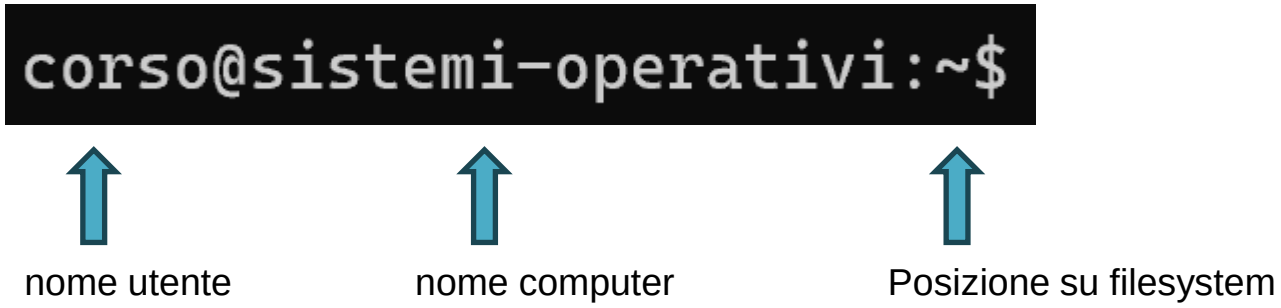
# Introduzione alla Riga di Comando

# Come interpretare la riga di comando

- Il terminale si presenta in questo modo. Ma cosa vuol dire ciò che c'è scritto?

