Algoritmi e Strutture Dati (18/10/2021)

* Code con priorità

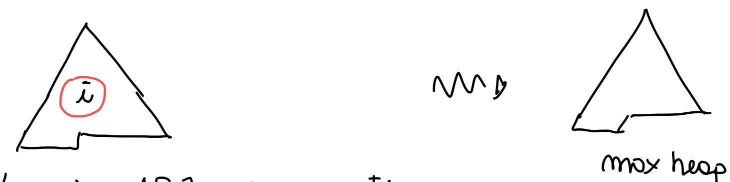
max heap A

- Max (A) $O(1)$
- Extract Max (A) $O(\log n)$

- Insert (A , k)
- Increase Key (A, i, δ)
- Decrease Key (A, i, δ)
- Remove (A, i)

?

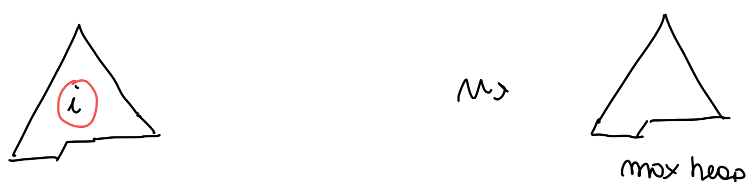
Max Heapify (A, i)



$\forall j \neq i \quad A[i] \geq$ discendenti

$O(\log n)$

Max Heapify Up (A, i)

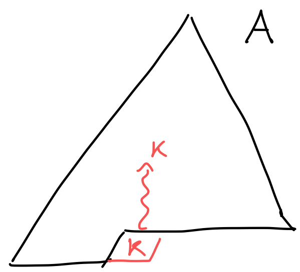


$\forall j \neq i \quad A[i] \leq$ antenati

$O(\log n)$

* Insert (A, k)

    A.size = A.size + 1
    A [A.size] = k
    MaxHeapifyUp (A, A.size)



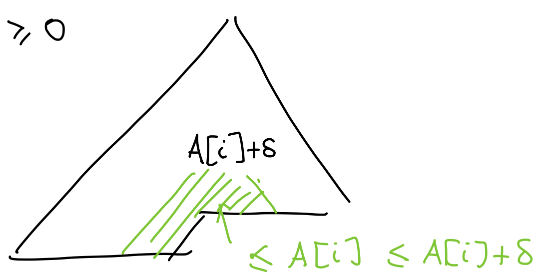
$O(\log n)$

* Increase Key (A, i, δ) $\delta \geq 0$

    A[i] = A[i]+δ

    Max Heapify Up (A, i)

