

# Fondamenti di Informatica

---

## CAP. 3

Accademia di belle Arti di Verona

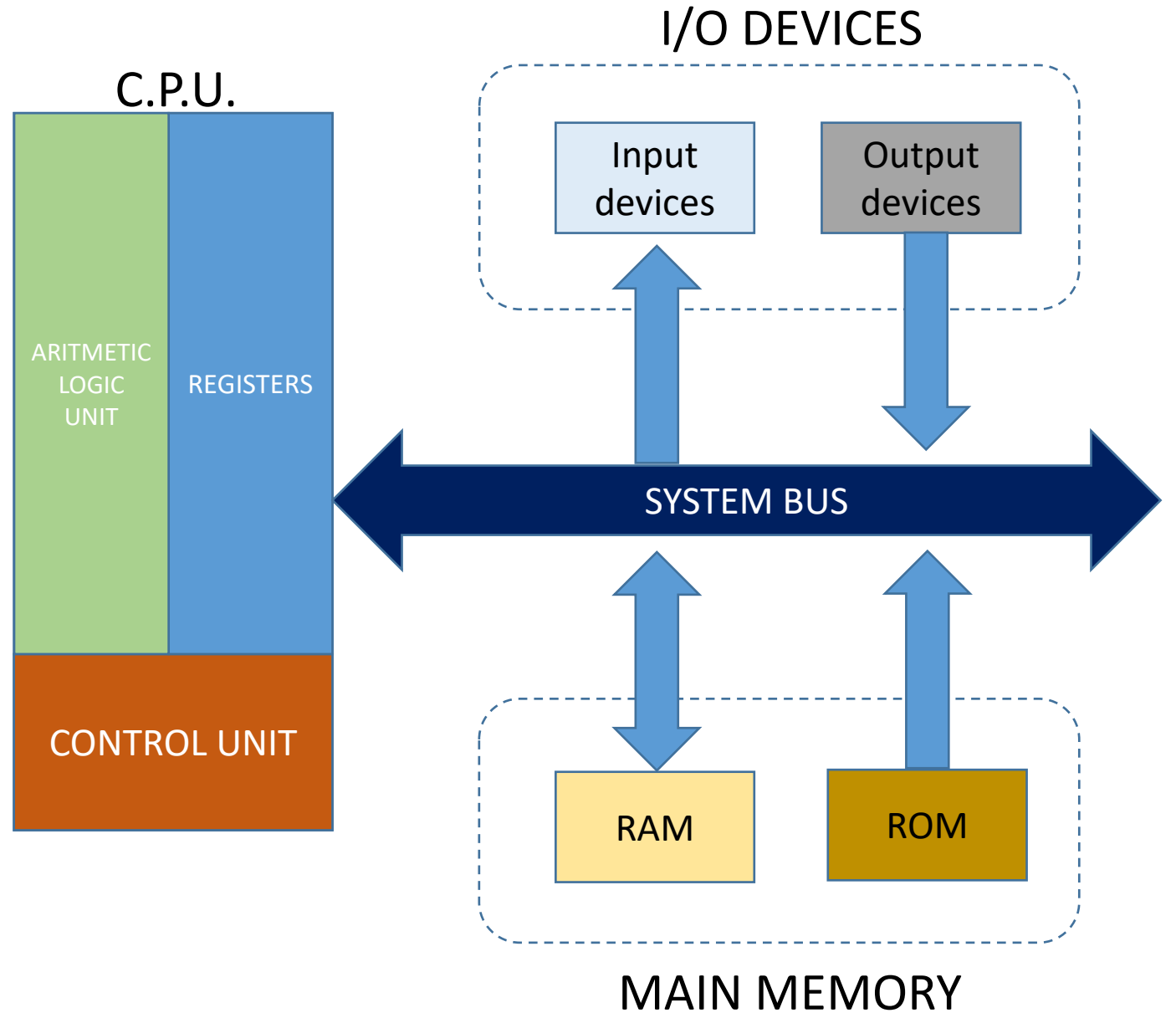
Università degli Studi di Verona

A.A. 2022-2023

Docente - Vincenzo Giannotti

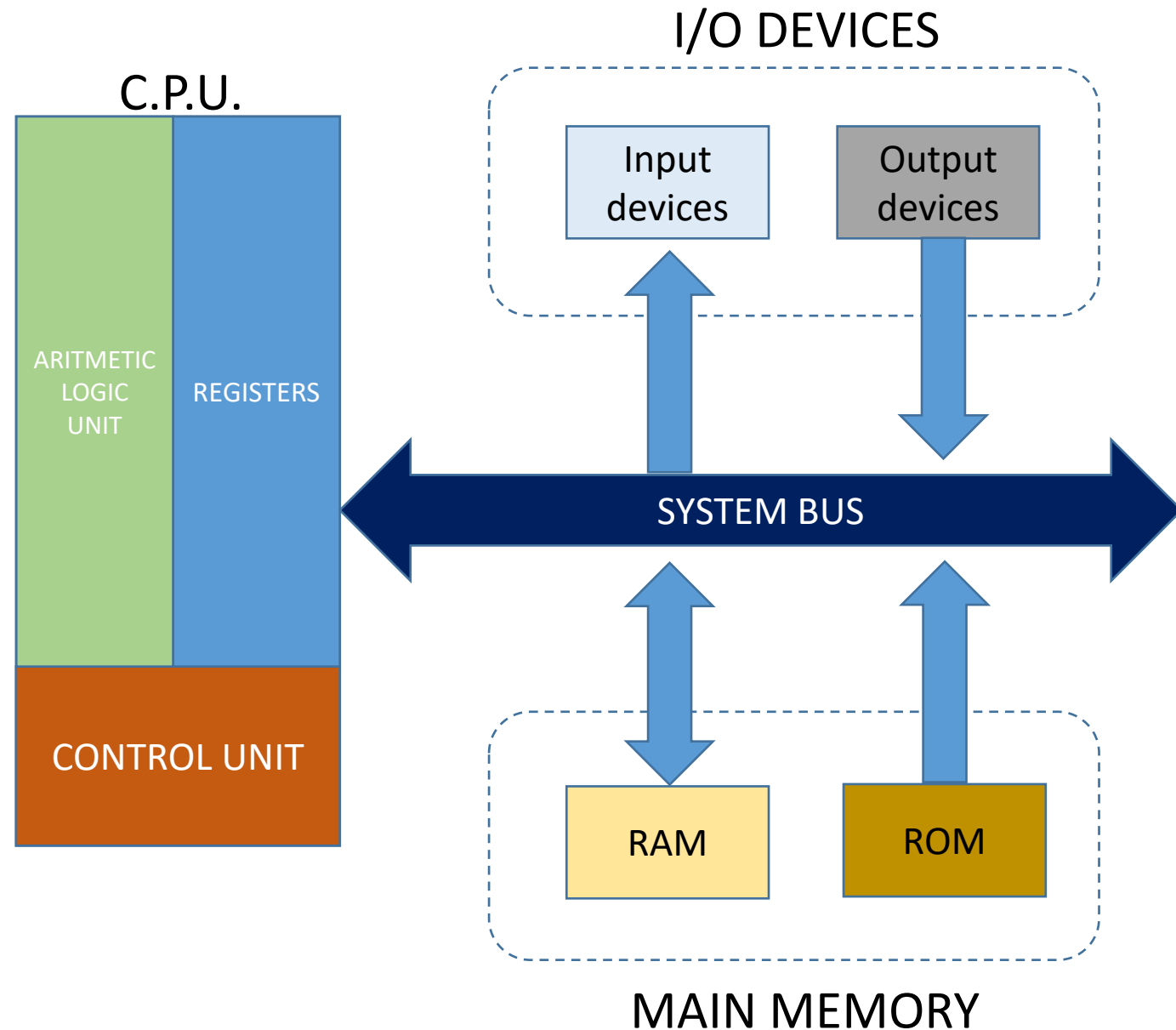
# CAPITOLO 3 – COME FUNZIONA IL COMPUTER

# Architettura di Von Neumann



# Struttura della CPU

- La CPU (microprocessore) rappresenta il cuore del Computer e per questo è detta Unità di Elaborazione Centrale.
- La CPU governa il Computer impartendo delle istruzioni a tutti gli altri organi e interpretando le istruzioni che noi stessi impartiamo: direttamente mediante comandi del SO o indirettamente con dei programmi.
- Essa si compone di tre parti: la ALU (Unità Aritmetica e Logica); I Registri; la CU (Unità di Controllo)



# L'Unità Aritmetica e Logica (ALU)

Innanzitutto il nostro computer deve poter svolgere delle operazioni. Se dobbiamo fare una operazione aritmetica, per esempio una somma o una divisione abbiamo bisogno di una «calcolatrice» che ci permetta di farlo. Questa calcolatrice risiede all'interno della CPU ed è la ALU. Poiché l'acronimo di ALU è **Unità Aritmetica e Logica** se ne deduce che questa parte della CPU non solo può effettuare operazioni di tipo aritmetico, ma anche operazioni di tipo logico, dette anche **operazioni booleane**.

## Operazioni logiche fondamentali Tabelle di verità

<u>A</u>	<u>B</u>	<u>A or B</u>	<u>A</u>	<u>B</u>	<u>A and B</u>	<u>A</u>	<u>not A</u>
0	0	0	0	0	0	0	1
0	1	1	0	1	0	1	0
1	0	1	1	0	0	<i>(negazione)</i>	
1	1	1	1	1	1		

*(somma logica)*                      *(prodotto logico)*

*Le tabelle elencano tutte le possibili combinazioni in ingresso e il risultato associato a ciascuna combinazione*

# ALU

Oltre agli operatori fondamentali possiamo trovare anche altri operatori booleani:

- **NAND** (NOT AND)

A	B	
0	0	1
0	1	1
1	0	1
1	1	0

- **NOR** (NOT OR)

A	B	
0	0	1
0	1	0
1	0	0
1	1	0

- **XOR** (OR esclusivo)

