Ordinamento

Insertion Sort
Bubble Sort

Il Problema dell'Ordinamento

- Elencare gli elementi di un insieme secondo una sequenza stabilita da una *relazione d'ordine*. Esempi:
  1. Ordinare una breve sequenza di numeri
  2. Mettere un elenco di nomi in ordine alfabetico
  3. Ordinare i record degli studenti Unisa secondo la data di nascita
- Nel caso 3, dobbiamo ordinare dei *record* in base ad una *chiave*
  - La chiave può essere un singolo campo o la combinazione di più campi

Algoritmi di Ordinamento
Proprietà

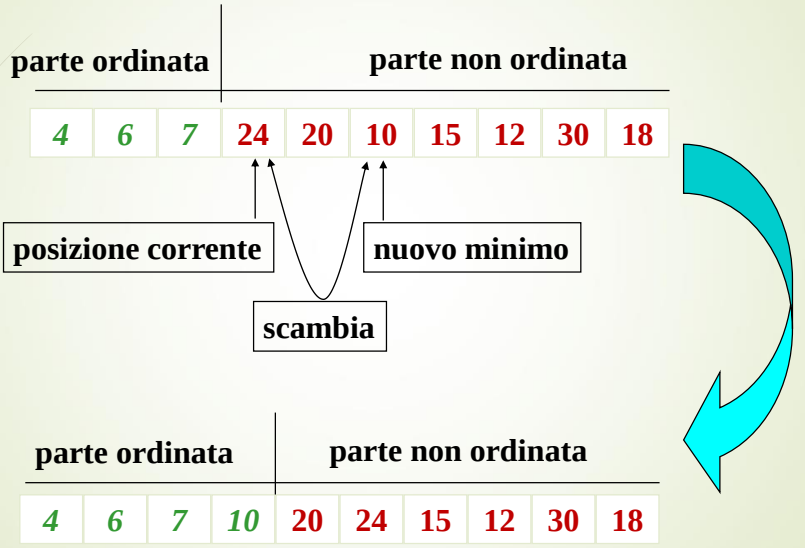
- **Stabile**: due elementi con la medesima chiave mantengono lo stesso ordine con cui si presentavano prima dell'ordinamento.
- **In loco**: in ogni dato istante al più è allocato un numero costante di variabili, oltre all'array da ordinare
- **Adattivo**: Il numero di operazioni effettuate dipende dall'input
- **Interno vs esterno**:
  - Interno: i dati sono contenuti nella memoria RAM.
  - Esterno: I dati sono residenti su disco o su nastro

Algoritmi Semplici e Avanzati

- Tutti gli algoritmi elencati in basso ordinano **per confronti**
- Algoritmi semplici. Numero di operazioni quadratico rispetto alla taglia dell'input: $O(n^2)$
  - *selection sort*
  - *insertion sort*
  - *bubble sort*
- Algoritmi avanzati. Più efficienti.
  - *Merge sort* (von Neumann, 1945)
    - Numero di operazioni rispetto alla taglia dell'input: O(n log n)
  - *Quicksort* (Hoare, 1961)
    - O(n log n) nel caso medio
    - quadratico nel caso peggiore

Hoare, C. A. R. (1961): Partition: Algorithm 63, Quicksort: Algorithm 64, and Find: Algorithm 65., Comm. ACM 4, pp. 321–322

