



Sistemi
Operativi

Bruschi
Monga Re

Shell

Shell
programming

Esercizi

I/O

Esercizi

Tabella
riassuntiva

Shell e file
system

File system

Unix power
tools

find

Archivi

Sistemi Operativi¹

Mattia Monga

Dip. di Informatica
Università degli Studi di Milano, Italia
mattia.monga@unimi.it

a.a. 2016/17

¹ © 2008–17 M. Monga. Creative Commons Attribuzione — Condividi allo stesso modo 4.0 Internazionale. <http://creativecommons.org/licenses/by-sa/4.0/deed.it>. Immagini tratte da [?] e da Wikipedia.



Sistemi
Operativi

Bruschi
Monga Re

Lezione XIV: Unix power tools

Shell

Shell
programming

Esercizi

I/O

Esercizi

Tabella
riassuntiva

Shell e file
system

File system

Unix power
tools

find

Archivi

Un vero linguaggio di programmazione



La shell è un vero e proprio linguaggio di programmazione (interpretato)

- Variabili (create al primo assegnamento, uso con \$, `export` in un'altra shell).
 - `x="ciao" ; y=2 ; /bin/echo "$x $y $x"`
- Istruzioni condizionali (valore di ritorno 0 \rightsquigarrow true)
 - ```
if ls piripacchio; then echo ciao; else echo buonasera; fi
```
- Iterazioni su insiemi
  - `for i in a b c d e; do /bin/echo $i; done`
- Cicli
  - `/usr/bin/touch piripacchio`  
`while /bin/ls piripacchio; do`  
 `/usr/bin/sleep 2`  
 `/bin/echo ciao`  
`done & ( /usr/bin/sleep 10 ; /bin/rm piripacchio )`

Sistemi Operativi

Bruschi Monga Re

Shell

Shell programming

Esercizi

I/O

Esercizi

Tabella riassuntiva

Shell e file system

File system

Unix power tools

find

Archivi



- 1 Per ciascuno dei file `dog`, `cat`, `fish` controllare se esistono nella directory `/bin` (hint: usare `/bin/ls` e nel caso scrivere ‘‘Trovato’’)
- 2 Consultare il manuale (programma `/usr/bin/man`) del programma `/bin/test` (per il manuale man `test`)
- 3 Riscrivere il primo esercizio facendo uso di `test`

Shell

Shell  
programming

Esercizi

I/O

Esercizi

Tabella  
riassuntiva

Shell e file  
system

File system

Unix power  
tools

find

Archivi

# Input e Output



In generale il paradigma UNIX permette alle applicazioni di fare I/O tramite:

## Input

- Parametri al momento del lancio
- Variabili *d'ambiente*
- File (tutto ciò che può essere gestito con le syscall `open`, `read`, `write`, `close`)
  - Terminale (interfaccia testuale)
  - Device (per es. il mouse potrebbe essere `/dev/mouse`)
  - Rete (socket)

## Output

- Valore di ritorno
- Variabili *d'ambiente*
- File (tutto ciò che può essere gestito con le syscall `open`, `read`, `write`, `close`)
  - Terminale (interfaccia testuale)
  - Device (per es. lo schermo in modalità grafica potrebbe essere `/dev/fb`)
  - Rete (socket)

Sistemi Operativi

Bruschi  
Monga Re

Shell

Shell programming  
Esercizi

I/O

Esercizi  
Tabella riassuntiva

Shell e file system

File system

Unix power tools

find

Archivi



Ad ogni processo sono sempre associati tre file (già aperti)

- Standard input (Terminale, tastiera)
- Standard output (Terminale, video)
- Standard error (Terminale, video, usato per le segnalazione d'errore)

Possono essere *rediretti*

- `/usr/bin/sort < lista` Lo stdin è il file lista
- `/bin/ls > lista` Lo stdout è il file lista
- `/bin/ls piripacchio 2> lista` Lo stderr è il file lista
- 

```
(echo ciao & date ; ls piripacchio) 2> errori 1>output
```



La **pipe** è un canale, analogo ad un file, bufferizzato in cui un processo scrive e un altro legge. Con la shell è possibile collegare due processi tramite una pipe anonima.

Lo stdout del primo diventa lo stdin del secondo

```
/bin/ls | sort
```

```
ls -lR / | sort | more
```

funzionalmente equivalente a

```
ls -lR >tmp1; sort <tmp1 >tmp2; more<tmp2; rm tmp*
```

Molti programmi copiano lo `stdin` su `stdout` dopo averlo elaborato: sono detti **filtri**.

