Corso di Potenziamento di Informatica

Riccardo Ortale

Introduzione al Corso

OBIETTIVI DEL CORSO

- Aspetti concettuali
  - Illustrare i concetti fondamentali dell'informatica
  - Fornire conoscenze di base sugli elaboratori elettronici e le reti informatiche
- Aspetti pratici
  - Utilizzo di un elaboratore
  - Uso di strumenti di produttività individuale (elaborazione testi, gestione e analisi di dati)
  - Utilizzo di programmi di reperimento di informazioni su Internet

PROGRAMMA DEL CORSO - I

- ***Parte teorica (aula)***
  - Problema, Algoritmo, Programma
  - Codifica dell'Informazione
  - Architettura del Calcolatore
    - ✓ Macchina di Von Neumann, Memorie, Periferiche
  - Sistemi Operativi
    - ✓ Macchine virtuali, Classificazione dei S.O.
  - Reti di Calcolatori
    - ✓ Topologie, Protocolli, Trasmissione, TCP/IP, Internet, Web

PROGRAMMA DEL CORSO - II

- ***Parte pratica (laboratorio)***
  - Sistemi Operativi: *Windows*
  - Sistemi di videoscrittura: *Word*
  - Fogli elettronici: *Excel*

LIBRI DI TESTO

- **Testi di riferimento**
  - D. Sciuto, G. Buonanno, W. Fornaciari, L. Mari. *Introduzione ai sistemi informatici.* McGraw-Hill, 2005.
  - *La guida McGraw-Hill alla patente europea del computer*, 2003.

- **Testi di consultazione**
  - D. Curtin et al., *Informatica di Base – II ed.*, McGraw-Hill, 2005.
  - *Collana ECDL*, Apogeo 2002*:*
    - ✓ S. Rubini: *Moduli 1 e 2 - Concetti di base e Gestione dei file*,
    - ✓ S. Rubini: *Mod. 3 - Elaborazione testi*,
    - ✓ S. Rubini: *Mod. 4 - Foglio elettronico*,
    - ✓ S. Rubini: *Mod. 7 - Reti informatiche*.

INDIRIZZI UTILI

- Studio docente:
  - ICAR-CNR c/o DEIS, Cubo 41C, I° Piano

- Indirizzo di posta elettronica:

  riccardo.ortale@icar.cnr.it

- Sito del corso (avvisi, materiale, …):

  http://staff.icar.cnr.it/ortale

CONCETTI INTRODUTTIVI

Informatica
Hardware & Software
Problema, Algoritmo, Programma

INFORMAZIONE E COMUNICAZIONE

- **Informazione**

  Notizia, dato o elemento che consente di avere conoscenza più o meno esatta di fatti, situazioni, modi di essere, diminuendo l'incertezza

- **Messaggio**

  Tutto ciò che porta informazione

- **Comunicazione**

  Scambio di informazione

INFORMATICA

- **INFORMATICA = Scienza della rappresentazione e dell'elaborazione dell'informazione**

  L'informatica studia le caratteristiche dell'informazione e i modi di usarla, immagazzinarla, trasportarla e manipolarla in modo automatico

- L'informatica ha due anime:
  - *tecnologica*: i calcolatori elettronici e i sistemi che li utilizzano,
  - *metodologica*: i metodi per la soluzione di problemi e la gestione delle informazioni.

