

# Introduzione a GNU/Linux e alla Shell Bash

(Corso di Reti di Calcolatori)

Francesco Versaci

(versacif@dei.unipd.it)

DEI – Università degli Studi di Padova

30 marzo 2010



UNIVERSITÀ  
DEGLI STUDI  
DI PADOVA

DEPARTMENT OF  
INFORMATION  
ENGINEERING  
UNIVERSITY OF PADOVA



- 1 Introduzione a GNU/Linux
- 2 La shell Bash
  - Generalità
  - Gestione Base dei Processi
- 3 Comandi
  - Comandi Principali
  - Comandi Avanzati

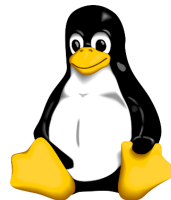


- 1 Introduzione a GNU/Linux
- 2 La shell Bash
  - Generalità
  - Gestione Base dei Processi
- 3 Comandi
  - Comandi Principali
  - Comandi Avanzati



## Caratteristiche

- Sistema operativo UNIX-like (~ POSIX)
- Scritto da zero
- Potente e versatile nel gestire le **reti**
- Software libero (tipicamente sotto **GPL**)
- Modulare



```
cesco@psiche:~$ /sbin/ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 
          inet addr:147.162.96.135  Bcast:147.162.96.255  Mask:255.255.255.0
          inet6 addr: fe80::21d:7dff:feaa:d055/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5294265 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2417471 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2248007577 (2.0 GiB)  TX bytes:1208016863 (1.1 GiB)
          Interrupt:23 Base address:0x6000

cesco@psiche:~$
```

## Generalità

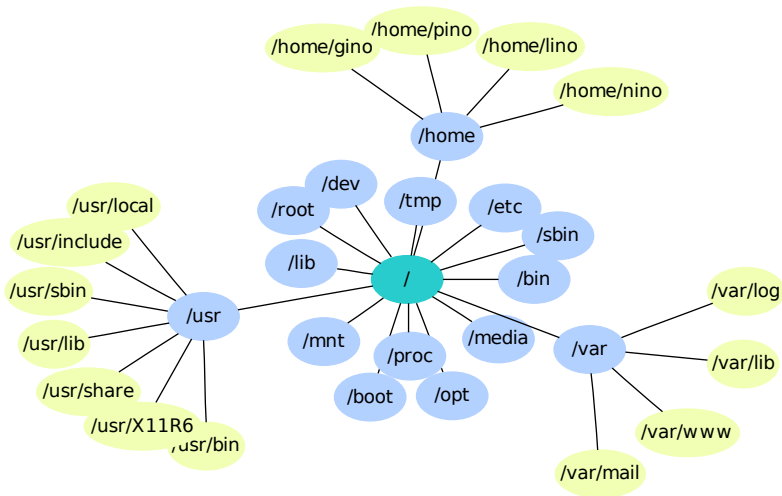
- **Everything is a file** (dischi, terminali, scheda audio, ...)
- Un unico albero su cui si montano i dispositivi

## Caratteri e directory speciali

- / Separatore directory
- .nome A inizio nome indica un file "nascosto"
- ~ La home directory dell'utente
- . La directory corrente
- .. La directory superiore



# Filesystem Hierarchy Standard (FHS) – 1/3



- /** La directory radice
- /bin** Eseguibili importanti per l'utente (es. `cat`, `ls`, `cp`)
- /sbin** Eseguibili per root (amministratore) (es. `mount`, `ifconfig`)
- /dev** Dispositivi (es., `/dev/sda`, `/dev/null`, `/dev/audio`)
- /boot** Kernel
- /etc** File di configurazione
- /root** Home dell'amministratore
- /home** Home degli utenti

- /lib** Librerie essenziali (per eseguibili in `/bin` e `/sbin`)
- /media** Montaggio dispositivi rimovibili (DVD, chiavette, usb)
- /mnt** Montaggio altri filesystem
- /proc** Informazioni su kernel e processi
- /tmp** File temporanei
- /usr** Eseguibili e librerie non essenziali
- /var** Dati vari (DB, web), log, posta



## **/usr**

**/usr/bin** Eseguibili per gli utenti

**/usr/sbin** Eseguibili per  
l'amministratore

**/usr/lib** Librerie

**/usr/share** Plugin, manuali, ...

**/usr/X11R6** L'X Window System

**/usr/local** Programmi installati  
manualmente

## **/var**

**/var/mail** La posta degli utenti

**/var/log** I log di sistema

**/var/lib** Informazioni salvate dai  
pacchetti

**/var/www** Siti web ospitati



## Utenti e Gruppi

Ad ogni file viene associato un **owner** Il proprietario del file (es. gino)

