# INFORMATICA

- Varie definizioni:
  - "Scienza degli elaboratori elettronici" (*Computer Science*)
  - "Scienza dell'informazione"
- Definizione proposta:
  - ***Scienza della rappresentazione e dell'elaborazione dell'informazione***

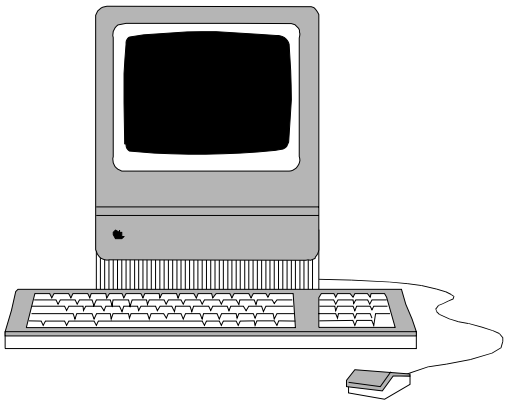


# L'informatica comprende:

- Metodi per la ***rappresentazione*** delle informazioni
- Metodi per la ***rappresentazione*** delle soluzioni
- ***Linguaggi di programmazione***
- ***Architettura*** dei calcolatori
- ***Sistemi operativi***
- ***Reti di calcolatori***
- Sistemi e applicazioni distribuite
- ***Tecnologie Web***
- Algoritmi
- ……

# ELABORATORE ELETTRONICO ("*COMPUTER*")

**Strumento** per la rappresentazione e l'elaborazione delle informazioni





# L'ELABORATORE

**Componenti principali**

- Unità centrale
- Video ("monitor")
- Tastiera e Mouse
- Lettore CD
- Dischi fissi ("hard disk")
- Dischetti ("floppy")

**Componenti accessori**

- Stampante
- Modem
- Scanner
- Tavolette grafiche

…

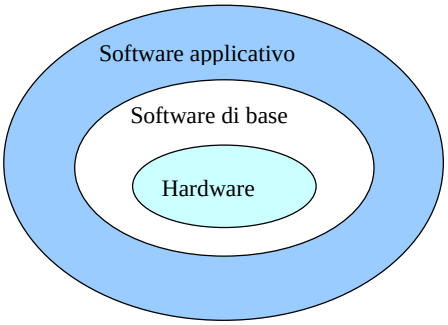
**HARDWARE**

# SOFTWARE

**Software: programmi** che vengono eseguiti dal sistema

**Distinzione fra:**

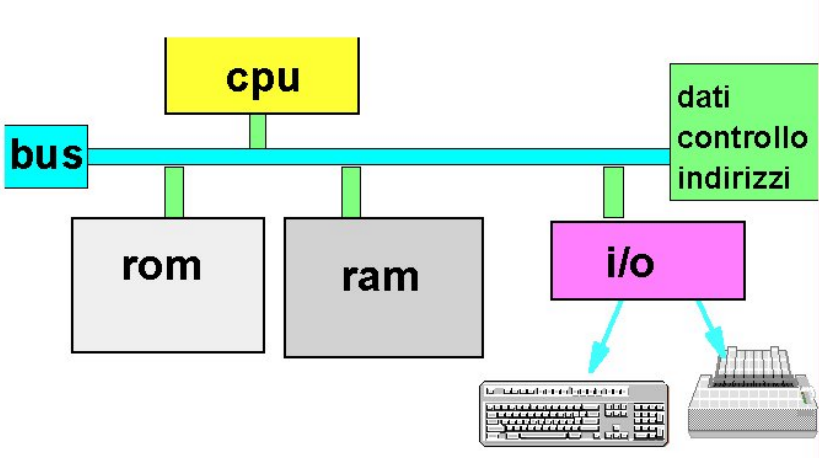
- *Software di base* (es. Sistema Operativo)
- *Software applicativo*






