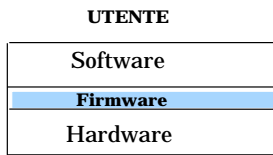


## Architettura di un Sistema di Elaborazione

Prima scomposizione di un "sistema informatico":

- **Hardware:** componenti fisici del sistema
- **Software:** i programmi che vengono eseguiti dal sistema

Il confine tra hardware e software in realtà non è sempre ben definito: in generale, vi è uno strato intermedio (il **firmware**) costituito da programmi direttamente mappati su circuiti elettronici.

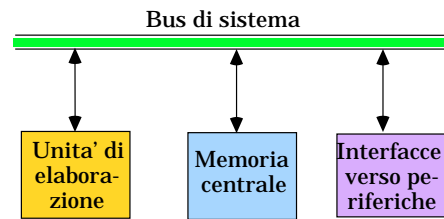


## Modello di Von Neumann Architettura di un elaboratore

Organizzata secondo il modello della **macchina di von Neumann** definita nei tardi anni '40 all'Institute for Advanced Study di Princeton.

È costituita da quattro elementi funzionali fondamentali:

- Unità centrale di elaborazione (**CPU**);
- **Memoria Centrale**;
- **Periferiche**;
- **Bus di sistema**.

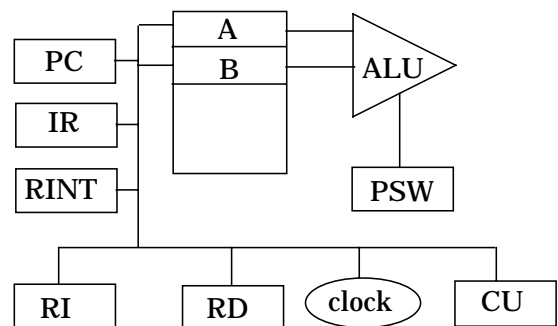


## Architettura di un elaboratore

- La **CPU** contiene i dispositivi elettronici in grado di acquisire, interpretare ed eseguire le istruzioni di ogni programma, trasformando i dati. Le istruzioni vengono eseguite **in sequenza**. Dati ed istruzioni vengono trasferiti da (e verso) la memoria centrale.
- La **memoria centrale** contiene sia le *istruzioni* che i *dati* (informazioni necessarie per eseguire un programma). Ha dimensioni limitate ed è volatile (cioè le informazioni memorizzate vengono perse allo spegnimento del computer).
- Le **periferiche** consentono uno scambio di informazioni fra l'elaboratore e l'esterno (*ingresso/uscita, memoria secondaria*). In particolare, la **memoria secondaria** (o memoria di massa) viene utilizzata per memorizzare grandi quantità di informazioni in modo persistente. Ha dimensioni elevate, ma l'accesso è meno rapido, rispetto alla memoria centrale.
- Il **bus di sistema** collega questi elementi funzionali. Fornisce il supporto fisico per la trasmissione dei dati tra i vari elementi.

## Unità di Elaborazione (Central Processing Unit, CPU)

È la parte che esegue e controlla l'elaborazione.



## Unità di Elaborazione (Central Processing Unit, CPU)

### Blocchi Componenti:

- l'**unità aritmetico-logica** (*Arithmetic Logic Unit, ALU*) che esegue le operazioni elementari necessarie per l'esecuzione delle istruzioni
- l'**unità di controllo** (*Control Unit, CU*), controlla e coordina l'attività della CPU. In particolare, è responsabile del trasferimento e della decodifica delle istruzioni dalla memoria centrale ai registri della CPU;
- il **clock** (orologio) "cadenza" le operazioni elementari, permettendo il sincronismo delle operazioni;
- vari **registri** (ad esempio, **A**, **B**, **PC**, **PSW**, etc.): un registro è una locazione utilizzata per memorizzare dati, istruzioni, o indirizzi all'interno della CPU. L'accesso ai registri è molto veloce.

