

# Algoritmi e Strutture Dati

Foglio 6  
28/03/2025

**Esercizio 1.** In un max-heap, dove potrebbe trovarsi il più piccolo elemento, supponendo che tutti gli elementi siano diversi?

**Esercizio 2.** Un array ordinato è un min-heap?

**Esercizio 3.** Usando la procedura MAX-HEAPIFY come modello, scrivete uno pseudocodice per la procedura MIN-HEAPIFY( $A, i$ ).

**Esercizio 4.** Dimostrate che il tempo di esecuzione nel caso peggiore di MAX-HEAPIFY su un heap di dimensione  $n$  è  $\Omega(n)$ .

Suggerimento: assegnate i valori ai nodi in modo che MAX-HEAPIFY sia chiamata ricorsivamente in ogni nodo di un cammino che scende dalla radice fino a una foglia.

**Esercizio 5.** Illustrate il funzionamento di BUILD-MAX-HEAP( $A$ ) sull'array

$$A = \langle 5, 3, 17, 10, 84, 19, 6, 22, 9 \rangle .$$

**Esercizio 6.** Supponete che gli oggetti in una coda di max-priorità siano semplicemente delle chiavi.

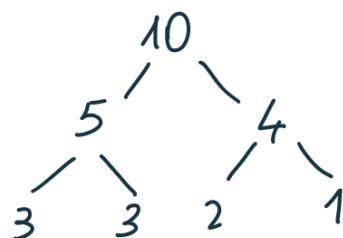
1. Illustrate il funzionamento di HEAP-EXTRACT-MAX( $A$ ) sull'heap

$$A = \langle 15, 13, 9, 5, 12, 8, 7, 4, 0, 6, 2, 1 \rangle .$$

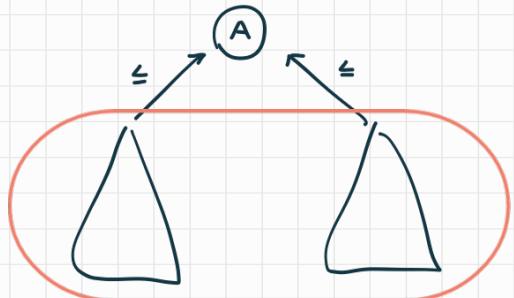
2. Illustrate il funzionamento di MAX-HEAP-INSERT( $A, 10$ ) sull'heap

$$A = \langle 15, 13, 9, 5, 12, 8, 7, 4, 0, 6, 2, 1 \rangle .$$

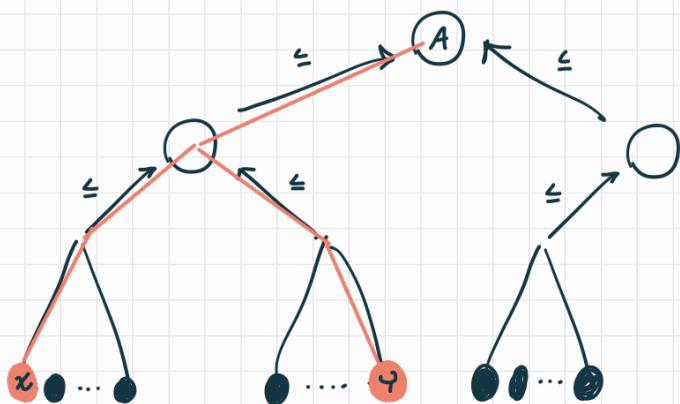
**Esercizio 7.** In un max-heap di dimensione  $n$ , è possibile implementare la procedura FIND-MIN che trova un elemento con chiave di valore minimo in  $O(\log n)$ ?



**Esercizio 1.** In un max-heap, dove potrebbe trovarsi il più piccolo elemento, supponendo che tutti gli elementi siano diversi?



Tutti gli elementi nei sottoalberi generati da un nodo A sono < del valore in A (elem. distinti)



Ad esempio, sicuramente gli elementi x, y sono < di tutti i loro rispettivi antenati



Il MINIMO sarà sicuramente tra le foglie !

→ se così non fosse, vorrebbe dire che il min. avrebbe almeno un figlio < di sé !!

- Quindi  $\min(A) \in A\left[\frac{n}{2}+1 \dots n\right]$ .

## Esercizio 2. Un array ordinato è un min-heap?



- Per rispettare la prop. del min-heap, deve valere

$$A[\text{PARENT}(i)] \leq A[i] \quad \forall i \in [1..n]$$

- Ovviamente,  $\text{PARENT}(i) := \left\lfloor \frac{i}{2} \right\rfloor \leq \left\lfloor \frac{i}{2} \right\rfloor \leq \frac{i}{2} < i \quad \forall i \in \mathbb{N}$

$$\Rightarrow \text{PARENT}(i) < i \quad \forall i \in \mathbb{N}$$

$$\Rightarrow A[\text{PARENT}(i)] \leq A[i] \quad \text{in quanto } A \text{ ordinato.}$$

✓

Esercizio 3. Usando la procedura MAX-HEAPIFY come modello, scrivete uno pseudocodice per la procedura MIN-HEAPIFY( $A, i$ ).

