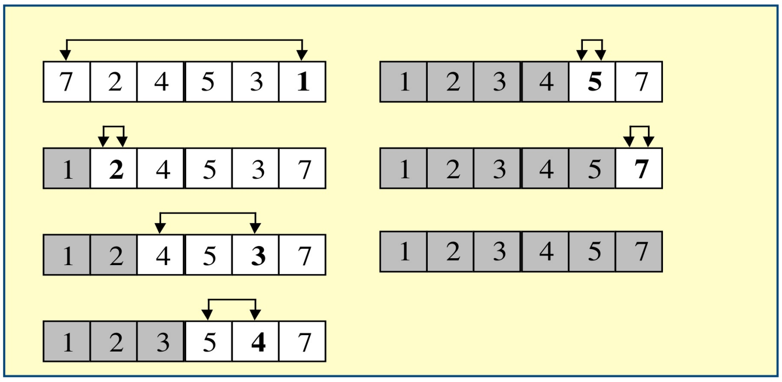
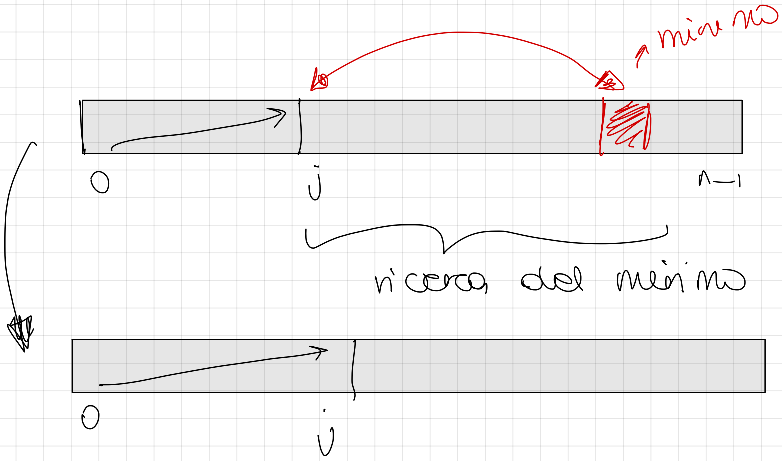
SELECTION SORT



passo j:

i primi j elementi in a[0...j-1] sono i j elementi più piccoli di a e sono già ordinati

→ scegliamo il più piccolo tra quelli rimanenti in a[j...n-1] e lo mettiamo nella posizione a[j] (facendo uno scambio)



al termine del passo j:

i primi j+1 elementi a[0...j] sono esattamente i primi j+1 elementi della sequenza finale ordinata.

↳ non si spostano più

SelectionSort (a)

// array a di n interi
word model

```
n = a.length
for (j = 0; j < n-1; j++) {                     → si ripete n-1 volte
    // ricerca del minimo in a[j ... n-1]
    min = j              // min: posizione del minimo corrente in a
    for (i = j+1; i < n; i++) {
            if (a[i] < a[min]) min = i;          si ripete n-j-1 volte
    }
    // scambio tra a[j] e a[min]
    temp = a[j];
    a[j] = a[min];
    a[min] = temp;
}
```

$(n-1) - (j+1) + 1$

$n - 1 - j \cancel{-1} \cancel{+1}$

Invariante (per la correttezza)

Prima dell'esecuzione della j-sima iterazione ($0 \leq j \leq n-2$) del for esterno, il segmento di array $a[0 \dots j-1]$ è ordinato e contiene i j elementi più piccoli di a.

ESERCIZIO

→ Inizializzazione

→ Conservazione

→ Conclusione

SOLUZIONE

→ Inizializzazione prima della prima iterazione, $j=0$

$a[0 \dots -1]$ è un array vuoto ✓

→ Conservazione ok, per ispezione diretta del codice

→ Conclusione alla fine, $j = n-1$

$a[0 \dots n-2]$ è ordinato e contiene gli $n-1$ elementi più piccoli dell'intero array.

in posizione $n-1$ c'è il massimo. ✓

Complessità

studiamo il numero di confronti

$T(n)$ è proporzionale al numero dei confronti

In tutti i casi,

$$C(n) = \underbrace{\sum_{j=0}^{n-2}}_{\text{iterazioni del for esterno}} \underbrace{\left(\sum_{i=j+1}^{n-1} 1 \right)}_{\text{iterazioni del for interno}}$$

1 confronto ∀ iterazione del for interno

$$C(n) = \sum_{j=0}^{n-2} \left(n-1 -(j+1) + 1 \right) = \sum_{j=0}^{n-2} \underbrace{(n-j-1)}_{}$$

