



# Introduzione alla Shell di UNIX

Alessandra Giordani

[agiordani@disi.unitn.it](mailto:agiordani@disi.unitn.it)

Lunedì 27 febbraio 2012

<http://disi.unitn.it/~agiordani/><sub>1</sub>



# Composizione di un sistema informativo:

- Hardware (CPU, periferiche, rete di calcolatori);
- Software:
  - **Sistema operativo:** interfaccia HW/SW  
kernel, moduli, system call ...
  - **Software di base:** utilities e servizi per applicazioni  
shell, compilatori, librerie ...
  - **Software applicativo:** programmi end-user  
servizi (web server), produttività (word processor) ...



# La shell interattiva

- La shell é l'interfaccia di un sistema operativo;
- La shell permette all'utente di:
  - lanciare programmi;
  - interagire con i programmi;
  - visualizzarne i risultati.
- Generalmente, ogni OS offre
  - una shell grafica (piú semplice),
  - una shell a riga di comando (meno intuitiva ma piú potente).



# Interfaccia a riga di comando

La shell si presenta come un prompt (\$) seguito da un cursore.

Per invocare un programma:

1. si scrive il nome del programma;
2. si scrivono gli eventuali argomenti o parametri del comando;
3. si preme il tasto invio (enter).

Esempio:

- Invocazione del programma date:

```
$date<invio>
```

- N.B.: il carattere \$ non va scritto! Indica il prompt della shell.

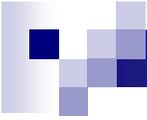


# man ti da una mano

Invocazione di `man` (con argomento):

```
$man date<invio>
```

- `man` visualizza il manuale di un programma, usatelo spesso!
- Anche `man` ha un manuale:
  - `$man man<invio>`
- Premere tasto `Q` per uscire dal manuale



# Processi e standard streams

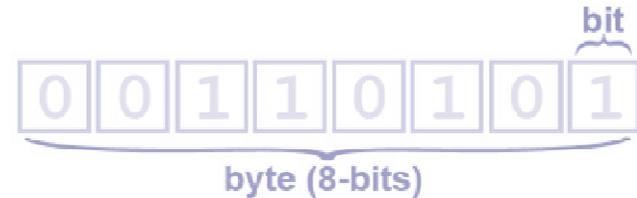
- Un programma può essere costituito da più processi
- Programmi semplici possono coincidere con un processo (es: `date`);
- Programmi complessi possono essere costituiti da centinaia di processi (es: un sistema operativo);

Ad ogni processo sono sempre associati 3 *flussi di dati*:

- **standard input** (`stdin`), per leggere l'input dell'utente;
- **standard output** (`stdout`), per richiedere input o fornire risultati;
- **standard error** (`stderr`), per scrivere messaggi di errore.

Inoltre, ogni processo può accedere a file di input/output.

# File e Filesystem



**File:** sequenza ordinata di **byte** memorizzata su un supporto

- l'interpretazione della sequenza dipende dall'utente o dal programma che ha creato il file
- i byte che compongono un file possono non essere memorizzati sequenzialmente sul supporto (frammentazione)
- il supporto contiene una mappa che consente di associare a ciascun file i frammenti che lo compongono
- il SO gestisce l'interazione tra SW e periferiche di memorizzazione

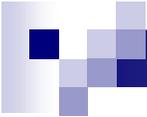
**Filesystem:** astrazione logica fornita dal SO che consente di:

- associare nomi ai file
- organizzare i file in contenitori logici (directory)



# I file di Unix

- Tipi di file Unix :
  - *regular* (-): collezione di byte non strutturata ←
  - *directory* (d) : directory ←
  - *buffered special file* (b) : file che rappresenta una periferica con interfaccia a blocchi
  - *unbuffered special file* (b) : file che rappresenta una periferica con interfaccia a caratteri
  - *link simbolico* (l) : file che rappresenta un nome alternativo per un altro file X, ogni accesso a questo file viene ridiretto in un accesso a X
  - *pipe* (p): file che rappresenta una pipe
  - *socket* (s) : file che rappresenta un socket



# Il Filesystem di Unix

È organizzato come un albero;

