

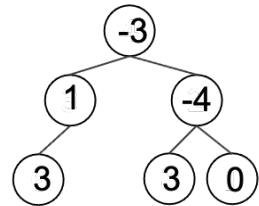
Esercizio 1

Scrivere una funzione **esercizio1** che prenda in input un albero binario A , i cui valori informativi sono dei numeri interi, e restituisca `true` se per ogni nodo v dell'albero, il suo valore informativo è minore o uguale al livello in cui si trova il nodo. Se la condizione non è verificata, la funzione deve restituire `false`.

L'albero è rappresentato da una classe `AlberoB`. Sia a una istanza della classe, l'interfaccia è la seguente:

- `a.figlio(SIN/DES)` restituisce il figlio sinistro o destro dell'albero a se esiste, altrimenti restituisce la costante `NULLO`;
- `a.padre()` restituisce il padre dell'albero a se esiste, altrimenti restituisce la costante `NULLO`;
- `a.radice()` restituisce il valore informativo associato all'albero a (il valore intero contenuto all'interno del nodo);
- `a.foglia()` restituisce `true` se l'albero a è una foglia, altrimenti `false`.
- `a.nullo()` restituisce `true` se l'albero a è nullo, altrimenti `false`.

Si assume che il primo livello sia il livello 1 (il livello della radice dell'albero A).

<i>Esempio:</i> in questo caso, la funzione dovrebbe restituire <code>true</code> poiché le condizioni di cui sopra valgono per ogni nodo dell'albero. Se, ad esempio, il nodo radice avesse valore informativo 3, allora la funzione dovrebbe restituire <code>false</code> , poiché esiste almeno un nodo (il nodo radice dell'albero) che ha un valore informativo maggiore del livello in cui si trova.	
---	--

Esercizio 2

Scrivere una funzione **esercizio2** che prenda in input un vettore V di elementi e un insieme A di coppie (x, y) dove x e y sono elementi di V , e un intero positivo k . La funzione dovrebbe restituire `true` se è possibile assegnare ad ogni coppia di A un numero da 1 a k in modo tale che non esistano due coppie in A con un elemento in comune che abbiano assegnato lo stesso numero.

Se non è possibile creare una associazione che renda vera questa condizione, la funzione dovrebbe restituire `false`.

Si può assumere che:

- V sia rappresentato da un `vector<string>` (`ArrayList<String>` se si usa Java)
- A sia rappresentato da un `vector<Coppia>` (`ArrayList<Coppia>` se si usa Java), dove `Coppia` è una classe con due campi pubblici `x` e `y` rappresentati rispettivamente gli elementi (stringhe) della coppia.

Esempio utilizzo Coppia: sia c una istanza di `Coppia` con gli elementi (a, b) , allora `c.x == "a"`, `c.y == "b"`.

<i>Esempio:</i> in questo caso, la funzione dovrebbe restituire <code>true</code> poiché è possibile assegnare ad ogni coppia di A un valore da 1 a k in modo che nessuna coppia che abbia un elemento in comune abbia lo stesso numero assegnato. Ad esempio: <ul style="list-style-type: none">• Assegno 1 alla coppia (a, e)• Assegno 2 alla coppia (a, d)• Assegno 3 alla coppia (a, b)• Assegno 3 alla coppia (e, c)• Assegno 1 alla coppia (c, d)• Assegno 2 alla coppia (b, e)	$V = [a, b, c, d, e]$ $A = [(a, e), (a, d), (a, b), (e, c), (c, d), (b, e)]$ $k = 3$
--	--