ELABORATORE ELETTRONICO

- **Elaboratore elettronico (o "*computer*" o *"calcolatore"*):**

  è una *macchina* per la rappresentazione, la memorizzazione e l'elaborazione e trasmissione delle informazioni

- La prima decomposizione di un calcolatore è relativa alle seguenti componenti:

  - **Hardware**

    la struttura fisica del calcolatore, costituita da componenti elettronici ed elettromeccanici

  - **Software**

    l'insieme dei programmi che consentono all'hardware di svolgere dei compiti utili

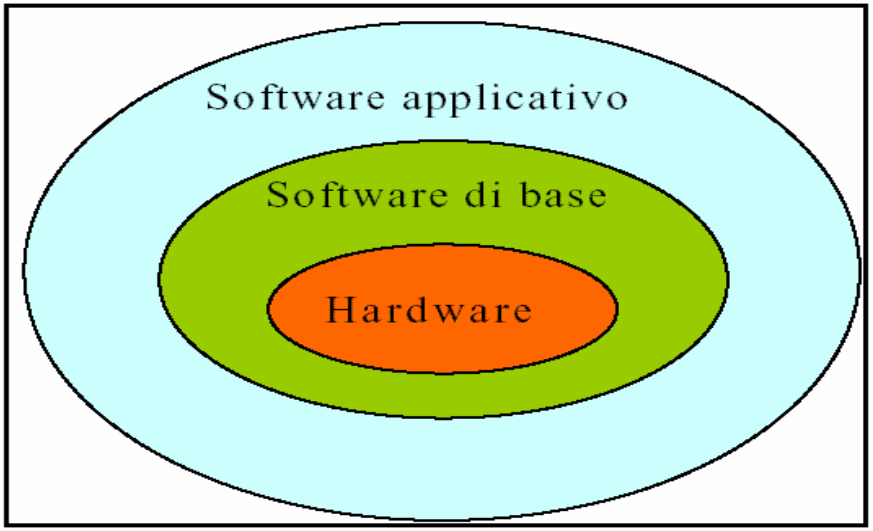
HARDWARE & SOFTWARE

- L'**hardware** è la *macchina reale*
  - Le operazioni (chiamate *istruzioni*) che l'hardware sa eseguire direttamente costituiscono il *linguaggio macchina* del calcolatore
  - Le istruzioni del linguaggio macchina sono molto semplici, ma il calcolatore può eseguirle in modo molto efficiente
- Il **software** ha lo scopo di mostrare ai suoi utenti il calcolatore come una *macchina virtuale* (non esistente fisicamente) più semplice da usare rispetto all'hardware sottostante

SOFTWARE

Programmi che vengono eseguiti dal sistema:

- **Software di Base** (Sistema Operativo)
- **Software Applicativo**



SOFTWARE: MACCHINE VIRTUALI

- **Macchine virtuali**
  - Semplificano la comunicazione fra uomo e hardware
  - Le diverse macchine e i relativi insiemi di operazioni sono via via più astratti: più vicini alla logica dell'utente e più lontani dalla logica del calcolatore come dispositivo elettronico
  - Alla fine, comunque, l'unico responsabile dell'esecuzione del software è l'hardware disponibile

- Il **software di base** permette una più semplice interazione con le componenti hardware (memorie, periferiche, …)

- Il **software applicativo** mostra all'utente il calcolatore come una macchina virtuale utilizzabile per la **risoluzione di problemi**

I PROBLEMI

I problemi affrontati dalle applicazioni informatiche sono di natura *molto varia:*

- Trovare il maggiore fra due numeri
- Dato un elenco di nomi e numeri di telefono, trovare il numero di una data persona
- Dati *a* e *b*, risolvere l'equazione $ax+b=0$
- Stabilire se una parola precede alfabeticamente un'altra
- Ordinare un elenco di nomi
- Creare, modificare e alterare suoni
- Analizzare, riconoscere e modificare immagini .....
- Gestione delle aziende (private e pubbliche)
- Supportare operazioni di commercio elettronico

I PROBLEMI

La descrizione del problema non indica direttamente (in genere) un modo per ottenere il risultato voluto

Differenza tra *specifica di un **problema*** e *specifica del **processo di risoluzione***

Risoluzione di un problema

processo che:

- dato un *problema*
- individua un opportuno *metodo risolutivo (algoritmo)*

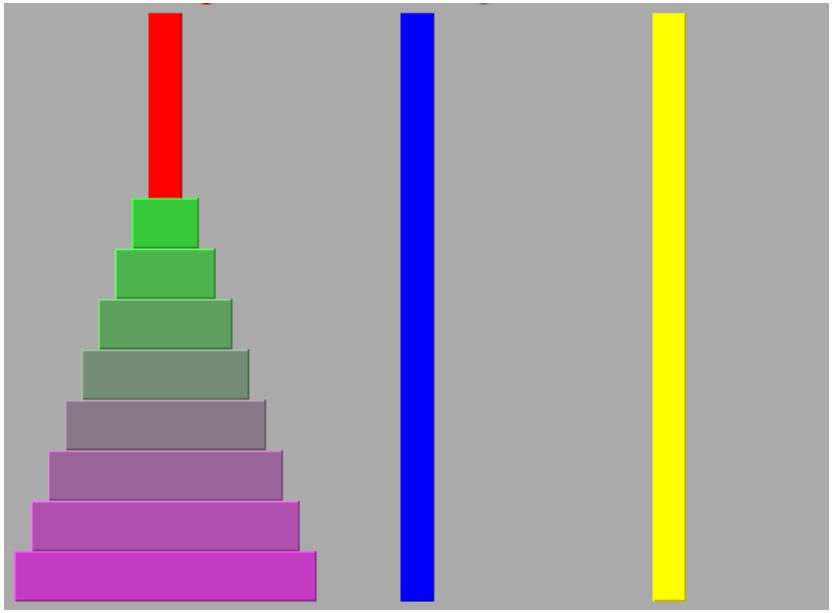
RISOLUZIONE DI UN PROBLEMA

TEORIA DEGLI ALGORITMI E DELLA COMPLESSITA'

- La teoria degli algoritmi
  - Fornisce algoritmi per risolvere problemi
  - Classifica la complessità dei problemi
- Trattabili (n^k)
  - Ricerca del massimo, ordinare *n* numeri
- Intrattabili (k^n)
  - Cricca
- Non risolvibili
  - Ci sono problemi non risolvibili da nessun modello di calcolo reale o astratto
  - Esempio: data una funzione f : N$\rightarrow$ N, stabilire se f(x) è costante per ogni valore di x

PROBLEMI INTRATTABILI: TORRI DI HANOI

- n = 64 (dischi)
- Mosse = 2^n-1 = 2^{64}-1 = 18'446'744'073'709'551'615
- 1 mossa al secondo = 5’845’580’504 secoli
- Età della Terra = 46’000’000 secoli
- Ci restano circa 5 miliardi e 800 milioni di secoli!



ALGORITMI

La parola **algoritmo** deriva dal nome di un autore scientifico persiano del IX secolo

- **Abu Ja'far Mohammed ibn Musa al-Khowarizmi** scrisse, circa nell'825, il trattato "*Kitab **al jabr** w'al-muqabala*" (forse *regole di trasposto e semplificazione*) dove descrisse delle regole per la semplificazione delle equazioni.
- **algebra** deriva da **al jabr** (parte del titolo del trattato)
- **algoritmo** deriva da **Khowarizmi** (ultima parte del nome dell'autore, indicante la città di nascita)

Il termine originario era *agorismo*, trasformato in *algoritmo* per analogia con *aritmetica*.

ALGORITMI

Si definisce *algoritmo* una *sequenza di azioni*
che trasformi i dati iniziali in un numero finito di passi,
elementari e non ambigui, per giungere al risultato finale.

Questa sequenza di azioni è valida
per un insieme di dati iniziali ben definito
e può essere eseguita da un opportuno esecutore.

