

## Architettura generale del calcolatore

- hardware
  - la macchina di Von Neumann
  - le periferiche
- software
  - il sistema operativo
  - i programmi applicativi

---

---

---

---

---

---

---

---

## La macchina di Von Neumann

- architettura di base dei moderni calcolatori
- risale agli anni '40
- descrive le componenti fondamentali del calcolatore ed il loro ruolo
- *dati e programmi sono ospitati nella stessa memoria (principale)*
- *un processore esegue programmi che elaborano dati*
- *dispositivi di input/output*

---

---

---

---

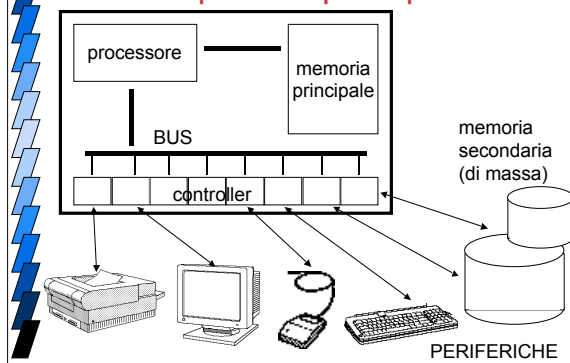
---

---

---

---

## Componenti principali




---

---

---

---

---

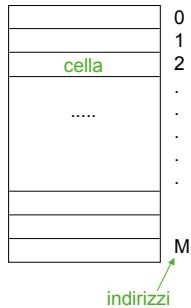
---

---

---

## La memoria principale (centrale)

- permette di "ricordare"
- astrattamente:
  - è una sequenza di componenti elementari che memorizzano una unità di informazione (bit)
  - detta RAM: Random Access Memory
  - le unità elementari sono organizzate in celle (usualmente da un byte)
  - ogni cella ha un suo indirizzo numerico che ne denota la posizione nella sequenza
  - tramite l'indirizzo possiamo accedere direttamente al contenuto di una cella sempre nello stesso tempo




---

---

---

---

---

---

---

---

---

---

## La memoria principale

- l'indirizzo è un numero intero
- può essere rappresentato col sistema binario
- il numero di bit necessari si chiama **spazio di indirizzamento**
  - es 32 ->  $2^{32}=4\text{GB}$  di memoria
- si può accedere non solo per byte ma anche per parola (word = 2-4 byte) in blocco unico
- caratteristiche della memoria principale:
  - dimensioni (8-128 Mbyte)
  - tempo di accesso (30-80 ns)
- la memoria principale è volatile!

---

---

---

---

---

---

---

---

---

---

## Il processore

- è la componente che realizza la fase di elaborazione nel calcolatore
- equivalente all'impiegato nella metafora dell'ufficio, quindi:
  - esegue una sequenza di istruzioni elementari detta **programma**
  - la sequenza è descritta con un metodo detto **linguaggio macchina**, specifico di ogni processore
  - secondo il modello di Von Neumann, il programma risiede nella memoria principale assieme ai dati su cui opera
  - tipi di istruzioni: lettura/scrittura da/in memoria, istruzioni aritmetiche, istruzioni logiche, spostamento di informazioni, istruzioni di salto

---

---

---

---

---

---

---

---

---

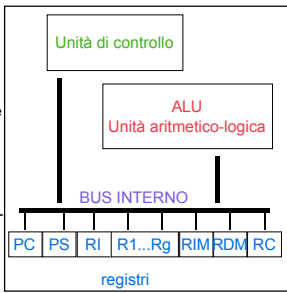
---

Marino Miculan, Università di Udine - 2002 - 87

## Componenti del processore

- **unità di controllo:** coordina le attività del processore
- **registri:** piccole unità di memoria veloci (speciali, generali)
  - PC= program counter (indirizzo della prossima istruzione)
  - RI=registro istruzioni (istruzione corrente)
  - PS=registro di stato (stato di esecuzione, errori)
  - R1...: memoria temporanea
- **ALU:** operazioni aritmetico-logiche tra registri
- **bus:** comunicazione tra le diverse entità

**PROCESSORE**



---

---

---

---

---

---

---

---

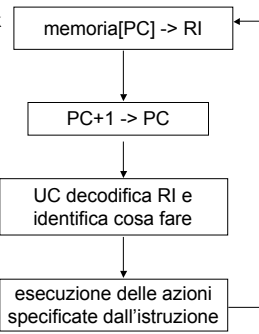
Marino Miculan, Università di Udine - 2002 - 88

## Funzionamento del processore

La frequenza di esecuzione del ciclo è scandita dal clock (16-2000 MHz)

**Azioni:**

- lettura: attivazione lettura da memoria (RIM, RDM, RC)
- scrittura: attivazione scrittura in memoria (RIM, RDM, RC)
- operazioni aritmetico-logiche: intervento della ALU
- salto: modifica di PC con l'indirizzo cui si deve andare



---

---

---

---

---

---

---

---

Marino Miculan, Università di Udine - 2002 - 89

## Tipi di processore

- **CISC: Complete Instruction Set Computers**
  - tante istruzioni anche complesse
  - linguaggio di uso semplice
  - più tempo per decodificare ed eseguire un'istruzione (più cicli di clock)
  - Motorola 680x0, Intel 80x86, Pentium
- **RISC: Reduced Instruction Set Computers**
  - poche istruzioni semplici
  - di difficile uso "diretto"
  - un ciclo di clock per decodificare ed eseguire
  - IBM/Motorola PowerPC, ARM, SPARC, MIPS, ALPHA

---

---

---

---

---

---

---

---

## Memoria secondaria / di massa

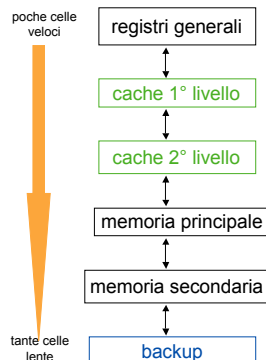
- alta capacità (50-100 volte la primaria)
- basso costo
- è più lenta della memoria principale (~10-30 ms)
- è permanente
- viene utilizzata per memorizzare permanentemente dati e programmi,
  - che vengono caricati (ovvero letti e ricopiati) nella memoria principale prima di essere utilizzati
- non sempre è permesso l'accesso diretto
- di solito è organizzata in blocchi  $\geq 1\text{KB}$  (per diminuire lo spazio di indirizzamento)
- non è usata direttamente dal processore

## Memoria secondaria: tecnologie

- dovendo essere permanente non può basarsi su tecnologie di tipo elettrico;
- magnetismo:
  - sostanze magnetizzabili
  - che assumono due possibili stati (polarizzazione positiva e negativa)
  - utilizzabili per rappresentare un bit
- ottica:
  - raggio laser (luce coerente, fascio ridottissimo)
  - superfici con piccolissimi forellini (in cui passa/non passa la luce, non viene/viene riflessa: due stati...)
  - scrittura difficile
- quindi: dischi magnetici, dischi ottici, nastri

## Gerarchie di memoria

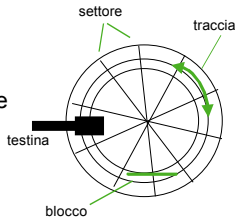
- Memoria CACHE
- è una memoria temporanea estremamente veloce,
- utilizzata per mantenere dati di uso frequente
  - che non stanno nei registri
  - senza accedere in continuazione alla memoria principale (più lenta)
- può essere organizzata in più livelli
- è molto costosa (per questo esiste...)