☞ Nei moderni sistemi di elaborazione la CPU è realizzata da un unico circuito integrato (chip): il **microprocessore**.

## CPU: registri principali

- **RD**, registro dati: viene utilizzato per trasferire dati da e verso la memoria centrale.
- **RI**, registro indirizzi: viene utilizzato per memorizzare l'indirizzo della cella corrente nella memoria centrale (la sorgente/destinazione del trasferimento di dati).
- **PC** (*Program Counter*), registro "contatore" del programma. Contiene l'indirizzo della prossima istruzione da eseguire.
- **IR** (*Instruction Register*), registro istruzione corrente. Contiene, istante per istante, l'istruzione che è attualmente in esecuzione.
- Registri **accumulatori** o di lavoro (ad esempio, **A**, **B**,...). Contengono operandi e risultati delle operazioni svolte dalle ALU.
- **RINT**, registro interruzioni (stato periferiche)
- **PSW** (*Program Status Word*), i cui bit forniscono informazioni sul risultato dell'ultima operazione eseguita dalla ALU (overflow, zero, carry, segno)

## CPU: Unità di Controllo

### Fetch/execute

- L'*unità di controllo* ha il compito di:
  - reperire dalla memoria centrale le istruzioni di un programma (**fetch**);
  - interpretarle;
  - farle eseguire (**execute**).
- La CPU si comporta quindi come segue:

#### ripeti

*FETCH* di una istruzione  
*EXECUTE* dell'istruzione

finche' istruzione = HALT oppure ERRORE

- **FETCH**: provoca il trasferimento e la decodifica della prossima istruzione da eseguire (il cui indirizzo è nel registro PC). Le istruzioni sono organizzate in memoria in sequenza.
- **EXECUTE**: l'elaboratore esegue l'istruzione trovata (che è stata caricata nel registro IR). Istruzioni particolari possono alterare il flusso sequenziale (salto, chiamata di sotto-programmi).

## Unità Aritmetico-logica (ALU)

Realizza le operazioni aritmetiche e logiche necessarie per l'esecuzione delle istruzioni.

### Ad esempio:

ALU a due operandi (contenuti nei registri A e B) in grado di eseguire le operazioni aritmetiche (somma, sottrazione, prodotto, divisione):

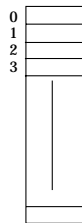
1. I due operandi vengono caricati nei registri A e B;
2. La ALU viene attivata in funzione da un codice operativo inviato dalla CU che specifica il tipo di operazione;
3. Nel registro A viene caricato il risultato dell'operazione eseguita dalla ALU;
4. Il registro PSW riporta sui suoi bit indicazioni sul risultato dell'operazione (riporto, overflow, etc.).

## Memoria centrale

- Contiene dati ed istruzioni relativi al programma in esecuzione.
- E' di dimensione limitata.
- E' un passaggio "obbligato" per dati e istruzioni: la CPU puo` scambiare direttamente informazioni soltanto con la memoria centrale.
- E' *volatile* ed, in generale, di dimensioni ridotte.

### Struttura della memoria centrale:

- E' una sequenza di celle di memoria, ciascuna contenente una sequenza di *bit* chiamata **parola** (word) di dimensione prefissata.
- Ogni parola e' caratterizzata da un indirizzo che la individua univocamente:



## Memoria Centrale

- La dimensione della parola cambia a seconda del tipo di calcolatore (8, 16, 32, 64 bit).
- La memoria centrale e' caratterizzata da una **Capacità**, che esprime la massima quantità di bit memorizzabili. Viene generalmente misurata in byte (1 byte=8 bit)
- La CPU può selezionare una particolare cella di memoria mediante l' **indirizzo** contenuto nel *registro indirizzi RI*.

## Memorizzazione: Unita` di misura

I sistemi di elaborazione sono realizzati con tecnologia digitale: le informazioni sono rappresentate mediante segnali elettrici a 2 valori di tensione { $V^{low}$ ,  $V^{high}$ } (oppure {0,1}).