A	B	
0	0	0
0	1	1
1	0	1
1	1	0

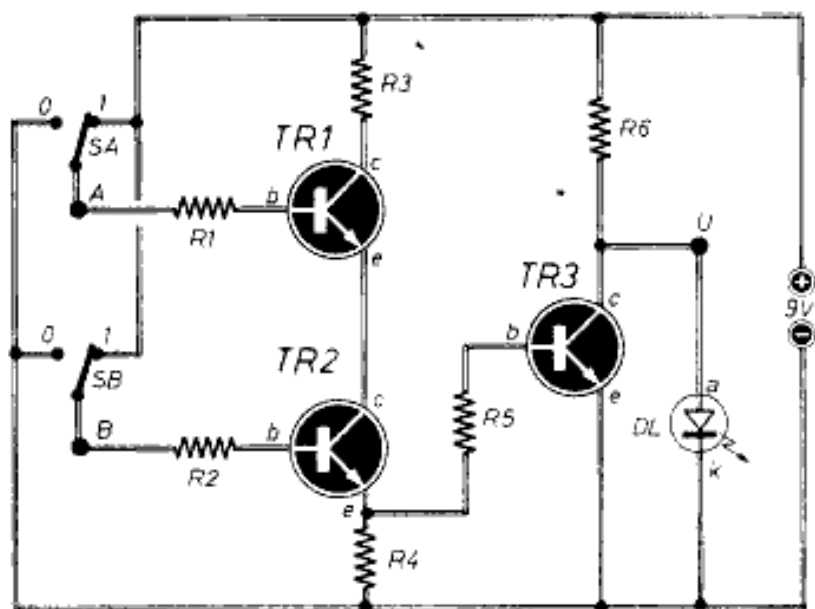
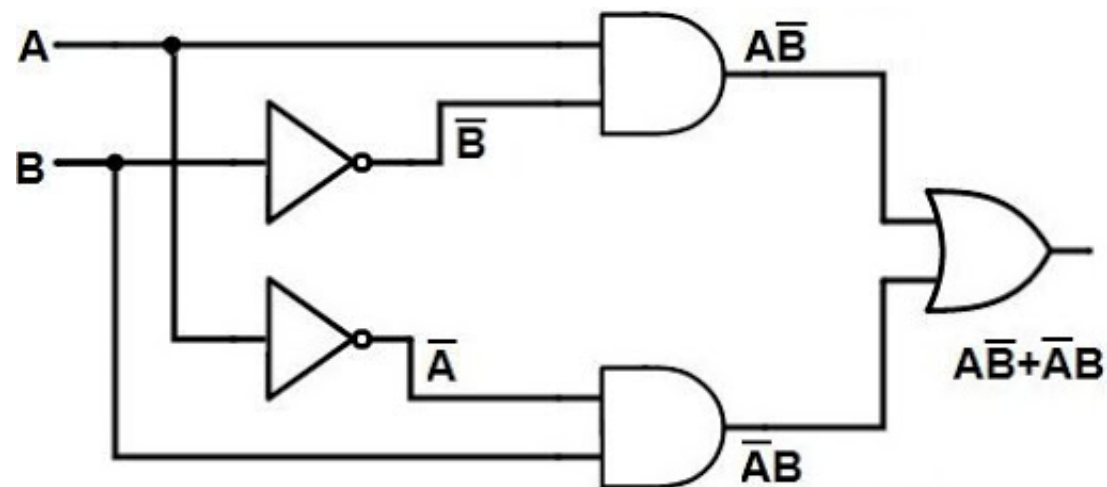


Tabella verita NAND

A	B	U
0	0	1
0	1	1
1	0	1
1	1	0



Figura tratta da [https://www.microst.it/Tutorial/funzioni\\_logiche\\_5.html](https://www.microst.it/Tutorial/funzioni_logiche_5.html)

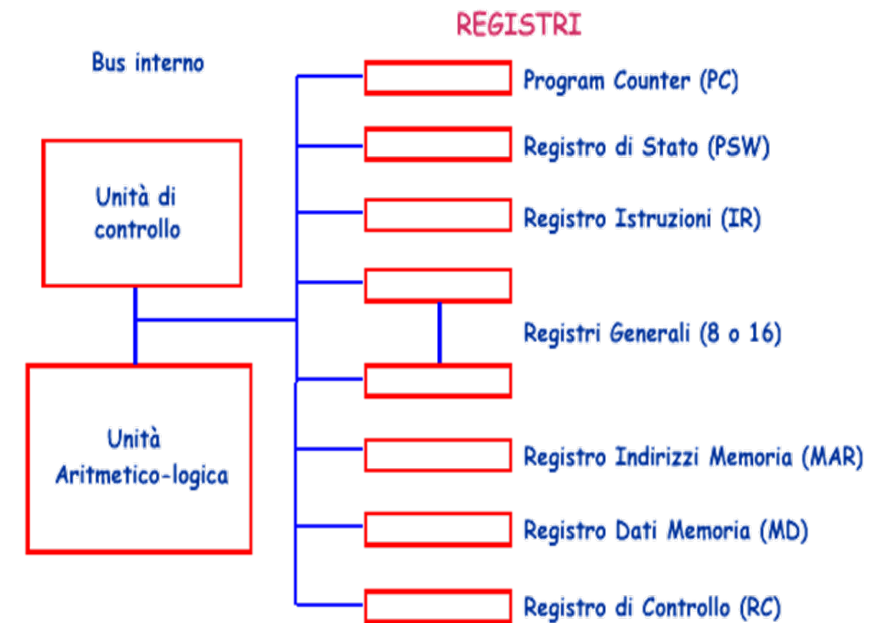


# I Registri

I Registri sono delle piccole unità di memoria locale che servono a mantenere fisicamente vicini alla ALU le informazioni che in quel momento sono essenziali per eseguire la elaborazione richiesta (i.e. metto gli operandi nei registri prima di sommarli).

Il valore di tutti i registri ad un dato istante, rappresenta lo «**stato**» della CPU in quell'istante.

- **program counter** (PC): memorizza l'indirizzo della prossima istruzione;
- **instruction register** (IR): contiene l'istruzione da eseguire;
- **memory address register** (MAR): contiene l'indirizzo della locazione di memoria da leggere o scrivere;
- **memory data register** (MDR o MD): serve a scambiare dati tra la CPU e la memoria;
- **program status word** (PSW): è il registro dei flag che memorizza particolari condizioni (riporto, overflow, risultato +/-, controllo parità);
- **control register** (RC) è il registro per i segnali di controllo (ready, wait, write, read,...).





# La Control Unit

L'ultima componente della CPU che funge da vero e proprio cervello del nostro computer è la Control Unit.

Cosa fa la Control Unit?

Essa svolge la delicata operazione di «interpretare» l'istruzione ricevuta e di impartire i conseguenti «ordini» a tutti gli organi coinvolti: ALU, Registri, Unità di Uscita etc.. affinché l'istruzione sia di fatto eseguita.

L'unità di controllo ad ogni clock riceve in Input:

1. il codice operativo dell'istruzione da eseguire
2. i segnali di flag che determinano lo stato della CPU
3. i segnali di controllo (interrupts, acknowledgments)

E produce in Output:

1. Segnali di controllo verso ALU, di trasferimento dati, di controllo verso altri dispositivi

# La CPU – Instruction set

- Come nelle lingue naturali, composte da fonemi, parole e frasi e così via, anche nel linguaggio dei computer, il **linguaggio macchina**, i simboli dell'alfabeto utilizzato (1 e 0) sono organizzati in «**parole**» e queste ultime possono essere utilizzate per comporre «**frasi**» secondo delle ben precise regole grammaticali.
  - Le frasi del linguaggio macchina sono dette «istruzioni»
  - Ogni istruzione ordina al processore di eseguire un'azione elementare che va a modificare lo stato interno del computer:
    - la lettura di una locazione di memoria
    - l'esecuzione della somma dei valori contenuti in due registri
    - l'incremento di un contatore
    - Il salto a una successiva istruzione
- .....

# La CPU - Instruction Set

- La programmazione del microprocessore normalmente avviene attraverso un linguaggio intermedio, rispetto al linguaggio macchina, detto **Linguaggio Assembly**. Le **istruzioni** in Assembly sono generalmente costituite da una parola iniziale detta **opcode (operation code)**, che specifica l'operazione da eseguire, seguita da altre parole che specificano gli eventuali **operandi** (o dati) a cui l'azione deve essere applicata (per esempio «somma 10 e 5»).
- Il formato e la sintassi delle operazioni sono proprie di ciascun processore e sono contenute nel **Instruction Set** di quel dato processore, appunto.
- Per completezza diciamo che, mentre i processori di un tempo (scalari) potevano eseguire una sola operazione per volta nello stesso ciclo di clock, quelli attuali sono spesso in grado di eseguire più operazioni simultanee sullo stesso processore (i.e. nei processori superscalari per il calcolo parallelo) o più operazioni su più processori integrati in un unico chip (multi core).

# Come funziona la CPU?

Come detto, la CPU coordina tutte le operazioni svolte dal computer; queste possono essere:

## **Funzioni interne alla CPU, come:**

- Interpretare le istruzioni del programma
- Effettuare operazioni logiche/aritmetiche
- Generare i segnali di controllo per i componenti interni
- Verificare la correttezza dei dati
- Sincronizzare le funzioni del sistema

## **Funzioni esterne alla CPU**

- Operare sulla main memory (lettura e scrittura)
- Attivare componenti esterni
- Ricevere segnali da componenti esterni

Vediamo ora cosa fa una CPU quando esegue una istruzione (ciclo macchina).

# Il ciclo macchina

Il funzionamento della CPU si basa sul seguente ciclo di interpretazione del linguaggio macchina:

**FETCH dell'istruzione:** il processore preleva l'istruzione dalla memoria centrale, all'indirizzo specificato dal registro **Program Counter** (PC) dopodiché il PC viene predisposto per la successiva istruzione.

**DECODE dell'istruzione:** una volta che l'istruzione è stata prelevata, la Control Unit determina di quale istruzione si tratti, quale operazione debba essere eseguita, come ottenere gli operandi.

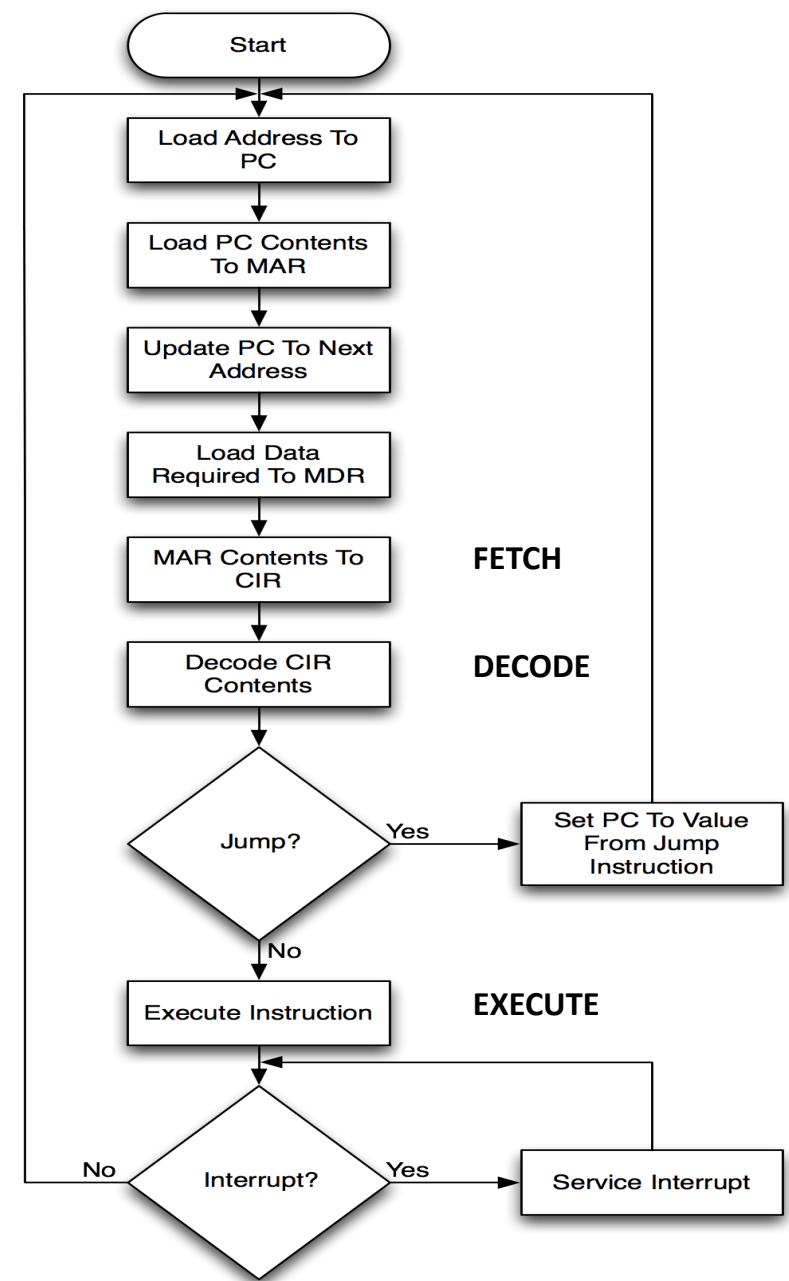
**EXECUTE dell'istruzione:** l'operazione viene eseguita. La Control Unit genera i segnali dovuti. Se l'istruzione è un salto condizionale viene di nuovo aggiornato il PC.