Marino Miculan, Università di Udine - 2002 - 93

## Dischi magnetici

- in plastica/vinile, ricoperti di materiale magnetizzabile
- durante lettura/scrittura, i dischi ruotano;
- le informazioni vengono lette/scritte da **testine** usuali,
- in un formato a **tracce** concentriche e **settori** (che intersecandosi danno i **blocchi**), impostato con l'operazione di **formattazione**




---

---

---

---

---

---

---

---

Marino Miculan, Università di Udine - 2002 - 94

## Dischi magnetici: caratteristiche

- Tempo di accesso alle informazioni: seek time + latency time + tempo di lettura
- hard disk:
  - supporto fisso, sigillati -> grande precisione,
  - e quindi alta capacità (100 MB-8 GB).
  - Velocità di rotazione (lineare costante): 3000-5000 giri/min
- floppy disk:
  - supporto rimovibile,
  - 1.5MB,
  - 300 giri/min

---

---

---

---

---

---

---

---

Marino Miculan, Università di Udine - 2002 - 95

## Dischi ottici

- tendenzialmente per sola lettura o WORM (Write Once, Read Many)
- quindi di uso diverso
- densità di memorizzazione superiore ai dischi magnetici
- capacità: tipicamente 650 MB
- memorizzazione dei dati **a spirale**
  - accesso intrinsecamente sequenziale
  - reso diretto con tabelle

---

---

---

---

---

---

---

---

## Nastri magnetici

- nastri flessibili coperti da sostanza magnetizzabile
- informazione memorizzata longitudinalmente
- accesso sequenziale ai dati
- costo molto basso
- alta affidabilità
- alta capacità
- molto lenti (sequenziale -> anche minuti)
- utilizzati per salvataggio di copie (**backup**)

---

---

---

---

---

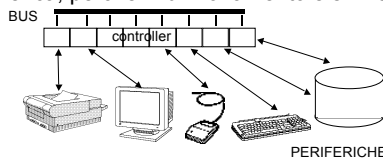
---

---

---

## Le periferiche

- dette anche **dispositivi di input/output**
- servono per "usare" il computer: in generale permettono l'interazione tra uomo e macchina
- ce ne sono tantissimi tipi, tutti funzionalmente analoghi
- anche la memoria secondaria viene considerata come periferica, perché il funzionamento è simile




---

---

---

---

---

---

---

---

## Caratteristiche comuni

- La funzione primaria è l'interazione dell'unità centrale con l'esterno (uomo/altra macchina):
  - **input**: immissione di dati
  - **output**: emissione di risultati
- ridotta autonomia: attività richiesta, gestita, controllata e coordinata dall'unità centrale (**master/slave**)
- una categorizzazione funzionale:
  - dispositivi **stupidi**: non sono in grado di elaborare i dati che trattano (es. tastiera, video)
  - dispositivi **intelligenti**: elaborano i dati trattati, in quanto dotati di proprio processore specializzato con memoria (es. stampanti postscript)
- le periferiche comunicano con l'unità centrale grazie ad un **controller** connesso al **bus**

---

---

---

---

---

---

---

---

## Modalità di funzionamento

- in generale le periferiche funzionano in modo **asincrono** rispetto all'unità centrale: non è possibile prevedere quando un determinato dato di input verrà fornito, o quando finisce una certa operazione di output
- è necessario un metodo di **sincronizzazione**:
  - la periferica **avvisa** il processore quando un dato è pronto o un'operazione è terminata
  - tramite un segnale hardware (detto **interrupt**) che il processore controlla ad ogni ciclo di clock prima di iniziare ad eseguire ogni istruzione, sospendendo l'attività in corso per gestire il dispositivo se necessario
  - le periferiche possono manipolare sia singoli byte (es. caratteri) sia blocchi di dati

---

---

---

---

---

---

---

---

## La tastiera

- principale dispositivo di input nei computer moderni
- periferica di input stupida a caratteri
- ha gli stessi tasti di una macchina da scrivere, più:
  - **tasti speciali**: es. break (interrompe l'esecuzione di un programma), PrintScreen (stampa il contenuto dello schermo), Help (attiva il sistema di aiuto all'utente, quando c'è)
  - **frecce direzionali** (N, S, E, O, più eventuali tasti aggiuntivi per lo spostamento a inizio/ fine pagina)
  - **tasti funzione**, che possono essere associati a determinate funzionalità dei programmi applicativi
  - **tastierino numerico** separato

---

---

---

---

---

---

---

---

## Il video

- principale dispositivo di output (temporaneo...) nei computer moderni
- l'immagine si forma accendendo o spegnendo (o illuminando con un certo valore di intensità) i punti (detti pixel) che costituiscono lo schermo (costituito da un tubo catodico, o da cristalli liquidi e derivati)
- caratteristiche salienti:
  - **dimensione** (in pollici, misura la diagonale, es. 15"; ora si dà sia la dimensione del video che quella "reale")
  - **dot pitch**: dimensione del punto fisico (in mm, es. 0.24)
  - **frequenza di refresh**: frequenza di visualizzazione delle immagini; spesso vengono gestite più frequenze (in Hz, es. 67)
  - **risoluzione**: in pixel, es. 800x600, 1024x768

---

---

---

---

---

---

---

---

## Le stampanti

- periferica di output per la stampa dei dati su carta
- quasi sempre l'immagine viene prodotta da un insieme di punti (come nel video)
- caratteristiche salienti:
  - **qualità di stampa**: si misura considerando il numero di punti stampabili per unità di superficie, **dots per inch (dpi)**
  - **velocità di stampa**: nelle stampanti orientate al carattere, si misura in linee/minuto o caratteri/secondo; nelle stampanti orientate alla pagina, in pagine/minuto
  - **colore**: di solito le stampanti sono b/n o colori; le sfumature vengono ottenute con retini ed altre tecniche discrete. Nelle stampanti a colori, è importante sapere come viene ottenuto il nero (CMY oppure CMYK)