- MIN-HEAPIFY ( $A, i$ ) presuppone che in  $A$  al più l'elemento  $i$  violi la proprietà dell'heap

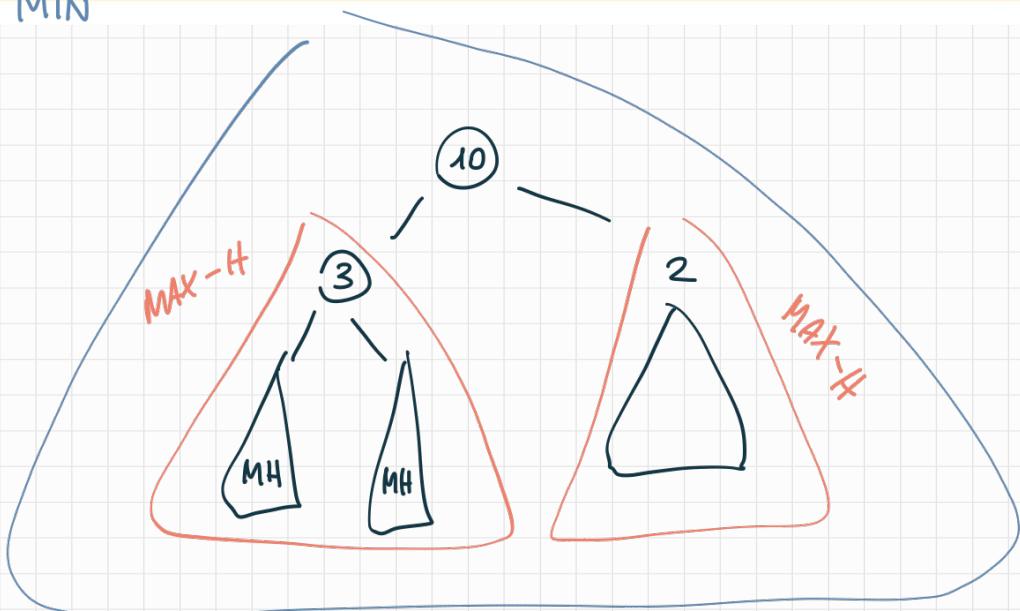
### ~~MIN~~ MAX-HEAPIFY ( $A, i$ )

```

1   $l = \text{LEFT}(i)$ 
2   $r = \text{RIGHT}(i)$ 
3  if  $l \leq A.\text{heap-size}$  and  $A[l] > A[i]$ 
    $\quad \underline{\text{largest}} = l$ 
4  else  $\underline{\text{largest}} = i$ 
5  if  $r \leq A.\text{heap-size}$  and  $A[r] > A[\underline{\text{largest}}]$ 
    $\quad \underline{\text{largest}} = r$ 
8  if  $\underline{\text{largest}} \neq i$ 
9    exchange  $A[i]$  with  $A[\underline{\text{largest}}]$ 
10   MAX-HEAPIFY ( $A, \underline{\text{largest}}$ )

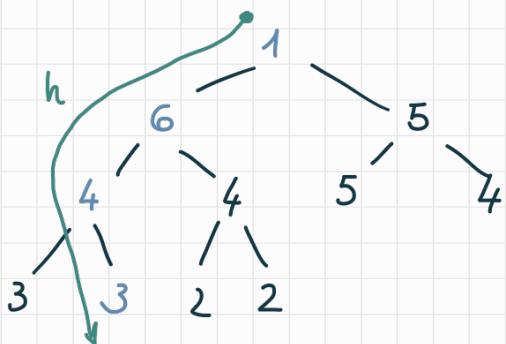
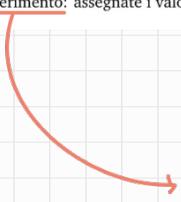
```

~~MIN~~



**Esercizio 4.** Dimostrate che il tempo di esecuzione nel caso peggiore di MAX-HEAPIFY su un heap di dimensione  $n$  è  $\Omega(n \log n)$ .

Suggerimento: assegnate i valori ai nodi in modo che MAX-HEAPIFY sia chiamata ricorsivamente in ogni nodo di un cammino che scende dalla radice fino a una foglia.



