# Appunti di Algoritmi e Strutture Dati

Algoritmi e Strutture Dati (prof. ???) - CdL Informatica Unimib - 23/24

Federico Zotti

06 Mar 2024

# Indice



  

# 1 Introduzione

Un'**algoritmo** è una sequenza di istruzioni **elementari** (devono essere comprese e eseguite dall'esecutore) che permettono di risolvere un problema computazionale (ovvero per ogni possibile input produce l'output corretto).

Per definire un **problema** è necessario specificare:

- Il tipo del parametro in input
- Il tipo del risultato in output
- Il legame tra input e output

Un'**istanza** di un problema si ottiene specificando uno dei possibili valori in input specifico per il problema.

## 1.1 Definizione dell'ordinamento di un vettore

**Sort:**

- Input: `Array Int` (`Dim` $n$) $\rightarrow A = \langle a_1, a_2, \dots, a_n \rangle$
- Output: `Array Int` (`Dim` $n$) $\rightarrow A' = \langle a'_1, a'_2, \dots, a'_n \rangle$

$A'$ è una permutazione di $A$, tale che $a'_1 \leq a'_{i+1} \quad \forall i \,.\, 1 \leq i \leq n-1$.

## 1.2 Scelta di un algoritmo

L'algoritmo migliore è quello che utilizza il minor numero di risorse.

Le risorse sono:

- Il tempo di esecuzione
- Lo spazio (memoria) utilizzato

### 1.2.1 Tempo di esecuzione

Per calcolare il tempo utilizziamo una funzione $T(n)$. $n$ rappresenta la quantità di dati in input.

  

- $T_p(n)$ rappresenta il caso peggiore
- $T_n(n)$ rappresenta il caso “medio” (non è la media dei due)
- $T_m(n)$ rappresenta il caso migliore

#### 1.2.1.1 Esempio

- Algoritmo 1: $T(n) = 100000 \cdot n$
- Algoritmo 2: $T(n) = 10 \cdot n^3$
- Algoritmo 3: $T(n) = 1 \cdot 2^n$

In questo caso il migliore dipende dal grado di $n$, dunque l’algoritmo 1 risulta quello più veloce. Per numeri di $n$ molto piccoli invece è meglio calcolare caso per caso il tempo.

  