- tutte le directory discendono da un'unica radice (root): /  
(lo stesso carattere é utilizzato per separare i livelli della gerarchia)
- un file nel sistema é identificato univocamente dal suo percorso assoluto (absolute path), ad es: `/home/giordani/pippo.txt`
- un file in una directory é identificato univocamente dal suo percorso relativo (relative path), ad es: `pippo.txt`

Esistono delle directory speciali:

- `.` indica la directory corrente,  
ad es: `/home/giordani/.` coincide con `/home/giordani`
- `..` indica la parent directory,  
ad es: `/home/giordani/./` coincide con `/home/`
- la directory `/` é speciale perché non ha parent directory  
(piú precisamente, `/./` coincide con `/`).

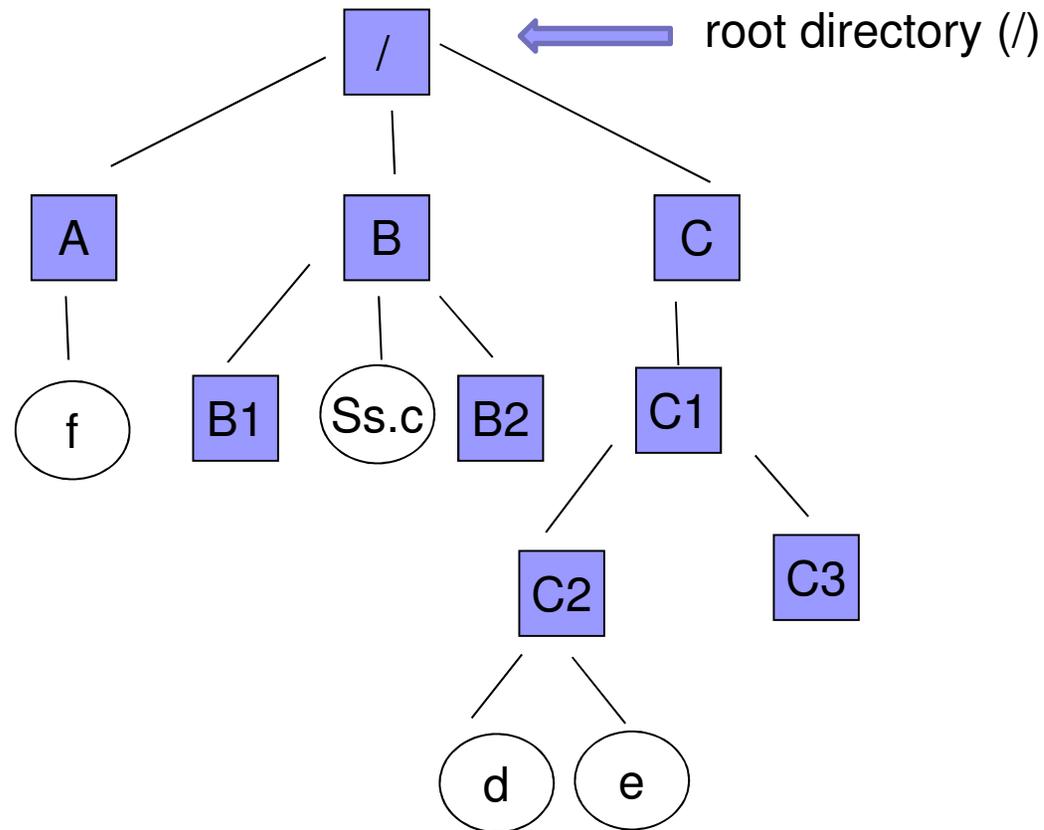


# Navigazione del Filesystem

Principali comandi (N.B.: `man <comando>` per istruzioni!):