\# confronti necessari e sufficienti per trovare il minimo in $a[j \dots n-1]$

$$= \underset{j=0}{(n-1)} + \underset{j=1}{(n-2)} + \underset{j=2}{(n-3)} + \dots + \underset{j=n-3}{2} + \underset{j=n-2}{1}$$

$$= \sum_{j=1}^{n-1} j = \frac{n(n-1)}{2} = \binom{n}{2}$$

↳ cresce come una funzione quadratica in n (dim. array)

⇒ esegue esattamente $\binom{n}{2}$ confronti

confronta tutte le coppie possibili di elementi dell'array

RIEPILOGO

# CONFRONTI	CASO OTTIMO	CASO MEDIO	CASO PESSIMO
Insertion Sort	$n-1$	$\frac{1}{2}\binom{n}{2}$	$\binom{n}{2}$
Selection Sort	$\binom{n}{2}$	$\binom{n}{2}$	$\binom{n}{2}$

COSTO IN TEMPO TASSO DI CRESCITA	CASO OTTIMO	CASO MEDIO	CASO PESSIMO
Insertion Sort	n	n^2	n^2
Selection Sort	n^2	n^2	n^2

COSTO in SPAZIO : costante

Insertion Sort e Selection Sort ordinano in loco (sul posto) al caso ottimo, medio e pessimo.

NOTAZIONE ASINTOTICA (Knuth, 1970)

- descrive il comportamento asintotico ~~delle~~ delle funzioni

  ↳ costo in tempo degli algoritmi, espresso ~~in~~ in funzione della dim. dei dati di input

$$f(n) \qquad n \geq 0$$

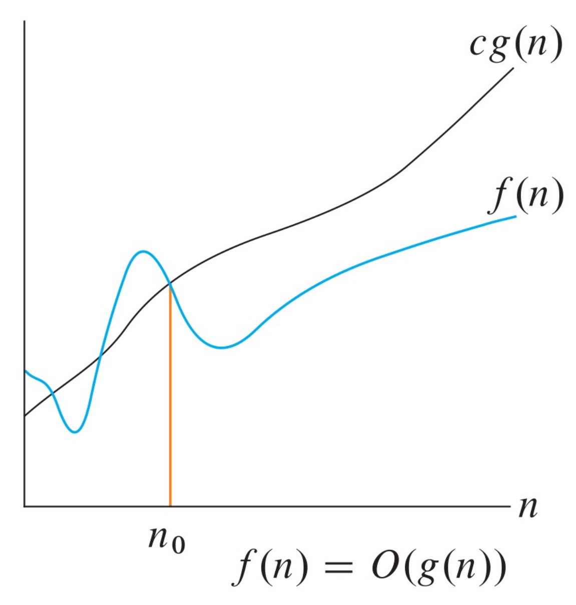
↳ descrive il tasso di crescita: come si comporta la funzione al ↗ di n

per confrontare funzioni

O	o-grande	→	$\leq$	a meno di fattori costanti e termini di ordine inferiore
Ω	omega-grande	→	$\geq$	
Θ	theta	→	$\approx$	
o	o-piccolo	→	$<$	
ω	omega-piccolo	→	$>$	

Notazione O

$$f(n) \in O(g(n))$$



$g(n)$ è un limite superiore asintotico per $f(n)$

↓

$f(n)$ CRESCE AL PIÙ COME $g(n)$ (a meno di un fattore costante)

$\leq$

$$O(g(n)) = \{ f(n) \mid \exists \text{ costanti positive } n_0, c \text{ t.c. } 0 \leq f(n) \leq c \cdot g(n) \ \forall n \geq n_0 \}$$

ESEMPI

$$f(n) = 7n^3 + 100n^2 - 20n + 6 \in O(n^3)$$

$f(n)$ è anche $O(n^4)$, $O(n^{10})$

Esempi di funzioni in $O(n^2)$:

n^2

$10n^2$

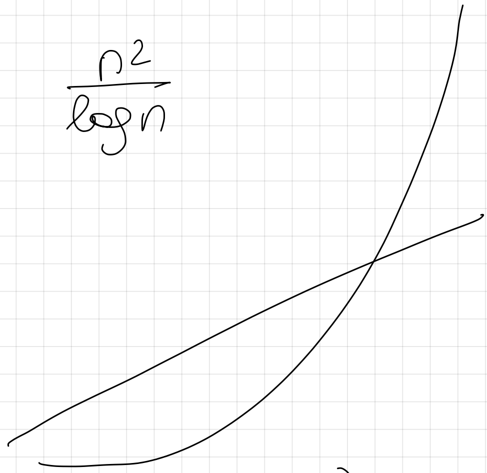
$n^2 + n + 7$

n

$\log n$

$n \log n$

$\frac{n^2}{\log n}$

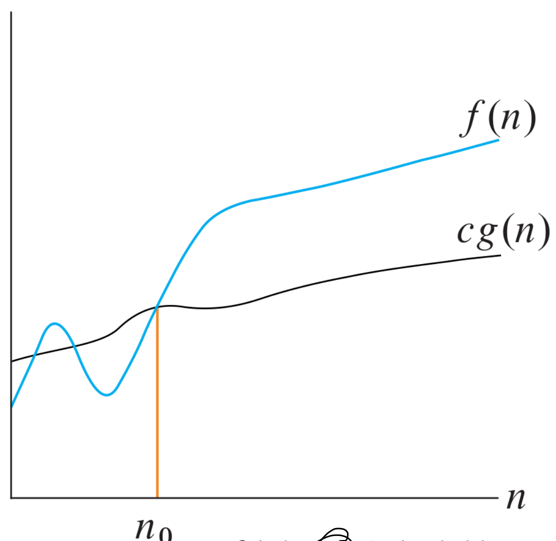


$$\binom{n}{2} = \frac{n(n-1)}{2} = \frac{n^2}{2} - \frac{n}{2} \in O(n^2)$$

Notazione Ω

$f(n) \in \Omega(g(n))$

$\Omega(g(n)) = \{ f(n) \mid \exists\, c, n_0$ costanti positive t.c. $f(n) \geq c \cdot g(n) \geq 0,\ \forall n \geq n_0 \}$



$f(n) = \Omega(g(n))$

$g(n)$: limite inferiore asintotico per $f(n)$

$\downarrow$

$f(n)$ cresce almeno come $g(n)$, a meno di un fattore costante

$\geq$

ESEMPI

$$f(n) = 7n^3 + 100n^2 - 20n + 6 \in \Omega(n^3)$$

$$\in \Omega(n^2)$$

$$\in \Omega(\log n)$$

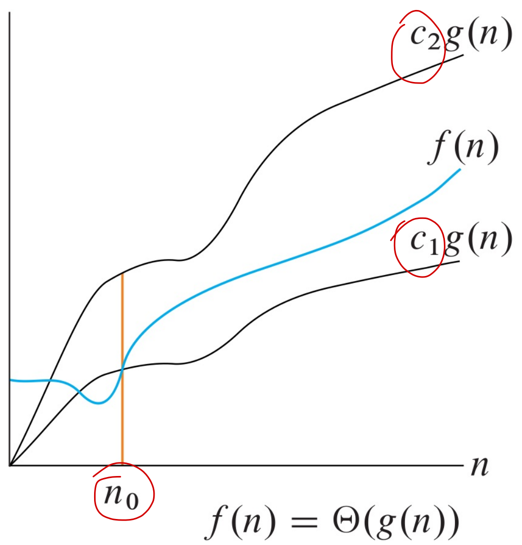
$$\in \Omega(n)$$

Esempi di funzioni in $\Omega(n^2)$:

$$n^2,\ n^3,\ \frac{n^4}{\log n},\ 2^n \in \Omega(n^2)$$

Notazione Θ

$f(n) \in \Theta(g(n)) \iff f(n) \in O(g(n))$ e $f(n) \in \Omega(g(n))$



$f(n) = \Theta(g(n))$

limite asintotico stretto

↓

$f(n)$ cresce come $g(n)$, a meno di fattori costanti

$\approx$

$$\Theta(g(n)) = \{ f(n) \mid \exists\, c_1, c_2, n_0 \text{ costanti positive tali che } 0 \le c_1 \cdot g(n) \le f(n) \le c_2 \cdot g(n),\ \forall n \ge n_0 \}$$