---

---

---

---

---

---

---

---

## Digressione: i caratteri

- **font**: insieme di caratteri tipografici che rappresentano l'intero insieme di caratteri stampabili con un determinato aspetto comune
- due categorie di dimensionamento/spaziatura:
  - fissa (i caratteri occupano tutti la stessa larghezza)
  - variabile/proporzionale
- dimensioni misurate in: **punto** (=1/72 di pollice) e **pica** (=1/6 di pollice) tra la parte più alta (es. t) e quella più bassa (es. p)
- caratteristiche del testo: font, dimensione e **stile**
- Esempi: Times, Chicago, Zapf Chancery, Courier, Arial 14, arial 20, **grassetto**, **corsivo**, sottolineato, ombreggiato, **di tutto**

---

---

---

---

---

---

---

---

## Tipi di stampanti

- (Stampanti a *margherita/testina rotante*: come le macchine da scrivere, stampano solo caratteri)
- Stampanti *ad aghi*: stampa ad impatto con una matrice di aghi che compone il carattere/ immagine al momento. 9-24 aghi, risoluzioni ~300 dpi, rumorose. Utili per stampare più copie con carta carbone e per moduli continui.
- Stampanti a *getto d'inchiostro*: inchiostro liquido magnetizzato viene spruzzato su un foglio, deviato tramite strumenti magnetici per formare l'immagine. Buona risoluzione, costo d'esercizio più alto delle precedenti.
- Stampanti *laser*: inchiostro in polvere (toner) depositato sulla carta tramite raggio laser e fissato con riscaldamento. Di solito sono dotate di propri processore e memoria, e costruiscono la rappresentazione della pagina completa partendo da una sua descrizione in linguaggio **Postscript**. Ottima risoluzione, più costose.
- Stampanti a *sublimazione*, a *cera*, ...: alta qualità, alto costo

---

---

---

---

---

---

---

---



## Una stampante particolare: il plotter

- stampanti particolari utilizzate per la stampa di disegni tecnici
- permettono la stampa "vettoriale" grazie ad una penna in grado di scorrere con continuità sul foglio
- due tipi:
  - plotter piani, ove la carta è disposta su un piano, e la penna si muove sulle due coordinate
  - a rullo, ove la penna si muove nella direzione x e la carta scorre nell'altra direzione

---

---

---

---

---

---

---

---

## Dispositivi di puntamento su video

- permettono all'utente di indicare una posizione su un video grafico; utilizzato nelle interfacce utente a finestre (cfr. più avanti...)
- **mouse**: scatolette da muovere manualmente su una superficie, dotate di uno o più pulsanti per input aggiuntivi, che registrano la posizione (spesso) per mezzo di una sfera rotante. Tecnologie: ottici, meccanici, optomeccanici.
- **trackball**: specie di mouse rovesciato comune nei computer portatili
- **trackpad**: area sensibile al contatto del dito (es. misura variazioni resistenza)
- **tavolette grafiche**: praticamente come il mouse, ma scorrono su un supporto specifico sensibile. Molto precise (di solito non a sfera).
- **penne luminose**: dispositivi con i quali si può puntare direttamente lo schermo. Basate su principi ottici, rilevano il passaggio del fascio elettronico del monitor, ed in base a quello ricavano la posizione della penna
- **schermi sensibili**: rilevano la presenza del dito direttamente sullo schermo (es. pressione o calore)

---

---

---

---

---

---

---

---

## Input di immagini

- **Scanner**: dispositivo per l'acquisizione di immagini da supporti tradizionali (carta, diapositive, ...). Risoluzione fino a 9600 dpi.
- **lettore di codici a barre**: specie di scanner specializzato nella lettura di codici a barre
- **schede di acquisizione video**: permettono la lettura e memorizzazione di immagini provenienti da sorgenti video (es. videocamere, VCR). Possono acquisire immagini fisse o video continuo (con limitazioni che dipendono da tutto l'hardware). Risoluzione fino a ~832x624 (PAL)
- **fotocamere digitali**: acquisizione di immagini fisse come in fotografia, collegate discontinuamente. Risoluzioni fino a 5000x5000

---

---

---

---

---

---

---

---

## Input/output di suoni

- esistono periferiche per la gestione dei suoni sia in input che in output
- input:
  - *schede di acquisizione sonora* in grado di campionare il suono con qualità CD. Da accoppiare a microfono o altra sorgente sonora
  - *lettore CD*: acquisizione dei dati digitali presenti sul CD
- output:
  - *schede di riproduzione sonora*: riconvertono suoni digitalizzati in emissioni sonore tramite altoparlanti.
  - *periferiche MIDI* (Musical Instrument Digital Interface): di solito sono strumenti musicali che possono essere guidati da calcolatore sia nell'emissione di suoni (sintetici) che nell'input (es. spartiti)

---

---

---

---

---

---

---

---

## Comunicazione

- periferiche che permettono la connessione del calcolatore con altri calcolatori
- *modem*: dispositivo che permette la comunicazione su linea telefonica convertendo segnali digitali interni al calcolatore in impulsi adatti alla trasmissione telefonica. Velocità disponibili: 14.4-57.6 Kb/s.
- *schede di rete*: dispositivi di input/output che permettono la comunicazione tra calcolatori su cavo, con velocità tipiche attorno a 10 Mbit/s. Ne parleremo più approfonditamente.

---

---

---

---

---

---

---

---

## Come sono collegate le periferiche?

- tramite interfacce standard di vario tipo, a seconda della periferica.
- di solito si trovano sul retro del computer...
- esempi:
  - *interfaccia seriale*: trasmissione ad 1 bit per volta, più segnali di handshake, velocità 9.6-230 Kbit/s. Per modem e qualche stampante.
  - *interfaccia parallela*: trasmissione a 8 bit per volta, usata per la connessione di stampanti
  - *SCSI*: interfaccia parallela ad alta velocità (2-10 Mb/s). Per memoria di massa e qualche fotocamera digitale
  - *USB* (Universal Serial Bus): recentissima seriale ad alta velocità, per dispositivi seriali, floppy disk, etc
  - *PCMCIA*: interfaccia formato carta di credito, ospita di tutto (fax, schede ethernet, memoria di massa, etc)
  - *slot di espansione interni*: basati su standard specifici (es. PCI, NuBus, VESA,...), di solito hanno altissima velocità.

