

# Komut Satırı ve Temel Komutlar

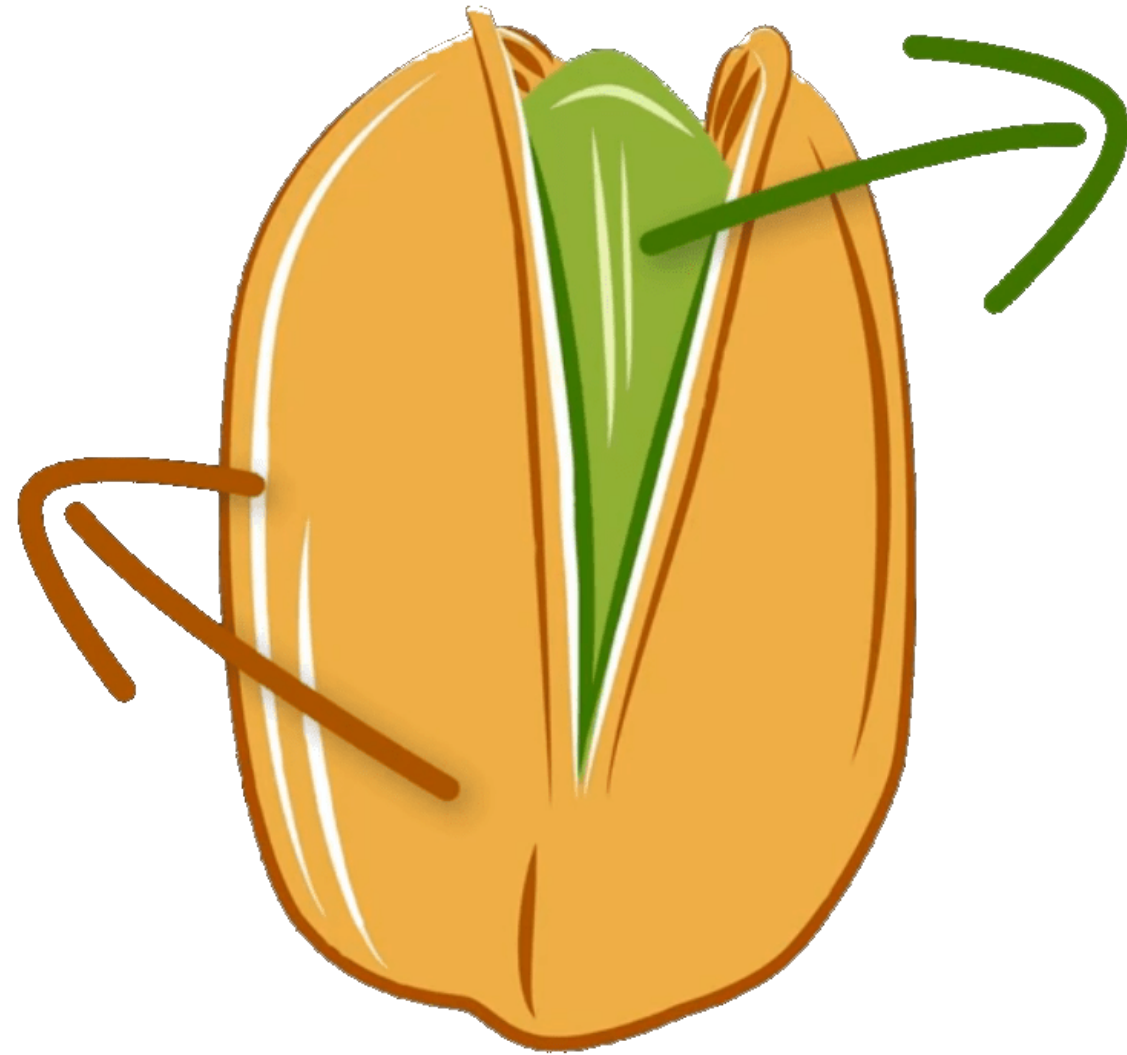
# İçerik

- Kabuk (Shell)
- Temel Komutlar
- Bilgi Alma Komutları

# Kabuk (Shell)

- Kullanıcıların işletim sistemi ile etkileşebileceği iki ana ortam mevcuttur.
- Bunlardan birisi grafik masaüstü, diğeri de komut satırıdır.
- Komut satırını sunan ve komutları yorumlayıp çalıştıran ise **kabuk programıdır**.
- Komut satırına girdiğiniz komutlar yorumlanıp çekirdeğe iletilir.

