Bu güvenli değil. Root şifre seçimi çok önemlidir.

- Paket Grubu Seçimi *"Package Group Selection"* Kurulum sırasında aşağıdaki paketlerinin kurulması için seçim yapılacaktır.

X Window sistem, GNOME masaüstü araçları, editörler, grafiksel internet, sunucu konfigürasyon araçları, Web sunucu (+ php-pgsql), mail sunucu, DNS ad Sunucu, Windows dosya sunucu, FTP sunucu, SQL DB sunucu, geliştirme araçları, çekirdek geliştirme, yönetim araçları, sistem araçları.

Seçim işlemi gerçekleştikten sonra "Next" "Sonraki" seçilecektir.

- Kurulum hakkında "About to Install" sonraki "Next".
- Gerekli medyaların kurulumu *"Required Install Media"* Devam *"Continue"*.
- Kurulum başladı "Installation is started".
- Açılış Disketi Oluşturulsun mu "Boot Diskette Creation" "No".
- Yeniden başla *"Reboot"*.
- Bu adımda, aşağıdaki mesajı içeren ekran gelecektir. Bu bölümler de Next diyerek geçilir.

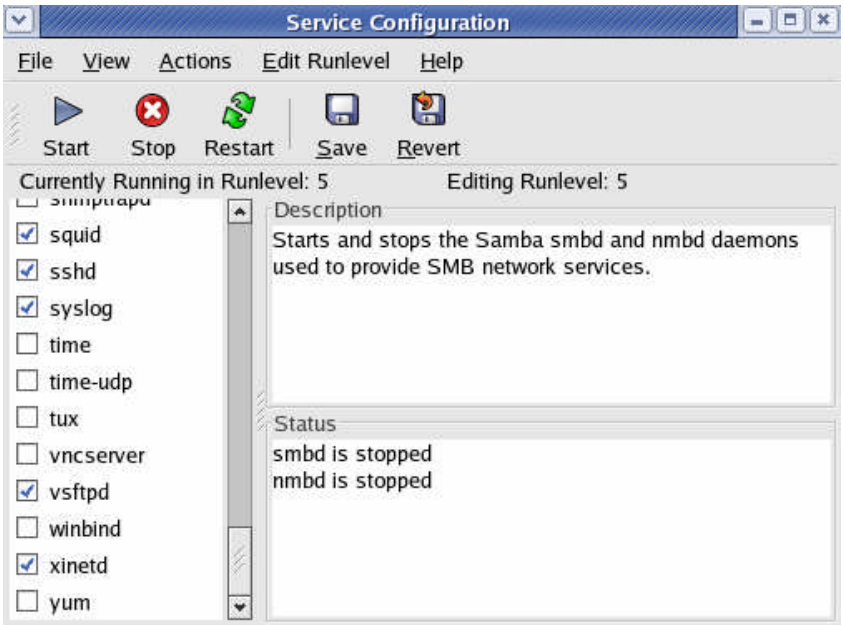
Hoşgeldiniz *"Welcome"* ->>> Lisans Anlaşması *"License Agreement"* evet *"Yes"* ->>> Tarih ve Zaman *"Date and Time"* Uygun Bir Değer Ayarlama *"Set an appropriate value."* ->>> Kullanıcı Hesabı *"User Account"* sonraki *"Next"* (Kullanıcı bir sonraki paragraftaki ayarları yapacak) ->>>Ses Kartı *"Sound Card"* sonraki *"Next"* ->>> Ek CD ler *"Additional CDs"* sonraki *"Next"* ->>> Kurulumu Bitir *"Finish Setup"* Sonraki *"Next"*

- Login (Bu bölüm daha önce verdiğimiz root şifresi ile geçilir.)

Username root
Password *****

- Services bölümünden aşağıdaki uygulamalar seçilir ve Sunucu yeniden başlatılır.

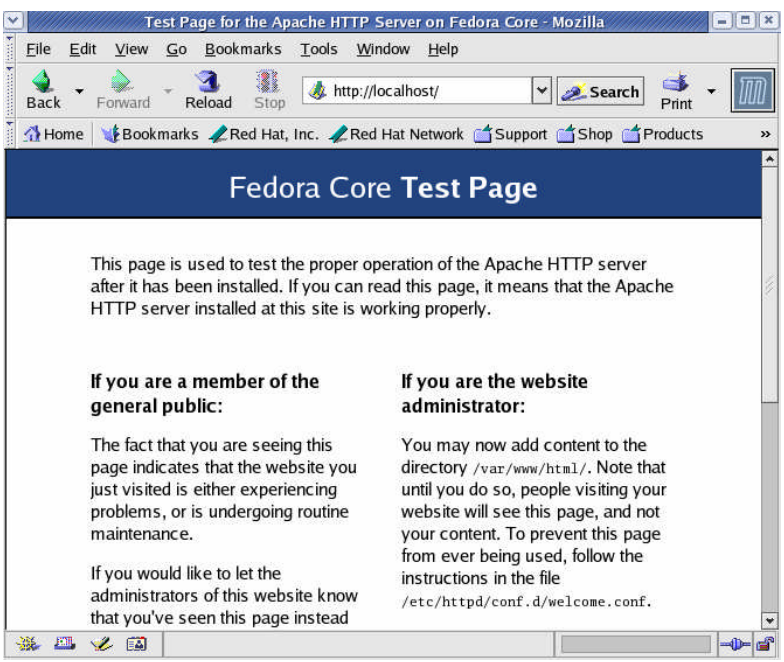
httpd, named, postgresql, sendmail, smb, squid, vsftpd



Şekil 1.4: Linux fedora services ekranı

1.3.3.2. Php Çalışma Testi

Bilgisayarda yer alan PHP modüllü web sunucusunun çalışıp çalışmadığını kontrol etmek için, öncelikle web tarayıcısı açılır. Ardından adres çubuğuna "*http://localhost*" yazılır. Eğer aşağıdaki gibi bir çıktı ile karşılaşılıyorsa web sunucusu çalışıyor demektir.



Şekil 1.5: Linux fedora apache web sunucu test ekranı

PHP modülünün doğru çalışıp çalışmadığını anlamak için takip eden program yazılır. Program yazımı için Linux'ta bulunan vi editör programını kullanılacaktır. Bu amaçla ayrıca Windows'taki not defteri programına benzer bir program olan gedit programı da kullanabilir. Dosya, "/var/www/html" klasörünün içinde oluşturulmalıdır. Tüm çalışmalar bu klasör içerisinde yapılacaktır.

Vi editörünü başlatmak için aşağıdaki komut kullanılabilir.

Not: Oluşturulacak dosyalarımızın uzantısı mutlaka php olmalıdır.

vi /var/www/html/test.php

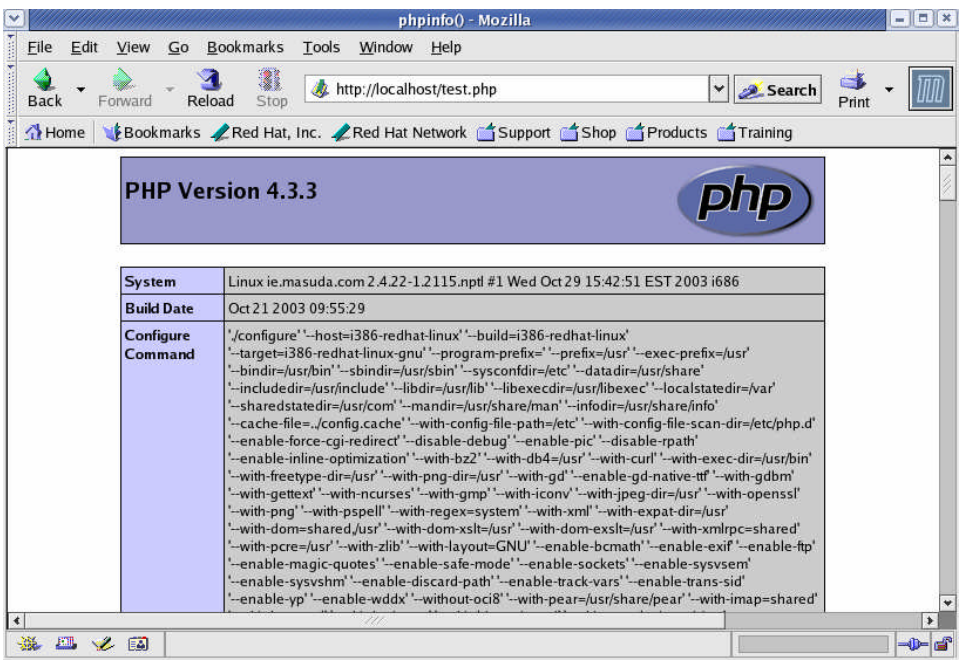
Dosya içerisine aşağıdaki kodlar yazılır.

```
<?php
        phpinfo();
?>
```

Kod 1.1: Php test programı

Aşağıdaki adres tarayıcının adres çubuğuna yazılır.

http://localhost/test.php



Şekil 1.6: Php test programı çıktısı

Görüldüğü üzere PHP test programı çalıştı. Çıktıda kullanılan PHP versiyonu görülmektedir. Yeni Fedora Linux versiyonlarında PHP 5 sürümü kullanılmaktadır.

Ek Çalışma

Windows İşletim Sistemi için: PHP ve Postgresql Modüllü Apache Web Sunucunun, Windows Kurulu Bir Bilgisayarda Çalıştırılması

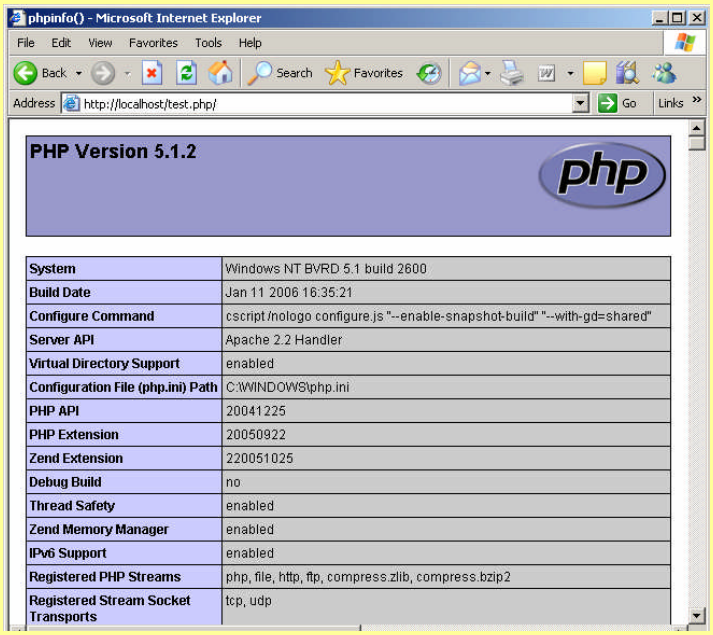
Bu modüller, bilgisayara ayrı ayrı kurabileceği gibi, bu amaçla geliştirilmiş hazır programlar da internetten indirip kullanılabilir. Bu çalışmada, Apache2triad programı kullanılacaktır. Bu program ile birlikte Windows İşletim Sistemi kurulu sisteme, Apache web suncu, PHP modülü, PostgreSQL veri tabanı sunucusu kurulacaktır. Bu program internetten ücretsiz olarak indirebilir. Programın simgesi aşağıdaki gibidir.

Program simgesi çift tıklanarak program bilgisayara kurulur. Kurulum sırasında şifreler istenecektir. Bunlar asla unutulmamalıdır. Kurulumda herhangi bir ekstra ayarlama yapmaya gerek yoktur. Sadece, kurulumda ilerideki uygulamalar için postgreSQL programının kurulup kurulmadığına dikkat edilmelidir.

Kurulum işlemi bittikten sonra, az önce Linux'ta yapılan "test.php" dosyası apache2triad klasörü içerisindeki htdocs klasörü içerisinde oluşturulmalıdır.

Web tarayıcısının adres çubuğuna aşağıdaki adres yazılır.

http://localhost/test.php



Ek Çalışma Şekil 1: PHP çalışma testi

Yukarıdaki çıktı, kurulumun başarılı olarak tamamlandığını göstermektedir. Bundan sonraki çalışmalar da test.php dosyasının bulunduğu klasör içerisine yapılacaktır. Eğer htdocs klasörü içerisinde klasör açarsanız adres ifadesi değişecektir. Örneğin htdocs klasörü içerisine okul klasörü oluşturup test.php dosyasını da bu klasör içerisine kopyalarsak adres aşağıdaki şekilde yazılmalıdır.

http://localhost/okul/test.php

Not : Çalışmalarnız için apache2triad benzeri programları da kullanabilirsiniz. Pogramın yapısına göre dosyalarınızı oluşturacağınız klasör değişebilir. Ayrıca Windows için geliştirilen IIS (internet Information Server) programını program ekle kaldır bölümünde yer alan windows bileşenlerinden kurabilirsiniz ancak IIS in php modülünü kendiniz eklemelisiniz. Bu bilgiyi internetten elde edebilirsiniz.

1.4. Betik Dilinin Temelleri

Bilgisayarda web sunucusu kurulumunu tamamladıktan sonra, php programı yazımına başlanabilir. Bu uygulamalarda Fedora Core içerisindeki Apache web sunucusu kullanıldığı için oluşturulan php dosyaları "/var/www/html" klasörü içerisine yazılır. Programlar bilgisayardaki işletim sisteminin durumuna uygun olarak gerekli klasöre yerleştirilmesi unutulmamalıdır. Örneğin Windows'ta apache2triad kurulu ise apache2triad klasörü içerisindeki htdocs klasörü kullanılacaktır.

1.4.1. Php'nin Başlangıç ve Bitiş İşaretçileri

Yazılan php kodları "<?PHP …. ?>" veya "<? …… ?>" işaretçileri arasına yazılmalıdır. PHP html kodları ile birlikte yazılabilir. HTML kodları içerisinde herhangi bir bölümde bu işaretçileri kullanarak PHP kodu yazılabilir.

Aşağıdaki ekrana "Merhaba Dünya" yazan programın kodları görülmektedir. Bu kodları uygulamak için dosyanın ismi merhaba php olacaktır.

Önemli Not: PHP de büyük küçük harf duyarlılığı vardır. Örneğin değişken ismi olarak $mesaj tanımlaması yapılmış ise $MESAJ veya $Mesaj bu değişkenden farklıdırlar.

vi /var/www/html/merhaba.php

```
1  <HTML>
2  <BODY>
3  <?PHP
4          echo "Merhaba Dünya";
5  ?>
6  </BODY>
7  </HTML>
```

Kod 1.2: "merhaba.php" programı

Programın açıklaması:

PHP ile HTML kodları iç içe kullanılabilir. Bu örnekte HTML kodları PHP kodlarından ayrı görülmektedir. Kodun 1. ve 2. satırında HTML kodları görülmektedir.

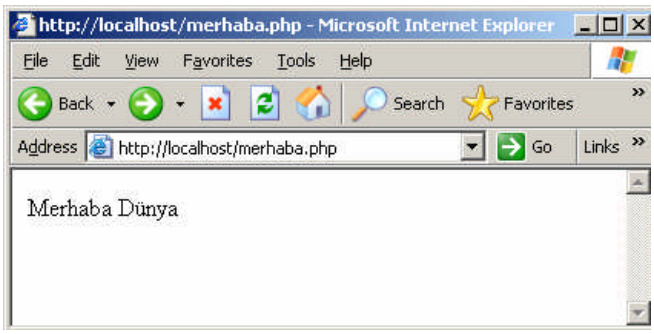
3. satırda php kodlarının başladığı belirtilmektedir.

4. satırda PHP'nin echo kodu ile web tarayıcının ekranına "Merhaba Dünya" yazılacaktır.

Not: PHP'de her kod satırında özel bir durum yok ise, (döngü ve koşullandırma hariç) mutlaka noktalı virgül ";" kullanılmalıdır.

5. satırda PHP kodlarının bittiği belirtilmekte. 6. ve 7. satırlarda ise daha önce başlatılan HTML ve BODY etiketleri sona ermektedir.

Bu aşamada, programın web tarayıcısında nasıl göründüğüne bakılacaktır. Bunun için web tarayıcısını açarak adres çubuğuna **"http://localhost/merhaba.php"** yazılır.



Şekil 1.7: "merhaba.php" ekran çıktısı

Eğer web tarayıcısının üzerinde farenin sağ tuşunu tıklayarak açılan "menüden kaynağı görüntüle" seçilirse html kodları görülebilir.

```
<HTML>
<BODY>
Merhaba Dünya</BODY>
</HTML>
```

Kod 1.3: Kaynağın görüntülenmesi

1.4.2. Echo Komutu

İstenilen yazıyı ya da değişkenleri ekrana yazdırmaya yarar (ayrıca bu komut içersine HTML komutları yazılabilir).

