ESAME DI ALGORITMI

Università degli Studi di Catania
Corso di Laurea Triennale in Informatica
17 febbraio 2025

Si risolvano i seguenti esercizi in un tempo non superiore a 3 ore. Si abbia cura di consegnare la risoluzione dei primi 3 esercizi in un foglio (FOGLIO A) separato da quello utilizzato per la consegna degli ultimi 3 esercizi (FOGLIO B).

—— FOGLIO A ——

1. Si consideri un array $A = [1, 2, 3, \dots, n]$ di n interi distinti ordinati in modo crescente. L'obiettivo è costruire da A un *heap binario massimo*, applicando l'algoritmo classico `BuildMaxHeap`, che esegue la procedura `MaxHeapify` sui nodi interni, partendo dal basso verso l'alto.
   (a) Calcolare il numero di scambi (swap) applicati durante l'esecuzione dell'algoritmo per $n = 10$.
   (b) Fornire una **stima asintotica** del numero di scambi effettuati da `BuildMaxHeap` in funzione della dimensione n dell'array. Motivare la risposta.

2. Scrivere una procedura `UpdateKey(H, i, k)` che aggiorni, in un Heap Binario Massimo, la chiave in posizione i con il nuovo valore k, e ripristini le proprietà dell'heap massimo. L'algoritmo deve gestire correttamente entrambi i casi in cui $k > H[i]$ e $k < H[i]$. Analizzare la complessità nel caso peggiore della procedura implementata, in funzione della dimensione dell'heap n.

3. Fornire un esempio concreto di un albero rosso-nero valido contiene 10 chiavi distinte in cui un'operazione di cancellazione provoca la diminuzione dell'altezza nera dell'albero. Successivamente, fornire un'altro esempio concreto di un albero rosso-nero valido contiene 10 chiavi distinte in cui un'operazione di inserimento provoca l'aumento dell'altezza nera dell'albero. Per entrambi gli esempi, disegnare la configurazione dell'albero prima e dopo le operazioni.

—— FOGLIO B ——

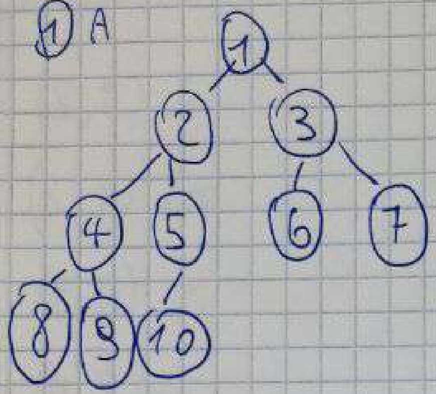
4. Si risolva l'equazione di ricorrenza $T(n) = 9T\left(\frac{n}{b}\right) + n\log^2 n$, al variare del parametro reale $b > 1$ utilizzando il metodo Master. Si stabilisca inoltre quale delle seguenti condizione sono soddisfatte dalla soluzione $T(n)$:
   (i) $T(n) = \Theta(n^2)$;
   (ii) $T(n) = \Omega(n)$;
   (iii) $T(n) = o(n \log^3 n)$.

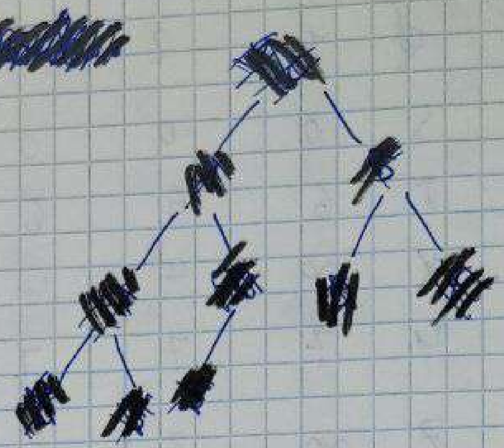
5. Si scriva la formula ricorsiva utilizzata dall'algoritmo ALL-PAIRS-SHORTEST-PATHS basato sulla moltiplicazione di matrici e si simuli tale algoritmo per trovare la tabella (matrice) dei cammini minimi tra tutte le coppie di vertici del grafo pesato definito dalla seguente matrice di adiacenza

$$W = \begin{pmatrix} 0 & 1 & \infty & 2 \\ \infty & 0 & 2 & \infty \\ -1 & \infty & 0 & \infty \\ \infty & \infty & 1 & 0 \end{pmatrix}.$$

6. All'aeroporto di Catania si gestiscono ogni giorno 114 voli, ciascuno con un orario di arrivo e uno di partenza. Ogni gate può essere utilizzato da un solo volo alla volta. Il vostro obiettivo è selezionare e pianificare i voli che verranno assegnati domani al gate 15, assicurandovi che non ci siano sovrapposizioni negli orari di utilizzo del gate, e che il numero di voli serviti sia il massimo possibile. (i) Quale problema tra quelli affrontati a lezione è strettamente collegato a questo scenario e può essere usato come modello per risolverlo? (ii) Quale tecnica algoritmica tra quelle studiate si può applicare per trovare una soluzione efficiente al problema? (iii) Dimostrare che il problema gode della proprietà di scelta greedy.