SHELL  
(Kabuk)



Kernel  
(Çekirdek)

# Kabuk Programı

En yaygın kullanılan kabuk programı **Bash (BourneAgain Shell)** olmakla birlikte sh, ksh, csh, zsh gibi farklı kabuklar da bulunmaktadır.

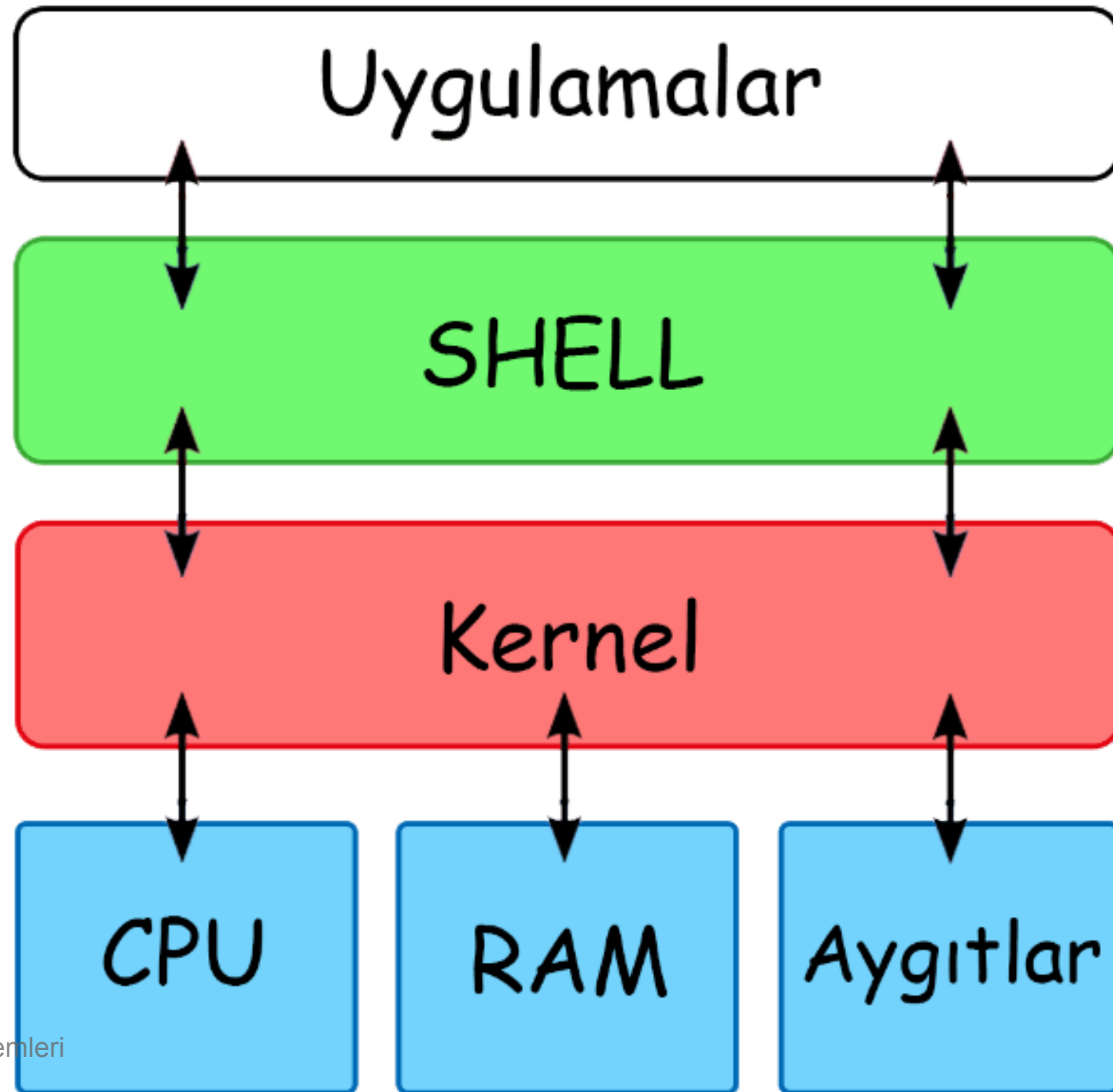


# Terminal (Konsol)

Konsol, kullanıcı ile kabuk arasında yer alarak kullanıcının komut girmesini sağlayan grafiksel ve komut satırı arayüzüne sahip bir araçtır.

