Introduzione all'interfaccia utente di Linux (comandi di linea)

E.Mumolo, DIA

Generalità

- `Logout: ^d, exit, shutdown (solo superuser)`→ Sconnessione

- Segnali: piccoli interi scambiati tra i processi con system call o tastiera.
  - Segnali da tastiera:
    `erase (^h - cancella un carattere),`
    `kill (^u – cancella la riga), stop (^s), start(^q),`
    `intr (^c), quit (^\ - core dump), eof (^d)`

- Standard input, standard output, standard error ← MOLTO IMPORTANTE
- Processi in foreground, → aspettano la terminazione dei comandi, non scrivono subito il prompt
- Processi in background → NON aspettano la terminazione dei comandi, scrivono subito il prompt
  → operatore &

- Pipe (|): creazione di un canale di comunicazione tra standard output e standard input
  - program1 | program2

- Redirezione (>, >>, <, <<): '>' → lo standard output scrive su file, '<' → lo standard input legge da file
  - Ridirezione su/da file: > filename / < filename
  - Concatenazione su/da file: >> filename / << filename

Uno sguardo alla struttura logica del file system

/	→	radice
/bin	→	file eseguibili del sistema – ls, pwd, cp, mv ....
/boot	→	file necessari per l'avvio del sistema, boot loader ...
/dev	→	file speciali che descrivono i dispositivi – dischi, scheda audio, porte seriali ...
/etc	→	file eseguibili, script, inizializzazione, configurazione sistema, file password, ...
/home	→	directory delle home directory degli utenti
/lib	→	librerie di sistema
/lost+found	→	contiene i file danneggiati
/mnt	→	punto di montaggio file system (mount point)
/proc	→	file system virtuale che contiene informazioni sui programmi in esecuzione
/sys	→	programmi di sistema
/tmp	→	direttorio temporaneo
/usr	→	file relativi alle applicazioni installate
/usr/include	→	header file libreria standard C
/usr/bin	→	file binari disponibili agli utenti
/var	→	file di sistema che variano con frequenza elevata
/var/spool	→	aree temporanee di spooling

Per chiedere informazioni - (0)

- Formato dei comandi:
  - comandi minuscolo separati da caratteri di controllo, generalmente: |, >, & ...
  - variabili d'ambiente sono in maiuscolo
- Primi comandi Unix: man <opzioni> <comando> interfaccia al manuale in linea
- Esempio: `$ man man`

```
MAN(1)                          Manual pager utils                          MAN(1)

NAME
       man - an interface to the on-line reference manuals

SYNOPSIS
       man  [-C  file]  [-d]  [-D]  [--warnings[=warnings]]  [-R encoding] [-L
       locale] [-m system[,...]] [-M path] [-S list]  [-e  extension]  [-i|-I]
       [--regex|--wildcard]   [--names-only]  [-a]  [-u]  [--no-subpages]  [-P
       pager] [-r prompt] [-7] [-E encoding] [--no-hyphenation] [--no-justifi-
       cation]  [-p  string]  [-t]  [-T[device]]  [-H[browser]] [-X[dpi]] [-Z]
       [[section] page ...] ...
       man -k [apropos options] regexp ...
       man -K [-w|-W] [-S list] [-i|-I] [--regex] [section] term ...
       man -f [whatis options] page ...
       man -l [-C file] [-d] [-D] [--warnings[=warnings]]  [-R  encoding]  [-L
       locale]  [-P  pager]  [-r  prompt]  [-7] [-E encoding] [-p string] [-t]
       [-T[device]] [-H[browser]] [-X[dpi]] [-Z] file ...
       man -w|-W [-C file] [-d] [-D] page ...
       man -c [-C file] [-d] [-D] page ...
       man [-?V]
```

- Opzione più usata :

  man -k keyword stampa le informazioni del manuale che contengono la keyword

Per chiedere informazioni - (1)

- Altri comandi basati su man:
  - `apropos <keyword>` – come man -k keyword
  - `whatis <keyword>` – scrive una riga riguardo al comando
  - `info` – consultazione dei documenti relativi a prodotti GNU
  - `file <nomefile>` – determina il tipo di file di nomefile
  - `whereis <comando>` - trova il binario, il sorgente e il manuale di un comando
  - `help nomecomando`