**group** Un gruppo di appartenenza (es. utenti)

## Permessi

È possibile specificare permessi di lettura (r), scrittura (w) ed esecuzione (x) per

- Il proprietario del file
- Per utenti del gruppo di appartenenza
- Per tutti gli altri

## Esempio

```
$ ls -l /dev/sd*
brw-rw---- 1 root disk 8, 0 24 feb 15:29 /dev/sda
brw-rw---- 1 root disk 8, 1 24 feb 15:29 /dev/sda1
brw-rw---- 1 root disk 8, 5 24 feb 15:29 /dev/sda5
brw-rw---- 1 root disk 8, 6 24 feb 15:29 /dev/sda6
```

- Programmi che girano in background (es. sshd, postfix, syslogd, cron)
- Tipicamente server



## Gestione dei Servizi

```
# /etc/init.d/ssh status
sshd is running.
# /etc/init.d/ssh stop
Stopping OpenBSD Secure Shell server: sshd.
# vim /etc/ssh/sshd_config
# /etc/init.d/ssh start
Starting OpenBSD Secure Shell server: sshd.
```

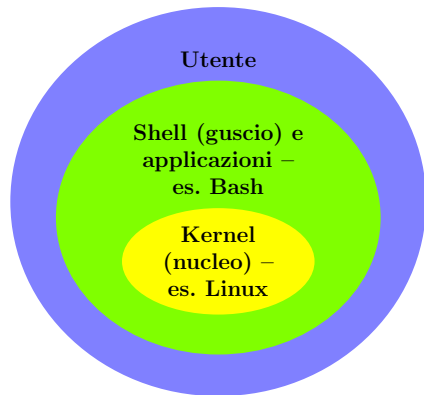
- 1 Introduzione a GNU/Linux
- 2 **La shell Bash**
  - Generalità
  - Gestione Base dei Processi
- 3 Comandi
  - Comandi Principali
  - Comandi Avanzati



# Cos'è una shell?

## A che serve?

- Interagire col kernel
- Gestire file e processi in modo potente e veloce
- Lanciare applicazioni
- Automatizzare operazioni frequenti e/o complesse



Schema a cipolla di un sistema operativo



# BASH

## Caratteristiche principali

- Segue lo standard POSIX
- Disponibile per molti sistemi operativi
- Programmabile, personalizzabile

## Login utente

```
spongebob login: cesco
Password:
cesco@spongebob:~$ pwd
/home/cesco
cesco@spongebob:~$ logout
spongebob login:
```

## Login root

```
spongebob login: root
Password:
spongebob:~# pwd
/root
spongebob:~# logout
spongebob login:
```

## Sintassi generale

```
$ nome-comando <opzioni> <argomenti>
```

## Spiegazione

**opzioni** modificano l'esecuzione del programma

- `--nome-opzione`
- `-o`

**argomenti** obiettivi del comando, di solito nomi di file

## Autocompletamento

Premere TAB semplifica molto la vita...

## Esempi

```
$ uname  
Linux  
$ uname --machine  
i686  
$ uname -m  
i686  
$ wc -l /etc/fstab  
12 /etc/fstab
```

## Sostituzione comando

```
$ dpkg -S `which import`  
imagemagick: /usr/bin/import
```

## Caratteri *jolly*

- \* qualsiasi stringa (nulla inclusa)
- ? carattere qualsiasi
- [...] carattere in intervallo
- [^...] carattere non in intervallo

## Espansione delle graffe

```
$ touch {g,p,tr}ino  
$ ls  
gino  pino  trino
```

## Espansione di percorso

```
$ ls ?ino  
gino  pino  
$ ls *ino  
gino  pino  trino  
$ ls [a-m]ino  
gino  
$ ls t*i*  
trino
```



## Caratteri usati

**backslash** per caratteri singoli

**virgolette** permettono di indicare nomi con spazi, ma non bloccano le sostituzioni di comando

**apici** bloccano anche le sostituzioni comando

## Esempi

```
$ touch "sale e pepe"
$ touch olio\ e\ aceto
olio e aceto sale e pepe
$ echo "$(ls)"
olio e aceto
sale e pepe
$ echo '$(ls) '
$(ls)
```



