Algoritmi e Strutture Dati

a.a. 2024/25

Prima prova intermedia del 16/01/2025

(ogni esercizio vale 10 punti)

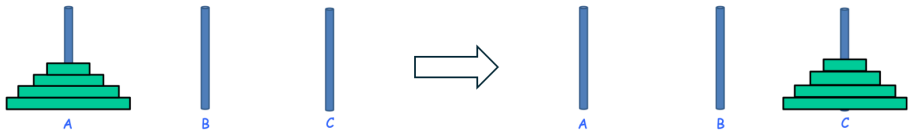
Cognome: _____ Nome: _____

Matricola: _____ E-mail: _____

1. Dato un albero binario i cui nodi *x* hanno i campi **left**, **right** e **key**, dove **key** è un numero intero:
   a. definire l'**altezza** di un nodo *x*;
   b. scrivere una funzione **efficiente** in C o C++ che ritorna il numero di nodi per i quali la chiave x->key è minore o uguale dell'altezza del nodo. Il prototipo della funzione è:
   `int lessHeight(PNode r)`
   c. valutare la complessità della funzione, **indicando eventuali relazioni di ricorrenza e mostrando la loro risoluzione**;
   d. specificare il linguaggio di programmazione scelto.

2. Per ordinare l'array A[1..n], si ordina in modo ricorsivo il sottoarray A[1.. n-1] e poi si inserisce A[n] nel sottoarray ordinato A[1.. n-1].
   a. Scrivere lo pseudocodice per questa variante **ricorsiva** dell'insertion sort.
   b. Fornire una ricorrenza per il suo tempo di esecuzione nel caso peggiore e risolverla in modo formale.
   c. Quale è il tempo di esecuzione nel caso migliore? Mostrare un esempio di input che determina il caso migliore.

3. Il problema della *torre di Hanoi* è un classico rompicapo matematico nel quale sono dati tre paletti e un certo numero di dischi di grandezza diversa. Il gioco inizia ponendo i dischi su un paletto in ordine decrescente, in modo da formare un cono. Lo scopo del gioco è portare tutti i dischi su un paletto diverso potendo spostare solo un disco alla volta e potendo mettere un disco solo su un altro più grande, mai su uno più piccolo. Nella configurazione riportata in figura il numero di dischi è 4, il paletto di partenza è A e il paletto di arrivo è C. Il paletto B viene utilizzato come paletto di "appoggio".



Il seguente algoritmo rappresenta una soluzione ricorsiva al problema:

```
procedure hanoi (n: integer; partenza, appoggio, arrivo:  palo);
begin
if (n = 1) then muoviDisco(partenza, arrivo)
else begin
   hanoi(n-1, partenza, arrivo, appoggio);
   muoviDisco(partenza, arrivo);
   hanoi(n-1, appoggio, partenza, arrivo);
end
```

dove `muoviDisco(partenza,arrivo)` è una semplice procedura che sposta il disco situato in cima al palo di partenza su quello di arrivo e ha complessità costante.

Qual è la complessità dell'algoritmo? Giustificare formalmente la risposta.