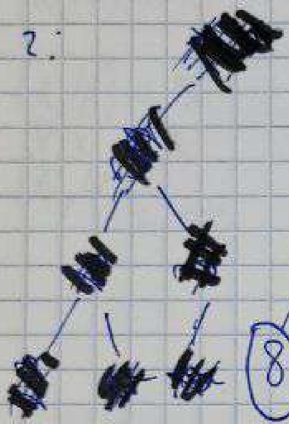
① A



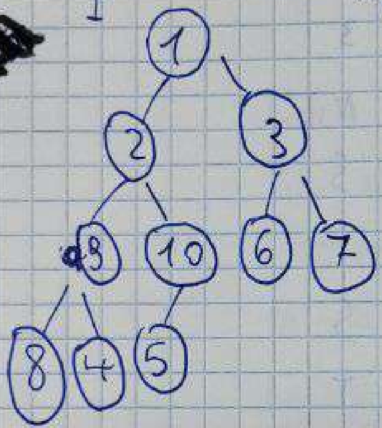


scambi = IIIIIIII = SWAP = 8

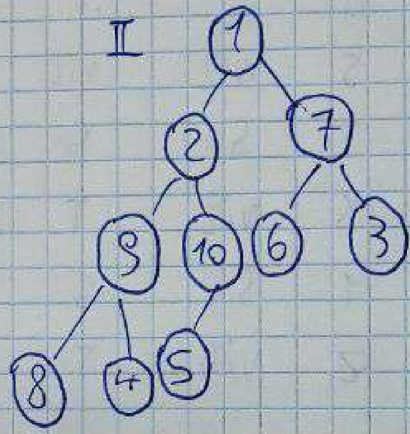
2:



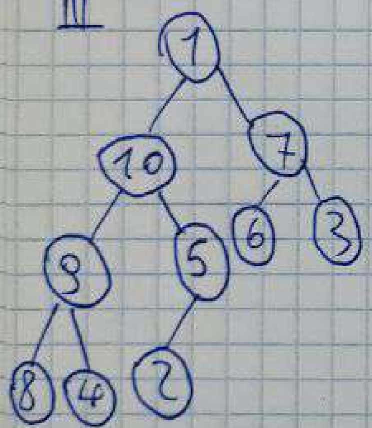
I



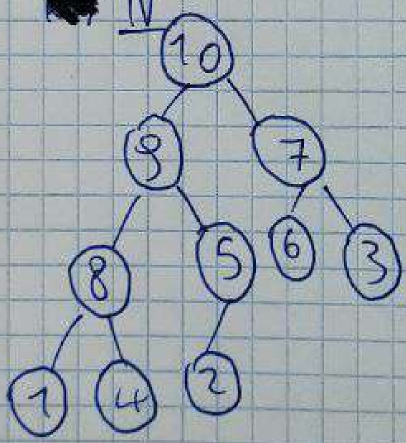
II



III



IV



① B nel caso peggiore ci saranno O(n) scambi

COMPITO 17/02/2025

La precedente data non è quella effettiva del compito, ma è quella riportata sul compito (errore di scrittura) l'appello di riferimento è quello di aprile.