ALGORITMI: PROPRIETÀ FONDAMENTALI

Non si può risolvere un problema senza prima fissare un insieme di “azioni”, di “mosse elementari” possibili per l'esecutore.
Bisogna conoscerne le caratteristiche, le mosse che sa eseguire ed il linguaggio che sa capire

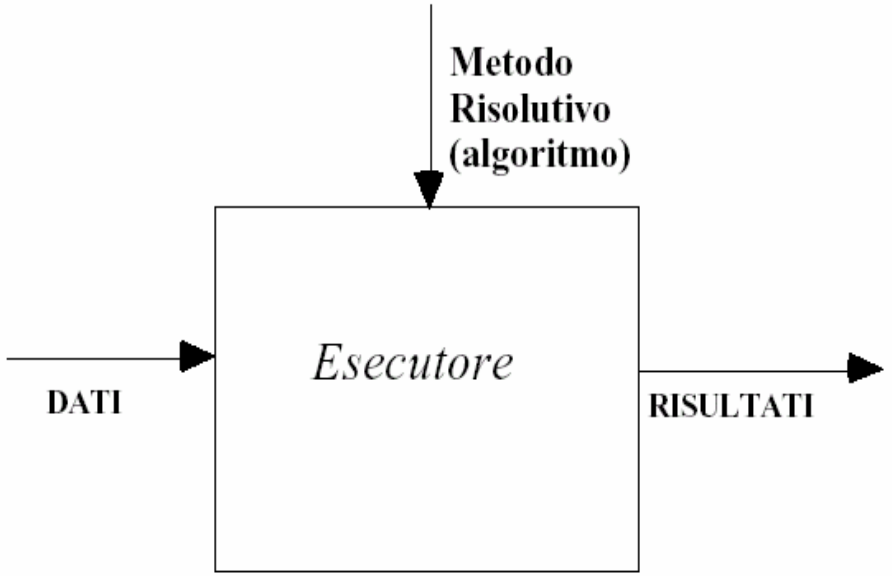
- **Non-ambiguità**: ogni azione deve essere *univocamente interpretabile* dall'esecutore,
- **Eseguibilità**: ogni azione deve essere *eseguibile in un tempo finito* da parte dell’esecutore dell’algoritmo,
- **Finitezza**: per ogni insieme di dati di ingresso, il numero totale di azioni da eseguire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

Proprietà desiderabile:

- **Efficienza**: deve risolvere il problema utilizzando al meglio le risorse a disposizione

ALGORITMI: ESECUTORE

Esecutore: una *macchina astratta* capace di *eseguire le azioni* specificate dall'algoritmo.



Rappresentazione degli algoritmi

1. Linguaggio naturale ← Linguaggi informali
2. Diagrammi di flusso ←
3. Pseudo-codice ← Linguaggi semi-formali
4. Linguaggio di programmazione ← Linguaggi formali

Variabili

- Un algoritmo descrive non una singola istanza di un problema, ma piuttosto un'intera classe di problemi.

  Per questo motivo, le operazioni comprese nell'algoritmo fanno riferimento a variabili, vale a dire "contenitori" per dati.

- Ogni variabile ha
  - nome
  - valore (iniziale e corrente)

- Esempio:
  - N = 2 (il valore iniziale di N è 2)
  - N = N + 3 (il valore corrente di N è 5)

Esempio di algoritmo: calcolo del MCD

Problema: calcolo del Massimo Comun Divisore (MCD) fra due interi M ed N

Algoritmo n° 1

1. Calcola l'insieme A dei divisori di M
2. Calcola l'insieme B dei divisori di N
3. Calcola l'insieme C dei divisori comuni di M e N, $C = A \cap B$
4. Il risultato è il massimo dell'insieme C

ESEMPIO: calcolo del MCD

Algoritmo n° 2

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

Strategia risolutiva:

- Finchè $M \neq N$:

  se $M>N$, sostituisci a M il valore $M' = M-N$

  altrimenti sostituisci a N il valore $N' = N-M$

- Il Massimo Comun Divisore è il valore finale ottenuto quando M e N diventano uguali