Per chiedere informazioni - (2)

- `date` → visualizza la data. Esempio:

```
$ date
Wed Mar 25 18:20:49 MET 2003
$
```

- `who` → mostra gli utenti attualmente collegati. Esempio:

```
$ who
em       tty7         2014-12-03 10:07 (:0)
em       pts/0        2014-12-03 10:27 (:0.0)
```

- `Uptime` → tempo di vita di un sistema, nr. utenti collegati, carico del sistema negli ultimi 1, 5, 15'

```
$ uptime
18:21  up  4:21,  5 users,  load average: 0.00, 0.06, 0.05
$
```

- `hostname` → nome dell'host

```
$ hostname
DIA-LNX011
```

Per chiedere informazioni - (3)

- users → visualizza gli utenti collegati attualmente- Esempio:

```
$ users
em em
```

- w → visualizza cosa stanno facendo gli utenti.

Qualche opzione:

-h sopprime l'intestazione

-l formato lungo d'uscita (default)

```
$ w
 10:38:12 up 31 min,  2 users,  load average: 0.57, 0.59, 0.59
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
em       tty7     :0                10:07   30:45   2:41   0.24s gnome-session
em       pts/0    :0.0              10:27    0.00s  0.60s  0.02s w
```

Comandi per il file system

- `ls [-1ltsaRn] [filename]` → lista il contenuto della directory

QUALCHE OPZIONE:

-1 stampa su una colonna

-l formato lungo

-n come –l ma visualizza gli ID al posto del nome del proprietario e del gruppo

-t ordina per data

-s mostra la dimensione dei file in blocchi

-a mostra tutti i file compresi . e ..

-R elenca il contenuto in modo ricorsivo

-n mostra UID e GID

-i mostra il nr. di i-node

NB: il primo carattere puo' essere: d (dir), l(sym.link), b(block) ,c(char), p(pipe),s(socket), -

Es.:

$ls -1

10

11

1q

2

…

Comandi per il file system

```
mumolo@DIA-LNX011:~$ ls -l
total 6956
-rw-rw-r--  1 mumolo mumolo     364 set 20 12:18 C:\nppdf32Log\debuglog.txt
drwxr-xr-x 22 mumolo mumolo    4096 set 25 11:21 Desktop
drwxr-xr-x  2 mumolo mumolo    4096 set 19 19:37 Documents
drwxr-xr-x  3 mumolo mumolo    4096 set 27 10:41 Downloads
-rw-r--r--  1 mumolo mumolo    8980 set 19 19:29 examples.desktop
drwxr-xr-x  2 mumolo mumolo    4096 set 19 19:37 Music
drwxrwxr-x  2 mumolo mumolo    4096 set 20 17:42 MyOldPC
drwxr-xr-x  2 mumolo mumolo    4096 set 19 19:37 Pictures
drwxr-xr-x  2 mumolo mumolo    4096 set 19 19:37 Public
drwxr-xr-x  2 mumolo mumolo    4096 set 19 19:37 Templates
-rw-r--r--  1 mumolo mumolo 6904458 set 13 12:17 test1.pdf
-rw-rw-r--  1 mumolo mumolo  124155 set 26 13:25 testbig.txt
-rw-rw-r--  1 mumolo mumolo   24394 set 26 13:25 testmed.txt
-rw-rw-r--  1 mumolo mumolo     123 set 26 11:23 testsmall.txt
drwxr-xr-x  2 mumolo mumolo    4096 set 19 19:37 Videos
drwxrwxr-x  5 mumolo mumolo    4096 set 26 15:23 vmware
drwxrwxr-x  4 mumolo mumolo    4096 set 26 15:23 VMWARE
```