②

UP

OLD

H[

iF

EL

EN

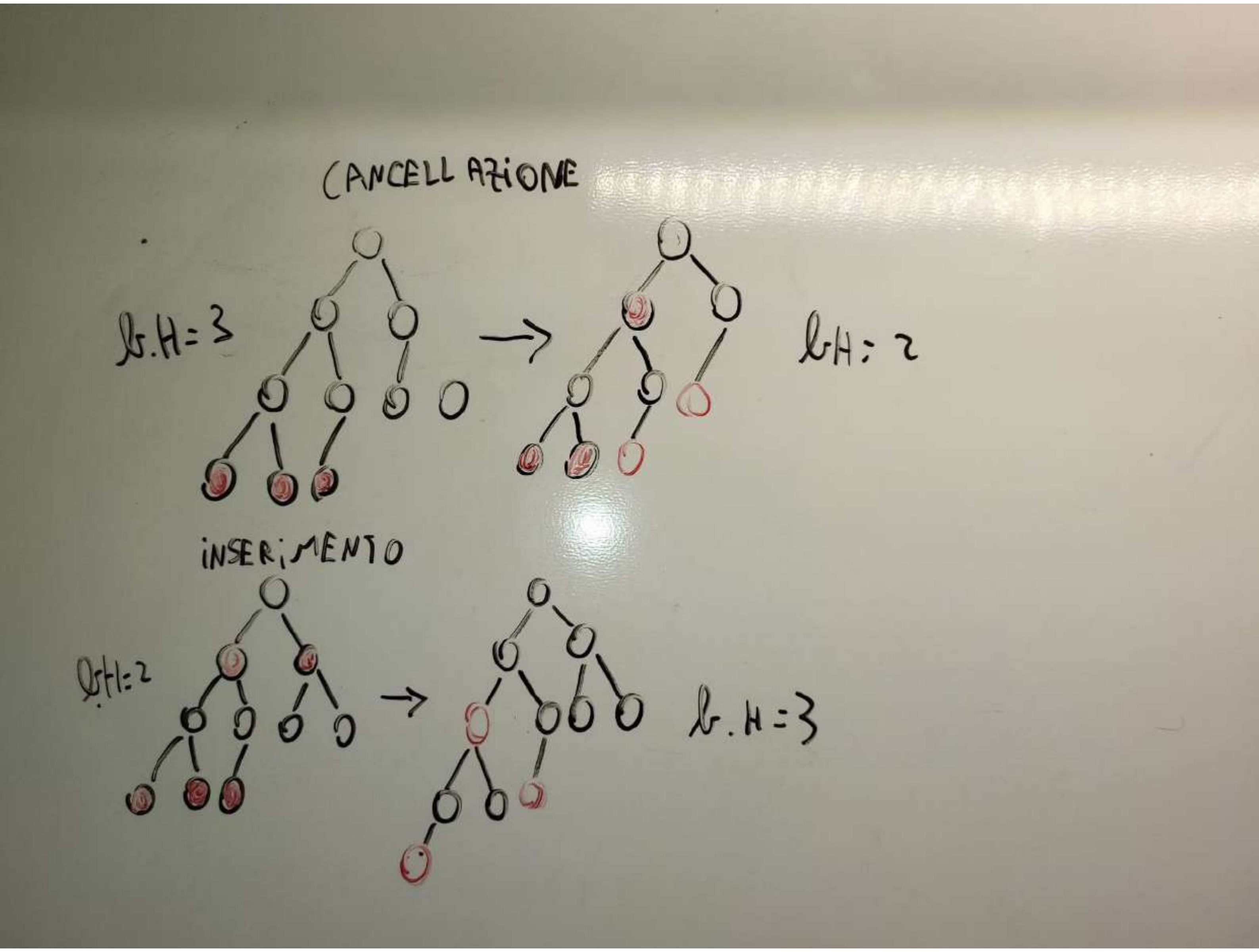
* L

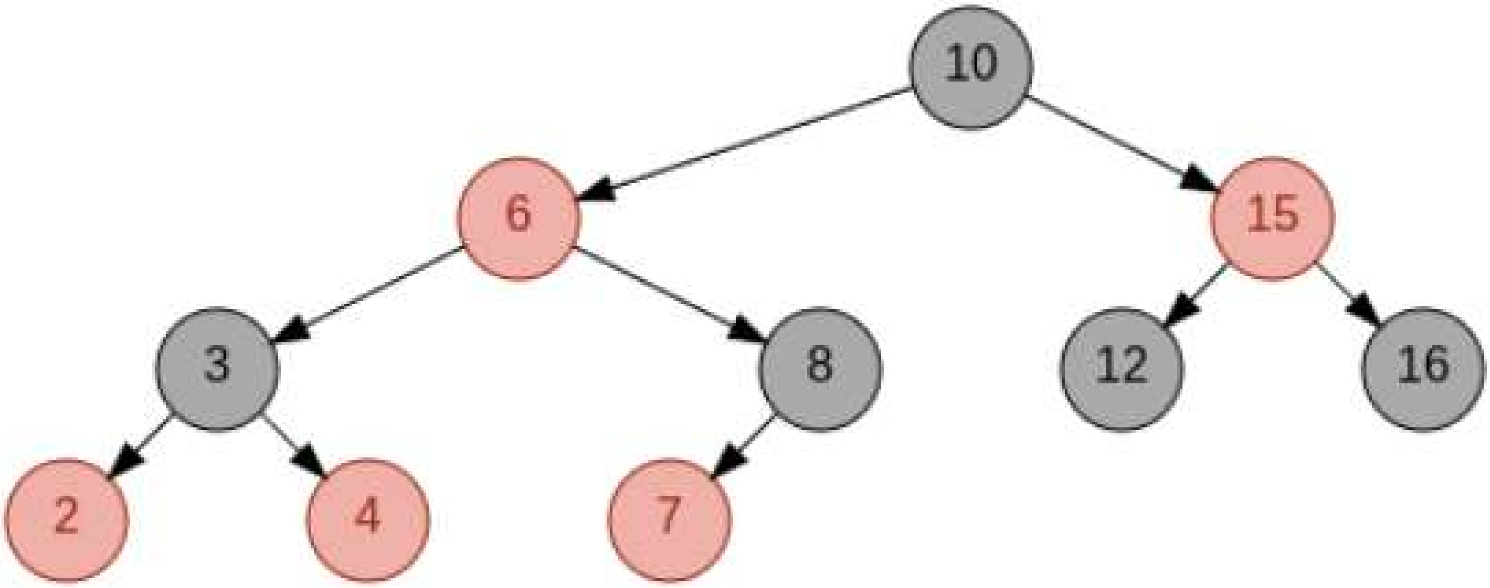
RA

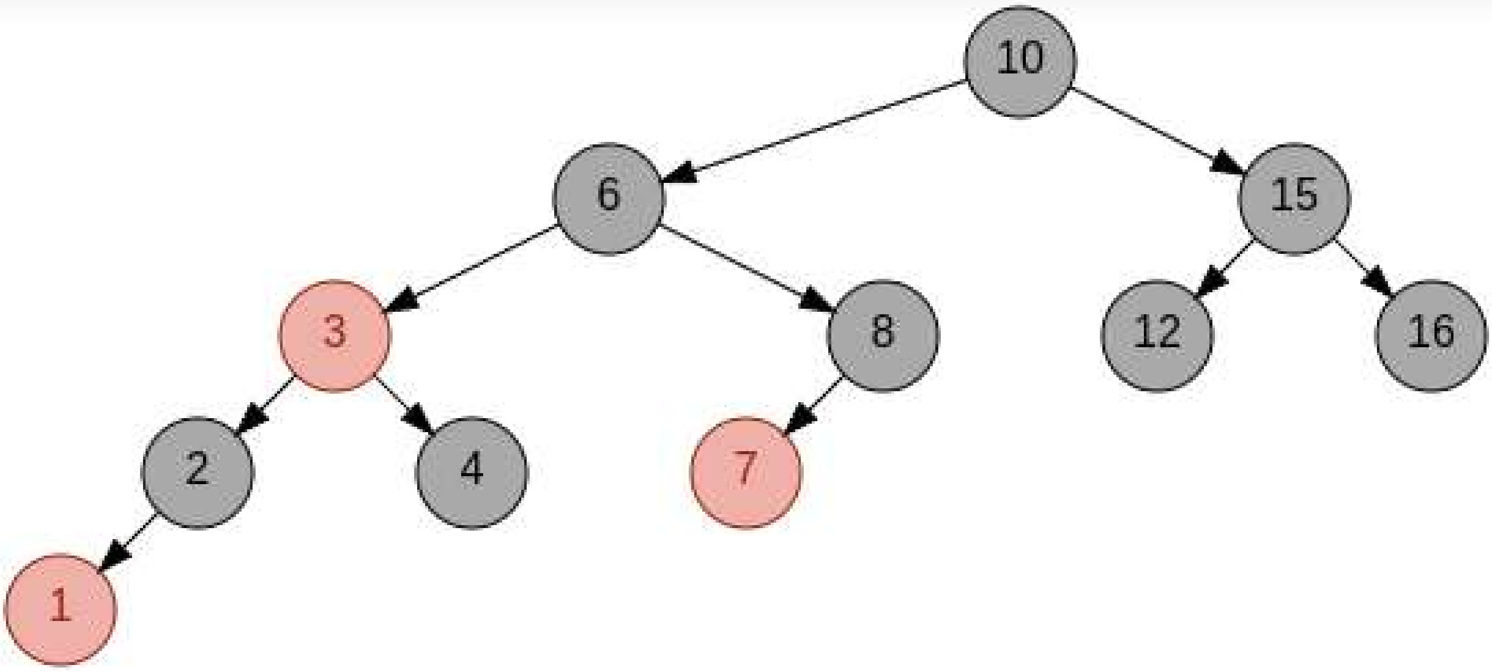
②

```
UPDATE-KEY (H, i, k)

OLD = H[i]

H[i] = k

if  k > OLD

    WHILE  i > 1   AND   H[i].PARENT < k   DO

        SWAP( H[i], H[i].PARENT)
        i = ⌊i/2⌋

    END WHILE

ELSE if  k < OLD

    MAX-HEAPIFY (H, i)

END if
```