Biz komutlarımızı bu araç aracılığı ile kabuğa ulaştırırız kabuk ise kullanıcıdan gelen girdileri yorumlayarak çekirdeğe aktarır.

**Örnek:** `gnome-terminal`



# Komut Satırı

Komut satırı, bir bilgisayar kullanıcısının, belirli metinleri (komutları) girerek, bilgisayarla iletişime geçmesini sağlayan uygulamadır.

Terminali açtığınızda sizi karşılayan ve komut girmenizin beklendiği ekrana `prompt` da denir. `prompt` 'ta kullanıcı için faydalı bilgiler olabilir.



# Komut Satırı Örneği

```
cezmi@server:~$
```

**cezmi:** Kullanıcı adı

**@:** ayraç

**server:** Bilgisayar adı

**:~** : Bulunduğu dizin

**\$:** Yetki işareti

**Not 1:** **\$** normal kullanıcı, **#** root kullanıcı

**Not 2:** Komut satırları özelleştirilebilir. Her zaman bu bilgileri göreceğinizin garantisi yoktur.

# Komut Yapısı

```
cezmi@server:~$ komut_ismi seçenekler parametreler
```

Örnek:

```
cezmi@server:~$ ls -al /etc
```

Seçenekler `-a` `-l` gibi ayrı ayrı da yazılabilir `-al` gibi bitişik de yazılabilir.

# Kabuk Programını Öğren

Terminal ekranına `echo $SHELL` yazıp `Enter` 'a basın.

```
cezmi@server:~$ echo $SHELL  
/bin/bash
```

Konsola girmiş olduğumuz komut yorumlandı ve neticesinde bize `/bin/bash` şeklinde bir çıktı verdi.

Bu çıktı bize mevcut sistemde kullanılan ana kabuk programının `BASH` olduğunu bildirmiş oldu.

Peki `$SHELL` ne, nasıl yorumlandı?

# Ortam Değişkenleri (Çevre Değişkenleri)

- Ortam değişkenleri, programlama dillerindeki değişkenler (variable) gibi belirli bilgileri taşıyan özel dizilerdir.
- Uygulamalar, bu ortam değişkenlerinin değerlerine göre davranışlarını değiştirebilir.
- Ortam değişkenlerine, değişken adının başına `$` eklenerek ulaşılır.
- Ortam/çevre kelimesi ile kast edilen programın çalıştırma ortamıdır.

# BASH Programlama Dili

BASH bir programlama dilidir. Programlama dillerinde de bildiğiniz gibi değişkenler vardır.

# Öntanımlı Ortam Değişkenleri

- **PATH** – Konumlar (path) listesidir. Tam konum (fullpath) belirtilmeden komut girildiğinde işletim sistemi komutu bu listedeki konumlarda arar.
- **HOME** – Kullanıcının dosya sistemi içindeki ev dizinini (home directory) gösterir.
- **TERM** – Kullanılan bilgisayar terminalinin türünü ya da terminal emulatörünü tanımlar.
- **SHELL** - Kullanılan kabuk programını gösterir.

---

## Uygulama:

Bu ortam değişkenlerinin içeriğini komut satırında görüntüleyin.

O andaki tüm ortam değişkenlerini görmek için:

```
env
```

# Çalışma Sorusu

- PATH'e yeni bir dizin nasıl eklenir ve bu değişikliği kullanıcı özelinde nasıl kalıcı hale getirebiliriz?



# Yardım Alma

Komutlar hakkında önceden hazırlanmış yardım dokümanlarına `man` isimli komutla erişebilirsiniz.

**Örnek:** `man` 'in kendi yardım dokümanına erişmek için:

```
man man
```

# Komut Geçmişi

- Komut satırında vermiş olduğunuz komutlar ev dizininde `.bash_history` isimli dosyada tutulmaktadır.
- Daha önce çalıştırdığınız komutlara yukarı ve aşağı yön tuşlarıyla erişebilirsiniz.
- `Ctrl + R` 'ye basıp bir kaç harf yazarak geçmişte uyguladığınız komuta da ulaşabilirsiniz.

