# Problema

A =

| 7 | 4 | 3 | 4 | 2 | 4 | 1 | 11 | 3 | 5 | 2 | 2 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |

Intero $K = 9$

$n = |A|$

OUTPUT: $i, j$ t.c. $K = A[i] + A[j]$

```
for (i = 0 ; i < n ; i++)
    for (j = i+1 ; j < n ; j++)
        if (A[i] + A[j] == k)
            return <i, j>

return <NULL, NULL>
```

A è ordinato :

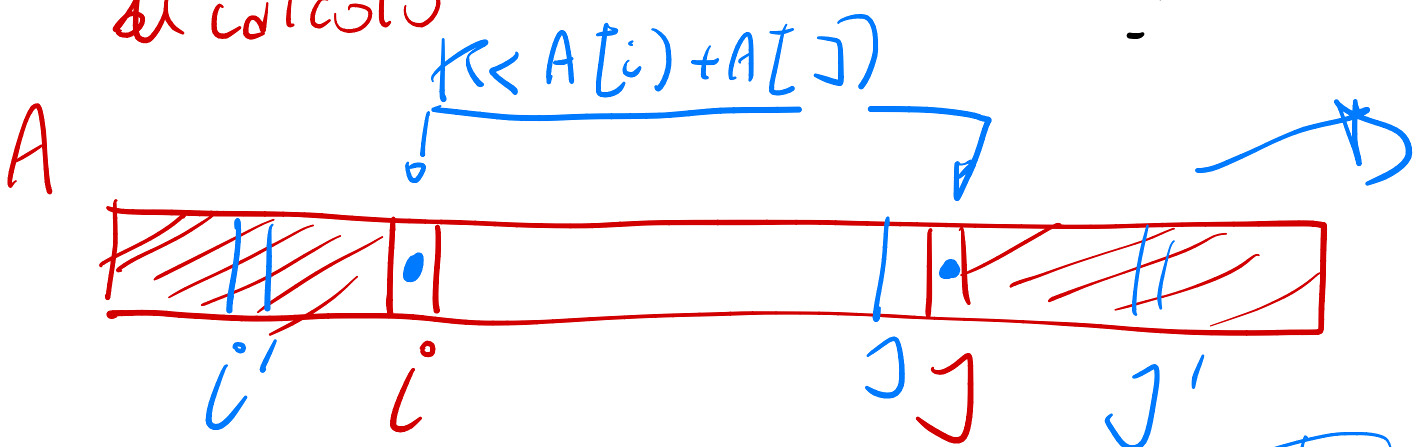
| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 2 | 2 | 3 | 3 | 4 | 4 | 4 | 5 | 7 | 11 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| i | | | | | | | | | | | j |

```
i=0, j=n-1
while (i<j) and k ≠ A[i]+A[j]):
    if (k < A[i]+A[j])) j--
    else // >
        i++
if (i<j) return <i,j>
    else return <NULL,NULL>
```

induzione

1. correttezza: dare sempre la risposta corretta
2. complessità: conteggio delle risorse utilizzate (tempo, spazio)

modello di calcolo




Invariante: $\forall\, i' \leq i \;\; \forall\, j' \geq j$
$A[i'] + A[j'] \neq k$

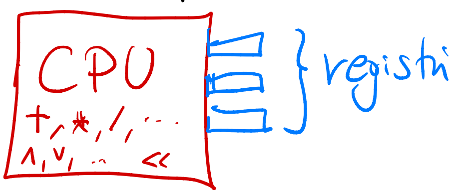
$i' \neq i$ or $j' \neq j$

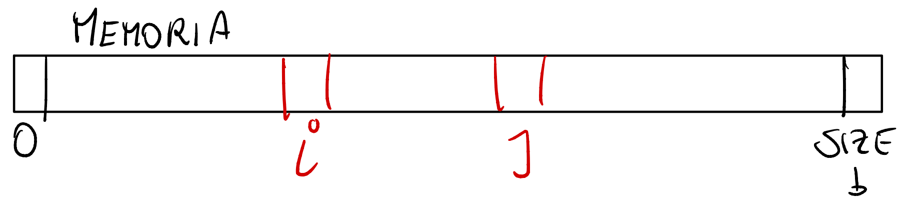
V ① Modello computazionale per il costo

② Analisi del costo computazionale degli algoritmi

RAM = Random Access Memory

modello semplice basato su quello di Von Neumann




- operazioni logico-aritmetiche, confronto
- operazioni manipolazione bit
- operazioni di controllo
- operazioni di trasferimento REGISTRI-MEMORIA