Comandi per il file system

```
$ ls -n
total 17354
-rw-r--r-- 1 3281 15006 12249 May 8 1997 10
-rw-r--r-- 1 3281 15006 7781 May 8 1997 11
-rw-r--r-- 1 3281 15006 775 Jan 23 11:38 1q
-rw-r--r-- 1 3281 15006 1 Nov 16 1995 2
-rw-r--r-- 1 3281 15006 8689 Nov 25 1999 20db.eps
-rw-r--r-- 1 3281 15006 21175 Apr 13 1996 3
-rw-r--r-- 1 3281 15006 1545 Jun 5 1998 38
-rw-r--r-- 1 3281 15006 7780 Feb 15 1999 40n_1.ps
…
$ ls –il
total 17354
49828 -rw-r--r-- 1 mumolo calcolat 12249 May 8 1997 10
49815 -rw-r--r-- 1 mumolo calcolat 7781 May 8 1997 11
49695 -rw-r--r-- 1 mumolo calcolat 775 Jan 23 11:38 1q
49877 -rw-r--r-- 1 mumolo calcolat 1 Nov 16 1995 2
50220 -rw-r--r-- 1 mumolo calcolat 8689 Nov 25 1999 20db.eps
50017 -rw-r--r-- 1 mumolo calcolat 21175 Apr 13 1996 3
50145 -rw-r--r-- 1 mumolo calcolat 1545 Jun 5 1998 38
```

Comandi per il file system

- `rm [-rfi] [filename]` → rimuove il/i file ordinari selezionati

QUALCHE OPZIONE

-r rimozione ricorsiva del contenuto delle directories. I link simbolici incontrati non vengono considerati. Una rimozione di una directory non vuota e protetta in scrittura fallisce sempre!!

-f rimozione di tutti i file (anche protetti in scrittura) senza avvisare. Se la directory e' protetta in scrittura i file non vengono mai rimossi ma non viene mostrato nessun avviso.

-i con questa opzione rm chiede conferma

ESEMPIO

rm –r prova cancella tutto dalla directory prova in giu'

rm –r prova/* cancella tutti i file e directory da prova in giu' ma non prova

- `rmdir [directory]` → rimuove una directory vuota

Comandi per il file system

- `cd <dir>` → cambia directory. Punto (.) è la directory corrente.
  Doppio punto (..) è la radice della directory corrente

NOTA
Il carattere ~ significa home directory
Il permesso di scrittura in un directory significa ricerca!

```
$ ls -l prova
total 4
drwxr-xr-x 2 mumolo calcolat 512 Mar 5 14:09 sub1
drwxr-xr-x 2 mumolo calcolat 512 Mar 5 14:09 sub2
$ chmod 600 prova
$ ls prova
sub1 sub2
$ ls -l prova
prova/sub1: Permission denied
prova/sub2: Permission denied
total 0
```

- `mkdir (crea sub-directory)`

Comandi per il file system

- `cp <file1> <file2>` → copia file

QUALCHE OPZIONE

-f Unlink. Se il file di destinazione non puo' essere sovrascritto, lo rimuove e continua

-i Interattivo. cp chiede conferma nel caso che la copia sovrascriva il file di destinazione

-r Ricorsivo. cp copia la directory e tutti I suoi file, incluso le sottodirectory ed i loro file alla destinazione

-R uguale a -r, eccetto che le pipe sono duplicate senza leggerle

Es. copia di una gerarchia di file:

$cp –r dirA/* dirB → tutta la gerarchia dei file da dirA in giu' si ritrova da dirB in giu'

$cp –r dirA dirB → sotto dirB si ritrova dirA con tutti i file e le directories

- `mv <file1> <file2>` → muove file

QUALCHE OPZIONE

-f mv muove i files senza chiedere conferma anche se sovrascrive una destinazione esistente

-i mv chiede conferma nel caso che il movimento sovrascriva la destinazione.

Comandi per il file system

- `ln <sorgente> <destinazione>` → Hard-Link. Aggiunge nella directory una coppia nome file – nr. inode senza aggiunger un file: lo stesso file avra' piu' nomi aumenta il numero di link. La cancellazione decrementa il nr. di link.

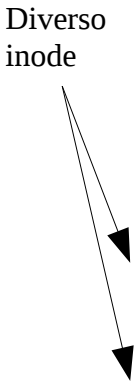
ESEMPIO:

```
$ ls -i miofile.txt
2360592 miofile.txt

$ ln miofile.txt mio

$  ls -il mio*
2360592 -rw-rw-r-- 2 mumolo mumolo 13 set 27 13:11 mio
2360592 -rw-rw-r-- 2 mumolo mumolo 13 set 27 13:11 miofile.txt

$ rm -i miofile.txt
rm: remove regular file ‘miofile.txt’? Y

$ ls -il mio*
2360592 -rw-rw-r-- 1 mumolo mumolo 13 set 27 13:11 mio
```

NOTA: il file linkato appare come file regolare!