$O(\log n)$



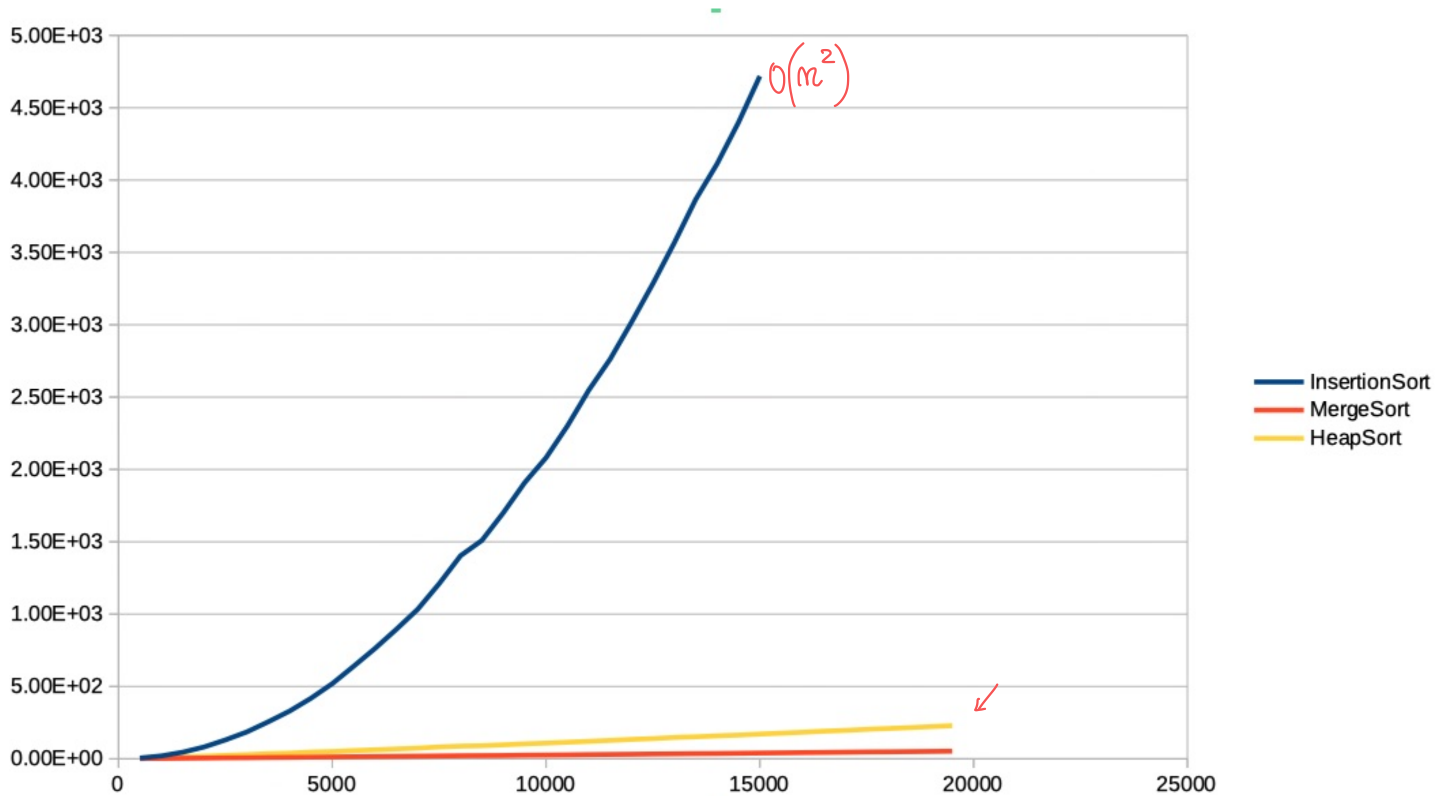
* Decrease Key (A, i, δ) $\delta \geq 0$