---

---

---

---

---

---

---

---

Marino Miculan, Università di Udine - 2002 - 111

## Esempio di funzionamento: la tastiera

- schiaccio un tasto sulla tastiera;
- i circuiti interni lo codificano con 8 bit;
- gli 8 bit vengono passati al controller, uno alla volta:

**TASTIERA:**

- mette 0V sul filo "dato", se il bit da trasmettere è 0, altrimenti 5V
- mette 5V sul filo "spedisco";
- attende che sul filo "ricevuto" ci siano 5V;
- mette 0V sul filo "spedisco"
- ripete 8 volte con gli 8 bit

**CONTROLLER:**

- attende che sul filo "spedisco" ci siano 5V
- legge il voltaggio dal filo "dato" e setta un bit nella sua memoria interna
- mette 5V sul filo "ricevuto"
- attende che sul filo "spedisco" ci siano 0V
- mette 0V sul filo "ricevuto"
- attende che sul filo "ricevuto" ci siano 5V

- il controller copia la sua memoria interna in una locazione della memoria centrale;
- mette 5V sul filo del bus di nome IRQ, collegato ad un bit del registro di stato RS, che diventa 1;
- il processore:
  - all'inizio di una istruzione, controlla gli interrupt in RS;
  - si accorge che il controller ha avvisato che c'è un dato;
  - "salva il contesto" (l'istruzione che sta eseguendo)
  - salta al programma di gestione della tastiera
  - che legge il valore dall'indirizzo 199 e fa quel che deve...
  - al termine, il processore torna al contesto salvato

---

---

---

---

---

---

---

---

---

---

Marino Miculan, Università di Udine - 2002 - 112

## Classi di elaboratori

- Possiamo distinguere gli elaboratori in classi a seconda di potenza di calcolo e numero di utenti:
  - **Personal Computer (PC)**: monoutente, dotati di video, tastiera e stampante, spesso connessi in rete locale. Es: IBM-compatibili (Windows), Macintosh
  - **desktop**: "da scrivania"
    - **laptop/notebook**: portatili con video LCD, dimensioni libro
  - **Personal Digital Assistants (PDA)/Palmtop**: elaboratori monoutente estremamente piccoli (agenda), da utilizzare in connessione con un PC, con dispositivi di puntamento a penna su video LCD (es. Windows CE)
  - **Micro e Mini Computer**: multiutente, con più terminali video. Potenza superiore a quella dei PC. Adesso si preferiscono le:
  - **Workstation**: elaboratori monoutente molto potenti, connessi in rete
  - **Mainframe**: sistemi multiutente molto potenti (più dei mini)
  - **Supercalcolatori**: potentissimi, per elaborazioni scientifiche (es. Cray). A volte hanno più processori.

---

---

---

---

---

---

---

---

---

---

Marino Miculan, Università di Udine - 2002 - 113

## Il software

- Il calcolatore, che abbiamo descritto nelle sue componenti hardware, da solo non fa niente
- è comunque programmabile, e quindi gli si può fare svolgere diversi compiti
- la programmazione diretta del processore è molto difficoltosa
- Servono meccanismi per:
  - *astrarre* dall'organizzazione fisica della macchina
  - *usare in modo simile* macchine diverse
  - avere *modalità semplici di interazione* con la macchina
  - avere un *linguaggio semplice per programmare* la macchina
  - avere dei *programmi applicativi* per svolgere compiti di alto livello

---

---

---

---

---

---

---

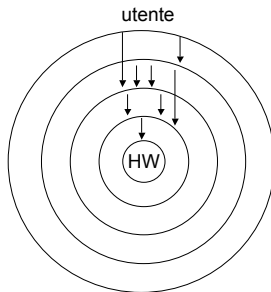
---

---

---

## La Macchina Virtuale

- I meccanismi appena illustrati si concretizzano nella realizzazione di **macchine virtuali** che implementano tramite software le funzionalità desiderate
- Ciò avviene gerarchicamente secondo una **struttura a cipolla**: ogni strato fornisce funzionalità sempre più astratte allo strato successivo (il primo è l'hardware, l'ultimo è l'utente); ogni strato è una **macchina virtuale**
- cambia il livello di astrazione, ma non le operazioni che possono essere fatte: tutto deve essere fattibile dall'hardware
- esempio di strati: l'impiegato e l'ufficio (che fa cose più astratte)



## Il software di base

- è l'insieme dei programmi in linguaggio macchina che realizzano la macchina virtuale che permette all'utente di interagire con la macchina
- Due categorie di funzioni:
  - **sistema operativo**
    - avviamento del calcolatore e creazione dell'ambiente virtuale
    - gestione del processore e dei processi
    - gestione della memoria principale
    - gestione della memoria secondaria
    - gestione delle periferiche
    - interazione tra utente e sistema
    - comunicazione tra gli utenti e tra gli elaboratori
  - **produzione di programmi**
    - traduzione tra linguaggi diversi (interpreti, compilatori)
    - strumenti per lo sviluppo di programmi

## Il sistema operativo (OS)

- componente software fondamentale di un elaboratore
- costituito da un insieme di programmi interagenti e cooperanti al fine di:
  - gestire efficientemente il calcolatore e le sue periferiche, cercando di sfruttare al massimo le risorse disponibili;
  - creare un ambiente virtuale che permetta l'interazione uomo/macchina
- realizzato secondo una struttura a cipolla
  - ad ogni livello funzionalità diverse:
    - *basso livello*: gestione efficiente di memoria, processore, periferiche
    - *alto livello*: interazione con l'utente

## Categorie di sistemi operativi

- Distinzione in base al numero di utenti:
  - **mono-utente**: elaboratori di tipo personale
  - **multi-utente**: elaboratori utilizzabili da più utenti contemporaneamente
- Distinzione in base al modo di elaborazione:
  - **mono-programmati (monotasking)**: elaboratori in grado di eseguire un solo programma alla volta
  - **multi-programmati (multitasking)**: elaboratori in grado di eseguire più programmi contemporaneamente
- il sistema operativo deve implementare funzionalità diverse a seconda della categoria
- le funzionalità non dipendono dal processore (sono più astratte): es. su *Pentium* possiamo usare un OS mono-utente (Windows) ma anche multi-utente (Linux)

---

---

---

---

---

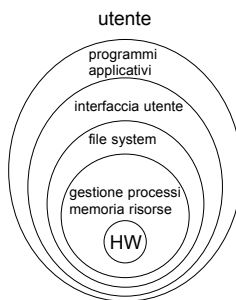
---

---

---

## Funzionalità principali

- avviamento del calcolatore e configurazione dell'ambiente virtuale
  - bootstrap
- gestione del processore e dei *processi*
- gestione della memoria principale
  - memoria reale e virtuale
- gestione della memoria secondaria
  - il file system
- gestione delle periferiche
- interazione con l'utente (a comandi o grafica)




