Laboratorio di Informatica di Base

Laurea in Informatica

Docente: *Carlo Drioli*

Web: `http://www.scienze.univr.it/fol/main?ent=oi&id=28279`

Laurea in Informatica Multimediale

Docente: *Barbara Oliboni*

Andrea Colombari, (colombari@sci.univr.it)

Lucidi a cura di Carlo Drioli (drioli@sci.univr.it) e Barbara Oliboni (oliboni@sci.univr.it)

Lezione 1

L'elaboratore elettronico

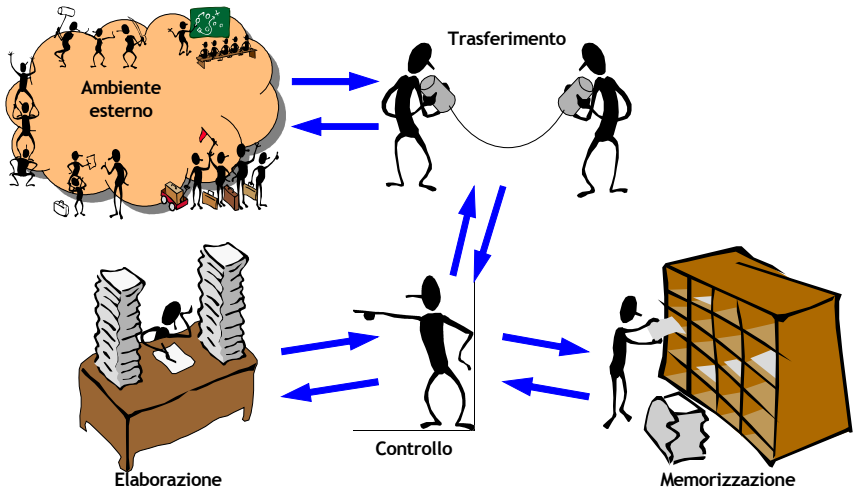
Materiale tratto dai lucidi ufficiali a corredo del testo:

D. Sciuto, G. Buonanno e L. Mari

"Introduzione ai sistemi informatici"

2005 - McGrawHill

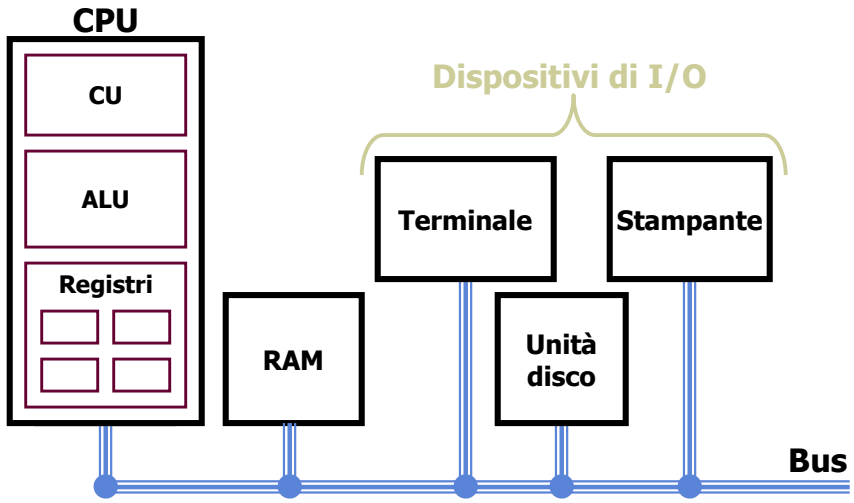
L'elaboratore elettronico



L'elaboratore elettronico (2)