## Uso degli alias

- Per velocizzare la digitazione di comandi frequenti
- Possono aggiungere opzioni di default ai comandi
- Possono essere aggirati col quoting

## Esempi

```
$ alias lc='ls --color=auto'  
$ alias ll='ls -l'  
$ alias mutt='mutt -y'  
$ mutt #==> mutt -y  
$ 'mutt' #==> mutt  
$ unalias ll
```



## A che servono?

- Per impostare opzioni per la shell
- Per impostare opzioni predefinite per alcuni programmi

## Esempio

```
$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/
games
$ export PATH=~ /mybin:$PATH
$ echo $PATH
/home/cesco/mybin:/usr/local/bin:/usr/bin:/bin:
/usr/bin/X11:/usr/games
```



## ~/ .bashrc

- Eseguito da shell non di login
- Quello globale è  
/etc/bash.bashrc

## Esempio

```
alias ls='ls --color=auto'  
alias ll='ls -ctrl'  
alias rm='rm -i'  
. /etc/bash_completion
```

## ~/ .bash\_profile

- Eseguito da shell di login  
(no terminali)
- Quello globale è  
/etc/profile

## Esempio

```
. ~/.bashrc  
PATH=~/.bin:"${PATH}"  
umask 077
```

## Archivio

Nel file `~/.bash_history` vengono memorizzati gli ultimi `$HISTSIZE` comandi.

## Combinazioni da tastiera

frecche su/giú scorrono l'history

**CTRL-R** consente la ricerca all'indietro

**ALT - .** Ripete argomento del comando precedente

## Esempio

```
$ less /etc/fstab
$ ...
$ ...
$ <CTRL-R>
(reverse-i-search) 'fs': less /etc/fstab
$ ls -l <ALT-.> /etc/fstab
-rw-r--r-- 1 root root 750  8 apr  2009 /etc/fstab
```



- 1 Introduzione a GNU/Linux
- 2 **La shell Bash**
  - Generalità
  - **Gestione Base dei Processi**
- 3 Comandi
  - Comandi Principali
  - Comandi Avanzati



## Diverse esecuzioni

**foreground** esecuzione in primo piano, blocca la shell

**background** esecuzione in sfondo, lascia libera la shell; si invoca aggiungendo il carattere & alla fine del comando

## Esempio

```
$ firefox &  
[1] 7984  
$ ...  
$ <chiudiamo firefox>  
$ ...  
[1]+  Done      firefox
```



## Combinazioni da tastiera

Mentre un processo è in esecuzione in foreground si può alterarne l'esecuzione con

**CTRL-\** si chiede al processo di uscire

**CTRL-C** viene ucciso il processo

**CTRL-Z** il processo viene sospeso e restituita la shell

## Riattivazione

Per riattivare un processo:

**bg** viene mandato in background

**fg** viene mandato in foreground

## Esempio

```
$ mc <CTRL-Z>
[1]+  Stopped          mc
$ vim <CTRL-Z>
[2]+  Stopped          vim
$ fg 1 # si riavvia mc
```



## I 3 canali principali

`stdin` canale di input

`stdout` canale di output

`stderr` canale per messaggi  
d'errore

## Redirezione dei canali

`< nome-file` reindirizza input

`> nome-file` reindirizza output

`2> nome-file` reindirizza `stderr`

`>> nome-file` aggiunge al file se  
già esistente

## Esempio

```
$ echo "6*7" > f.txt
```

```
$ echo "10+3" >> f.txt
```

```
$ cat f.txt
```

```
6*7
```

```
10+3
```

```
$ bc < f.txt
```

```
42
```

```
13
```

```
$ cat /etc/shadow > p 2>&1
```

```
$ # = cat /etc/shadow &> p
```

```
$ cat p
```

```
cat: /etc/shadow: Permissi  
on denied
```

## Pipeline

Si può anche collegare lo *stdout* di un processo con lo *stdin* di un altro, utilizzando il carattere `|` e formando una pipeline.

