Famiglia

Trovare i limiti superiori e inferiori più stretti possibili per la seguente famiglia di equazioni di ricorrenza, per valori di a interi positivi.

$$T(n) = \begin{cases} aT(\lfloor n/2 \rfloor) + n^{a-1} & n \geq 2 \\ 1 & n < 2 \end{cases}$$

Inserisci l'albero (03/07/2020)

Sia T un albero binario contenente $n \geq 1$ nodi e A un vettore ordinato contenente n interi distinti.

Oltre ai normali campi di un albero binario, ogni nodo t di T contiene un campo $t.size$ che contiene il numero di nodi del sottoalbero radicato in t (già inizializzato) e un campo $t.value$ (inizialmente vuoto).

Scrivere un algoritmo

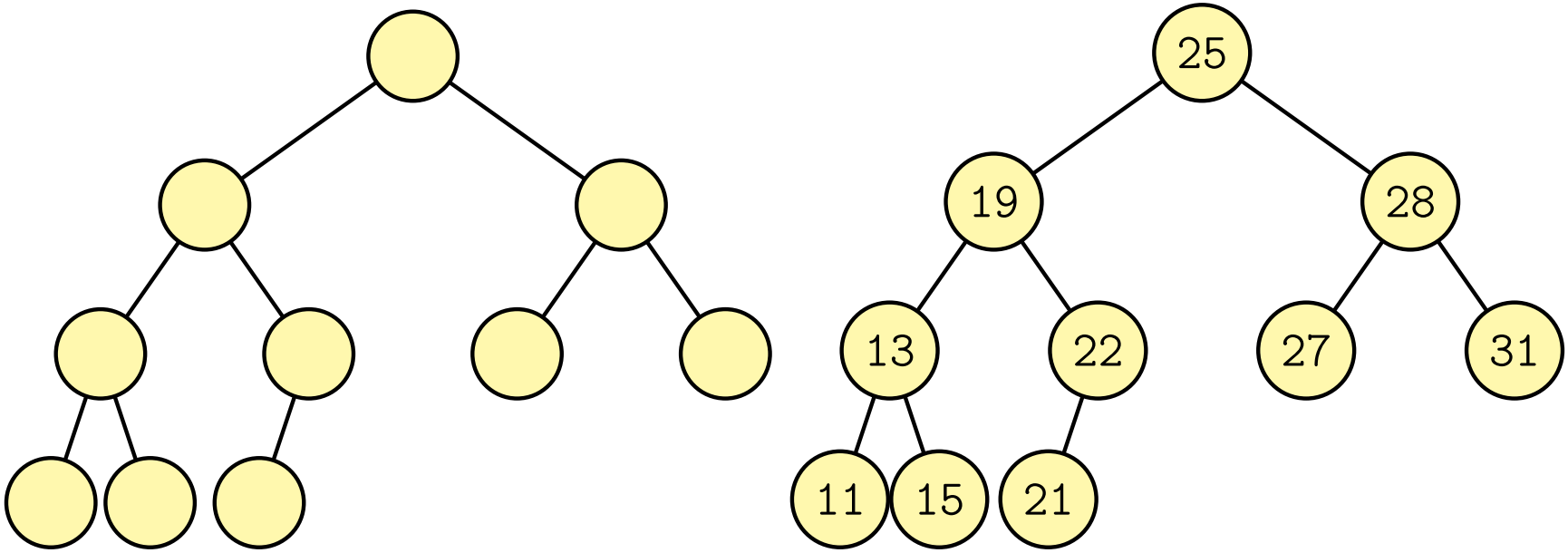
binaryInsert(Tree T, **int**[ ] A, **int** n)

che inserisca tutti i valori di A in T, in modo tale che l'albero risultante sia un albero binario di ricerca.

Discutere correttezza e complessità computazionale dell'algoritmo proposto.

Inserisci l'albero (03/07/2020)

Esempio: a sinistra, un sottoalbero T dato in input, a destra lo stesso sottoalbero dopo che sono stati inseriti i valori
$A = [11, 13, 15, 19, 21, 22, 25, 27, 28, 31]$.



Maggioranza

Scrivere una funzione

boolean hasMajority(**int**[ ] A, **int** n)

che prenda in input un vettore ordinato A contenente n interi e restituisca **true** se A contiene un valore di maggioranza, ovvero un valore che compare più di $n/2$ volte; restituisca **false** altrimenti. Soluzioni di costo computazionale lineare non verranno prese in considerazione.

Discutere correttezza e complessità dell'algoritmo proposto.

Minecraft

In Minecraft, una carta geografica è rappresentata da una matrice $n \times n$ di celle, ognuna delle quali rappresenta un'altitudine intera. Un valore ≤ 0 corrisponde ad un mare, mentre un valore > 0 corrisponde ad un'isola.