- Per questo motivo, l'unita` logica di memorizzazione (e, in generale, di rappresentazione delle informazioni) e' il **bit** (binary digit):

☞ un **bit** e' una grandezza il cui dominio di variazione e' composto dai due valori {0,1}.

- il **byte** equivale ad 8 bit

### Unita` successive:

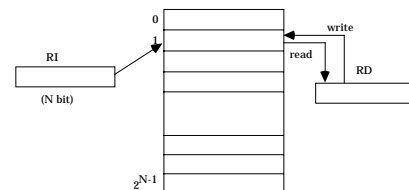
- potenze in base 2 del byte (con esponente multiplo intero di 10):

<b>Kilobyte</b>	$2^{10}$ byte	1024 byte ( $\sim 10^3$ )	<b>KB</b>
<b>Megabyte</b>	$2^{20}$ byte	1048576 byte ( $\sim 10^6$ )	<b>MB</b>
<b>Gigabyte</b>	$2^{30}$ byte	$\sim 10^9$ byte	<b>GB</b>
<b>Terabyte</b>	$2^{40}$ byte	$\sim 10^{12}$ byte	<b>TB</b>

## Indirizzamento

Se il registro indirizzi RI e' lungo N bit, si possono indirizzare  $2^N$  celle di memoria (con indirizzo da 0 a  $2^N - 1$ ).

Ad esempio, con N=10, si indirizzano 1024 parole.



Indirizzata una cella attraverso RI, si possono eseguire operazioni di lettura e di scrittura da e verso il *registro dati* RD.

**Operazione di lettura:** Trasferisce il contenuto della cella di memoria indirizzata dal Registro Indirizzi nel Registro Dati.

**Operazione di scrittura:** Trasferisce il contenuto del Registro Dati nella cella di memoria indirizzata dal Registro Indirizzi.

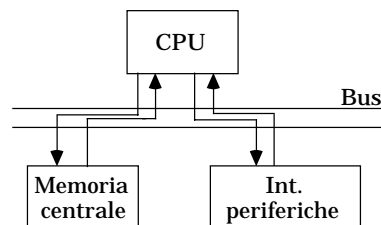
Le operazioni vengono coordinate dalla Control Unit (CU).

## Dispositivi fisici utilizzati per la memoria centrale

- **RAM:** Random Access Memory (ad accesso casuale): su di essa si possono svolgere operazioni sia di lettura che di scrittura.
  - **ROM:** Read Only Memory (a sola lettura): non volatili e non scrivibili; in esse vengono contenuti i dati e programmi per inizializzare il sistema.
  - **PROM, EPROM, etc.:** ROM programmabili, eventualmente riscrivibili.
- ☞ **Firmware** : e' costituito dal software memorizzato nelle ROM (codice microprogrammato).

## Bus di sistema

- Interconnette la CPU, la memorie e le interfacce verso dispositivi periferici (I/O, memoria di massa, etc.)
- Collega due unita' funzionali alla volta: una trasmette e l'altra riceve. Il trasferimento avviene sotto il controllo della CPU (Control Unit).



Su questo supporto (spesso costituito da più linee) viaggiano dati, indirizzi e comandi. Si distinguono spesso tali linee in:

- bus **dati** (data bus)
- bus **indirizzi** (address bus)
- bus **comandi** (command bus)

## Bus di Sistema

- **Bus dati:** bidirezionale. Serve per trasmettere dati dalla memoria al registro dati o viceversa.
- **Bus indirizzi:** unidirezionale. Serve per trasmettere il contenuto del registro indirizzi alla memoria. Viene selezionata una specifica cella per successive operazioni di lettura o scrittura.
- **Bus comandi:** unidirezionale. Ad esempio, comando di lettura o scrittura verso la memoria; comando di stampa verso una periferica (interfaccia).

Se la dimensione (numero di bit) del bus dati è uguale alla dimensione della parola, si può trasferire in parallelo un intero dato. Altrimenti occorrono più trasferimenti.

