

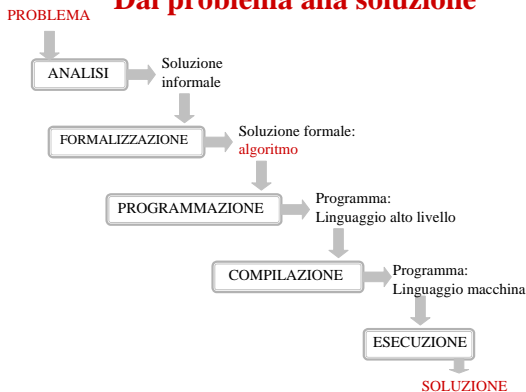
## Laboratorio di Informatica

### Introduzione alla programmazione

## Calcolatore

- **Informatica**: Scienza della rappresentazione e dell'elaborazione automatica dell'informazione
- **Calcolatore**: Sistema elettronico **programmabile** al fine di svolgere diverse funzioni
- **Programmabile**: in grado di eseguire un processo computazionale (**algoritmo**) in base a regole specificate attraverso un programma

## Dal problema alla soluzione



## Analisi

- L'analisi del problema è il primo passo; deve fornire
  - un *nome* e una *breve descrizione* di che cosa si vuol fare;
  - un elenco di *requisiti*: richieste a cui deve soddisfare il programma

## Analisi: Specifica funzionale

Una specifica funzionale indica

- quali sono i *dati iniziali*, cioè quelli da elaborare
  - detti anche *ingressi*
- che *risultato* si vuole, *in funzione degli ingressi*
  - detto anche *uscita*

## Analisi: l'ipotenusa

- Date le misure  $x$  e  $y$  di due cateti di un triangolo rettangolo, calcolarne l'ipotenusa  $i$ :
  - Input:  $x, y$  reali positivi
  - Output:  $i$

## Algoritmo

(da AL-KHOWARIZMI, Baghdad 825 dc):

- Descrizione non ambigua di un numero finito di operazioni, la cui esecuzione termina in un tempo finito
- In senso lato, una procedura che eseguita passo a passo risolve, entro un determinato periodo di tempo, un problema.

NB: il concetto di algoritmo prescinde dall'esistenza del calcolatore

## Algoritmo per calcolare l'ipotenusa

- Date le misure  $x$  e  $y$  di due cateti di un triangolo rettangolo:
  - Controlla che  $x$  e  $y$  siano reali positivi
  - Calcola  $x^2$
  - Calcola  $y^2$
  - Calcola  $z = x^2 + y^2$
  - L'ipotenusa è data dalla radice quadrata di  $z$ :  
ipotenusa  $i = \sqrt{z}$

## Chi è l'esecutore dell'algoritmo?

- Se vogliamo **automatizzare** l'esecuzione degli algoritmi, dobbiamo rappresentarli e comunicarli al calcolatore in un linguaggio a lui comprensibile e non ambiguo:

Linguaggi di programmazione

Algoritmo  Programma

- **Programma**: sequenza di istruzioni scritte in un linguaggio di programmazione comprensibile e non ambiguo per il calcolatore

## Linguaggi di programmazione

Un linguaggio di programmazione è costituito da:

- un **vocabolario**
- un insieme di **regole sintattiche** che specificano come comporre istruzioni ben formate
- una **semantica** che associa un "significato" alle istruzioni ben formate, cioè l'azione denotata da ciascuna istruzione.