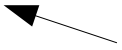
Comandi per il file system

- `ln –s <sorgente> <destinazione>` → Symbolic o Soft Link.
- A differenza del link hard, crea un nuovo file, con un nuovo inode
- Tipo file link simbolico: l



ESEMPIO:

```
$ ln -s mio altro
$ ln -s mio suo
$ ls -il mio
49671 -rw-r--r-- 1 mumolo calcolat 12 Mar 5 16:26 mio
$ ls -il altro
49760 lrwxrwxrwx 1 mumolo calcolat 3 Mar 5 16:44 altro -> mio
$ ls -il suo
49762 lrwxrwxrwx 1 mumolo calcolat 3 Mar 5 16:44 suo -> mio
$
```



NOTA: i file linkati appaiono come link

- `chmod file` → cambia i permessi:
  - chmod pga <nomefile> dove p, g e a sono i permessi per proprietario, gruppo e tutti gli altri in ottale.
  - Esempio: `$chmod 666 mio`

Comandi per il file system

Esempi di chmod:

```
$ ls -l mio
-rw-r--r-- 1 mumolo calcolat 12 Mar 5 16:26 mio
$ chmod 666 mio
$ ls -l mio
-rw-rw-rw- 1 mumolo calcolat 12 Mar 5 16:26 mio
$ chmod uga-w+r-x mio
$ ls -l mio
-r--r--r-- 1 mumolo calcolat 12 Mar 5 16:26 mio
```

- `find [path] [-n nome] [-print]` → ricerca ricorsiva di directories

Esempio piu' semplice (mostra tutti i file utente):

```
$ find . -print      → stampa tutti I file dalla directory corrente
./.wastebasket       → ricerca dalla directory corrente
./sub                → stampa tutti I file alla dsotto directory sub
./sub/dati
./sub/dato
```

- `chown nuovo_user file` → cambia proprietario di file

Comandi per il file system

Altro esempio di find:

```
$ find . -name "*.c" -print
/net/cli-ser/gc.c
./net/cli-ser/ech.c
./net/cli-ser/gech.c
./net/echo-cli/echoclient.c
./net/finger/finger.c
./net/finger/lprint.c
find: cannot read dir ./qq/: Permission denied
./phase/PVC-3.0-linux/PVC-3.0-linux/CMUSIC_GEN/gen/cspline.c
./phase/PVC-3.0-linux/PVC-3.0-linux/CMUSIC_GEN/gen/gen0.c
```

…

NOTA: find METTE A DISPOSIZIONE ALTRI PREDICATI:

-perm (selezione sui permessi), -type (selezione sul tipo di file), -user (selezione sull'utente), -group (selezione sul gruppo), -size (selezione sulla dimensione: esatta con =, inferiore con <, superiore con >

ESEMPIO:

find . –mtime 0 → file modificati 0 giorni fa

find . –mtime 1 → file modificati ieri

find . –mtime –2 → file modificati oggi e ieri

Comandi per il file system

- grep [OPTIONS] PATTERN [FILE...] : stampa le linee del file che contengono il pattern.

  Se non metto il file usa lo standard input: posso usare grep in pipe

- Alcune opzioni:

`-n` → stampa il numero di riga

`-i` → case insensitive

`-r` → ricerca ricorsiva nelle sottodirectory

`-c` → stampa il numero di match

Esempi:

```
$grep main *
$grep –r main *.c
$ls –l | grep rwxrwxrwx
$ps –ef | grep bash
$ps –ef | grep root
```

Comandi per il file system

- `touch [file]` → aggiorna la data di ultima modifica del file alla data corrente

ESEMPIO:

```
$ ls -l a*.c
-rw-r--r-- 1 mumolo calcolat 11146 May 12 1998 array.c
-rw-rw-rw- 1 mumolo calcolat 400 Jan 14 1997 assign.c
ingsun2/home/mumolo $ ls -l b*.c
-rw-r--r-- 1 mumolo calcolat 905 Jan 14 1997 back.c
-rw-rw-rw- 1 mumolo calcolat 1033 Jan 14 1997 boh.c
-rw-rw-rw- 1 mumolo calcolat 796 Jan 14 1997 builtin.c
$ touch b*.c
$ ls -l b*.c
-rw-r--r-- 1 mumolo calcolat 905 Mar 5 14:24 back.c
-rw-rw-rw- 1 mumolo calcolat 1033 Mar 5 14:24 boh.c
-rw-rw-rw- 1 mumolo calcolat 796 Mar 5 14:24 builtin.c
$
```

