## Sommario

- Definizione informale di algoritmo
- Definizione informale di linguaggio
- Definizione di interprete e traduttore

## Algoritmo: etimologia

- Parola entrata in uso negli anni '50 per sostituire la parola *algorismo* che designava il processo di calcolare con i numeri arabi

- Etimologia
  - Creduta nel medioevo (*ma tuttora sospettata da molti studenti*): dal greco *algiros* (doloroso) + *arithmos* (numero)
  - Quella vera: dal nome dell'autore di un testo di algebra (825 d.C.) Abu Ja'far Mohammed ibn Mûsâ al-Khowârizmî

## Algoritmi: concetti informali

- Un algoritmo è una procedura computazionale **ben definita** che trasforma un insieme di dati di **ingresso** in un insieme di dati di **uscita**, al fine di risolvere un problema.

- Un algoritmo viene descritto da una **sequenza finita** di passi computazionali che hanno l'effetto di trasformare l'ingresso nell'uscita desiderata.

## Algoritmo: concetti informali (cont.)

- Esempi di "algoritmo" nella vita quotidiana:
  - Istruzioni di montaggio di un mobile
  - Calcolo del massimo comun divisore fra più numeri naturali
  - Prelevamento di denaro a un terminale Bancomat: inserire tessera magnetica, digitare codice, specificare importo da prelevare, ritirare la carta, prelevare il contante
- L'algoritmo deve essere *comprensibile* al suo esecutore
  - Se un libretto di istruzioni fosse disponibile solo in inglese non sarebbe compreso da chi conosce solo l'italiano.

## Algoritmi e programmi

- Nel campo dell'informatica, possiamo definire i calcolatori elettronici come esecutori di algoritmi.
- Gli algoritmi vengono descritti tramite **programmi**, cioè sequenze di istruzioni in un opportuno linguaggio comprensibile al calcolatore.
- Compito dell'esperto informatico:
  - **Produrre algoritmi**
  - **Codificarli in programmi**

## Correttezza ed efficienza degli algoritmi

- Un'**istanza** di un problema è un particolare valore ammissibile assunto dai dati in ingresso
- Un algoritmo è **corretto** se perviene alla soluzione del compito senza difettare di alcun passo fondamentale
  - **termina** su **ogni istanza** del problema (la sequenza di passi computazionali è **finita**)
  - Produce l'uscita desiderata su **ogni istanza** del problema
- Un algoritmo è **efficiente** se perviene alla soluzione nel modo più veloce e/o usando la minor quantità di risorse

## Esempio: utilizzo di un lettore portatile di CD musicali

- Si vuole ascoltare il brano n. 13
  - Se siamo a casa, colleghiamo l'alimentatore ad una presa elettrica
  - Se non è disponibile la presa, verifica batterie. Se manca qualche batteria o è scarica, inserimento o sostituzione batterie
  - Accensione lettore CD: il display indica 'No disk'
  - Inserimento CD musicale
  - Premere ripetutamente forward finché nel display non compare 13
  - Indossare le cuffie e premere play

## Esempio: gestione di una biblioteca

- Una piccola biblioteca che contenga scaffali in cui sono disposti i libri
  - Posizione dei libri invariabile negli scaffali
- Ad ogni libro è associata una scheda
  - Cognome e nome autori (nell'ordine in cui compaiono nel libro)
  - Titolo
  - Data di pubblicazione
  - Numero dello scaffale
  - Numero d'ordine della posizione nello scaffale

## Esempio: gestione di una biblioteca (cont.)

- Le schede sono conservate in ordine alfabetico rispetto al cognome del primo autore
- Semplice algoritmo per accedere ad un libro
  - Cercare la scheda nello schedario
  - Segnare su un foglietto scaffale e posizione del libro
  - Cercare lo scaffale indicato
  - Cercare il libro sullo scaffale. Se è presente, si compila il modulo di prestito con data e nome del richiedente

## Esempio: gestione di una biblioteca (cont.)

- Tecnica di ricerca nello schedario
  - Si esamina la prima scheda
  - Se il nome e il titolo coincidono con quelli cercati, la ricerca termina con successo, altrimenti si passa alla scheda successiva
  - Si continua con tutte le schede finché tutte le schede sono esaurite: se non si trova il libro, si deve cercare altrove
- Con questa tecnica di ricerca, nel caso migliore si esamina una scheda (la prima); nel caso peggiore si devono esaminare tutte le schede!

## Algoritmi definizione di procedura

- La **procedura effettiva** è la sequenza di passi che deve essere seguita per risolvere il problema in base all'algoritmo
  - Tutti i problemi sono elementari
  - E' fissato l'ordine di esecuzione dei problemi
  - E' definito come ogni problema da i risultati e come questi vengono utilizzati

## Algoritmi: sottoproblemi

- Ogni problema viene quindi scomposto in problemi elementari
- si definisce la procedura di soluzione di ogni singolo problema
- i risultati di ogni sottoproblema sono utilizzati per trovare la soluzione del problema

## Algoritmi: sottoproblemi es.

- Andare da Piazza D'Armi di Cagliari a Via dei Mille a Sassari
- Analisi: uscire da Cagliari, prendere la 131, entrare a Sassari arrivare in Via dei Mille; Sottoproblemi:
  - Andare da Piazza D'Armi alla 131
  - Percorrere la 131 fino al miglior ingresso di Sassari
- Abbiamo spezzato il problema e risolto i problemi elementari, che a loro volta possono essere ulteriormente scomposti

