

# Introduzione al corso di Fondamenti di Informatica A

*Percorso di Preparazione agli Studi di Ingegneria*

Università degli Studi di Brescia

*Dott.ssa Daniela Fogli*

## Premessa

- Docenti
- Argomenti del corso e obiettivi
- Libri di testo e libri consigliati
- Orario del corso e orario di ricevimento
- Modalità d'esame
- Sito web del corso

## Argomenti e obiettivi del corso

- Principali **argomenti** del corso:
  - Il concetto intuitivo di calcolatore
  - La codifica dell'informazione
  - L'architettura dei sistemi di elaborazione
  - Il sistema operativo
  - Le applicazioni software
  - Cenni alle reti di calcolatori
  - Il progetto degli algoritmi
  - I linguaggi di programmazione
  - Programmazione in C
- **Obiettivi** del corso:
  - Acquisire gli elementi di base per comprendere il funzionamento degli odierni sistemi informatici

## Libri di testo e libri consigliati

### Libro di testo:

- **G. Guida, M. Giacomini, "Fondamenti di Informatica", Franco Angeli, 2006**

### Altri libri consigliati:

- D. Sciuto, G. Buonanno, L. Mari, "Introduzione ai sistemi informatici", Terza edizione, McGraw-Hill, 2005
- S. Ceri, D. Mandrioli, L. Sbattella, "Informatica arte e mestiere", McGraw Hill, 1999
- M. R. Laganà, M. Righi, F. Romani, "Informatica – Concetti e Sperimentazioni", Apogeo, 2003
- S. C. Sawyer, B. K. Williams, "Tecnologie dell'informazione e della comunicazione", McGraw Hill, 2002
- D. P. Curtin, K. Foley, K. Sen, C. Morin, "Informatica di base", McGraw Hill, 2002 (con CD-ROM)

## Orario del corso e di ricevimento

- Orario del corso:  
MERCOLEDI' 8.30 – 11.30 B04  
VENERDI' 8.30 – 11.30 V1 (ELAB2)
- Orario di ricevimento:  
MARTEDI' 15.30 – 17.30 (Fogli)  
GIOVEDI' 10.00 – 13.00 (Giacomin)  
LUNEDI' 16.30 – 18.15 (Saetti)

## Modalità d'esame

- Una **prova scritta** consistente in:
  - Domande (domande a quiz, domande a risposta aperta, esercizi) sulla teoria
  - Esercizio di programmazione in linguaggio C per un totale di 32 punti (30 e lode)
- Una **prova orale** facoltativa

## Sito web del corso

<http://zeus.ing.unibs.it/FI-PPING/>

- Possibile scaricare (**downloading**) le diapositive delle lezioni e delle esercitazioni
- I documenti sono in formato **pdf**
- Per aprirli occorre avere il programma **Acrobat Reader** (a sua volta scaricabile dal sito web <http://www.adobe.com/it/products/acrobat/readstep2.html>)

## Cos'è l'informatica?

- “l'informatica è lo *studio sistematico degli algoritmi che descrivono e trasformano l'informazione: la loro teoria, analisi, progetto, efficienza, realizzazione applicazione*” [ACM – Association for Computing Machinery]
- L'informatica è dunque una **scienza**: l'elaborazione dell'informazione avviene in modo sistematico e rigoroso
- L'elaborazione può essere automatizzata

## Algoritmi e linguaggi di programmazione

- **Algoritmo**: una sequenza di operazioni comprensibili ed eseguibili da un esecutore
- Un esempio di **esecutore**: il calcolatore
- Necessità di linguaggi per la descrizione di algoritmi: **linguaggi di programmazione**
- Esempi di linguaggi di programmazione: *Fortran, Cobol, Basic, C, Pascal, Ada, C++, Java, Prolog, Lisp*