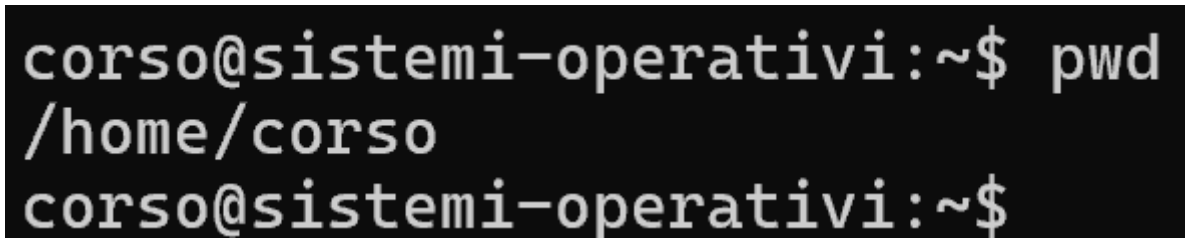


# Eseguiamo il nostro primo comando

Esecuzione comando

Output comando

Nuovo prompt



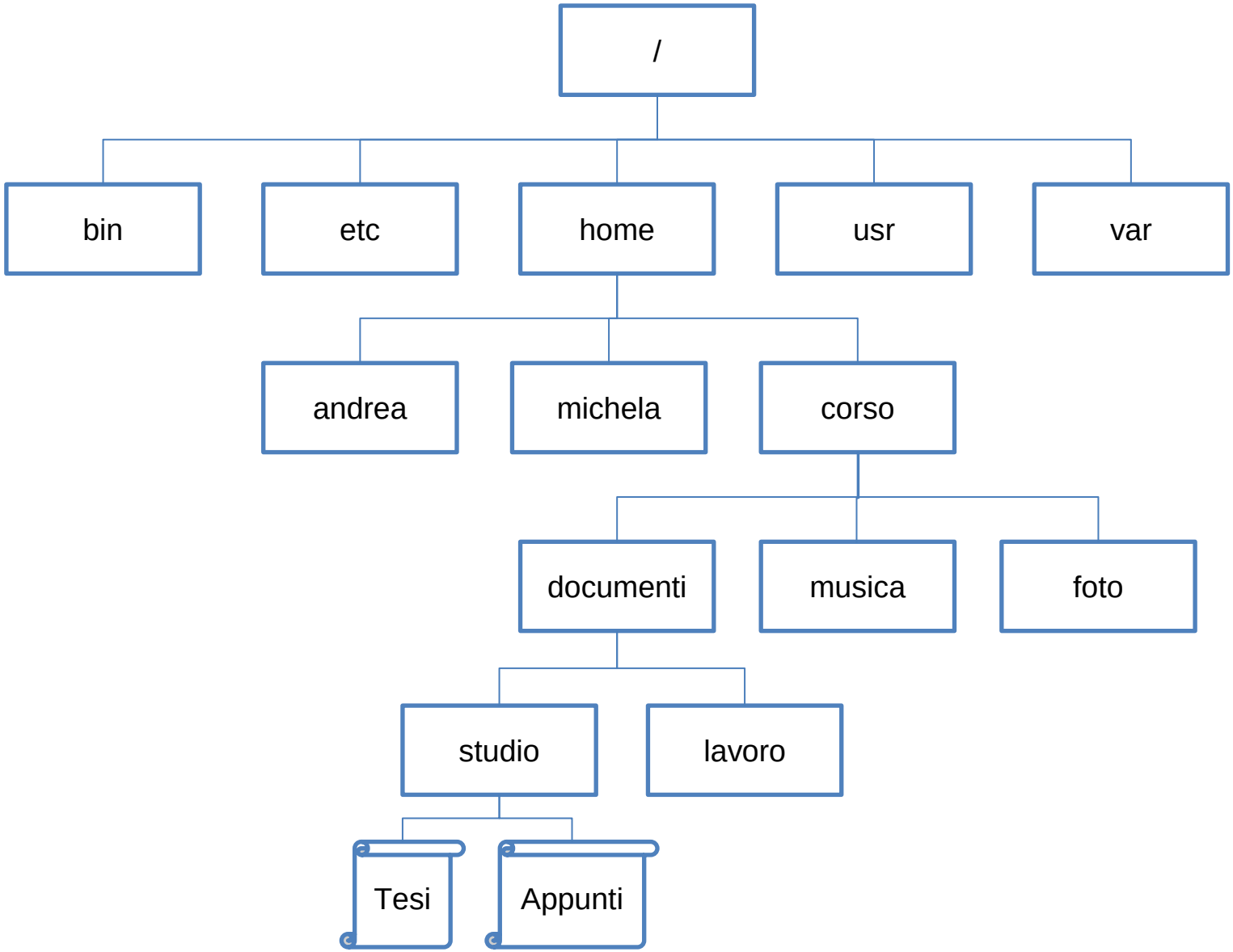

**Attenzione:** Linux è *case sensitive*. Ciò vuol dire che il comando *pwd* è diverso da *PWD*, e che */home/corso* è diverso da */home/Corso*

# Cosa ci dice il nostro primo comando?

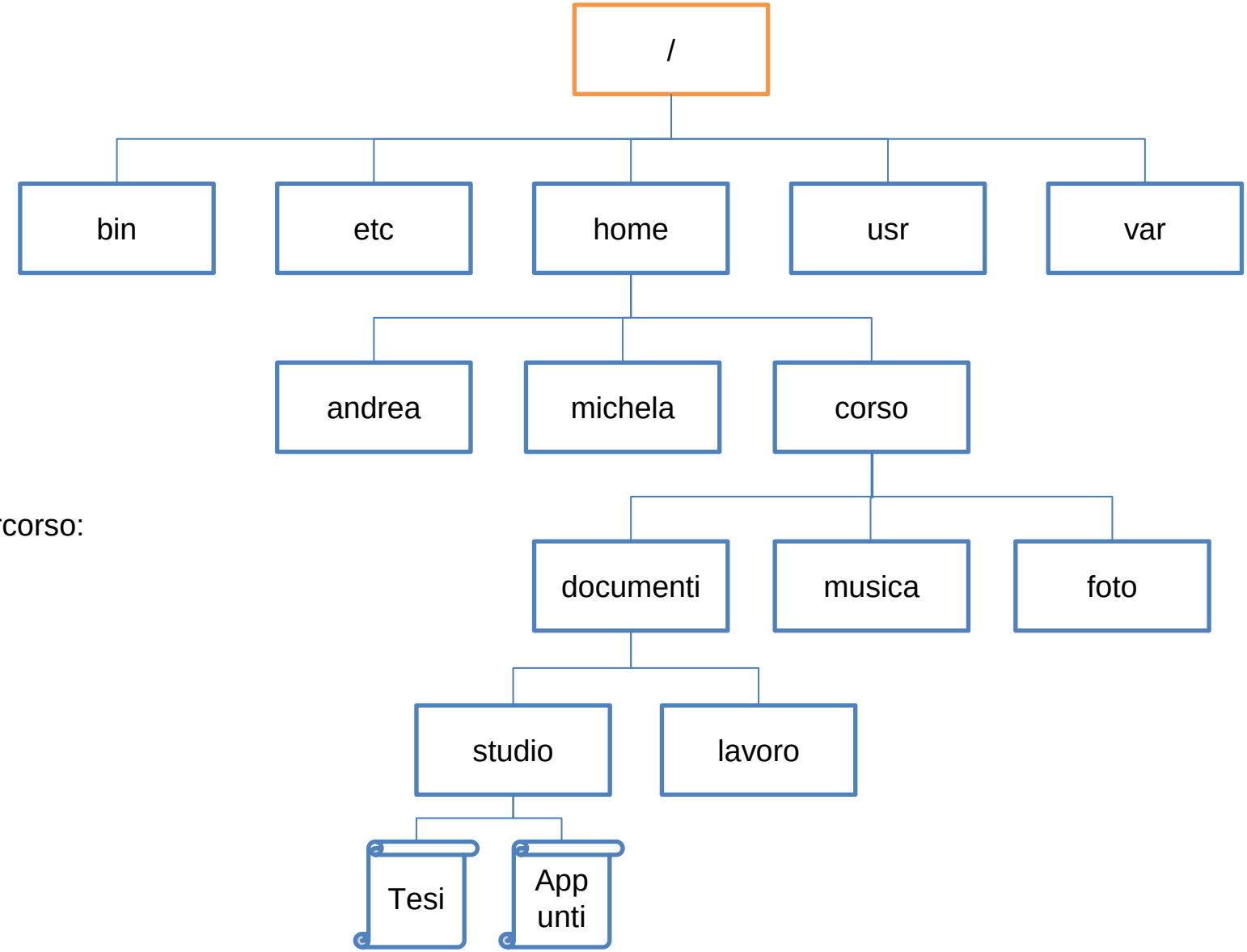
```
corso@sistemi-operativi:~$ pwd
/home/corso
corso@sistemi-operativi:~$
```