Dene: `history`

# Temel Komutlar

# pwd (print working directory)

- İçinde bulunulan dizinin yolunu verir.
- Nerede olduğumuzu öğrenmek istediğimizde kullanırız.

```
$ pwd  
/home/cezmi
```

# cd (change directory)

Komut satırında dizinler arasındaki geçişler `cd` komutuyla yapılır.

`cd` komutu parametre olarak hedef dizinin yolunu alır.

```
$ cd /var/log  
$ pwd  
/var/log
```

# Mutlak Yol

Eğer bir dizinin ya da dosyanın yolu kökten itibaren yani `/` ile başlıyorsa buna **mutlak yol** denir.

## Örnek:

```
/etc  
/home/cezmi  
/usr/local/bin
```

```
$ cd /usr  
$ cd /usr/local/bin  
$ cd /etc
```

## Göreceli Yol (Bağıl Yol)

Göreceli yollar kökten itibaren değil, bulunulan dizinden itibaren işleme konur. Yolun başında `/` bulunmaz.

```
$ cd /usr  
$ cd local/bin  
$ pwd  
/usr/local/bin
```

# Özel Dizin Gösterimleri

Karakter	Temsil Ettiği Dizin
.	Mevcut dizini belirtir
..	Bir üst dizini belirtir.
~	Kullanıcının ev dizinini belirtir.
-	Bir önceki dizini belirtir.



# Özel Dizinleri Deneyelim

```
$ cd /usr/local/bin
$ pwd
/usr/local/bin
$ cd ..
$ pwd
/usr/local
$ cd .
$ pwd
/usr/local/bin
$ cd ~
$ pwd
/home/cezmi
```

- `.` mevcut dizinde bulunan programların çalıştırılmasında kullanılır.

Örnek: `./program`

- Sadece `cd` komutunu çalıştırdığınızda ev dizinine gidersiniz.

```
$ cd  
$ pwd  
/home/cezmi
```

- `~` karakteri yol gösteriminde kullanılabilir.

```
$ ls -al ~/Masaüstü  
$ cd ~/Masaüstü  
$ pwd  
/home/cezmi/Masaüstü
```

## Diğer kullanıcıların ev dizinleri

- Başka bir kullanıcının ev dizinine gitmek için `~kullanıcı` gösterimi kullanılabilir.

```
$ cd ~emrecan/Masaüstü  
$ pwd  
/home/emrecan/Masaüstü
```

# Alıştırmalar

```
1. $ cd /usr/local/bin  
$ cd ../../local  
$ pwd  
???
```

```
2. $ cd /bin  
$ cd ../usr/share/zoneinfo  
$ pwd  
???
```

```
3. $ cd ~/../../usr/local  
$ pwd  
???
```

# ls

Dizin içeriğini (dosya ve alt dizinleri) listeler.

ls komutunun pek çok parametresi vardır. Yardım sayfaları (man ls) kullanılarak detaylı bilgi alınabilir.

- `ls -a` . ile başlayan dosyaları yani gizli dosyaları da gösterir.
- `ls -l` dosya haklarını, sahibini, grubunu, değişiklik zamanı gibi detaylı bilgileri sunar.

**Not:** `ls` komutuna bir dosya verilirse yalnızca o dosyayı listeler, bir dizin verilirse dizinin içerisindekileri listeler.

# Örnek

```
cezmi@server:~$ ls -al
total 24
drwxr-xr-x 3 cezmi cezmi 4096 Mar 30 10:52 .
drwxr-xr-x 4 root  root  4096 Mar 24 08:10 ..
-rw-r--r-- 1 cezmi cezmi  220 Mar 24 08:10 .bash_logout
-rw-r--r-- 1 cezmi cezmi 3771 Mar 24 08:10 .bashrc
drwx----- 2 cezmi cezmi 4096 Mar 30 10:52 .cache
-rw-r--r-- 1 cezmi cezmi  807 Mar 24 08:10 .profile
```

Dosya boyutlarını daha **okunaklı(human readable)** halde görebilmek için **-h** parametresi eklenebilir. Yani: `ls -alh`

# Dosya türleri

`ls -l` komutunun çıktısında sol başta dosya türünü belirten bir karakter yer alır. Bu karakterlerin anlamları şöyledir:

b	Blok dosya.
c	Karakter dosya.
d	Dizin.
l	Sembolik bağlantı.
s	Soket haberleşme dosyası.
p	FIFO, pipe.
-	Normal dosya.

# mkdir

Dizin oluşturmak için kullanılır.