**WRITEBACK:** si ritorna all'inizio del ciclo e ci si dispone per la successiva istruzione.

Il numero di cicli che il processore è in grado di svolgere in un secondo, ne determina la velocità. La velocità è anche determinata dall'architettura del processore (16, 32, 64 bit). Nei nostri computer di casa le velocità di elaborazione si aggirano normalmente intorno a qualche GigaHertz.

# Esempio di ciclo macchina

1. Si acquisisce l'indirizzo corrente dalla memoria - in base all'indirizzo del Program Counter (PC) - e lo si inserisce nel Registro degli Indirizzi di Memoria (MAR)
2. Si incrementa il PC per predisporre per la successiva istruzione
3. Se serve prelevare un dato dalla memoria questo viene caricato nell'apposito registro MDR. Questa eventuale operazione (sulla memoria) viene eseguita al ciclo successivo
4. Il contenuto di MAR va nel Registro delle Istruzioni (IR)
5. La Control Unit decodifica il tipo di istruzione contenuta in IR
6. Se l'istruzione è un «salto» si aggiorna PC e si torna allo Start
7. Altrimenti viene eseguita l'istruzione: la Control Unit genera una serie di segnali che realizzano l'istruzione
8. Si verifica l'esistenza di Interrupt ed eventualmente si cede il controllo
9. Si torna allo Start (o si finisce) completando il «**ciclo macchina**».

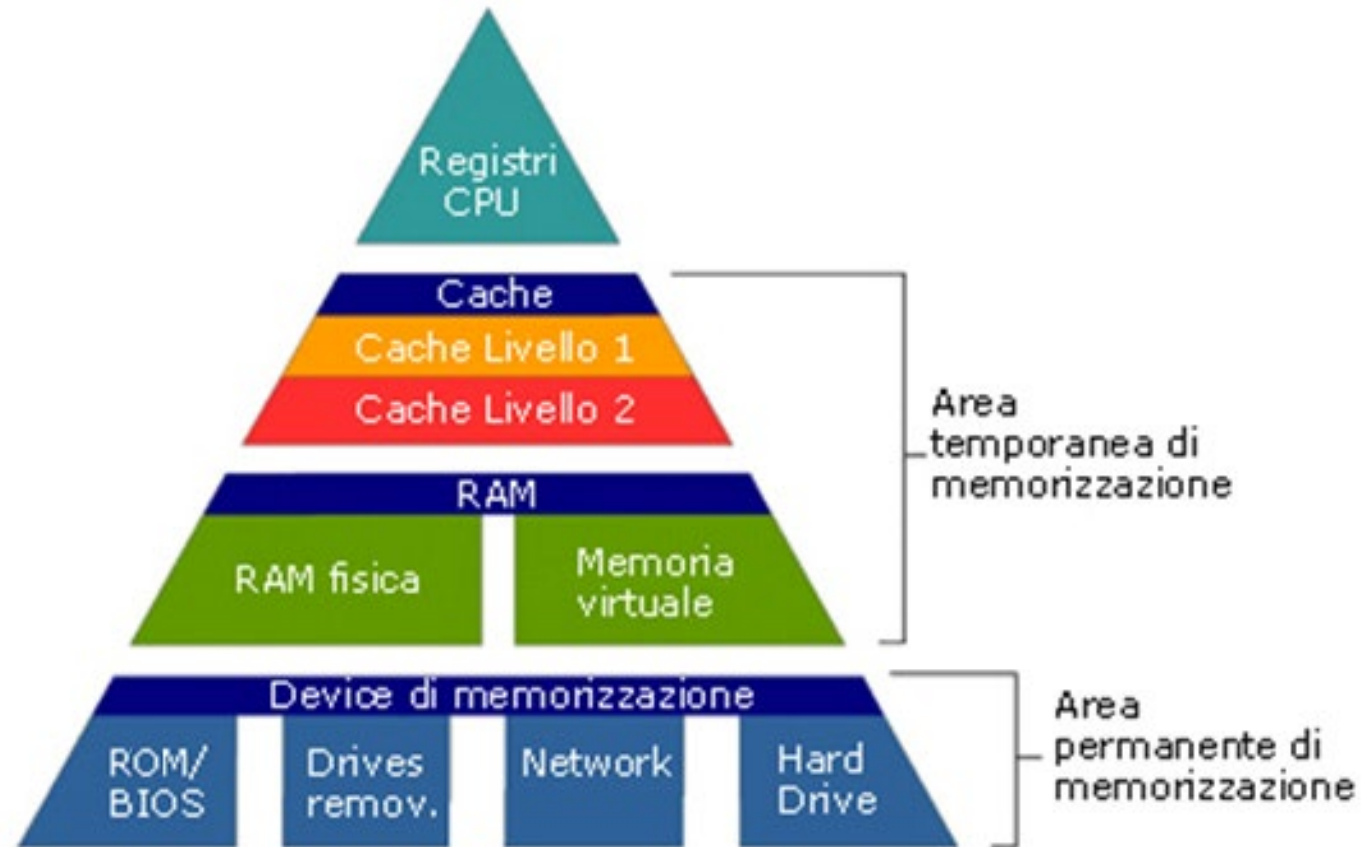


# La Memoria del Computer

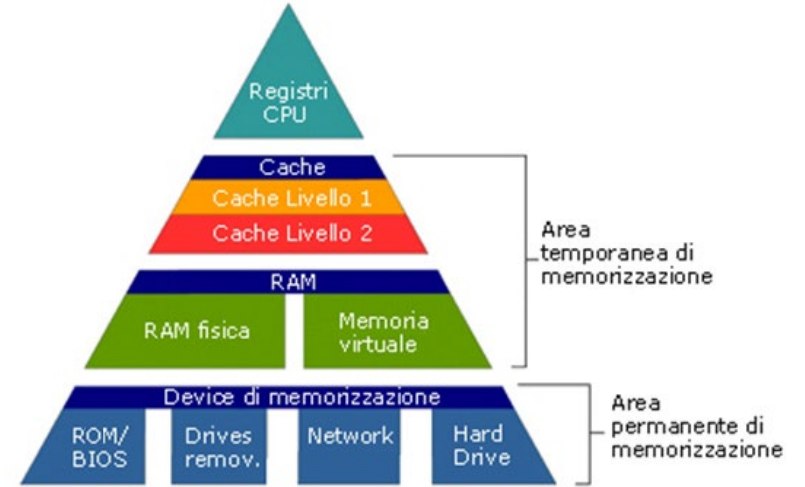
Come abbiamo visto, per poter lavorare la CPU sempre la necessità di interagire con la Memoria centrale. Queste due componenti lavorano in simbiosi per un motivo molto semplice:

- Sia i dati, sia i programmi risiedono nella Memoria centrale (architettura Von Neumann)
- Principale compito della CPU è quello di eseguire le istruzioni di un programma presente in memoria centrale leggendo e scrivendo i dati, che pure risiedono in memoria centrale.

La memoria del computer tuttavia non è data solo dalla memoria centrale ma anche da altri tipi di memoria. Vediamo quali:



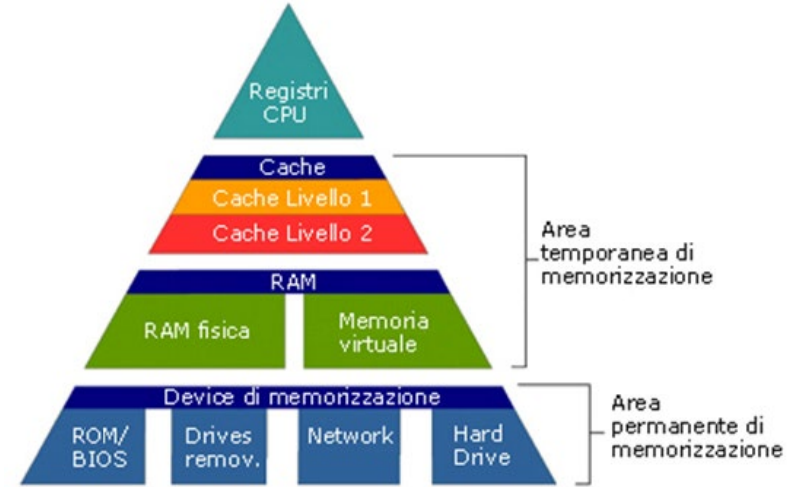
# La Memoria del Computer



- **Memoria RAM** (Random Access Memory) - la RAM è una memoria volatile utilizzata per archiviare i dati temporaneamente. È la memoria principale del computer ed è utilizzata per eseguire programmi e applicazioni. La RAM è molto veloce, ma ha una capacità limitata e richiede un'alimentazione costante per mantenere i dati memorizzati.
- **Memoria ROM** (Read-Only Memory) - la ROM è una memoria non volatile utilizzata per archiviare i dati permanentemente. La ROM è spesso utilizzata per archiviare il Firmware del computer, che consente al sistema operativo di avviarsi. La ROM non può essere modificata, ma può essere letta direttamente dal processore.
- **Memoria cache** - la cache è una memoria veloce utilizzata per archiviare temporaneamente i dati a cui il processore accede frequentemente e ai quali è più probabile che debba accedere ai passi successivi. La cache aiuta a migliorare le prestazioni del computer riducendo i tempi di accesso ai dati. Ci sono diversi livelli di cache nel computer, a partire dal livello 1 che ha la capacità più piccola ma la maggiore velocità.
- **Memoria di massa** - la memoria di massa o memoria di archiviazione è utilizzata per archiviare i dati permanentemente, anche quando il computer è spento. La memoria di archiviazione può essere interna, come il HDD o lo SSD, o esterna, come le unità flash USB o i dischi rigidi esterni. La memoria di archiviazione ha una capacità maggiore rispetto alla RAM, ma ha una velocità di accesso più lenta.



# La Memoria del Computer



- **La RAM virtuale** è un'area di memoria di archiviazione temporanea utilizzata dal sistema operativo del computer per ampliare la capacità della memoria RAM fisica del sistema. In altre parole, la RAM virtuale permette al computer di utilizzare una porzione dello spazio di archiviazione sul disco rigido come se fosse RAM fisica, estendendo così la memoria disponibile per l'esecuzione dei programmi.
- Il sistema operativo utilizza la RAM virtuale in modo dinamico, ovvero quando la memoria RAM fisica del sistema è esaurita, il sistema operativo sposta alcune parti dei dati dal RAM fisico alla RAM virtuale per fare spazio ai nuovi dati. Ciò significa che l'uso della RAM virtuale può rallentare le prestazioni del computer rispetto alla RAM fisica, poiché il disco rigido ha una velocità di accesso molto minore rispetto alla RAM.
- Tuttavia, l'utilizzo della RAM virtuale è utile per evitare che il sistema operativo esaurisca completamente la memoria RAM fisica, cosa che potrebbe causare l'interruzione di programmi e il blocco del sistema. Inoltre, la RAM virtuale può essere impostata manualmente per consentire ai programmi di utilizzare più memoria rispetto a quella fisicamente disponibile, ma ciò potrebbe anche causare una riduzione delle prestazioni complessive del sistema.