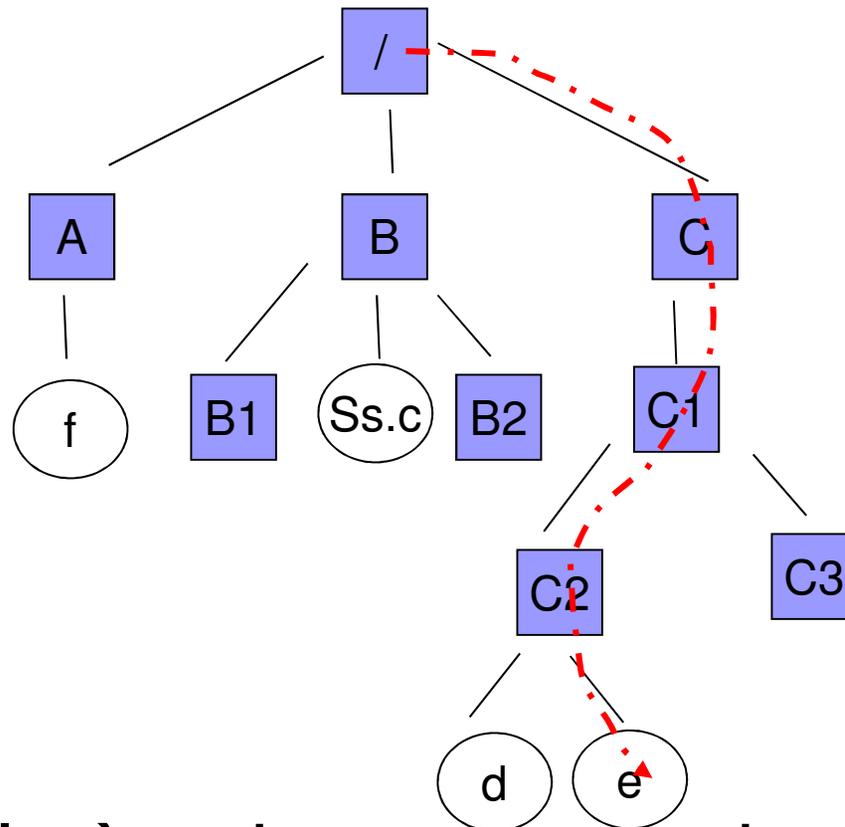
- `pwd` → visualizza directory corrente;
- `cd <dirname>` → `dirname` diventa la directory corrente.
  - Se l'argomento viene omissso, la home dell'utente diventa la directory corrente
- `ls <dirname>` → visualizza il contenuto di una directory.
  - Se l'argomento é omissso, visualizza il contenuto della directory corrente

# Il FS di Unix è gerarchico



- Esempio di FS.