- **Kullanımı:**

```
echo "ekrana yazdırılacak ifade";
```

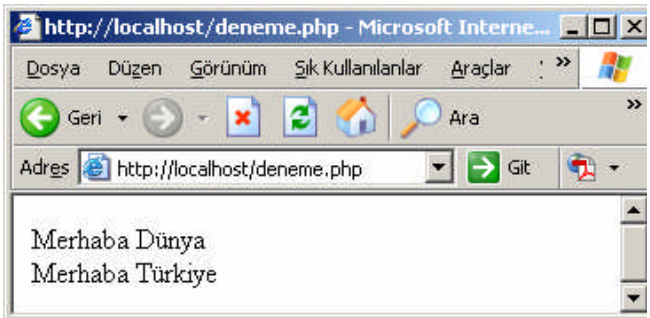
Örnek 1.1:

(düzyazı ve HTML etiketleri ile kullanım)

```
1  <?PHP
2  echo "Merhaba Dünya";
3  echo "<br> Merhaba Türkiye";
4  ?>
```

Kod 1.4: Php kodu içerisinde html kullanma

Programın ekran çıktısı aşağıdaki gibi olacaktır.



Şekil 1.8: Program çıktısı

Programın açıklaması:

2. satırdaki echo komutu içerisine düz bir metin yazılmış durumda, bu halde iken ekrana olduğu gibi "Merhaba Dünya" yazdırılacaktır. 3. satırdaki echo komutu içerisinde HTML etiketlerinden olan "<br>" komutu kullanılmıştır. Bu komut icra edilerek imlecin bir alt satıra geçmesi sağlanmıştır. Bu sayede "Merhaba Türkiye" yazısı bir alt satıra yazdırılacaktır. Ayrıca echo fonksiyonu parantez kullanılarak da yazılabilir.

Örnek 1.2:

(Değişkenlerin echo içerisinde kullanımı)1"Merhaba Dünya"

```
1  <?PHP
2  $mesaj="Merhaba Dünya";
3  echo ("Değişkendeki mesajınız : $mesaj");
4  ?>
```

Kod 1.5: "ech1.php" kodları

Şekil 1.9: Program çıktısı

Programın açıklaması ;

Görüldüğü gibi programın 2. satırında $mesaj adında bir değişken kullanılmıştır. Bu değişkenin içerisine de "Merhaba Dünya" kelime grubu atanmış. Bu türde olan değişkenleri de ekrana yazdırırken echo komutu içerisinde $mesaj olarak yazmamız yeterli olacaktır.

NOT: Eğer echo içerisinde bir dizi kullanıyorsak yazım kuralı değişir. Aşağıdaki örneği inceleyelim.

```
$dizi=array("deger1"=>"merhaba");
echo "dizi içindeki deger {$dizi ['deger1']} dir..";
```

Echo satırında "{ }" işaretlerine dikkat edilmelidir. Bu işaretleri kullanmanın amacı dizi içersinde kullanılan tırnak işaretlerinin echo komutunu ilgilendirmemesidir. Yani bu durumda daha önce yazdığımız değişken yazdıran programdan farksız duruma gelir.

Örnek 1.3:

(Echo komutu içerisinde uzun paragraflar yazılmak istendiğinde.**)**

```
echo <<<END
Artık demir almak günü gelmişse zamandan,
Meçhule giden bir gemi kalkar bu limandan.
Hiç yolcusu yokmuş gibi sessizce alır yol;
Sallanmaz o kalkışta ne mendil ne de bir kol.
Rıhtımda kalanlar bu seyahatten elemli,
Günlerce siyah ufka bakar gözleri nemli.
Biçare göüller. Ne giden son gemidir bu.
Hicranlı hayatın ne de son matemidir bu.
Dünyada sevilmiş ve seven nafile bekler;
Bilmez ki, giden sevgililer dönmeyecekler.
Bir çok gidenin her biri memnun ki yerinden.
Bir çok seneler geçti; dönen yok seferinden
END;
```

Kod 1.6: Echo komutu ile uzun paragraf yazımı

Eğer echo içerisinde uzun bir paragraf kullanılmak istenirse "echo" ifadesinden sonra paragrafa "<<END" ile başlanır. Paragraf sonunda ise "END" ifadesi kullanılır. "Echo" fonksiyonu gibi "print( )" ve "printf( )" fonksiyonları da kullanılabilir. Sonraki uygulamalarda echo ve print komutları kullanılacaktır.

1.4.3. Değişkenler

Bilindiği üzere değişkenler kayıtların (sayı veya karakter grubunun) içerisinde tutulduğu yapılardır. Değişkenler program süresince aktif haldedir. Program bittiği anda değişkenler silinir. PHP de değişken tanımlamaları otomatik olarak yapılır. Yani değişkenin türünü belirtme zorunluluğu yoktur. PHP de değişken önüne "$" işareti eklenir. Örneğin "$mesaj" mesaj adında değişkeni ifade eder. Yapılan atamaya göre değişken tipleri belirlenir. Değişken tipleri ve açıklamaları tabloda belirtilmiştir.

Tip	Php deki tip tanımlayıcılar	Örnek değerler
Mantıksal Veri Tipi	Bool	TRUE FALSE
Tamsayı Tipi	Int	10 20
Kayan Noktalı (ondalıklı) Tip	Float	10.0 20.00
Karakter katarı tipi	String	"10" '20'

Tablo 1.1: Değişken tipleri ve açıklamaları

Örnek 1.4:

```
<?  $mantiksal=TRUE;
    $tamsayi=20;
    $ondalik=10.25;
    $kkatari="Ahmet ZAPIR";

        print("Mantiksal = $mantiksal <br>");
        print("Tamsayi = $tamsayi <br>");
        print("Ondalikli ifade= $ondalik <br>");
        print("Karakter Katari = $kkatari<br>");
?>
```

Kod 1.7: Farklı değişken tiplerini yazdıran program

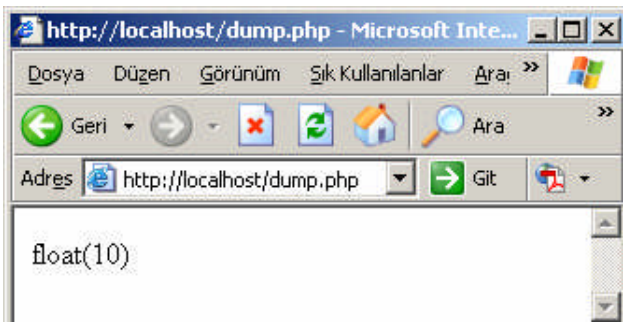
Şekil 1.10: Kod 1.9 program çıktısı

PHP de otomatik tip uygulamasının uygun olmadığı durumlarda zorunlu tip dönüşümü kullanılabilir.

```
<?
        $i=(float)10;
        var_dump($i);
?>
```

Kod 1.8: Değişken tipini gösteren program

Tip dönüşümü, C programlama dilinde olduğu gibi parantez "( )" içerisinde yazılarak yapılabilir. Buradaki var_dump fonksiyonu değişkenin tipini döndürmektedir.



Şekil 1.11: Kod 1.8 program çıktısı

Dönüştürme Kodu	Dönüştürme Sonrası Tip
(int)	Tam Sayı (Integer)
(float)	Kayan Noktalı, Ondalıklı (Floating point)
(string)	Karakter Katarı (Character string)
(array)	Dizi (Array)
(object)	Nesne (Object)

Tablo 1.2: PHP'de kullanılan tip dönüştürme komutları

1.4.4. Sabitler

Sabitleri değişkenlerden ayırt edebilmek için genellikle sabit isimleri büyük harfle yazılır. Sabit tanımlamak için "define()" kullanılır. Kullanımı aşağıdaki örnek kodlarda verilmiştir.

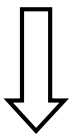
```
define("SABITIM",100);

define("SABITSRTINGIM","deneme");
```

Bir sabit, değişken gibi de kullanılabilir ancak sabitin başına $ eklemek koşulu ile. $SABITIM ve SABITIM birbirinden tamamen farklı şeylerdir.

1.4.5. Argümanlar

PHP'de program içerisine bir argüman (parametre) göndermek istenirse, argümanlar adres çubuğunda URL bölümüne eklenmelidir. Argümanlar aşağıdaki URL yapısındaki gibi verilir.

http://localhost/add.php?sayi1=11&sayi2=21



argüman

PHP programlarında argüman değerlerini değişkene atmak için $_GET kullanılır.

Örneğin: sayi1=11

11 değerini bu diziden almak için $_GET ["sayi1"] kullanılır.

GET metodunun kullanım alanı ve kullanım koşulları form kullanımı konusunda anlatılacaktır.

Argümanların içinden GET metodu ile değişkenleri alarak, gönderilen iki sayıyı toplayan program:

```
<?
        if(isset($_GET["sayi1"])==FALSE){
                $sayi1=10;
        }else{
                $sayi1=$_GET["sayi1"];
        }

        if(isset($_GET["sayi2"])==FALSE){
                $sayi2=20;
        }else{
                $sayi2=$_GET["sayi2"];
        }

        $sonuc=$sayi1 + $sayi2;
        print("$sayi1 + $sayi2 = $sonuc");
?>
```

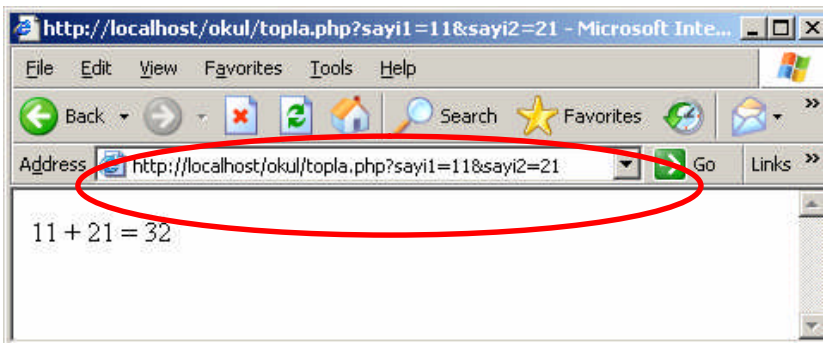
Kod 1.9: URL' den alınan argümanları toplatan program

Programın açıklaması ;

if(isset($_GET["sayi1"])==FALSE) ifadesinden anlaşılan sayi1 argümanının URL'ye eklenmemiş olmasıdır. If koşul ifadesi daha sonraki konularda açıklanacaktır.

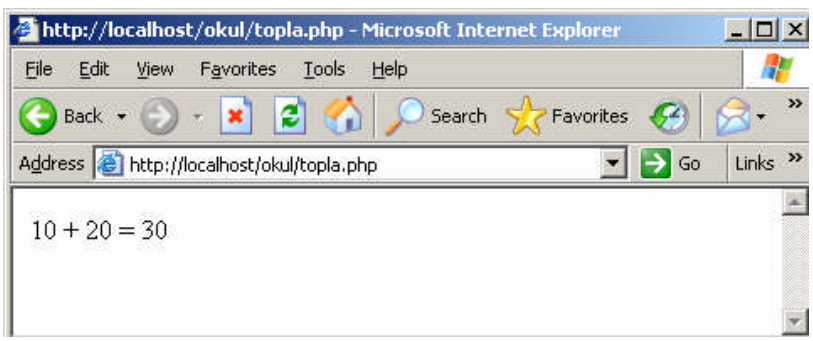
*sayi1=$_GET["*sayi1*"]* ifadesi ile argüman yumağı içerisinden sayi1 değişkeninin alındığı belirtilmektedir.

Program "topla.php" adı ile kaydedilip çalıştırılır.



Şekil 1.12: "topla.php" program çıktısı

Eğer URL'de herhangi bir argüman gönderilmez ise oluşacak çıktı aşağıdaki gibidir.



Şekil 1.13: "topla.php" program çıktısı (argümansız)

Yazılan programda argüman gönderilmediğinde sayi1 10'a eşitleniyor, sayi2 ise 20'ye eşitleniyor. Dolayısıyla sonuç 30 olarak bulunuyor.

UYGULAMA FAALİYETİ

Aşağıdaki uygulama faaliyetini işlem basamaklarına uygun olarak yapınız.

- Bir web sayfası içerisinde adres çubuğunda verilen beş sayıyı çarparak ekrana sonucu yazdıran PHP programını yapınız.

İşlem Basamakları	Öneriler
➢ PHP modülü olan web sunucu kurunuz. ➢ Programınızı yazarak uygun klasöre kaydediniz. ➢ Programınızı web tarayıcınızdan çağırınız.	➢ Web sunucunuzun PHP modülü olup olmadığını kontrol etmek için küçük bir program ile test ediniz. ➢ Programınızı öncelikle kâğıt üzerinde yazınız. ➢ Program hatalı ise tarayıcı herhangi bir çıktı görünmeyecektir. Buna dikkat ediniz. ➢ PHP'nin başlangıç ve bitiş etiketlerine "işaretçilerine" dikkat ediniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki sorulara uygun şıkları bularak cevap veriniz.

1. Aşağıdakilerden hangisi PHP'nin başlangıç ve bitiş etiketleridir?

A) " <? ?>" C) "<% %>"
B) "<PHP PHP>" D) "/? ?\"

2. Aşağıdakilerden hangisi tarayıcıda (browser) ekrana yazı yazdıran komuttur?

A) lpt C) write
B) echo D) yaz

3. Aşağıdakilerden hangisi PHP dilinin bir özelliği değildir?

A) Yüksek Performanslı Apache Modülünün olması.
B) Nesne- Yönelimli Olması
C) Yazım düzeninin kolay olması
D) Mutlaka derleme gerektirmesi.

4. Aşağıdakilerden hangisi PHP'de bir değişken olabilir?

A) &mesaj C) #mesaj
B) $mesaj D)?mesaj

5. Aşağıdaki değişken değerlerinden hangisi mantıksaldır?

A) $degisken="DOGRU" C) %degisken=LOGIC
B) &degisken=FALSE. D) $degisken=TRUE

6. Hangi komut ile değişkenin türü ekrana yazdırılabilir?

A) echo( ) C) (float)
B) printf ( ) D) var_dump( )

7. Adres çubuğunda görünen argümanları alabilmek için hangi komut kullanılır?

A) $_SET C) $_LET
B) $_GOT D) $_GET

DEĞERLENDİRME

Cevaplarınızı cevap anahtarı ile karşılaştırınız. Doğru cevap sayınızı belirleyerek kendinizi değerlendiriniz. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt yaşadığınız sorularla ilgili konuları faaliyete geri dönerek tekrar inceleyiniz.

ÖĞRENME FAALİYETİ–2

AMAÇ

Betik dili ile sunucu taraflı programlarda döngü ve koşulları hatasız bir şekilde kullanabileceksiniz.

ARAŞTIRMA

Bu öğrenme faaliyetinden önce aşağıdaki hazırlıkları yapmalısınız.

- Nesne yönelimli programlama konusunu araştırınız.

2. KOŞULLAR VE DÖNGÜLER

2.1. If Yapısı

If komutunun kullanımı c dilindeki kullanıma benzerdir. If içerisindeki koşul doğru ise bir altındaki program bölümü çalıştırılacak, eğer koşul sağlanmıyor ise else ile çevrelenmiş program kodları çalışacaktır. Else ifadesinden sonra if eklenerek bir başka durumda sorgulanabilir. Kullanımı ile ilgili yazım kuralı aşağıda verilmiştir.

```
if(durum 1){
        Durum 1 doğrulandığında çalışacak bölüm
    }else if(durum 2){
        Durum 2 doğrulandığında çalışacak bölüm
        .
        .
        .
    }else{
        Bu bölüm tüm durumlar sağlanmadığında çalışacaktır.
}
```

Örnekler :

- if($sayi1>10){

echo " sayi1 değişkeni 10 dan büyüktür.";
}

- if($sayi1>10){
echo “sayi1 değişkeni 10 dan büyüktür.”;
} else {
echo “ sayi1 değişkeni 10 dan küçük ya da eşittir.”;
}

- if ($sayi1>10){
echo “sayi1 değişkeni 10 dan büyüktür.”;
} else if($sayi1>5){
echo “ sayi1 değişkeni 10 dan küçük ya da eşit ve 5 ten büyüktür.”;
} else {
echo “ sayi1 değişkeni 5 ten küçük ya da eşittir.”;
}

if ifadesi içerisinde mantıksal “ve” ile “veya” kullanarak birden fazla koşul sorgulanılabilir. “ve” ifadesi “&&” ile simgelenirken “veya” ifadesi ise “||” simgeleri ile ifade edilir.

if($sayi1>10 && $sayi<20) bu ifadede, sayi1 değişkeninin 10’dan büyük olması **ve** sayi2 değişkeninin de 20’den küçük olması durumunda işletilecek demektir. Yani her iki koşul doğru olduğunda if bloğu içerisindeki komutlar icra edilecektir.

if($sayi1>10 || $sayi<20) anlamı sayi1 değişkeni 10’dan büyük olması veya sayi2 değişkeninin 20’den büyük olması durumunda if bloğu icra edilecektir. Yani iki koşuldan biri bile geçerli olsa if bloğu icra edilecektir.

