Programmation fonctionnelle

Répétition 6

29 mars 2007

Correction des exercices proposés

1.* Ecrire une fonction `breadth-first` qui prend comme argument un arbre n-aire dont uniquement les feuilles sont étiquetées, et renvoie la liste des étiquettes des feuilles, obtenue par un parcours en largeur d'abord, et de gauche à droite, de l'arbre.

2.$^{\star\clubsuit}$ Un entier naturel est *parfait* s'il est égal à la somme de ses diviseurs positifs propres. Ecrire un prédicat `perfect?` déterminant si son argument est un nombre parfait.

Exemple: Le nombre 28 est parfait parce que $28 = 1 + 2 + 4 + 7 + 14$; le nombre 32 n'est pas parfait parce que $32 \neq 1 + 2 + 4 + 8 + 16$.

Abstraction sur les données

3. On décide de représenter les nombres complexes par des paires pointées dont le `car` est la partie réelle et le `cdr` la partie imaginaire. Ecrire les fonctions

```
(make-from-real-imag re im)
(make-from-ang-imag rho ang)
(add-complex z1 z2)
(sub-complex z1 z2)
(mul-complex z1 z2)
(div-complex z1 z2)
(real-part z)
(imag-part z)
(magnitude z)
(angle z)
```

Exercices d'interrogation sur les arbres

4.$^{\clubsuit}$ Définition d'une structure de données

Un *N-arbre* est un arbre dont

– chaque nœud est étiqueté par un nombre naturel, sa *clef* ;
– chaque nœud a 0 ou 1 fils gauche ;
– chaque nœud a 0 ou 1 fils droit.

Un N-arbre est représenté par une liste de trois éléments. Le premier est la clef de la racine, le deuxième est la représentation du fils gauche s'il existe et la liste vide sinon. De même, le troisième élément est la représentation du fils droit s'il existe et la liste vide sinon. Un N-arbre est *conditionné* si la clef de tout nœud interne est plus grande ou égale aux clefs de tous ses descendants de gauche, et plus petite ou égale aux clefs de tous ses descendants de droite.

1. Ecrire un prédicat `ntree?` à un argument, qui renvoie `#t` si son argument est un N-arbre (en fait, sa représentation) et `#f` sinon.
   `(ntree? '(5 (4 () (7 () ())) (3 () ())))` ⇒ `#t`
2. Ecrire un prédicat `condit?` prenant comme argument un N-arbre et renvoyant `#t` si ce N-arbre est conditionné et `#f` sinon.
   `(condit? '(5 (4 () (7 () ())) (3 () ())))` ⇒ `#f`
3. Ecrire un prédicat `present?` prenant comme arguments un N-arbre conditionné et un nombre naturel et renvoyant `#t` si le nombre est la clef d'un nœud de l'arbre, et `#f` sinon.

5.♣ Un N-arbre est un arbre dont chaque nœud admet un nombre quelconque de fils ; seules les feuilles sont étiquetées, l'étiquette étant un nombre naturel. La fonction f prend comme argument un N-arbre a et renvoie le N-arbre $f(a)$ obtenu en remplaçant chaque feuille de a étiquetée $n > 0$ par un nœud interne dont les n fils sont des feuilles étiquetées $n-1$. On demande de programmer la fonction f.