## Interfacce di Ingresso/Uscita

Consentono il collegamento dell'elaboratore con le varie periferiche (dischi, terminali, stampanti, ...). Sono diverse a seconda del tipo di periferica (sia hardware, che software).

### Periferiche:

- **Tastiera**, e' un dispositivo di ingresso (input) che consente al calcolatore di acquisire dati dall'utente. I dati vengono immessi come sequenze di caratteri (uno per ogni tasto premuto).
- **Mouse**, e' un dispositivo di ingresso che puo' essere utilizzato come integrazione (o persino in sostituzione) della tastiera.
- **Video**, e' un dispositivo di uscita che consente la visualizzazione di dati e risultati dell'elaborazione. E' caratterizzato da vari parametri, tra cui:
  - numero di colori rappresentabili
  - dimensioni dello schermo (in pollici)
  - risoluzione (numero di punti per pollice quadrato)
  - capacita' grafica

- **Stampanti**, dispositivi di uscita che producono la visualizzazione su carta (o altri supporti simili) di dati e risultati dell'elaborazione.

Sono caratterizzate da:

- *velocità* di stampa (byte, carattere, o pagine al secondo),
- *risoluzione* (numero di punti per pollice quadrato),
- capacità grafica,
- rumorosità,
- set di caratteri (o font) stampabili.

Varie tecnologie disponibili: ad aghi, a margherita, a getto d'inchiostro, laser, a trasferimento termico.

- **Terminali**, hanno una tastiera ed un video. Possono essere *alfanumerici* o *grafici*.
- Anche la **memoria secondaria** (o di massa) è vista come dispositivo periferico.

## Memoria secondaria (o di massa)

La memoria secondaria si basa su dispositivi per la memorizzazione di grandi masse di dati.

I dati memorizzati in questo tipo di memoria sopravvivono all'esecuzione dei programmi (*persistenti*).

La *capacità* (dimensione della memoria) varia molto da dispositivo a dispositivo: dalle decine di mega-byte ( $10^6$  byte) ai giga-byte ( $10^9$  byte) o tera-byte ( $10^{12}$  byte).

Anche la *velocità di accesso/trasferimento* varia da dispositivo a dispositivo (comunque molto superiore a quella della memoria centrale).

$T_{\text{accesso}}$ (memoria centrale)	≈	100 nsec
$T_{\text{accesso}}$ (dischi magnetici)	≈	10-20 msec
$T_{\text{accesso}}$ (dischetti)	≈	100 msec

(1 msec =  $10^{-3}$  sec; 1 nsec =  $10^{-9}$  sec)

## Dispositivi di memoria di massa

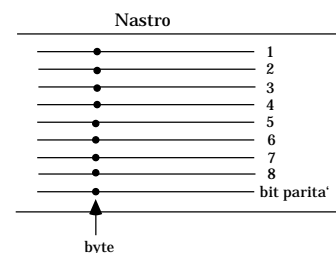
Due classi fondamentali in base al metodo di accesso consentito:

1. ad *accesso sequenziale* (ad esempio, nastri): per cercare un dato è necessario accedere a tutti quelli che lo precedono sul dispositivo;
2. ad *accesso diretto* ai dati (ad esempio, dischi). E' possibile accedere direttamente a qualunque dato memorizzato, grazie all'indirizzamento di porzioni (blocchi) del dispositivo.

Nel caso di dispositivi magnetici (nastri o dischi) l'informazione è presente in memoria come *stato di polarizzazione magnetica*, che può essere positivo o negativo (codifica binaria).

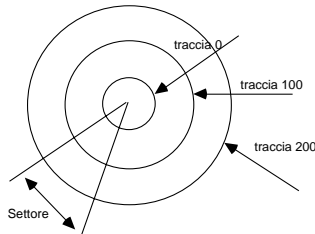
## Nastri magnetici

- Sono fettucce di materiale magnetizzabile arrotolate su supporti circolari, o in cassette.
- Sulla fettuccia sono tracciate delle piste orizzontali parallele. Di solito, 9 piste parallele di cui 8 corrispondono ad un byte e la nona è il bit di parità.