- *pwd* sta per **p**rint **w**orking **d**irectory
  - Cioè stampa la directory (cartella) in cui stiamo lavorando
- Il concetto di *working directory* è fondamentale
  - E' la directory dove ci troviamo all'interno del *filesystem*
  - E' la directory all'interno della quale effettuiamo le operazioni
    - Creazione, modifica, rimozione, … di file ed altre directory
- */home/corso* indica la directory *corso* all'interno della directory *home*, all'interno della directory principale del filesystem (*root directory*)

# Il Filesystem

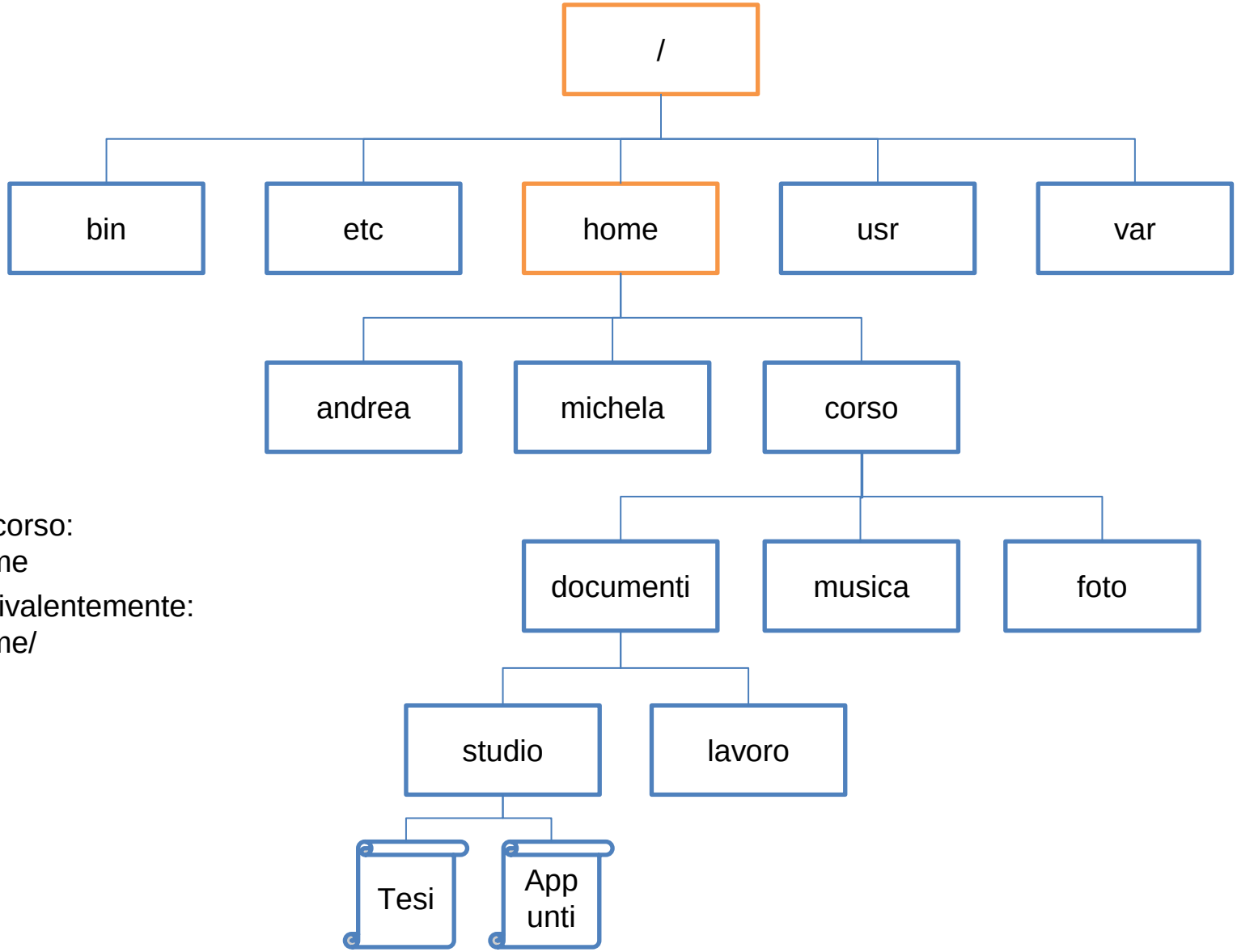


# Il Filesystem




Percorso:
/

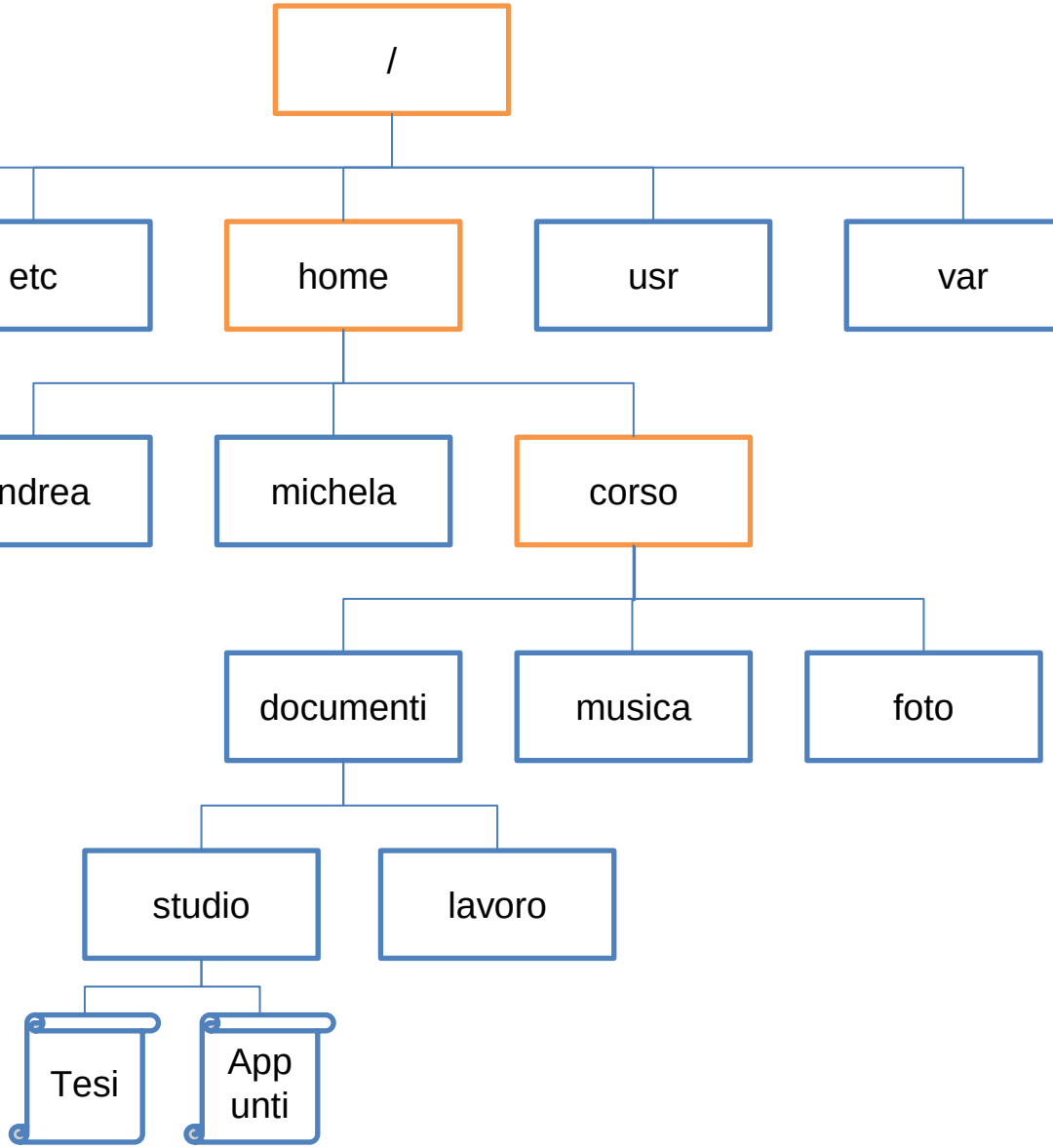
# Il Filesystem




Percorso:
/home

Equivalentemente:
/home/

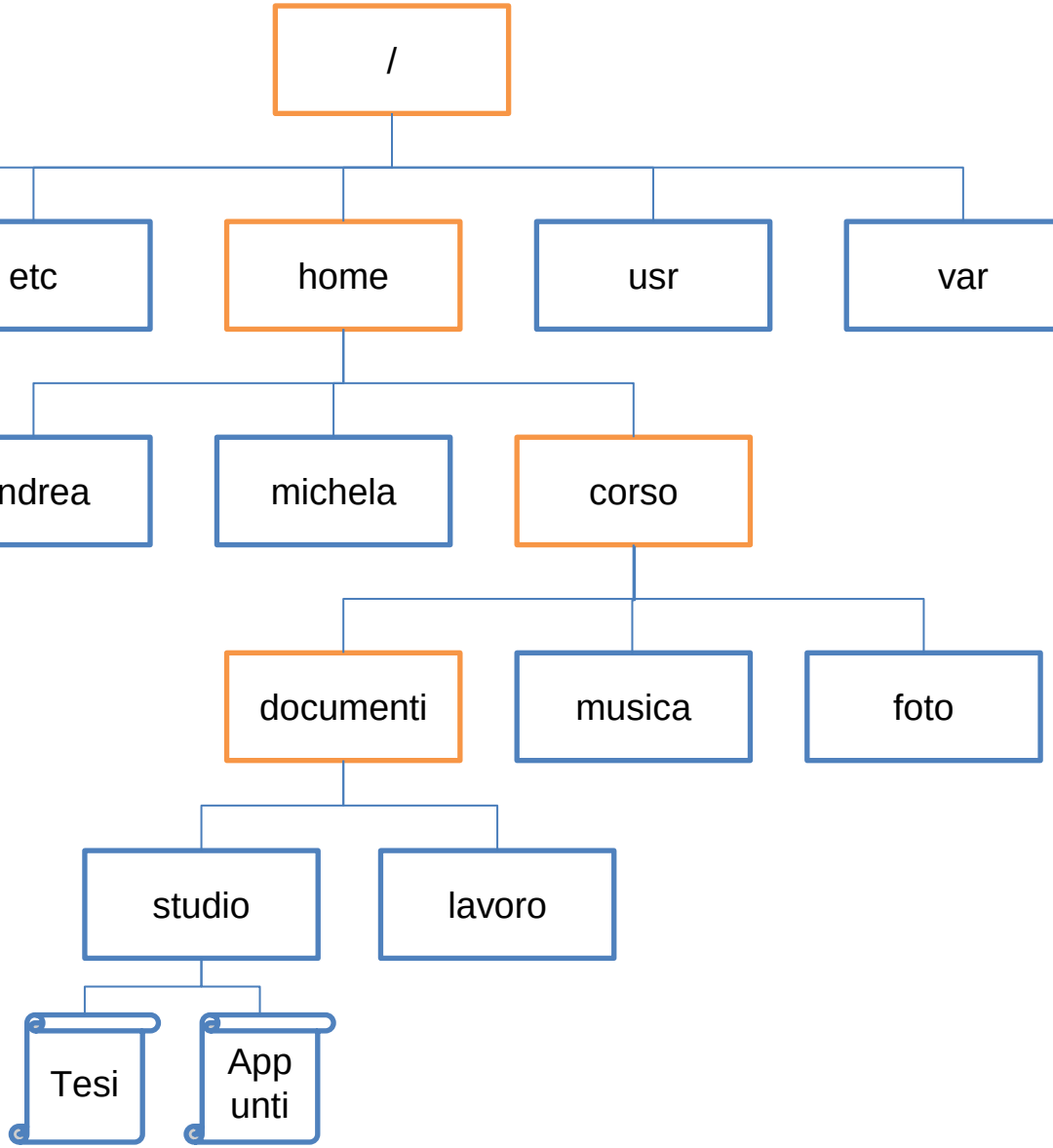
# Il Filesystem




Percorso:
/home/corso

Equivalentemente:
/home/corso/

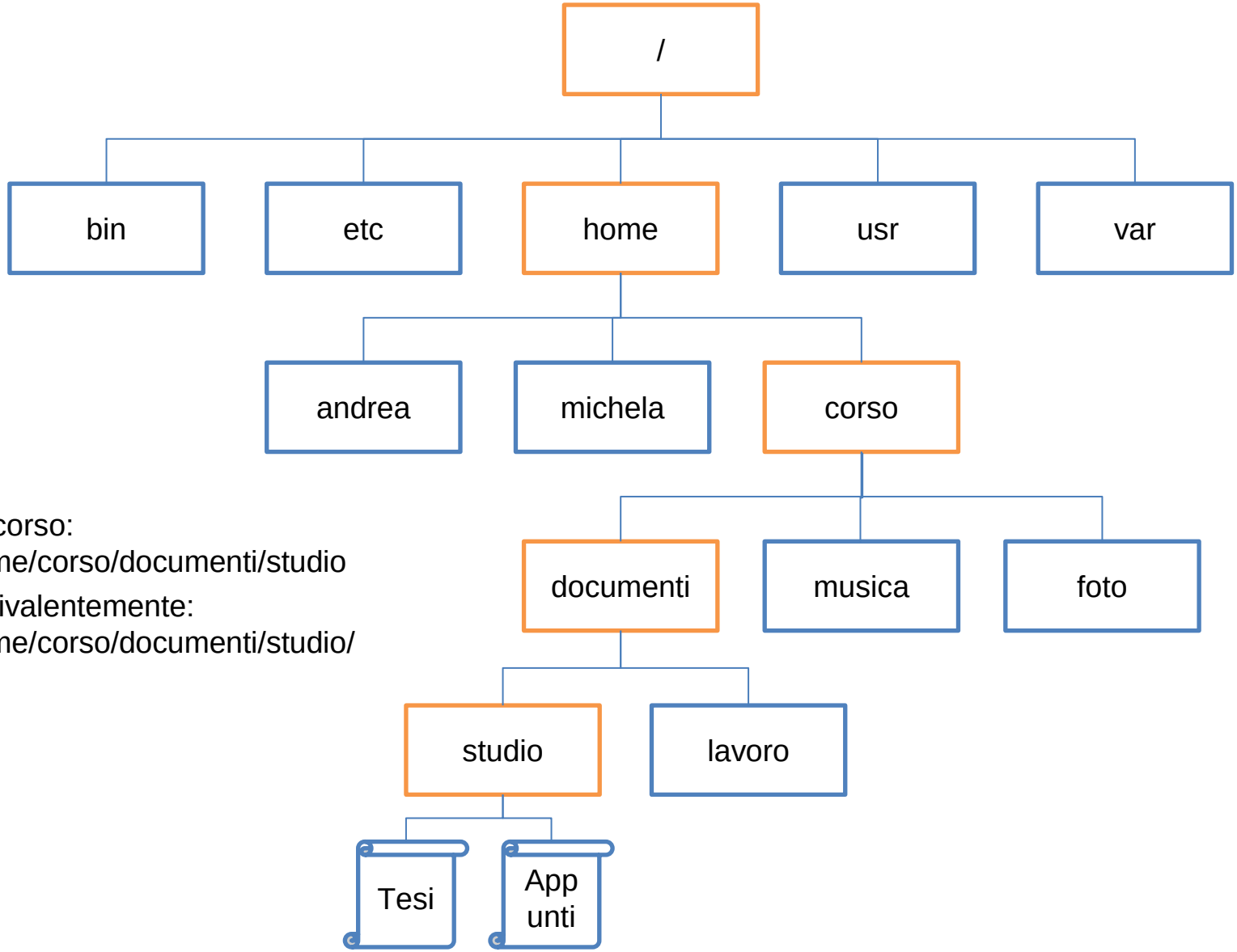