Shell

Shell  
programming  
Esercizi

I/O

Esercizi  
Tabella  
riassuntiva

Shell e file  
system

File system

Unix power  
tools

find

Archivi



```
ls | sort

int main(void){
int fd[2], nbytes; pid_t childpid;
char string[] = "Hello, world!\n";
char readbuffer[80];

pipe(fd);
if(fork() == 0){
 /* Child process closes up input side of pipe */
 close(fd[0]);
 write(fd[1], string, (strlen(string)+1));
 exit(0);
} else {
 /* Parent process closes up output side of pipe */
 close(fd[1]);
 nbytes = read(fd[0], readbuffer, sizeof(readbuffer));
 printf("Received string: %s", readbuffer);
}
return(0);
}
```

Shell

Shell  
programming  
Esercizi

I/O

Esercizi  
Tabella  
riassuntiva

Shell e file  
system  
File system

Unix power  
tools  
find  
Archivi





```
if(fork() == 0)
{
 /* Close up standard input of the child */
 close(0);

 /* Duplicate the input side of pipe to stdin */
 dup(fd[0]);
 execlp("sort", "sort", NULL);
}
```

Shell

Shell  
programming  
Esercizi

I/O

Esercizi  
Tabella  
riassuntiva

Shell e file  
system  
File system

Unix power  
tools  
find  
Archivi



Con una pipe è possibile “collegare” lo `stdout` di un programma con lo `stdin` di un altro.

Per usare l'output di un programma sulla riga di comando di un altro programma, occorre usare la **command substitution**

```
/bin/ls -l $(/usr/bin/which sort)
```

Shell

Shell  
programming  
Esercizi

I/O

Esercizi  
Tabella  
riassuntiva

Shell e file  
system

File system

Unix power  
tools

find

Archivi



- 1 Scrivere una *pipeline* di comandi che identifichi il le informazioni sul processo dropbear (`ps`, `grep`)
- 2 Scrivere una *pipeline* di comandi che identifichi il solo processo con il PPID piú alto (`ps`, `sort`, `tail`)
- 3 Ottenere il numero totale dei file contenuti nelle directory `/usr/bin` e `/var` (`ls`, `wc`, `expr`)
- 4 Si immagini di avere un file contenente il sorgente di un programma scritto in un linguaggio di programmazione in cui i commenti occupino intere righe che iniziano con il carattere `#`. Scrivere una serie di comandi per ottenere il programma senza commenti. (`grep`)
- 5 Ottenere la somma delle occupazioni dei file delle directory `/usr/bin` e `/var` (`du`, `cut`)

Shell

Shell  
programming  
Esercizi

I/O  
Esercizi  
Tabella  
riassuntiva

Shell e file  
system

File system

Unix power  
tools

find  
Archivi



| Prog. (sez. man) | Descrizione                                             |
|------------------|---------------------------------------------------------|
| ls (1)           | list directory contents                                 |
| echo (1)         | display a line of text                                  |
| touch (1)        | change file timestamps                                  |
| sleep (1)        | delay for a specified amount of time                    |
| rm (1)           | remove files or directories                             |
| cat (1)          | concatenate files and print on the standard output      |
| man (1)          | an interface to the on-line reference manuals           |
| test (1)         | check file types and compare values                     |
| sort (1)         | sort lines of text files                                |
| date (1)         | print or set the system date and time                   |
| less (1)         | file perusal filter for crt viewing                     |
| which (1)        | locate a command                                        |
| ps (1)           | report a snapshot of the current processes.             |
| tail (1)         | output the last part of files                           |
| wc (1)           | print the number of newlines, words, and bytes in files |
| test (1)         | check file types and compare values                     |
| grep (1)         | print lines matching a pattern                          |
| cut (1)          | remove sections from each line of files                 |
| du (1)           | print disk usage                                        |

Sistemi Operativi

Bruschi Monga Re

Shell

Shell programming  
Esercizi

I/O

Esercizi

Tabella riassuntiva

Shell e file system

File system

Unix power tools

find

Archivi



- “A Brief Introduction to Unix (With Emphasis on the Unix Philosophy)”, Corey Satten <http://staff.washington.edu/corey/unix-intro.pdf>
- [http://en.wikipedia.org/wiki/Unix\\_philosophy](http://en.wikipedia.org/wiki/Unix_philosophy)
- “The UNIX Time-Sharing System”, Ritchie; Thompson <http://www.cs.berkeley.edu/~brewer/cs262/unix.pdf>

