

INFORMATICA

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

1

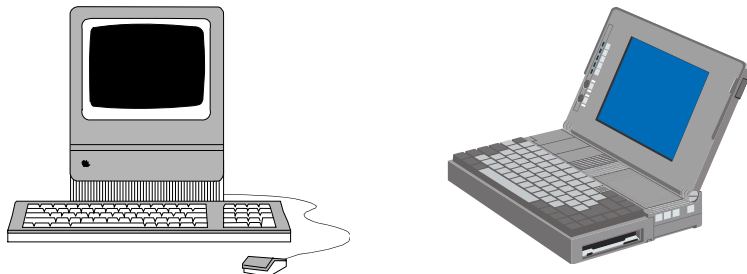
L’informatica comprende:

- 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
-

2

ELABORATORE ELETTRONICO (“COMPUTER”)

Strumento per la rappresentazione e l’elaborazione delle informazioni



3

L’ELABORATORE

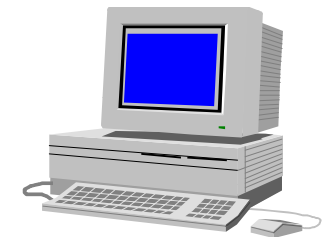
Componenti principali

- Unità centrale
- Video (“monitor”)
- Tastiera e Mouse
- Lettore CD/DVD
- Dischi fissi (“hard disk”)
- Dischetti (“floppy”)/USB stick

Componenti accessori

- Stampante
- Modem/interfaccia di rete
- Scanner
- Tavolette grafiche

...

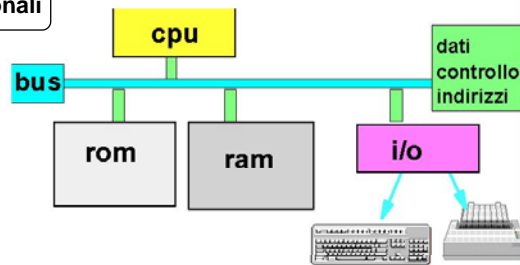


HARDWARE

4

ARCHITETTURA DI UN ELABORATORE

Unità funzionali



Ispirata al modello della **Macchina di Von Neumann** (Princeton, Institute for Advanced Study, anni '40)

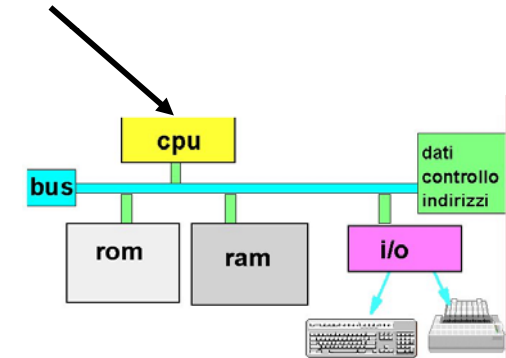
Macchina di Von Neumann:

- Non distingue fra RAM e ROM
- Non aveva un bus ma collegamenti punto-punto

HARDWARE

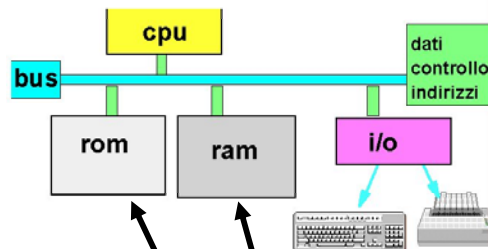
CPU (Central Processing Unit), o **Processore**