# Il Filesystem




Percorso:
/home/corso/documenti

Equivalentemente:
/home/corso/documenti/

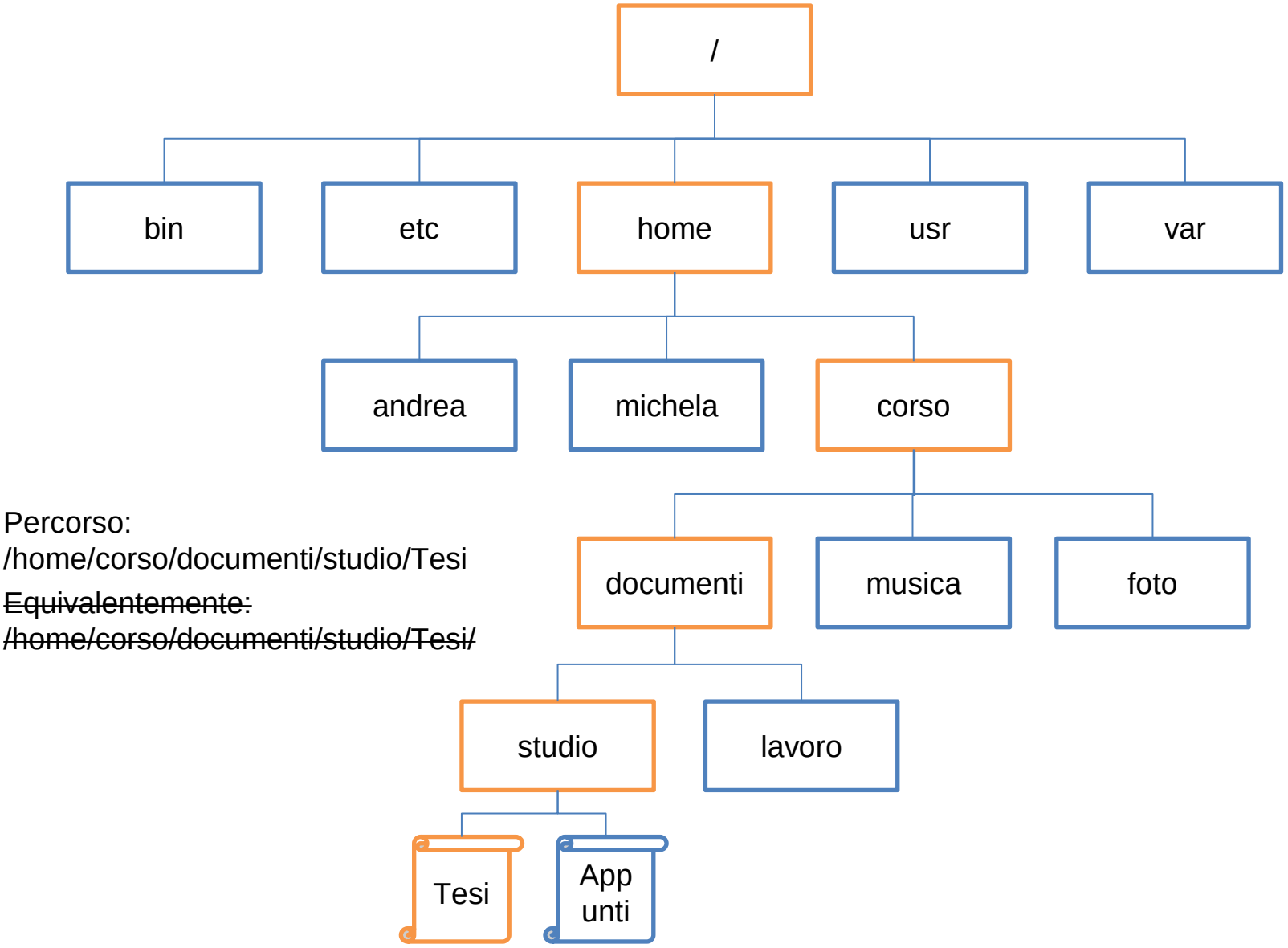
# Il Filesystem




Percorso:
/home/corso/documenti/studio

Equivalentemente:
/home/corso/documenti/studio/

# Il Filesystem




Percorso:
/home/corso/documenti/studio/Tesi

~~Equivalentemente:~~
~~/home/corso/documenti/studio/Tesi/~~

# Come cambiare la working directory?

```
corso@sistemi-operativi:~$ pwd
/home/corso
corso@sistemi-operativi:~$ cd /
corso@sistemi-operativi:/$ pwd
/
corso@sistemi-operativi:/$
```

- Il comando *cd* sta per **c**hange **d**irectory
- Il primo comando mi ha informato del fatto che mi trovo in */home/corso*
- Il secondo comando mi ha fatto spostare nella directory */*
  - Notare che **il comando *cd*, se eseguito con successo, non produce alcun output**
- Il terzo comando mi ha informato del fatto che mi trovo in */*