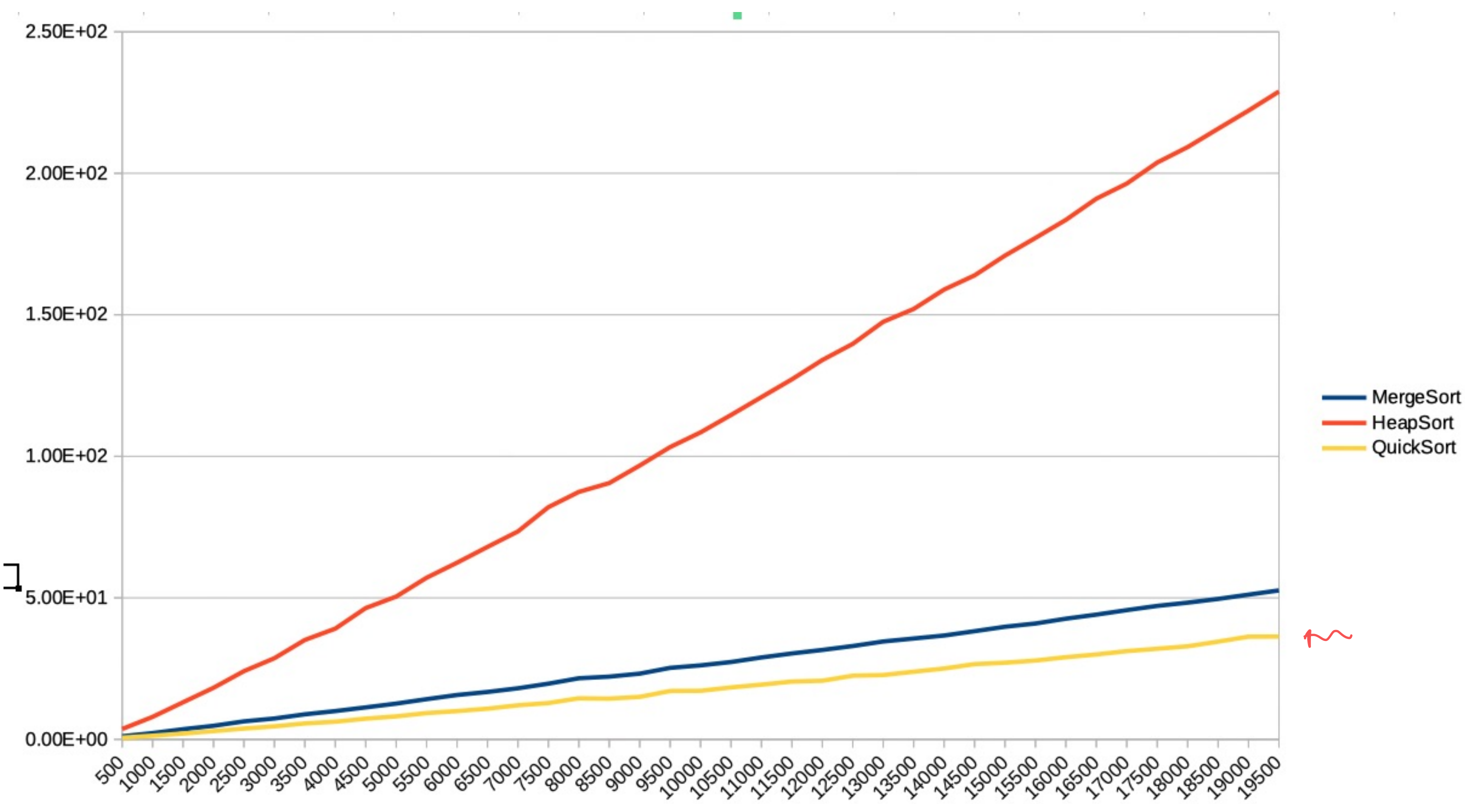
* Remove (A, i)

ESERCIZIO

Quick Sort

Tempo di esecuzione (ms) al variare del # elementi (media su 10 istanze casuali)





* $\boxed{\text{Tony Hoare} \quad 1961}$

→ complessità caso peggiore $\Theta(n^2)$

→ " caso medio $\Theta(n \log n)$

→ costanti (moltipl. / additive) sono basse

→ randomizzazione induce caso medio $\Theta(n \log n)$

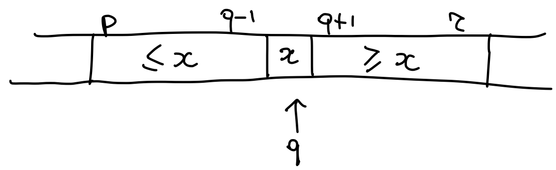
→ in loco

* Divide et impera

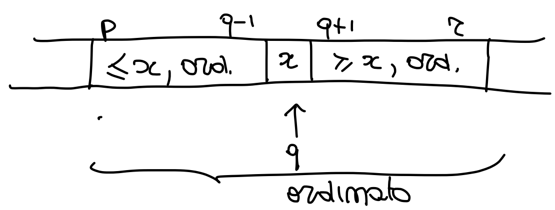
per ordinare $A[p..r]$

→ partition : → sceglie un elemento x in $A[p..r]$: pivot

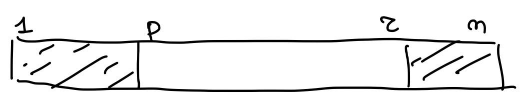
→ partiziona $A[p..r]$



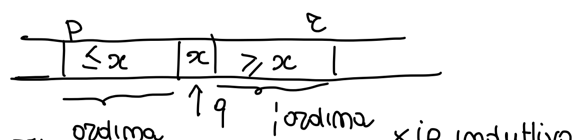
→ impera : ordina (ricorsivamente) $A[p..q-1]$ e $A[q+1..r]$