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



6

HARDWARE



RAM & ROM

- Dimensioni relativamente limitate
- Accesso molto rapido

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

7

HARDWARE

ATTENZIONE

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

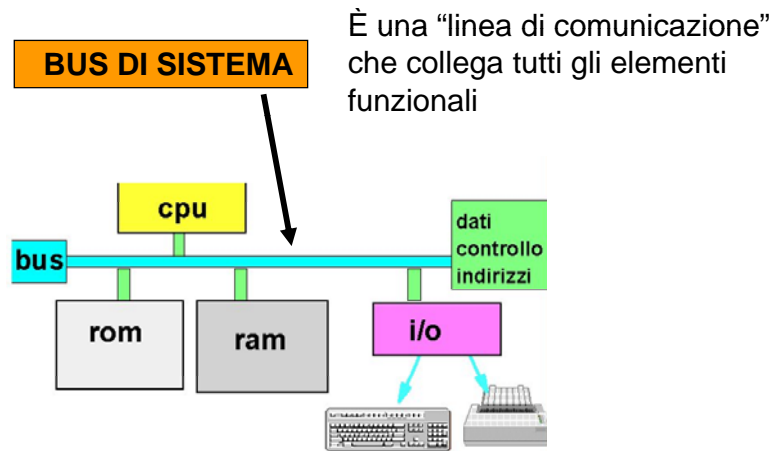
- usata per memorizzare dati e programmi

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 (tipicamente *firmware*)

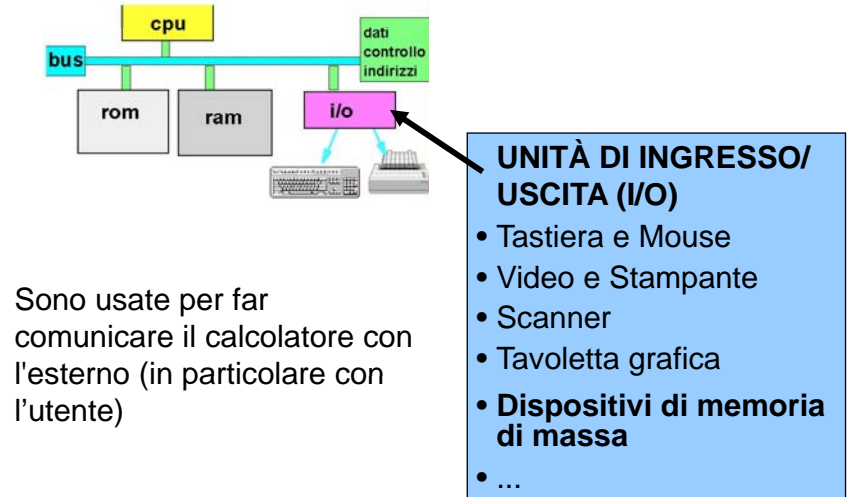
8

HARDWARE



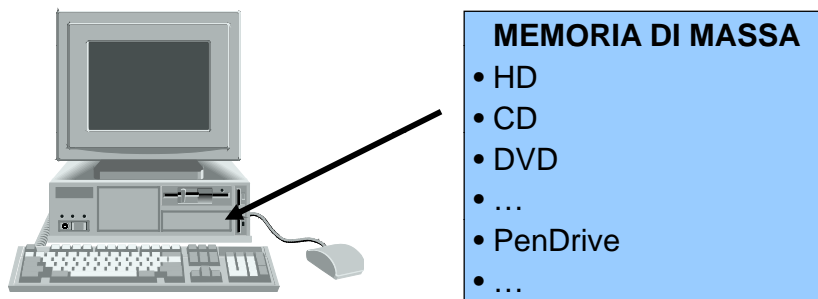
9

HARDWARE



10

HARDWARE



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

11

TECNOLOGIA DIGITALE

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

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

- tensione alta (V_H , ad es. 5V o 3.3V)
- 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:** $0 \leftrightarrow V_H$, $1 \leftrightarrow V_L$

12

TECNOLOGIA DIGITALE (segue)

Dati e 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**

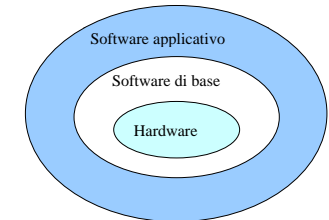
13

SOFTWARE

Software: programmi che vengono eseguiti dal sistema

Distinzione fra:

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



14

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**

15

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, ...)