# Come tornare da / in */home/corso*?

- Prima di tutto devo andare in /home. Da lì poi posso andare in /home/corso
  - *cd home*
  - *cd corso*

```
corso@sistemi-operativi:/$ cd home
corso@sistemi-operativi:/home$ pwd
/home
corso@sistemi-operativi:/home$ cd corso
corso@sistemi-operativi:~$ pwd
/home/corso
corso@sistemi-operativi:~$
```

# Percorsi relativi, percorsi assoluti

- Tutti i percorsi che iniziano con / sono detti assoluti
  - Perché specificano il percorso nella sua interezza (ad esempio /home/corso)
- Percorsi che non iniziano con / sono detti relativi
  - Ne abbiamo già utilizzati due! Quali?
    - *home* e *corso*
- Quando si usa un percorso relativo, Linux lo trasforma in assoluto tramite un'operazione di concatenamento con il percorso rappresentante la working directory
  - Ad esempio, quando abbiamo eseguito *cd corso*:
    - *corso* era il percorso relativo
    - La working directory era /home/
    - */home/ + corso ➔ /home/corso*
    - Quindi è come se avessimo eseguito *cd /home/corso*

# Directory speciali

- Ogni directory contiene a sua volta due directory speciali, chiamate:
  - .
  - ..
- La directory . rappresenta la directory corrente
  - Quindi */home/corso/.* è esattamente uguale a */home/corso*
  - Ma anche */home/corso/./.* è esattamente uguale a */home/corso*
  - Sembra strano ed inutile, ma presto ci tornerà comodo
- La directory .. Rappresenta la directory superiore
  - Quindi /home/corso/.. è esattamente uguale a /home
  - /home/corso/../.. è esattamente uguale a…?
    - /
- Ma perché dovremmo complicarci la vita in questo modo?
  - Se mi trovo in /home/corso non è sufficiente scrivere *cd /home* se voglio andare in */home*?

# La directory ..

- Se mi trovo in /home/corso non è sufficiente scrivere *cd /home* se voglio andare in */home*?
  - Si, è sufficiente. Però è scomodo, perché devo ricordarmi del nome della directory superiore
  - E' molto più facile scrivere *cd ..*

```
corso@sistemi-operativi:~$ pwd
/home/corso
corso@sistemi-operativi:~$ cd ..
corso@sistemi-operativi:/home$ pwd
/home
corso@sistemi-operativi:/home$
```

# Navigazione con ..

- Come posso raggiungere */* a partire da */home/corso* senza utilizzare il simbolo / ?

```
corso@sistemi-operativi:~$ pwd
/home/corso
corso@sistemi-operativi:~$ cd ..
corso@sistemi-operativi:/home$ pwd
/home
corso@sistemi-operativi:/home$ cd ..
corso@sistemi-operativi:/$ pwd
/
corso@sistemi-operativi:/$
```

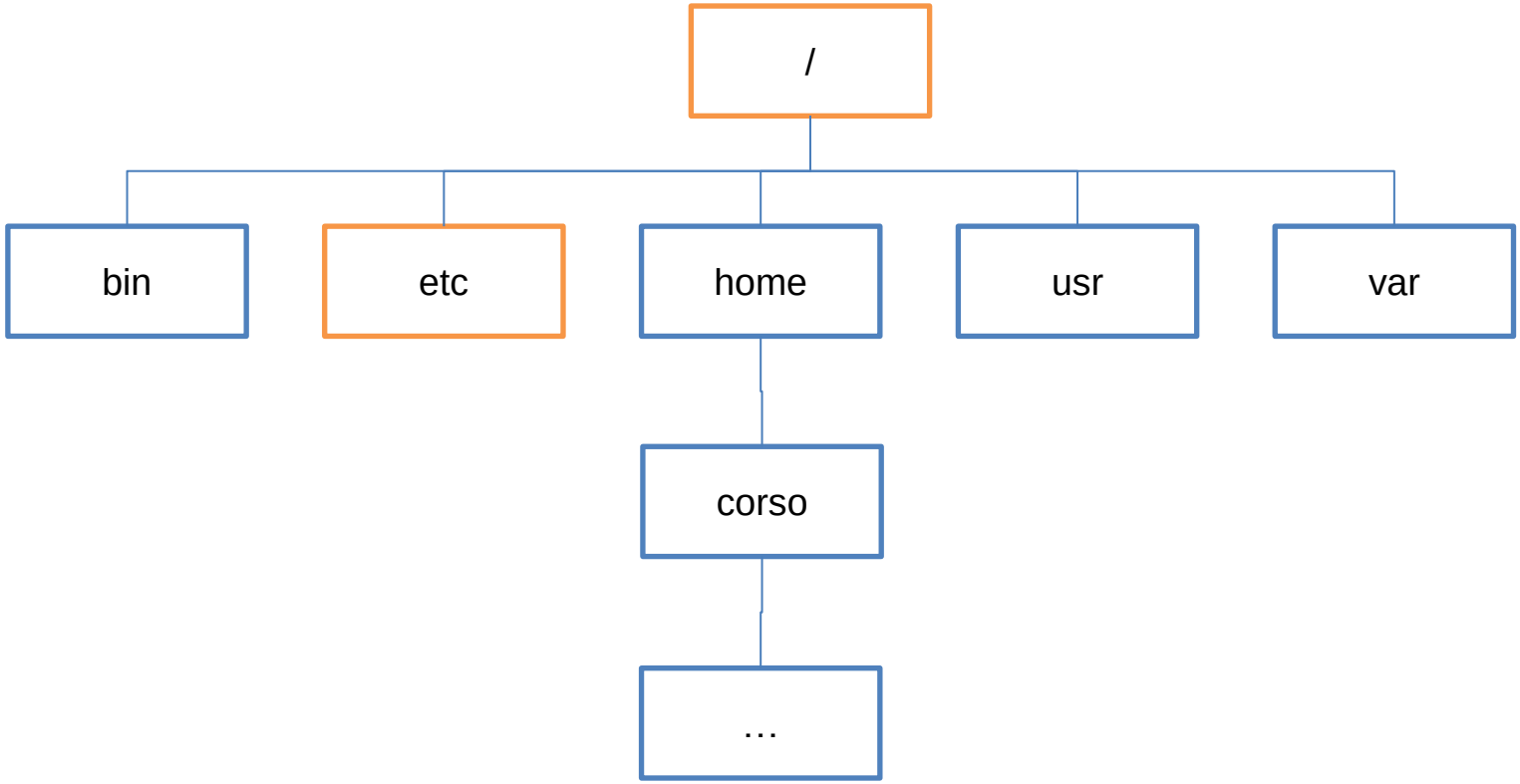
- Non è necessario utilizzare *pwd* ogni volta dopo *cd*

```
corso@sistemi-operativi:~$ cd ..
corso@sistemi-operativi:/home$ cd ..
corso@sistemi-operativi:/$ pwd
/
corso@sistemi-operativi:/$
```