Yukarıdaki ve, veya’lar ile yapılan zincirleme sorguların arttırılması mümkündür.

If fonksiyonundaki karşılaştırma operatörleri C dilindeki gibidir. Karşılaştırma operatörleri ve anlamları aşağıdaki tabloda verilmiştir.

Eşitlik Operatörleri	= = !=	İki tarafta eşit olduğunda doğru İki tarafta eşit olmadığında
İlişkisel operatörler	> < >= <=	Sol taraf sağ taraftan daha büyük olduğunda doğru Sağ taraf sol taraftan daha büyük olduğunda doğru Sol taraf sağ tarafa eşit veya büyük olduğunda doğru Sağ taraf sol tarafa eşit veya büyük olduğunda doğru

2.2. Switch-Case Yapısı

Switch yapısı uzun if ifadeleri gerektiği yerlerde tercih edilir. Birden çok durumun karşılaştırılması gereken yerlerde kullanışlı bir yapıdır. Kullanımı aşağıdaki gibidir.

```
switch(değişken){
case deger1:
        komutlar...;
        break;
case deger2:
        komutlar;
        break;
case deger3:
        komutlar;
        break;
default:
        komutlar;
        break;
}
```

Switch ile ilgili örnekler yapalım.

Örnek 2.1:

Bu örnekte, 1 ile 5 arasında girilen sayının yazı karşılığını gösteren program, switch-case yapısı kullanılarak yazdırılacaktır.

```
<?
$sayi=4;
switch($sayi){
        case 1:
                echo "BİR";
                break;
        case 2:
                echo "İKİ";
                break;
        case 3:
                echo "ÜÇ";
                break;
        case 4:
                echo "DÖRT";
                break;
        case 5:
                echo "BEŞ";
                break;
        default :
                echo "girilen sayi 1 ile 5 arası değildir.";
                break;
}
?>
```

Kod 2.1: Switch-case yapısı ile ilgili program

Bu bölümde break ifadesi eğer karşılaştırma doğru ise diğer seçeneklere lüzum kalmaması içindir. Duruma göre break kullanılmayabilir.

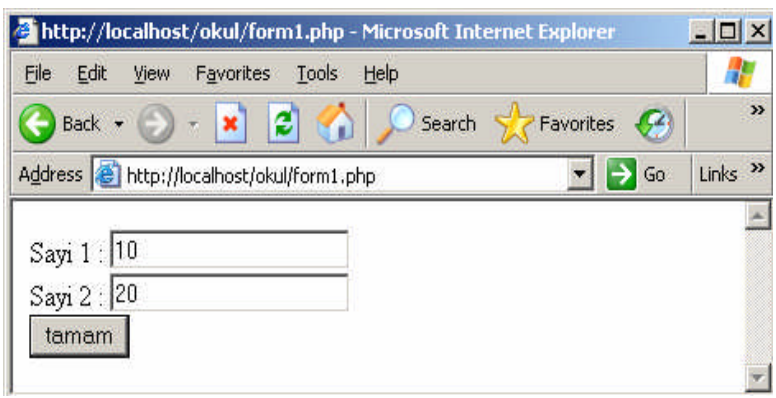
2.3. Sayfalar Arası Argüman İletimi (Get ve Post Metodu)

PHP'de sayfalar arası veri gönderimi diğer betik dillerinde olduğu gibi Get ve Post metodu ile yapılır. Aşağıdaki örnekte olduğu gibi HTML kodları içinde yer alan <form> başlangıcı ile </form> bitiş etiketleri arasındaki nesneler (metin kutusu adı ve değeri, buton adı ve değeri action ile belirtilen sayfaya iletilir. Action özelliği kullanılmaz ise sayfa yine kendine argümanları gönderir. Aşağıda örnek form kodu görülmektedir.

```
<HTML>
<BODY>
<form method=GET>
Sayi 1 : <input type=text name=sayi1><br>
Sayi 2 : <input type=text name=sayi2><br>
<input type=submit name=tamam value=tamam>
</form>
</BODY>
</HTML>
```

Kod 2.2: "form1.php" programı get metodu kullanımı

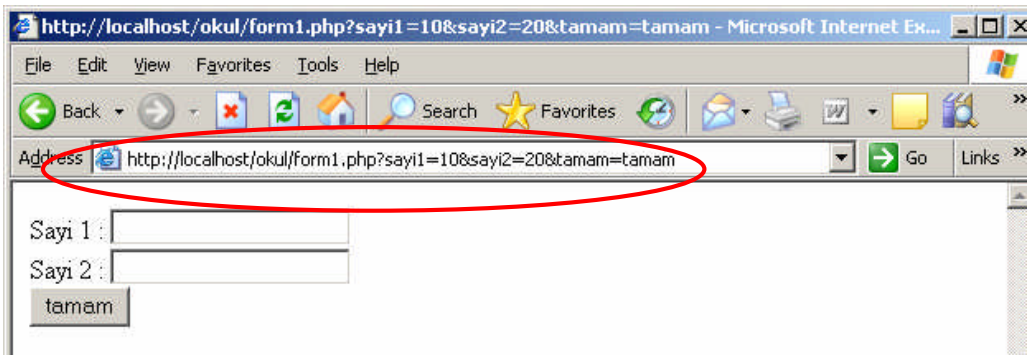
Yukarıdaki kod form1.php adıyla kaydedilip ve çalıştırılır.



Şekil 2.1: "form1.php" program

Görüldüğü gibi, sayi1 adlı metin kutusuna 10, sayi2 kutusuna da 20 sayısı girilir. Tamam butonuna basıldığında, bu metin kutular içerisindeki sayılar herhangi bir action özelliği kullanılmadığı için aynı sayfaya GET metodu kullanılarak gönderilecektir.

Bir sonraki şekilde web tarayıcının adres çubuğuna dikkat edilmelidir.



Şekil 2.2: "form1.php" programı tamam tıklandıktan sonra

Görüldüğü üzere adres çubuğunda argümanlar otomatik olarak sıralandı. Aslında bu argümanlar kullanıcıdan sunucuya gönderildi. Argümanları ve değişkenleri tek tek inceleyelim.

http://localhost/okul/form1.php?sayi1=10&sayi2=20&tamam=tamam
1. arg. 2. arg. 3. argüman

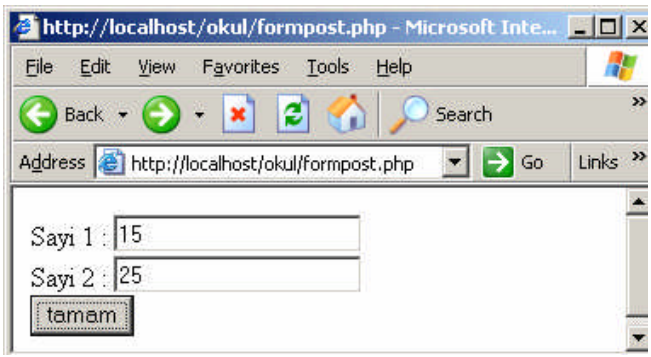
Argümanlar URL'de "?" işaretinden sonraki bölümdür. Burada sayi1 metin kutusundan gönderilen değerin 10 olduğu görülmektedir. "&" işaretinden sonra diğer bir argüman olan sayi2 değeri 20 olarak iletilmiş durumdadır. Son olarak da butonun değeri, tamam argümanı içerisinde, value değeri olarak "tamam" değeri gönderilmiştir.

Argümanları gönderirken POST metodunu kullanmak için aynı program kodunun form etiketinin içerisine metot olarak POST verilir.

```
<HTML>
<BODY>
<form method=POST>
Sayi 1 : <input type=text name=sayi1><br>
Sayi 2 : <input type=text name=sayi2><br>
<input type=submit name=tamam value=tamam>
</form>
</BODY>
</HTML>
```

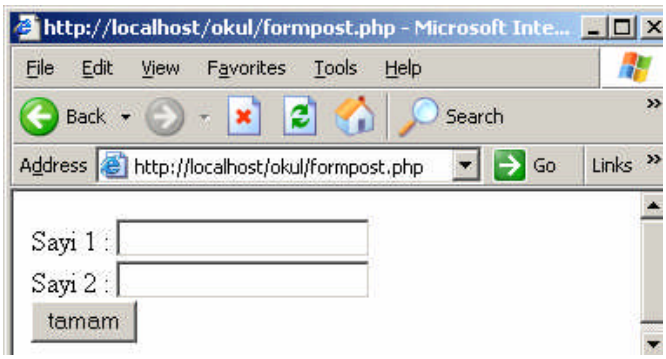
Kod 2.3: "formpost.php" programı post metodu kullanımı

POST metodu kullanıldığında argümanlar yine gönderilecek fakat bu defa gönderilen argümanlar adres çubuğunda gösterilmeyecektir. Program formpost.php olarak kaydedilir.



Şekil 2.3: "formpost.php" programı ilk ekran

Tamam butonu tıklandığındaki durum:



Şekil 2.4: "formpost.php" programı tamam

Görüldüğü gibi, argümanlar bu defa adres çubuğuna aksettirilmedi. Ancak yine de argümanlar iletilmiş oldu. Bu defa, değerleri almak için $_POST ifadesi kullanılacaktır.

2.3.1. Get ve Post Metodu Arasındaki Farklar

Sayfalar arasında iletişimde kullanılan GET ve Post Metodunun farklılıkları şunlardır.

- GET metodu ile gönderilen argümanlar kullanıcı tarafından adres çubuğunda görülebilir. Ancak bu durum POST metodu için geçerli değildir. Bazı durumlarda POST metodunun argümanlarının görüntülenmemesi güvenlik amaçlı kullanılabilir.

- GET metodu ile gönderilen argümanlar $_GET ile alınırken POST metodu kullanılarak gönderilen argümanlar ise $_POST ile alınabilir.

- GET metodu ile gönderilebilecek argümanların sayısı 255 karakter ile sınırlıdır. Ancak POST metodu ile gönderilecek argüman sınırı yoktur.

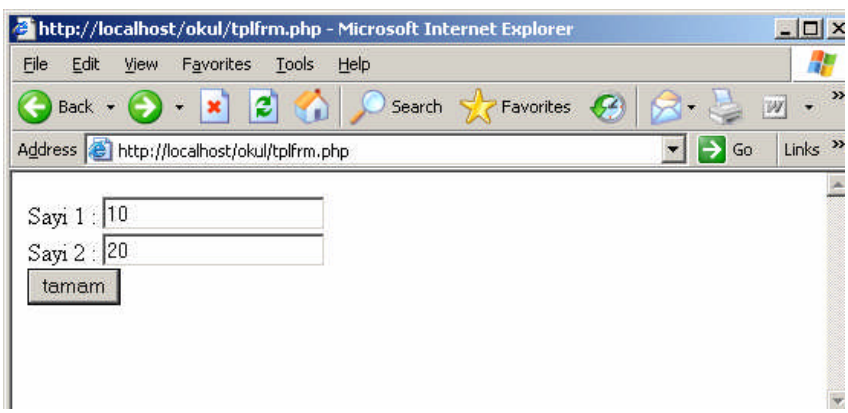
2.3.1.1. Get ve Post Metodu İle İlgili Örnekler

Örnek 2.2:

Örnekte, birinci sayfada girilen iki sayı, GET metodu kullanılarak yine aynı sayfa içerisinde toplatılmaktadır.

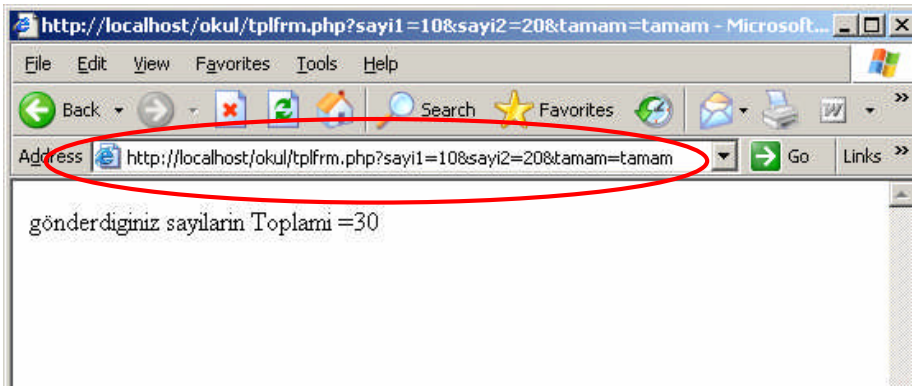
```
<HTML>
<BODY>
<? if(isset($_GET[sayi1])==FALSE){
echo $_GET[tamam];?>
<form method=GET>
Sayi 1 : <input type=text name=sayi1><br>
Sayi 2 : <input type=text name=sayi2><br>
<input type=submit name="tamam" value="tamam">
</form>
<?
} else {
$sayi1=$_GET[sayi1];
$sayi2=$_GET[sayi2];
$toplam=$sayi1+$sayi2;
echo "gönderdiğiniz sayıların Toplamı =$toplam";
}
?>
</BODY>
</HTML>
```

Kod 2.4: Get metodu kullanılarak yapılan toplama programı "tplfrm.php"



Şekil 2.5: "tplfrm.php" program çıktısı

Sayılar girildikten sonra tamam butonu tıklanır.



Şekil 2.6: "tplfrm.php" program çıktısı

Herhangi bir argüman girilmediği zaman sadece metin kutuları ile tamam düğmesi görüldü. Metin kutularına rakamlar girildiğinde ve tamam butonuna tıklandığında else bölümünde belirtilen kodlar çalıştırıldı.

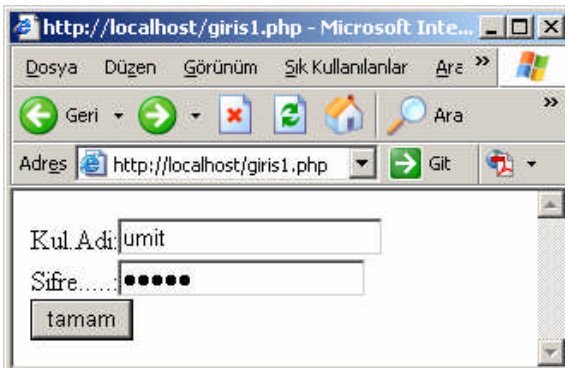
Örnek 2.3:

İki metin kutusunun birine kullanıcı adı, diğerine de şifre girilecektir. Giriş düğmesine basıldığında kullanıcı adı ve şifre kontrol edilecek, ikisi de doğru ise "login oldunuz" eğer yanlış ise "login olamadınız" ifadesi yazdırılacaktır.

```
<HTML><BODY>
<? if(isset($_POST[tamam])==FALSE){
echo $_GET[tamam];?>
<form method=POST>
Kul.Adi:<input type=text name=kullanici><br>
Sifre.....:<input type=password name=sifre><br>
<input type=submit name="tamam" value="tamam">
</form>
<? } else {
$kulad=$_POST[kullanici];
$pass=$_POST[sifre];
if($kulad=="umit" && $pass=="cihan") {
        echo "Başarı ile Login oldunuz";
}else{
        echo "Login olamadınız…";
}} ?>
</BODY></HTML>
```

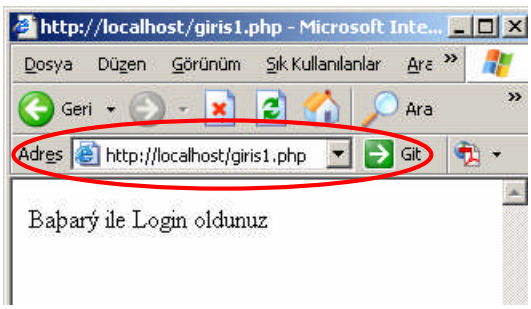
Kod 2.5: Post metodu kullanılarak yapılmış program "giris1.php"

Programın çalıştırılması:



Şekil 2.7: "giris1.php" program çıktısı

Kullanıcı adı ve şifre girildiğinde diğer sayfaya POST metodu ile gönderilecektir. Ancak GET metodunda olduğu gibi argümanlar URL'ye eklenmeyecektir.