## Esempio

```
$ ps -e | grep bash
7191 pts/3 00:00:00 bash
8582 pts/5 00:00:00 bash
8732 pts/5 00:00:00 bash
```

## Sequenze

Per combinare comandi:

`com1 ; com2` `com2` è eseguito dopo `com1`

`com1 && com2` idem, ma solo se `com1` non ha dato errori

## Esempi

```
$ sleep 20m; sudo halt

$ ./configure && make
```



- 1 Introduzione a GNU/Linux
- 2 La shell Bash
  - Generalità
  - Gestione Base dei Processi
- 3 **Comandi**
  - **Comandi Principali**
  - Comandi Avanzati



## echo

- Stampa una stringa nello *stdout*
- Utile negli script e nelle sequenze di comandi

## Esempio

```
$ make && echo "Fatto..."
```

## touch

- Crea nuovi file vuoti
- Cambia data di accesso o modifica di file esistenti

## Esempi

```
$ touch main.cc ; make

# mount -o remount,ro /
# touch me
touch: cannot touch `me':
Read-only file system
# fsck /
```



## cat

- Visualizza uno o piú file sullo *stdout*
- Utile per unire piú file (vedi *split*)
- Visualizza su schermo file piccoli
- Può essere usato anche per scrivere piccoli file

## Esempi

```
$ cat /etc/fstab  
$ cat >> TODO.txt
```

## less

- Visualizzatore di file
- Usato dal `man`

/ Ricerca nel file

```
less -i Abilita ricerca  
case-insensitive
```

## Esempio

```
$ dmesg | less
```

## head

Visualizza le prime righe di un file

`head -n` Mostra le prime `n` righe

## Esempio

```
$ find . | sort | head -3
.
./..abook
./..abook/addressbook
```

## tail

- Visualizza le ultime righe di un file
- Utile per seguire file di log

`tail -n` Mostra le ultime `n` righe

`tail -f` Aspetta nuovi caratteri dal file

## Esempio

```
# tail -f /var/log/messages
$ tail -f wget-log
```

## help

Aiuto sui comandi interni `bash`

## man

Aiuto per programmi e comandi

`man -L en` Richiede il manuale in lingua inglese

## Esempi

```
$ help for
$ man -L en vim
$ gunzip -c bash.1.gz | gro
ff -man > ~/bash-man.ps
```

## info

- Manuali in ipertesto
- Spesso piú completi
- Non molto usato

<INVIO> Segue il link

| (L minusc.) Torna indietro

/ Ricerca

## Esempi

```
$ info wget
$ info find
```

`pwd`

Stampa la directory corrente

`pwd -P` stampa directory fisica, ignorando i symlinks

## Esempio

```
$ pwd
/home/cesco/mp3
$ pwd -P
/store/audio/mp3
```

`cd`

Cambia directory

`cd` torna alla home

`cd -` torna alla directory precedente

## Esempio

```
$ pwd
/home/cesco/mp3
$ cd /etc
$ pwd
/etc
$ cd -
/home/cesco/mp3
```



## Sintassi...

Visualizza contenuto directory e informazioni file

`ls -l` visualizza anche informazioni

`ls -R` mostra ricorsivamente sottodirectory

`ls -a` mostra anche file nascosti (`.nomefile`)

`ls -t` ordina secondo orario

`ls -r` inverti ordine

## ...sintassi

`ls -d` mostra directory senza entrarci

`ls -h` human-readable, dimensioni leggibili (es. 9,5M)

`ls --color` usa colori

## Esempi

```
$ touch {p,g,l}ino
$ mkdir dir{1,2,3}
$ ls -d */
dir1/  dir2/  dir3/
$ ls -ctrl ~/downloads
```

## cp

Serve per copiare file o directory

cp -R copia directory  
ricorsivamente

cp -p preserva attributi

cp -v elenca file durante la copia

cp -u copia solo file piú recenti

## mv

Sposta e/o rinomina file

mv -v elenca file durante lo  
spostamento

mv -i chiede conferma prima  
di una sovrascrittura

## Esempi

```
$ cp -v /media/usbdisk/*.jpg ~/foto
```

```
$ mv vecchio-nome.jpg nuovo-nome.jpg
```

`rm`

Cancella file e directory

`rm -R` cancella  
ricorsivamente le  
sottodirectory

`rm -i` chiede conferma a  
ogni file

`rm -f` forza la rimozione

## Esempio

```
$ rm -R .[^.]*
```

`ln`

Crea link fra file, alias per  
raggiungerli da diversi percorsi

`ln -s` crea link simbolico (il piú  
usato)

`ln -f` cancella un eventuale link  
già esistente

## Esempio

```
$ ln -s /store/mp3 ~/mp3
```



## chown

Cambia proprietario e gruppo di un file

`chown -R` Ricorsivo (si applica ai contenuti delle directory)