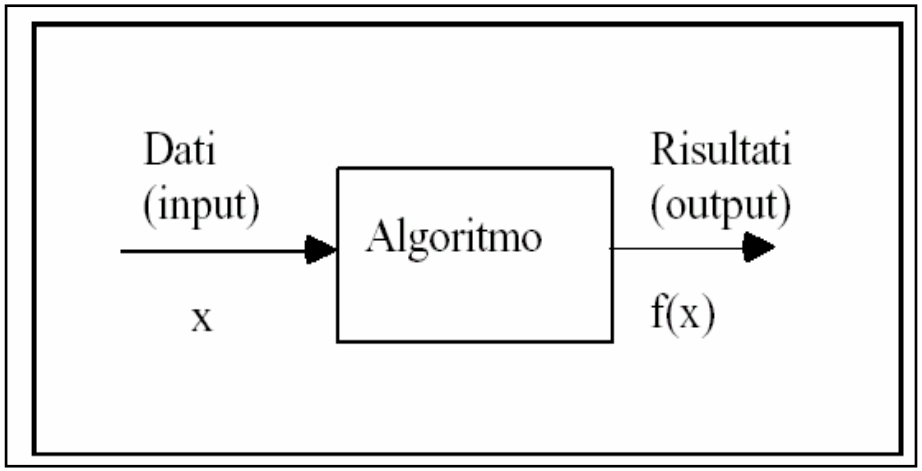
ESEMPIO: calcolo del MCD

Calcoliamo il MCD di M = 24 e N = 14.

1. M=24, N=14
   24>14 → M = 24-14 = 10
2. M=10, N=14
   10<14 → N = 14-10 = 4
3. M=10, N=4
   10>4 → M = 10-4 = 6
4. M=6, N=4
   6>4 → M = 6-4 = 2
5. M=2, N=4
   2<4 → N = 4-2 = 2
6. M=2, N=2
   2=2 → "il MCD di 24 e 14 è 2"

ALGORITMI EQUIVALENTI

- In generale un algoritimo può essere visto come una **funzione** da un dominio di ingresso (***input)*** ad dominio di uscita (***output***)



- Due algoritmi si dicono **equivalenti** quando:
  - hanno stesso dominio di ingresso e stesso dominio di uscita;
  - in corrispondenza degli stessi valori nel dominio di ingresso producono gli stessi valori nel dominio di uscita.

ESEMPIO: calcolo del MCD

Algoritmo n° 3

Dati due interi M e N ($M \geq N$)

1. Dividi M per N, e sia R il resto della divisione;
2. Se R=0 allora termina. Il Massimo Comun Divisore è N;
3. Altrimenti assegna a M il valore di N ed a N il valore del resto R
   e torna al punto 1.

ESEMPIO: calcolo del MCD

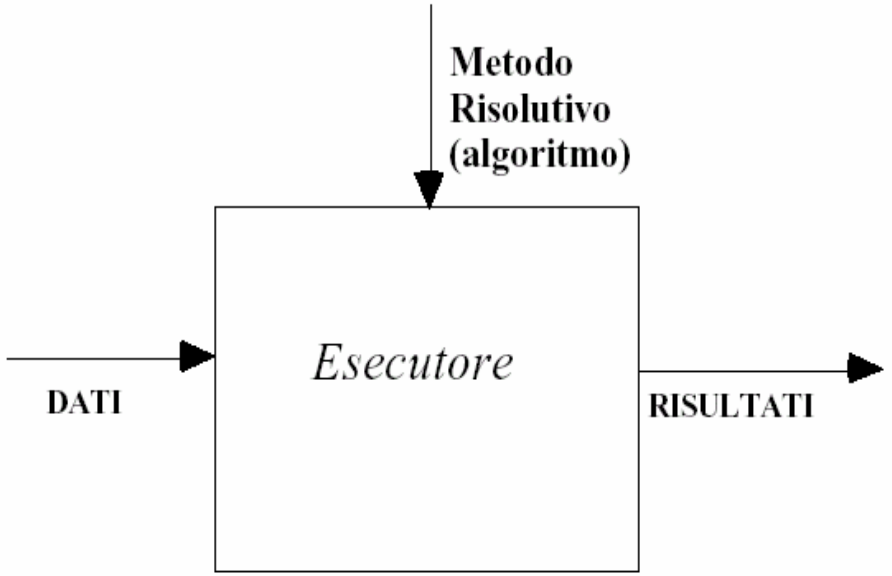
Calcoliamo il MCD di M = 24 e N = 14.