---

---

---

---

---

---

---

---

## Avviamento del calcolatore

- al momento dell'accensione è necessario caricare dalla memoria secondaria nella principale una parte del S.O., necessaria alla gestione delle risorse:
- programmi per la gestione dei processi e del processore
  - programmi per la gestione della memoria
  - programmi per la gestione delle periferiche
  - programmi per la gestione del file system
  - (programma per l'interazione con l'utente)
- ciò viene effettuato da un ulteriore programmino, che viene eseguito esattamente all'accensione e che usualmente è memorizzato in ROM (Read-Only Memory: parte di memoria principale non riscrivibile e permanente)
- questo programma di caricamento iniziale è in una posizione nota al processore e non necessita del S.O.
- i primi programmi caricati sono situati in posizioni note

---

---

---

---

---

---

---

---

## Avviamento del calcolatore

- man mano che si caricano i pezzi del S.O., si “creano” le macchine virtuali che forniscono funzionalità sempre più astratte
- inoltre:
  - vengono identificate le risorse disponibili (periferiche, memoria secondaria)
  - per ognuna risorsa viene lanciato il programma di gestione
  - a volte viene verificato lo stato della risorsa
- il bootstrap è anche il momento in cui vengono lanciati i programmi *antivirus*, in modo che il controllo sia possibile prima dell'ingresso di eventuali virus

---

---

---

---

---

---

---

---

## Gestione del processore

- essendo la più importante componente di un sistema di elaborazione, il processore deve essere gestito efficientemente
- il suo compito è *eseguire programmi*:  
**un processo è un programma in esecuzione**
- anche se il processore esegue una istruzione alla volta, potrebbero esserci più processi in esecuzione, alternativamente. Come e perché?
- sistemi monotasking: uso inefficiente del processore, perché nei tempi di attesa dovuti all'accesso alla memoria secondaria o alle periferiche il processore non fa niente.
- multitasking: sfrutta le attese per parallelizzare

---

---

---

---

---

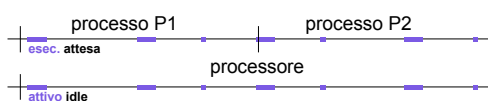
---

---

---

## Il processo

- qualunque processo alterna fasi di esecuzione a fasi in cui è bloccato in attesa di qualche evento esterno (es. accesso a risorse, risposta dell'utente)
- la velocità di elaborazione del processore è migliaia di volte superiore a quella di risposta delle periferiche, e milioni rispetto a quella dell'utente
- quindi i tempi di attesa, nei programmi che fanno uso di risorse generiche, sono molto maggiori dei tempi di esecuzione
- il processore passa quindi molto tempo inattivo (**idle**)




---

---

---

---

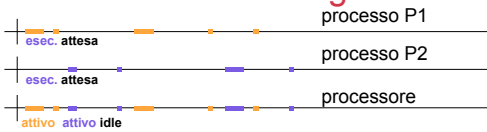
---

---

---

---

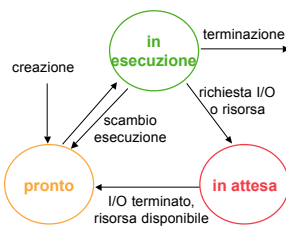
## Multitasking



- ogni volta che P1 è in attesa di evento esterno, P2 viene mandato in esecuzione
- c'è contemporaneità macroscopica che corrisponde ad una rapida alternanza
- il numero di processi attivi viene detto **grado di multiprogrammazione**
- diversi modelli di multiprogrammazione: **batch**, **time-sharing**, **realtime**

## I processi

- in un sistema multitasking, un processo può trovarsi in 3 stati (**in esecuzione**, **in attesa**, **pronto**).
- Un solo processo è in esecuzione, mentre può essercene più d'uno negli altri due stati, organizzati in **code**
- nei sistemi time-sharing, il tempo è suddiviso tra i processi: c'è uno scambio di esecuzione se non ci sono pause "naturali"



## I problemi del multitasking

- **gestione dei processi:** operazioni necessarie per ricordare e gestire lo stato di ogni processo
  - **immagine** di un processo: codice + dati
  - **tabella dei processi:** informazioni su tutti i processi
  - **cambio di contesto:** avviene ogni volta che il processo in esecuzione viene fermato e ne viene eseguito un altro
- **scheduling del processore:** come scegliere il prossimo processo da mandare in esecuzione?
  - massimizzando l'uso del processore;
  - massimizzando il numero di processi eseguiti;
  - minimizzando il tempo di esecuzione dei processi;
  - minimizzando il tempo di attesa dei processi
  - Es: Round Robin, FCFS
- **coordinamento, sincronizzazione e mutua esclusione**
  - conflitto e competizione nella condivisione di risorse
  - scambio di dati tra processi

## I processi di sistema

- i processi relativi al sistema operativo sono particolari, ed alcuni di questi sono sempre attivi e presenti in memoria principale
- vengono mandati in esecuzione non appena necessario (es. tramite interrupt) e hanno priorità sempre maggiore dei processi utente
- questo succede anche nei sistemi monotasking

---

---

---

---

---

---

---

---

## Gestione della memoria principale

- Problema: se più processi devono essere eseguiti contemporaneamente, in qualche modo devono condividere la memoria principale
- Altro problema: se più processi devono utilizzare la memoria, si pone il problema di come non sovrapporre/danneggiare dati
- entrambi i problemi sono validi anche per i processi di sistema, che sono sempre in memoria e sempre attivi, e devono essere protetti dai processi utente
- soluzione: registri **fence** (staccionata)
- Può essere utile dare ai processi una visione astratta della memoria, che gli permetta di usare virtualmente tutta la memoria anche se condivisa con altri processi, ed a volte anche più di quella presente fisicamente

---

---

---

---

---

---

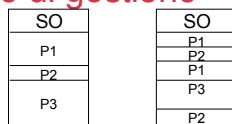
---

---

## Metodologie di gestione

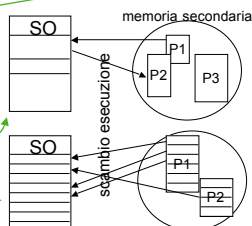
- Memoria reale: suddivisione della memoria principale tra i processi

- partizioni multiple con allocazione contigua
- allocazione non contigua



- Memoria virtuale: tutta la memoria è virtualmente disponibile per ogni processo utilizzando porzioni di memoria secondaria

- swapping
- demand paging




---

---

---

---

---

---

---

---



## Memoria reale e virtuale

- memoria reale:
  - la memoria a disposizione di ogni processo dipende da quanti processi sono in esecuzione,
  - ed è sempre minore o uguale alla memoria principale
- memoria virtuale
  - in generale il tempo perso nel cambio di contesto è relativamente elevato
  - basata sullo swapping di processi: un processo vede al massimo tutta la memoria principale
  - demand paging: la memoria a disposizione del singolo processo non è legata alle caratteristiche fisiche della macchina.
    - politiche di scelta delle pagine "inutili" possono rendere comunque estremamente efficiente il demand paging