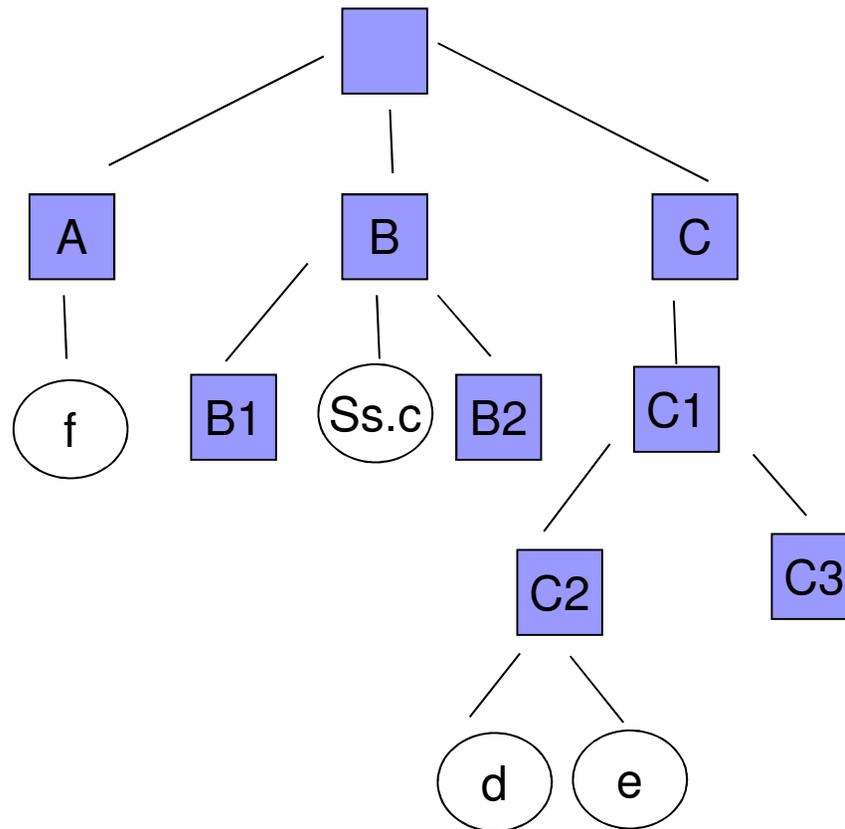
# Path name assoluto



- Ogni file è univocamente determinato dal cammino che lo collega alla radice

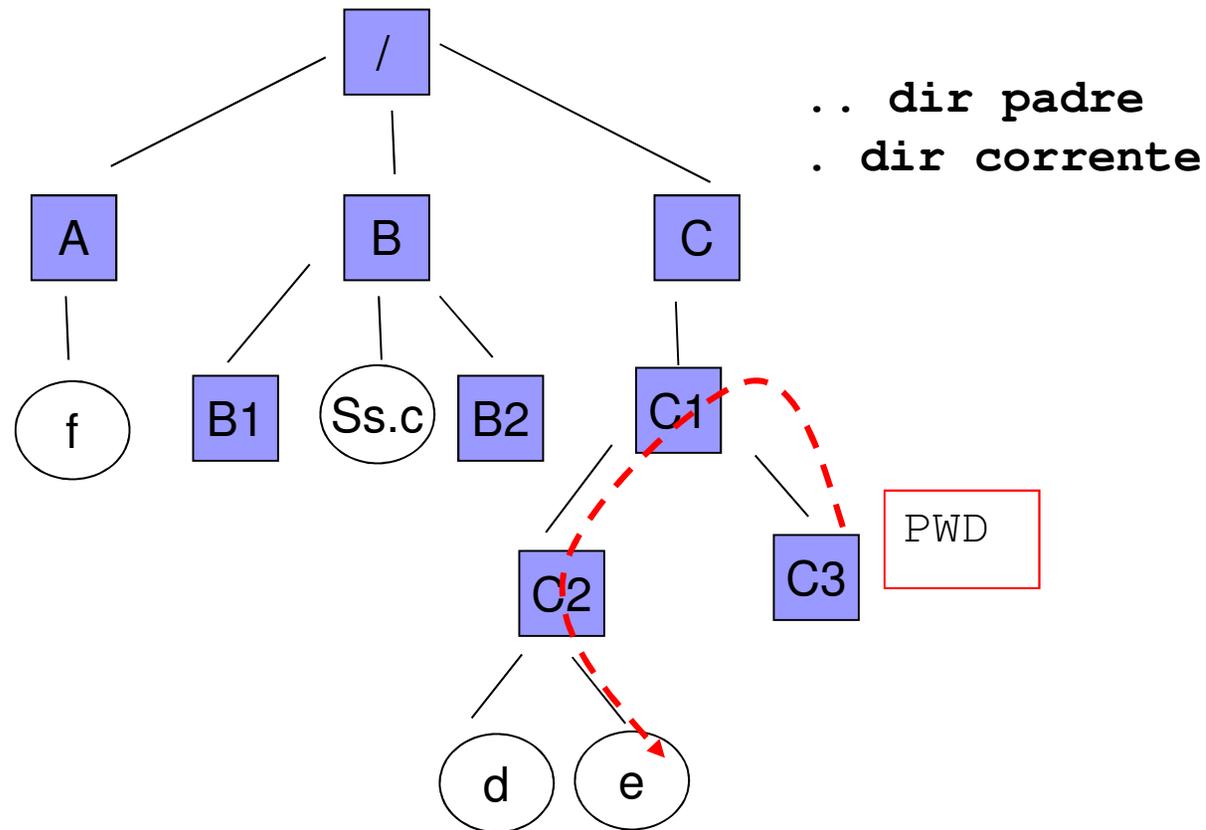
□ /C/C1/C2/e

# Path name relativo



- Ogni shell ha associata una *working directory*
  - comando `pwd` mostra la directory corrente
  - comando `cd` cambia la directory corrente