1. **M=24, N=14**
   24/14 = 1,R=10 → M=N=14, N=R=10
2. **M=14, N=10**
   14/10 = 1,R=4 → M=N=10, N=R=4
3. **M=10, N=4**
   10/4 = 2, R=2 → M=N=4, N=R=2
4. **M=4, N=2**
   4/2 = 2, R=0 → "il MCD di 24 e 14 è 2"

ALGORITMI: ESECUZIONE

- L'**esecuzione** delle azioni *nell'ordine specificato dall'algoritmo* consente di ottenere, a partire dai dati di ingresso, i risultati che risolvono la particolare *istanza* del problema.

Esecutore: una *macchina astratta* capace di *eseguire le azioni* specificate dall'algoritmo.



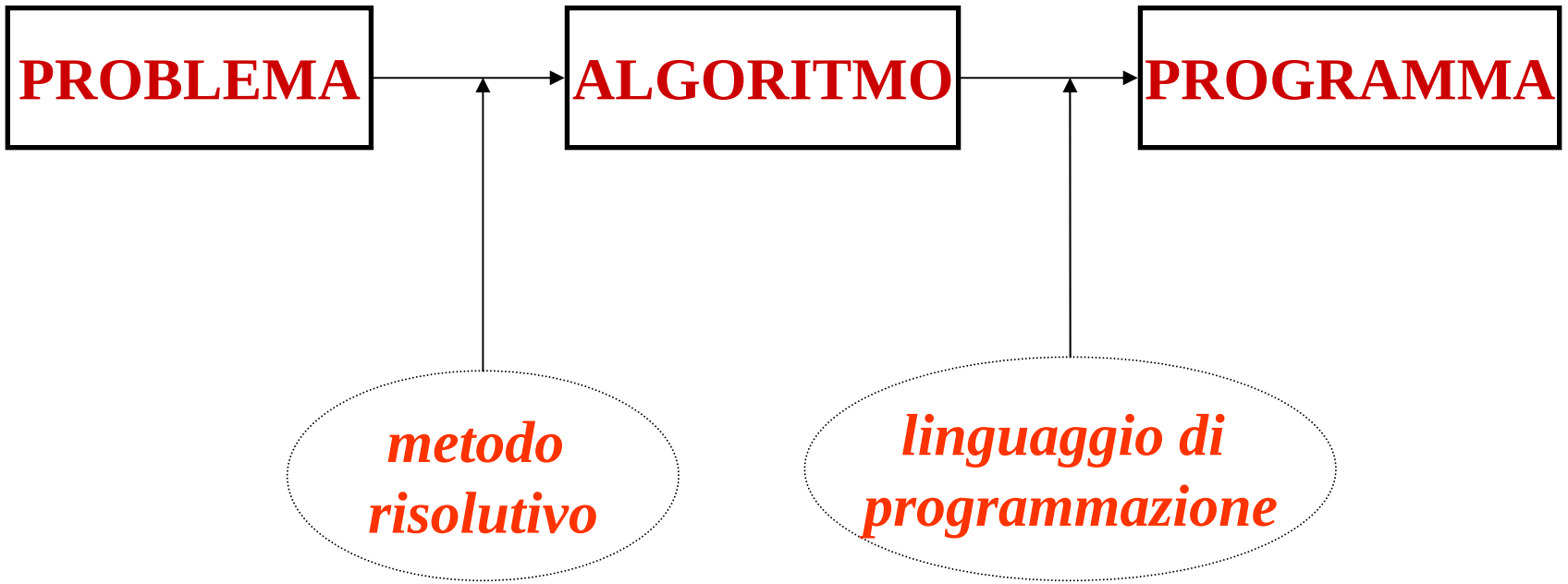
ALGORITMI: ESECUZIONE TRAMITE COMPUTER

La **programmazione** è l'attività con cui si predispone l'elaboratore ad eseguire un *particolare insieme di azioni* su una *particolare tipologia di dati*, allo scopo di *risolvere un problema*.

RISOLUZIONE DI PROBLEMI CON L'ELABORATORE ELETTRONICO

- Ogni elaboratore è una macchina in grado di eseguire **azioni** elementari su **dati**
- L'esecuzione delle azioni elementari è richiesta all'elaboratore tramite comandi chiamati **istruzioni**
- Le istruzioni sono espresse attraverso **frasi** di un opportuno **linguaggio di programmazione**
- Un **programma** non è altro che la formulazione testuale di un algoritmo in un linguaggio di programmazione

ALGORITMI E PROGRAMMI



LINGUAGGI AD ALTO LIVELLO

È opportuno impostare la soluzione di un problema a partire dalle "mosse elementari" del **linguaggio macchina**?

- SI, per risolvere il problema *con efficienza*
- NO, se la macchina di partenza ha mosse di livello *troppo basso* (difficile progettare un algoritmo)

Linguaggi di Programmazione ad Alto Livello (di astrazione)

- le istruzioni corrispondono ad operazioni più complesse
- esempi: Pascal, Basic, C, C++, Java

Operazioni Elementari

- Operazioni aritmetiche e assegnamento di valori a singole variabili