---

---

---

---

---

---

---

---

## Gestione della memoria secondaria: il file system

- la memoria secondaria è normalmente realizzata tramite dischi magnetici, per i due scopi trattati:
  - memorizzazione permanente di dati e programmi
  - supporto della memoria principale (virtuale)
- la memorizzazione permanente avviene tramite il **file system**:
  - strutturazione e gestione delle informazioni,
  - organizzate in **file**,
  - allo scopo di:
    - dare all'utente una *visione logica* dei file
    - gestire i file nella memoria secondaria

---

---

---

---

---

---

---

---

## Il file

- meccanismo di strutturazione delle informazioni per l'aggregazione di informazioni elementari in strutture più complesse
- basato sui concetti di:
  - **campo**: insieme di byte che codifica una singola informazione (numerica, alfanumerica, immagine, etc)
  - **record**: insieme di campi logicamente correlati che assieme costituiscono una nuova informazione complessa
  - **file**: sequenza di record
- categorie:
  - record a lunghezza costante / a lunghezza variabile
  - file strutturati / file di testo
  - file ad accesso sequenziale / diretto / indicizzato

---

---

---

---

---

---

---

---

## Organizzazione logica dei file

- astrazione dei file fisici su disco dotandoli di:
  - identificazione con nome logico: permette di astrarre dai dettagli di memorizzazione
  - operazioni sui file (creazione, rimozione, copia, visualizzazione, stampa, lettura/modifica/scrittura, ridenominazione, esame delle caratteristiche): permettono di lavorare con i file
  - organizzazione strutturata dei file in insiemi e sottoinsiemi: aiutano nella gestione di grandi quantità di file, di solito grazie ad una organizzazione gerarchica degli stessi in directory e sotto-directory
  - meccanismi di protezione dei file da altri utenti/programmi (in sistemi multiutente): deve essere possibile stabilire l'utente proprietario del file e permettere solo ad esso lettura, scrittura, esecuzione

---

---

---

---

---

---

---

---

## Organizzazione gerarchica dei file

- molti S.O. permettono di vedere il file system come **albero** di directory e file (che fanno da **foglie**)
- organizzazione logica gerarchica dei file
- la directory è un file particolare che contiene informazioni strutturate, tra cui: la directory "padre" (che sarà una directory) e l'elenco dei figli (file o directory)
- ogni S.O. fornisce un metodo **-pathname-** per identificare univocamente un file nella gerarchia (es. il percorso **-path-** completo per raggiungerlo). Si hanno pathname **assoluti** e **relativi** e mezzi per riferimenti all'indietro

---

---

---

---

---

---

---

---

## Organizzazione fisica dei file

- dal livello logico:
  - macchina virtuale: record, file, directory
- a quello fisico
  - macchina virtuale: blocchi di dati
- (fino alla magnetizzazione del disco...)
- Problemi da considerare:
  - quali file sono memorizzati, dove, con che caratteristiche
  - ottimizzazione dello spazio su disco per evitare sprechi
- Soluzione: disco diviso in due parti
  - **device directory**: tabella con informazioni sui file (nome, data, dimensione, proprietario, tipo, dove è memorizzato)
  - spazio per la memorizzazione (a blocchi contigui o sparsi)

---

---

---

---

---

---

---

---

Marino Miculan, Università di Udine - 2002 - 135

## Esempi

- **Allocazione contigua:** ogni file è in blocchi consecutivi, il descrittore del file contiene inizio e lunghezza
  - veloce; spazio sprecato
- **Allocazione sparsa:** blocchi situati ovunque nello spazio di memorizzazione
  - spazio usato efficientemente
  - **indexata:** il descrittore del file contiene una tabella con tutti gli indirizzi dei blocchi
  - **linkata:** il descrittore contiene l'indirizzo del primo blocco, e poi ogni blocco ha un riferimento al blocco successivo

---

---

---

---

---

---

---

---

Marino Miculan, Università di Udine - 2002 - 136

## Gestione delle periferiche

- Problemi da considerare:
  - fornire all'utente una visione *astratta* delle periferiche, con comandi semplici e ad alto livello per il loro uso
  - ottimizzare l'uso dei dispositivi
- modello **master/slave**
- Due tipi di dispositivi:
  - a **controllo di programma:** il S.O. accede alla memoria privata del dispositivo (di solito piccola)
  - **DMA** (Direct Memory Access): leggono e scrivono i dati direttamente in memoria centrale
- il S.O. viene avvertito della necessità di operazioni tramite **interrupt**, e manda in esecuzione il **driver** del dispositivo
- Astrazione:
  - l'utente deve poter usare periferiche logicamente simili nello stesso modo, anche se fisicamente sono diverse (es "print" per stampare sia su laser che su ink-jet)

---

---

---

---

---

---

---

---

Marino Miculan, Università di Udine - 2002 - 137

## Interazione con l'utente

- Del calcolatore, cosa viene visto dall'utente?
  - alcune delle funzionalità del S.O. (es. gestione file e periferiche ad alto livello)
  - programmi applicativi
  - accesso ai dati
- in generale l'utente accede alle funzionalità del sistema tramite **un'interfaccia utente**, basata su un **linguaggio di interazione tra utente e sistema**
- due grandi famiglie di linguaggi:
  - ad interazione **testuale:** l'utente utilizza comandi testuali (codici mnemonici) con sintassi ben precisa
  - ad interazione **grafica:** l'utente accede ai comandi ed ai dati per mezzo di elementi grafici selezionabili con dispositivi di puntamento video

---

---

---

---

---

---

---

---

## Funzionalità dell'interfaccia

- gestione dei file
  - operazioni del livello logico del file system
- gestione delle periferiche
  - operazioni di stampa
  - funzionalità di tastiera e mouse
- gestione dei programmi applicativi
  - memorizzazione nel file system
  - esecuzione dei programmi
- accesso ai dati
  - visualizzazione dati di tipo testo, etc
- (funzionalità di base per i programmi applicativi)
  - gestione tastiera, mouse, video, stampante,...

---

---

---

---

---

---

---

---

## Interazione testuale

- è realizzata per mezzo di un processo che:
  - viene mandato in esecuzione all'avvio della macchina;
  - segnala la disponibilità ad accettare comandi visualizzando un **prompt** (es ">" sul video)
  - l'utente digita **nome e parametri** di un comando, poi preme *return*;
  - il processo analizza quanto è stato scritto e:
    - se è corretto, manda in esecuzione il programma corrispondente;
    - altrimenti segnala l'errore all'utente
  - poi si mette di nuovo in attesa visualizzando il prompt
- questo processo è detto **interprete di comandi** (*Command Line Interpreter, CLI*)
- la sintassi dipende dal S.O.