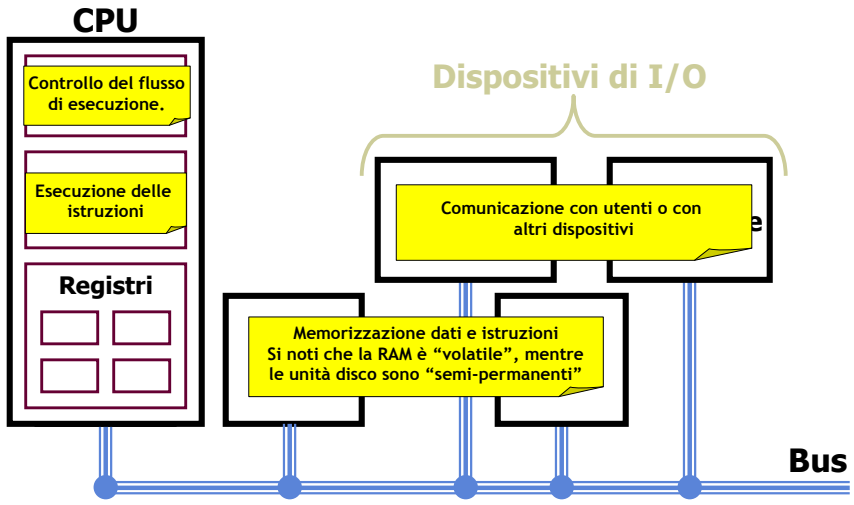
- Un *elaboratore elettronico* deve essere in grado di:
  - eseguire istruzioni su dati e controllare il flusso dell'esecuzione
    → **CPU**, Central Processing Unit:
    - Unità logico aritmetica (**ALU**, Arithmetic Logic Unit)
    - Unità di controllo (**CU**, Control Unit)
    - **Registri**
  - memorizzare le istruzioni e i dati su cui esse operano
    → memoria centrale (**RAM**, Random Access Memory) e **unità disco** (o memoria di massa)
  - interagire con gli utenti e con eventuali altri sistemi
    → dispositivi di ingresso/uscita, i.e. input/output (**I/O**)

L'architettura di Von Neumann

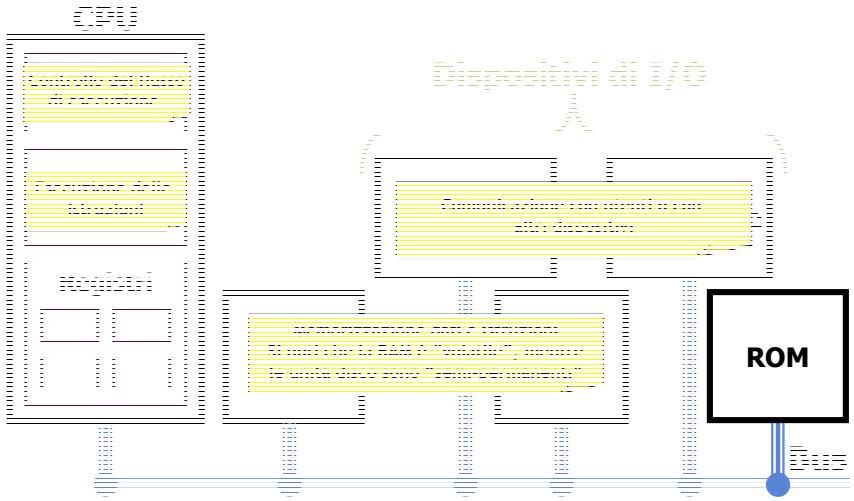


Introduzione ai sistemi informatici 3/ed
D. Sciuto, G. Buonanno e L. Mari
Copyright © The McGraw-Hill Companies srl

L'architettura di Von Neumann

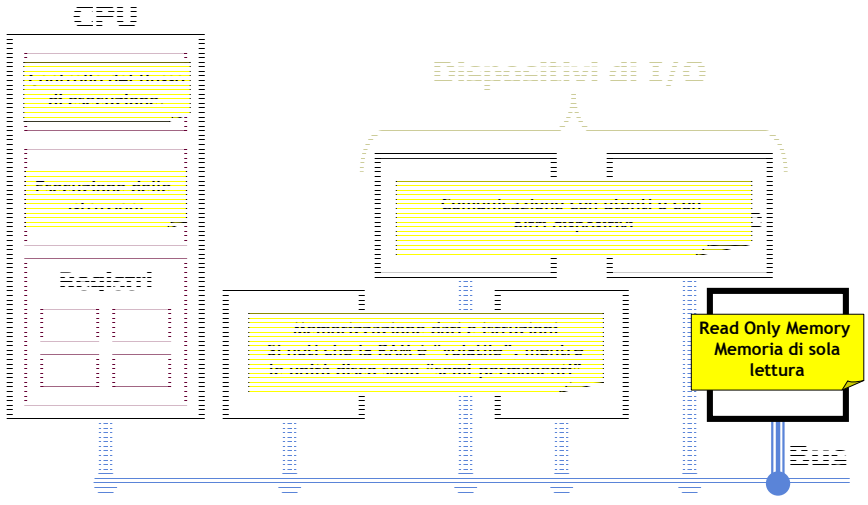


L'architettura di Von Neumann



Introduzione ai sistemi informatici 3/ed
D. Sciuto, G. Buonanno e L. Mari
Copyright © The McGraw-Hill Companies srl