# HARDWARE

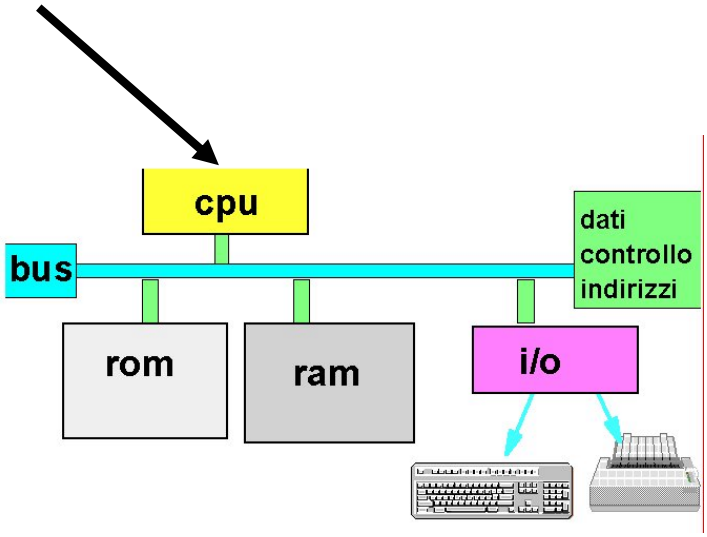
È composto da un insieme di ***unità funzionali***



# HARDWARE

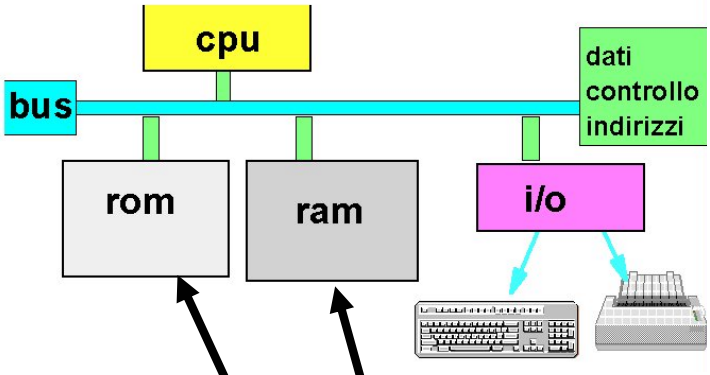
**CPU (Central Processing Unit)**, o **Processore**

**CPU:** Svolge le elaborazioni e il trasferimento dei dati, cioè *esegue i programmi*






# HARDWARE




**RAM & ROM**

- Dimensioni relativamente limitate
- Accesso molto rapido

**RAM (*Random Access Memory*), e**
**ROM (*Read Only Memory*)**
**Insieme formano la *Memoria centrale***

# HARDWARE

**ATTENZIONE**

**RAM è volatile** (perde il suo contenuto quando si spegne il calcolatore)

- usata per memorizzare dati e programmi

**ROM è persistente** (mantiene il suo contenuto quando si spegne il calcolatore) ma il suo ***contenuto è fisso e immutabile***

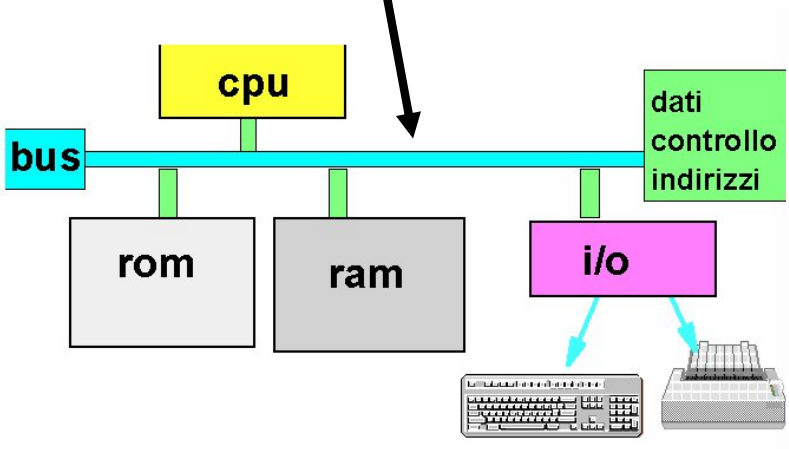
- usata per memorizzare programmi di sistema