- Il parametro più importante è la *densità* misurata in bit per pollice (**bpi**, *bit per inch*).
- I dati su nastro sono organizzati in zone contigue dette **registrazioni** (record).
- Tutte le **elaborazioni** sono **sequenziali** (lentezza delle operazioni di lettura/scrittura su un preciso record).
- Ormai svolgono solo una funzione di copia di riserva (**backup**).

## Dischi magnetici



Un disco è costituito da un certo numero di *piatti* di materiale magnetizzabile con due superfici che ruotano attorno ad un perno centrale.

Ciascuna superficie ha una serie di cerchi concentrici o **tracce** e viene suddivisa in spicchi di ugual grandezza chiamati **settori**. Tutte le tracce equidistanti dal centro formano un **cilindro**.

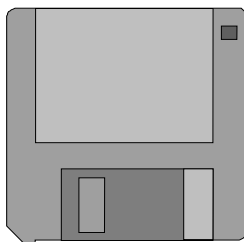
La testina si sposta longitudinalmente lungo le tracce. I dati sono scritti occupando posizioni successive lungo le tracce. Corrispondono ad uno stato di *polarizzazione* (positiva o negativa) del materiale magnetizzabile che costituisce i dischi.

Ogni *blocco* di ingresso/uscita è selezionabile mediante la terna *<superficie, traccia, settore>* (*indirizzo*).

### Ingresso (o uscita) da (o verso) *<superficie, traccia, settore>* :

- 1• spostamento della testina (seek) verso la traccia richiesta,
- 2• attesa affinché il settore arrivi sotto la testina,
- 3• trasferimento dei dati in (o da) memoria centrale, solitamente eseguito da un processore dedicato (Direct Memory Access, DMA).

## Dischetti (floppy disk):



- Sono dischetti portatili che vengono utilizzati per trasferire informazioni (file) tra computer diversi.
- Costituiti da un unico disco con due superfici.
- vari tipi, in base al diametro (3.5, 5.25 e 8 pollici)
- I dischi vengono **formattati** dal Sistema Operativo che li suddivide in tracce e settori e ne determina la densità (e la capacità). Tipicamente, **1.44** Mbyte.

## Capacità delle memorie

Tipo di memoria	Capacità
Memoria centrale	1-256 Mbyte
Dischi magnetici	80-2000 Mbyte
Dischi floppy	0.7-2 Mbyte
Nastri (bobina)	20-400 Mbyte
Nastri (cassetta)	200-5000 Mbyte
Dischi ottici (CD)	600-4000 Mbyte

## Personal Computer

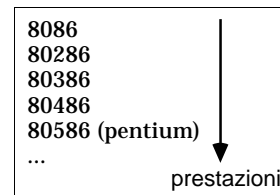


- Memoria di massa generalmente composta da disco rigido (**hard disk**) fisso, dischetti (**floppy disk**) e CD estraibili.
- L'informazione nella memoria di massa e' organizzata in **archivi** (o **file**) caratterizzati da un nome.
- Varie classi di PC in base al tipo di processore (Intel, Macintosh, etc.)

## Personal Computer

### “IBM-compatibili”:

hanno processori della famiglia *Intel 80x86* :



- le prestazioni sono influenzate anche da altri parametri :
  - frequenza del clock
  - dimensione RAM
  - velocita' del BUS
  - ...
- unita' di misura delle prestazioni:

**MIPS** (migliaia di istruzioni per secondo)

**Mflops** (migliaia di operazioni floating point per secondo)

- I Personal Computer hanno struttura semplice, costo ridotto, ma prestazioni limitate.

## Altri sistemi di calcolo

### Workstation:

sistemi generalmente dedicati ad un utente, ma con capacita' di supportare piu' attivita' contemporanee. Prestazioni piu' elevate dei PC.