ESEMPIO

$$f(n) = 7n^3 + 10n^2 - 100n$$

$$f(n) \in \Theta(n^3)$$

$$\cancel{f(n) \in \Theta(n^2)}$$

$$f(n) = \Omega(n^2)$$

$$f(n) \in O(n^4)$$

$$\cancel{f(n) \in \Theta(n^4)}$$

Notazione o-piccolo

$$f(n) \in o(g(n)) \qquad \lim_{n \to +\infty} \frac{f(n)}{g(n)} = 0$$

$f(n)$ cresce più lentamente di $g(n)$

$<$

Notazione ω-piccolo

$$f(n) = \omega(g(n)) \qquad \lim_{n \to +\infty} \frac{f(n)}{g(n)} = +\infty$$

$>$

$f(n)$ cresce più velocemente di $g(n)$

ESEMPI

$f(n) = n^3$

$f(n) = o(n^4)$ $\quad$ $f(n) = \omega(n)$

$= o(n^3 \log n)$ $\quad$ $f(n) = \omega(n^2 \log n)$

$\cancel{= o(n^3)}$ $\quad$ $\cancel{f(n) = \omega(n^3)}$

$\cancel{f(n) = \omega(n^4)}$

PROPRIETÀ

TRANSITIVA

$f(n) = \Theta(g(n))$ e $g(n) = \Theta(h(n)) \Rightarrow f(n) = \Theta(h(n))$

vale anche per O, Ω, o, ω

RIFLESSIVA

$f(n) = \Theta(f(n))$ $\quad$ vale anche per O, Ω

SIMMETRICA

$f(n) = \Theta(g(n)) \iff g(n) \in \Theta(f(n))$

SIMMETRIA - TRASPOSTA

$f(n) = O(g(n)) \iff g(n) = \Omega(f(n))$

$f(n) = o(g(n)) \iff g(n) = \omega(f(n))$

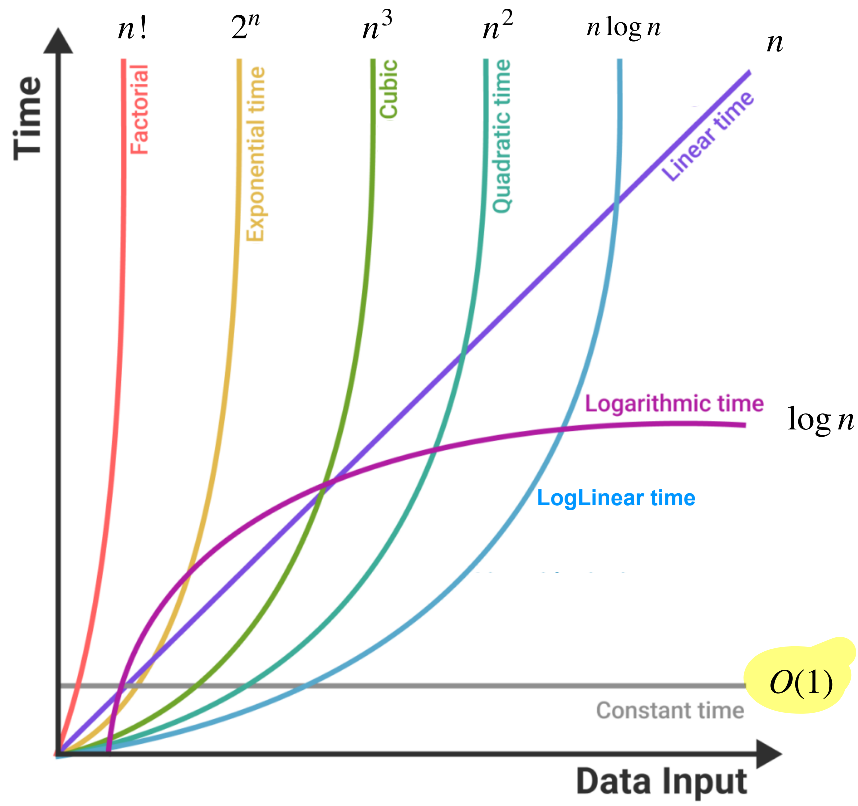
Notazione per costanti : $O(1)$

$\underbrace{T(n) = O(1)}$ $\iff$ costo costante

$f(n) \in O(1)$

$\exists\, c, n_0$ costanti positive t.c.

$\forall n \geq n_0 \quad f(n) \leq c$



① $f_1(n) = O(g_1(n))$ $\qquad f_2(n) = O(g_2(n))$

$$f_1(n) + f_2(n) = O(\max\{g_1(n), g_2(n)\})$$

② se $f(n)$ è un polinomio di grado d

$$f(n) \in \Theta(n^d)$$

$$f(n) = c_1 \cdot n^d + c_2 \cdot n^{d-1} + \dots + n + c$$

$$f(n) = 3n^2 \in \begin{cases} O(n^2) \\ O(n^4) \\ \cancel{O(n \log n)} \\ O(n^2 \log n) \\ \Theta(n^2) \\ \cancel{\Theta(n^2 \log\log n)} \\ \Omega(n^2) \\ \cancel{\Omega(n^2 \log n)} \\ \Omega\left(\frac{n^2}{\log n}\right) \end{cases}$$

$n^2 < n^4$

ESERCIZIO

Ordinare le funzioni

$\cancel{\sqrt{n}}$, $\cancel{n}$, $\cancel{\log \log n}$, $\cancel{\log n}$, 2^n, 3^n, n^n,

$\cancel{n \log n}$, $\cancel{n^2}$, $\cancel{\log^2 n}$, $\cancel{n \log^2 n}$, $\cancel{2}$

rispetto al tasso di crescita

2, $\log\log n$, $\log n$, $\log^2 n$, $\sqrt{n}$, n, $n \log n$,

$n \log^2 n$, n^2, 2^n, 3^n, n^n