Scrivere un algoritmo che prenda in input una matrice A di interi e la sua dimensione positiva n, e restituisca l'altezza media dell'isola più elevata.

Discutere informalmente la correttezza della soluzione proposta e calcolare la complessità computazionale.

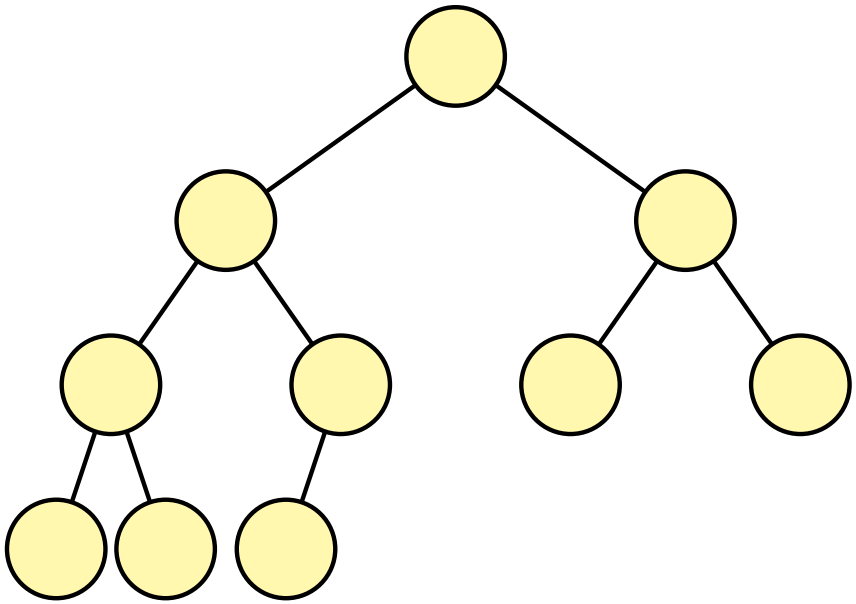
Minecraft

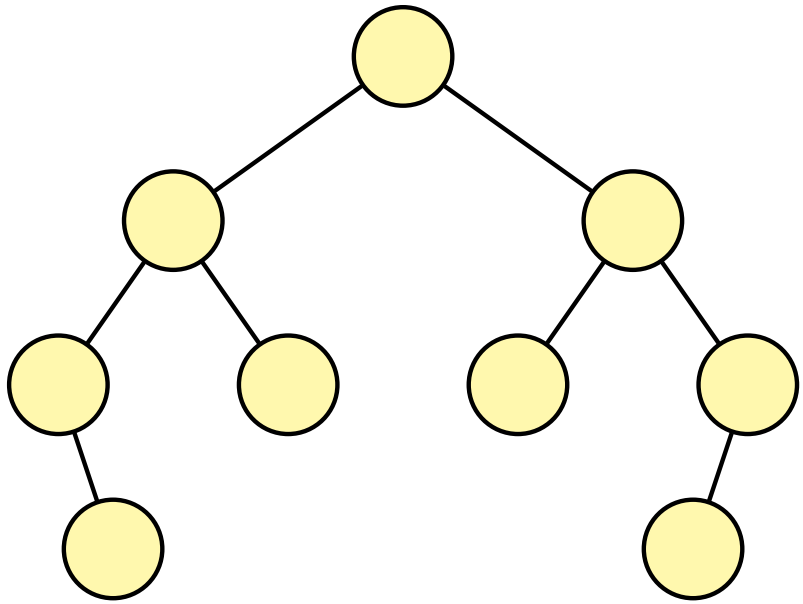
Ad esempio, nella matrice seguente ci sono tre isole, una 2×3 in alto a sinistra con altezza media 1.5, una 4×1 a destra con altezza media 1, una 1×2 in basso a sinistra con altezza media 2. L'isola con altezza media più alta è quella in basso a sinistra, e quindi l'algoritmo deve restituire 2.

1	1	1	0	1
2	2	2	-1	1
0	-1	0	-2	1
0	0	-1	-2	1
2	2	-1	0	0

Alberi simmetrici

Scrivere un algoritmo **boolean** isSymmetric(TREE T) che prenda in input un albero binario T non vuoto e restituisca **true** se T è simmetrico, **false** altrimenti. Un albero binario è simmetrico se il sottoalbero sinistro della radice è un'immagine *speculare* del sottoalbero destro della radice. L'albero binario a sinistra non è simmetrico, mentre lo è quello di destra.





Discutere correttezza e complessità computazionale dell'algoritmo