* LO "SPOSTAMENTO" MASSIMO CHE IL NODO PERCORRE È DALLA RADICE ALLA FOGLIA O VICEVERSA, DUNQUE $O(\log n)$


CANCELLAZIONE
b.H = 3
bH: 2
INSERIMENTO
b.H: 2
b.H = 3


10
6
15
3
8
12
16
2
4
7


10
6
15
3
8
12
16
2
4
7
1

$b > 1$

4A $T(n) = 9T\left(\frac{n}{b}\right) + n\log^2 n$

$w(n) = n^{\log_b 9}$

I $w(n)$ è un upper bound di $f(n)$ per $b > 9$

$O\left(n^{\log_b 9 - \varepsilon}\right) \Rightarrow \Theta\left(n^{\log_b 9}\right)$

II $w(n)$ è asintoticamente simile a $f(n)$ per $b = 9$

$\Theta\left(n^{\log_9 9} \cdot \log^k n\right) = \Theta\left(n \cdot \log^k n\right) \Rightarrow \Theta\left(n\log^3 n\right)$

III $w(n)$ è un lower bound per $1 < b < 9$

$\Omega\left(n^{\log_b a + \varepsilon}\right) \Rightarrow \Theta\left(n\log^2 n\right)$

4B

$$T(n) = \begin{cases} \Theta(n^2) \rightarrow \text{è soddisfatto per } b = 3 \\ \Omega(n) \rightarrow \text{è soddisfatto per } 1 < b \leq 9 \\ o(n\log^3 n) \rightarrow \text{è soddisfatto per } b > 9 \end{cases}$$

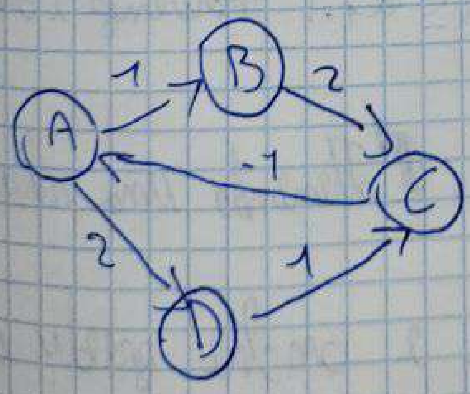
⑤

$$l_{ij}^{(0)} = \begin{cases} 0 & \text{si } i = j \\ \infty & \text{si } i \neq j \end{cases}$$

$$l_{ij}^{(m)} = \min_{1 \le k \le n} \left\{ l_{ik}^{(m-1)} + w_{kj} \right\}$$

$$W = \begin{pmatrix} 0 & 1 & \infty & 2 \\ \infty & 0 & 2 & \infty \\ -1 & \infty & 0 & \infty \\ \infty & \infty & 1 & 0 \end{pmatrix} \quad L^{(2)} = \begin{pmatrix} 0 & 1 & 3 & 2 \\ -1 & 0 & 2 & \infty \\ -1 & 0 & 0 & 1 \\ 0 & \infty & 1 & 0 \end{pmatrix}$$

$$L^{(3)} = \begin{pmatrix} 0 & 1 & 3 & 2 \\ 1 & 0 & 2 & 3 \\ -1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} ; \; L^{(4)} = L^{(3)}$$



⑥ i) l'activity selection, si trova una configurazione di attività con cardinalità γ massima e $W = \sum_{i=1}^{\gamma} x_i.tempo$ minimo

ii) Si può utilizzare un algoritmo greedy, dove si va a preferire l'attività che ha un $T = ft - fs$ minore, ovvero l'attività disponibile al range temporale minore

iii) Presa S^* una soluzione ottima composta da $\{a_1, \dots, a_\gamma\}$ se vado a cambiare a_1 con l'attività a_i che in quel range avrà un T inferiore o uguale ad a_1, ottengo una soluzione $S' = \{a'_1, \dots, a_\gamma\}$ e $a'_1.T \leq a_1.T$ mantengo l'ottimalità di S^*, se reitero il processo γ volte, ottenendo una soluzione $S^\gamma = \{a'_1, \dots, a'_\gamma\}$ che sarà ottima e anche greedy in quanto $S^*.W \geq S^\gamma.W$