## chmod

Cambia i permessi

**Ottale** Permessi espressi in base 8

**Mnemonico** Per cambiare alcuni permessi

## Esempio

```
# touch esempio; ls -l esempio
-rw-r--r-- 1 root root 0 29 mar 15:47 esempio
# chown cesco.cesco esempio; chmod 640 esempio
# ls -l esempio
-rw-r----- 1 cesco cesco 0 29 mar 15:47 esempio
# chmod o+r esempio; ls -l esempio
-rw-r--r-- 1 cesco cesco 0 29 mar 15:52 esempio
```



df

Mostra spazio libero dei dispositivi

df -h human-readable

df -T mostra tipo dei filesystem

du

Mostra spazio usato da file

du -h human-readable

du -s non mostra file  
ricorsivamente

## Esempi

```
$ df -Th
```

Filesystem	Type	Dimens.	Usati	Disp.	Uso%	Montato su
/dev/hda5	ext3	5,5G	2,1G	3,2G	40%	/
/dev/hda6	ext3	3,7G	328M	3,4G	9%	/home
/dev/hda8	ext3	27G	2,1G	24G	9%	/store
tmpfs	tmpfs	118M	0	118M	0%	/dev/shm

```
$ du -sh ~
```

```
264M    /home/cesco
```



`ps`

Visualizza i processi attivi

```
ps -u nome mostra processi di
    nome
```

```
ps -e mostra tutti i processi
```

```
ps ax mostra tutti i processi
    con linea di comando
```

## Esempio

```
$ ps -u cesco
```

`top`

Programma grafico per gestire i processi attivi

? aiuto

s scegli tempi di aggiornamento

F ordina

u filtra per utente

k manda segnale

r cambia priorità

## Esempio

```
$ top
```

`su`

Apri una shell come altro utente (*root* di default)

`su -` Apri una shell di login

## Esempi

```
$ su -
Password:
# su - postgres
$ psql
```

`nice`

Avvia un comando con una certa priorità

`priorità` numero da -20 (max. priorità) a 20 (minimo)

`permessi` solo *root* può avviare a priorità negative

`nice -n` Avvia con priorità *n*

`renice n PID` Cambia priorità

## Esempio

```
$ nice -15 make
# nice --10 xcdroast
```

```
kill
```

Manda un segnale a un processo

```
kill -s PID Invia segnale s
```

```
killall
```

Manda un segnale a tutti i processi con lo stesso nome

```
killall -s nome Invia  
segnale s
```

## Segnali

15 Termina (default)

3 Quit

9 Uccidi

19 Ferma

18 Riattiva

## Esempio

```
$ ps -e | grep firefox  
4888 ? 00:06:29 firefox  
$ kill -15 4888  
$ killall -15 firefox
```



- 1 Introduzione a GNU/Linux
- 2 La shell Bash
  - Generalità
  - Gestione Base dei Processi
- 3 **Comandi**
  - Comandi Principali
  - **Comandi Avanzati**



## Sintassi

Crea e gestisce archivi

`tar -c` Crea un archivio

`tar -x` Estrai un archivio

`-f file` Leggi/scrivi su file  
anziché su stdin/out

## Sintassi

`-z` Archivio compresso con gzip

`-v` Elenca file (de)compressi

## Esempi

```
$ mkdir roba; touch roba/{p,g}ino
$ tar -czf roba.tar.gz roba
$ rm -R roba; tar -xvzf roba.tar.gz
roba/
roba/pino
roba/gino
```



## Sintassi

- t Elenca i contenuti
- r Aggiungi a un archivio
- u Aggiorna un archivio

## Sintassi

- j Archivio compresso con bzip2
- p Preserva permessi
- h Segui i link simbolici

## Esempi

```
$ touch roba/{c,m}iao
$ gunzip roba.tar.gz
$ tar -rf roba.tar roba/?iao
$ gzip roba.tar

$ tar -xjf pacchetto.tar.bz2
```



## Sintassi

Divide un file in pezzi piú piccoli

```
split file pref Divide  
file in prefaa, prefab,  
...
```

`-b N` Fa pezzi di N byte

## Unione

Per riunire i pezzi si usa semplicemente il `cat`:

```
$ cat pref* > file
```

## Esempi

```
$ du -b filone  
10485760          filone  
$ split -b 3m filone p-  
$ du -b p-*  
3145728 p-aa  
3145728 p-ab  
3145728 p-ac  
1048576 p-ad  
$ cat p-* > unito  
$ du -b unito  
10485760          unito  
$ cmp filone unito
```