16

IL SISTEMA OPERATIVO

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

Spesso è venduto insieme all'elaboratore

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
- ...



17

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 (SO)
→ il SO realizza una "macchina virtuale"

18

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*

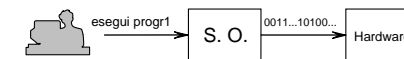
Attraverso il SO il livello di interazione fra utente ed elaboratore viene elevato:

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

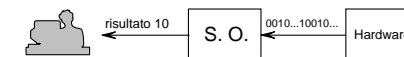
I sistemi operativi si sono evoluti nel corso degli anni (interfacce grafiche, Mac, Windows, ...)

19

ESEMPIO



e viceversa:



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

20

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**)

21

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**

22

L'ELABORATORE ELETTRONICO

- Il calcolatore elettronico è uno strumento in grado di eseguire insiemi di **azioni** ("mosse") **elementari**
- le azioni vengono **eseguite** su oggetti (**dati**) per **produrre** altri oggetti (**risultati**)
- l'esecuzione di azioni viene richiesta all'elaboratore attraverso **frasi** scritte in un qualche **linguaggio** (**istruzioni**)

23

PROGRAMMAZIONE

L'attività con cui si predispongono l'elaboratore a **eseguire un particolare insieme di azioni su particolari dati**, allo scopo di **risolvere un problema**



24

ALCUNE DOMANDE FONDAMENTALI

- Quali istruzioni esegue un elaboratore?
- Quali problemi può risolvere un elaboratore?
- *Esistono problemi che un elaboratore non può risolvere?*
- Che ruolo ha il linguaggio di programmazione?

25

PROBLEMI DA RISOLVERE

I problemi che siamo interessati a risolvere con l'elaboratore sono di **natura molto varia**:

- Dati due numeri trovare il **maggiore**
- Dato un elenco di nomi e relativi numeri di telefono **trovare** il numero di telefono di una determinata persona
- Dati a e b, **risolvere l'equazione** $ax+b=0$
- Stabilire se una parola viene **alfabeticamente** prima di un'altra
- **Somma** di due numeri interi
- Scrivere tutti gli n per cui l'equazione: $X^n + Y^n = Z^n$ ha soluzioni intere (**problema di Fermat**)
- **Ordinare** una lista di elementi
- Calcolare il **massimo comune divisore** fra due numeri dati
- Calcolare il **massimo** in un insieme

26

RISOLUZIONE DI PROBLEMI

- La descrizione del problema non fornisce (in generale) un metodo per risolverlo
 - Affinché un problema sia risolvibile è necessario che la sua definizione sia chiara e completa
- **Non tutti** i problemi sono risolvibili attraverso l'uso del calcolatore. Esistono classi di problemi per le quali la soluzione automatica non è proponibile. Ad esempio:
 - se il problema presenta infinite soluzioni
 - per alcuni dei problemi **non è stato trovato un metodo risolutivo**
 - per alcuni problemi è stato dimostrato che **non esiste un metodo risolutivo automatizzabile**

27

RISOLUZIONE DI PROBLEMI

- Noi ci concentreremo sui problemi che, ragionevolmente, **ammettono un metodo risolutivo**
➡ **funzioni calcolabili**
- Uno degli obiettivi del corso è presentare le **tecnologie** e le **metodologie di programmazione**
 - **Tecnologie**: strumenti per lo sviluppo di programmi
 - **Metodologie**: metodi per l'utilizzo corretto ed efficace delle tecnologie di programmazione

28

RISOLUZIONE DI PROBLEMI

La risoluzione di un problema è il processo che dato un problema e individuato un opportuno metodo risolutivo, trasforma i dati iniziali nei corrispondenti risultati finali