## Örnekler:

- Bulunduğumuz dizinde `foo` isminde bir dizin oluşturmak için:

```
mkdir foo
```

- Bulunduğumuz dizini değiştirmeden `/tmp` dizini altında dizin oluşturmak için:

```
mkdir /tmp/foo
```

- Bulunduğumuz dizinde tek komutla birden fazla dizin açmak için:

```
mkdir bar baz
```



- İç içe dizin oluşturmak için `-p` parametresi kullanılır:

```
mkdir -p dizin1/dizin2/dizin3
```

# touch

Boş bir dosya oluşturmak için veya mevcut dosyanın erişim ve değiştirilme zamanını(**mtime**) güncellemek için kullanılır.

Örnek:

```
touch /tmp/benioku
```

## cp

Dosya ve dizin kopyalar. Bir dosyayı başka bir dosya olarak kopyalayacağı gibi birkaç dosyayı bir dizine de kopyalayabilir.

`cp kaynak hedef` şeklinde çalışır.

# İnteraktif kopyalama

Eğer bir dosyayı A dizininden B dizinine kopyalarsak ve B dizini içerisinde A dizininden taşıdığımız belge ile aynı isimde dosya varsa, taşıdığımız dosya mevcut konumda bulunan aynı isimdeki eski dosyanın üzerine yazılacaktır.

Bu da B konumunda var olan eski dosyanın önceki içeriğinin **yok olması demek.**

**-i parametresi** `cp` komutunu interaktif modda çalıştırır.

Eğer hedef dosya mevcut ise kullanıcıyı uyarır. "y" tuşuna basarak hedef dosyanın üzerine yaz denir.

```
cp -i eski_dosya yeni_dosya
```

## cp örnekleri

- Bulduğumuz dizindeki birden fazla dosyayı farklı bir dizine kopyalamak için:

```
cp dosya1 dosya2 dosya3 /tmp
```

- Dizinleri alt dizinleriyle birlikte kopyalamak için **"-r"** parametresi kullanılır:

```
cp -r eski_dizin /tmp/yeni_dizin
```

## mv

Bir dosyayı veya dizini başka bir dosya veya dizin olarak taşır ya da birkaç dosya veya dizini başka bir dizine taşır.

`mv kaynak hedef` şeklinde çalışır.

`cp` komutundaki gibi hedef dosya varsa üzerine yazar. **"-i"** ile interaktif mod kullanılabilir.

## mv örnekleri

```
mkdir dizin1  
mv dizin1 dizin2  
mv dizin2 /tmp  
  
touch dosya1  
mv dosya1 /tmp/dizin2  
  
# kontrol edelim  
ls -l /tmp/dizin2
```

# rm

Dosya veya dizin siler. Bu komutu kullanırken dikkatli olun.

**Sildiğiniz dosya "çöp kutusuna" gitmiyor, doğrudan siliniyor!**

**Önemli parametreleri:**

- f:** Silerken sormaz, varolmayan bir dosya için bilgi vermez. **Dikkatli kullan!**
- i:** Silmeden önce sorar. (interaktif)
- r:** Dizinleri ve alt dizinleri silmek için gerekli.



# rm örnekleri

```
touch silinecek_dosya.txt  
rm silinecek_dosya.txt
```

```
mkdir silinecek_dizin  
rm -rf silinecek_dizin
```

# echo

Kendisine verilen metni yazdırır.

```
$ echo Selam  
Selam
```

# cat

Dosya ve dosyaların içeriğini okuyup, içeriğini standart çıkışa(stdout) yazar.

```
$ echo Selam > selam.txt
$ cat selam.txt
Selam

# birden fazla dosyayı da okur
$ echo selam > selam1.txt
$ echo selam > selam2.txt
$ cat selam1.txt selam2.txt
selam
selam
```

# find

Dosya ve dizin aramak için kullanılır.

Kullanımı:

```
find dizin seçenekler
```

Dizin adı belirtilmezse bulunulan dizin ifade edilir.