Şekil 2.8: "giris1.php" tamam tıklandıktan sonra

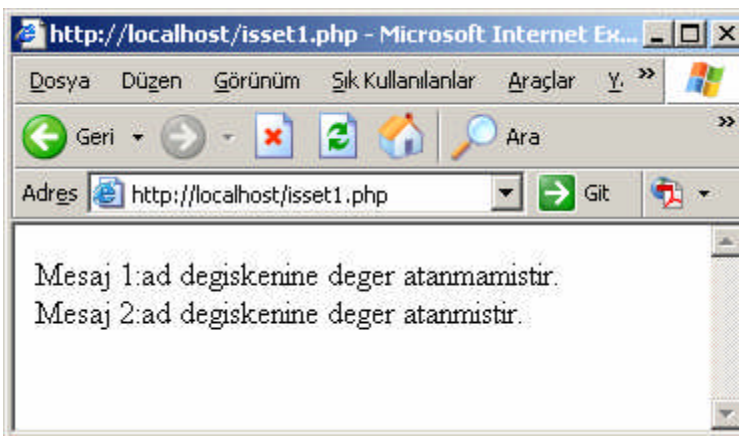
2.3.1.2. Isset Fonksiyonu

Şu ana kadar yapılan sayfalar arası veri iletimi yapan programlarda, isset fonksiyonu kullanılmıştır. Aşağıda isset fonksiyonunun ne işe yaradığı bir programla gösterilmiştir. Program kodları isset1.php dosyası olarak kaydedilir.

```
<?php
$ad=NULL;
if(!isset($ad)){
        echo "Mesaj 1:ad degiskenine deger atanmamistir.";
} else {
        echo "Mesaj 1:ad değiskenine deger atanmistir.";
}
$ad="bulent";
if(!isset($ad)){
        echo "<br>Mesaj 2:ad degiskenine deger atanmamistir.";
} else {
        echo "<br>Mesaj 2:ad degiskenine deger atanmistir.";
}
?>
```

Kod 2.6: Isset fonksiyonun kullanımı ile ilgili örnek

Çıktısı da şekildeki gibi olacaktır.



Şekil 2.9: "isset1.php" program çıktısı

Değişken içerisine bir atama yapılmadığı zaman (NULL olduğu zaman), isset fonksiyonundan FALSE değeri döner. Değişken içerisine değer atandığında (programda "bulent") isset fonksiyonundan dönen değer TRUE olur. isset fonksiyonu tam olarak değişken içerisine değer atanıp atanmadığını gösteren bir fonksiyondur. Bu fonksiyon sonraki örnekler için sık sık kullanılacaktır.

2.4. Döngü Yapısı

PHP'de bir işlemi istenilen sayıda tekrarlamak veya belirli koşullara bağlı olarak işlemleri tekrarlattırmak amacı ile döngüler kullanılır. PHP'de C dilinde kullanılan döngü yapıları kullanılabilir. Bu döngü çeşitleri ve kullanım şekilleri aşağıda tek tek incelenecektir.

2.4.1. While Döngüsü

While döngüsü parantez içerisindeki koşul ya da koşullar grubu doğru olduğu sürece çalışan döngü yapısıdır. Kullanım şekilleri aşağıda verilmiştir.

Kullanım şekli:

```
While(Koşul ya da koşullar grubu){
        Kodlar…….
}
```

Örnek 2.4:

```
<?
while(1){
        echo “merhaba”;
} ?>
```

Kod 2.7: While ile sonsuz döngü

Bu döngü sonsuz bir döngüdür. Parantez içerisinin 1 olması koşulun sürekli olarak doğru olduğu anlamına gelir. Bu kod parçası çalıştırıldığında web tarayıcımızda sürekli olarak merhaba yazılacaktır. Bu tip döngülerden çıkmak için her hangi bir koşul içerisinde exit komutu kullanılabilir.

Örnek 2.5:

```
<?
$a=0;
While($a<10){
        echo “$a <br>”;
        $a++;
}
?>
```

Kod 2.8: While ile koşullu döngü

Yukarıdaki kodda, döngü $a değişkeni 10’dan küçük olduğu sürece tekrarlanacaktır. Görüldüğü üzere while döngüsü içerisinde $a++ ifadesi ile $a değişkeni sürekli bir arttırılmaktadır.

Örnek 2.6:

```
<?
$a=0;
$b=20;
While($a<10 || $b>=0){
        echo “a sayisi=$a <br>”;
    echo “b sayisi=$b”;
        $a++;
    $b- -;
}
?>
```

Kod 2.9: While ile çoklu koşullu döngü

Yukarıdaki örnekte döngü $a değişkeni 10’dan küçük olduğu sürece veya $b değişkeni 0’dan büyük olduğu sürece işletilecektir. Döngü içerisinde $b- - ifadesi ile $b değişkeni sürekli azaltılacaktır. Bu durumda, döngü, $b değeri 20 oluncaya kadar tekrar edilecektir.

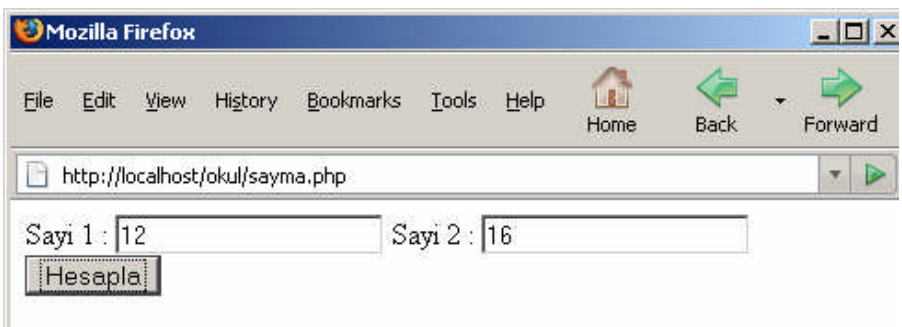
While ifadesi içerisinde koşullar aynen if yapısındaki gibi “||” veya “&&” bağlacı ile birbirine bağlanabilir. Burada “||” bağlacı mantıksal veya (or) yı ifade eder, “&&” bağlacı ise mantıksal ve (and) işlemini görür.

Metin kutularına girilen iki sayının, küçük olanından büyük olanına kadar olan tam sayıları, tamam butonuna bastıktan sonra ekrana yazdıran program “while” döngüsü ile, aşağıda gösterildiği gibi yapılabilir.

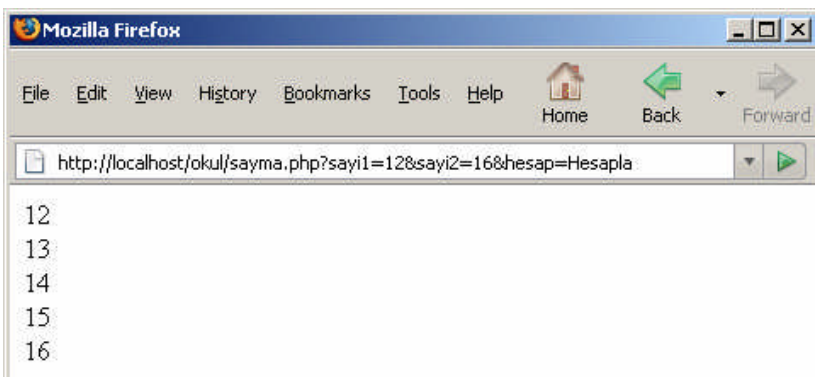
Program adı : “sayma.php”

```
<HTML><BODY>
<?
if(isset($_GET[sayi1])==FALSE){
?>
        <form> Sayi 1 : <input type=text name=sayi1>
          Sayi 2 : <input type=text name=sayi2>
          <br><input type=submit name=hesap value=Hesapla>
          </form>
<?}else{
          $a=$_GET[sayi1];$b=$_GET[sayi2];
          if($a>$b){
                    $b_sayi= $a;$k_sayi=$b;
          }else{
                    $b_sayi=$b;$k_sayi=$a;
          }
          while($k_sayi<=$b_sayi){
                    echo "$k_sayi <br>";$k_sayi++;
}}  ?></BODY></HTML>
```

Kod 2.10: While ile çoklu koşullu döngü



Şekil 2.10: "sayma.php" program çıktısı 1



Şekil 2.11: "sayma.php" program çıktısı 2

2.4.2. For Döngüsü

For döngüsü sıkça kullanılan döngülerdendir. For döngüsünün kullanım şekli aşağıdaki gibidir.

```
for (başlangıc değeri ; koşul ; işlem){
        Kodlar….
}
```

For döngüsünün farklı kullanım şekilleri aşağıda gösterilmiştir.

Örnek 2.7:

```
for( $i=0;$i<10;$i++){
        echo “$i <br>”;
}
```

Kod 2.11: For döngüsü

Koddaki döngüde $i sayısı 0’dan başlayarak 10’dan küçük olduğu sürece birer arttırılacaktır. Sonuçta echo komutu sayesinde ekrana yazdırılacaktır.

Örnek 2.8:

```
for( $i=10;$i>0;$i- -){
        echo “$i <br>”;
}
```

Kod 2.12: Azalan for döngüsü

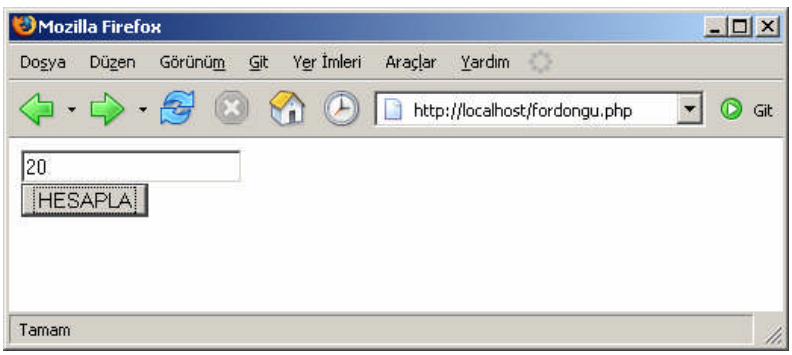
Yukarıdaki kod parçasında ise $i sayısının başlangıç değeri 10 olacak ve ardından $i değişkeni 0’dan büyük olduğu sürece $i sayısından birer eksiltilecektir. Echo komutu ile sonuçlar alt alta yazdırılacaktır.

Örnek 2.9:

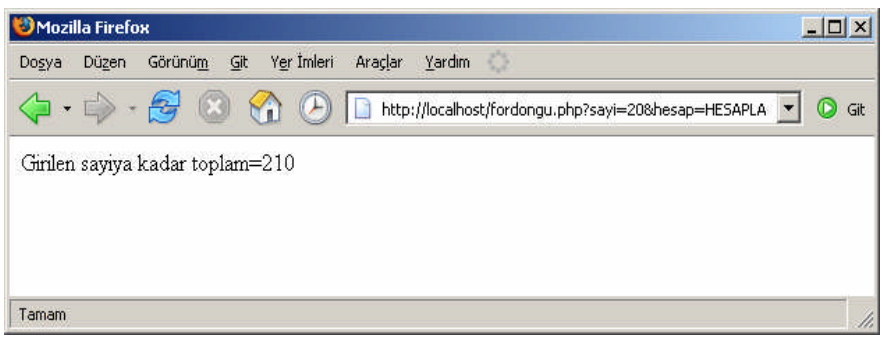
Bu örnekte klavyeden girilen bir sayıya kadar olan sayıların toplamı, “hesapla” butonuna tıklandığında yazdırılmaktadır.

```
<HTML>
<BODY>
<? if(isset($_GET[hesap])==FALSE){ ?>
<form><input type=text name=sayi>
<br><input type=submit name=hesap value=HESAPLA>
</form>
<? }else{
        $sayi=$_GET[sayi];
        for($i=1;$i<=$sayi;$i++){
                $toplam=$toplam+$i;
        }
echo "Girilen sayiya kadar toplam=$toplam";
}?>
```

Kod 2.13: "fordongu.php" program kodları



Şekil 2.12: "fordongu.php" program çıktısı 1



Şekil 2.13: "fordongu.php" program çıktısı 2 (Hesapla düğmesi tıklanınca)

2.5. Diziler

Diziler indisli değişkenlerdir. Aynı isimde toplanmış değişkenler kümesi olarak da açıklanabilir. Örneğin öğrencilerin numaralarını tutan bir dizi olabilir. Diziler indisli

değişkenler olduğu için bir döngü içerisinde sıralama, ortalama bulunması gibi işlemler kolaylıkla yapılabilir. PHP'de tek boyutlu veya iki boyutlu diziler oluşturulabilir.

PHP'de bir dizi oluşturmak için aşağıdaki komut dizisi kullanılabilir. Dizi eleman sayısı [ ] simgesi içerisine yazılır.

1. Yol array( ) kullanarak dizi oluşturma,

```
$dizimiz=array(veri1, veri2, veri3 …… veriN);
```

2. Yol Dizi elemanları, değişken atamadaki gibi verilerek de dizi oluşturulmuş olur. PHP otomatik olarak girilen her veriyi bir sonraki indise ekler.

```
$dizimiz[]=veri1;
$dizimiz[]=veri2;
        .
        .
        .
$dizimiz[]=veriN;
```

3. Yol İndis numarası yerine anahtar kullanımı:

```
$dizimiz[anahtar1]=veri1;
$dizimiz[anahtar2]=veri2;
        .
        .
$dizimiz[anahtarN]=veriN;
```

4. Yol array ve anahtar kullanarak oluşturma:

```
$dizimiz=array("anahtar1"=>veri1,"anahtar2"=> veri2,  ……
"anahtarN"=>veriN);
```

Yukarıdaki yolları kullanarak tek boyutlu dizi oluşturulabilir. İki boyulu dizi aşağıdaki gibi oluşturulabilir.

```
$dizimiz[indis][indis]=veri;
$dizimiz[indis][indis]=veri;
```

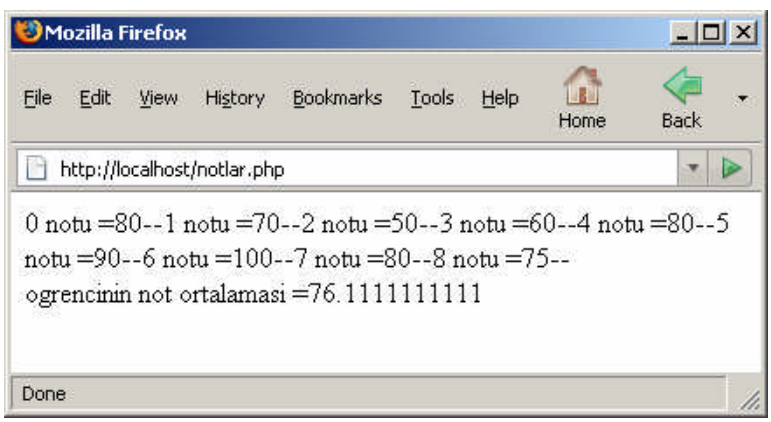
Örnek2.10:

Bu uygulamada bir öğrenciye ait 9 adet notun ortalaması buldurulacaktır. Programın ismi notlar.php olsun.

```
<HTML><BODY>
<?
$notlar=array(80,70,50,60,80,90,100,80,75);
$toplam=0;
for($i=0;$i<9;$i++){
        echo "$i notu =$notlar[$i]--" ;
        $toplam=$toplam+$notlar[$i];
}
$ortalama=$toplam/9;
echo "<br> ogrencinin not ortalamasi =$ortalama";
?>
</BODY></HTML>
```

Kod 2.14: "notlar.php" program kodları

Programın çıktısı şekildeki gibi olacaktır.



Şekil 2.14: "notlar.php" program çıktısı

Örnek2.11:

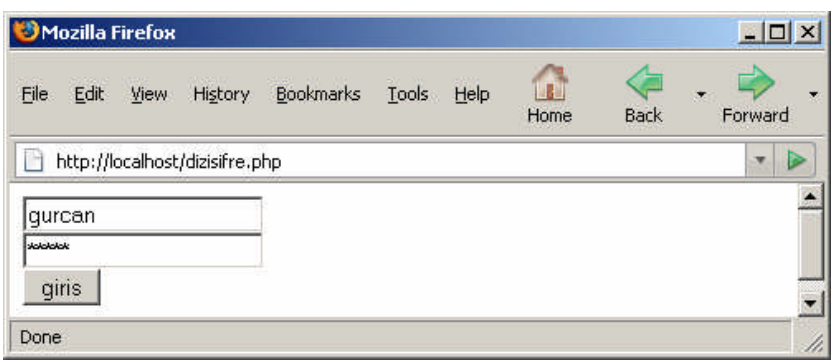
Bu uygulamada dizi ile ilgili bir çalışma yapılacaktır. Programda kullanıcı adları dizide anahtar olarak girilecek, veri olarak ise şifreler verilecektir. Kullanıcı adına karşılık

gelen şifre doğru ise ekrana “kullanıcı adı ile şifreniz uyumludur” şeklinde mesaj yazdırılacaktır. Program, dizisifre.php olarak kaydedilir.

```
<HTML><BODY>
<?
if(isset($_GET[ok])==FALSE){
        echo "<form>";
        echo "<input type=text name=kulad>";
        echo "<br><input type=password name=sifre>";
        echo "<br><input type=submit name=ok value=giris>";
        echo "</form>";
}else{
        $bilgi=array("gurcan"=>"12345","neslihan"=>"54321");
        $kulad=$_GET[kulad];
        $sifre=$_GET[sifre];
        if($bilgi[$kulad]==$sifre)
                echo "kullanici adi ve sifre dogru";
        else
                echo "kullanici adi ya da sifre yanlis.";
}
?>
</BODY></HTML>
```

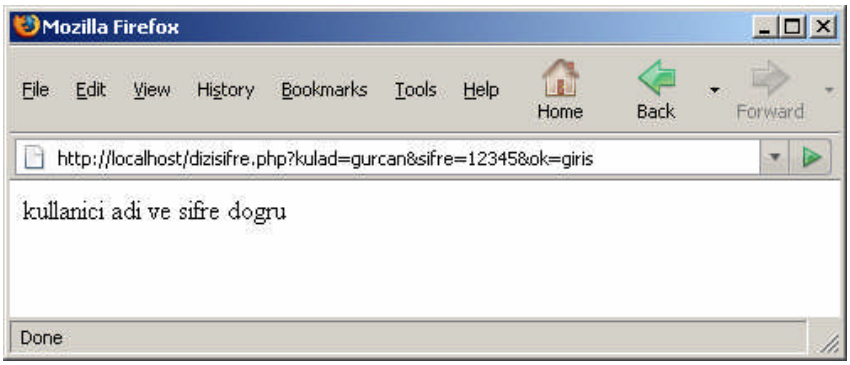
Kod 2.15: “dizisifre.php” program kodları

Programın doğru çalışıp çalışmadığı kontrol edilir.