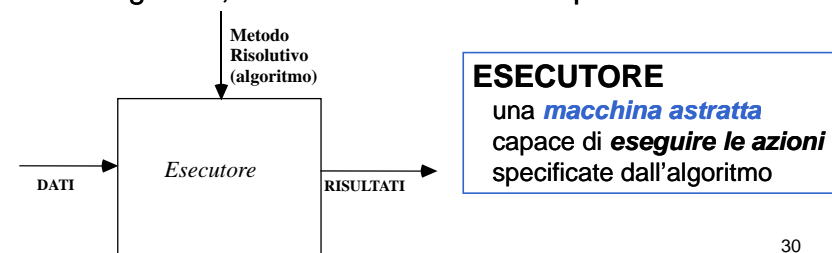
Affinché la risoluzione di un problema possa essere realizzata attraverso l'uso del calcolatore, tale processo deve poter essere definito come *sequenza di azioni elementari*

29

ALGORITMO

Un algoritmo è una sequenza **finita** di mosse che risolve **in un tempo finito** una *classe* di problemi

L'esecuzione delle azioni *nell'ordine specificato dall'algoritmo* consente di ottenere, a partire dai dati di ingresso, i risultati che risolvono il problema



30

ALGORITMI: PROPRIETÀ

- **Eseguibilità**: ogni azione deve essere *eseguibile* dall'esecutore *in un tempo finito*
- **Non-ambiguità**: ogni azione deve essere *univocamente interpretabile* dall'esecutore
- **Finitezza**: il numero totale di azioni da eseguire, per ogni insieme di dati di ingresso, deve essere finito

31

ALGORITMI: PROPRIETÀ (2)

Quindi, l'algoritmo deve:

- essere *applicabile a qualsiasi insieme di dati di ingresso* appartenenti al **dominio di definizione** dell'algoritmo
- essere costituito da operazioni appartenenti ad un determinato **insieme di operazioni fondamentali**
- essere costituito da **regole non ambigue**, cioè interpretabili in modo **univoco** qualunque sia l'esecutore (persona o "macchina") che le legge

32

ALGORITMI E PROGRAMMI

- Ogni elaboratore è una macchina in grado di eseguire azioni elementari su oggetti detti **DATI**
- L'esecuzione delle azioni è richiesta all'elaboratore tramite comandi elementari chiamati **ISTRUZIONI** espresse attraverso un opportuno formalismo: il **LINGUAGGIO di PROGRAMMAZIONE**
- La formulazione testuale di un algoritmo in un linguaggio comprensibile a un elaboratore è detta **PROGRAMMA**

33

PROGRAMMA

Un **programma** è un **testo** scritto in accordo alla **sintassi** e alla **semantica** di un linguaggio di programmazione

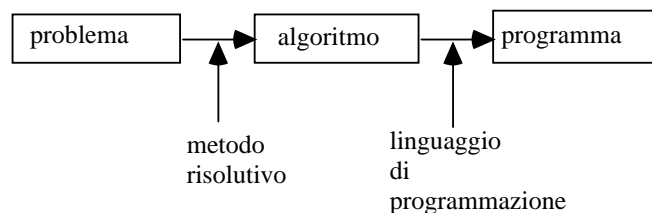
Un **programma** è la **formulazione testuale**, in un certo linguaggio di programmazione, di un **algoritmo** che risolve un dato *problema*

34

ALGORITMO & PROGRAMMA

Passi per la risoluzione di un problema:

- individuazione di un procedimento risolutivo
- scomposizione del procedimento in un insieme ordinato di azioni ➡ **ALGORITMO**
- rappresentazione dei dati e dell'algoritmo attraverso un formalismo comprensibile dal calcolatore ➡ **LINGUAGGIO DI PROGRAMMAZIONE**



35

UN ESEMPIO DI PROGRAMMA (in linguaggio C)

```
main() {
    int A, B;
    printf("Immettere due numeri: ");
    scanf("%d %d", &A, &B);
    printf("Somma: %d\n", A+B);
}
```

