UNIVERSITÀ
DEGLI STUDI
DI TRIESTE

MODULO 2: Bubblesort

Prof.ssa Giulia Cisotto

giulia.cisotto@units.it

Trieste, 15 maggio 2025

ALGORITMO

Input: un elenco di numeri da ordinare (come una fila di carte)

1. **Confronta** i primi due numeri dell'elenco:
   - Se il primo è maggiore del secondo, scambiali di posto.
   - Altrimenti, lasciali così.
2. Spostati alla coppia successiva (secondo e terzo numero) e fai lo stesso confronto e possibile scambio.
3. Continua così, confrontando ogni coppia adiacente, fino alla fine dell'elenco.
4. **Dopo il primo giro, il numero più grande si sarà spostato alla fine** (come una bolla d'aria che sale in superficie — da qui il nome *Bubble Sort*).
5. Ripeti lo stesso procedimento, ignorando l'ultimo elemento già in posizione.
6. Continua a ripetere i passaggi, ogni volta confrontando una coppia in meno, *finché non ci sono più scambi da fare.*
7. A quel punto, l'elenco è completamente ordinato in ordine crescente.

ESEMPIO

Array da ordinare

PSEUDOCODICE*

```
BUBBLE-SORT(A,n)                    //input: array A di lunghezza n
for i=1 to n-1                      //aggiungi commento
    flag_scambi = Falso             //flag che verifica se sono stati fatti scambi
    for j = 1 to n-i                //aggiungi commento
        if A[j] > A[j+1]            //aggiungi commento
            scambia A[j] con A[j+1]     //aggiungi commento
            flag_scambi = Vero          //aggiungi commento
    if flag_scambi = Falso              //aggiungi commento
        return
```

*ottimizzato rispetto al libro di testo

PRESTAZIONI: CALCOLO NUMERO ITERAZIONI

```
BUBBLE-SORT(A,n)
for i=1 to n-1
    flag_scambi = Falso
    for j = 1 to n-i
        if A[j] > A[j+1]
            scambia A[j] con A[j+1]
            flag_scambi = Vero
    if flag_scambi = Falso
        return
```

Al massimo *n-1* iterazioni

Al massimo *n-i-1* iterazioni

COSTI SIGNIFICATIVI
$c_1 \cdot (n-1)$
c_2
$c_3 \cdot (n-i-1)$
c_4
c_5
c_6
c_7

Calcolo costo complessivo

$$T(n) = c_1 \cdot (n-1) + c_2 + c_3 \cdot (n-i-1) + \ldots$$

$$\ldots + c_4 + c_5 + c_6 + c_7$$

Si nota che:

$$c_3 \cdot (n-i-1) = \sum_{i=1}^{n-1} \sum_{j=1}^{n-i} c_3$$

Tenendo solo i costi dominanti:

$$T(n) = O(n) + \sum_{i=1}^{n-1} \sum_{j=1}^{n-i} O(1) =$$

$$= O(n) + \sum_{i=1}^{n} i = O(n) + \frac{n(n+1)}{2} = O(n^2)$$

Nota. c1, c2, .. sono delle costanti che quantificano il costo di esecuzione della relativa istruzione. Per **esempio c3 rappresenta il costo dell'istruzione for** e include il controllo della condizione e l'incremento dell'indice i.

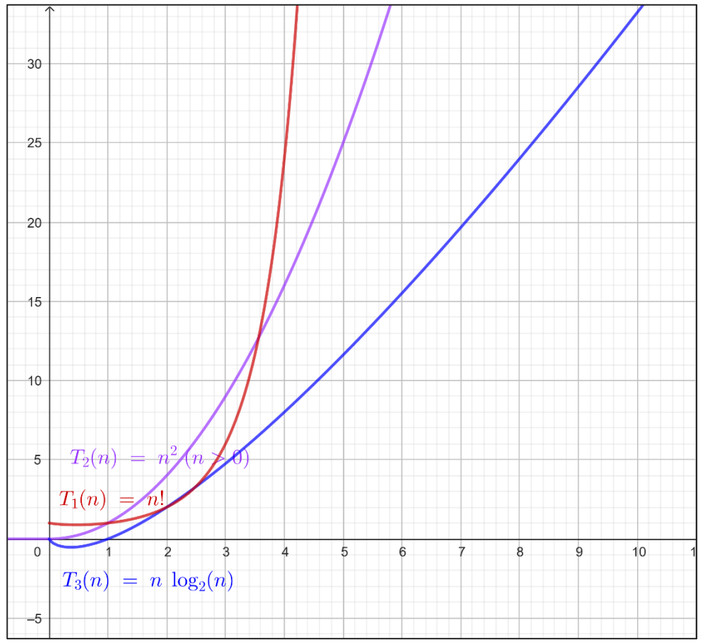
CONFRONTO CON ALTRI ALGORITMI DI ORDINAMENTO

Caso migliore (array già ordinato): O(n)

Caso medio/peggiore: $O(n^2)$

Algoritmo	T(n)
Bogosort	$n!$
Insertion sort	$n*n = n^2$
Mergesort	$n*\log_2 n$
Bubblesort	$O(n^2)$

bogosort insertionsort mergesort



Potete usare Geogebra online per creare lo stesso grafico e poi aggiungere l'andamento dell'algoritmo assegnato.

UNIVERSITÀ
DEGLI STUDI
DI TRIESTE

Prossimo laboratorio: 16 ***maggio, h.11:00, aula 5B***