

INFORMATICA

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

1

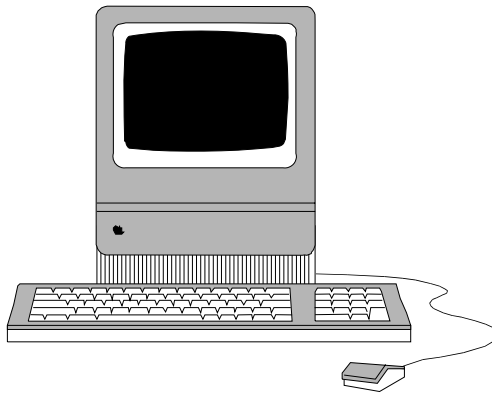
L’informatica comprende:

- 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

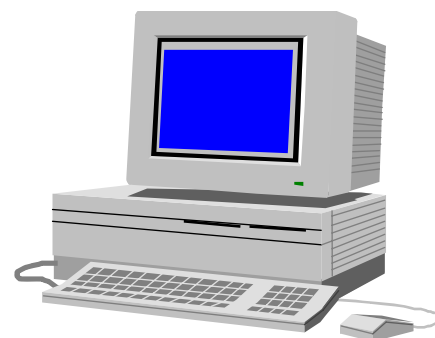
Componenti principali

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

Componenti accessori

- Stampante
- Modem
- Scanner
- Tavolette grafiche

...



HARDWARE

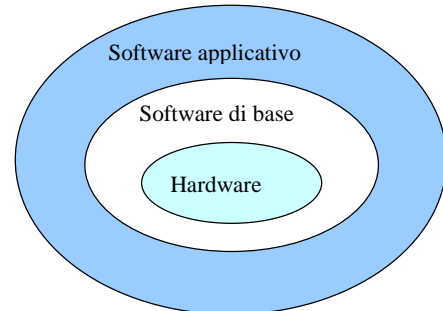
4

SOFTWARE

Software: programmi che vengono eseguiti dal sistema

Distinzione fra:

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



5

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**



6

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, ...)

7

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
- ...



8

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”**

9

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

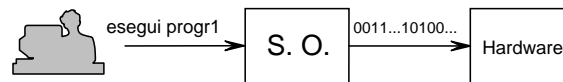
Attraverso il SO il livello di interazione fra utente ed elaboratore viene elevato:

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

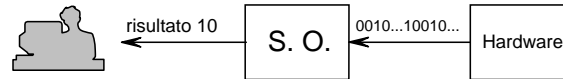
I sistemi operativi si sono evoluti nel corso degli ultimi anni (interfacce grafiche, Mac, Windows, ...)

10

ESEMPIO



e viceversa:



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

11

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**)

12

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*

13

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*)

14

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



15

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?

16

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

17

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

18

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

19

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

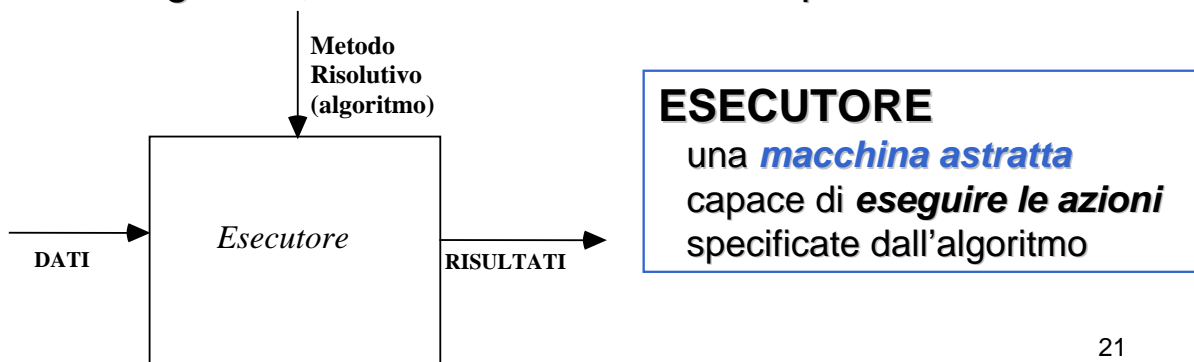
Affinché la risoluzione di un problema possa essere realizzata attraverso l'uso del calcolatore, tale processo deve poter essere definito come **sequenza di azioni elementari**

20

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



21

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

22

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

23

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**

24

PROGRAMMA

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

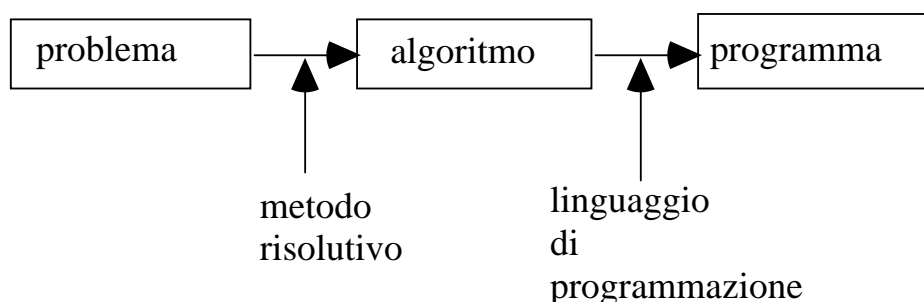
Un **programma** è la **formulazione testuale**, in un certo linguaggio di programmazione, di un **algoritmo** che risolve un dato *problema*

25

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**



26

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);
}
```

27

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

28

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*

29

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

30

ALGORITMI EQUIVALENTI

Due algoritmi si dicono *equivalenti* quando:

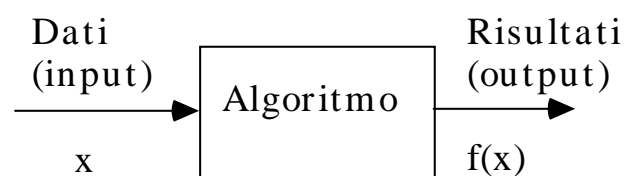
- hanno lo stesso **dominio di ingresso**
- hanno lo stesso **dominio di uscita**
- in corrispondenza degli **stessi valori del dominio di ingresso producono gli stessi valori nel dominio di uscita**

31

ALGORITMI EQUIVALENTI (2)

Due algoritmi *equivalenti*

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



32

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)**

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

33

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

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

34