### Mini-calcolatori:

Macchine capaci di servire decine di utenti contemporaneamente, collegati tramite terminali.

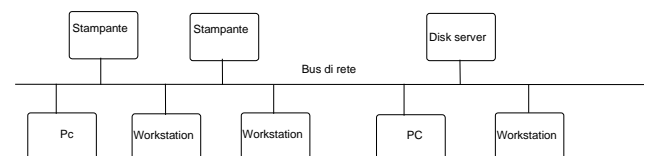
### Super-calcolatori:

Hanno molti processori e grandi memorie di massa (centinaia o migliaia di terminali)

- ☞ Possibilita' di connettere assieme vari calcolatori di tipo anche diverso (**reti di calcolatori**).

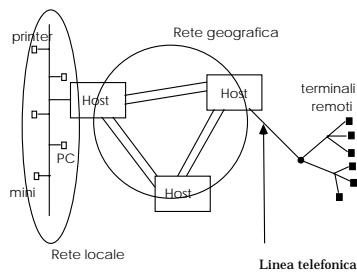
### Reti locali:

collegano elaboratori fisicamente vicini (nello stesso ufficio o stabilimento). L'obiettivo e' la condivisione di risorse:



### Reti geografiche:

- Collegano elaboratori distribuiti su un'area geografica di dimensioni estese (anche intercontinentali). Ad esempio: **Internet**



Evoluzione e complessità sia dell'hardware che del software (*protocolli di collegamento*)

### Hardware/Software

CPU, memoria centrale e dispositivi sono realizzati con **tecnologia digitale**.

Dati ed operazioni vengono codificati mediante sequenze di bit

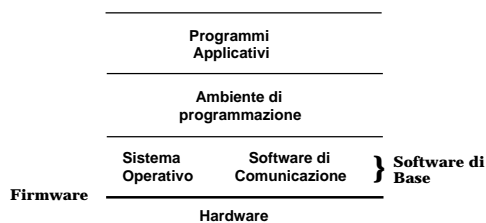
```
01000110101 ....
```

L'utilizzo della sola struttura fisica (**hardware**) dell'elaboratore imporrebbe all'utente di esprimere i propri comandi attraverso sequenze di bit (**linguaggio macchina**).

Per questo motivo, ogni elaboratore è corredato da un insieme di programmi che elevano il livello di interazione utente-macchina, avvicinando il linguaggio di interazione al linguaggio naturale: il **software**.

### Software

È un insieme di programmi. Per semplicità si può pensare ad una organizzazione a strati, ciascuno con funzionalità di livello più alto rispetto a quelli sottostanti (**macchina virtuale**)



**Firmware:** confine fra hardware e software. È uno strato di microprogrammi scritti su memorie permanenti dai costruttori che agiscono direttamente al di sopra dello strato hardware.

### Sistema Operativo

Insieme di programmi che rendono l'elaboratore *operativo ed usabile*.

Il Sistema Operativo opera direttamente al di sopra di hardware e firmware)

Le funzioni messe a disposizione dipendono dalla complessità del sistema di elaborazione:

- Gestisce le risorse disponibili;
- Interpreta ed esegue comandi elementari;
- Stampa, legge, visualizza su video;
- Gestisce la memoria centrale ed
- Organizza e gestisce la memoria di massa;
- Gestisce un sistema multi-utente;
- etc...

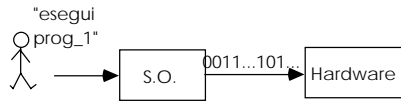
In pratica l'utente "vede" la macchina solo attraverso il Sistema Operativo.

Attraverso il S.O. il livello di interazione utente-elaboratore viene elevato:

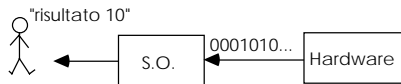
- **senza S.O.:** sequenze di bit per esprimere istruzioni e dati
- **con S.O.:** parole "chiave" (comandi) programmi dati (interi, reali, caratteri, etc.)