# Path name relativo (2)



- Il PNR è il cammino dalla Working Directory
  - `./../C2/e` (il '.' iniziale si può omettere)

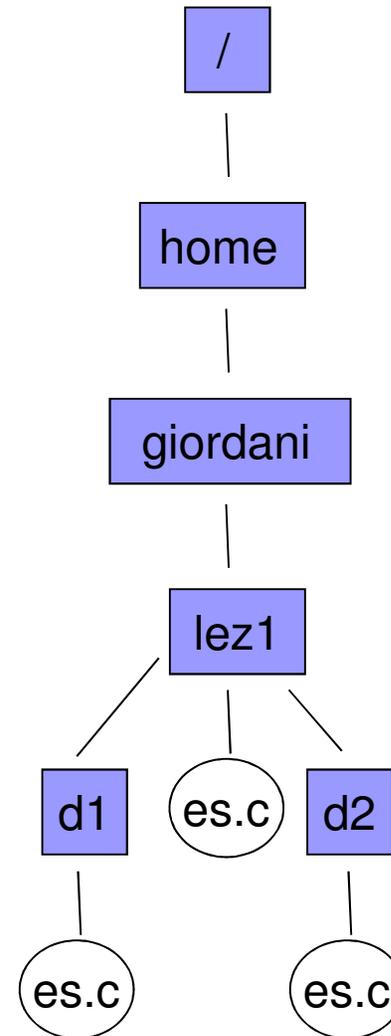


# Principali comandi

- `mkdir <dirname>` crea una directory;
- `rmdir <dirname>` elimina una directory (solo se vuota);
- `touch <filename>` crea un file (se non esistente);
- `cp <source> <dest>` copia il file source in dest;
- `cp -r <source> <dest>` copia la directory source in dest;
- `mv <oldname> <newname>` rinomina/sposta file e directory;
- `rm <filename>` elimina un file;
- `rm -r <dirname>` elimina ricorsivamente una directory e il suo contenuto.

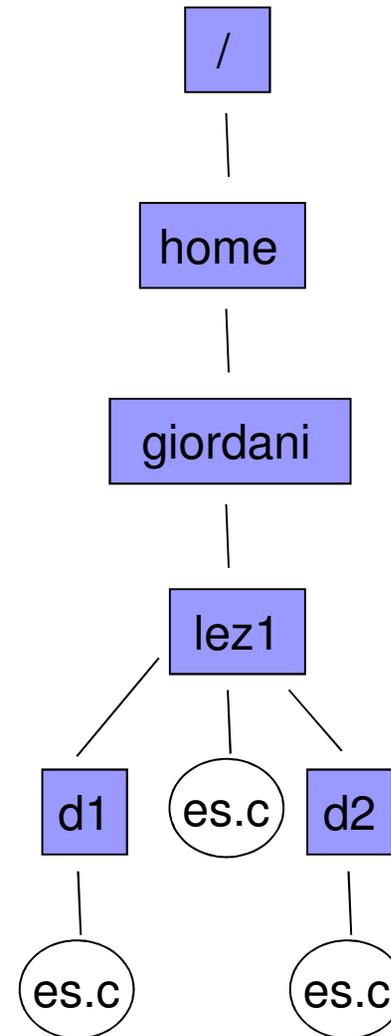
# Esercizio 1

- Riprodurre questa gerarchia nella vostra home directory.