## Architettura (struttura) dei sistemi informatici

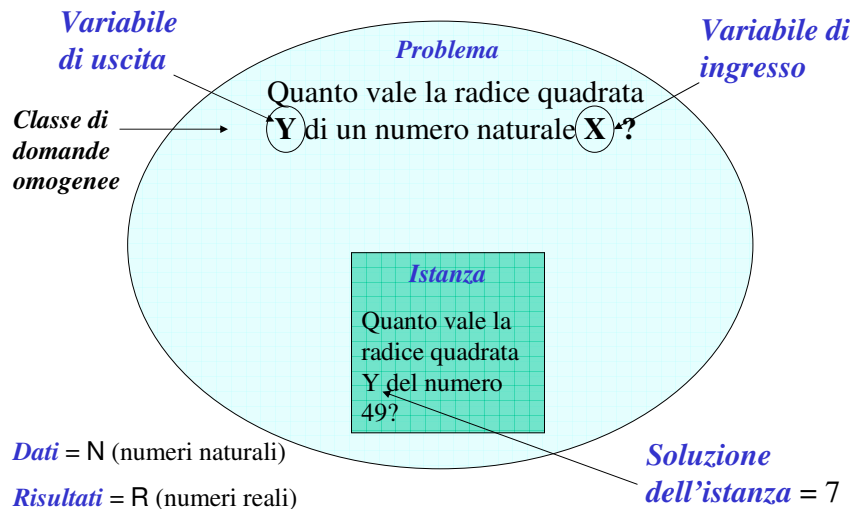
- **Hardware**:
  - Cosa compone un calcolatore elettronico?
  - Le reti di calcolatori
- **Software** (Programmi eseguibili da parte di un calcolatore)
  - *Software di base*: sistema operativo
  - *Software di “sistema”*: compilatori, interpreti, ...
  - *Software applicativi*: gestione banca, gestione biblioteca, CAD, Word, Excel, Explorer

## Il concetto intuitivo di calcolatore

## I problemi e la loro risoluzione...

- **Problema**: *classe di domande omogenee* alle quali è possibile dare risposta mediante una *procedura uniforme*
- **Istanza** del problema: ogni *specifica domanda* della classe
- **Variabili di ingresso**: termini variabili che caratterizzano la *formulazione* di un problema
- **Variabili di uscita**: termini variabili che caratterizzano le *soluzioni attese* di un problema
- **Dati**: *valori* che possono assumere le *variabili d'ingresso*
- **Risultati**: *valori* che possono assumere le *variabili d'uscita*
- **Soluzione** di un'istanza di un problema: risposta alla specifica domanda che l'istanza rappresenta

## Esempio



## Elaborazione di informazione

- Una certa classe di problemi richiede l'**elaborazione di informazione**
- Problema di elaborazione dell'informazione: insieme di **dati** di partenza e **risultato** ricercato
- Una **procedura di risoluzione** di un problema genera un risultato sulla base dei dati di partenza



## Elaborazione di informazione (2)

Elaborazione delle informazioni

➡ risolvere in modo efficace problemi



**Esempio:** problema di far entrare un pianoforte in una stanza che ha solo una porta...

**Strategia A (intervento diretto):** provare a far entrare il pianoforte in orizzontale... se non passa provare a girare il pianoforte, inclinarlo ....

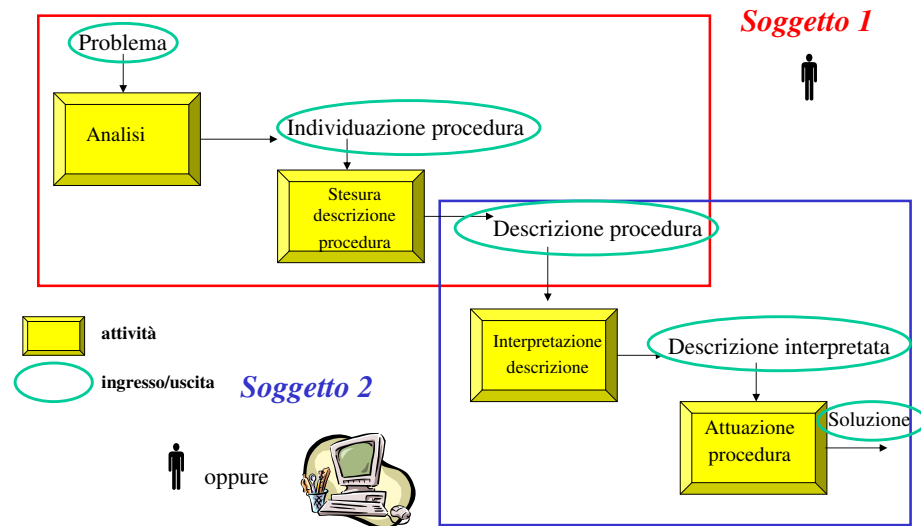
**Strategia B (intervento guidato da informazione):** misurare le dimensioni di porta e pianoforte (orizzontale, verticale, altezza...); quindi confrontare i valori numerici delle misure ed elaborare una soluzione: "far passare il pianoforte in verticale" oppure "allargare la porta"