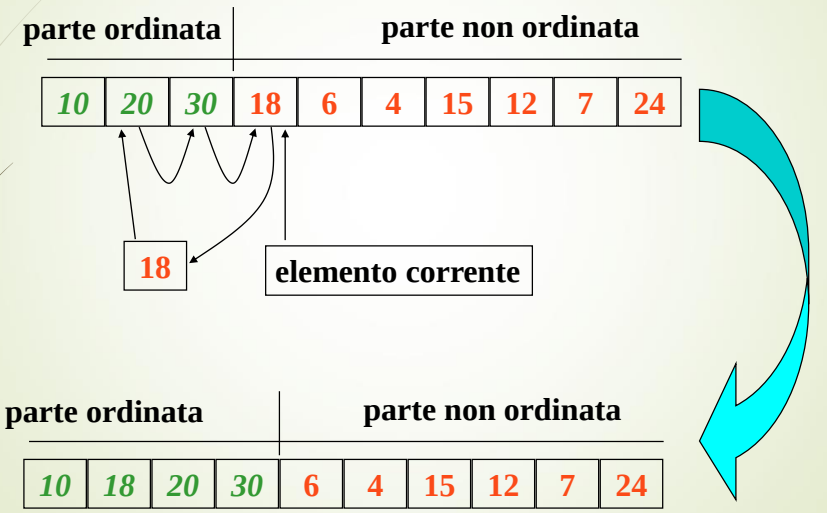
Selection Sort



Insertion Sort

- Visita totale: ad ogni passo gli elementi che precedono l'elemento corrente sono ordinati
  - si inserisce l'elemento corrente nella posizione che garantisce il mantenimento dell'ordinamento
  - gli elementi precedenti maggiori sono spostati in avanti
  - … il primo elemento è già ordinato …

Insertion Sort



Progettazione

Funzione insertion_sort

- for(i = 1; i < n; i++)
  1. inserisci l'elemento in posizione i nell'array ordinato a[0..i-1]

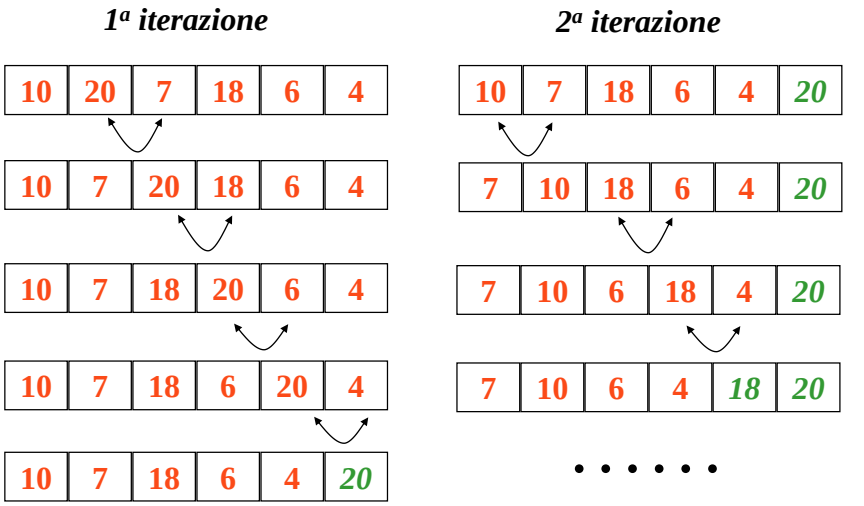
Raffinamento del programma principale: definiamo delle funzioni corrispondenti agli step individuati

- **Inserisci (a, val, n)**

Bubble Sort

- Algoritmo iterativo:
  - Si effettuano n-1 visite dell'array
    - Alla visita i-esima, si confrontano elementi adiacenti dal primo al (n-i)-esimo elemento
    - Elementi adiacenti che non risultano ordinati vengono scambiati
- NB: ad ogni passo l'elemento più grande viene portato nella sua posizione finale …
  - dopo il passo i-esimo, gli elementi tra le posizioni n-i ed n-1 risultano ordinati e nelle loro posizioni finali

Bubble Sort



Algoritmo di Bubble sort

for(i=1; i < n; i++)

scambia gli elementi adiacenti che non risultano ordinati tra le posizioni 0 e n-i

Bubble Sort

Versione adattiva

- Se in una visita dell'array non è stato effettuato alcuno scambio, allora l'array è ordinato
  - In tal caso si può interrompere

Algoritmo di Bubble sort

```
boolean ordinato = false;
for(i=1; i < n && ! ordinato; i++)
    ordinato = true;
    scambia gli elementi adiacenti che non risultano ordinati tra le
        posizioni 0 e n-i e poni ordinato a false se viene effettuato
        almeno uno scambio
```