Shell

Shell  
programming  
Esercizi

I/O  
Esercizi

Tabella  
riassuntiva

Shell e file  
system  
File system

Unix power  
tools  
find  
Archivi



- <http://www.gnu.org/software/coreutils/manual/coreutils.html>

Shell

Shell  
programming

Esercizi

I/O

Esercizi

**Tabella  
riassuntiva**

Shell e file  
system

File system

Unix power  
tools

find

Archivi



- Ogni processo (compresa la shell stessa) ha associata una *directory di lavoro* (**working directory**), che può essere cambiata col comando (interno alla shell) `cd`
- I programmi fondamentali per operare sul file system

|                        |                                         |
|------------------------|-----------------------------------------|
| <code>ls (1)</code>    | list directory contents                 |
| <code>cp (1)</code>    | copy files and directories              |
| <code>rm (1)</code>    | remove files or directories             |
| <code>mv (1)</code>    | move (rename) files                     |
| <code>mkdir (1)</code> | make directories                        |
| <code>rmdir (1)</code> | remove empty directories                |
| <code>df (1)</code>    | report file system disk space usage     |
| <code>du (1)</code>    | estimate file space usage               |
| <code>pwd (1)</code>   | print name of current/working directory |



Ad ogni file vengono associati dei *permessi*, che definiscono le azioni permesse sui dati del file

- **Read:** leggere il contenuto del file o directory
- **Write:** scrivere (cambiare) il file o directory
- **eXecute** eseguire le istruzioni contenute nel file o accedere alla directory

| R | W | X |   |
|---|---|---|---|
| 1 | 1 | 0 | 6 |
| 1 | 0 | 1 | 5 |
| 1 | 0 | 0 | 4 |
| 1 | 1 | 1 | 7 |

I permessi possono essere diversi per 3 categorie di utenti del sistema:

- **User:** il “proprietario” del file
- **Group:** gli appartenenti al gruppo proprietario
- **All:** tutti gli altri





- Cambiare il proprietario
  - `chown utente[:gruppo] file`
- Cambiare il gruppo
  - `chgrp gruppo file`
- Cambiare i permessi
  - `chmod 755 file`
  - `chmod +x file`
  - `chmod a=rw file`
  - `chmod g-x file`
- (per creare un utente: `adduser`)

Shell

Shell  
programming  
Esercizi

I/O

Esercizi

Tabella  
riassuntiva

Shell e file  
system

File system

Unix power  
tools

find

Archivi



Il proprietario di un processo in esecuzione è normalmente *diverso* dal proprietario del file contenente un programma (e diverso ad ogni esecuzione)

- effective UID bit: il processo assume come proprietario il proprietario del file del programma
- SUID root
- `chmod 4555 file`
- `chmod u+s file`

Shell

Shell  
programming  
Esercizi

I/O

Esercizi  
Tabella  
riassuntiva

Shell e file  
system

File system

Unix power  
tools

find

Archivi



Per selezionare file con determinate caratteristiche si usa `find`  
`find` percorso predicato  
Seleziona, nel sottoalbero definito dal percorso, tutti i file per cui il predicato è vero  
Spesso usato insieme a `xargs`  
`find` percorso predicato | `xargs` comando  
funzionalmente equivalente a  
comando `$(find percorso predicato)`  
ma evita i problemi di lunghezza della riga di comando perché  
`xargs` si preoccupa di “spezzarla” opportunamente.

Shell

Shell  
programming  
Esercizi

I/O  
Esercizi

Tabella  
riassuntiva

Shell e file  
system

File system

Unix power  
tools

`find`  
Archivi

# Due espressioni idiomatiche



Sistemi  
Operativi

Bruschi  
Monga Re

Spesso si vuole fare un'operazione per ogni file trovato con `find`. L'espressione piú naturale sarebbe:

```
for i in $(find percorso predicato); do
 comando $i
done
```

Questa forma presenta due problemi: può eccedere la misura della linea di comando e non funziona correttamente se i nomi dei file contengono *spazi*

Shell

Shell  
programming  
Esercizi

I/O  
Esercizi

Tabella  
riassuntiva

Shell e file  
system

File system

Unix power  
tools

**find**

Archivi



# Due espressioni idiomatiche

Un'alternativa è

```
find percorso predicato -print0 | xargs -0 -n 1
```

In questo modo (`-print0`) i file trovati sono separati dal carattere `0` anziché spazi e `xargs` è capace di adattarsi a questa forma.