## Sistema Operativo

Il S.O. traduce le richieste dell'utente in opportune sequenze di impulsi da sottoporre alla macchina fisica:

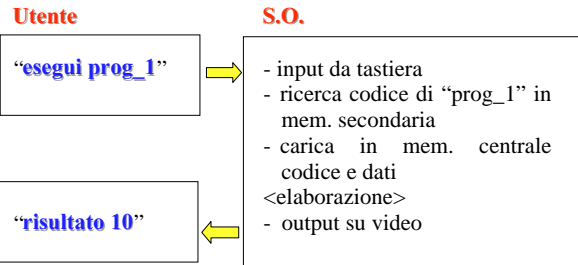


e viceversa:



## Sistema Operativo

Qualsiasi operazione di accesso a risorse della macchina implicitamente richiesta dal comando di utente viene esplicitata dal S.O. (ad esempio, accesso a memoria centrale, secondaria, oppure I/O verso video, tastiera, etc.)



## Classificazione dei Sistemi Operativi:

### In base al numero di utenti:

- **Mono-utente:** un solo utente alla volta può utilizzare il sistema
- **Multi-utente:** più utenti contemporaneamente possono interagire con la macchina.

⇒ nel caso di più utenti collegati, il S.O. deve fornire a ciascun utente l'astrazione di un sistema "dedicato".

### In base al numero di programmi in esecuzione:

- **Mono-programmato:** il sistema può gestire l'esecuzione di al più un programma alla volta.
- **Multi-programmato:** il sistema operativo è in grado di portare avanti l'esecuzione contemporanea di più programmi (mantenendo una sola CPU).

⇒ nel caso di multi-programmazione il S.O. deve gestire l'unità di elaborazione (CPU) suddividendola tra i vari programmi.

### Esempi:

- **MS-DOS:** mono-utente, monoprogrammato
- **UNIX:** multiutente, multiprogrammato
- **Windows'95, OS/2:** monoutente, multiprogrammato

## Software di comunicazione

È l'insieme dei programmi che si occupano di supportare la comunicazione tra macchine collegate in rete. La comunicazione avviene utilizzando **protocolli** che garantiscono un corretto scambio dei dati e messaggi.

### Esempio:

- **rete internet** (protocollo TCP/IP):
  - *telnet*: apertura di una sessione remota
  - *ftp*: trasferimento di file
  - *mail*: posta elettronica
  - ...

## Programmi applicativi

Risolvono problemi specifici degli utenti.

- *word processor*: elaborazione di testi.
- *fogli elettronici* (spreadsheet): gestione di tabelle e grafici
- *data base*: gestione di archivi
- Sono scritti in **linguaggi di programmazione** di alto livello.
- Essendo di **alto livello**, risentono in misura ridotta o nulla delle caratteristiche dell'architettura dell'ambiente sottostante (*portabilità*).

## Ambiente di programmazione

E' l'insieme dei programmi che consentono la scrittura, la verifica e l'esecuzione di nuovi programmi (fasi di sviluppo).

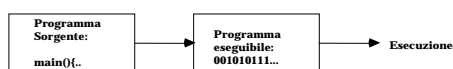
### Sviluppo di un Programma

- Affinche' un programma scritto in un qualsiasi linguaggio di programmazione (ad es. il C) sia comprensibile (e quindi eseguibile) da un calcolatore, e' necessaria un'azione di traduzione dal linguaggio originario al linguaggio macchina
- Questa operazione viene normalmente svolta da speciali programmi, detti **traduttori**.

Programma	Traduzione
<pre>main() { int A;   ...   A=A+1;   if....</pre>	<pre>00100101 11001.. 1011100..</pre>

I traduttori provvedono a convertire il codice di programmi scritti in un particolare linguaggio di programmazione (programmi *sorgenti*), nella corrispondente rappresentazione in linguaggio macchina (programmi *eseguibili*).