## Oltre la CPU

Come abbiamo visto, la CPU esegue una ad una le istruzioni affidatele dal programma utilizzando i dati che le sono stati forniti in ingresso (tramite la Tastiera, il mouse, il microfono, il disco fisso etc..) e che ora risiedono in memoria centrale. L'esecuzione delle istruzioni comporta la generazione di risultati verso i dispositivi di output.

Per esempio, quando scriviamo una lettera con un programma di videoscrittura possiamo controllare a **Monitor** il risultato di ciò che abbiamo scritto e, una volta che abbiamo terminato, archiviare definitivamente il risultato (la lettera in questo caso) in una memoria di massa - in un **File** - e poi se vogliamo anche inviarlo a una **Stampante**.

Tutte le operazioni elementari eseguite dalla CPU, che sottintendono quelle di più alto livello appena citate, non riusciamo a percepirle perché operiamo nei confronti del computer mediante una **serie di comandi** che rendono trasparente sia l'architettura del computer stesso, sia le modalità di esecuzione dei programmi.

Per ottenere questo risultato è necessario operare tramite un software che gestisca le risorse hardware e software del computer, consentendo la comunicazione tra le varie parti del sistema e fornendo un'interfaccia utente per l'interazione con il computer stesso.

# Il Sistema Operativo

# Il Sistema Operativo

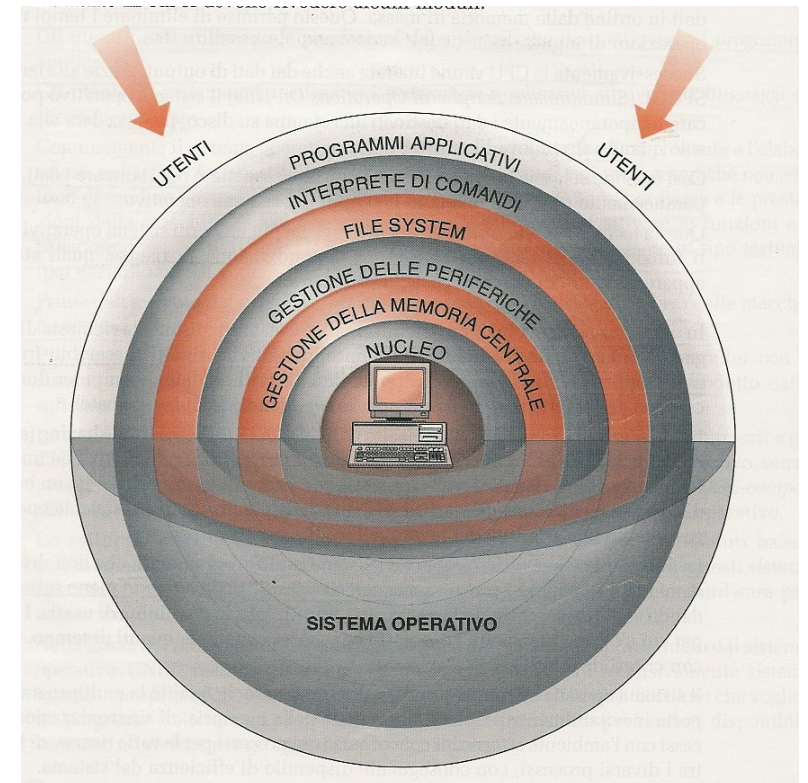
Cos'è dunque che rende trasparente all'utilizzatore tutte queste operazioni?

Cosa fa in modo che io possa disinteressarmi di come archiviare un file o di come stampare un documento o di come gestire l'attivazione di un dispositivo con priorità real time?

Si tratta del

«**SISTEMA OPERATIVO**»

una componente fondamentale del computer che però questa volta non è di tipo Hardware, ma di tipo Software.



# Il Sistema Operativo

Analogamente al cervello umano che col sistema nervoso governa il funzionamento delle attività del corpo, il **Sistema Operativo** supervisiona le attività della macchina in cui si trova e di tutte le sue componenti: la memoria di massa, la memoria centrale, il processore, le schede accessorie, il lettore cd, la stampante, il mouse, la tastiera etc.

Il Sistema Operativo permette all'utente di interfacciarsi facilmente al sistema hardware, evitandogli una interazione diretta che sarebbe alquanto difficoltosa anche per un tecnico.

Esistono diversi tipi di sistemi operativi. I più comuni sono Windows, MacOS e Linux per i computer e IOS e Android per i dispositivi *mobile*.

Ogni sistema operativo ha la sua propria interfaccia utente, il suo set di strumenti di gestione e le sue funzionalità specifiche.

# Il Sistema Operativo

Poiché un Sistema Operativo deve essere in grado di gestire, trattare e comunicare con la macchina in maniera trasparente ed indipendente da essa, è intuibile che più il software viene realizzato in simbiosi con l'hardware, maggiore sarà la sua specializzazione e maggiore sarà la sua efficienza.



Apple, per esempio, progetta sia l'hardware che il software per i propri dispositivi, il che significa che questi possono essere ottimizzati l'uno per l'altro. Ciò consente ai sistemi operativi Apple di funzionare in modo più efficiente rispetto ai sistemi operativi che devono supportare un'ampia varietà di hardware di terze parti

# Il Sistema Operativo

- Un Sistema Operativo ha come fondamentale compito quello di coordinare, sincronizzare, gestire ed eseguire dei **processi** ossia delle sequenze di attività (**task**) che il processore deve portare a termine su richiesta dell'utente.
- Per realizzare questi task il Sistema Operativo ha bisogno di poter utilizzare delle **risorse**: sono risorse, per esempio, il processore stesso, le diverse memorie, centrali o di massa, il monitor, la tastiera e così via. Anche una operazione semplicissima come quella di premere un tasto della tastiera visualizzando il carattere corrispondente sul monitor, comporta l'esecuzione di diversi processi e l'allocazione di svariate risorse.



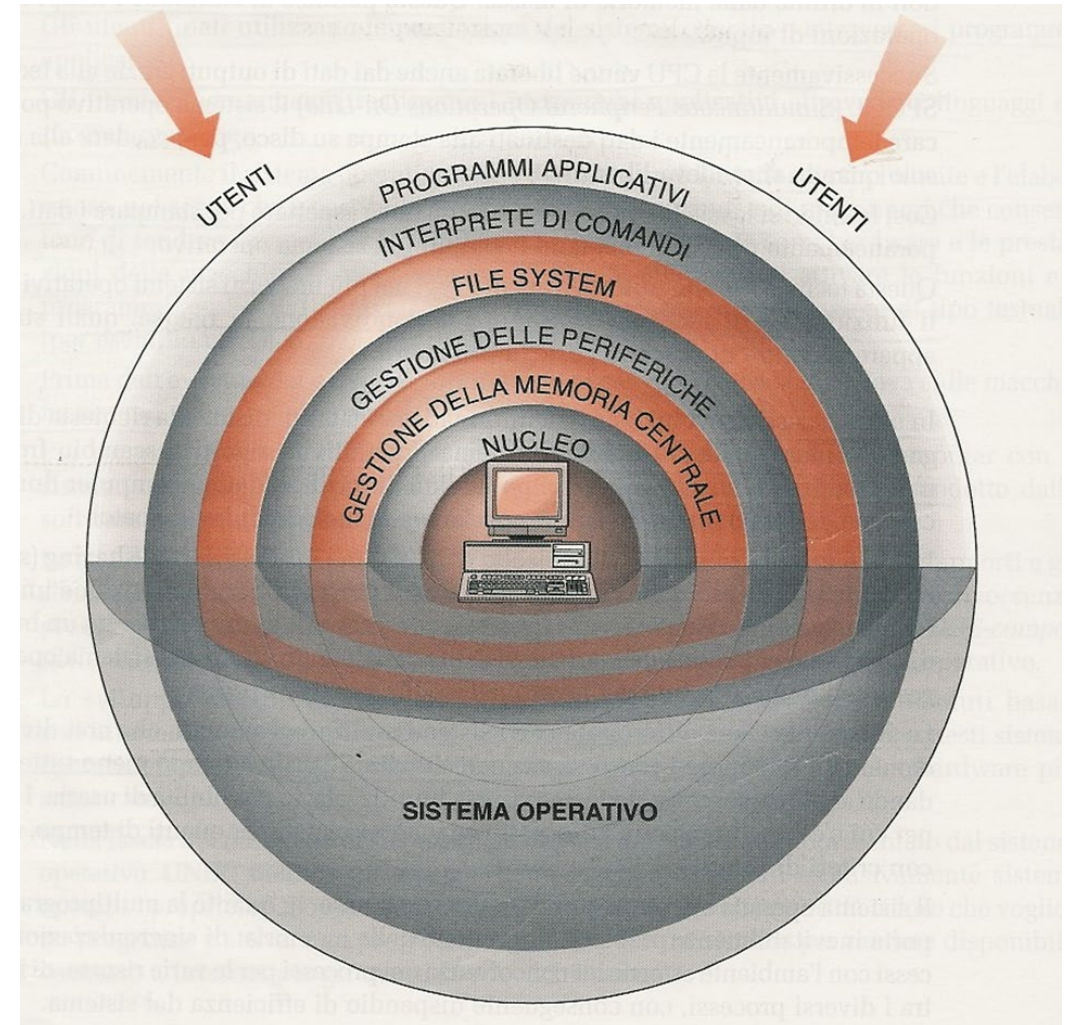
# Alcuni compiti del Sistema Operativo



# La gestione del Hardware

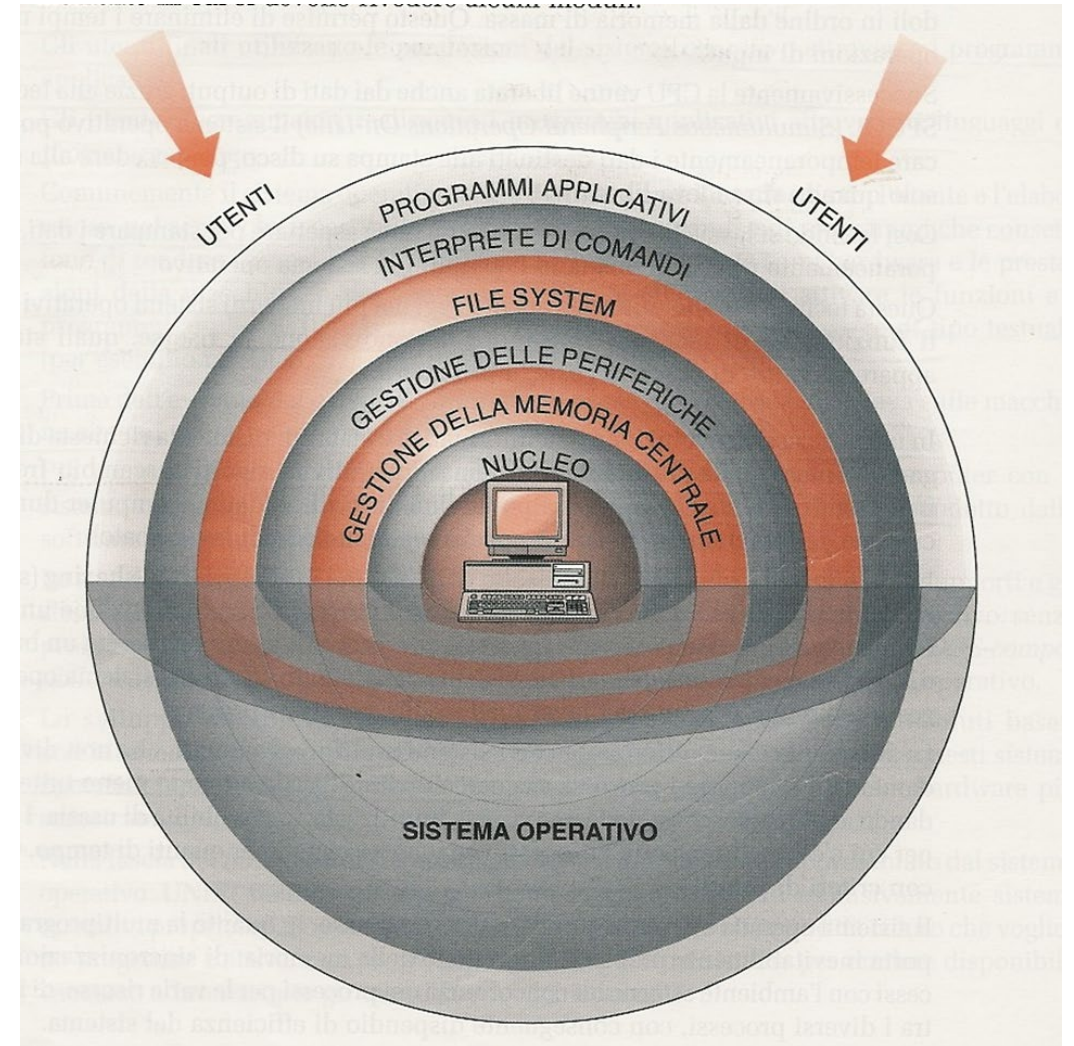
Insieme coi Driver delle periferiche e col Firmware, il **Kernel** fornisce il livello più elementare di controllo su tutti i dispositivi hardware del computer. Alcune delle funzioni del Kernel sono descritte nel seguito:

1. L'esecuzione di un programma applicativo attraverso la creazione di un processo in cui il Kernel:
  - assegna al programma lo spazio di memoria e le risorse richieste
  - stabilisce la priorità del processo rispetto ad altri processi nei sistemi multi-tasking (in grado di eseguire più programmi contemporaneamente)
  - carica il codice binario del programma nella memoria
  - avvia l'esecuzione del programma applicativo
  - interagisce con l'utente e coi dispositivi hardware.



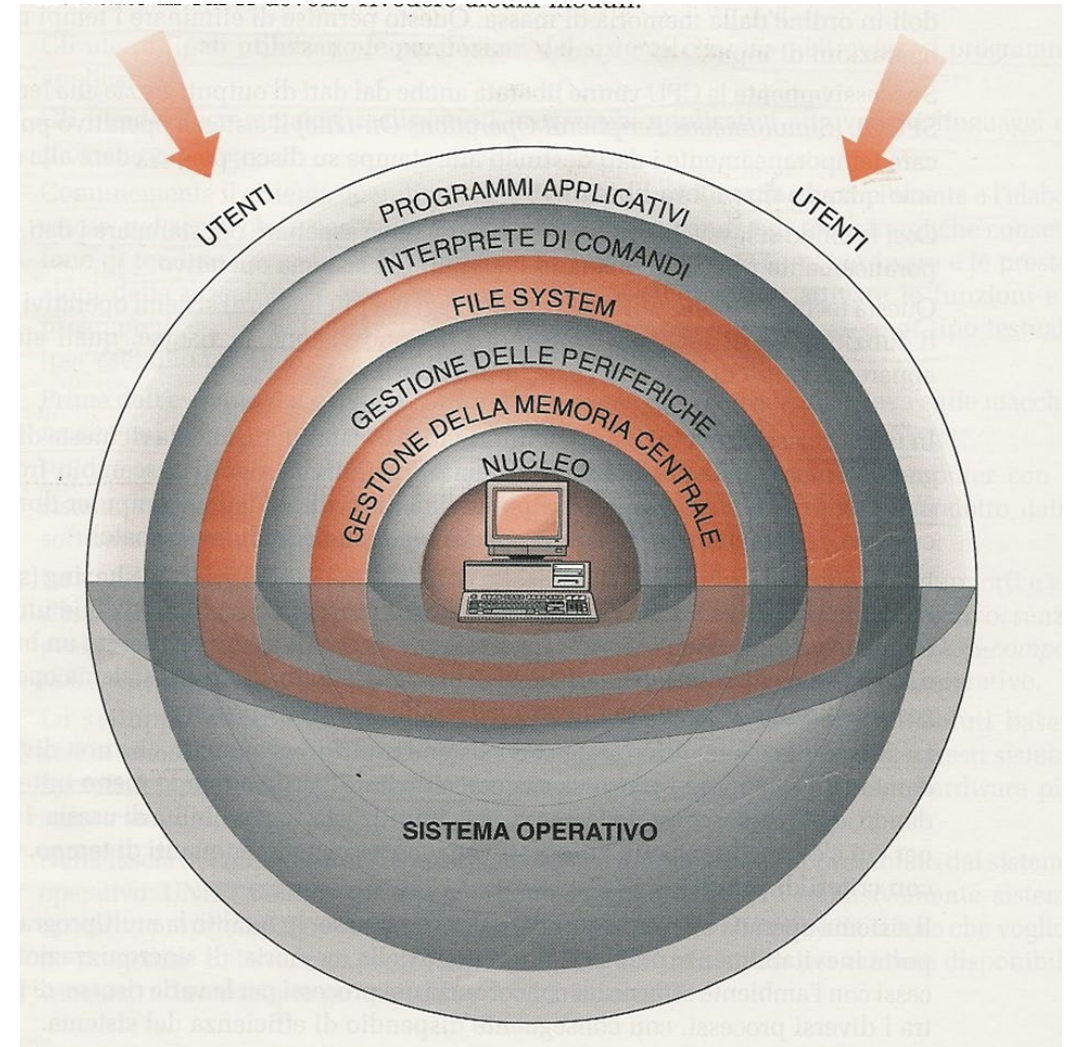
# La gestione del Hardware

2. Il Kernel del sistema operativo è responsabile della gestione di tutta la memoria di sistema utilizzata dai programmi. Questo assicura che un programma non interferisca con la memoria già in uso da un altro programma. Poiché il tempo viene condiviso dai programmi, ognuno di essi deve avere un accesso indipendente alla memoria.
3. Il Kernel può inoltre ottimizzare la **gestione delle memorie** attraverso delle tecniche di virtualizzazione; in tal modo esso può decidere quale memoria concedere ad un dato programma e può migliorarne l'impiego attribuendo delle porzioni condivise di memoria per l'esecuzione di più compiti.



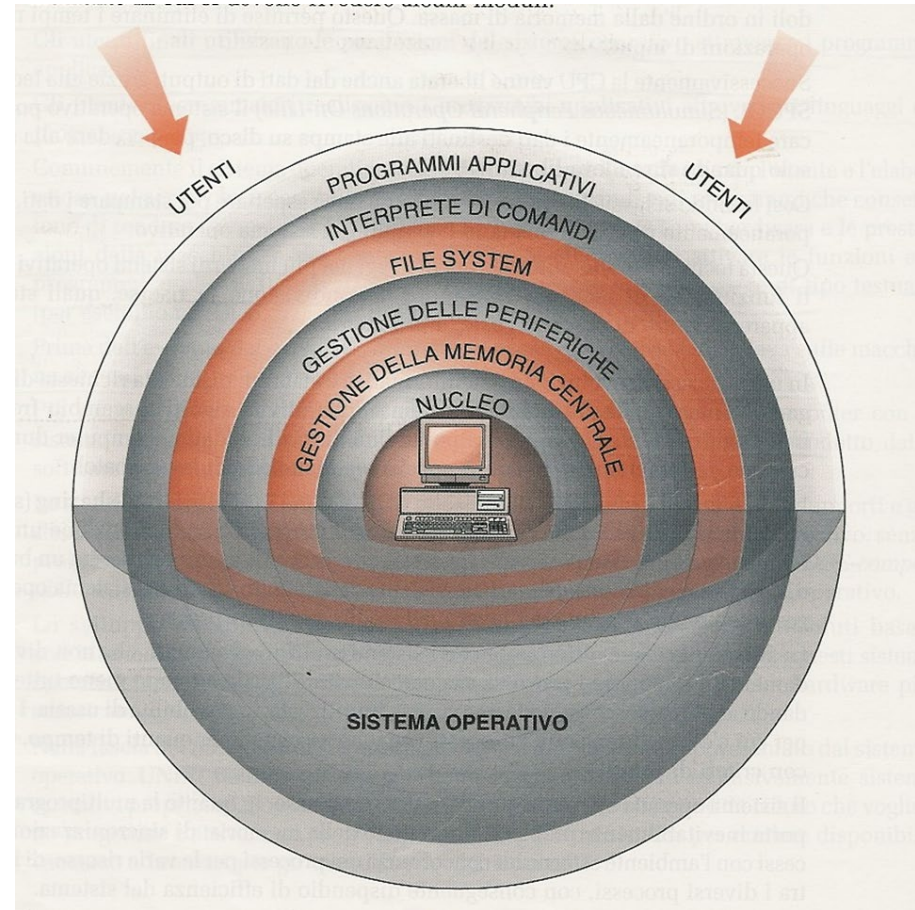
# La gestione dei Processi

- All'interno del Kernel può anche essere effettuata la gestione dei processi. Questa viene realizzata dallo **Scheduler**, un programma che scandisce il tempo di esecuzione dei vari processi e assicura che ciascuno di essi venga eseguito per il tempo richiesto.
- Normalmente lo scheduler gestisce anche lo stato dei processi e può sospendere l'esecuzione nel caso questi siano in attesa senza fare nulla, assegnando le risorse inutilizzate ad altri processi che ne hanno bisogno (esempio classico è la richiesta di dati da disco).
- Nei sistemi operativi realtime lo scheduler si occupa anche di garantire una *timeline*, cioè un tempo massimo di completamento per ciascun task in esecuzione, ed è notevolmente più complesso.