L'architettura di Von Neumann



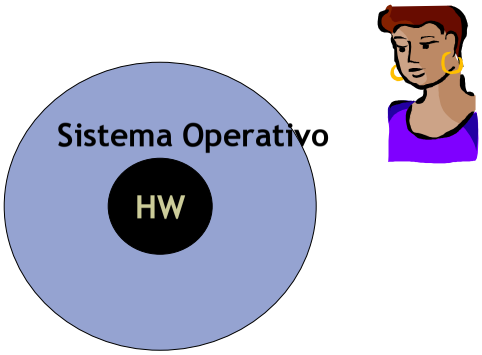
Il bootstrap

- La fase di accensione di calcolatore è detta *bootstrap*
- Un calcolatore è fatto per eseguire delle istruzioni che si trovano in memoria centrale
  → PROBLEMA:
  All'accensione la RAM è vuota!
  → SOLUZIONE:
  Utilizzo di una ROM in cui è "fuso" un programma di bootstrap che viene eseguito all'accensione
- Il programma di bootstrap ha il compito di caricare in memoria centrale il **Sistema Operativo (S.O.)** a partire da un'unità disco

Il Sistema Operativo

Materiale tratto dai lucidi ufficiali a corredo del testo:
D. Sciuto, G. Buonanno e L. Mari
"Introduzione ai sistemi informatici"
2005 - McGrawHill

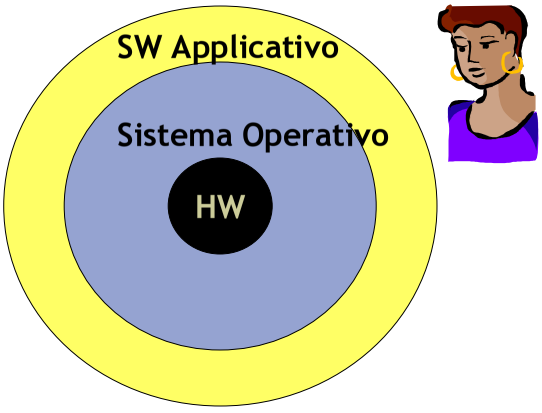
Il sistema operativo



- Il S.O. come necessario **intermediario**

Introduzione ai sistemi informatici 3/ed
D. Sciuto, G. Buonanno e L. Mari
Copyright © The McGraw-Hill Companies srl

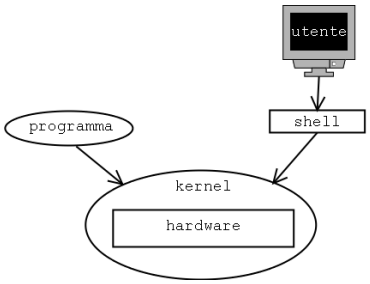
Il sistema operativo



- Il S.O. come necessario **intermediario**
- SW = Sistema Operativo + SW applicativo

Il sistema operativo

- Il sistema operativo fornisce dei servizi ai programmi applicativi e agli utenti rendendo utilizzabili le risorse fisiche presenti nel sistema di calcolo.
- Il sistema operativo può essere inteso come uno strumento che virtualizza le caratteristiche dell'hardware sottostante, offrendo di esso la visione di una macchina astratta più potente e più semplice da utilizzare di quella fisicamente disponibile.



Il sistema operativo: funzioni

- Il S.O. deve svolgere delle funzioni di base:
  - Esecuzione di applicazioni
  - Accesso ai dispositivi di ingresso/uscita
  - Archiviazione di dati e programmi
  - Controllo di accesso
  - Gestione di tempi e costi di utilizzo (contabilizzazione)
  - Gestione dei malfunzionamenti

Il sistema operativo: elementi

- Un S.O. tipico è costituito dai seguenti elementi:
  - Sistema di gestione del processore
  - Sistema di gestione della memoria
  - Sistema di gestione delle periferiche (I/O)
  - Sistema di gestione dei file (file system)
  - Sistema di gestione degli utenti e dei relativi comandi (interprete comandi o shell)
  - Sistema di gestione della rete.

