Travaux pratiques

option informatique

Exercices de révision (2)

Arbres binaires

Pour représenter un arbre binaire étiqueté par le type *'a*, on définit le type :

```
type 'a btree = Nil | Node of 'a * 'a btree * 'a btree ;;
```

Exercice 1.
Rédiger les fonctions suivantes :

- **size**, de type *'a btree -> int*, retourne le nombre de nœuds d'un arbre ;
- **leaf**, de type *'a btree -> int*, retourne le nombre de feuilles d'un arbre, c'est-à-dire le nombre de nœuds dont les deux fils sont égaux à **Nil** ;
- **member**, de type *'a -> 'a btree -> bool*, détermine si l'un des nœuds de l'arbre possède une étiquette donnée ;
- **height**, de type *'a btree -> int*, calcule la hauteur d'un arbre (on convient que la hauteur de **Nil** est -1).

Exercice 2.
Rédiger une fonction **tag** qui retourne la liste des étiquettes portées par les nœuds d'un arbre parcouru par ordre préfixe (respectivement infixe, suffixe).

```
tag : 'a btree -> 'a list
```

Exercice 3. (fonctionnelles sur les arbres)
Rédiger une fonction **map_tree** équivalente à la fonction **map**, mais s'appliquant à des arbres binaires.

```
map_tree : ('a -> 'b) -> 'a btree -> 'b btree
```

Rédiger de même une fonction **fold_tree** équivalente à **list_it** pour les arbres binaires.

```
fold_tree : ('a -> 'b -> 'b -> 'b) -> 'a btree -> 'b -> 'b
```

Utiliser la fonction **fold_tree** pour redéfinir les fonctions **size**, **height**, **tag**.

Exercice 4.
Un arbre binaire est dit *symétrique* lorsqu'il est égal à **Nil** ou lorsque son fils gauche est l'image miroir de son fils droit.
Rédiger une fonction **symmetric** qui détermine si un arbre binaire est symétrique.

```
symmetric : 'a btree -> bool
```

Arbres n-aires

Pour représenter un arbre dans lequel chaque nœud peut avoir un nombre arbitraire de fils, on définit le type :

```
type 'a ntree = Nil | Node of 'a * ('a ntree list) ;;
```

Exercice 5.
Rédiger les fonctions suivantes :

- **size**, de type *'a ntree -> int*, retourne le nombre de nœuds d'un arbre ;
- **member**, de type *'a -> 'a ntree -> bool*, détermine si l'un des nœuds de l'arbre possède une étiquette donnée ;
- **height**, de type *'a ntree -> int*, calcule la hauteur d'un arbre ;
- **sum**, de type *int ntree -> int*, calcule la somme des valeurs des étiquettes de cet arbre.

Il pourra être judicieux d'utiliser les fonctionnelles sur les listes pour répondre à ces questions.