StBinTree (Ejemplos)

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

```
~ % ledit ocaml
OCaml version 5.2.0
Enter #help;; for help.
# #load_rec "stBinTree.cmo";;
# open StBinTree;;
# let leaf = leaftree;;
val leaf : 'a -> 'a StBinTree.t = <fun>
# let complete_tree n = (* da el mismo valor para (2*i) y (2*i+1) *)
      let rec aux i =
          if 2*i > n then leaf i
          else comb i (aux (2*i)) (aux (2*i+1))
   in aux 1  ;;
val complete_tree : int -> int StBinTree.t = <fun>
# let t9 = complete_tree 9;;
val t9 : int StBinTree.t =
  SBT (SBT (SBT (Leaf 8, 4, Leaf 9), 2, Leaf 5), 1, SBT (Leaf 6, 3, Leaf 7))
# size t9, height t9;;
- : int * int = (9, 4)
# breadth t9;;
- : int list = [1; 2; 3; 4; 5; 6; 7; 8; 9]
# let t9' = mirror t9;;
val t9' : int StBinTree.t =
  SBT (SBT (Leaf 7, 3, Leaf 6), 1, SBT (Leaf 5, 2, SBT (Leaf 9, 4, Leaf 8)))
# breadth t9';;
- : int list = [1; 3; 2; 7; 6; 5; 4; 9; 8]
# preorder t9;;
- : int list = [1; 2; 4; 8; 9; 5; 3; 6; 7]
# inorder t9;;
- : int list = [8; 4; 9; 2; 5; 1; 6; 3; 7]
# postorder t9;;
- : int list = [8; 9; 4; 5; 2; 6; 7; 3; 1]
# find_in_depth ((<) 3) t9;;
- : int = 4
# find_in_depth ((<) 3) t9';;
- : int = 7
# exists ((<) 7) t9;;
- : bool = true
# exists ((<) 8) t9;;
- : bool = true
# for_all ((<) 7) t9;;
- : bool = false
# for_all ((<) 0) t9;;
- : bool = true
# leaves t9;;
- : int list = [8; 9; 5; 6; 7]
# leaves (right_b t9);;
```

```
- : int list = [6; 7]
# leaves (right_b t9');;
- : int list = [5; 9; 8]
# let t9b = map (fun n -> n mod 2 = 0) t9;;
val t9b : bool StBinTree.t =
  SBT (SBT (SBT (Leaf true, true, Leaf false), true, Leaf false), false,
   SBT (Leaf true, false, Leaf false))
# breadth t9b;;
- : bool list = [false; true; false; true; false; true; false; true; false]
# let t9c = map (fun n -> if n mod 2 = 0 then 2 * n else n) t9;;
val t9c : int StBinTree.t =
  SBT (SBT (SBT (Leaf 16, 8, Leaf 9), 4, Leaf 5), 1,
   SBT (Leaf 12, 3, Leaf 7))
# breadth t9c;;
- : int list = [1; 4; 3; 8; 5; 12; 7; 16; 9]
# let tr = left_b (left_b t9);;
val tr : int StBinTree.t = SBT (Leaf 8, 4, Leaf 9)
# breadth tr;;
- : int list = [4; 8; 9]
# let t = replace_when ((<) 2) t9 tr;;
val t : int StBinTree.t =
  SBT (SBT (SBT (Leaf 8, 4, Leaf 9), 2, SBT (Leaf 8, 4, Leaf 9)), 1,
   SBT (Leaf 8, 4, Leaf 9))
# breadth t;;
- : int list = [1; 2; 4; 4; 4; 8; 9; 8; 9; 8; 9]
# let ta = cut_below ((<=) 3) t9;;
val ta : int StBinTree.t = SBT (SBT (Leaf 4, 2, Leaf 5), 1, Leaf 3)
# breadth ta;;
- : int list = [1; 2; 3; 4; 5]
# from_bin (to_bin t9) = t9;;
- : bool = true
# #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 ***StBinTree*** genera la misma salida, puede probar lo siguiente

```
~ % ocaml -no-version -noprompt binTree.cmo stBinTree.cmo < ex2.ml > out
~ % diff out ex2.out
~ %
```