- Supponiamo di avere in radice l'elemento minimo di A e che A sia composto da elementi distinti; inoltre i sottoalb.  $\leftarrow$  e  $\rightarrow$  sono Max-Heap validi.
- Abbiamo già osservato che sotto queste ipotesi il minimo di un Max-Heap valido deve trovarsi nelle foglie
- Visto che MAX-HEAPIFY è corretto, sposterà l'elemento 1 fino all'ultimo livello, ossia in una foglia
- Poiché ad ogni chiamata avviene al più uno scambio tra il nodo elaborato e uno dei due figli, vuol dire che faccio h scambi

$h$  chiamate a MAX-HEAPIFY

$\Leftrightarrow \lfloor \log n \rfloor \quad \ll \quad \ll$

$\Leftrightarrow \Theta(\log n) \quad \ll \quad \ll$

- Ogni chiamata costa  $\Theta(1) \Rightarrow$  costo complessivo  $\Theta(\log n)$

**Esercizio 5.** Illustrate il funzionamento di BUILD-MAX-HEAP( $A$ ) sull'array

$$A = \langle 5, 3, 17, 10, 84, 19, 6, 22, 9 \rangle .$$

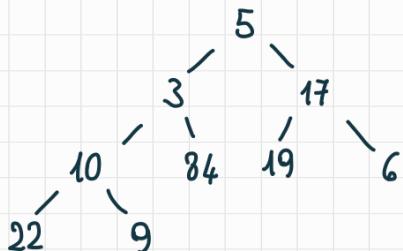
BUILD-MAX-HEAP( $A, n$ )

```

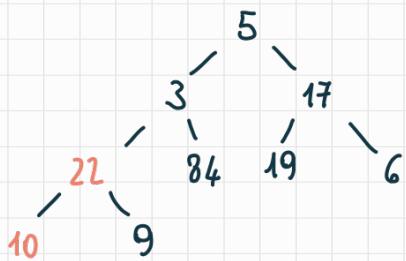
1  A.heap-size = n
2  for i = ⌊n/2⌋ downto 1
3      MAX-HEAPIFY(A, i)

```

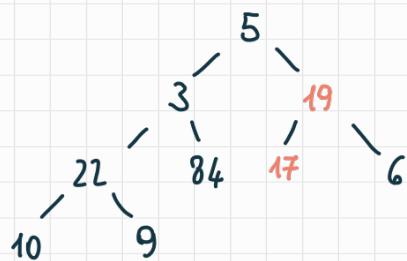
•  $n = 9 \rightarrow A.\text{heap-size} = 9$



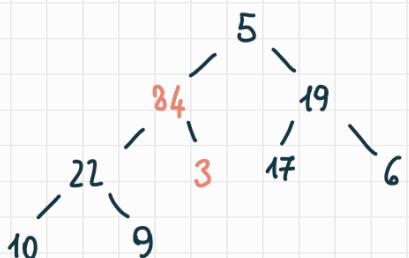
•  $i = \lfloor 9/2 \rfloor = 4$



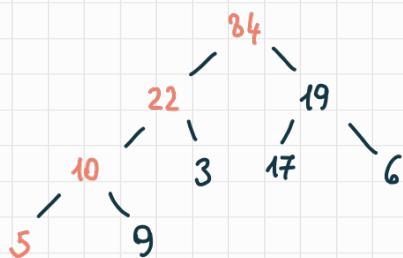
•  $i = 3$



•  $i = 2$



•  $i = 1$



**Esercizio 6.** Supponete che gli oggetti in una coda di max-priorità siano semplicemente delle chiavi.

1. Illustrate il funzionamento di HEAP-EXTRACT-MAX( $A$ ) sull'heap

$$A = \langle 15, 13, 9, 5, 12, 8, 7, 4, 0, 6, 2, 1 \rangle .$$

2. Illustrate il funzionamento di MAX-HEAP-INSERT( $A, 10$ ) sull'heap

$$A = \langle 15, 13, 9, 5, 12, 8, 7, 4, 0, 6, 2, 10 \rangle .$$

MAX-HEAP-MAXIMUM( $A$ )

```

1 if  $A.\text{heap-size} < 1$ 
2   error "heap underflow"
3 return  $A[1]$ 
```

MAX-HEAP-EXTRACT-MAX( $A$ )

```

1  $\text{max} = \text{MAX-HEAP-MAXIMUM}(A)$ 
2  $A[1] = A[A.\text{heap-size}]$ 
3  $A.\text{heap-size} = A.\text{heap-size} - 1$ 
4  $\text{MAX-HEAPIFY}(A, 1)$ 
5 return  $\text{max}$ 
```