➡ "confronto tra misure" è un esempio di elaborazione delle informazioni

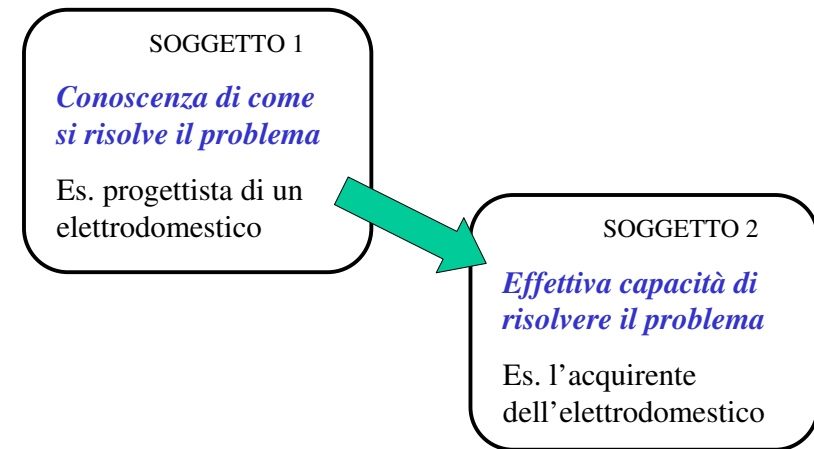
## Risoluzione di un problema

- Le fasi del procedimento di risoluzione di un problema:
  - Analisi del problema e individuazione di una procedura di risoluzione
  - Descrizione della procedura di risoluzione
  - Interpretazione della procedura di risoluzione
  - Attuazione della procedura di risoluzione

## Processo di risoluzione



## Esempio: montaggio di un elettrodomestico



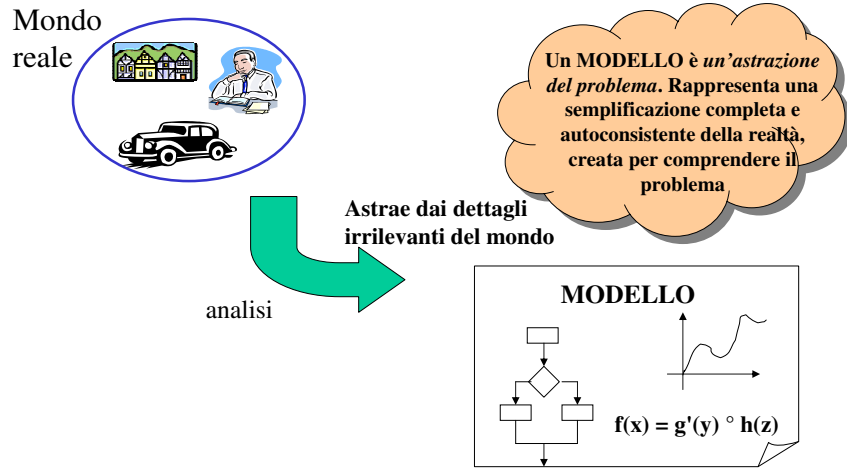
## Analisi del problema

- L'analisi di un problema consiste in:
  - *Comprensione del problema*: eliminando ogni ambiguità nella sua formulazione, focalizzando gli obiettivi, evidenziando i dati impliciti ed espliciti del problema
  - *Modellazione del problema*: creando un modello
  - *Ricerca di una procedura di soluzione*

## Modellazione del problema

- Cos'è un *modello*?
- Dato un certo problema, un modello del problema è una *rappresentazione semplificata* del problema stesso che evidenzia:
  - Gli **elementi** del problema
  - Le loro **proprietà** e le **relazioni** fra di essi
- Un modello è solo un *ausilio* alla risoluzione del problema, non è sufficiente da solo a risolvere il problema
- E' importante che la *procedura di risoluzione* trovata sul modello possa essere *interpretata* correttamente in modo da essere trasferita sulla realtà

