INFO0054 - Programmation fonctionnelle

Répétition 3: Récursion sur les nombres et sur les listes

Jean-Michel Begon

02 Mars 2017

Mise en commun

Exercice 1.
Version *tail-recursive* de `reverse` et `length` ainsi que leurs spécifications.

Récursion sur les nombres

Exercice 2.
Les nombres de Gribomont sont définis comme suit :

si $n < 3$, $f(n) = n$
si $n \geq 3$, $f(n) = f(n-1) + f(n-2) + f(n-3)$.

Écrire une fonction qui permet de calculer les nombres de Gribomont.

Récursion sur les listes

Exercice 3.
Ecrire une fonction `concat` qui prend deux listes en arguments et qui renvoie la concaténation, dans l'ordre, de celle-ci.

Exercice 4.
Écrire une fonction `big` qui prend comme arguments un nombre entier `n` et une liste `l` de nombres entiers, telle que `(big n l)` est la liste des éléments de `l` plus grands que `n`.

```
(big 5 '(7 -3 6 1 -2 5 6)) ⇒ (7 6 6)
```

Exercice 5.
Écrire une fonction `remove-all` à deux arguments `l` et `n` dont les valeurs sont respectivement une liste et un nombre entier, qui retourne la liste privée de toutes les occurrences de `n`.

```
(remove-all '(2 7 1 7 3 1) 1) ⇒ (2 7 7 3)
```

Exercice 6.
Que faudrait-il changer dans la fonction `remove-all` pour qu'elle renvoie la liste privée de la première occurrence de `n` ?

Spécification

Exercice 7.
Corriger la spécification suivante

> Si `n` est un naturel, `x` est un naturel quelconque et `acc` est la liste vide, alors `(ml n x acc)` renvoie une liste de taille `n` dont chaque élément est `x`.

se rapportant à la fonction

```
(define ml
    (lambda (n x acc)
        (if (zero? n) acc
            (ml (- n 1) x (cons x acc)))))
```

Exercice 8.
Ecrire une spécification pour la fonction suivante :

```
(define xyz
  (lambda (x y z)
    (cond ((null? y) z)
          ((= x (car y)) (xyz x (cdr y) (+ 1 z)))
          (else (xyz x (cdr y) z)))))
```

Récursion sur les listes — le retour

Exercice 9.
Définir la fonction `filter_` à deux arguments, un prédicat unaire `p?` et une liste d'éléments appartenant à son domaine `ls`, et renvoyant la liste des éléments de `ls` pour lesquels l'application du prédicat est `#t`.
Remarque : la fonction `filter` est prédéfinie, nous la redéfinissons sous un autre nom à titre d'exercice.

Exercice 10.
Redéfinir les fonctions `big` et `remove-all` à l'aide de la fonction `filter`.