## Haskell — Arbre binari P37072_ca

Aquest problema planteja l'escriptura de diverses funcions sobre arbres binaris genèrics. La definició dels arbres ve donada per:

**data** *Tree a* = *Node a* (*Tree a*) (*Tree a*) | *Empty* **deriving** (**Show**)

És a dir, un arbre amb elements de tipus *a* és, o bé un arbre buit, o bé un node que arrela un element (de tipus *a*) amb dos altres arbres. La declaració **deriving** (**Show**) permet mostrar els arbres senzillament.

1. Feu una funció *size* :: *Tree a* → **Int** que, donat un arbre, retorni la seva talla, és a dir, el nombre de nodes que conté.
2. Feu una funció *height* :: *Tree a* → **Int** que, donat un arbre, retorni la seva alçada, assumint que els arbres buits tenen alçada zero.
3. Feu una funció *equal* :: **Eq** *a* ⇒ *Tree a* → *Tree a* → **Bool** que, donat dos arbres, indiqui si són el mateix.
4. Feu una funció *isomorphic* :: **Eq** *a* ⇒ *Tree a* → *Tree a* → **Bool** que, donat un arbres, indiqui si són el isomorfs, és a dir, si es pot obtenir l'un de l'altre tot girant algun dels seus fills.
5. Feu una funció *preOrder* :: *Tree a* → [*a*] que, donat un arbre, retorni el seu recorregut en pre-ordre.
6. Feu una funció *postOrder* :: *Tree a* → [*a*] que, donat un arbre, retorni el seu recorregut en post-ordre.
7. Feu una funció *inOrder* :: *Tree a* → [*a*] que, donat un arbre, retorni el seu recorregut en in-ordre.
8. Feu una funció *breadthFirst* :: *Tree a* → [*a*] que, donat un arbre, retorni el seu recorregut per nivells.
9. Feu una funció *build* :: **Eq** *a* ⇒ [*a*] → [*a*] → *Tree a* que, donat el recorregut en pre-ordre d'un arbre i el recorregut en in-ordre del mateix arbre, retorni l'arbre original. Assumiu que l'arbre no té elements repetits.
10. Feu una funció *overlap* :: (*a* → *a* → *a*) → *Tree a* → *Tree a* → *Tree a* que, donats dos arbres, retorni la seva superposició utilitzant una funció. Superposar dos arbres amb una funció consisteix en posar els dos arbres l'un damunt de l'altre i combinar els nodes doble resultants amb la funció donada o deixant els nodes simples tal qual.

### Puntuació

Cada apartat puntua 10 punts.

## Exemple d'entrada 1

```
let t7 = Node 7 Empty Empty
let t6 = Node 6 Empty Empty
let t5 = Node 5 Empty Empty
let t4 = Node 4 Empty Empty
let t3 = Node 3 t6 t7
let t2 = Node 2 t4 t5
let t1 = Node 1 t2 t3
let t1' = Node 1 t3 t2
size t1
height t1
equal t2 t3
isomorphic t1 t1'
preOrder t1
postOrder t1
inOrder t1
breadthFirst t1
build [1,2,4,5,3] [4,2,5,1,3]
overlap (+) t2 t3
overlap (+) t1 t3
```

## Exemple de sortida 1

```
7
3
False
True
[1,2,4,5,3,6,7]
[4,5,2,6,7,3,1]
[4,2,5,1,6,3,7]
[1,2,3,4,5,6,7]
Node 1 (Node 2 (Node 4 Empty Empty) (Node 5 Empty Empty)) (Node 3 Empty Empty)
Node 5 (Node 10 Empty Empty) (Node 12 Empty Empty)
Node 4 (Node 8 (Node 4 Empty Empty) (Node 5 Empty Empty)) (Node 10 (Node 6 Empty Empty) (Node 7 
```

## Informació del problema

Autoria: Jordi Petit

Generació: 2026-02-03T17:09:45.093Z