# find parametreleri

```
-name dosya_ismi: Aranacak dosyanın ismi
-type dosya_tipi: Aranacak dosyanın tipi
  f: Normal dosya
  d: Dizin
  l: sembolik bağlantı
-mtime -N: son N gün önce değişmiş dosyalar (geçmiş ile şimdi arasında)
-mtime +N: N günden önce değişmiş dosyalar
-size +-boyut: Aranacak dosya boyutu
-exec: Çıkan arama sonuçları üzerinde komut çalıştırmayı sağlar.
```

# find örnekleri

- `/etc` dizini altında adı `local` ile başlayan **dosyaları** bul.

```
$ find /etc -name local* -type f
/etc/default/locale
/etc/locale.alias
/etc/locale.gen
```

- `/etc` dizini altında adı `local` ile başlayan **dizinleri** bul.

```
$ find /etc -name local* -type d
/etc/polkit-1/localauthority.conf.d
/etc/polkit-1/localauthority
/etc/initramfs-tools/scripts/local-bottom
/etc/initramfs-tools/scripts/local-premount
/etc/initramfs-tools/scripts/local-top
/etc/apparmor.d/local
```

# Çalışma Soruları

- `/bin` dizini altında boyutu 1 MB'tan büyük dosyaları bul.
- Kullanıcının ev dizininde son 1 gün içerisinde değişen dosyaları bul.
- Bulunduğunuz dizinde 30 MB'tan büyük 40 MB'tan küçük dosyaları bul.

# grep

Dosya ya da komut çıktısında bir ifade aramak için yani bir nevi filtreleme işlemi için kullanılır.

Kullanımı: `grep "aranacak_ifade" dosya_adı`

Örnek:

```
grep "hasan" isimler.txt
```

- Linux'ta **büyük küçük harf duyarlılığı** olduğu için yukarıdaki örnekte "Hasan" ya da "HASAN" gibi büyük harf içeren kelimeleri yakalamayacaktır. Bunu kapatmak için `-i` parametresi kullanılır.



# grep örnekleri

```
# tek bir dosyada bul  
grep 'kelime' dosya
```

```
# birden fazla dosyada bul  
grep 'kelime' dosya1 dosya2 dosya3
```

```
# dizindeki ve tüm alt dizinlerdeki dosyalarda bul  
grep 'kelime' -r .
```

```
# komut çıktısında bul  
ps ax | grep 'kelime'  
dpkg --get-architecture | grep 'i386'
```

# Bağlantılar

**Inode(düğüm)**, dosyanın sahibi, oluşturulma tarihi, boyutu, tipi, erişim hakları, en son erişim tarihi ve en son değişikliklerin yapıldığı tarih gibi birçok meta verileri içeren yapıdır. Her dosyanın disk üzerinde benzersiz bir inode değeri vardır.

---

- Dosya ve dizinler arasında bağlantı oluşturmak için `ln` komutu kullanılır.
- `ln parametre kaynak hedef` şeklinde kullanılmaktadır.
- 2 tür bağlantı çeşidi bulunmaktadır.
  - Katı Bağlantılar (hard links)
  - Sembolik Bağlantılar (symbolic links)

# Katı Bağlantılar (hard links)

- Katı bağlantıda orijinal dosya ile bağlantı oluşturulan dosyanın birebir aynısı iki farklı dosya oluşur. Katı (hard) bağlantılarda yapılan değişiklikler orijinal dosyayı etkiler. Orijinal dosya silinse bile katı bağlantı içeriği korumaya devam eder.
- Bu durumu disklerde kullanılan RAID-1 teknolojisine benzetebiliriz. Orijinal dosya silinse bile verilere diğer dosya üzerinden erişilebilir.
- `ln` ile oluşturulur.
- Katı bağlantılar yalnızca dosyalar için yapılabilir ve dosyaların aynı disk üzerinde olması gereklidir. Yani farklı iki disk arasında katı bağlantı yapamazsınız.

# Katı Bağlantı Örneği

```
$ cd /tmp
$ echo merhaba > birinci
$ ln birinci ikinci
$ ls -i birinci ikinci
457767 birinci 457767 ikinci
$ echo selam > ikinci
$ cat birinci
selam
```

# Sembolik Bağlantılar (symbolic links)

- Dosyaların kısayolu görevini görür ve görevi yalnızca ilgili dosyaya yönlendirme yapmaktır.
- MS Windows sistemlerindeki karşılığı kısa yol oluşturmaktır.
- Sembolik bağlantının silinmesi orijinal dosyayı silmez.
- Orijinal dosya silinirse de sembolik link işlevsiz hale gelecektir.
- `ln -s` ile oluşturulur.