## Relazione tra realtà e modello



## Un esempio di problema e di procedura di risoluzione

- **Problema:** come si fa una telefonata?
- **Procedura di risoluzione:**
  1. Solleva il ricevitore
  2. Componi il numero
  3. Attendi il segnale di libero
  4. *Se* arriva il segnale di libero **allora** attendi che venga qualcuno a rispondere **altrimenti** riaggancia
  5. *Se* dopo un po' non arriva nessuno a rispondere **allora** riaggancia **altrimenti** fai la tua conversazione

## Procedure di risoluzione e algoritmi...

- La procedura di risoluzione è quindi espressa come *sequenza di operazioni* la cui esecuzione porta alla soluzione del problema → **ALGORITMO RISOLUTIVO**



## Procedura di risoluzione di un problema

- Risolvere un problema = risolvere un'opportuna successione di problemi più semplici → **SCOMPOSIZIONE IN SOTTO-PROBLEMI**
- I sotto-problemi potrebbero dover essere a loro volta scomposti in *sotto-sotto-problemi*, e così via
- La scomposizione deve giungere fino ai **problemi elementari (o primitivi)**
- Ad ogni problema elementare corrisponde una **istruzione elementare**, che rappresenta la maniera di descrivere il problema elementare in modo che l'esecutore sia in grado di interpretarlo correttamente

## Problemi, istruzioni e azioni elementari

- Ogni istruzione elementare è associata ad una **azione elementare**: azione che può essere direttamente eseguita da un esecutore
- Esempio: nel problema della telefonata “componi il numero” può essere considerata un’azione elementare, in quanto solitamente non occorre spiegare di più (tutto dipende però dal soggetto esecutore...)

## Esempio di scomposizione di un problema

- Gestione di una biblioteca in cui:
  - I libri sono disposti sugli **scaffali**
  - La **posizione** di ogni libro è data da: **numero dello scaffale e posizione nello scaffale**
  - Esiste uno **\_schedario** ordinato in cui ogni scheda riporta le seguenti informazioni:
    - **Cognome e nome dell’autore**
    - **Titolo del libro**
    - **Data di pubblicazione**
    - **Numero dello scaffale in cui si trova**
    - **Posizione nello scaffale**

## Problema: richiesta di un libro

- Come procedo?
  1. Decido quale libro richiedere
  2. Prelevo il libro
- Il secondo passo va dettagliato, ovvero va scomposto in sotto-problemi (**procedura per raffinamenti successivi o top-down**)

## Il problema diventa...

1. Decido quale libro richiedere
2. **Cerco la scheda del libro nello schedario**
3. Mi segno numero dello scaffale e posizione nello scaffale
4. Cerco lo scaffale
5. Cerco il libro nella sua posizione all’interno dello scaffale
6. Prelevo il libro

Non è un problema elementare!

## Cercare la scheda...

- Scompongo in sotto-sotto-problemi:
  1. Prendo la prima scheda
  2. Il titolo, l'autore e la data corrispondono a quelli del libro che sto cercando? *Se sì allora* ho individuato la scheda, altrimenti passo alla scheda successiva e ripeto il controllo
  3. Se le schede sono esaurite, allora il libro non esiste
- Esistono metodi più efficienti per risolvere lo stesso problema... come fareste voi?

## Un metodo più efficiente...

1. Esamino la scheda centrale dello schedario
2. Se la scheda corrisponde al libro cercato allora termino la ricerca
3. Altrimenti cerco (con lo stesso metodo) nella metà inferiore o superiore dello schedario a seconda che il libro cercato preceda o segua il libro indicato sulla scheda

In realtà il passo 2 deve “accorgersi” anche se il libro non esiste, diventa:

**“se la scheda corrisponde al libro cercato oppure se la parte di schedario da esaminare è vuota allora termino la ricerca”**

## Due punti di vista...

- Punto di vista del soggetto *descrittore*: svolge attività di **scomposizione progressiva del problema**, fino ai **problemi elementari** descritti da **istruzioni elementari**
- Punto di vista del soggetto *esecutore*: svolge **attività di attuazione** delle **azioni elementari** associate alle istruzioni elementari identificate dal soggetto descrittore
- Istruzioni → aspetto descrittivo
- Azioni → aspetto esecutivo

## Descrizione di procedure di risoluzione

- Il soggetto descrittore descrive un *algoritmo*

un algoritmo è una *sequenza di istruzioni* che, operando sui dati iniziali, consentono di ottenere i risultati che costituiscono la soluzione del problema