Şekil 2.15: “dizisifre.php” program çıktısı 1

Kullanıcı adı olarak “gurcan” şifre olarak da “12345” girip, ardından giriş tuşuna basılır.



Şekil 2.16: "dizisifre.php" program çıktısı (Giris düğmesi tıklanınca)

2.5.1. Diziler İle İlgili Fonksiyonlar

Aşağıda diziler ile birlikte kullanabilecek ve işleri kolaylaştıran bazı fonksiyonlar anlatılmıştır.

2.5.1.1. Count( ) Fonksiyonu

Dizi içerisindeki eleman sayısını döndürür.

Kullanımı ;

```
count($dizi);
```

2.5.1.2. Sort( ) Fonksiyonu

Dizi içerisindeki elemanları küçükten büyüğe veya A dan Z ye sıralamak için kullanılır.

Kullanımı ;

```
sort($dizi);
```

2.5.1.3. Rsort( ) Fonksiyonu

Dizi içerisindeki elemanları büyükten küçüğe veya Z den A ya sıralamak için kullanılır.

Kullanımı ;

```
rsort($dizi);
```

2.5.1.4. In_Array( ) Fonksiyonu

Dizi içerisinde bir değerin var olup olmadığını arar. Var ise TRUE yani '1' değeri döner yok ise FALSE yani '0' değeri döner.

Kullanımı ;

```
in_array(aranan_değer,$dizi,[bool kesinlik]);
```

Burada kesinlik ifadesi birebir eşleşmeyi ifade eder. Eğer büyük küçük harf eşleşmesi yapılmasını istenirse, TRUE istenmez ise FALSE yazılmalıdır.

2.5.1.5. Array_key_exists( ) Fonksiyonu

Dizi içerisindeki anahtar değerin var olup olmadığını kontrol eder.

```
array_key_exists(anahtar,$dizi);
```

2.5.1.6. Array_count_values( ) Fonksiyonu

Dizi içerisindeki elemanları sayar. Her bir elemandan kaçar adet olduğunu bulur.

```
array_count_values(anahtar,$dizi);
```

2.5.1.7. Extract( ) Fonksiyonu

Dizi içerisinde verilen her bir elemanı, farklı değişkenlere böldüren fonksiyondur.

Kullanımı ;

```
extract($dizi,yayma_türü, "önek");
```

Burada yayma türü içerisinde kullanılabilecek parametreler ve ne anlama geldikleri aşağıdaki tabloda verilmiştir.

Parametre	İşleyişi
EXTR_OVERWRITE	Çakışma olduğunda var olan değer silinir. Bu durum asli durumdur.
EXTR_SKIP	Çakışma olduğunda var olan değer silinmez.
EXTR_PREFIX_SAME	Eğer çakışma olursa değişken isiminin önüne "önek" eklenir.
EXTR_PREFIX_ALL	"önek" tüm değişkenlerin başına eklenir.
EXTR_PREFIX_INVALID	veri tanımlayıcıda hata olduğu ya da sayısal bir değer olduğunda değişken ismine "önek" eklenir.

Tablo 2.1: Extract fonksiyonu argümanları

2.5.1.8. Compact( ) Fonksiyonu

Verilen değişken ve değerleri ile bir dizi oluşturmak için kullanılır.

Kullanımı ;

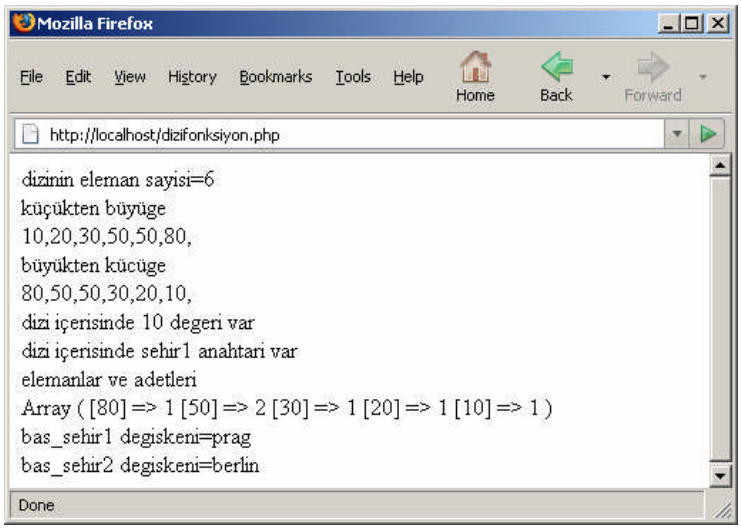
```
$ad1=”ahmet”;
$ad2=”ümit”;
$ad3=”gökhan”;
$ad4=”okan”;
$dizi=compact(“ad1”,”ad2”,”ad3”,”ad4”);
```

Bu durumda oluşan dizi ve elamanları array(“ad1”=>”ahmet”, “$ad2”=>”ümit”, “$ad3”=>”gökhan”, “$ad4”=>”okan” şeklinde olacaktır.

Yukarıda gösterilen fonksiyonlar, örnek bir programda kullanılacak ve daha iyi anlaşılmalarına çalışılacaktır. Programın ismi dizifonksiyon.php olsun.

```
<HTML><BODY><?
$dizi1=array(10,50,80,20,50,30);
$dizi2=array("sehir1"=>"prag", "sehir2"=>"berlin" ,sehir3"=>"tokyo");
$say=count($dizi1);
echo "dizinin eleman sayisi=$say <br>";
sort($dizi1);echo "küçükten büyüge<br>";
        for($i=0;$i<$say;$i++)
                echo "$dizi1[$i],";
rsort($dizi1);echo "<br>büyükten kücüge<br>";
        for($i=0;$i<$say;$i++)
                echo "$dizi1[$i],";
if(in_array(10,$dizi1)==TRUE)
        echo "<br>dizi içerisinde 10 degeri var";
else
        echo "<br>dizi içerisinde 10 degeri yok";

if(array_key_exists("sehir1",$dizi2)==TRUE)
        echo "<br>dizi içerisinde sehir1 anahtari var<br>
        elemanlar ve adetleri<br>";
else
        echo "<br>dizi içerisinde sehir1 anahtari yok ";
$say2=array_count_values($dizi1);
print_r ($say2) ;
extract($dizi2,EXTR_PREFIX_ALL,"bas");
echo "<br>bas_sehir1 degiskeni=$bas_sehir1<br>
        bas_sehir2 degiskeni=$bas_sehir2"; ?> </BODY></HTML>
```

Kod 2.16: “dizifonksiyon.php” program kodları



Şekil 2.17: "dizifonksiyon.php" program çıktısı

2.5.2. Önceden Tanımlı Diziler

PHP'de tanımlama yapmadan kullanılan dizilerdir. Bu dizileri PHP'de belirli şartlarda kullanılabilir. Önceden tanımlı diziler ve açıklamaları tabloda verilmiştir.

$_COOKIE	HTTP çerezleri tarafındnan girilir.
$_GET	Değişkenler GET methodu kullanıldığı zaman oluşur.
$_POST	Değişkenler POST methodu kullanıldığı zaman oluşur.
$_REQUEST	$_COOKIE, $_GET, and $_POST kullanıldığında yüklenen değerler için
$_ENV	Çevresel değişkenler
$_FILES	Her hangi bir dosya gönderildiğinde "up-load" oluşan bilgiler.
$_SERVER	WWW sunucuda tanımlanmış değişkenlerdir.
$_SESSION	Oturum yönetiminde kullanılır.

2.6. Fonksiyonlar

Program içerisinde tekrar edilen işlemleri alt program halinde yazılmasına ve ihtiyaç olduğunda çağrılmasına olanak sağlayan yapılardır.

2.6.1. Php' de Fonksiyon Tanımlama

Fonksiyon tanımlamada kullanılan kelime dizisi aşağıdaki gibidir.

```
function Fonksiyon belirteci (Argüman 1[=varsayılan değer], Argüman
2[=Varsayılan dğer], ...){
                Fonksiyonun ana gövdesi
                        .
                        .
                        .
                Return [dönecek değer];
```

"Return" ifadesi bir ya da birden çok kullanılabilir. Genelde bunu kullanmaya ihtiyaç duyulmaz. Fonksiyondan bir değer döndürülmek istenmediğinde bu komut kullanılmaz.

Yapılan örnekler, fonksiyon yapısının daha iyi anlaşılmasına yardımcı olacaktır.

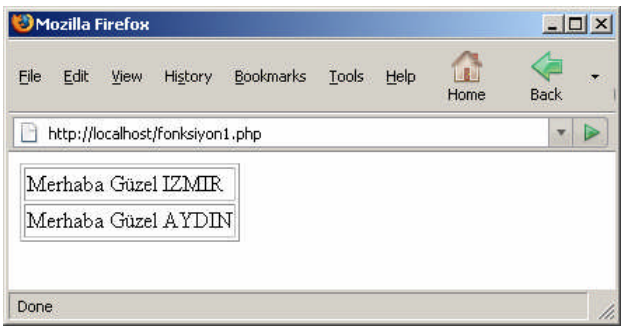
Örnek2.12:

İlk örnekte, bir fonksiyon çağrılacak ve fonksiyon içersinde ekrana "Merhaba Güzel AYDIN" yazdırılacaktır. Fonksiyon dışında ise "Merhaba Güzel IZMIR" yazdırılsın. Dosya ismi olarak "foksiyon1.php" verilir.

```
<HTML><BODY>
<? function yaz()
{
        echo "<td>Merhaba Güzel AYDIN</td></tr></table>";
}

// fonksiyonu cagiralım....
echo "<table border=1><tr><td>Merhaba Güzel IZMIR</td></tr><tr>";
 yaz();
?>
</BODY></HTML>
```

Kod 2.17: "fonksiyon1.php" program kodları

Programın tarayıcıdaki görüntüsü aşağıdaki gibi olacaktır.



Şekil 2.18: "fonksiyon1.php" program çıktısı

Görüldüğü gibi program ilk olarak, alt kısımdaki echo ifadesinden itibaren çalışmıştır. Daha sonra, yaz fonksiyonunu çağırarak diğer echo ifadesini, yani “Merhaba Güzel AYDIN” ifadesi çağrılır.

NOT: <table> <tr> ve <td> etiketleri HTML kodlarıdır. <table> ile yeni bir tablo oluşturulur. Tablo içerisinde satır açmak için <tr></tr> etiketleri kullanılır. Sütun oluşturmak için ise <td></td> etiketleri kullanılır. Tablo bitiminde ise </table> ifadesi kullanılır. Bu örnekte amaç tablonun açılışının fonksiyon dışında başlatılmış olup fonksiyon içerisinde bitirilebildiğini göstermektir.

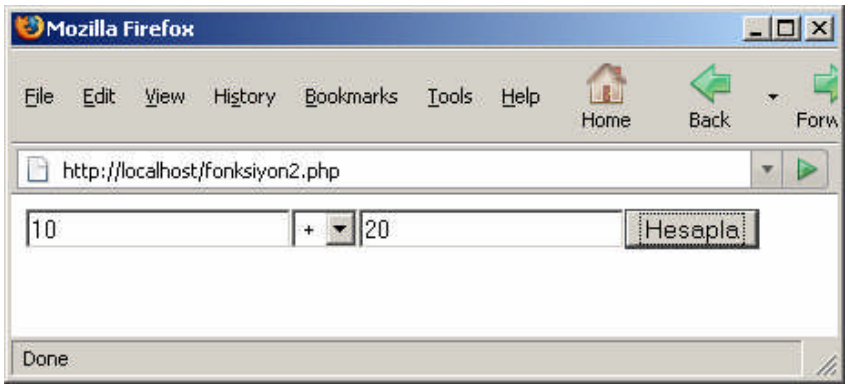
Örnek2.13:

Bu programda ise kullanıcı tarafından seçilen işlem ile birlikte gerekli fonksiyonu çağırarak seçilen işleme göre sonucu bulduran program yapılacaktır.

```
<HTML><BODY>
<? function topla($a,$b)
{
        $toplam=$a+$b;
        echo "toplam=$toplam";
}
function carp($a,$b){
        $carpim=$a*$b;
        echo "Carpim=$carpim";
}
if(isset($_GET[ok])==FALSE){
        echo "<form><input type=text name=sayi1>";
        echo "<select name=islem><option value=+>+</option>";
        echo "<option value=*>*</option></select>";
        echo "<input type=text name=sayi2>";
        echo "<input type=submit name=ok value=Hesapla>";
        echo "</form>";
}else{
        if($_GET[islem]=='+'){
                topla($_GET[sayi1],$_GET[sayi2]);
        }else{
                carp($_GET[sayi1],$_GET[sayi2]);
        }
}?> </BODY></HTML>
```

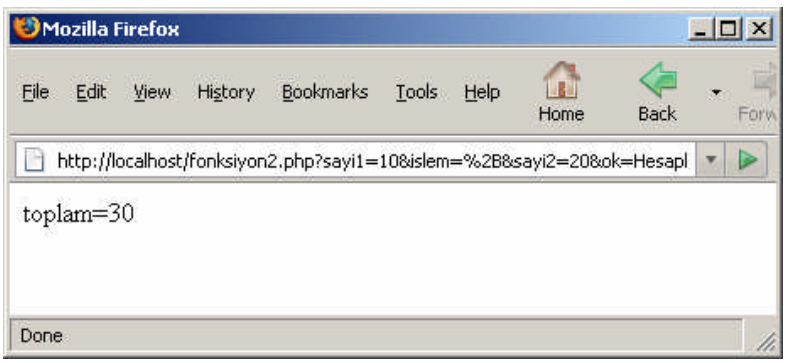
Kod 2.18: “fonksiyon2.php” program kodları

Programın tarayıcıdaki görüntüsü aşağıdaki gibi olacaktır.



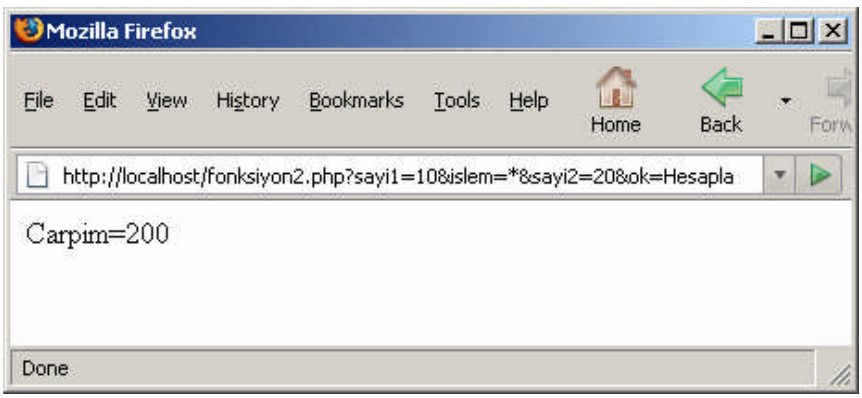
Şekil 2.19: "fonksiyon2.php" program çıktısı 1

10 ve 20 sayılarını yazdıktan sonra toplama işlemini seçerek hesapla düğmesine basalım.



Şekil 2.20: "fonksiyon2.php" program çıktısı (+ seçilip Hesapla tıklanınca)

Yine aynı sayılarla çarpma işlemi seçilip Hesapla düğmesine basılırsa.



Şekil 2.21: "fonksiyon2.php" program çıktısı (* seçilip Hesapla tıklanınca)

NOT: Html'de <select> etiketi kullanılarak yeni bir liste kutusu eklenebilir. Liste kutusundaki her bir seçenek için ise <option></option> etiketleri kullanılmalıdır. En sonunda ise </select> etiketi kapatılmalıdır. Gönderilen argüman, $_GET ile alınır. (Formda POST metodu kullanılmış olsa idi $_POST ile alınır).

