BinTree (Ejemplos)

A continuación se muestra una ejemplo de uso del módulo BinTree en una sesión con el compilador interactivo ocaml

```
~ % ocaml
OCaml version 4.14.0
Enter #help;; for help.

# #load "binTree.cmo";;
# open BinTree;;
# let comb x l r =
      left_replacement (right_replacement (leaftree x) r) l;;
val comb : 'a -> 'a BinTree.t -> 'a BinTree.t -> 'a BinTree.t = <fun>
# let leaf x = leaftree x;;
val leaf : 'a -> 'a BinTree.t = <fun>
# let complete_tree n =
      let rec aux i =
          if i > n then empty
          else comb i (aux (2*i)) (aux (2*i+1))
   in aux 1;;
val complete_tree : int -> int BinTree.t = <fun>
# let t8 = complete_tree 8;;
val t8 : int BinTree.t = <abstr>
# size t8, height t8;;
- : int * int = (8, 4)
# breadth t8;;
- : int list = [1; 2; 3; 4; 5; 6; 7; 8]
# let t8' = mirror t8;;
val t8' : int BinTree.t = <abstr>
# breadth t8';;
- : int list = [1; 3; 2; 7; 6; 5; 4; 8]
# preorder t8;;
- : int list = [1; 2; 4; 8; 5; 3; 6; 7]
# inorder t8;;
- : int list = [8; 4; 2; 5; 1; 6; 3; 7]
# postorder t8;;
- : int list = [8; 4; 5; 2; 6; 7; 3; 1]
# find_in_depth ((<) 3) t8;;
- : int = 4
# find_in_depth ((<) 3) t8';;
- : int = 7
# exists ((<) 7) t8;;
- : bool = true
# exists ((<) 8) t8;;
- : bool = false
# for_all ((<) 7) t8;;
```

```
- : bool = false
# for_all ((<) 0) t8;;
- : bool = true
# leaves t8;;
- : int list = [8; 5; 6; 7]
# leaves (right_b t8);;
- : int list = [6; 7]
# leaves (right_b t8');;
- : int list = [5; 8]
# let t8b = map (fun n -> n mod 2 = 0) t8;;
val t8b : bool BinTree.t = <abstr>
# breadth t8b;;
- : bool list = [false; true; false; true; false; true; false; true]
# let t8c = map (fun n -> if n mod 2 = 0 then 2 * n else n) t8;;
val t8c : int BinTree.t = <abstr>
# breadth t8c;;
- : int list = [1; 4; 3; 8; 5; 12; 7; 16]
# let tr = left_b (left_b t8);;
val tr : int BinTree.t = <abstr>
# breadth tr;;
- : int list = [4; 8]
# let t = replace_when ((<) 1) t8 tr;;
val t : int BinTree.t = <abstr>
# breadth t;;
- : int list = [1; 4; 4; 8; 8]
# let ta = cut_below ((<=) 3) t8;;
val ta : int BinTree.t = <abstr>
# breadth ta;;
- : int list = [1; 2; 3; 4; 5]
# let tb = cut_above ((<=) 3) t8;;
val tb : int BinTree.t = <abstr>
# breadth tb;;
- : int list = [1; 2]
# #quit;;
~ %
```

Aunque es interesante que le eche un ojo a toda las frases de este ejemplo, si simplemente quiere asegurarse de que con su módulo BinTree genera la misma salida, puede probar lo siguiente

```
~ % ocaml -no-version -noprompt binTree.cmo < ex1.ml > out
~ % diff out ex1.out
~ %
```