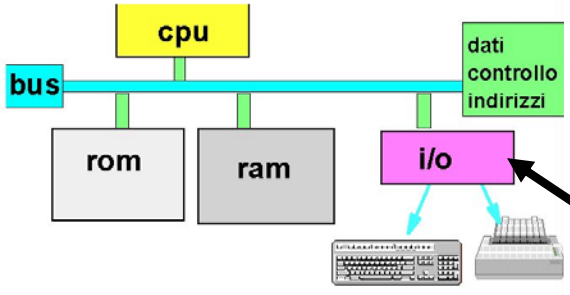
# HARDWARE

**BUS DI SISTEMA**

È una “linea di comunicazione” che collega tutti gli elementi funzionali



# HARDWARE



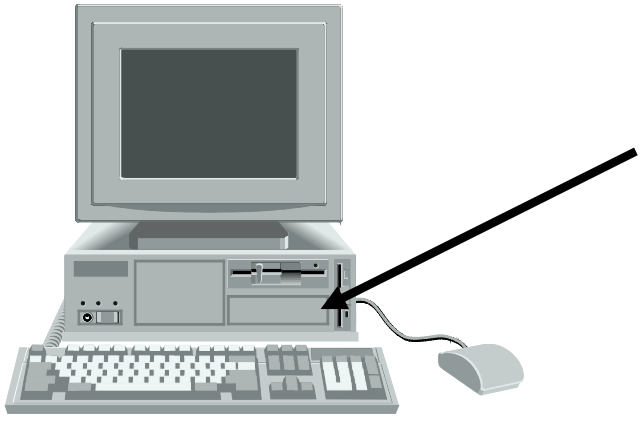

Sono usate per far comunicare il calcolatore con l'esterno (in partico-lare con l'utente)

**UNITÀ DI INGRESSO / USCITA (I/O)**

- Tastiera e Mouse
- Video e Stampante
- Scanner
- Tavoletta grafica
- **Dispositivi di memoria di massa**
- ...



# HARDWARE



**MEMORIA DI MASSA**

- Dischi
- CD
- Nastri
- ...

- memorizza **grandi quantità** di informazioni
- **persistente** (le informazioni non si perdono spegnendo la macchina)
- accesso molto meno rapido della memoria centrale (**milli**secondi contro **nano**secondi; differenza $10^6$)

# TECNOLOGIA DIGITALE

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

Dati ed operazioni vengono codificati a partire da due valori distinti di grandezze elettriche:

- tensione alta ($V_H$, ad es. 5V)
- tensione bassa ($V_L$, ad es. 0V)

A tali valori vengono convenzionalmente **associate le due cifre binarie 0 e 1:**

- **logica positiva: $1 \leftrightarrow V_H$, $0 \leftrightarrow V_L$**
- logica negativa: $0 \leftrightarrow V_H$, $1 \leftrightarrow V_L$



# TECNOLOGIA DIGITALE (segue)

Dati ed operazioni vengono codificati tramite **sequenze di bit**

**01000110101 ....**

CPU è in grado di operare soltanto in aritmetica binaria, effettuando operazioni *elementari*:

- somma e differenza
- scorrimento (shift)
- ...

Lavorando direttamente sull'hardware, **l'utente è forzato a esprimere i propri comandi *al livello della macchina*, tramite *sequenze di bit***

## RAPPRESENTAZIONE DELL'INFORMAZIONE

- **Internamente a un elaboratore, ogni informazione è rappresentata tramite *sequenze di bit (cifre binarie)***
- **Una sequenza di bit *non dice "che cosa" essa rappresenta***

Ad esempio, 01000001 può rappresentare:

- l'intero 65, il carattere 'A', il boolean '*vero', …*
- … il valore di un segnale musicale,
- … il colore di un puntino sullo schermo...



## INFORMAZIONI NUMERICHE

- **La rappresentazione delle *informazioni numeriche* è di particolare rilevanza**
- **In questa sede ci limiteremo ai *numeri naturali (interi senza segno)***

**Dominio: N = { 0,1,2,3, …}**

## NUMERI NATURALI (interi senza segno)

**Dominio: N = { 0,1,2,3, …}**

**Rappresentabili con diverse notazioni**

- ***non posizionali***
  - **ad esempio la notazione romana: I, II, III, IV, V, .... IX, X, XI…**
  - **Risulta difficile l'utilizzo di regole generali per il calcolo**
- ***posizionale***
  - **1, 2, .. 10, 11, ... 200, ...**
  - **Risulta semplice l'individuazione di regole generali per il calcolo**



## NOTAZIONE POSIZIONALE

- **Concetto di *base* di rappresentazione *B***
- **Rappresentazione del numero come *sequenza di simboli (cifre)* appartenenti a un *alfabeto* di *B* simboli distinti**
- **ogni simbolo rappresenta un valore compreso fra 0 e *B*-1**

Esempio di rappresentazione su N cifre:

$$d_{n-1} \ \dots \ d_2 \ d_1 \ d_0$$

# NOTAZIONE POSIZIONALE

**Il valore di un numero espresso in questa notazione è ricavabile**

- **a partire dal valore rappresentato da ogni simbolo**
- ***pesandolo* in base alla *posizione* che occupa nella sequenza**