→ combina : $\emptyset$



QuickSort (A, p, r)
  if p < r
    q = Partition (A, p, r)
    QuickSort (A, p, q-1)
    QuickSort (A, q+1, r)

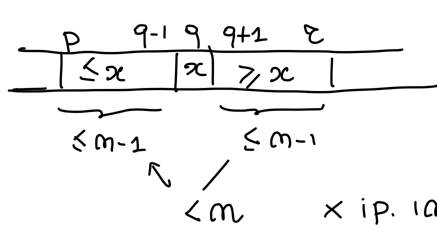


⇓ $A[p..r]$ ordinato

Correttezza : induz. su $n = r - p + 1$

$(n \leq 1)$ array con 0 o 1 elem. è ordinato, non faccio niente

$(n > 1)$



x ip. induttiva QuickSort ordina $A[p..q-1]$, $A[q+1..r]$

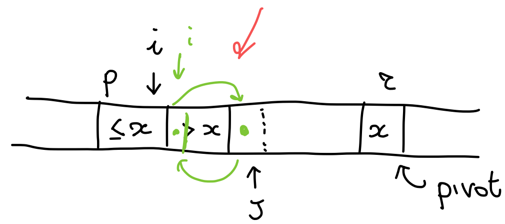
quindi

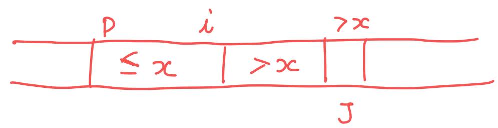
$$\underbrace{\underset{\text{ordinato}}{A[p..q-1]} \leq \overset{x}{\overset{\shortparallel}{A[q]}} \leq \underset{\text{ordinato}}{A[q+1..r]}}_{A[p..r] \text{ ordinato}}$$

* Partizionamento

(1ª) Partizionamento deterministico

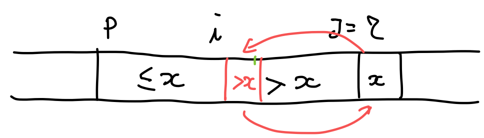
pivot $x = A[r]$





Partition (A, p, r)

```
x = A[r]
i = p-1
for j = p to r-1
    if A[j] ≤ x
        i = i+1
        A[i] ↔ A[j]
    // else
A[i+1] ↔ A[r]
return i+1
```



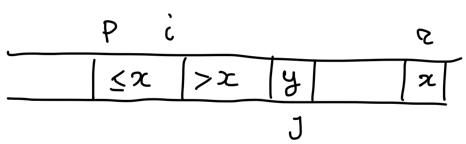
Correttezza: invariante
- $A[p..i] \leq x$
- $A[i+1..j-1] > x$
- $A[r] = x$

inizializzazione $i = p-1$, $j = p$

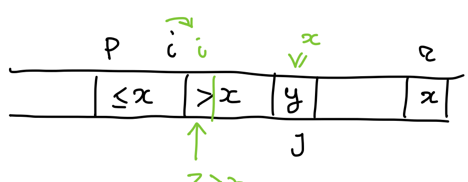
$A[p..i] = A[p..p-1] = \emptyset$

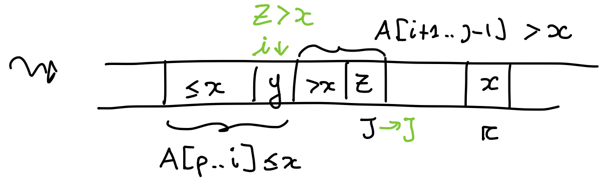
$A[i+1..j-1] = A[p..p-1] = \emptyset$

conservazione:

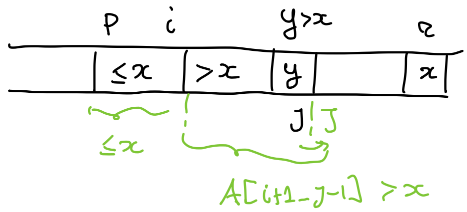


$y \leq x$? si

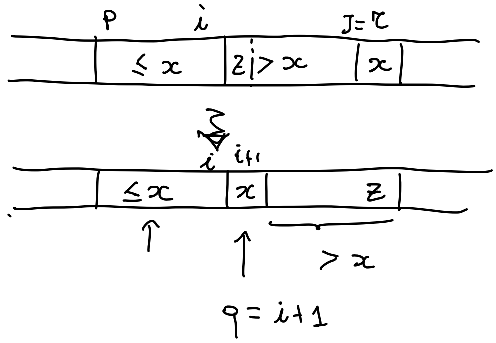




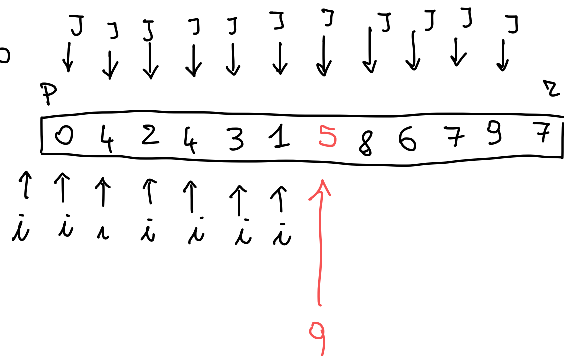
NO



conclusione



Esempio

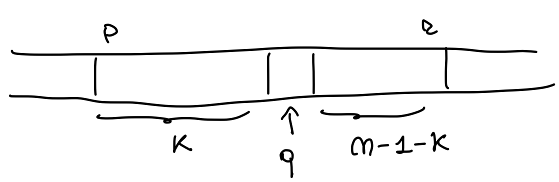


costo Partition : $an + b = \Theta(n)$

Costo del QuickSort

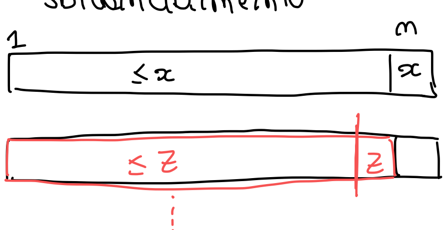
QuickSort (A, p, r)
if p < r
q = Partition (A, p, r)
QuickSort (A, p, q-1) ← dim $n_1 = k$
QuickSort (A, q+1, r) ← dim $n_2 = n-1-k$

$$T(n) = \Theta(n) + T(k) + T(n-1-k)$$



* Caso peggiore

massimo sbilanciamento



$$T(n) = \Theta(n) + T(n-1) + \underbrace{T(0)}$$

$$T(n) = T(n-1) + \underbrace{\Theta(n)}_{an+b}$$

$T(n)$
$an+b$
$\vdots$
$T(n-1)$
$a(n-1)+b$
$\vdots$
$T(n-2)$
$a(n-2)+b$
$\vdots$
c

$$T(n) = \sum_{i=n}^{1} (ai+b)$$

$$= \sum_{i=1}^{n} (ai+b)$$

$$= a\sum_{i=1}^{n} i + nb$$

$$= a\frac{(n+1)n}{2} + nb$$

$$\sim a'n^2 + b'n + c'$$

per essere preciso:

① concreto

$$T(n) = \begin{cases} c & n \leq 1 \\ T(n-1) + an+b & n > 1 \end{cases}$$

ipotesi: $T(n) = a'n^2 + b'n + c'$

prova induttiva che determina a', b', c'