=

```
corso@sistemi-operativi:~$ cd ../../
corso@sistemi-operativi:/$ pwd
/
corso@sistemi-operativi:/$
```

# Domanda




Come arrivare in */etc* partendo da */home/corso*?

***cd* con percorso assoluto**
*cd /etc*

***cd* con percorso relativo**
*cd ../../etc*

# La home directory con ~

- La home directory è una directory molto speciale all'interno della quale vengono solitamente archiviati tutti i documenti (testi, musica, video, ...) di un determinato utente.
  - Ogni utente ha la propria home directory
  - Convenzionalmente, ogni utente ha la propria home directory all'interno della directory /home, chiamata con il proprio nome utente.
- Il simbolo ~ è una scorciatoia per rappresentare la home directory
  - Quindi, ovunque io mi trovi, posso utilizzare *cd ~* per spostarmi nella mia home directory
  - Sulle tastiere italiane il simbolo ~ non c'è (bisognerebbe digitarne il codice ASCII)
  - Fortunatamente, c'è un modo ancora più veloce di tornare alla propria home directory: *cd*

# Creazione di directory

- Molto spesso ci troviamo a dover creare una directory per meglio organizzare i nostri file.
- La creazione di una directory in Linux si ottiene eseguendo il comando ***mkdir***
  - Uso: *mkdir <percorso>,* dove *<percorso>* indica la directory che si vuole creare, che può essere specificata sia con un percorso assoluto che con un percorso relativo
  - Esempio: *mkdir /home/corso/tutorial* crea una directory tutorial all'interno della directory */home/corso*
    - Avrei potuto alternativamente spostarmi all'interno della directory */home/corso* e creare la directory tutorial con un percorso relativo: *mkdir tutorial*
- *mkdir* permette anche la creazione di più directory allo stesso tempo
  - Esempio: *mkdir dir1 dir2 dir3*
- Cosa succede se provo ad eseguire *mkdir* senza parametri?

# Mostrare il contenuto di una directory

- Ovviamente non dobbiamo imparare a memoria tutto l'albero del filesystem.
  Il comando ***ls*** (sta per ***list***) serve ad elencare il contenuto di una directory.

```
corso@sistemi-operativi:~$ mkdir /home/corso/tutorial
corso@sistemi-operativi:~$ cd /home/corso/tutorial
corso@sistemi-operativi:~/tutorial$ mkdir dir1 dir2 dir3
corso@sistemi-operativi:~/tutorial$ ls
dir1  dir2  dir3
corso@sistemi-operativi:~/tutorial$
```

- Notiamo che il comando *mkdir* non ha creato una directory dentro l'altra, ma tutte e tre allo stesso livello, dentro *tutorial*.
  - E' comunque possibile creare tre directory (diciamo *dir4, dir5, dir6*) una dentro l'altra eseguendo: *mkdir –p dir4/dir5/dir6*
  - L'opzione *–p* istruisce mkdir a creare (se non esistono) tutte le directory specificate all'interno del percorso contenuto nell'argomento
  - Domanda: *dir4/dir5/dir6* è assoluto o relativo?

# Il problema degli spazi

- Mio cugino deve inviarmi dei documenti e vorrei archiviarli tutti all'interno di una directory a lui dedicata. Proviamo a creare una directory con uno "spazio" tra due parole, chiamata *mio cugino*

```
corso@sistemi-operativi:~/tutorial$ mkdir mio cugino
corso@sistemi-operativi:~/tutorial$
```

- Ha funzionato!!! Proviamo ad utilizzare *ls* per vedere effettivamente questa nuova directory.

```
corso@sistemi-operativi:~/tutorial$ ls
cugino  dir1  dir2  dir3  mio
corso@sistemi-operativi:~/tutorial$
```

- Oops... in realtà ha creato due directory, una chiamata *mio* e l'altra chiamata *cugino*. C'era da aspettarselo!