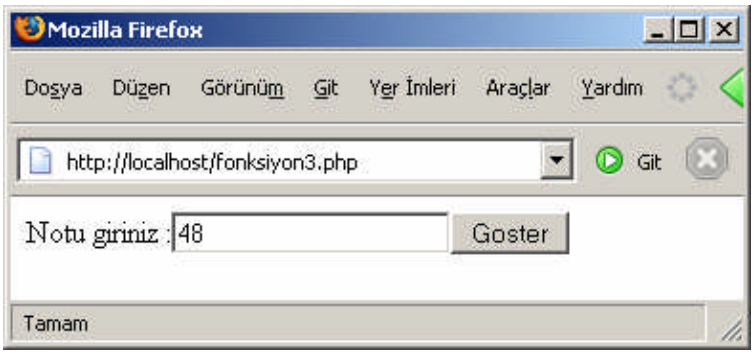
Örnek2.14:

Bu örnekte, fonksiyon içerisinde return ifadesi kullanılacaktır. Girilen notun değerini yazdıran program aşağıda gösterilmiştir.

```
<HTML><BODY>
<? function notlar($not1){
        switch($not1){
                case (($not1>=0) && ($not1<20)):
                        $sonuc="sifir";break;
                case (($not1>=20) && ($not1<45)):
                        $sonuc="bir";break;
                case (($not1>=45) && ($not1<55)):
                        $sonuc="iki";break;
                case (($not1>=55) && ($not1<70)):
                        $sonuc="üc";break;
                case (($not1>=70) && ($not1<=85)):
                        $sonuc="dort";break;
                case (($not1>=85) && ($not1<=100)):
                        $sonuc="bes";break;
                default:
                        $sonuc="not_uyumsuz";break;}
        return $sonuc;}
if(isset($_GET[goster])==FALSE){
        echo "<form >Notu giriniz :<input type=text name=not1>";
        echo "<input type=submit name=goster value=Goster></form>";
}else{
        $islem=notlar($_GET[not1]);
        echo "Notunuz =$islem";}?>
</BODY></HTML>
```

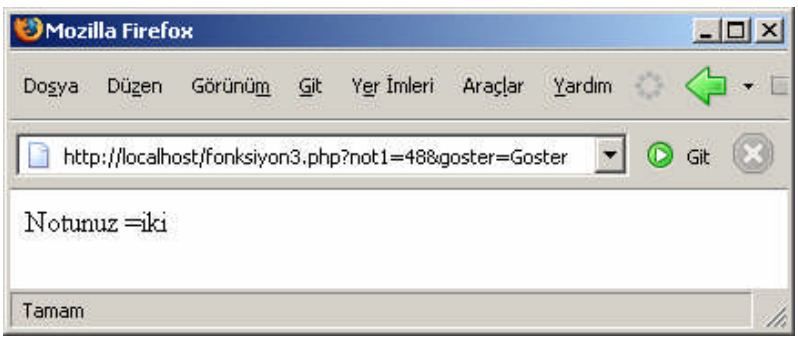
Kod 2.19: "fonksiyon3.php" program kodları

Programın tarayıcıdaki görüntüsü aşağıdaki gibi olacaktır.



Şekil 2.22: "fonksiyon3.php" program çıktısı 1

Not olarak 48 girildi ve göster düğmesine basıldı.



Şekil 2.23: "fonksiyon3.php" program çıktısı (Göster düğmesi tıklanınca)

2.6.2. Fonksiyonlarda Varsayılan Argüman

```
function varsayilan($a =1){
        return $a;
}
```

Yukarıdaki örnekte verilen şekilde varsayılan argüman değeri yazılabilir, böylelikle fonksiyona herhangi bir argüman gönderilmediğinde bu değer kullanılır. Bu varsayılan argüman olarak ifade edilir.

Başka bir deyişle varsayilan( ) şeklinde kullanıldığında, fonksiyondan "1" değeri döndürür.

Ya da varsayilan(100) şeklinde kullanıldığında, fonksiyondan "100" değeri döndürülür.

2.6.3. Fonksiyonlarda Varsayılan Argüman

PHP'de fonksiyon tanımlama sırasında bazen argümanlara karar verilmeyebilir. Bu durumda da fonksiyona argümanlar gönderilebilir. Bu durumda "func_num_args" ve "func_get_arg" gibi değişebilir argümanlı fonksiyon operatörleri kullanılabilir.

Alt kısımda "func_num_args" ve "func_get_arg" ile ilgili örnek verilmiştir.

```
function toplam(){
                $n = func_num_args();
                $sonuc = 0;
                for ($i=0;$i < $n;$i++){
                        $sonuc += func_get_arg($i);
                }
                return $sonuc;
        }
$toplam1=toplam(1,2,3,5);
echo $toplam1;
```

Kod 2.20: Argüman sayısına göre değişen program kodları

Bu ifadede gönderilen argüman sayısı herhangi bir şekilde sınırlandırılmamıştır. Bu nedenle öncelikle gönderilen argümanların sayısı bulunmalıdır. Bunun için de ***"func_num_args( )"*** fonksiyonu kullanılmıştır. Bir döngü içerisinde ise bu gönderilen argümanlar, ***"func_get_args($i)"*** fonksiyonu ile tek tek sıra ile alınmıştır.

2.6.4. Fonksiyonlarda Referans

PHP'de fonksiyonların argümanları fonksiyon içerisinde değiştirilemez. Bunun nedeni bu argümanın sadece değerinin kullanılıyor olmasıdır. Fonksiyon içerisinde argümanın değiştirilmesi ana programdan gönderilen argümanı değiştirmez.

Örnek olarak aşağıdaki fonksiyonu yorumlarsak:

```
function toplam($n){
                $n++;
        }
```

Aşağıdaki bölümde bu fonksiyon çağrılacaktır.

```
$n = 1;
        toplam($n);
        print("$n\n");
```

“toplam” fonksiyonunu çağırdığımızda, sonuç yine $n=1 oluyor. Başka bir deyişle $n değişmemiş oluyor.

Ardından geçiş referansı kullanarak argüman gönderilir. Aşağıdaki örnekte görüldüğü gibi, gönderilen değişkenin önüne “&” işareti eklenir.

```
$n = 1;
        topla(&$n);
        print(“$n\n”);
```

Bu program çalıştırıldığında sonuç $n=2 olacaktır.

Eğer argümana, aşağıdaki fonksiyonda tanımlandığı gibi “&” ön eki eklenirse referanslı geçiş her zaman kullanılabilir.

```
function toplam2(&$n){
                $n++;
        }
```

2.6.4.1 Lokal ve Global Değişkenler

Fonksiyonlarda kullanılan değişkenler genelde lokal değişkendir. Bu da bu değişkene fonksiyonun dışından erişimini imkânsız kılar. Bu nedenle aşağıdaki örnekteki iki adet $a tamamen farklıdır.

```
function deneme ( ){
                $a = 1;
        }

        $a = 100;
```

Fonksiyonların dışında da kullanılabilmesi için bu değişkenlerin global tanımlanması gerekir. Tanımlama sırasında “global” kelimesi kullanılmalıdır.

```
function deneme ( ){
                global $a;
                print($a);
        }

        $a = 100;
        deneme();
```

Bu durumda ana programda kullanılan $a değişkeni yine deneme fonksiyonunda kullanılabilir.

2.6.4.2. Statik Değişkenler

Statik bir değişkenin etkisi sadece fonksiyon içinde geçerlidir. Ancak fonksiyon sona erse de değişken son değerini sürdürür. Statik değişkenler local değişkenlerden bu noktada farklılık gösterir.

Değişken değeri fonksiyonun en son çağırılması sırasında atanan değer olur.

Örnek olarak aşağıdaki fonksiyon gösterilebilir.

```
function deneme(){
        static $a;
        $a++;
        return $a;
    }
```

Bu fonksiyon ilk çağrıldığında 1 değerini döndürür fakat ikinci çağrılışında dönen değer iki olacaktır.

2.7. Sınıflar (Class)

Sınıflar, değişkenler ve belli özellikleri ile benzeşen fonksiyonların bir araya gelmesi ile oluşan bir yapıdır. Sınıf yapısı içerisinde değişkenler "özellik" fonksiyonlar ise "metot olarak tanımlanır. Sınıf kavramı nesne yönelimli programlamanın (Object Oriented Programing) temel taşıdır.

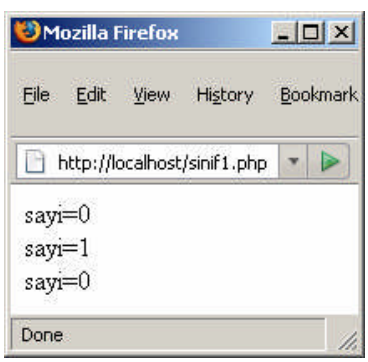
2.7.1. Php'de Sınıf Tanımlama

Basit bir sınıf oluşturarak, PHP'de sınıf "class" kavramı anlatılacaktır.

```
<?class SayiciSinif
 {
    //  ÖZELLİKLER
        var $say;
    //  METHODLAR
          function sifirla(){
                    $this->say=0;
          }
          function yukari_say(){
                    $this->say++;
          }
          function asagi_say(){
                    $this->say--;
          }
          function sayi_al(){
                    return $this->say;
          }
}
          //  Yeni Nesne Tanımlama
          $sayici = new SayiciSinif;
          $sayici->sifirla();
          echo "sayi=".$sayici->sayi_al()."<br>";
          $sayici->yukari_say();
          echo "sayi=".$sayici->sayi_al()."<br>";
          $sayici->asagi_say();
          echo "sayi=".$sayici->sayi_al()."<br>";?>
```

Kod 2.21: "sinif1.php" program kodları

Kodun tarayıcıdaki görüntüsü aşağıdaki gibi olacaktır.



Şekil 2.24: "sinif1.php" program çıktısı

Programda daha önce kullanılmayan ifadeler ve kullanım şekilleri aşağıdaki gibidir.

- **class sayici**

Bu ifade ile yeni bir sınıf oluşturulacağı belirtiliyor.

- **var $say;**

Sınıf içerisinde tüm fonksiyonlarda geçerli olacak bir değişken tanımlanmıştır. Fonksiyon içerisinde tanımlamaktan farklıdır.

- **$this->say;**

"var" ile sınıfın özellikler kısmında tanımlanan değişkenleri fonksiyon içerisinde kullanmak istendiğinde say değişkeninin ön kısmına "$this->" ifadesi eklenmelidir.

- **$sayici=new SayiciSinif;**

SayiciSinif sınıfından "sayici" adında yeni bir nesne oluşturuldu. İstenirse farklı adlarda birden fazla nesne oluşturulabilir.

- **$sayici->sifirla( );**

Yeni bir nesne tanımlandıktan sonra sınıf içerisindeki fonksiyonları çağırmak istendiğinde fonksiyon isminin "sifirla( )" önüne nesne "$sayici->" ifadesi eklenmelidir. Sifirla fonksiyonu sinif içerisindeki $say özelliğini sıfır yapmaya yarar.

Görülmektedir ki sınıf ifadesi "new" ifadesi ile bir nesneye atandı ve ardından bu nesne vasıtasıyla sınıf içerisindeki fonksiyonlar birer birer çağrıldı.

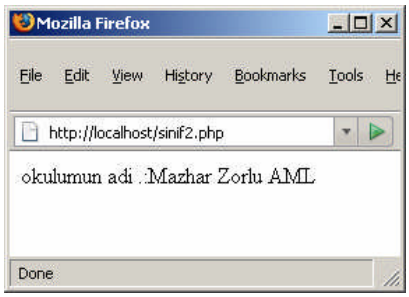
2.7.2. Php'de Yapıcı (Constructor) Fonksiyonlar

Sınıf oluşturulurken otomatik olarak çalıştırılması istenilen sınıf içerisinde tanımlanan fonksiyonlara yapıcı (Constructor) fonksiyonlar denir. Yapıcı fonksiyon ile sınıf adı aynı olmalıdır. Yapıcı fonksiyonlar sayesinde nesne oluşturulduğu anda nesne içerisine parametreleri aktarabiliriz. Aşağıdaki örnekle yapıcıların işlevi anlatılacaktır.

```
<?
class okul
{
        var $ad;
        // Constructor Fonksiyon
        function okul($ad){
                $this->ad=$ad;
        }
}

$okulum=new okul("Mazhar Zorlu AML");
echo "okulumun adı .:".$okulum->ad;
?>
```

Kod 2.22: "sinif2.php" program kodları (Constructor kullanımı)

Programın tarayıcıdaki görüntüsü aşağıdaki gibi olacaktır.



Şekil 2.25: "sinif2.php" program çıktısı

Görüldüğü gibi nesne oluşturulduğu anda sınıf ile aynı isme sahip olan fonksiyon otomatik olarak çalıştırıldı. Gönderilen parametre, sınıfın $ad özelliğine aktarıldı, ardından echo komutu ile bu özellik ekrana yazdırıldı.

PHP'de diğer dillerdeki gibi (java, c++) deconstructorlar yoktur.

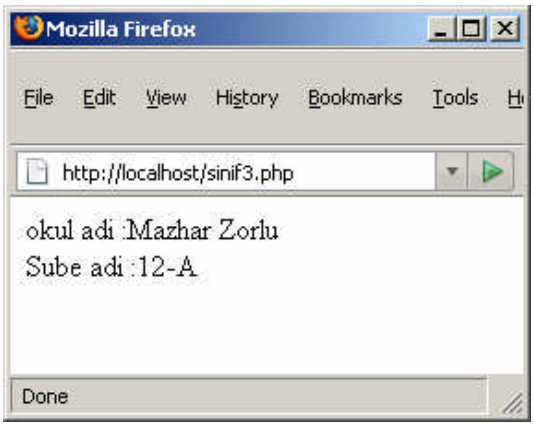
2.7.3. Sınıflarda Kalıtım (Inheritence)

Var olan bir sınıf üzerine yeni özellikler ve metotlar eklemeye, sınıflarda kalıtım (miras) adı verilir. Yeni oluşturulan sınıf, diğer sınıfın tüm özelliklerini ve metotlarını da almış olur. Aşağıdaki örnekte kalıtım olayı anlatılmaya çalışılacaktır.

```
<? class okul
{
        var $okulad;
        var $subead;
        function okul($okuladi){
                $this->okulad=$okuladi;
        }
}
class sube extends okul
{

        function sube($subeadi)
        {
                $this->subead=$subeadi;
        }
}
$bul= new sube("12-A");
$bul->okul("Mazhar Zorlu");
echo "okul adi :".$bul->okulad."<br>Sube adi :".$bul->subead;
?>
```

Kod 2.23: Sinif3.php program kodları (Sınıflarda kalıtım)

Programın tarayıcıdaki görüntüsü aşağıdaki gibi olacaktır.



Şekil 2.26: "sinif3.php" program çktısı

İlk defa kullanılan ifadelere bakılırsa;

- **class sube extends okul**

Burada şube adında yeni bir sınıf oluşturulmaktadır. Şube sınıfı extends ifadesinden sonra yazılan sınıftan "okul" dan miras almaktadır. Bunun anlamı şube class'ı okul class'ındaki tüm özellikleri ve metotları kullanabilir hale geldi demektir.

- **$bul= new sube("12-A");**

Şube class'ından $bul adında yeni bir nesne türetildi. Bu nesne miras alınan class'ın da özelliklerine ve metotlarına ulaşabilecektir. Sube class'ı içerisindeki "constructor" yapıcı fonksiyonuna (sinif adı ile aynı ada sahip) ilk olarak "12-A" verisi gönderiliyor.

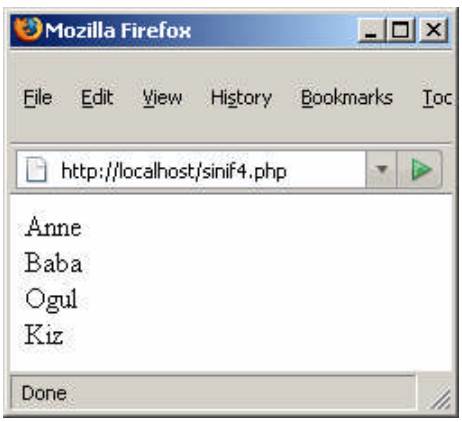
- **$bul->okul("Mazhar Zorlu");**

Burada şube class'ından türetilmiş nesne sayesinde okul class'ındaki, constructor fonksiyon olan "okul" fonksiyonuna da okul ismi "Mazhar Zorlu" verisi gönderiliyor. Görüldüğü gibi, bu nesne ile her iki class'a ulaşılmıştır. Bu durumu bize sınıflarda kalıtım olayı sağlamıştır.

Farklı bir örnekle, sınıflarda kalıtım olayı incelenmiştir.

```
<? class aile
{
        function fertler()
        {
                echo "Anne<br>";
              echo "Baba<br>";
        }
}

class aile2 extends aile
{
        function fertler()
        {
                parent::fertler();
                echo "Ogul<br>";
             echo "Kiz<br>";
        }
}
$goster=new aile2;
$goster->fertler();

?>
```

Kod 2.24: "sinif4.php" program kodları



Şekil 2.27: "sinif4.php" program çıktısı

Kodlara bakıldığında aile2 sınıfının içerisinde **parent::fertler( )** gibi bir ifade görülür. Bunun anlamı, "aile" sınıfının mirasçısı olan aile2 sınıfının içerisinden, miras veren "ebeveyn" sınıfından, fertler fonksiyonu (metodu) çağrılmıştır. Yani aile2 sınıfı içerisindeki fertler fonksiyonuna aile sınıfındaki fertler fonksiyonunu eklemiş olduk.