36

ALGORITMI: ESEMPI

• Soluzione dell'equazione $ax+b=0$

- leggi i valori di a e b
- calcola $-b$
- dividi quello che hai ottenuto per a e chiama x il risultato
- stampa x

• Calcolo del massimo di un insieme

- Scegli un elemento come massimo provvisorio max
- Per ogni elemento i dell'insieme: se $i > max$ eleggi i come nuovo massimo provvisorio max
- Il risultato è max

NOTA: si utilizzano **VARIABILI**, ossia nomi simbolici usati nell'algoritmo per denotare dati

37

ALGORITMI: ESEMPI

Stabilire se una parola P viene alfabeticamente prima di una parola Q

- leggi P, Q
- **ripeti quanto segue:**
 - **se** prima lettera di $P <$ prima lettera Q
 - **allora** scrivi vero
 - **altrimenti se** prima lettera $P >$ Q
 - **allora** scrivi falso
 - **altrimenti** (le lettere sono =)
togli da P e Q la prima lettera
- **fino** a quando hai trovato le prime lettere diverse

– *Nota: funziona solo con P e Q di uguale lunghezza e con parole diverse*

– *Esercizio proposto: rilassare tali condizioni*

38

ALGORITMI: ESEMPI

• Somma degli elementi dispari di un insieme

Detto INS l'insieme di elementi considero un elemento X di INS alla volta senza ripetizioni. Se X è dispari, sommo X a un valore S inizialmente posto uguale a 0. Se X è pari non compio alcuna azione

• Somma di due numeri X e Y

Incrementare il valore di Z , inizialmente posto uguale a X per Y volte

- poni $Z = X$
- poni $U = 0$
- finché U è diverso da Y
 - incrementa Z ($Z=Z+1$)
 - incrementa U ($U=U+1$)
- Il risultato è Z

39

ALGORITMI EQUIVALENTI

Due algoritmi si dicono **equivalenti** quando:

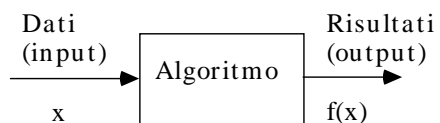
- hanno lo stesso **dominio di ingresso**
- hanno lo stesso **dominio di uscita**
- in corrispondenza degli **stessi valori del dominio di ingresso producono gli stessi valori nel dominio di uscita**

40

ALGORITMI EQUIVALENTI (2)

Due algoritmi *equivalenti*

- forniscono lo **stesso risultato**
- ma possono avere ***diversa efficienza***
- e possono essere **profondamente diversi!**



41

ALGORITMI EQUIVALENTI (3)

ESEMPIO: calcolo del M.C.D. fra due interi M, N

- **Algoritmo 1**
 - Calcola l'insieme A dei divisori di M
 - Calcola l'insieme B dei divisori di N
 - Calcola l'insieme C dei divisori comuni = $A \cap B$
 - Il risultato è il massimo dell'insieme C
- **Algoritmo 2 (di Euclide)**

$$\text{MCD}(M,N) = \begin{cases} M \text{ (oppure } N) & \text{se } M=N \\ \text{MCD}(M-N, N) & \text{se } M>N \\ \text{MCD}(M, N-M) & \text{se } M<N \end{cases}$$

42

ALGORITMI EQUIVALENTI (4)

ESEMPIO: calcolo del M.C.D. fra due interi M, N

Algoritmo 2 (di Euclide)

Finché $M \neq N$:

- se $M > N$, sostituisci a M il valore $M' = M - N$
- altrimenti sostituisci a N il valore $N' = N - M$
- Il Massimo Comune Divisore è il valore finale ottenuto quando M e N diventano uguali

$$\text{MCD}(M,N) = \begin{cases} M \text{ (oppure } N) & \text{se } M=N \\ \text{MCD}(M-N, N) & \text{se } M>N \\ \text{MCD}(M, N-M) & \text{se } M<N \end{cases}$$

43