Comandi per la gestione processi

- `ps [opzioni]` → mostra informazioni sui processi attivi

QUALCHE OPZIONE:

-a informazioni su tutti i processi piu' utilizzati
-d informazioni su tutti i processi tranne i processi leader
-e info su tutti e processi che girano correntemente
-g grplist mostra solo i processi che appartengono ai gruppi listati (in termini di ID dei leader)
-G gidlist mostra solo i processi con real-group-ID elencati nella lista separata da virgola o spazio
-f formato completo
-l mostra una lista con formato lungo
-L mostra informazioni sui thread attivi nei processi selezionati
-o format mostra informazioni secondo un formato specificato
-p proclist mostra solo i processi con PID elencati
-t term mostra solo i processi associati con il terminale indicato
-u uidlist mostra solo i processi con l'effective-user-ID elencato
-U uidlist mostra solo i processi con i real-user-ID elencati

Comandi per la gestione processi

FORMATI D'USCITA
S (l) lo stato del processo:
O running
S il processo e' in attesa di un evento
R pronto in memoria
Z stato di Zombie: il processo e' terminato ma il padre non lo sta
aspettando
T stoppato
UID (f,l) effective-user-ID del processo
PID (all) il process-ID
PPID (f,l) il process-ID del padre
PRI (l) la priorita' del processo
SZ (l) dimensione totale (virtuale) del processo
STIME (f) l'istante di partenza del processo, in ore, minuti, secondi
TTY (all) il terminale che controlla il processo. Se non c'e' nessun terminale Å ?
TIME (all) il tempo cumulative di esecuzione
Se e' specificato –j:
PGID il PID del leader del gruppo al quale appartiene il processo
Se e' stato specificato –L:
LWP l'ID del thread
NLWP il numero di thread del processo

Comandi per la gestione processi

Esempi di ps

```
$ps
  PID TTY          TIME CMD
 2561 pts/0    00:00:00 bash
 3386 pts/0    00:00:00 ps
```

Nr di processi 'children'

Starting time

Execution time

```
$ps -f   → formato 'full'
UID        PID  PPID  C STIME TTY          TIME CMD
mumolo    2561  2551  0 10:48 pts/0    00:00:00 bash
mumolo    3388  2561  0 12:08 pts/0    00:00:00 ps -f

$ps -e        → tutti i processi che stanno eseguendo

$ps -ef
```

Process size

Real memory size

Il core assegnato al processo

```
$ps -F       → formato esteso
UID        PID  PPID  C    SZ   RSS PSR STIME TTY          TIME CMD
mumolo    2561  2551  0  6742  5564   1 10:48 pts/0    00:00:00 bash
mumolo    3571  2561  0  5662  2740   3 12:30 pts/0    00:00:00 ps -F
```

Comandi per la gestione processi

- `kill -s PID` → manda il segnale s (numerico o mnemonico) al processo PID, che deve appartenere all'utente (oppure root)

ESEMPIO:

```
kill -9 100 –165 → manda il segnale 9 al processo con PID=100
e a tutti i processi del gruppo 165
kill -s kill 100 -165 → stessa cosa, segnale mnemonico
kill -s KILL 100 -165
```

- `top` → mostra i processi in esecuzione visualizzando il loro carico

Dispositivi

- `du [directory]` → visualizza per ogni subdirectory, il numero di blocchi usati)

- `df` → visualizza il nome del file system, nome dispositivo, nr. Blocchi liberi, i-node disponibili. Blocchi di 512 byte.