## Sintassi

Cerca dei file e ci fa qualcosa (default: *stampa*)

`find dir <opzioni>` Cerca file in `dir`

`-name 'espr'` Cerca file per nome

`-iname 'espr'` Cerca file per nome ignorando  
(maius/minus)cole

`-type f/d` Cerca solo file/directory

## Esempi

```
$ find /store -iname '*venez*.jpg'  
/store/foto/varie/venezia-01.jpg
```

```
$ find ~ -type d -name 'mutt*'  
/home/cesco/dump/mutt-1.5.9
```



## Sintassi

- `-maxdepth n` Scendi ricorsivamente fino al livello `n`
- `-mindepth n` Scendi ricorsivamente dal livello `n`
- `-size ±dim` Dimensione (maggi/min)ore di `dim`
- `-ctime ±num` File modificati da piú/meno di `num` giorni
- `-perm` Gestisci permessi dei file

## Esempi

```
$ find . -maxdepth 1 -size -3k
./file-piccolo.est
$ find . -ctime +30
./oldfile.est
./dir/vecchio.est
```

## Sintassi

- `-print0` Stampa separati da carattere *null* (0x00)
- `-delete` Cancella i file
- `-exec` Esegui qualcosa per ogni file

## `xargs`

Prende nomi di file in stdin e costruisce una riga di comando

`xargs -0` Filename separati da *null*

`xargs -r` Se stdin vuoto non eseguire comando

## Esempi

```
$ find ~ -name '*~' -delete
$ find . -name '*.eps' -exec epstopdf {} \;
$ find . -type f -perm 755 -print0 | xargs -0r
chmod 644
$ find . -type f -perm 755 -exec chmod 644 {} +
```



## Sintassi

( `espr` ) Raggruppa espressioni  
! `espr` Inverti espressione (NOT)  
`espr1` `espr2` AND delle espressioni  
`espr1 -o espr2` OR delle espressioni

## Esempi

```
$ find . ! -iname '*.jpg' -ctime -7  
$ find . \( -name '*.h' -o -name '*.cc' \) -exec  
c wc -l {} + | sort -n  
53 ./prog/sim-nonmax/punto.h  
69 ./prog/sim-nonmax/test2.cc  
92 ./prog/sim-sing/test.cc  
214 totale
```





## Sintassi

Cerca del testo dentro i file

```
grep testo file Cerca testo in file
```

- i Ignora maiuscole e minuscole
- l Stampa solo i nomi dei file
- Z Nomi file separati da *null*

## Esempi

```
$ find . -name '*.txt' -exec grep -i mucca {} +  
./farm.txt:Mucca, toro, vitello  
$ grep -lZ brutto * | xargs -0r rm
```



## Espressioni regolari

- (esp) Racchiude espressione
- `^/$` Inizio/fine riga
- `.` Carattere qualunque
- `e*` da 0 a infinite e
- `e{n,m}` e ripetuta da m a n volte
- `e?` e opzionale

## Esempi

```
$ D=/usr/share/dict/italian
$ grep -E '^Lin' $D
Linus Linux
$ grep -E 'ucca$' $D
Lucca mucca parrucca pilucca
stucca trucca zucca
$ grep -E '^(vio)?\.ino$' $D
Gino fino lino pino sino tin
o vino violino
$ grep -E '^a.*b.*c.*d.*' $D
abboccando abbracciando abbr
acciandola abbracciandolo ab
dicando
```

## Sintassi

### Copia di tutto

`if=file` File di input (stdin di default)

`of=file` File di output (stdout di default)

`bs=dim` Dimensione blocchi

`count=c` Numero blocchi

`skip=n` Salta `n` blocchi dall'input

`seek=n` Salta `n` blocchi dall'output

## Esempi

```
$ dd if=/dev/sda of=usb.img bs=8M
```

```
$ dd if=/dev/zero of=zero-file bs=1M count=10
```





## Machtelt Garrels

### *Introduzione a Linux*

<http://www.codex.altervista.org/introlinux.html>



## Brian Fox e Chet Ramey

### *Manuale Bash*

`man bash`



## Autori Vari

### *Tutte le pagine man*

```
find /usr/share/man -name '*. [1-9].gz' -exec  
man -l {} \;
```