- Tali istruzioni sono determinate tramite la scomposizione del problema in sottoproblemi
- La risoluzione di un sottoproblema è detta *passo* (*step*) dell'algoritmo



# Computazione

- **Computazione**: esecuzione di un algoritmo in corrispondenza di certi dati
- **Passo di computazione**: ogni singolo passo elementare che l'esecutore compie durante l'esecuzione di un algoritmo
- **Sequenza di computazione**: sequenza di passi elementari che l'esecutore compie in corrispondenza di certi dati iniziali durante l'esecuzione di un algoritmo

**Algoritmo = concetto statico**



**Computazione = concetto dinamico**



# Proprietà di un algoritmo

- **Finitezza**: un algoritmo deve essere costituito da un numero finito di istruzioni
- **Definitezza**: le istruzioni di cui un algoritmo è costituito devono appartenere a un insieme finito e prefissato di tipi elementari
- **Univocità**: ogni istruzione deve essere univocamente interpretabile ed eseguibile
- **Effettività**: deve esistere un esecutore in grado di eseguire ogni istruzione dell'algoritmo in un tempo finito

# Altre proprietà

- **Determinismo**: per qualunque dato di ingresso, a ogni passo della computazione, esiste al più un passo successivo. Ovvero: assegnato un dato di ingresso, esiste una e una sola computazione possibile dell'algoritmo
- **Correttezza**: l'algoritmo perviene alla soluzione del compito cui è preposto
- **Efficienza**: l'algoritmo perviene alla soluzione del compito impiegando il numero minimo di risorse fisiche
  - Risorse fisiche: tempo, memoria, ....
- **Terminazione**: l'esecuzione di un algoritmo deve terminare in un numero finito di passi

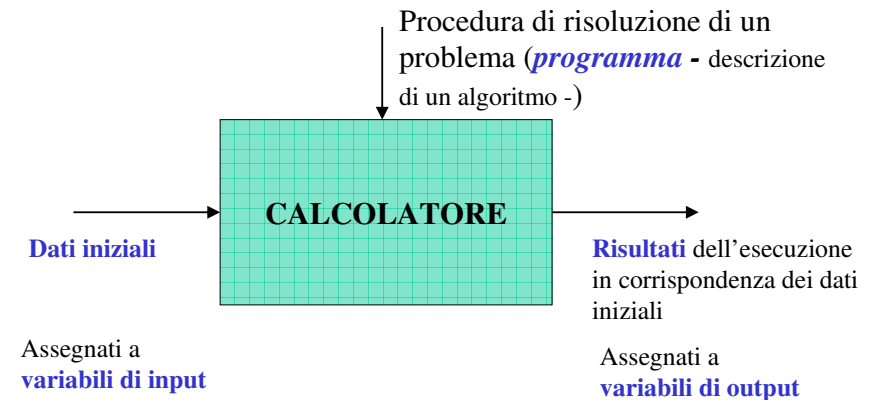
# Soggetto esecutore

- Il **soggetto esecutore (soggetto 2)** deve essere in grado di **interpretare** la descrizione della procedura di risoluzione
- Deve inoltre essere in grado di **eseguire** le azioni presenti nella descrizione interpretata
- Il **calcolatore** è un **esecutore di procedure di risoluzione** identificate e descritte da esseri umani (in genere un team di progettisti, programmatori e utenti)
- Quando/perché usare un calcolatore come esecutore?

## Algoritmi, programmi e calcolatori

- Un algoritmo deve essere comprensibile per il suo esecutore
- **Programma**: *descrizione formale di un algoritmo attraverso un linguaggio di programmazione*
- Un programma è una **sequenza di istruzioni** scritte in un opportuno linguaggio comprensibile dal calcolatore
- Il **calcolatore** è un **esecutore di programmi**
- Il **compito di un esperto informatico** consiste nel produrre algoritmi e codificarli in programmi

## Il calcolatore come esecutore (una prima definizione)



## Il calcolatore come esecutore (una definizione rivisitata)

- Un calcolatore è un sistema che, ricevendo in ingresso la descrizione, in un opportuno linguaggio, di un algoritmo risolvete  $A[In, Out]$  (cioè un programma) per un certo problema  $P[In, Out]$  e un dato In, produce un risultato Out, ovvero la soluzione Out dell'istanza  $P[In, Out]$
- Un calcolatore è un **esecutore universale di programmi**

## Esempio