# Sembolik Bağlantı Örneği

```
$ ln -s ikinci ucuncu
$ ls -il birinci ikinci ucuncu
457767 -rw-rw-r-- 2 emreca emreca 0 Nis  4 10:23 birinci
457767 -rw-rw-r-- 2 emreca emreca 0 Nis  4 10:23 ikinci
457904 lrwxrwxrwx 1 emreca emreca 6 Nis  4 10:24 ucuncu -> ikinci
$ cat ucuncu
selam
$ rm -f ikinci
$ cat ucuncu
cat: ucuncu: Böyle bir dosya ya da dizin yok
$ ls -il birinci ucuncu
457767 -rw-rw-r-- 1 emreca emreca 6 Nis  4 12:39 birinci
457904 lrwxrwxrwx 1 emreca emreca 6 Nis  4 12:39 ucuncu -> ikinci
```

# Bağlantılar Özet

Link oluşturma genelde bulunan dosya sisteminde disk alanı yetersiz hale geldiğinde dosyaları başka bir dosya sistemine taşıdıktan sonra eski dosya/dizin yolunu kullanan kişi ve uygulamaların hala eski yol üzerinden erişimleri için kullanılır.

Örnek:

```
$ ln -s /mnt/yeni_disk/data /var/lib/data
$ ls /var/lib/data
...
```

Veriler `yeni_disk` 'te olmasına rağmen burada verilere `/var/lib/data` altından erişebilir olduk.

# Temel Paket Yönetimi

**Paket:** Bir yazılımın derlenmiş halini veya kaynak kodlarını, dokümanlarını ve kurulum betiklerini içeren bir arşiv dosyasıdır.

**Paket Yöneticisi:** Paketlerin kurulumu kaldırılması güncellenmesi gibi işlemleri yöneten yazılımdır.

**Paket Deposu:** Kullanılan dağıtımın, kendi geliştiricileri tarafından hazırlanmış paketleri sunduğu bir sunucu.



# Paket Kurulumu ve Güncelleme

**apt** Ubuntu'daki **paket yöneticisi**dir.

Tek bir paketi kurmak ve güncellemek arasında fark yoktur.  
Aşağıdaki gibi kurulum/güncelleme yapılabilir.

```
sudo apt update  
sudo apt install yazılım_adı
```

**Dene:** `cowsay` , `cmatrix`

# Sunucu Güncelleme

Sunucudaki tüm paketleri güncellemek için:

```
sudo apt update  
sudo apt upgrade
```

**Uyarı:** Yukarıdaki işlemi yapmadan önce mutlaka sistemin yedeği alınmalı, sanallaştırma teknolojisi kullanılıyorsa "snapshot"ı alınmalı ve ardından bu işlem yapılmalıdır.

# Bilgi Alma Komutları

# uname

Bu komut genel olarak sistemde kullanılan çekirdek hakkında bilgiler verir.

```
cezmi@server:~$ uname -a  
Linux server 5.4.0-104-generic #118-Ubuntu SMP Wed Mar 2 19:02:41 UTC 2022 x86_64 x86_64 x86_64 GNU/Linux
```

# hostname

Bilgisayarın adını gösterir.

```
cezmi@server:~$ hostname  
server
```

Bu isim `/etc/hostname` dosyasında saklanır. Kalıcı isim değişikliği için bu dosya güncellenir.

```
cezmi@server:~$ cat /etc/hostname  
server
```

# lsb\_release

Kullanılan dağıtım hakkında bilgiler verir.

## Örnek:

```
cezmi@server:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 20.04.4 LTS
Release:        20.04
Codename:       focal
```

# whoami

Komutu çalıştıran kullanıcının kim olduğunu söyler.

```
cezmi@server:~$ whoami  
cezmi
```

## W

Sistemde hangi kullanıcı nereden bağlanmış ne çalıştırıyor bilgisini söyler.

```
cezmi@server:~$ w
 13:37:14 up  2:45,  2 users,  load average: 0.00, 0.00, 0.00
USER      TTY      FROM          LOGIN@      IDLE        JCPU        PCPU  WHAT
cezmi     tty1     -             10:52       54:20       0.06s       0.02s  -bash
cezmi     pts/0    10.0.2.2      10:53       0.00s       0.34s       0.00s  w
```



# uptime

Bilgisayarın ne kadar zamandır açık olduğu bilgisini verir.

```
cezmi@server:~$ uptime  
13:39:37 up 2:47, 2 users, load average: 0.00, 0.00, 0.00
```