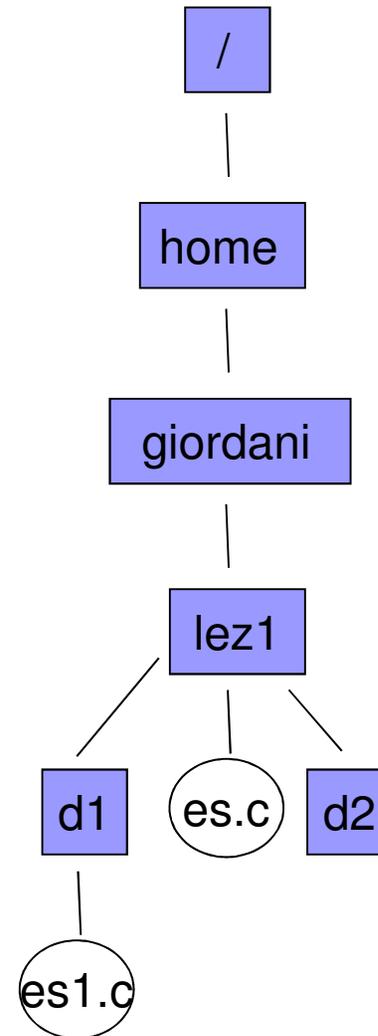
# Soluzione 1

- `pwd` dovrebbe stampare la vostra home (altrimenti usare `cd`):  
`/home/nome.cognome/`
- `mkdir lez1` crea la cartella `lez1`
- `cd lez1` entra nella cartella `lez1`
- `touch es.c` crea il file `es.c`
- `mkdir d1` crea la cartella `d1`
- `cp es.c d1/es.c` copia il file nella directory corrente (`lez1`) nella sottodirectory `d1`
- `cp -r d1 d2` copia la sottodirectory `d1` e il suo contenuto in `d2`.
- controllate il risultato usando `ls -l`



## Esercizio 2

- Modificare il filesystem per ottenere questa nuova gerarchia nella vostra home directory.



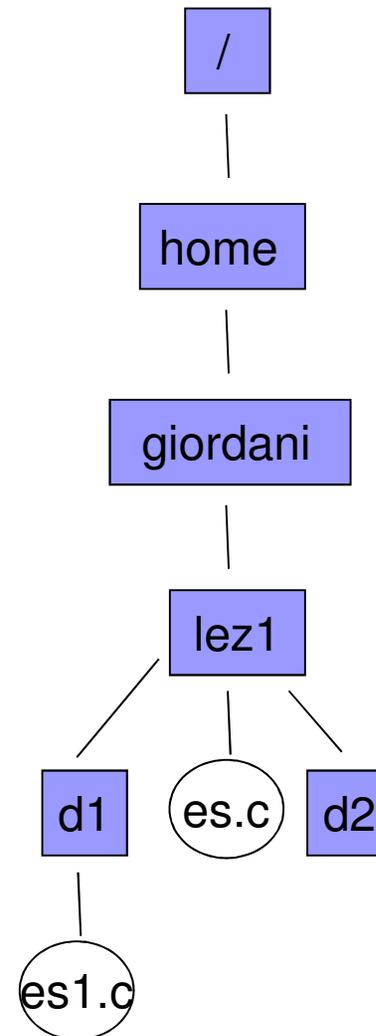
# Soluzione 2

- `pwd` dovrebbe stampare la vostra directory `lez1` (altrimenti usare `cd`):  
`/home/nome.cognome/lez1`
- `cd d1` entra nella cartella `d1`
- `mv es.c es1.c` rinomina il file
- `cd ../d2` entra nella cartella `d2`
- `rm es.c` elimina il file `es.c`

lo stesso risultato si poteva ottenere così

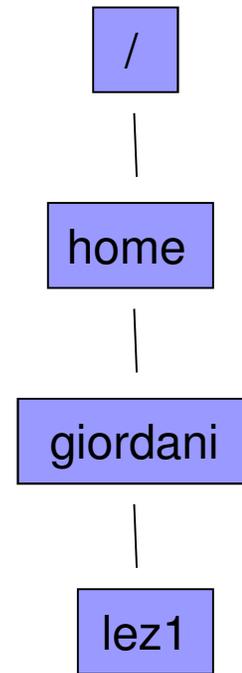
- `mv d1/es.c d1/es1.c`
- `rm d2/es.c`

controllate il risultato usando `ls -l`



# Esercizio 3

- Modificare il filesystem per ottenere questa nuova gerarchia nella vostra home directory.