## Programma

un insieme finito, ordinato di passi  
elementari  
eseguibili  
non ambigui  
che determinano un procedimento atto a risolvere in  
un tempo finito  
un problema o una classe di problemi  
fornendo la soluzione corretta (**correttezza**)  
per ogni istanza iniziale (**generalità**)

Possibilmente **efficiente!!!**

## Efficienza

- Un algoritmo è efficiente se raggiunge la soluzione del problema nel minor tempo possibile ed utilizzando il minor numero di risorse

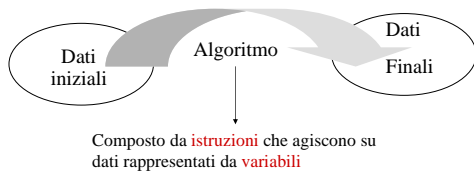
## Diversi livelli di espressività

- **Linguaggio Macchina:**
  - 001010001110000
  - totalmente illeggibile per l'uomo
  - Perfettamente non ambiguo per la macchina
- **Linguaggio Assembly**
  - LOAD x
  - ADD y
  - STORE z
  - simbolico del linguaggio macchina
- **Linguaggio ad alto livello, (es MATLAB)**
  - $\text{Area\_rettangolo} = \text{base} * \text{altezza};$
  - Le istruzioni corrispondono ad azioni più articolate
  - Richiedono uno sforzo di codifica inferiore da parte del programmatore

## Linguaggi ad alto livello

- Il programmatore scrive un programma in un linguaggio ad alto livello, più espressivo, facilmente manipolabile e comprensibile
- **Traduttori:**
  - **Compilatori:** traducono programmi scritti in uno specifico linguaggio ad alto livello in linguaggio macchina. Generano un file eseguibile  
Esempio: C
  - **Interpreti:** non traducono l'intero programma, ma una istruzione alla volta, immediatamente prima della sua esecuzione. Non generano nessun file eseguibile.
- Esempio: Matlab, Java

## Algoritmo



## Dati e istruzioni

- Tipi di dati
  - Dati elementari
    - Numeri interi
    - Numeri reali Caratteri alfanumerici
    - Dati logici o booleani
  - Dati strutturati
    - Vettori, matrici, ...
- Istruzioni
  - Operazioni di input/output
  - Operazioni aritmetico-logiche
  - Strutture di controllo

## Variabili

- Le **variabili** di programma sono un'astrazione della nozione di area di memoria contenente dei dati
- Una variabile è completamente definita da:  
NOME TIPO VALORE

## Rappresentazione grafica delle variabili

anni: intero  
19

- **Nome:** 'anni' lo userò per far riferimento alla variabile
- **Tipo:** 'intero' definisce la quantità di spazio da allocare in memoria, oltre a determinare l'insieme di operazioni applicabili alla variabile stessa
- **Valore:** '19' è il valore corrente o stato della variabile e contenuto della cella di memoria

## Variabili: nome

- Una lettera seguita da un qualsiasi numero, lettera o underscores
  - Numero massimo di caratteri: 31
  - Esempio giusto: iscritti\_2007\_2008
  - Esempio sbagliato: iscritti 2007-2008
- Consigli:
  - scegliere nomi significativi
  - Dividere parole diverse con underscore
- Matlab è "case sensitive": distingue lettere minuscole da lettere maiuscole
  - anni\_luca
  - Anni\_Luca

## Variabili: tipo

- Tipo dei dati : insieme di elementi *rappresentabili finitamente*, dotato di *operazioni* che si suppongono calcolabili dall'esecutore dell'algoritmo
- *ES.: il tipo dei numeri reali (double)*
  - è l'insieme dei numeri reali rappresentabili con 128 bit
  - dotato delle seguenti operazioni:
    - +, -, \*, /, ^

## Variabili: tipo

- In Matlab tutti i numeri sono memorizzati come 'double' secondo la codifica IEEE standard:
  - MANTISSA: 16 decimali;
  - CARATTERISTICA: +- 308
- CASTING: quando necessario è possibile fare una 'forzatura' ad un tipo specifico:
  - int8, int16, int32, uint8, uint16, uint32,
  - single (64 bit), double (128 bit)

## Variabili: valore


- è il valore corrente o stato della variabile
- è contenuto della cella di memoria
- Nell'esempio è '19'