UYGULAMA FAALİYETİ

Aşağıdaki uygulama faaliyetini işlem basamaklarına uygun olarak yapınız.

- Bir web sayfasında girilen iki sayıdan büyük olandan küçük olanı çıkaran ve büyük sayıyı ekrana yazdıran programı yazınız.

İşlem Basamakları	Öneriler
➢ Yapacağınız programı önce bir kağıtta tasarlayınız. ➢ Program yazımına geçiniz. ➢ Programınızı uygun klasör içerisine kaydediniz. ➢ Programınızı web gezgininde test ediniz.	➢ Program tasarımında istenilen çıktıları da kâğıt üzerinde tasarlayınız. ➢ Program yazımında noktalamalara ve komutların doğru yazımına dikkat ediniz. ➢ Test aşamasında programınızı değişik değerler ile test ediniz.

UYGULAMA FAALİYETİ

Aşağıdaki uygulama faaliyetini işlem basamaklarına uygun olarak yapınız.

- Açılış sayfasında kullanıcı adının ve şifresinin bulunduğu bir ekranın geldiği, kullanıcı adlarının ve şifrelerinin bir dizide bulunduğu girilen kullanıcı adının ve şifrenin doğru olup olmadığını kontrol eden programı fonksiyonlar da kullanarak yapınız.

İşlem Basamakları	Öneriler
➢ Programı tasarlayınız. ➢ Program yazımına geçiniz. ➢ Programınızı uygun klasör içerisine kaydediniz. ➢ Program içerisinde fonksiyonları yazınız. ➢ Programınızı web tarayıcısında test ediniz.	➢ Program tasarımında istenilen çıktıları da kâğıt üzerinde tasarlayınız. ➢ Program yazımında noktalamalara ve komutların doğru yazımına dikkat ediniz. ➢ Dizi tanımlamalarını doğru yerde yapmaya dikkat ediniz. ➢ Test aşamasında programınızı değişik değerler ile test ediniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki sorulara uygun şıkları bularak cevap veriniz.

1. PHP de koşul ifadesi içerisinde birden fazla koşul var ise bunları aşağıdaki işaretlerden hangileri ile bağlayabiliriz?

A) " ##", "%%"
B) "||", "&&"
C) "||", "^^"
D) "&&", "??"

2. Aşağıdakilerden hangisi sonsuz bir döngüdür?

A) a=1;while(a<5) a++;
B) a=5;while(1) a++;
C) a=6;while(a>5) a--;
D)a=1;while(a==4) a++;

3. Aşağıdakilerden hangisi if deki koşul sağlanmadığında çalışması istenilen bölümün başına koyulan ifadedir?

A) else
B) not-if
C) while
D) do

4. Fonksiyon içerisinden değer döndürmek istendiğinde aşağıdaki ifadelerden hangisi kullanılır?

A) turn
B) default
C) return
D) back

5. Aşağıdaki fonksiyonlardan hangisi ile dizi içerisinde bir elemanın var olup olmadığını bulabiliriz?

A) in_arrays( )
B) array_key_exists()
C) in_array( )
D) array( )

6. Hangisi dizi ile ilgili bir fonksiyon değildir?

A) array_key_exists( )
B) sort ( )
C) in_array( )
D) array_keys_founds()

DEĞERLENDİRME

Cevaplarınızı cevap anahtarı ile karşılaştırınız. Doğru cevap sayınızı belirleyerek kendinizi değerlendiriniz. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt yaşadığınız sorularla ilgili konuları faaliyet geri dönerek tekrar inceleyiniz

ÖĞRENME FAALİYETİ–3

AMAÇ

Betik dili ile sunucu taraflı programlarda oturum nesnesini hatasız bir şekilde kullanabileceksiniz.

ARAŞTIRMA

Bu öğrenme faaliyetinden önce aşağıdaki hazırlıkları yapmalısınız.

- PHP'de güvenlik konusunu araştırınız.

3. BETİK DİLİNDE OTURUM YÖNETİMİ

3.1. Php' de Oturum Yönetimi

Bir web sayfasında kişi sayfaya girdiği anda bazı değişkenler oluşturularak sayfalar içerisinde kişiler ile ilgili sınırlandırmalar getirilebilir. Bunu da iki çeşit oturum yönetimi yöntemi ile çözebiliriz. Bunlardan birincisi SESSION (oturum) nesnesi ikincisi ise COOKIE (çerez) nesnesi olarak adlandırılır.

3.1.1. Sunucu Taraflı Oturum Yönetimi (Session)

PHP'de oturum yönetimi sayfayı ziyaret eden bir kişinin bilgilerini veya ziyaretçiye ait değişkenlerin geçici bir süre sunucu bilgisayarda kaydedilmesine, gerektiğinde bu bilgilerin çağırılabilmesine olanak sağlar. Örneğin sayfaya giren birinin login olduktan sonra site içerisindeki diğer sayfalara uğradığında login durumunun devam etmesinin gerektiği durumlarda oturum "session" nesnesi kullanılabilir. Kişi web sayfasını ziyaret ettiğinde yeni bir oturum başlatılabilir. Bu durumda sunucuda otomatik olarak bir dosya oluşturulur. Bu dosya oturum bitirildiği zaman veya belli bir süre geçtikten sonra yok edilmesi sağlanabilir.

Sesion yönetiminde kullanılan komutlar ve açıklamaları aşağıdaki gibidir.

- ***session_start():*** Bu komut yeni bir oturum başlatılmak istendiğinde veya var olan oturumun devam ettirilmesi istendiğinde mutlaka kullanılan bir fonksiyondur.

- ***session_register():*** Bu komut ile açılan oturum içerisinde oturum değişkeni oluşturulabilir. Modüldeki uygulamalarda, **$_SESSION[]** komutu ile yeni bir oturum değişkeni oluşturup bu değişken istenildiğinde çağrılacaktır. Ayrıca istenildiğinde bu değişkenin var olup olmadığını **session_registered()** komutu ile kontrol ettirebilir veya **session_unregister()** komutu ile de bu değişken yok edilebilir.

- ***session_destroy() :*** Açılmış durumda olan bir oturumu bitirmek yani yok etmek amacı ile kullanılır. Bu komut işletildiğinde sunucuda oluşturulan oturum dosyasının silinmesi sağlanacaktır.

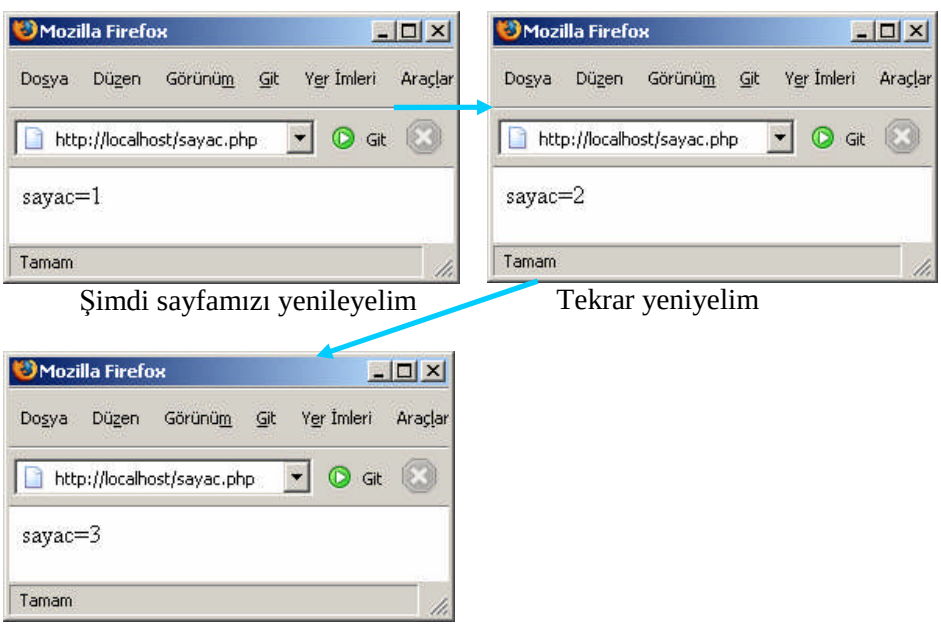
NOT: İstenirse oturum süre ile kısıtlanabilir. Bunun içinde **session_cache_expire()** komutu ile de bu süre kısıtlamalı oturumlar oluşturulabilir.

Aşağıdaki örneklerde oturum "session" yönetimi anlatılmaya çalışılacaktır.

İlk örnekte bir sayaç yapılacaktır. Dosya "sayac.php" olarak kaydedilir.

```
<?
        session_start();
        $_SESSION[sayac]++;
        echo "sayac=".$_SESSION[sayac];
?>
```

Kod 3.1: "sayac.php" program kodları



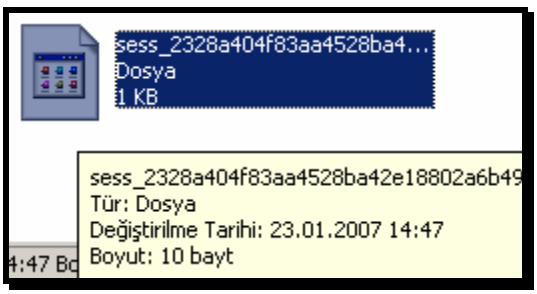
Şekil 3.1: "sayac.php" program çıktısı

Kodda kullanılan fonksiyonların açıklaması:

- **session_start( ):**

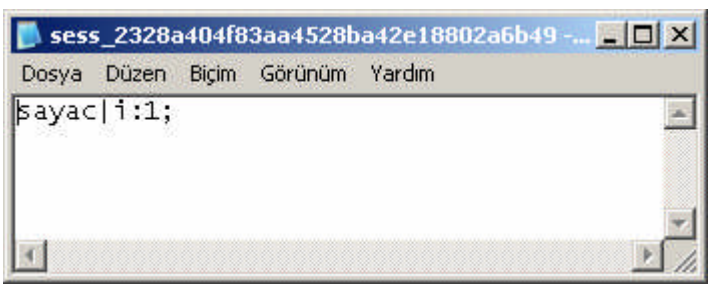
Bu komut görüldüğü anda sunucu bilgisayarda bir dosya oluşturulmaktadır. Dosya ismi 32 karakterden oluşan PHPSESSID ile sess ifadesinin birleşiminden oluşur. Oluşturulan dosya adı sess_PHPSESSID şeklindedir. Örnekte, oluşan dosya ismi aşağıda görüldüğü gibi "sess_2328a404f83aa4528ba42e18802a6b49" dir. Dosya içerisine bakıldığında eğer oturum değişkeni oluşturulmuş ise o değişkenin değeri görülebilir.

Sayac.php dosyası yeniden çağrılır. (http://localhost/syac.php) Bu dosyayı çağırdıktan sonra Linux'ta tmp klasörü içerisindeki aşağıda gösterilen tarzda bir dosya görülecektir.



Şekil 3.2: Oturum nesnesinin oluşturduğu dosya

Uygun bir editör programında dosya açıldığında aşağıdaki gibi bir içerik ile karşılaşılır.



Şekil 3.3: Oturum nesnesinin oluşturduğu dosya içeriği

- **$_SESSION[sayac]**

Sayac adında bir oturum değişkeni oluşturulur. Programda, bu oturum değişkeninin her çağırılışında artırılması sağlanmıştır.

İkinci örnekte de kullanıcı adı ve şifresi istenilen bir sayfadan, eğer kullanıcı adı ve şifresi doğru ise login oldunuz mesajından sonra diğer bir sayfaya link vererek login durumuna göre "hala login durumdasınız" veya "Bu sayfayı görüntüleme yetkiniz yok..."

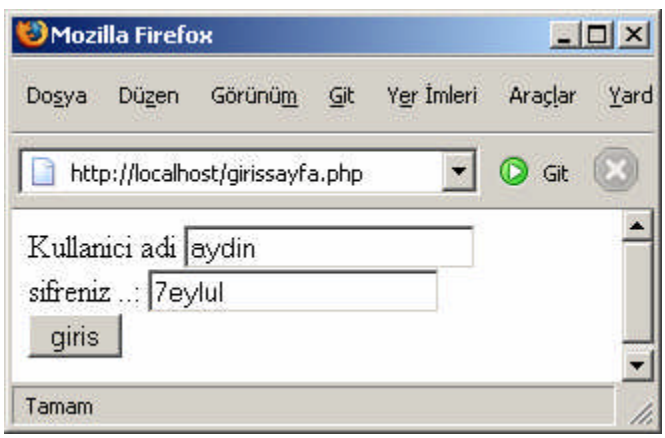
mesajı verilmesi istenmiştir. Ayrıca istenildiği zaman logout olunacaktır. Aynı sayfaya girildiğinde "Bu sayfayı görüntüleme yetkiniz yok..." mesajı görüntülenecektir.

1. sayfaya girissayfa.php ismi verilir.

```
<HTML><BODY>
<form action=login.php>
Kullanici adi <input type=text name=kullanici>
<br> sifreniz ..: <input type=text name=sifre>
<br> <input type=submit name=giris value=giris>
</form>
</BODY>
<HTML>
```

Kod 3.2: "girissayfa.php" program kodları

Bu sayfada kullanıcı adı ve şifresi sorduruluyor. Ekran görüntüsü aşağıdaki gibi olacaktır.



Şekil 3.4: "girissayfa.php" program çıktısı

2. sayfanın ismide "login.php" olacaktır.

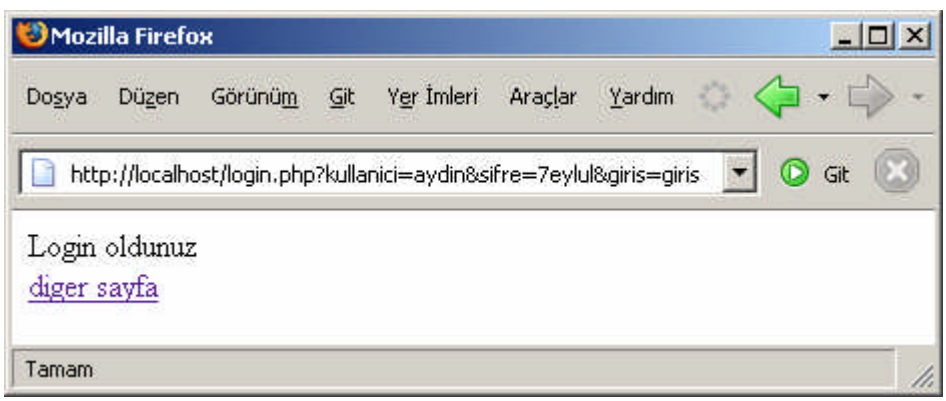
```
<?
if($_GET[kullanici]=="aydin" &&
$_GET[sifre]=="7eylul"){
session_start();
$_SESSION[login]="ok";
echo "Login oldunuz";
echo "<br><a href=diger.php> diger sayfa </a>";
}else{
        echo "sifre ya da kullanici adi hatali";
}
?>
```

Kod 3.3: "login.php" program kodları

Bu sayfada giriş sayfasında sorulan kullanıcı adı ve şifresi karşılaştırılmaktadır. Eğer doğru ise yani kullanıcı adı "aydin", şifresi de Aydın'ın kurtuluş günü olan "7eylül" ise **sessin_start()** komutu ile bir oturum başlatılmaktadır. Daha sonraki adımda ise **$_SESSION[login]="ok"** komutu ile login adında bir oturum değişkeni oluşturulur. Bu değişkenin değeri de "**ok"** olarak atanır. Daha sonra da "diger.php" sayfası için ekrana bir link verilir.

Eğer kullanıcı adı ve şifresi doğru değil ise de ekrana "şifre" ya da "kullanici" adı hatalı şeklinde mesaj yazdırılır.

Kullanıcı adı ve şifre doğru girilmiş ise görüntü şekildeki gibi olacaktır.