MAX-HEAP-INCREASE-KEY( $A, x, k$ )

```

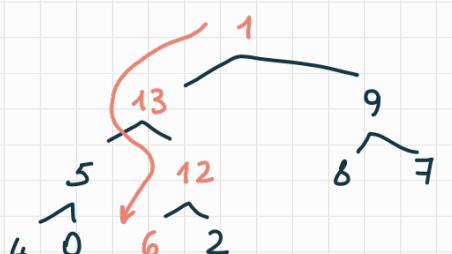
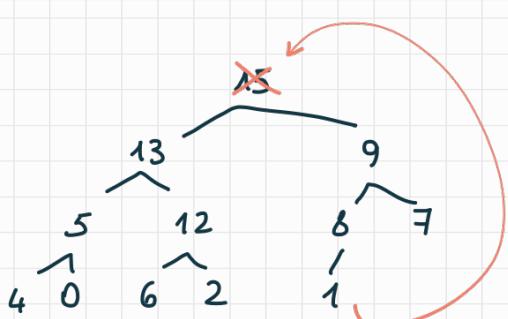
1 if  $k < x.\text{key}$ 
2   error "new key is smaller than current key"
3  $x.\text{key} = k$ 
4 find the index  $i$  in array  $A$  where object  $x$  occurs  $\Theta(1)$ 
5 while  $i > 1$  and  $A[\text{PARENT}(i)].\text{key} < A[i].\text{key}$ 
6   exchange  $A[i]$  with  $A[\text{PARENT}(i)]$ , updating the information that maps
      priority queue objects to array indices
7    $i = \text{PARENT}(i)$ 
```

MAX-HEAP-INSERT( $A, x, n$ )

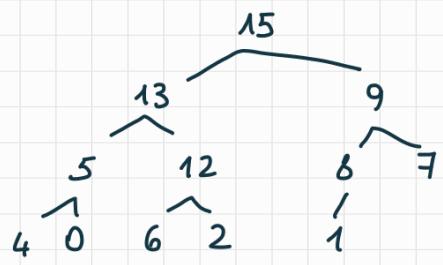
```

1 if  $A.\text{heap-size} == n$ 
2   error "heap overflow"
3  $A.\text{heap-size} = A.\text{heap-size} + 1$ 
4  $k = x.\text{key}$ 
5  $x.\text{key} = -\infty$ 
6  $A[A.\text{heap-size}] = x$ 
7 map  $x$  to index  $\text{heap-size}$  in the array
8  $\text{MAX-HEAP-INCREASE-KEY}(A, x, k)$ 
```

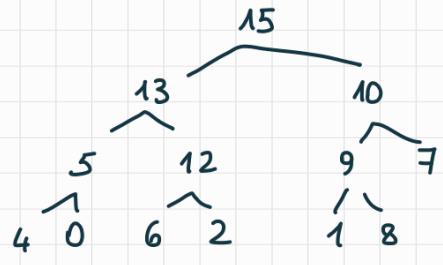
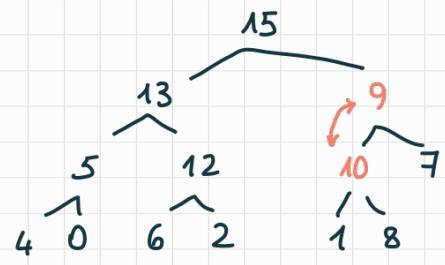
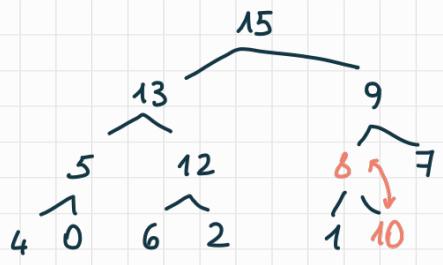
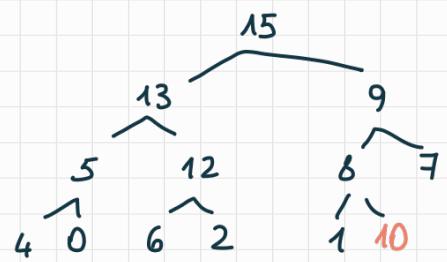
$\Theta(\log n)$



MAX-HEAPIFY( $A, 1$ )



$A[A.\text{heap\_size}] = 10$



**Esercizio 7.** In un max-heap di dimensione  $n$ , è possibile implementare la procedura FIND-MIN che trova un elemento con chiave di valore minimo in  $O(\log n)$ ?

- Non è possibile, in quanto dovrei controllare entrambi i sottoalberi generati da ogni nodo :

$$T(n) \approx 2 T\left(\frac{n}{2}\right) + \Theta(1) = \Theta(n)$$

- Se controllassi solo le foglie  $\rightarrow$  sono  $\Theta\left(\frac{n}{2}\right) \dots$