$df

```
Filesystem     1K-blocks      Used Available Use% Mounted on
udev             3923948         4   3923944   1% /dev
tmpfs             787220      1552    785668   1% /run
/dev/sda6      425085288 259786508 143682652  65% /
none                   4         0         4   0% /sys/fs/cgroup
none                5120         0      5120   0% /run/lock
none             3936080       436   3935644   1% /run/shm
none              102400        44    102356   1% /run/user
```

- lsusb lista i dispositivi usb

- lspci lista i dispositivi su bus pci

Visualizzazione dei file

- `head [-count] [file] count e' il nr. di linee da visualizzare (default 10)`
- `tail [-count] [file] (vedi sopra)`
- `more [-dlfpcsu] [-num] [+/pattern] [+linenum] [file …]`
- `less [-# shift] [+[+]cmd] [--] [filename]...`
- `most [-1bCcdMstuvwz][+lineno] [+c] [+d] [+s] [+u] [+/string] [filename...]`
- `pg [-cn] [+startline] [+/pattern] [file]`

Visualizzano il file una schermata alla volta fornendo il prompt dopo ogni schermata.

- » Per avanzare di schermata -> spazio.
- » Al prompt si possono dare alcuni comandi,
- » more è un vecchio programma: non può tornare indietro
- » less è più recente con più funzioni. Torna idietro con 'b
- » most è il più recente. Si sposta nel file con con le frecce
- » pg si muove con -1, +1

Filtri in Unix

DEF: UN FILTRO E' UN PROGRAMMA CHE LEGGE DALLO STANDARD INPUT E SCRIVE UN QUALCHE RISULTATO SULLO STANDARD OUTPUT.

IL MODO PIU' OVVIO DI COMUNICAZIONE TRA FILTRI E' MEDIANTE UNA PIPELINE

- `cat` (concatenate) Non fa' nulla: copia dallo standard input allo standard output.

Legge dalla tastiera fino ad EOF (^d). Usi di cat:

1) per creare un file: cat > file

2) per aggiungere dati ad un file esistente: cat >>file

3) per visualizzare un file: cat <file

4) per copiare due file: cat < file1 > file2

Estensioni: concatenazione di file. cat file1 file2 file3 > file4

Opzioni:

cat [-bns] file

-n numera le righe

-nb non numera le righe bianche

-s sostituisce piu' linee bianche con una linea

Filtri in Unix

- `cut` → Estrae colonne o campi di dati dal file.

Possibili sintassi:

cut –blista [file] o cut –bn1-n2 [file] dove –b specifica la posizione in byte all'interno di ogni riga

cut –clista [file] dove –c specifica la posizione in caratteri

cut –flista [–d<carattere>] [file] dove –f specifica la posizione in campi e –d il delimitatore

QUALCHE OPZIONE

list lista separata da virgole o spazi. Esempio: 1,4,7 oppure 3-

-b list esempio –b1-72 sono I primi 72 byte

-c list esempio: -c1-72 individua I primi 72 caratteri

-d delim il carattere che segue –d e' il delimitatore di campo (solo con opzione –f) default e' Tab

ESEMPI:

```
$cat > a.txt
Questo e' un file di testo
preso come esempio per
realizzare alcune funzioni
di elaborazioni di testo
in Unix.
```

Filtri in Unix

```
$ cut -c1-5 a.txt
Quest
preso
reali
di el
in Un
$ cut -d' ' -f1-2 a.txt
Questo e'
preso come
realizzare alcune
di elaborazioni
in Unix.
$ cut -d' ' -f3-4 a.txt
un file
esempio per
funzioni
di testo
```

Filtri in Unix

- `paste [-d char]` file combina colonne di dati.

L'opzione –dchar mette il char tra i campi

ESEMPIO:

```
$ cut -c1-5 a.txt > a1.txt
$ cut -c6-7 a.txt > a2.txt
$ cut -c8- a.txt > a3.txt
$ paste a1.txt a2.txt a3.txt
```

Quest o e' un file di testo
preso c ome esempio per
reali zz are alcune funzioni
di el ab orazioni di testo
in Un ix .

$ paste -d' ' a1.txt a2.txt a3.txt
Quest o e' un file di testo
preso c ome esempio per
reali zz are alcune funzioni
di el ab orazioni di testo
in Un ix .