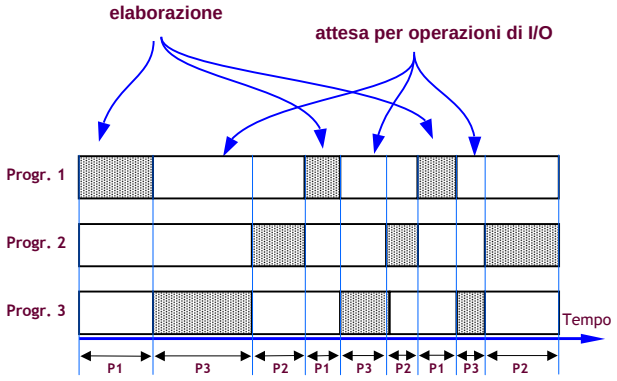
Introduzione ai sistemi informatici 3/ed
D. Sciuto, G. Buonanno e L. Mari
Copyright © The McGraw-Hill Companies srl

Il sistema operativo: tipologie

- L'evoluzione dei sistemi di calcolo ha visto succedersi varie tipologie di S.O.:
  - Sistemi batch (esecuzione non interattiva di task)
  - Sistemi interattivi (interazione con l'utente)
  - Sistemi uniprogrammati e monoutente (1 solo l'utente, 1 solo programma utente per volta in esecuzione)
  - Sistemi time-sharing: interattivi, multiprogrammati e multiutente (più utenti e più programmi utente in esecuzione)

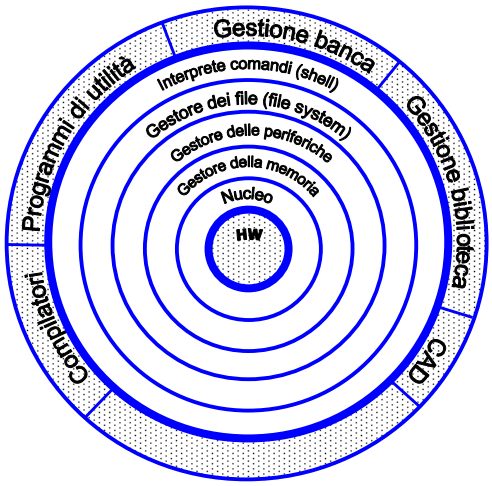
Il sistema operativo: tipologie (2)

Time-sharing



Il sistema operativo: organizzazione

- Un classico modello di organizzazione del S.O. è quello a “strati”, detto anche a “buccia di cipolla”:
  - Determina una gerarchia di macchine virtuali
  - La m.v. al livello N è determinata dall’hardware e dagli strati del S.O. fino a quel livello



Il nucleo (kernel)

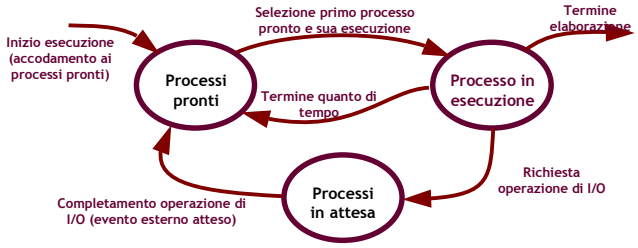
- Per la gestione dei processi il S.O. deve:
  - Interagisce direttamente con l'hardware
  - Occuparsi dell'esecuzione dei programmi e della risposta agli eventi esterni generati dalle unità periferiche di I/O.
  - Gestire i processi corrispondenti ai programmi che sono contemporaneamente attivi.
  - Gestire il contesto di esecuzione dei vari processi
  - Attuare una politica di alternanza (*scheduling*) nell'accesso alla CPU da parte dei processi in esecuzione.
  - Essere eseguito sempre in modalità privilegiata (detta kernel mode)

La gestione dei processi

- Un programma è un insieme di istruzioni da eseguire (entità statica)
- Un processo è un'istanza di un programma in esecuzione (entità dinamica)
- Un processo è costituito dal programma e dal contesto di esecuzione
  (program counter, registri, memoria, etc.)
- In un s.o. multiprogrammato possono esistere più processi in esecuzione
- I processi possono essere eseguiti in kernel mode (ad es. i servizi forniti dal sistema operativo) o user mode (ad es. i programmi applicativi)

La gestione dei processi (2)

- Compiti del s.o. per la gestione dei processi
  - Creazione/terminazione dei processi
  - Sospensione/ripristino dei processi
  - Sincronizzazione/comunicazione tra processi
  - Gestione di situazioni di stallo (blocco critico o deadlock)
  - Esempio di diagramma di stato dei processi