# La gestione della memoria centrale

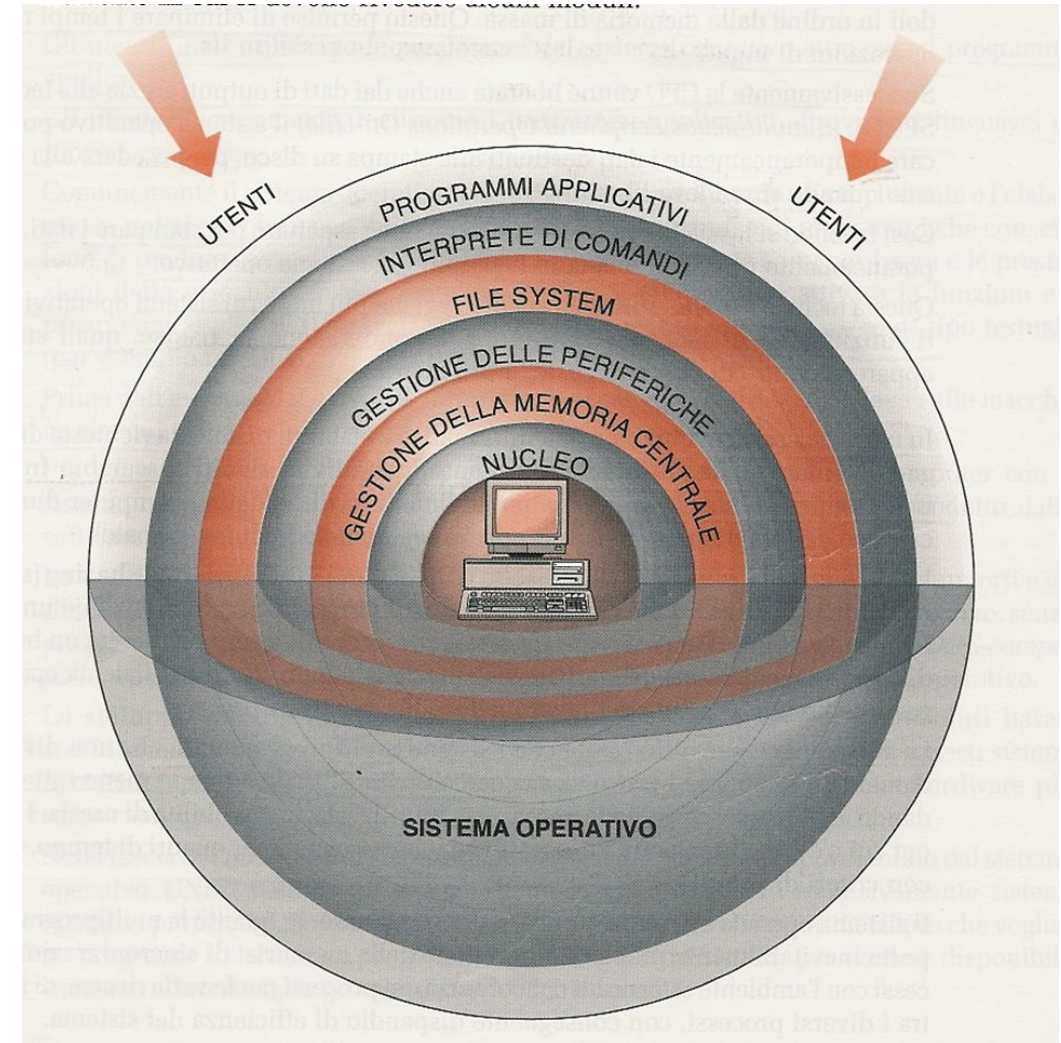
- A livello hw ogni sistema è equipaggiato con un unico spazio di memoria accessibile direttamente da CPU e dispositivi
- Nella gestione generale della Memoria Centrale i compiti del Sistema Operativo sono, in linea di massima:
  - allocare uno spazio di memoria a ciascun processo
  - deallocare la memoria di un processo concluso
  - proteggere il sistema separando gli spazi di indirizzi associati ai diversi processi in modo che non interferiscano ne tra loro ne col supervisore
  - realizzare i collegamenti tra gli indirizzi logici specificati dai processi e le corrispondenti locazioni nella memoria fisica che possono essere anche non contigue
  - gestire la memoria virtuale, ossia gli spazi di indirizzi logici di dimensioni superiori allo spazio fisico, da allocare sulla memoria di massa



Nome	Stato	1% CPU	24% Memoria	2% Disco	0% Rete	1% GPU	Motore GPU	Consumo elett...	Tendenza cons...
<b>Applicazioni (2)</b>									
> Gestione attività		0,2%	27,1 MB	0 MB/s	0 Mbps	0%		Molto basso	
> Microsoft PowerPoint		0%	119,4 MB	0 MB/s	0 Mbps	0%		Molto basso	
<b>Processi in background (97)</b>									
ACMON (32 bit)		0%	0,8 MB	0 MB/s	0 Mbps	0%		Molto basso	
> Adobe Acrobat Update Service ...		0%	0,7 MB	0 MB/s	0 Mbps	0%		Molto basso	
> Adobe Genuine Software Integri...		0%	2,2 MB	0 MB/s	0 Mbps	0%		Molto basso	
> Adobe Genuine Software Servic...		0%	1,4 MB	0 MB/s	0 Mbps	0%		Molto basso	
Adobe IPC Broker (32 bit)		0%	1,4 MB	0 MB/s	0 Mbps	0%		Molto basso	
> Adobe Update Service (32 bit)		0%	0,9 MB	0 MB/s	0 Mbps	0%		Molto basso	
> Antimalware Service Executable		0%	130,6 MB	0 MB/s	0 Mbps	0%		Molto basso	
AnyDesk (32 bit)		0%	13,8 MB	0 MB/s	0 Mbps	0%		Molto basso	
> AnyDesk (32 bit)		0%	14,2 MB	0 MB/s	0 Mbps	0%		Molto basso	
Application Frame Host		0%	11,0 MB	0 MB/s	0 Mbps	0%		Molto basso	
> Applicazione sottosistema spoo...		0%	4,9 MB	0 MB/s	0 Mbps	0%		Molto basso	
> ASLDR Service (32 bit)		0%	0,8 MB	0 MB/s	0 Mbps	0%		Molto basso	
ASUS GIFTBOX (32 bit)		0%	44,9 MB	0 MB/s	0 Mbps	0%		Molto basso	
ASUS GIFTBOX (32 bit)		0%	0,4 MB	0 MB/s	0 Mbps	0%		Molto basso	
ASUS GIFTBOX (32 bit)		0%	6,9 MB	0 MB/s	0 Mbps	0%		Molto basso	
ASUS GIFTBOX (32 bit)		0%	9,2 MB	0 MB/s	0 Mbps	0%		Molto basso	
ASUS Patch For Touch Panel		0%	0,4 MB	0 MB/s	0 Mbps	0%		Molto basso	
ATK Media (32 bit)		0%	0,9 MB	0 MB/s	0 Mbps	0%		Molto basso	

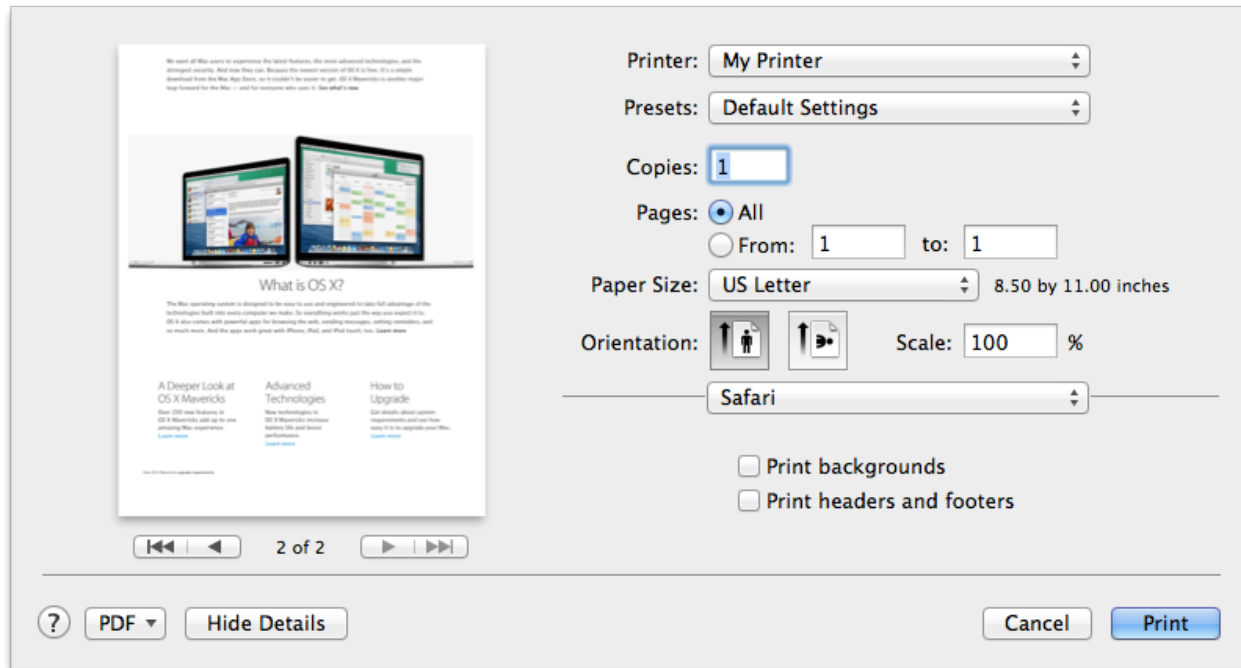
# La gestione delle Periferiche

- Altro compito fondamentale del SO è la gestione delle periferiche (stampante, mouse, tastiera, monitor etc.). Per realizzare questo compito il SO spesso utilizza del software aggiuntivo che ne diviene parte integrante e che dipende dallo specifico hardware di quel dispositivo. Questo software aggiuntivo è chiamato **Driver**.
- I driver forniscono al kernel un'interfaccia standardizzata per comunicare con i dispositivi hardware, consentendo al sistema operativo di gestire i dispositivi in modo efficiente e coerente.



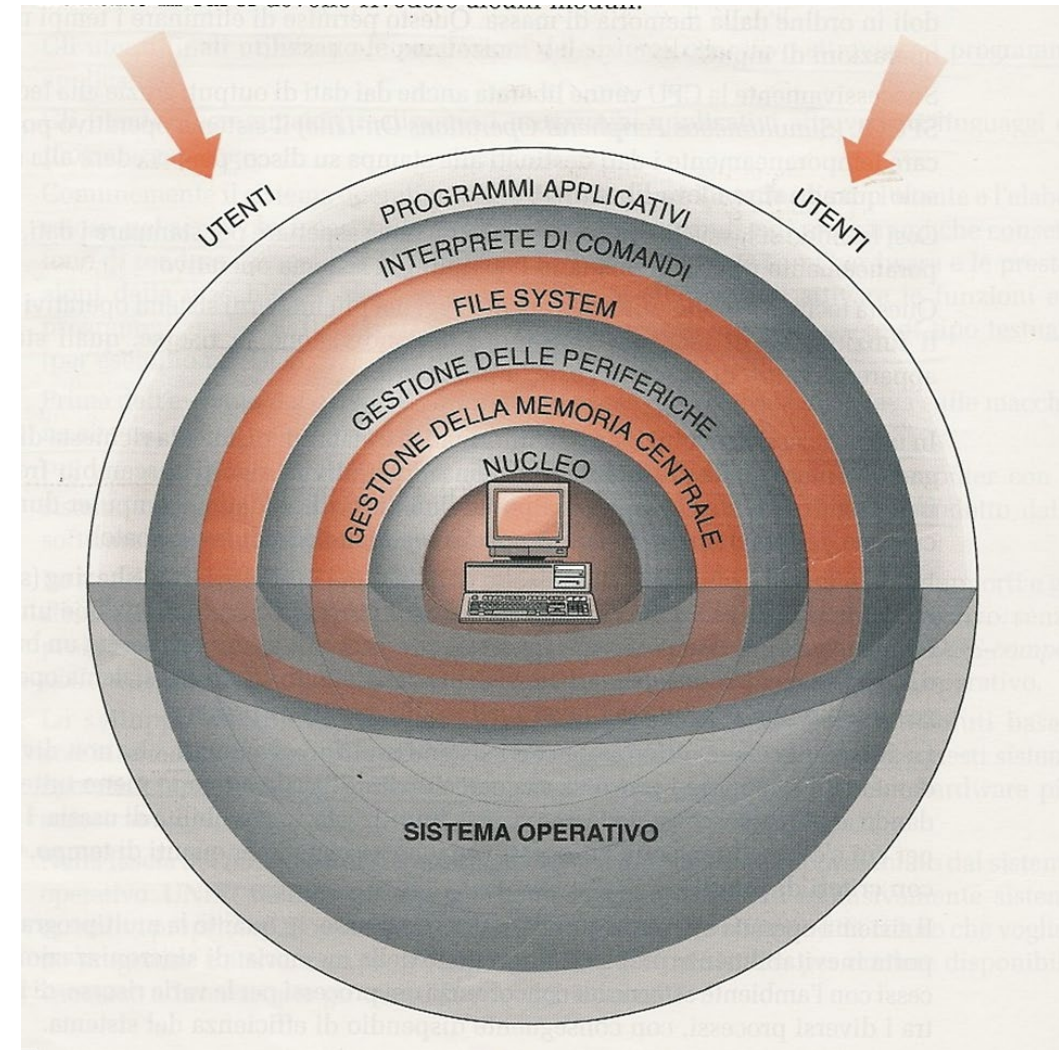
# La gestione delle Periferiche

- Talvolta, nella gestione della stampante, viene impiegato un cosiddetto **Spooler** di stampa. Questo è un componente del SO che riceve in ingresso i file da stampare dai programmi utilizzati dall'utente e li accoda mandandoli in stampa uno alla volta verso la stampante.
- Questo permette ai programmatori di applicativi di non doversi preoccupare della gestione delle code di stampa e di eventuali conflitti sulla stampante che dovessero verificarsi per via di altri processi concomitanti in quanto lo spooler accetterà qualsiasi documento e farà sembrare la stampante sempre disponibile.