Şekil 3.5: "login.php" program çıktısı

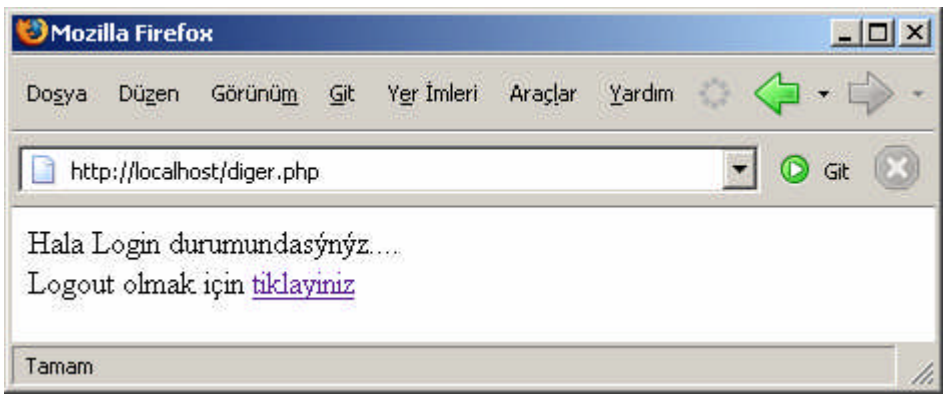
Diğer sayfa linkine tıklandığında görüntülenmesi istenilen sayfanın kodlarını aşağıdaki gibidir.

```
<?
session_start();

if($_SESSION[login]=="ok"){
echo "Hala Login durumundasınız.... ";
echo "<br>Logout olmak için <a
href=logout.php>tiklayiniz</a>";
}else{
        echo "Bu sayfayı görüntüleme yetkiniz yok...";
}
?>
```

Kod 3.4: "diger.php" program kodları

Görüldüğü gibi, bu sayfaya girer girmez oturum tekrar başlatıldı. **session_start();** . if satırında ise $_SESSION[login] değişkeninin "ok" olup olmadığını kontrol edilir. Eğer "ok" durumunda ise sayfaya "login durumundasınız..." yazdırılır. Bir sonraki satırda da, "Logout olmak için tıklayınız..." şeklinde bir link verilir. Bu tıklandığında "logout.php" dosyası çağırılacaktır.

Eğer $_SESSION[login] değişkeni "ok" den farklı ya da oturum bitirilmiş ise ekrana "bu sayfayı görüntüleme yetkiniz yok şeklinde mesaj yazdırılacaktır.



Şekil 3.6: "diger.php" program çıktısı

En son olarak da kullanıcı logout linkini tıkladığında çağırılan sayfanın kodlarını aşağıda görülmektedir.

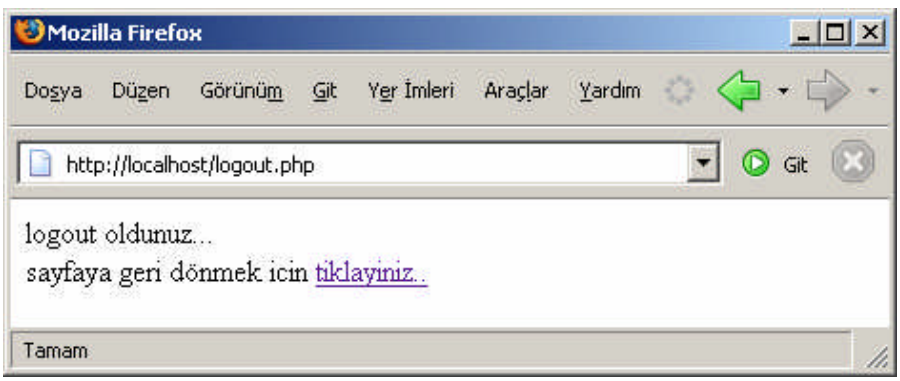
```
<?
session_start();
session_destroy();
echo "logout oldunuz...";
echo "<br>sayfaya geri dönmek icin <a
href=diger.php>tiklayiniz..</a>";

?>
```

Kod 3.5: "logout.php" kodları

Bu sayfada oturum sona erdirilecektir. Ancak bundan önce var olan oturumu komut ile tekrar başlatmak gerekmektedir **session_start()**. Daha sonra da bu oturumun yok edilmesi gerekmektedir. Bu işlem de **session_destroy()** komutu ile yapılabilir. Tabi ki diğer satırda da kullanıcı durumdan haberdar edilir **"logout oldunuz..."** .

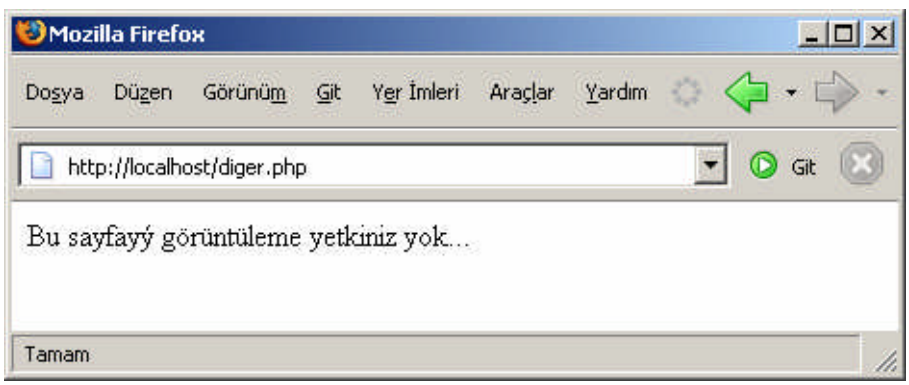
Ayrıca "hala login durumdasınız" yazan sayfaya geri dönüş için de link eklenir.

Bir önceki sayfada, logout düğmesine basıldığında şekildeki sayfa görülecektir.



Şekil 3.7: "logout.php" program çıktısı

Şekildeki, "sayfaya geri dönmek için tıklayınız" linkine tıklanır.



Şekil 3.8: "diger.php" program çıktısı logout yapıldıktan sonra

Az önce login durumunun hala devam ettiğini gösteren sayfada, “bu sayfayı görüntüleme yetkiniz yok... ” yazısı çıkmıştır.

Görüldüğü gibi, kişilerin sayfayı görüntüleme yetkilerinin olup olmadığı, **session** yönetimi ile güvenli bir şekilde kontrol ettirildi. Bu uygulamalar istenildiği gibi arttırılabilir. Birçok defa özellikle forum sayfalarında sayfa görüntüleme yetkisinin olup olmadığı bu yöntemle sorgulanabilir. Eğer kişinin oturum süresince bir değişken ile login durumu kontrol ettirilirse farklı kişilerin sayfalardaki istenmeyen bölümlerinin görüntülemesi engellenebilir.

SESSION nesnesi ile, oturum yönetimi çok geniş bir konudur. Burada sadece iki küçük örnekle bu konu anlatılmaya çalışılmıştır. Daha ayrıntılı ve çok bilgiye internetten ulaşılabilir.

3.1.2. İstemci Taraflı oturum Yönetimi (Cookie “çerez”)

COOKIE yani Türkçe’ye çevrildiğinde, çerezler kullanıcı sayfaya giriş yaptığında, kullanıcı bilgisayarında oluşturulan dosyalardır. Bu dosyalara SESSION nesnesindeki gibi istenilen değişkenler eklenebilir veya istendiğinde silinebilir. COOKIE oluşturmak için ve COOKIE içerisindeki bilgilere ulaşmak için kullanılan komutlar ve açıklamaları aşağıda verilmiştir.

- **Setcookie(“degisken adı”,”değisken değeri”,”cookie ömrü”,”cookie klasörü”,”cookie domaini”, “güvenlik”,”httponly”);**

Bu fonksiyon ile bir çerez oluşturabilir. Setcookie fonksiyonu içerisindeki parametrelerin anlamları şöyledir.

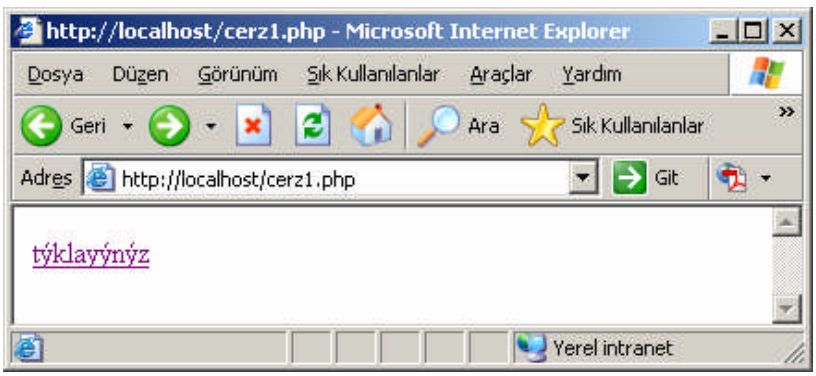
- **Değişken adı** **:** Çerez içerisinde oluşturulacak değişkenin adı yazılır.
- **Değişken değeri** **:** Oluşturulan değişkene atılacak değer.
- **Cookie ömrü** **:** Oluşturulan çerezin saniye değeri olarak ömrü şimdiki zamana eklenir. Kullanımı ***time()+(saniye süre)***
- **Cookie klasörü** **: O**luşturulan çerezin bulunacağı klasör.
- **Cookie domaini** **:** Çerezin etkin olacağı domain adı. Örneğin “deneme.com”.
- **Güvenlik** **:** Çerezin “https” güvenli http protokolünde çalışıp çalışmayacağı.
- **Httponly** **:** Çerezin “http” protokolünde sadece açılabileceği.

Yapılacak örnekte, ömrü bir saat olan bir çerez oluşturulacaktır.

```
<?
setcookie("degisken","prag",time()+3600);
echo "<a href=cerz2.php> tıklayınız </a>";
?>
```

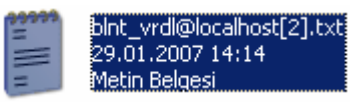
Kod 3.6: “cerz1.php” program kodları

Kodda görüldüğü gibi, değişken ve değer olarak “prag” bilgisi bir saat ömürlü bir cookie içerisine yazıldı. Tarayıcıdaki görüntüsü:



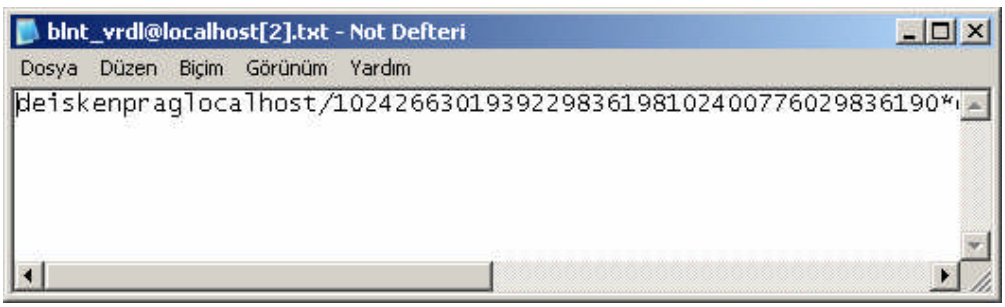
Şekil 3.9: “cerz1.php” program çıktısı

Kod tarayıcıda çalıştırıldığında cookie’lerin bulunduğu klasörde bir uzantısı txt olan bir dosya oluşturuldu.



Şekil 3.10: Cookie dosyası

Dosya içine bakılırsa;



Şekil 3.11: Cookie dosyası içeriği

Bu programda *setcookie("degisken","prag",time()+3600);* ifadesi ile yeni bir cookie oluşturuldu. Değişken içerisine “prag” ifadesi atandı. Diğer bir sayfada da bu cookie içeriği okutulup, ekrana yazdırılacaktır.

```
<?
echo $_COOKIE[degisken];
?>
```

$_COOKIE[degisken] : İfadesi ile cookie içerisinden değişkeni alarak echo komutu ile ekrana yazdırıldı.

Bu şekilde web sitesini ziyaret eden bir kişiye çerez göndererek daha sonraki girişinde, direkt olarak çerezden okuyarak login yapılabilir.

Örnekler bu şekilde çoğaltılabilir. Bu modülde başlangıç seviyesinde bilgiler vermekle yetinilmiştir. Bu konuda da, internetten çok sayıda doküman ve örnek bulunabilir.

UYGULAMA FAALİYETİ

Aşağıdaki uygulama faaliyetini işlem basamaklarına uygun olarak yapınız.

- Bir web sayfasında başlangıç sayfasında kullanıcı adı ve şifresi girilecek, şifre doğru ise diğer sayfadaki spor sayfası ve haberler sayfasına linkler görünecek. Ayrıca yine bu sayfada logout linki olacak. Spor sayfasının ve haber sayfasının içeriği de sadece kullanıcı adı ve şifresi doğru olanlar tarafından okunabilecek. Her sayfada bir logout düğmesi olacak. Logout olunduğunda spor sayfası ve haber sayfasına girildiğinde sayfaya girişinizde "sayfa görüntüleme yetkiniz yoktur" yazacak.

Yukarıdaki koşuları sağlayan programı oturum yönetimi kullanarak yazınız.

İşlem Basamakları	Öneriler
➢ Programı tasarlayınız. ➢ Program kodlarını yazınız. ➢ Kodları kontrol ederek gerekli klasöre kaydediniz. ➢ Programınızı web tarayıcınızda test ediniz.	➢ Programı yazıma geçmeden mutlaka tasarlayınız. ➢ Tasarım aşamasından sonra komut yazımında noktalama ve karşılaştırma ifadelerinin doğruluğuna dikkat ediniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki sorulara uygun şıkları bularak cevap veriniz.

1. PHP de oturum içerisinde bir değişken tanımlama için hangi komut kullanılır?

A) $_SESSION[ ] **C)** $SESSION[ ]
B) SESSION$ **D)** SESSION[ ]

2. Oturum başlatmak için aşağıdaki komutlardan hangisi kullanılmalıdır?

A) session_begin( ) **C)** session_start( )
B) session_register( ) **D)** session( )

3. Oturum (session) başlatıldığında aşağıdaki olaylardan hangisi gerçekleşir?

A) Kullanıcı bilgisayarında geçici bir dosya oluşturulur.
B) Sunucu bilgisayarda bir dosya oluşturulur.
C) Hem kullanıcı hem de sunucu bilgisayarda bir dosya oluşturulur.
D) Ne kullanıcı bilgisayarında ne de sunucu bilgisayarda her hangi bir dosya oluşturulmaz.

4. Çerezler (Cookies) ile ilgili aşağıdakilerden hangisi yanlıştır?

A) Çerez dosyacı kullanıcı bilgisayarında oluşturulur.
B) Çerez dosyası içerisinde değişken ismi ve değeri tutulur.
C) Çerez ile bir kişinin siteyi bir dahaki ziyaretini kontrol edebiliriz.
D) Çerez dosyası sunucu bilgisayarda oluşturulur.

5. Aşağıdaki komutlardan hangisi ile doğru bir şekilde iki gün süre ile bilgisayarda kaydedilecek bir çerez oluşturulabilir?

A) *setcookie("isim","umit",time()+3600);*
B) *setcookie("soyad","cihan",time()+4800);*
C) *setcookie("isim","mustafa",time());*
D) *setcookie("soyad","Nazman",time()+7200);*

DEĞERLENDİRME

Cevaplarınızı cevap anahtarı ile karşılaştırınız. Doğru cevap sayınızı belirleyerek kendinizi değerlendiriniz. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt yaşadığınız sorularla ilgili konuları faaliyet geri dönerek tekrar inceleyiniz.

MODÜL DEĞERLENDİRME

Modülde yaptığınız uygulamaları tekrar yapınız. Yaptığınız bu uygulamaları aşağıdaki tabloya göre değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. Web sunucunuzu doğru olarak kurabildiniz mi ?		
2. Web sunucunun doğru olarak çalışıp çalışmadığını test ettiniz mi ?		
3. Test programınızı yazdınız mı ?		
4. Programınız hatasız bir şekilde çalıştı mı ?		
5. Echo komutunu kullanabildiniz mi ?		
6. For döngüsünü hatasız bir şekilde kullanabildiniz mi ?		
7. While döngüsünün kullanımını hatasız bir şekilde uygulayabildiniz mi ?		
8. Dizi oluşturarak fonksiyonlarını test ettiniz mi ?		
9. Fonksiyon oluşturabildiniz mi ?		
10. Sınıf oluşturabildiniz mi ?		
11. Sınıflarda kalıtım kullanarak hatasız bir şekilde program yapabildiniz mi ?		
12. Oturum nesnesi kullanarak sayfa yaptınız mı ?		
13. Çerez programı yaptınız mı ?		

DEĞERLENDİRME

Hayır cevaplarınız var ise ilgili uygulama faaliyetini tekrar ediniz. Cevaplarınızın tümü evet ise bir sonraki modüle geçebilirsiniz.

CEVAP ANAHTARLARI

ÖĞRENME FAALİYETİ 1'İN CEVAP ANAHTARI

1	**A**
2	**B**
3	**D**
4	**B**
5	**D**
6	**D**
7	**D**

ÖĞRENME FAALİYETİ 2'NİN CEVAP ANAHTARI

1	**B**
2	**B**
3	**A**
4	**C**
5	**C**
6	**D**

ÖĞRENME FAALİYETİ 3'ÜN CEVAP ANAHTARI

1	**A**
2	**C**
3	**D**
4	**D**
5	**D**

KAYNAKÇA

- MASUDA, Yoichi, Bülent VARDAL, **Web Sistem Uygulamaları,** 12. Sınıf Ders Kitabı, Ağustos, 2005.