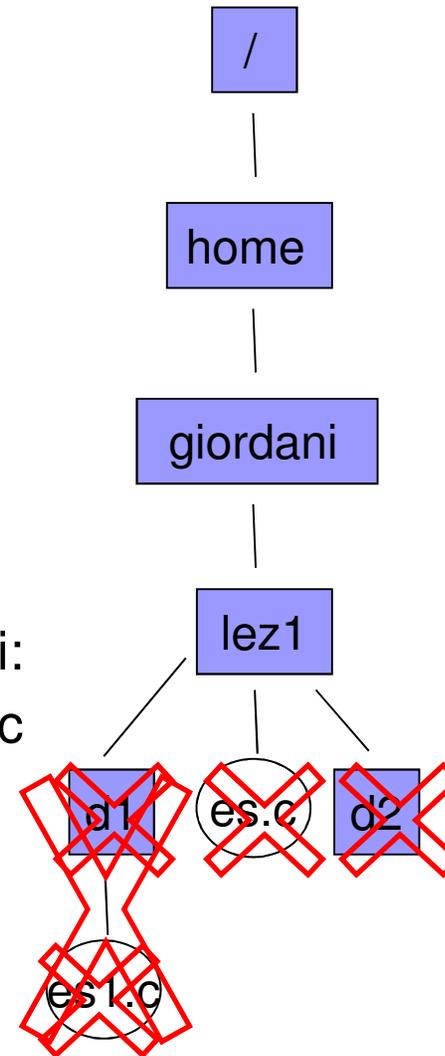


# Soluzione 3

- `pwd` dovrebbe stampare la vostra directory `lez1` (altrimenti usare `cd`):  
`/home/nome.cognome/lez1`
- `rmdir d2` elimina `d2`
- `rm es.c` elimina il file
- `rmdir d1` da errore perché la cartella non è vuota. Si può procedere in 2 modi:
  1. `rm d1/es1.c` elimina prima il file `es.c`
  2. `rmdir d1` elimina la cartella vuota

oppure:

- `rm -r d1`
- controllate il risultato usando `ls -l`



# Attributi di un file Unix

- File = nome + dati + attributi
- Alcuni attributi dei file unix :

□ es. `ls -l`

```
-rw-r--r-- 1 giordani users 0 Feb 27 2012 es.c  
drw-r--r-- 1 giordani users 1064 Feb 27 2012 d1
```

Tipo del file  
(regolare, -)  
(directory, d)

# Attributi di un file Unix (2)

- File = nome + dati + attributi
- Alcuni attributi dei file unix :

```
□ es. ls -l es.c  
-rw-r--r-- 1 giordani users 0 Feb 27 2012 es.c
```

## Protezione

r - permesso di lettura (directory, listing)

w- permesso di scrittura (directory, aggiungere file)

x - permesso di esecuzione (directory, accesso)

# Attributi di un file Unix (3)

- File = nome + dati + attributi
- Alcuni attributi dei file unix:

□ es. `ls -l es.c`

```
-rw-r--r-- 1 giordani users 0 Feb 27 2012 es.c
```

The diagram illustrates the output of the command `ls -l es.c`. The output is `-rw-r--r-- 1 giordani users 0 Feb 27 2012 es.c`. Three red boxes highlight the fields `giordani`, `users`, and `Feb 27 2012`. Arrows point from these boxes to labels below: `Proprietario del file` for `giordani`, `Gruppo` for `users`, and `Data ultima modifica` for `Feb 27 2012`.

Proprietario del file

Gruppo

Data ultima modifica

- Aprite il file usando un editor di testo:  
`gedit es.c` per scrivere qualcosa all'interno e salvate.

# Attributi di un file Unix (4)

- File = nome + dati + attributi
- Alcuni attributi dei file unix :

□ es. `ls -l es.c`

```
-rw-r--r-- 1 giordani users 1024 Feb 27 2012 es.c
```

Numero di blocchi su disco utilizzati

Lunghezza in byte  
del file  
**E' CAMBIATA!**



# CAMBIO ORARIO & AULE

- Per motivi di spazio è stato cambiato l'orario
  - lunedì dalle 11.30 alle 13.00
  - mercoledì rimane invariato
- e la separazione nelle aule:
  - Tutti i matematici in aula PC 104 (FBK)
  - I fisici il cui cognome inizia per A...P in aula PC 202 (Polo Ferrari)
  - I restanti fisici (R-Z) in aula 104 (FBK)