# La Gestione di File e Cartelle

- Tra i diversi compiti del Sistema Operativo c'è anche quello fondamentale di gestire file e cartelle: il cosiddetto **File System**.
- In questo caso il SO si occupa di operazioni quali: archiviare e proteggere le risorse; renderle disponibili agli utenti di una rete; gestire le modifiche a tali risorse.
- Questi strumenti includono per esempio la gestione delle cartelle condivise, la crittografia dei dati e così via. Quando una cartella è condivisa, il SO si preoccupa che gli utenti che vi accedono tramite la rete siano autorizzati e che più utenti non effettuino modifiche simultanee.
- Il File System si occupa anche di come memorizzare i file all'interno del disco fisso, di come indicizzarne le varie parti e così via.





File Home Condividi Visualizza

Questo PC > Desktop > DISEGNI DEFINITIVI

Nome	Ultima modifica	Tipo	Dimensione
DESTINAZIONE NUOVA COSTRUZIONE 1...	25/01/2016 15:24	File PDF	4 KB
ELABORATO COMPARATIVO 1-200	25/01/2016 15:15	File PDF	52 KB
MODELLO ISTAT	25/01/2016 16:19	File PDF	37 KB
<b>ORIENTAMENTO FOTO</b>	24/01/2016 17:38	File PDF	39 KB
PLANIMETRIA ABITAZIONE.bak	24/01/2016 17:18	File BAK	54 KB
PLANIMETRIA ABITAZIONE	25/01/2016 15:42	AutoCAD Drawing	54 KB
planimetria di inquadramento 1-200 CO...	24/01/2016 17:38	File BAK	91 KB
planimetria di inquadramento 1-200 CO...	25/01/2016 15:16	AutoCAD Drawing	91 KB
PLANIMETRIA I PIANO 1-100	25/01/2016 15:38		
PLANIMETRIA PIANO TERRA 1-100	25/01/2016 15:40		
plot	25/01/2016 15:40		
STATO DI FATTO 1-200	24/01/2016 16:02		
STATO DI PROGETTO 1-200	24/01/2016 16:04		

Proprietà - ORIENTAMENTO FOTO

Generale Sicurezza Dettagli Versioni precedenti

ORIENTAMENTO FOTO

Tipo di file: File PDF (.pdf)

Apri con: Microsoft Edge [Cambia...](#)

Percorso: C:\Users\VincenzoG\Desktop\DISEGNI DE

Dimensioni: 38.0 KB (39.010 byte)

Dimensioni su disco: 40.0 KB (40.950 byte)

Data creazione: domenica 24 gennaio 2016, 17:38:08

Ultima modifica: domenica 24 gennaio 2016, 17:38:09

Ultimo accesso: domenica 24 gennaio 2016, 17:38:09

Attributi:  Sola lettura  Nascosto [Avanzate...](#)

OK Annulla Applica

# L'interfaccia Utente

L'interfaccia utente del sistema operativo (**SHELL**) è la parte del sistema operativo che consente all'utente di interagire con il sistema. Ci sono diverse tipologie di interfaccia utente, tra cui l'interfaccia a riga di comando (Command Line Interface) e l'interfaccia grafica (Graphic User Interface). L'interfaccia utente permette all'utente di eseguire le applicazioni, accedere ai file e alle cartelle, modificare le impostazioni del sistema e interagire con altri utenti.



GUI di Windows 10

# L'interfaccia Utente



GUI di OS X Mountain Lion

# L'interprete di Comandi

- la CLI (Command Line Interface) è un'interfaccia utente basata su testo che consente agli utenti di interagire con il sistema operativo e le applicazioni tramite la digitazione di comandi su una riga di comando, anziché utilizzare un'interfaccia grafica.
- La CLI è molto più flessibile della GUI e consente più possibilità di interazione, tuttavia è più complessa da utilizzare e per questo viene utilizzata soprattutto dagli utenti esperti e dagli amministratori di sistema per eseguire operazioni avanzate e automatizzare i processi di sistema.
- Per esempio: **find /path/to/directory -name "\*.txt" -type f -print0 | xargs -0 grep "search\_term"** - questo comando su Linux cerca tutti i file con estensione ".txt" all'interno della directory specificata e tutte le sue sottodirectory. Successivamente, cerca all'interno di questi file una parola o una stringa specificata da "search\_term".

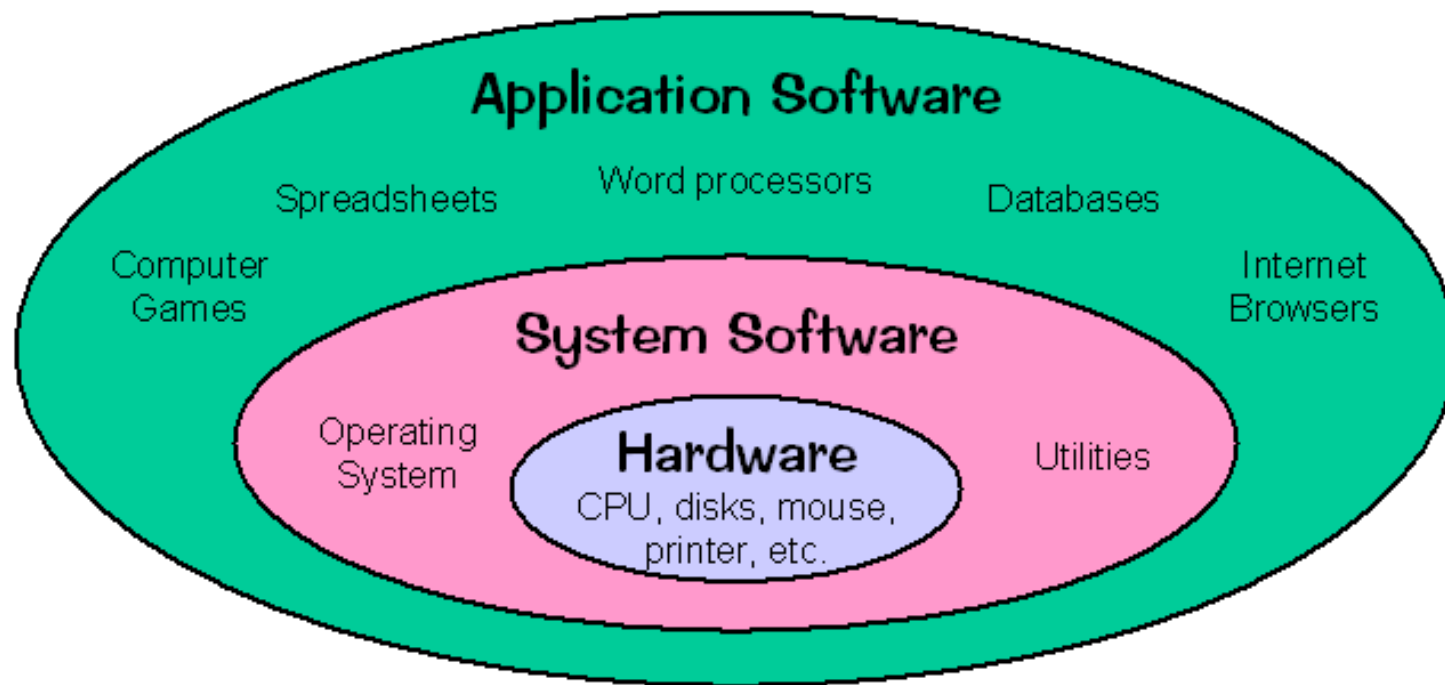
```
Prompt dei comandi
Microsoft Windows [Versione 10.0.14393]
(c) 2016 Microsoft Corporation. Tutti i diritti sono riservati.

C:\Users\VincenzoG>dir
Il volume nell'unità C non ha etichetta.
Numero di serie del volume: CCA9-42C6

Directory di C:\Users\VincenzoG

13/03/2017  23:54    <DIR>          .
13/03/2017  23:54    <DIR>          ..
08/01/2016  17:20    <DIR>          .ITtrading
25/05/2016  18:59    <DIR>          .matplotlib
09/08/2016  10:14    <DIR>          .oracle_jre_usage
25/05/2016  18:58    <DIR>          .qgis2
16/01/2017  15:26    <DIR>          Contacts
14/03/2017  07:26    <DIR>          Creative Cloud Files
15/03/2017  09:18    <DIR>          Desktop
19/12/2016  16:30    <DIR>          dikeliteTmpdir
03/03/2017  17:20    <DIR>          Documents
15/03/2017  09:17    <DIR>          Downloads
23/09/2016  16:38    <DIR>          Dropbox
08/01/2016  15:58    <DIR>          EfiRedattore
16/01/2017  15:26    <DIR>          Favorites
16/01/2017  15:26    <DIR>          Links
16/01/2017  15:26    <DIR>          Music
15/03/2017  08:55    <DIR>          OneDrive
16/01/2017  15:26    <DIR>          Pictures
14/11/2016  09:04                1.068 pixinsight-license
14/03/2017  16:38                74 pslog.txt
```

Dal software  
di sistema al  
software  
applicativo



# Software applicativo

- A differenza del sistema operativo e delle sue componenti (la maggior parte delle quali si trovano preinstallate al momento dell'acquisto), una applicazione non è considerata necessaria al funzionamento del computer per cui, genericamente, si può parlare di:
  - **software di base**, ossia tutte le istruzioni fondamentali di cui un computer ha bisogno per funzionare e per far funzionare il software applicativo
  - **software applicativo**, che fornisce all'utente finale determinate funzionalità specialistiche e di utilità, nei più svariati settori: scientifico, ufficio, business, gioco etc.
- Il software applicativo può essere sviluppato in maniera dedicata per un dato sistema operativo, oppure essere multiplatforma se contiene delle estensioni che ne consentono l'utilizzazione su più sistemi operativi.

# Software applicativo

In informatica, il termine **Software Applicativo** si riferisce a qualsiasi programma o insieme di programmi che svolgono una specifica funzione o un insieme di funzioni, che possono essere utilizzate dagli utenti per svolgere compiti specifici su un computer o su un altro dispositivo.

Le applicazioni software possono essere classificate in base alla loro funzione, alla loro piattaforma di destinazione, alla loro licenza o ad altri fattori. Ad esempio, le applicazioni software possono essere suddivise in:

- **Applicazioni di produttività:** queste applicazioni includono programmi di elaborazione testi, fogli di calcolo, presentazioni e altri strumenti utili per la creazione e la gestione di documenti e dati.
- **Applicazioni di comunicazione:** queste applicazioni comprendono programmi di posta elettronica, messaggistica istantanea, videoconferenza e altri strumenti per la comunicazione in tempo reale o asincrona tra gli utenti. Per esempio: Zoom, Microsoft Teams, WhatsApp, Telegram, Skype.
- **Applicazioni di intrattenimento:** queste applicazioni includono giochi, software di editing multimediale, applicazioni musicali e altre applicazioni per il tempo libero e l'intrattenimento. Per esempio: Netflix, YouTube, Spotify, Apple Music, Google Play Music.