Filtri in Unix

- `crypt [key]` codifica dati mediante una chiave. Uso: crypt > file e crypt < file

Esempio

```
$ crypt < a.txt > acrypt.txt
Enter key:
$ cat acrypt.txt
fÅ¾Ã"µrÄ„` ‡ú|ÎÈ5×´ ^ÁÚ¼§!¸ZØÎV
ÌÈ
u,ê…q»617„Fæ[ïøúY¾v›¬“½aa•¾ÿJLK
¨òE¤•ƒ L ;áœÊî:9éØÇ'š ‡Qkú‡ü?0z]
$
$crypt < acrypt.txt
Enter key:
Questo e' un file di testo
preso come esempio per
realizzare alcune funzioni
di elaborazioni di testo
in Unix.
$
```

Altri comandi importanti

- `sort [-dfnru] [-o outfile] [file…]`

Ordina i dati del file.

–f tratta maiuscole come minuscole.
–n riconosce i numeri e li ordina in modo numerico.
–r ordina i dati in modo inverso.
-k ordina secondo il numero di colonna dato dopo il k
-u ordina e rimuove linee duplicate

- Esempi

ls -l | sort -k5 -n ordina secondo la dimensione
ls -l | sort -k9 ordina secondo il nome del file
ls -l|sort -M -k6 ordina secondo i mesi

sort –n [-o outfile] sortedfile.. Legge file gia' ordinati e li fonde.

Altri comandi importanti

- `tr [-ds] [set1] [set2]` Legge dei dati e sostituisce i caratteri specificati con altri caratteri.
  Esempio `tr a A < file1 > file2`

Se il secondo set e' piu' corto del primo, l'ultimo carattere e' ripetuto.

Opzioni:

-d cancella tutti i caratteri specificati

-s sostituisce le ripetizioni del carattere specificato con un solo carattere

Importante: tr -s " " → sostituisce spazio ai caratteri ripetuti. Vedi $ls -l | tr -s " "

- `uniq [-cdu] [infile] [outfile]` Esamina i dati linea per linea cercando linee duplicate e puo': tenere solo le linee duplicate (-d), tenere solo le linee uniche (-u), eliminare le linee duplicate e contare quante volte le linee sono duplicate (-c).

- `wc [-lwc] [file]` Conta linee (l), parole(w) e caratteri(c) dello standard input o del file

- echo stringa stampa la stringa

- read variabile legge in variabile

Gestione della rete

`ping [option(s)] host name|IP address`

Manda un piccolo pacchetto alla destinazione e ottiene immediata risposta

`nslookup`

Risove conversioni di nome in indirizzi IP

`telnet [option(s)] host name o IP address`

Protocollo di comunicazione con calcolatore remoto

ftp `[option(s)] host name|porta`

Trasferimento di file

wget `[option(s)] URL`

Scarica un file dall'indirizzo indicato

`netstat`

Visualizza connessioni, tabelle di routing, statistiche di interfaccia etc

Varie

- Esecuzione di piu' comandi indipendenti in una riga: separazione con ;

Esempio:

```
$ls;ps;date
    a   b   C:\nppdf32Log\debuglog.txt  Documents  examples.desktop  missfont.log  MyOldPC
    p~ p.sh   Public  Templates  testbig.txt  testsmall.txt  vmware   a~  b~  Desktop
    Downloads  file.txt          Music         p        Pictures  p.sh~  sub     test1.pdf
    testmed.txt  Videos
  PID TTY          TIME CMD
 2817 pts/0    00:00:00 bash
 3578 pts/0    00:00:00 ps
mar  4 ott 2016, 11.15.45, CEST
```

- Raggruppamento di un comando con $(...)

Esempio:

```
$ls -1|head -1                → visualizza il nome del primo file
$cp $(ls -1|head -1) primo    → copia il primo file in primo
```

Qualche esercizio

Linee di comando linux

- Terminare un processo dato per nome
- Rimuovere un file dando il suo Iode
- Scrivere il nome del file con piu' piccolo Inode della directory corrente
- Scrivere il nome del file con più grande Inode
- Copiare il file con l'inode più piccolo nel file 'small'
- Scrivere dimensione e nome del file più grande
- Scrivere dimensione e nome del file più piccolo
- Link hard del file più grande in 'large'
- Link simbolico del file più piccolo in 's'
- Copiare il file piu' piccolo nel file 'smaller'
- Numero di directory nella directory corrente
- Nome del processo che sta eseguendo da +tempo
- Scrivere il PPID del processo che sta eseguendo da +tempo
- Scrivere il PPID del processo dato come argomento