$d_{n-1}$ … $d_2$ $d_1$ $d_0$

**Posizione n-1:** pesa $B^{n-1}$

**Posizione 1:** pesa $B^1$

**Posizione 0:** pesa $B^0$ (unità)



# NOTAZIONE POSIZIONALE

**In formula:**

$$v = \sum_{k=0}^{n-1} d_k B^k$$

*dove*

- *B = base*
- *ogni cifra $d_k$ rappresenta un valore fra 0 e B-1*

**Esempio (base B=4):**

**1 2 1 3**

**$d_3$ $d_2$ $d_1$ $d_0$**

*Valore* = **1** * $B^3$ + **2** * $B^2$ + **1** * $B^1$ + **3** * $B^0$ = *centotre*

## NOTAZIONE POSIZIONALE

**Quindi, *una sequenza di cifre non è inter-pretabile* se non si precisa la base in cui è espressa**

**Esempi:**

| Stringa | Base | Alfabeto | Calcolo valore | Valore |
|---|---|---|---|---|
| "12" | *quattro* | {0,1,2,3} | 4 * 1 + 2 | *sei* |
| "12" | *otto* | {0,1,...,7} | 8 * 1 + 2 | *dieci* |
| "12" | *dieci* | {0,1,...,9} | 10 * 1 + 2 | *dodici* |
| "12" | *sedici* | {0,..,9, A,., F} | 16 * 1 + 2 | *diciotto* |



## NOTAZIONE POSIZIONALE

**Inversamente, ogni numero può essere espresso, *in modo univoco*, *come sequenza di cifre in una qualunque base***

**Esempi:**

| Numero | Base | Alfabeto | Rappresentazione |
|---|---|---|---|
| *venti* | *due* | {0,1} | **"10100"** |
| *venti* | *otto* | {0,1,...,7} | **"24"** |
| *venti* | *dieci* | {0,1,...,9} | **"20"** |
| *venti* | *sedici* | {0,..,9, A,., F} | **"14"** |

## CONCLUSIONE

**Quindi:**

- ***i numeri sono concetti, che esistono in quanto tali***
- ***la loro rappresentazione può invece variare a seconda delle convenzioni adottate***

**Non bisogna *confondere* un numero con una sua RAPPRESENTAZIONE!**



## NUMERI E LORO RAPPRESENTAZIONE

- **Internamente**, ***un elaboratore adotta per i numeri interi (non negativi) una rappre-sentazione binaria (base B=2)***
- ***Esternamente**, le costanti numeriche che scriviamo nei programmi e i valori che stampiamo a video / leggiamo da tastiera sono invece sequenze di caratteri ASCII*

**Il passaggio dall'una all'altra forma richiede dunque un processo di *conversione***

## Esempio: RAPPRESENTAZIONE INTERNA/ESTERNA

- **Numero**: ***centoventicinque***
- **Rappresentazione interna binaria (16 bit):**

  **00000000 01111101**

- **Rappresentazione esterna in base 10:**

  ***occorre produrre la sequenza di caratteri ASCII '1', '2', '5'***

  **00110001 00110010 00110101**

  vedi tabella ASCII

## Esempio: RAPPRESENTAZIONE INTERNA/ESTERNA

- **Rappresentazione esterna in base 10:**

  ***È data la sequenza di caratteri ASCII '3', '1', '2', '5', '4'***

  vedi tabella ASCII

  **00110011 00110001 00110010 00110101 00110100**

- **Rappresentazione interna binaria (16 bit):**

  **01111010 00010110**

- **Numero**: ***trentunomiladuecentocinquantaquattro***

## CONVERSIONE STRINGA/NUMERO

**Si applica la definizione:**

$$v = \sum_{k=0}^{n-1} d_k B^k$$

**le cifre $d_k$ sono note, il valore *v* va calcolato**

$$= d_0 + B * ( d_1 + B * ( d_2 + B * ( d_3 + ... )))$$

Ciò richiede la valutazione di un polinomio

→ ***Metodo di Horner***



## CONVERSIONE NUMERO/STRINGA

- **Problema:** ***dato un numero, determinare la sua rappresentazione in una base data***
- **Soluzione (*notazione posizionale*): *manipolare la formula* per dedurre un algoritmo**