ESERCIZIO

② astratto

$$T(n) = \Theta(n^2)$$

(a) $T(n) = O(n^2)$

(b) $T(n) = \Omega(n^2)$

(a) $\boxed{T(n) \leq c_1 n^2}$ $\quad c_1 > 0$, n "grande"

$$T(n) = T(n-1) + \underbrace{\Theta(n)}_{\leq dn} \quad d > 0$$

$$\leq T(n-1) + dn$$

$n-1 < n$ ip. ind. $T(n-1) \leq c_1 (n-1)^2$

$$\le c_1 (n-1)^2 + dn$$

$$= c_1 n^2 - 2c_1 n + c_1 + dn$$

$$= c_1 n^2 - \underbrace{\big(\underbrace{(2c_1 - d)} n - c_1 \big)}$$

$$2c_1 - d > 0 \qquad c_1 > \frac{d}{2}$$

$\vdots$

$$\le c_1 n^2$$

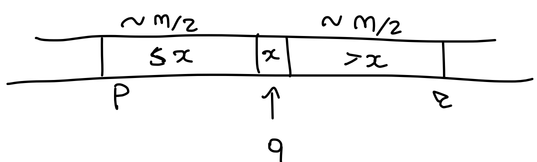
(b) $T(n) \ge c_2 n^2$

$\vdots$ ESERCIZIO

$\Rightarrow T^{QS}_{max}(n) = \Theta(n^2)$

* <u>Caso migliore</u>

partizione perfettamente bilanciata.



$$T(n) = \underset{\substack{\uparrow \\ a=2}}{2}\, T\Big(\frac{n}{\underset{\substack{\uparrow \\ b=2}}{2}}\Big) + \underset{\substack{\uparrow \\ f(n)}}{\Theta(n)}$$

Master Theorem

$$n^{\log_2 2} = n^1 = n \qquad f(n) = \Theta(n)$$

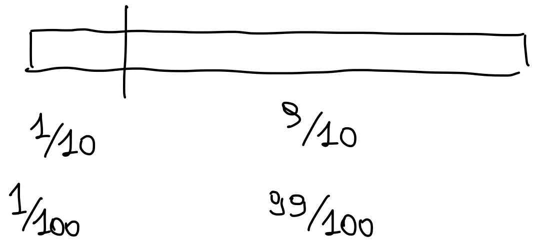
$\Rightarrow$ CASO 2 $\quad T(n) = \Theta\big(n^{\overset{1}{\log_b a}} \log n\big) = \Theta(n \log n)$