} costo uniforme

## ② Costo computazionale

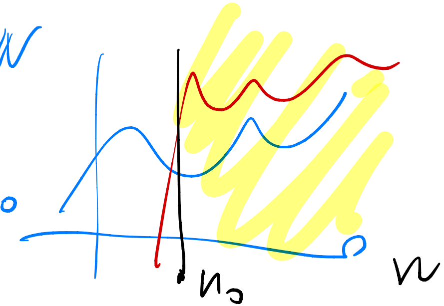
✓ • tempo di esecuzione : conteggio del numero di passi eseguiti da un algoritmo (istruzione RAM)
• spazio di lavoro : numero di celle di memoria utilizzate (talvolta non si conteggiano quelle contenenti i dati di ingresso)
(• energia)

Analisi asintotica : per $n \to +\infty$ dimensione dei dati d'ingresso un algoritmo richiede $f(n)$ tempo/spazio

Notazione asintotica : costanti moltiplicative e ordini inferiori sono tralasciati perché $n \to +\infty$

$$f(n) = \frac{3}{2}n^3 + n\lg^2 n - 5n + 18 = \Theta(n^3)$$

$O(f(n))$ = insieme delle funzioni $g : \mathbb{N} \to \mathbb{N}$
t.c. esistono costanti $c > 0$ e $n_0 \geq 1$
per cui vale $g(n) \leq c \cdot f(n) \quad \forall n > n_0$




$g(n) = O(f(n))$

$\Omega(f(n))$

$g(n) \geq c \cdot f(n) \quad \forall n > n_0$ [varianti: per infiniti valori di $n$]

$\Theta() = O + \Omega$

Algoritmo → derivare la sua complessità $f(n)$?

- operazioni della RAM hanno costo costante $O(1)$

es. `int x = 5;` $O(1)$

`x = x + y;` $O(1)$

- Costrutto condizionale

$$\text{IF } \underbrace{(\text{guardia})}_{\text{sempre valutata}} \; \underbrace{\{\text{BLOCCO-THEN}\}}_{} \text{ ELSE } \underbrace{\{\text{BLOCCO-ELSE}\}}_{}$$

uno solo

$$\text{Costo} = \text{costo}(\text{guardia}) + \max\{\text{costo}(\text{BLOCCO-THEN}), \text{costo}(\text{BLOCCO-ELSE})\}$$

- costrutto iterativo

$$\text{FOR } (\underbrace{i=0}_{O(1)};\ \underbrace{i<m}_{O(1)};\ \underbrace{i++}_{O(1)})\ \{\underbrace{\text{CORPO}}_{t_i = \text{costo(CORPO) all'iterazione } i}\}$$

$$\text{costo} = m + \sum_{i=0}^{m-1} t_i$$

$$\text{WHILE } (\text{guardia})\ \{\underbrace{\text{CORPO}}_{t_i = \text{costo(CORPO)}}\}$$

→ $m$ = # volte in cui guardia è vera

$t'_i$ = costo(guardia)

$$\text{costo} = \sum_{i=0}^{m} t'_i + \sum_{i=0}^{m-1} t_i$$

- chiamata di funzione

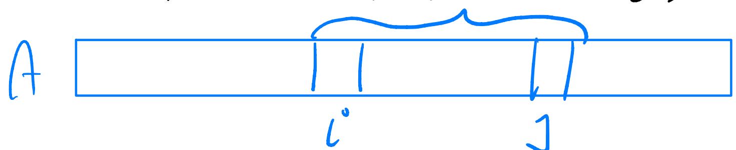
  costo = costo(CORPO_FUNZIONE)

- costo(BLOCCO) = somma dei costi delle singole istruzioni/costrutti che lo compongono

# Esempio: segmento di somma massima

Array A di interi, di cui almeno uno positivo e uno negativo

segmento $A[i,j] = A[i]\,A[i+1] \cdots A[j]$ $\quad i \le j$




$$\text{somma}(i,j) = \sum_{i \le k \le j} A[k]$$

Trovare $i, j$ t.c. $\text{somma}(i,j) \geq \text{somma}(i',j') \quad \forall\, i' \le j'$