$$v = \sum_{k=0}^{n-1} d_k B^k$$

***v* è noto, le cifre $d_k$ vanno calcolate**

$$= d_0 + B * ( d_1 + B * ( d_2 + B * ( d_3 + ... )))$$

## CONVERSIONE NUMERO/STRINGA

- **Per trovare le cifre bisogna *calcolarle una per una,* ossia…**
- **… bisogna trovare un modo per *isolarne una dalle altre***

$$v = d_0 + B * (\ldots)$$

**Osservazione:**
**$d_0$ è la sola cifra *non moltiplicata per B***

**Conseguenza:**
**$d_0$ è ricavabile come *v modulo B***



## CONVERSIONE NUMERO/STRINGA

***Algoritmo delle divisioni successive***

- **si divide *v* per B**
  - *il resto* costituisce la cifra meno significativa ($d_0$)
  - *il quoziente* serve a iterare il procedimento
- **se tale quoziente è zero, l'algoritmo termina;**
- **se non lo è, lo si assume come nuovo valore *v'*, e si itera il procedimento con il valore *v'***

# CONVERSIONE NUMERO/STRINGA

## Esempi

| Numero | Base | Calcolo valore | Stringa |
|---|---|---|---|
| *quattordici* | 4 | 14 / 4 = 3 con resto 2<br>3 / 4 = 0 con resto 3 | "32" |
| *undici* | 2 | 11 / 2 = 5 con resto 1<br>5 / 2 = 2 con resto 1<br>2 / 2 = 1 con resto 0<br>1 / 2 = 0 con resto 1 | "1011" |
| *sessantatre* | 10 | 63 / 10 = 6 con resto 3<br>6 / 10 = 0 con resto 6 | "63" |
| *sessantatre* | 16 | 63 / 16 = 3 con resto 15<br>3 / 16 = 0 con resto 3 | "3F" |



# NUMERI NATURALI: valori rappresentabili

- **Con N bit, si possono fare *$2^N$ combinazioni***
- **Si rappresentano così i numeri da 0 a $2^N$-1**

## Esempi

- **con 8 bit, [ 0 …. 255 ]**
  *In C: unsigned char = byte*
- **con 16 bit, [ 0 …. 65.535 ]**
  *In C: unsigned short int (su alcuni compilatori)*
  *In C: unsigned int (su alcuni compilatori)*
- **con 32 bit, [ 0 …. 4.294.967.295 ]**
  *In C: unsigned int (su alcuni compilatori)*
  *In C: unsigned long int (su molti compilatori)*

## OPERAZIONI ED ERRORI

**La rappresentazione binaria rende possibile fare *addizioni e sottrazioni con le usuali regole algebriche***

Esempio:

```
5 +        0101
3 =        0011
---        --------
8          1000
```



## ERRORI NELLE OPERAZIONI

Esempio (supponendo di avere solo 7 bit per la rappresentazione)

```
 60 +       0111100
 75 =       1100011
-----      --------
135        10011111
```

**Errore!**
**Massimo numero rappresentabile:**
**$2^7$-1 cioè 127**

- **Questo errore si chiama *overflow***
- **Può capitare *sommando due numeri dello stesso segno* il cui risultato non sia rappresentabile utilizzando il numero massimo di bit designati**

# ESERCIZIO RAPPRESENTAZIONE

Un elaboratore rappresenta numeri interi su ***8 bit*** dei quali ***7*** sono dedicati alla rappresentazione del modulo del numero e ***uno*** al suo ***segno***. Indicare come viene svolta la seguente operazione aritmetica:

59 – 27

in codifica binaria



# ESERCIZIO RAPPRESENTAZIONE

**Soluzione**

59 →0 0111011

-27 →1 0011011

Tra i (moduli dei) due numeri si esegue una sottrazione:

```
   0111011
-  0011011
   -------
   0100000
```

che vale 32 in base 10

# Differenze tra numeri binari

•Cosa avremmo dovuto fare se avessimo avuto 27-59 ???

•Avremmo dovuto invertire i due numeri, calcolare il risultato, e poi ricordarci di assegnare 1 al bit rappresentante il segno.

•Il procedimento è quindi complesso. Per ovviare a tale problema, si usa la notazione "complemento a 2", che permette di eseguire tutte le differenze tramite delle somme!