## Sviluppo di Programmi



### Due categorie di traduttori:

- i **Compiler**: accettano in ingresso l'intero programma e producono in uscita la rappresentazione dell'intero programma in linguaggio macchina.
- gli **Interpreti**: traducono ed eseguono direttamente ciascuna istruzione del *programma sorgente*, istruzione per istruzione.

### Quindi:

- **Compiler**: per ogni programma da tradurre, lo schema viene percorso una volta sola prima dell'esecuzione.
- **Interprete**: lo schema viene attraversato tante volte quante sono le istruzioni che compongono il programma; ad ogni attivazione dell'interprete su una particolare istruzione, segue l'esecuzione dell'istruzione stessa.

☞ l'esecuzione di un programma compilato e' piu' veloce dell'esecuzione di un programma mediante interprete.

## Fasi di Sviluppo di un Programma

Lo sviluppo di un *semplice* programma avviene attraverso l'attuazione di una sequenza di fasi.

### Dato un problema da risolvere:

- 1) scelta del metodo risolutivo e rappresentazione mediante un **algoritmo**.
- 2) rappresentazione dell'algoritmo nel linguaggio di programmazione scelto (ad esempio, C): si ottiene il programma in forma sorgente. La fase di scrittura del programma viene normalmente detta di **editing**.
- 3) **compilazione**. Il programma sorgente prodotto nella fase di editing viene tradotto dal compilatore in linguaggio macchina (programma eseguibile). In questa forma, il programma e' pronto per essere eseguito. La fase di compilazione puo' rilevare eventuali errori (generalmente sintattici) contenuti nel programma sorgente.

4) **debugging**. Analisi del corretto funzionamento del programma. Può essere agevolmente eseguita mediante speciali programmi detti **debugger**. Mediante un debugger è possibile effettuare l'esecuzione controllata del programma.

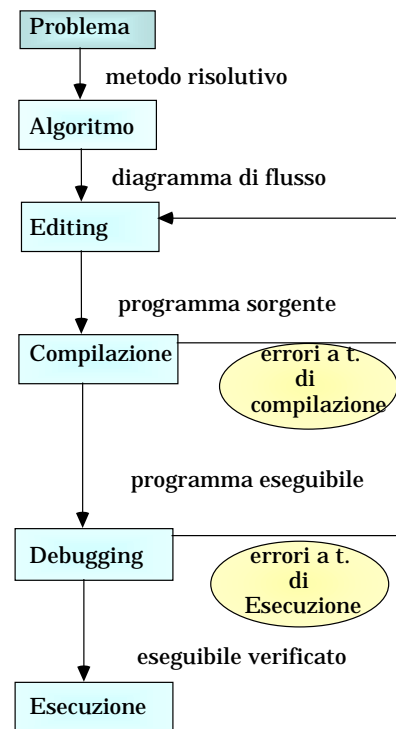
Ad esempio:

- esecuzione di una istruzione per volta
- visualizzazione dei valori di variabili
- punti di arresto *breakpoint*

☞ rilevazione di errori non riscontrabili in fase di compilazione (errori a tempo di esecuzione)

5) **esecuzione**. Quando il programma è stato opportunamente verificato, può essere infine eseguito per la risoluzione del problema di partenza.

## Fasi di Sviluppo



## Componenti di un ambiente di programmazione

- **Editor**: serve per la costruzione di file che contengono testi (cioè sequenze di caratteri). In particolare tramite un editor si scrive il *programma sorgente*.
- **Compilatore**: opera la traduzione di un programma sorgente scritto in un linguaggio ad alto livello in un *programma oggetto* scritto in un linguaggio direttamente eseguibile dal calcolatore.
- **Linker**: (collegatore o correlatore) nel caso in cui il programma sia suddiviso in moduli *oggetto* separati compilati separatamente provvede a collegarli per formare un unico *programma eseguibile*.
- **Debugger**: (scopritore di banchi, cioè errori) serve per scoprire ed eliminare errori presenti durante l'esecuzione di un programma, ma non rilevati in fase di compilazione.