Un'alternativa piú generale che mostra la potenza del linguaggio di shell che non distingue fra comandi e costrutti di controllo di flusso (sono tutti “comandi” utilizzabili in una pipeline)

```
find percorso predicato | while read x; do
 comando $x
done
```

`read` `x` legge una stringa e la assegna alla variabile `x`.

Sistemi  
Operativi

Bruschi  
Monga Re

Shell

Shell  
programming  
Esercizi

I/O

Esercizi

Tabella  
riassuntiva

Shell e file  
system

File system

Unix power  
tools

find

Archivi



- 1 Trovare il file piú “grosso” in un certo ramo
- 2 Copiare alcuni file (ad es. il cui nome segue un certo pattern) di un ramo in un altro mantenendo la gerarchia delle directory
- 3 Calcolare lo spazio occupato dai file di proprietà di un certo utente
- 4 Scrivere un comando che conta quanti file ci sono in un determinato ramo del filesystem

Shell

Shell  
programming  
Esercizi

I/O

Esercizi

Tabella  
riassuntiva

Shell e file  
system

File system

Unix power  
tools

find

Archivi



Un archivio *archive* è un file di file, cioè un file che contiene i byte di diversi altri file e i relativi *metadati*. (Cfr. con una *directory*, che è un file speciale, che sostanzialmente contiene solo l'elenco dei file)

- `ar` L'archiviatore classico, generalmente utilizzato per le librerie (provare `ar t /usr/lib/i86/libc.a`)
- `tar` Tape archive, standard POSIX  
`tar cvf archivio.tar lista_files`

Gli archivi possono essere compressi con `compress` o, piú comunemente, con `gzip` o `bzip2`.  
I file `.zip` sono archivi compressi.

Shell

Shell  
programming  
Esercizi

I/O  
Esercizi  
Tabella  
riassuntiva

Shell e file  
system  
File system

Unix power  
tools  
find  
Archivi



Altre utility “standard” di cui è bene conoscere almeno l'esistenza

| Prog. (sez. man) | Descrizione                                           |
|------------------|-------------------------------------------------------|
| uniq (1)         | report or omit repeated lines                         |
| cut (1)          | remove sections from each line of files               |
| tr (1)           | translate or delete characters                        |
| dd (1)           | convert and copy a file                               |
| tee (1)          | read from standard input and write to standard output |
| sed (1)          | stream editor for filtering and transforming text     |
| seq (1)          | print a sequence of numbers                           |

Inoltre è molto utile conoscere le [espressioni regolari](#) (man 7 re\_format), usate da grep, sed, ecc.





Altre utility “standard” di cui è bene conoscere almeno l'esistenza

| Prog. (sez. man) | Descrizione                               |
|------------------|-------------------------------------------|
| basename (1)     | strip directory and suffix from filenames |
| dirname (1)      | strip non-directory suffix from file name |
| uniq (1)         | report or omit repeated lines             |
| cut (1)          | remove sections from each line of files   |
| tr (1)           | translate or delete characters            |
| dd (1)           | convert and copy a file                   |
| stat (1)         | display file or file system status        |

cd invece non è un programma, ma un comando interno della shell (che differenza fa?)

Shell

Shell  
programming

Esercizi

I/O

Esercizi

Tabella  
riassuntiva

Shell e file  
system

File system

Unix power  
tools

find

Archivi



- 1 Creare un archivio `tar.gz` contenente tutti i file la cui dimensione è minore di 50KB
- 2 Rinominare un certo numero di file: per esempio tutti i file `.png` in `.jpg`
- 3 Creare un file da 10MB costituito da caratteri casuali (usando `/dev/random`) e verificare se contiene la parola `JOS`
- 4 Trovare l'utente che ha il maggior numero di file nel sistema
- 5 Trovare i 3 utenti che, sommando la dimensione dei loro file, occupano più spazio nel sistema.

Shell

Shell  
programming  
Esercizi

I/O

Esercizi  
Tabella  
riassuntiva

Shell e file  
system

File system

Unix power  
tools

find

Archivi