# date

İsminden de anlaşılacağı gibi bu komut bize sistemin o anki tarih ve saat bilgisini verir. Farklı formatlarda çıktı da verebilmekte. Yardım için: `man date`

```
cezmi@server:~$ date  
Wed Mar 30 13:40:16 UTC 2022
```

**Alıştırma:** Bugünün tarihini 'GG.AA.YYYY' şeklinde ekrana yazdırın.

# which

Herhangi bir komutun tam yol bilgisini öğrenmek için kullanılır.

```
cezmi@server:~$ which date  
/usr/bin/date
```

**Alıştırma:** **ls** programını ev dizini altına `listele` adıyla kopyalayın ve `listele` programıyla ev dizinini listeleyin.

# **dmidecode**

Bu komutun işlevi sistemin donanım ve BIOS bilgilerini göstermektir.

# df

Sistemdeki disklerin kullanımı hakkında bilgiye ulaşmak için kullanılır.

```
cezmi@server:~$ df -Th
```

Filesystem	Type	Size	Used	Avail	Use%	Mounted on
udev	devtmpfs	445M	0	445M	0%	/dev
tmpfs	tmpfs	98M	1.1M	97M	2%	/run
/dev/mapper/ubuntu--vg-ubuntu--lv	ext4	9.0G	4.4G	4.1G	52%	/
tmpfs	tmpfs	489M	0	489M	0%	/dev/shm
tmpfs	tmpfs	5.0M	0	5.0M	0%	/run/lock
tmpfs	tmpfs	489M	0	489M	0%	/sys/fs/cgroup
/dev/sda2	ext4	877M	108M	708M	14%	/boot
/dev/loop2	squashfs	44M	44M	0	100%	/snap/snapd/14978
/dev/loop3	squashfs	62M	62M	0	100%	/snap/core20/1376
/dev/loop4	squashfs	68M	68M	0	100%	/snap/lxd/22526
/dev/loop5	squashfs	44M	44M	0	100%	/snap/snapd/15177
tmpfs	tmpfs	98M	0	98M	0%	/run/user/1001
/dev/loop6	squashfs	62M	62M	0	100%	/snap/core20/1405
/dev/loop0	squashfs	68M	68M	0	100%	/snap/lxd/22753

# du

Bir dizinin, içerdiği tüm dosyalar ile birlikte diskte kapladığı toplam alanı verir.

**Tavsiye:** `ncdu` isimli program daha güzel sonuçlar veriyor.

```
cezmi@server:~$ du -h /bin/  
106M      /bin/
```

**Alıştırma:** `ncdu` isimli paketi kurup `/var` dizini altında çalıştırın.

# free

Bu komut ile kullanılan bellek miktarını öğrenebilirsiniz.

```
cezmi@server:~$ free -h
```

	total	used	free	shared	buff/cache	available
Mem:	976Mi	160Mi	124Mi	1.0Mi	691Mi	665Mi
Swap:	1.7Gi	1.0Mi	1.7Gi			

# lspci

Bütün PCI aygıtlarını listelemeye yarar. Genellikle anakarta takılı olan donanımların marka/model bilgisine erişmek için kullanılır.

```
emrean@ubuntu:~$ lspci
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371AB/EB/MB PIIX4 IDE (rev 01)
00:02.0 VGA compatible controller: VMware SVGA II Adapter
00:03.0 Ethernet controller: Intel Corporation 82540EM Gigabit Ethernet Controller (rev 02)
00:04.0 System peripheral: InnoTek Systemberatung GmbH VirtualBox Guest Service
00:05.0 Multimedia audio controller: Intel Corporation 82801AA AC'97 Audio Controller (rev 01)
00:06.0 USB controller: Apple Inc. KeyLargo/Intrepid USB
00:07.0 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 08)
00:0b.0 USB controller: Intel Corporation 82801FB/FBM/FR/FW/FRW (ICH6 Family) USB2 EHCI Controller
00:0d.0 SATA controller: Intel Corporation 82801HM/HEM (ICH8M/ICH8M-E) SATA Controller [AHCI mode] (rev 02)
```



# ps

Çalışan süreçler(process) için rapor sunar.

```
ps -ax
```

# htop

Etkileşimli süreç görüntüleyici.

```
htop
```

# Kaynaklar

- <https://linux-dersleri.github.io>
- <https://www.pardus.org.tr/kitaplar/LPI-Sertifikasyon-Kitabi-pardus.org.tr.pdf>