$$T^{QS}_{min}(n) = \Theta(n \log n)$$

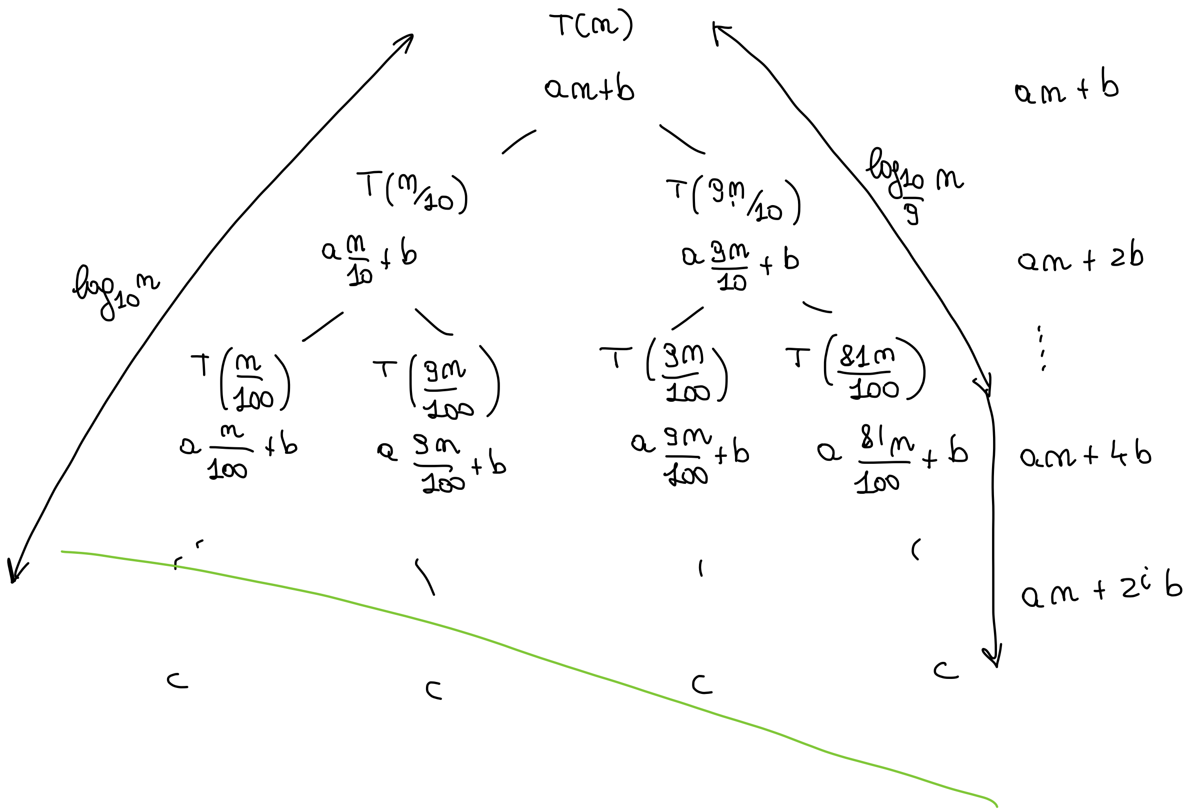
* <u>Caso Medio</u>

$$T^{QS}_{med}(n) = \Theta(n \log n)$$

(OSSERVAZIONE 1) Partizionamento proporzionale

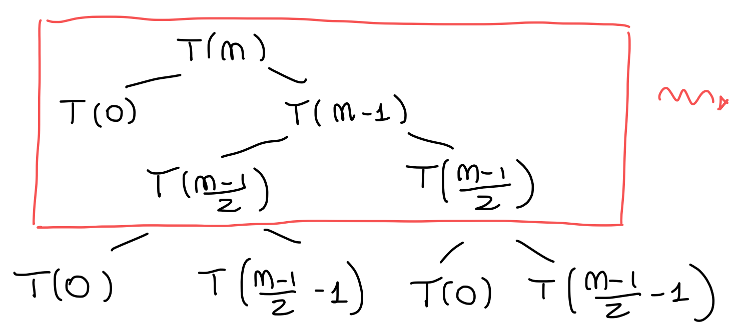


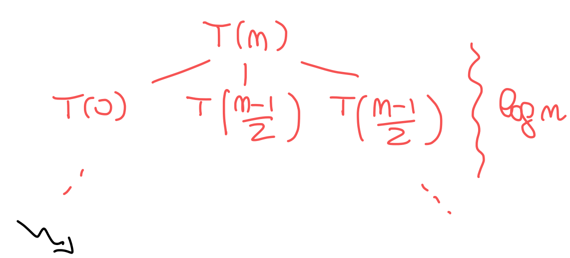
$$T^{QS}(n) = T^{QS}\left(\tfrac{1}{10}n\right) + T^{QS}\left(\tfrac{9}{10}\right) + \underbrace{\Theta(n)}_{an+b}$$



$$T(n) \sim a\, n \log_{\frac{10}{9}} n + bn$$
$$= \Theta(n \log n)$$

(OSSERVAZIONE 2) Alternanza di partizionamenti "ottimi" e "pessimi"





$T(n) = \Theta(n \log n)$