
Haskell — Arbres binaris de cerca**P87706_ca**

Es vol tenir un mòdul per a arbres binaris de cerca (binary search trees, BSTs) estàndards. Per això, es defineix el tipus següent:

data *BST a* = *E* | *N a (BST a) (BST a)* **deriving** (**Show**)

Sobre aquest tipus cal crear les operacions següents:

- *insert* :: **Ord** *a* ⇒ *BST a* → *a* → *BST a*
Retorna el resultat d'inserir un element en un arbre binari de cerca. Si l'element ja hi era, el resultat és l'arbre original.
- *create* :: **Ord** *a* ⇒ [*a*] → *BST a*
Retorna un arbre binari de cerca inserint l'un rera l'altre la llista d'elements donats.
- *remove* :: **Ord** *a* ⇒ *BST a* → *a* → *BST a*
Retorna el resultat d'esborrar un element d'un arbre binari de cerca. Si l'element no hi era, el resultat és l'arbre original.
(Nota: hi ha moltes maneres d'implementar l'esborrat; no importa quina trieu mentre sigui prou ràpida.)
- *contains* :: **Ord** *a* ⇒ *BST a* → *a* → **Bool**
Indica si un element es troba o no en un arbre binari de cerca.
- *getmax* :: *BST a* → *a*
Retorna l'element més gran d'un arbre binari de cerca no buit.
- *getmin* :: *BST a* → *a*
Retorna l'element més petit d'un arbre binari de cerca no buit.
- *size* :: *BST a* → **Int**
Retorna el nombre d'elements en un arbre binari de cerca.
- *elements* :: *BST a* → [*a*]
Retorna els elements d'un arbre binari de cerca en ordre.

Puntuació

- | | |
|---|-----------------|
| • test-1: Funció <i>insert</i> . | 10 Punts |
| • test-2: Funció <i>create</i> i les anteriors. | 10 Punts |
| • test-4: Funcions <i>getmin</i> i <i>getmax</i> i les anteriors. | 10 Punts |
| • test-3: Funció <i>contains</i> i les anteriors. | 10 Punts |
| • test-5: Funcions <i>elements</i> i <i>size</i> i les anteriors. | 10 Punts |
| • test-6: Funció <i>remove</i> i les anteriors. | 10 Punts |
| • test-7: Tests d'eficiència, genericitat i integritat amb totes les funcions. | 40 Punts |

Exemple d'entrada

```
let t = create [3,4,1,2]
t
size t
getmin t
getmax t
elements t
map (contains t) [0..5]
insert t 0
elements $ remove t 3
```

Exemple de sortida

```
N 3 (N 1 E (N 2 E E)) (N 4 E E)
4
1
4
[1,2,3,4]
[False,True,True,True,True,False]
N 3 (N 1 (N 0 E E) (N 2 E E)) (N 4 E E)
[1,2,4]
```

Informació del problema

Autor : Jordi Petit

Generació : 2017-04-20 16:55:08

© *Jutge.org*, 2006–2017.

<http://jutge.org>