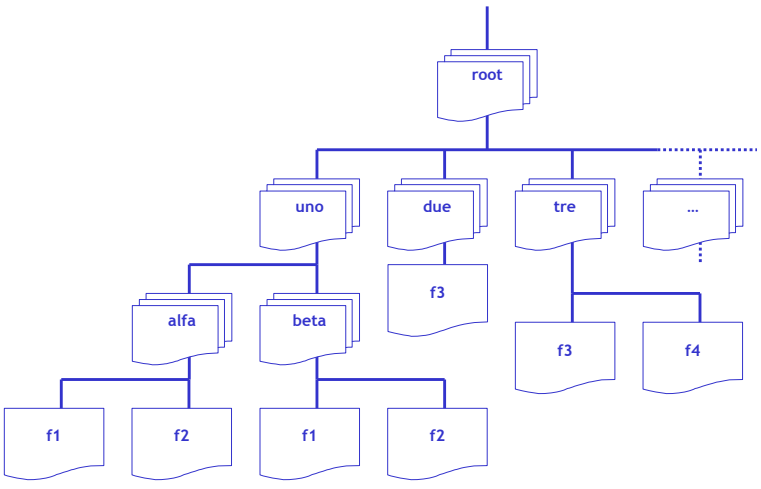
La gestione della memoria

- Il gestore della memoria centrale
  - Risolve le relative esigenze dei vari processi in esecuzione
  - Consente ai programmi di lavorare in un proprio spazio di indirizzamento virtuale e di ignorare quindi le effettive zone di memoria fisica occupata
  - Protegge programmi e relativi dati caricati nella memoria di lavoro
  - Fornisce alle macchine virtuali di livello superiore la possibilità di lavorare come se esse avessero a disposizione una memoria dedicata, di capacità anche maggiore di quella fisicamente disponibile.

Il file system

- Permette di organizzare i dati contenuti nella memoria di massa in forma strutturata
- I dati sono raccolti in strutture logiche dette files, identificate da un nome (filename)
- I files sono organizzati in più contenitori logici (cartelle o directory) secondo una struttura ad albero
- Ciascun file è individuato mediante il suo percorso (path) assoluto, ottenuto giustapponendo i nomi dei nodi che si incontrano dalla radice al file (separati da un carattere apposito, in linux **/**)
  Esempio: **/**uno**/**alfa**/**f1

Il file system (2)



La gestione del file system

- Il gestore del file system fornisce i servizi di base per
  - La creazione/cancellazione di file e cartelle
  - La manipolazione di file e cartelle esistenti
  - La copia di dati su supporti diversi
  - L'associazione tra file e dispositivi di memorizzazione secondaria (memorie di massa)
  - La gestione di collegamenti (link o alias) tra file e cartelle. Un collegamento è un riferimento a un oggetto (file o cartella) presente nel file system.

La gestione delle periferiche

- Il gestore delle periferiche
  - Fornisce una visione del sistema in cui i processi possono operare mediante periferiche astratte
  - Maschera le caratteristiche fisiche delle periferiche e le specifiche operazioni di ingresso/uscita
  - Mette a disposizione di ogni processo delle periferiche virtuali
  - In alcuni S.O. come linux, il gestore fa in modo che file, cartelle e periferiche I/O vengano presentati all'utente in modo uniforme

L'interprete di comandi

- E' un modulo del S.O. direttamente accessibile dall'utente
- Ha la funzione di interpretare i comandi che gli giungono e di attivare i programmi corrispondenti
- Svolge operazioni di lettura della memoria centrale dei programmi da eseguire
- Alloca la memoria necessaria e vi carica programmi e dati
- Crea e attiva il processo relativo al programma

Introduzione a Linux

Testo di riferimento:

M. Bertacca, e A. Guidi
"Introduzione a Linux"
McGrawHill

Il S.O. GNU/Linux: introduzione

- GNU/Linux è un sistema operativo libero di tipo Unix, distribuito con licenza GNU GPL (General Public License)
- Il kernel Linux nasce dalla collaborazione a distanza, grazie a Internet, di numerosi sviluppatori volontari
- E' un sistema **multiutente** e **multiprocesso** (**multitasking**)
- Adotta un proprio file system per la memorizzazione dei file (ext2fs) ma e' compatibile con i principali file system in uso (es. MS-DOS, Fat32)
- E' dotato di potenti interfacce a caratteri (**shell** di comando) ma anche di ambienti desktop evoluti come **KDE** e **GNOME**

Il SO Linux: accesso alla macchina

- Per accedere ad una sessione Linux è necessario disporre di un nome utente riconosciuto dal sistema (login) e di una parola d'ordine (password)

```
login: mialogin
password: *******
```

- Dopo l'autenticazione, l'utente accede alla propria cartella di lavoro (home directory, ad es. /home/mialogin) e può interagire con il S.O. con un'interfaccia, ad es. un interprete di comandi (shell)
- Il prompt ($) avvisa l'utente che l'interprete è pronto ad accettare comandi
- Per terminare la sessione di lavoro si possono usare i comandi logout o exit. Il sistema tornerà nello stato di richiesta di login e password

```
$ logout
    oppure
$ exit
```

Il SO Linux: accesso alla macchina (2)

- La shell legge i comandi dell'utente, li interpreta e richiede al sistema l'esecuzione delle operazioni richieste dal comando.