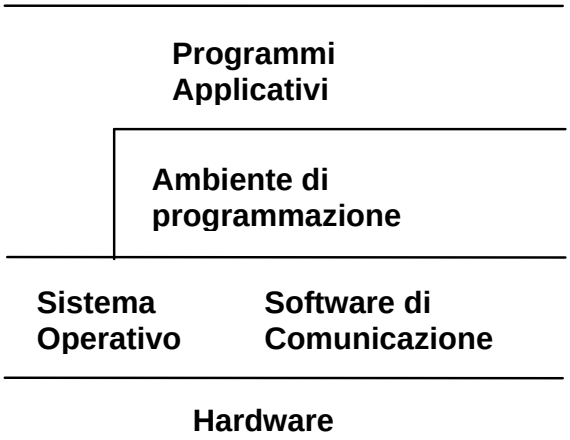


# IL SOFTWARE

**Software:**
insieme (complesso) di programmi

**Organizzazione a strati**, ciascuno con funzionalità di livello più alto rispetto a quelli sottostanti

Concetto di
***MACCHINA VIRTUALE***



# IL FIRMWARE

**Firmware:**
il confine fra hardware e software

**È uno strato di *micro-programmi***, *scritti dai costruttori*, che agiscono direttamente al di sopra dello strato hardware

Sono memorizzati su una speciale *memoria centrale permanente* (ROM, EPROM, …)



# IL SISTEMA OPERATIVO

Strato di programmi che opera *al di sopra di hardware e firmware* e **gestisce l'elaboratore**

Solitamente, è venduto insieme all'elaboratore

**Spesso si può scegliere tra *diversi sistemi operativi*** per lo stesso elaboratore, con diverse caratteristiche

**Esempi:**

- Windows 95/98/XP
- Windows NT/2000
- Linux v.2.6
- MacOs X
- Symbian
- Palm OS
- ...

# FUNZIONI DEL SISTEMA OPERATIVO

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

- gestione delle risorse disponibili
- gestione della memoria centrale
- organizzazione e gestione della memoria di massa
- interpretazione ed esecuzione di comandi elementari
- gestione di un sistema multi-utente

**Un utente "vede" l'elaboratore solo tramite il Sistema Operativo**
**→ il SO realizza una "macchina virtuale"**



# FUNZIONI DEL SISTEMA OPERATIVO

**Qualsiasi operazione di accesso a risorse** implicitamente richiesta da comando utente **viene esplicitata dal SO**

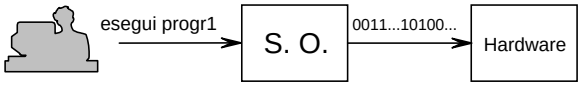
**Conseguenza**:
diversi SO possono realizzare *diverse macchine virtuali* **sullo stesso elaboratore fisico**

Attraverso il SO il livello di interazione fra utente ed elaboratore viene elevato:

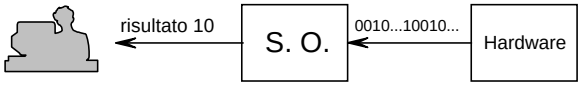
- senza SO: sequenze di bit
- con SO: comandi, programmi, dati

I sistemi operativi si sono evoluti nel corso degli ultimi anni (interfacce grafiche, Mac, Windows, ...)

# ESEMPIO




e viceversa:




| Utente: | Sistema Operativo: |
|---|---|
| "esegui progr1" | - input da tastiera<br>- ricerca codice di "progr1" su disco<br>- carica in memoria centrale codice e dati<br><elaborazione> |
| Utente: | Sistema Operativo: |
| "stampa 10" | - output su video |



# PROGRAMMI APPLICATIVI

**Risolvono problemi specifici degli utenti:**

- *word processor*: elaborazione di testi (*es. MSWord*)
- *fogli elettronici*: gestione di tabelle, calcoli e grafici (*es. MSExcel*)
- *database*: gestione di archivi (*es. MSAccess*)
- *suite* (integrati): collezione di applicativi capaci di funzionare in modo integrato come un'applicazione unica (*es. Open Office*)

- Sono scritti in **linguaggi di programmazione** di alto livello
- Risentono in misura ridotta delle caratteristiche della architettura dell'ambiente sottostante (***portabilità***)

# AMBIENTI DI PROGRAMMAZIONE

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

**Sviluppo di un programma**

- Affinché un programma scritto in un qualsiasi linguaggio di programmazione sia comprensibile (e quindi eseguibile) da un calcolatore, *occorre* ***tradurlo*** dal linguaggio originario al linguaggio della macchina

- Questa operazione viene normalmente svolta da speciali programmi, detti ***traduttori***