19  
anni: intero

## Creare variabili

- Alcuni linguaggi richiedono di dichiarare le variabili che si useranno nel programma
- **MATLAB non richiede dichiarazioni, e le variabili vengono create dinamicamente mediante un assegnamento**

## Assegnamento

- Serve ad assegnare un valore ad una variabile
- Esempio: anni ← 19
- Se la variabile non esiste la creo:   
anni: intero
- Se la variabile esiste già, sostituisco il valore



## Assegnamento

- In MATLAB: `anni = 19`

Attenzione: l'operatore "="  
significa assegnare,  
non è l'uguaglianza matematica!!!

## Assegnamento

- L'assegnamento prevede che
  - a **sx** dell'uguale ci sia *una ed una sola variabile*
  - a **dx** ci sia un'espressione valutabile:
    - Tutte le variabili usate devono essere state precedentemente create ed inizializzate
    - Sintatticamente corretto
    - L'ordine di valutazione è come quello usato in matematica

## Espressioni

- **COSTANTE:** Espressione semplicissima:

- `anni_maria = 19`

19  
`anni_maria`

- **Operazioni con costanti:**

- `anni_maria = 19 + 1`
- Valuto l'espressione: "19+1" (20)
- assegno 20 a "anni\_maria"

~~19~~ → 20  
`anni_maria`                      `anni_maria`

## Espressioni

- **Con una variabile**

- `anni_maria = 19`
- `anni_luca = anni_maria + 3`
- **VALUTO:**
  - sostituisco al nome della var 'anni\_maria' il suo valore corrente
  - Valuto: 19 + 3 (22)
  - Assegno 22 a "anni\_luca"

19                      22  
`anni_maria`                      `anni_luca`

## Espressioni

- **Con più variabili**

- Qual è l'età di Luca sapendo che è più giovane di Roberto e più vecchio di Chiara e che ha la stessa differenza di anni dai due?
  - `anni_chiara = 15`
  - `anni_roberto = 25`
  - `anni_luca = (anni_chiara + anni_roberto) / 2`
  - **VALUTO:**
    - sostituisco al nome delle var 'anni\_chiara' e 'anni\_roberto' il valore corrente
    - Valuto: (15 + 25) / 2
    - Assegno a "anni\_luca"

15                      25                      →                      20  
`anni_chiara`                      `anni_roberto`                      `anni_luca`

## Espressioni

- **Con più variabili, (un altro es.):**

- `anno_corrente = 2007`
- `anno_nascita_maria = 1976`
- `anni_maria = anno_corrente - anno_nascita_maria`

## Espressioni

2007

anno\_corrente

- **Riassegnamento:**
  - $\text{anno\_corrente} = \text{anno\_corrente} + 1$
- VALUTO l'espressione "anno\_corrente + 1":
  - Sostituisco al nome "anno\_corrente" il suo valore corrente (2007)
  - Valuto l'espressione numerica "2007+1" (2008)
  - Assegno 2008 a "anno\_corrente"

2008

anno\_corrente

## Algoritmo per calcolare l'ipotenusa

- Date le misure  $x$  e  $y$  di due cateti di un triangolo rettangolo:
  - Calcola  $x^2$
  - Calcola  $y^2$
  - Calcola  $z = x^2 + y^2$
  - L'ipotenusa è data dalla radice quadrata di  $z$ :  
ipotenusa  $i = \sqrt{z}$

## Esercizio: Conversione Euro / Lire

- Scrivere un programma che dato un valore in Euro, lo converte in lire italiane

## Esercizio: Conversioni di temperature

- Scrivere un programma che dato un valore in gradi centigradi, lo converte in gradi Fahrenheit
- Scrivere un programma che dato un valore in gradi centigradi, lo converte in Kelvin

## Esercizio: Scambio di variabili

- Scrivere un programma che date due variabili  $a$  e  $b$  inizializzate a scelta, ne scambia i valori e li stampi.