---

---

---

---

---

---

---

---

## Interazione testuale

- entità in gioco (correlate, ma diverse!):
  - comando
  - programma mandato in esecuzione
  - file in cui il programma è memorizzato
- i parametri dei comandi indicano:
  - su **cosa** il comando deve agire (**argomenti**; es. altri file)
  - **come** il comando deve agire (**opzioni** che permettono di scegliere tra comportamenti diversi del comando)
- Esempi:
  - MS-DOS: `dir c:\prova /p`  
-> mostra i file nella directory "c:\prova" pagina per pagina
  - Unix: `ls -l /home/prova`  
-> mostra i file nella directory "/home/prova", stampandone tutte le caratteristiche

---

---

---

---

---

---

---

---

### Interazione testuale: pro e contro

- viene usata sempre meno nei personal computer, perché non è molto *user-friendly*
- è necessario conoscere e ricordare la sintassi dei comandi, le opzioni possibili, la posizione degli argomenti
- ci sono delle facilitazioni: help-on-line, abbreviazioni automatiche, etc.
- continua ad essere usata a livello tecnico (es. programmatori, sistemisti, etc), perché se il sistema di interazione è conosciuto molto bene, permette un'interattività molto veloce e riproducibile
- ed è anche possibile automatizzare facilmente sequenze di operazioni, tramite il concetto di script:
  - MS-DOS: batch files, file di testo eseguibili che contengono comandi
  - Unix: shell scripts: idem

---

---

---

---

---

---

---

---

### Interfaccia grafica

- come superare i limiti di utilizzabilità delle interfacce a carattere da parte di non esperti?
- Metodi alternativi basati su altre tecniche
- Interfaccia grafica (*Graphical User Interface, GUI*):
  - nata da studi c/o Palo Alto Research Center della Xerox, primi anni '80
  - 1983-84: alcuni ex-Xerox progettarono Apple Lisa -> Macintosh
  - poi il resto (Window, X-Windows, Amiga Intuition, GEM, Atari, NextStep, ...)
- l'accesso alle funzionalità del calcolatore avviene tramite un video grafico con un sistema di puntamento (es. mouse)

---

---

---

---

---

---

---

---

### Metafore per le GUI

- La *metafora della scrivania*
  - Il video rappresenta il piano di una *scrivania*, sul quale appaiono gli "oggetti" come apparirebbero su una scrivania vera
  - gli oggetti sono rappresentati da *icone* (piccole immagini esplicative del contenuto dell'oggetto)
  - i dati/programmi sono visti come "oggetti concreti" con comportamento ragionevolmente prevedibile
  - si opera sui dati selezionando gli oggetti coinvolti ed applicando una particolare operazione o strumento
- il file system è rappresentato da uno *schedario* che contiene
  - i files di dati, equivalenti ai documenti presenti su una scrivania
  - le directory, equivalenti a *cartelle* dentro uno schedario
  - i programmi applicativi (equivalenti agli strumenti che si trovano sulla scrivania, es. rubrica, penna, etc)

---

---

---

---

---

---

---

---

Metafore per le GUI

Gli elementi grafici di base:

- **Cursore**: piccola componente grafica che appare sul video, la cui posizione è correlata al movimento del mouse. Di solito è a forma di freccia, ma può cambiare a seconda del contesto (es. orologio).
- **Menu**: metodo per accedere ai comandi senza conoscerne nome o sintassi. Un menu è costituito da una lista di comandi che appare secondo varie modalità (pull down, pop up, sensibili al contesto, etc), di solito attivate da un click col mouse su un titolo o su un oggetto
- **Icone**: piccole immagini associate agli oggetti gestiti dal S.O. (principalmente quelli del file system). L'aspetto richiama il contenuto dei documenti o l'uso degli strumenti.
- **Pulsanti**: stilizzazione di pulsanti veri e propri, vengono usati per risposte che generano azioni immediate (**buttons**), per scelte alternative mutuamente esclusive (**radio buttons**, metafora dei bottoni delle vecchie radio), per selezionare opzioni (**check box**, metafora dei quadratini da questionario).
- **Finestre**: sono aree visuali del video che mostrano dati o programmi in esecuzione, icone, testo, grafica, etc. Hanno accessori per chiusura, spostamento, ridimensionamento, etc. Alcune sono dette **dialoghi**

---

---

---

---

---

---

---

---

Metafore per le GUI

Azioni possibili:

- Apertura di icone: tramite posizionamento con il mouse e "click", causa il lancio del programma associato o l'apertura di un documento con il programma che l'ha generato
- Apertura/chiusura di finestre: creazione/rimozione di un'area visiva contenente dati e/o ulteriori elementi grafici.
- **Cut&paste**: selezione di oggetti o porzioni di oggetti (es. testo o grafica), loro eliminazione (cut) dalla posizione corrente e deposito in un'area di sistema detta clipboard, da cui la selezione può essere ricopiata altrove (paste)
- **Drag&Drop**: trascinamento di oggetti da un punto ad un altro.
- Differenze tra sistemi: numero di click del mouse, posizione degli elementi grafici delle finestre, e poco altro

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

## S.O.: casi di studio

- Caratteristiche che studieremo:
  - modello di esecuzione
  - processi e memoria
  - file system
  - linguaggio di interazione
- Sistemi che vedremo:
  - MS DOS
  - Unix
  - MacOS
  - Windows

---

---

---

---

---

---

---

---

## MS DOS (Microsoft Disk Operating System)

- sistema operativo:
  - monoutente,
  - monoprogrammato,
  - con interfaccia di interazione testuale
- memoria reale (con limitazioni)
- funziona con processori Intel (e compatibili), da 8088 in su (a seconda della versione)
- non molto evoluto, e ormai poco usato
- file system ad albero (ad alberi)
- caratteristico linguaggio di comandi
- è (stato) la base per Windows

---

---

---

---

---

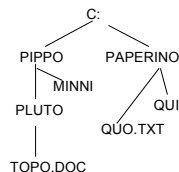
---

---

---

## MS DOS: il file system

- ad ogni supporto di memoria secondaria è associato un albero, la cui radice ha come nome convenzionale una lettera:
  - A:, B: sono i dischi floppy
  - C:, D:, ... sono dischi rigidi, CD, etc.
- nomi di file: massimo 8 caratteri, più 3 di estensione. L'estensione identifica il tipo di file: eseguibili (.COM, .EXE, .BAT), di sistema (.SYS), di testo (.TXT), proprietari (.DBF, .DOC, ...). Maiuscole e minuscole indistinte.
- pathname: i nomi delle directory e dei file nel percorso sono separati da "\". Es: C:\PIPPO\PLUTO\TOPO.DOC (backslash).
- pathname relativi: sono possibili. "." permette di risalire al padre. Es: da PLUTO, ..MINNI