  Esempio: il comando passwd permette di impostare una nuova password:

```
$ passwd
Enter the new password (minimum of 5, maximum of
8 characters)
New Password:
```

Il SO Linux: comandi principali

- In generale la sintassi di un comando Linux è:

```
comando [opzioni] [argomenti]
```

- Un manuale dei comandi descrive l'utilizzo e le caratteristiche di ogni comando. Le pagine del manuale si invocano con man argomento e hanno tutte la seguente struttura comune:

NAME: riporta il nome del comando e una breve descrizione delle sue funzioni
SYNOPSIS: descrive la sintassi del comando
DESCRIPTION: descrive lo scopo e il funzionamento del comando
OPTIONS: riporta il funzionamento di tutte le opzioni
ENVIRONMENT: descrive eventuali variabili d'ambiente che interagiscono con il comando
AUTHOR: note sull'autore del comando
COPYRIGHT: note su copyright
BUGS: eventuali errori o malfunzionamenti noti
SEE ALSO: eventuali altre pagine del manuale a cui fare riferimento

Il SO Linux: comandi principali (2)

- Elencare il contenuto di una cartella
- Sintassi:

```
ls [opzioni…] [cartella…]
```

  - Opzioni:
    - -l (informazioni estese)
    - -a (visualizza file nascosti, cioè inizianti con il **.**)
    - -R (visualizza sottocartelle)
- Esempio:

```
$ ls -laR
```

Il SO Linux: comandi principali (3)

- Cambiare la cartella corrente
- Sintassi:

```
cd path_nuova_directory
```

  - Opzioni:
    - cartella corrente: .
    - cartella padre: ..
    - home directory: ~
- Esempio:

```
$ cd ..

$ cd ./home/mialogin/miacartella (se esiste)
```

Il SO Linux: comandi principali (5)

- Creare nuove cartelle
- Sintassi:

```
mkdir nome_cartella
```

- Esempio:

```
$ mkdir nuovacartella1 nuovacartella2
```

Il SO Linux: comandi principali (6)

- Copiare file e cartelle

```
cp [opzioni…] sorgente… destinazione
```

- Spostare o rinominare file e cartelle

```
mv [opzioni…] sorgente… destinazione
```

- Visualizzare path assoluto cartella corrente

```
pwd
```

- Eliminare file

```
rm [opzioni…] file
```

- Eliminare una cartella

```
rmdir cartella
```

Caratteri jolly o *metacaratteri*

***** Sostituisce un insieme di zero o più caratteri qualsiasi

Esempio:

```
$ ls c*
```

? Sostituisce un carattere qualsiasi

Esempio:

```
$ ls co??o.txt
```

[ ] Permettono di specificare una lista e/o un intervallo di caratteri possibili

Esempio:

```
$ ls [a-c]*.txt
```

Il SO Linux: i processi

- Linux è un sistema operativo multitasking: può eseguire contemporaneamente più programmi
- Un programma in esecuzione è definito processo
- Ad ogni processo viene assegnato un identificativo univoco: PID
- Un processo può essere attivo o sospeso ed eseguito in foreground (fg) o in background (bg)
- Un programma può essere eseguito in bg usando il carattere **&** (`$ ls c* &`) e può essere sospeso con la combinazione di tasti CTRL+Z
- Il sistema operativo fornisce comandi per visualizzare informazioni sui processi e per gestirne l'esecuzione.

Comandi per operare sui processi

- Visualizzare informazioni sui processi

```
ps [opzioni…] [PID]
```

- Eliminare un processo

```
kill [opzioni…] PID
```

- Visualizzare i processi sospesi o in background

```
jobs
```

- Riprendere l'esecuzione in foreground di processi sospesi o in background

```
fg [PID]
```

Comandi per operare sui processi (2)

- Attivare l'esecuzione in background di processi sospesi

```
bg [PID]
```

- Monitorare l'utilizzo delle risorse da parte dei processi

```
top [opzioni]
```