# Software applicativo

- **Applicazioni web:** queste applicazioni sono eseguite all'interno di un browser web e possono includere programmi di posta elettronica online, social network, servizi di cloud computing e altri strumenti per l'elaborazione e lo scambio di informazioni online.
- **Applicazioni di grafica:** programmi per il disegno, la modellazione 3D, il fotoritocco e l'elaborazione video. Questi programmi sono utilizzati in diversi settori, tra cui la grafica pubblicitaria, la produzione cinematografica e televisiva, l'architettura e l'ingegneria. Esempi di applicazioni di grafica includono Adobe Photoshop, Adobe Illustrator, Autodesk 3ds Max, Autodesk Maya, Cinema 4D, Blender e Final Cut Pro.
- **Applicazioni di calcolo:** Le applicazioni di calcolo sono progettate per eseguire calcoli complessi su grandi quantità di dati. Questi programmi sono utilizzati in diversi settori, tra cui l'ingegneria, la scienza, la finanza e la ricerca. Esempi di applicazioni di calcolo includono MATLAB, Mathematica, SPSS per la statistica e Microsoft Excel.
- **Applicazioni GIS:** I Sistemi Informativi Geografici (GIS, Geographic Information Systems) sono software utilizzati per acquisire, gestire, analizzare e visualizzare dati geografici. Esempi in questo caso: QGIS, GRASS, MapBox, ESRI ArcGIS.
- **Applicazioni di sistema:** queste applicazioni comprendono strumenti di backup, utility di sistema, antivirus e altre applicazioni utili per la gestione e la manutenzione del computer. Per esempio: Norton AntiVirus, McAfee AntiVirus, Google Backup and Sync, Microsoft OneDrive Backup.



# Software Open Source

Si parla di software open-source (OSS) quando il codice sorgente viene reso disponibile con una licenza in cui il titolare del copyright prevede il diritto di studiare, modificare e distribuire il software a chiunque e per qualsiasi scopo.

Il software open-source può essere sviluppato in maniera aperta e collaborativa da parte di community che si costituiscono allo scopo e in generale i suoi vantaggi sono riconducibili a:

- 1. Libertà:** gli utenti possono eseguire, copiare, distribuire, studiare, modificare e migliorare il software senza dover chiedere il permesso ai proprietari del software.
- 2. Personalizzazione:** il codice sorgente aperto consente agli sviluppatori di personalizzare il software per adattarlo alle proprie esigenze.
- 3. Collaborazione:** il modello open source promuove la collaborazione e il lavoro di squadra, poiché gli sviluppatori possono contribuire al software in modo cooperativo.
- 4. Innovazione:** l'accesso al codice sorgente e la possibilità di modificarlo favoriscono l'innovazione, poiché gli sviluppatori possono creare nuove funzionalità o migliorare quelle esistenti.

# Software Open Source

Tuttavia, ci sono anche alcune limitazioni del software open source, tra cui:

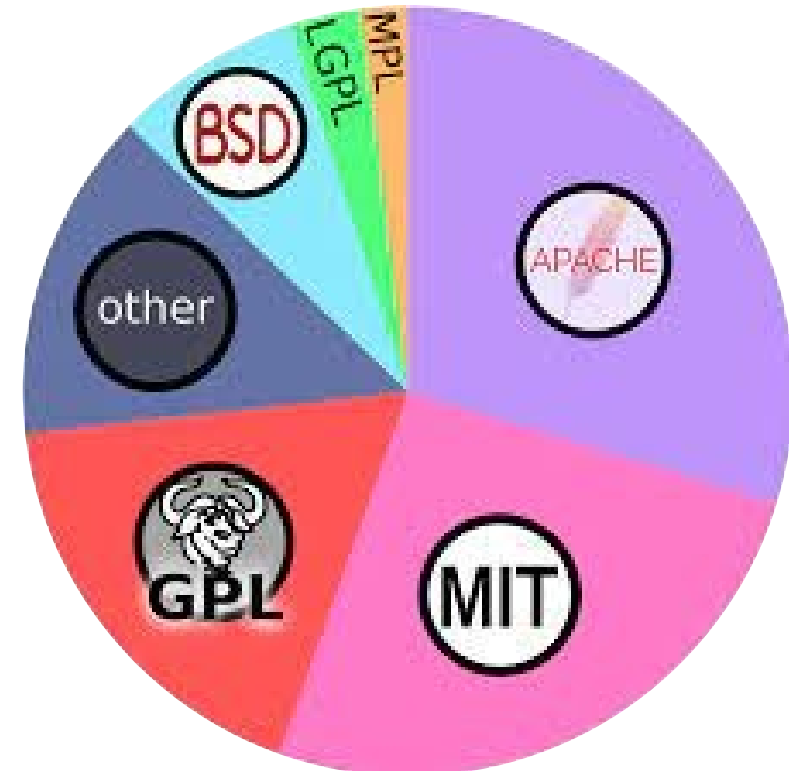
1. **Supporto:** il software open source spesso non ha un supporto tecnico professionale disponibile come quello fornito dai fornitori di software proprietari.
2. **Sicurezza:** l'apertura del codice sorgente può rendere il software vulnerabile ad attacchi informatici e minacce alla sicurezza.
3. **Conoscenze tecniche:** per utilizzare e personalizzare il software open source, gli utenti devono avere conoscenze tecniche approfondite di programmazione e di configurazione.
4. **Costi nascosti:** sebbene il software open source sia spesso gratuito, ci possono essere costi nascosti come quelli per la personalizzazione, l'implementazione e la formazione degli utenti.



# Software Open Source

Il Software Open Source può essere rilasciato con diverse tipologie di licenza, dove ognuna ha le proprie specifiche condizioni. A puro titolo di esempio alcune delle licenze open source più importanti e utilizzate includono:

- 1. GPL (General Public License):** è una delle licenze open source più utilizzate ed è stata creata dalla Free Software Foundation. Essa prevede che il codice sorgente sia disponibile al pubblico e che eventuali modifiche o sviluppi siano rilasciati sotto la stessa licenza. Inoltre, la GPL prevede che il software derivato dalla modifica del software originale debba essere rilasciato anche sotto la GPL.
- 2. Apache License:** è una licenza open source che prevede l'uso, la modifica e la distribuzione del software con poche restrizioni. Essa consente anche l'uso commerciale del software e non richiede che le modifiche siano rilasciate sotto la stessa licenza. Inoltre, la Apache License prevede che gli sviluppatori che utilizzano il software debbano fornire crediti e avvisi di copyright.
- 3. Creative Commons License:** è una serie di licenze open source che prevedono la condivisione di opere creative come musica, video e immagini. Esse prevedono diverse restrizioni sul modo in cui le opere possono essere utilizzate e distribuite.



# Software Open Source

- **Linux** è uno dei sistemi operativi open source più noti, disponibile in numerose varianti o distribuzioni, sviluppate da diverse organizzazioni o comunità di sviluppatori. Per esempio: Ubuntu, Fedora, Debian e altre.
- **Blender** è una 3D creation suite che supporta una grande varietà di operazioni di elaborazione grafica: modeling, animation, rendering, motion tracking, video editing e altro.
- **MySQL** è un potente DBMS relazionale utilizzato anche da alcuni dei content management system di maggior successo come WordPress, Joomla, Drupal.
- **Apache** è la piattaforma server Web modulare più diffusa, in grado di operare su una grande varietà di sistemi operativi, tra cui UNIX/Linux, Microsoft Windows e OpenVMS.
- **WordPress** è una piattaforma software di «Personal Publishing» e di Content Management System (CMS) per la creazione e distribuzione di siti Internet.

# Linguaggi di Programmazione

Come abbiamo più volte ripetuto, tutta l'informatica ruota intorno all'attività della **programmazione** nella quale le idee e le esigenze sono rappresentate in algoritmi e quindi in sequenze di istruzioni da impartire ai computer. Per poter programmare un computer servono alcuni strumenti fondamentali:

- I **linguaggi di programmazione** sono un insieme di regole e convenzioni utilizzate per scrivere codice che un computer può comprendere e eseguire.
- I **compilatori** sono programmi che traducono il codice sorgente scritto in un linguaggio di programmazione in codice macchina, ovvero il linguaggio che il computer può eseguire direttamente. Il processo di compilazione è generalmente diviso in tre fasi: analisi lessicale, analisi sintattica e analisi semantica.
- **Debugger**: un debugger è un software che aiuta a individuare gli errori e i problemi nel codice sorgente. Consente ai programmatori di eseguire il codice linea per linea, di fermarsi su specifiche istruzioni e di controllare le variabili in tempo reale.
- Gli **interpreti** sono programmi che eseguono direttamente il codice sorgente scritto in un linguaggio di programmazione. L'interprete analizza e esegue il codice una riga alla volta. Sono alternativi ai compilatori per alcuni specifici linguaggi (i.e. Python)

# Linguaggi di Programmazione

Un linguaggio di programmazione è considerato tale solo se è Turing-completo, cioè in grado di eseguire ogni tipo di calcolo che può essere eseguito da una macchina di Turing universale e quindi da un computer.

Le caratteristiche e le strutture dei linguaggi di programmazione variano in base al tipo di linguaggio. Tuttavia, in generale, i linguaggi di programmazione sono costituiti da un insieme di regole sintattiche e semantiche che definiscono come scrivere istruzioni che il computer può capire ed eseguire.

Le strutture dei linguaggi di programmazione possono includere le seguenti componenti:



# Linguaggi di Programmazione

- **Variabili:** le variabili sono spazi di memoria utilizzati per memorizzare i dati. Possono essere utilizzate per contenere numeri, stringhe di testo, booleani e altri tipi di dati.
- **Tipi di dati:** i tipi di dati indicano il tipo di informazione che può essere memorizzata in una variabile. Possono includere numeri interi, numeri in virgola mobile, stringhe di testo, booleani e altri tipi di dati.
- **Operatori:** gli operatori sono simboli utilizzati per eseguire operazioni su variabili e dati. Possono includere operatori matematici (+, -, \*, /), operatori di confronto (==, !=, >, <) e altri tipi di operatori.
- **Istruzioni di controllo del flusso:** le istruzioni di controllo del flusso sono utilizzate per modificare il flusso di esecuzione del programma. Possono includere istruzioni condizionali (if, else), cicli (for, while) e altri tipi di istruzioni.
- **Funzioni:** le funzioni sono blocchi di codice riutilizzabili che possono essere utilizzati per eseguire un'azione specifica. Possono essere scritte dal programmatore o incluse in librerie di funzioni predefinite.

```
REM -----  
REM elevamento a potenza X elevato alla Y  
REM X e Y sono due numeri naturali.  
REM -----  
REM  
INPUT "Inserisci la base "; X  
INPUT "Inserisci l'esponente "; Y  
LET Z = 1  
  FOR I = 1 TO Y  
    LET Z = Z * X  
  NEXT  
PRINT X; "^"; Y; "="; Z  
END  
REM -----FINE PROGRAMMA-----
```



# Prossimo Capitolo – La codifica dell'Informazione Numerica

Nel prossimo capitolo vedremo come viene codificata l'informazione attraverso i soli due simboli a disposizione nel mondo digitale e come sia possibile rappresentare un numero o un'immagine all'interno di un computer.