---

---

---

---

---

---

---

---

## MSDOS: Il linguaggio di comandi

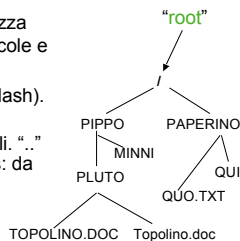
- `CD path`: cambia la directory
- `DIR [path]`: mostra il contenuto di una directory
- `MKDIR nome`: crea una nuova directory
- `RMDIR path`: cancella una directory
- `DEL file`: cancella un file
- `TYPE file`: mostra un file di testo
- `COPY path1 path2`: copia un file
- `nomefile`: manda in esecuzione il file eseguibile corrispondente al nome
- Buona parte dei comandi sono residenti in memoria e vengono lanciati durante il boot tramite il programma `COMMAND.COM`

## Unix

- sistema operativo:
  - multiutente,
  - multitasking,
  - con interfaccia di interazione testuale + altri tipi
- memoria virtuale
- esistono implementazioni per qualsiasi processore
- usato per micro/mini computers, servers, etc.
- file system ad albero unico
- linguaggio di comandi più o meno standard, con diverse varianti (shell)
- Diversi "dialetti" Unix, tutti molto simili: BSD 4.2, AT&T SystemV -> SCO, Solaris, Linux, HP/UX, ...
- esistono diverse interfacce grafiche per Unix (XWindows, Motif, ...)

## Unix: il file system

- albero unico, la cui radice coincide con uno dei dischi;
- gli altri dischi possono essere collegati ad una qualsiasi sottodirectory, tramite l'operazione di mount
- nomi di file: max 32 caratteri, estensioni comprese (a lunghezza libera, anche più d'una); maiuscole e minuscole sono diverse.
- pathname: il separatore è "/" (slash).  
Es: `PIPPO/PLUTO/Topolino.doc`
- pathname relativi: sono possibili. "..."  
permette di risalire al padre. Es: da `PLUTO`, `../MINNI`
- i files hanno delle proprietà:  
es. readable, writable, executable





## Unix: modello di esecuzione


- Sistema con gestione raffinata di
  - **processore**: time sharing con variante di round-robin
  - **memoria** virtuale basata su demand paging + swapping
- multi-utenza: ogni utente
  - è identificato da un nome (**login name**) e accede tramite parola chiave (password)
  - ha una **home directory** di cui è "proprietario", entro la quale i files saranno di sua "proprietà": nel senso che l'utente può decidere che privilegi (R, X, W) associare ad essi rispetto a se stesso, al gruppo cui appartiene, a tutti.
  - c'è un utente particolare, "**superuser**", che accede con nome root ed ha accesso a tutte le risorse del S.O.
- multitasking; a livello di utente, si può decidere se mandare in esecuzione i programmi in
  - **foreground**: come in MSDOS, si aspetta la terminazione;
  - **background**: il programma viene mandato in esecuzione, ed immediatamente è possibile lanciarne un altro contemporaneamente
- il file system usa una variante dell'allocazione indexata

## Unix: il linguaggio di comandi

- l'interprete comandi si chiama **shell**, e può essere scelto da ogni singolo utente tra più versioni (es. C-shell, Bourne Shell, tcsh,...); cambiano i particolari
- alle shell si accede solo tramite login name e password
- oltre 300 comandi molto brevi
- certe periferiche sono gestite come file speciali (es. monitor, stampante, tastiera)
- esistono due modalità operative caratteristiche:
  - **redirezione**: ogni programma ha 3 file standard (stdin, stdout, stderr) normalmente associati a tastiera e video; si può cambiare l'associazione, facendo sì che il programma legga da o scriva su un file. Esempio: `ls nomedir > file.txt`
  - **accodamento** (pipe): l'output di un programma va direttamente come input in un altro programma. Esempio: `ls nomedir | more`
- esteso dispositivo di help in linea: **man**

## Unix: il linguaggio di comandi

- `cd path`: cambia la directory
- `ls [path]`: mostra il contenuto di una directory
- `mkdir nome`: crea una nuova directory
- `rmdir path`: cancella una directory
- `rm file`: cancella un file
- `cat < file`: mostra un file di testo
- `cp path1 path2`: copia un file
- `pwd`: mostra il path della directory corrente
- `nomefile`: manda in esecuzione il file eseguibile corrispondente al nome
- `comando1 & comando2 &...:` lancia in background i comandi seguiti da &
- `comando <path1 >path2`: sostituisce stdin e stdout di comando con i file path1 e path2
- `comando1 | comando2`: pipe di comando1 e comando2, cioè l'output di comando1 è l'input di comando2



MacOS 7-9

Marino Miculan, Università di Udine - 2002 - 156

- Sistema operativo:
  - monoutente
  - multiprogrammato (con limitazioni)
  - con interfaccia di interazione grafica totalmente integrata
  - semplice sistema di memoria virtuale
- il file system è ad albero; la radice visibile è la scrivania, su cui si trovano i dischi presenti nel sistema
- funziona su processori Motorola 680X0 e IBM/Motorola PowerPC
- i files sono peculiari: hanno una data fork (parte dati) ed una resource fork (per codice e dati addizionali)
- utilizza un mouse ad un tasto
- integrazione completa tra S.O. e applicazioni (funziona tutto allo stesso modo)

---

---

---


---

---

---

---

---



Windows

Marino Miculan, Università di Udine - 2002 - 157

- Sistema operativo:
  - monoutente
  - multitasking
  - con interfaccia di interazione grafica ormai totalmente integrata (prima di WIndows95 si appoggiava a MSDOS)
  - memoria virtuale
- il file system è ad albero, con la radice analoga a quella del MacOS;
- funziona su processori Intel e compatibili (Pentium)
- mouse a due/tre tasti

---

---

---

---

---

---

---

---



I programmi applicativi

Marino Miculan, Università di Udine - 2002 - 158

- programmi che forniscono funzionalità di alto livello all'utente.
- Alcune classi:
  - elaboratori di testi
  - fogli elettronici
  - basi di dati
  - programmi di grafica
  - statistica
- I programmi applicativi si appoggiano al sistema operativo ed alla sua interfaccia
  - ogni programma funziona sul S.O. specifico per cui è stato creato
  - è necessario farne versioni diverse per ogni S.O.

---

---

---

---

---

---

---

---