  Es. C = (A + B) (C ← A + B)

- Condizioni sul valore di singole variabili
  Es. Se **A** > **B** allora …

- Lettura e scrittura di variabili
  Es. Leggi A oppure Stampa B

Le strutture di controllo: SEQUENZA

- Le istruzioni devono semplicemente essere eseguite nell'ordine in cui sono presentate:
  - solleva il ricevitore
  - componi il numero
  - …
- Una sequenza di istruzioni può essere "raggruppata" in modo da diventare una nuova macro-istruzione:
  - INIZIO
    - solleva il ricevitore
    - componi il numero
    - …
  - FINE

Le strutture di controllo: ITERAZIONE

- Le istruzioni devono essere eseguite ripetutamente fino a che non si verifica una determinata condizione

- Esempio:
  - RIPETI
    - componi il numero
  - FINO a che la linea è libera

Le strutture di controllo: CONDIZIONE

- Le istruzioni da eseguire sono determinate dalla valutazione di una data condizione

- Esempio:
  - SE il numero è libero
  - ALLORA
    - attendi la risposta
    - conduci la conversazione
    - deponi il ricevitore
  - ALTRIMENTI
    - deponi il ricevitore

Pseudo-codice: calcolo della potenza

- ***Problema***: Calcolare *a* elevato alla *n* (a^n)
  - Utilizziamo le ***variabili*** *K*, *Ris*
  - Inizialmente *Ris* = 1 e *K* = *n*

- ***Algoritmo***:
  - Fino a che *K* >0
    Calcola *Ris* • *a* e memorizzalo in *Ris*
    Decrementa *K*
  - *Correttezza*:
  - Al termine *Ris* = a^n

Programma: calcolo della potenza

Programma
(in Pascal):

```
Program Potenza;
Begin
    Integer Ris,N,A;
    Read(N);
    Read(A);
    Ris=1;
    While (N>0) Do
        Begin
            Ris=Ris*A;
            N=N-1;
        End;
    Write(Ris);
End.
```

Il programma in linguaggio ad alto livello va tradotto in **linguaggio macchina** (comprensibile all'elaboratore)

FINE