## Algoritmi: sottoproblemi es.

- Calcolare la superficie di un triangolo rettangolo
  - S=b*h/2
- Calcolare la superficie di un rettangolo
  - S=b*h
- Calcolare la superficie di un cerchio
  - S=pigreco*$r^2$

## Algoritmi: sottoproblemi es.

- Calcolare la superficie di una figura geometrica non elementare
- Algoritmo:
  - calcolare superficie del semicerchio S1
    - calcolare il raggio in funzione della base minore del trapezio
  - calcolare superficie del trapezio S2
    - superficie del rettangolo S21 + superficie dei 2 triangoli S22a ed S22b
    - S2=S21+S22a+S22b
  - S=S1+S2

## Linguaggi per la programmazione di algoritmi

- Nell'idea informale di algoritmo esiste il concetto implicito di **esecuzione di "istruzioni"**
- Queste istruzioni devono essere espresse in modo **preciso** e **non ambiguo**
- Se esprimiamo le istruzioni in linguaggio "naturale" l'ambiguità viene sostanzialmente rimossa dall'intelligenza umana e dalla conoscenza di un contesto informativo comune
  - Le istruzioni degli algoritmi vengono eseguiti da macchine che non hanno *buon senso!*

## Linguaggi per la programmazione di algoritmi (cont.)

- Agli albori dell'informatica il linguaggio di programmazione coincideva con il linguaggio della macchina, cioè con l'insieme di comandi che la macchina era in grado di eseguire
- Nella seconda metà degli anni '50 il linguaggio di programmazione si "alzò di livello"
  - Più adatto a codificare algoritmi
  - Più vicino al linguaggio naturale

  La *traduzione* del programma in linguaggio macchina è affidata alla macchina stessa

## Linguaggi e calcolatori elettronici

- La progettazione degli attuali calcolatori elettronici prevede diversi livelli di *astrazione*
  - Circuiti logici
  - µprocessore
  - Sistema operativo
  - Linguaggi di programmazione di *alto livello*
- Si parla di progettazione *strutturata*
  - le istruzioni di *alto livello* che eseguono funzionalità complesse vengono tradotte in *istruzioni semplici* nei livelli inferiori.

## Organizzazione *strutturata* di un calcolatore

## Livelli di astrazione di un calcolatore

- Livello 0: *porte logiche*
  - Eseguono istruzioni logiche (AND, OR, ecc.) su segnali binari (0, 1)
- Livello 1: *Microarchitettura*
  - Insieme di porte logiche che eseguono operazioni su *insiemi di bit*
- Livello 2: ISA (*Instruction Set Architecture)*
  - E' l'insieme di istruzioni che possono essere eseguite dal µprocessore
- Livello 3: Sistema Operativo
  - Gestisce le risorse del calcolatore (µprocessore, memoria, periferiche)
- Livelli 4, 5: Linguaggi di programmazione di algoritmi

## Livelli di astrazione di un calcolatore

- Ciascun livello mette a disposizione un proprio *linguaggio* che viene *tradotto* o *interpretato* a seconda dei casi
- A livelli inferiori (L0, L1 e L2) le istruzioni sono *codificate* da opportuni segnali elettrici
  - Passaggio fra livelli: *traduzione (a volte interpretazione)*
- A **livello** più **alto** si collocano **linguaggi** più **vicini al linguaggio *naturale*** usato per la descrizione di **algoritmi**.
  - Passaggio fra livelli: *interpretazione (a volte traduzione)*

## Linguaggi di *alto livello*

- Primo linguaggio di programmazione di alto livello: FORTRAN (FORmula TRANslator), per calcolo numerico

- Di poco posteriore il COBOL (COmmon Business Oriented Language), orientato alle applicazioni gestionali e all'elaborazione dati
- N.B. questi linguaggi, anche se datati, sono ancora molto diffusi

## Linguaggi di *alto livello*

- Altri linguaggi basati su uno **studio dei principi della programmazione**:
  - Capostipite: ALGOL 602
  - Pascal: molto diffuso nella didattica dell'informatica
  - C, il linguaggio che ha attualmente maggior successo
  - ADA, usato dal Dipartimento della Difesa (DoD) degli USA
  - Linguaggi orientati agli oggetti: corrispondenza fra gli oggetti che caratterizzano una certa applicazione e la loro codifica. Ad es.: C++, Java, SmallTalk

## Ambiente di programmazione

- Strumenti per facilitare la scrittura dei programmi e verificarne la correttezza.
  - **Editor**: serve per scrivere il *programma sorgente*, cioè il **testo** che contiene le istruzioni nel linguaggio prescelto
  - **Compilatore**: Trasforma un programma sorgente in *programma oggetto*, cioè in un programma in linguaggio macchina. Se vi sono errori di correttezza, viene avvisato il programmatore e il programma oggetto non viene generato.
  - **Interprete**: per alcuni linguaggi non si usa il compilatore, ma un interprete che esegue direttamente il codice sorgente come ad esempio **MATLAB**

## Ambiente di programmazione (cont.)

  - **Linker**: Collega insieme vari programmi oggetto che fanno parte di un unico programma suddiviso in *moduli* coordinati fra loro, generando il *programma eseguibile*.
  - **Debugger**: Consente di eseguire il programma passo passo verificando l'esecuzione delle istruzioni del programma sorgente e individuando eventuali errori
- Gli ambienti di programmazione forniscono in genere anche un insieme di *funzioni di libreria*, cioè di algoritmi comuni a molti programmi (es. ordinamento di un vettore di numeri)