S02 : Haskell (Listes et récursivité)

Enseignant : Stéphane LE PEUTREC
Assistant : Jonathan LAUPER

Instructions

- Deadline : jeudi suivant à 11:00

Exercice 1

Développez les fonctions suivantes.

- f1 l : retourne les trois premiers éléments de la liste l. La liste est entièrement retournée en résultat si elle contient moins de trois éléments. Créer deux version de cette fonction : une version avec motif et une version sans motif.
  exemple : f1 [4,8,2,7,9,3] retourne [4,8,2]

- fibonacci n : retourne la valeur de fibonacci(n)

  Définition de la suite de Fibonacci :
  f(0) = 0
  f(1) = 1
  f(n) = f(n-1) + f(n-2) // pour n≥2

Exercice 2

Développez les fonctions récursives sur les listes qui suivent (sans utiliser les fonctions prédéfinies équivalentes).

- last' l : retourne le dernier élément de la liste l. La liste ne doit pas être vide.
  exemple : last' [4,8,2] retourne 2

- delete' e l : retourne la liste l sans sa première occurrence de e
  exemple : delete' 5 [1,5,7,5,8] retourne [1,7,5,8]

- maximum' l : retourne le plus grand élément de la liste l
  exemple : maximum' [1,5,7,5] retourne 7

- scalarProduct l1 l2 : retourne le produit scalaire des listes l1 et l2.

  Rappel : le produit scalaire des listes [v1,v2,v3] et [u1,u2,u3] est égal à v1*u1 + v2*u2 + v3*u3
  Note : si les deux listes ne sont pas de la même taille on ignore les éléments restant de la liste la plus grande
  exemple : scalarProduct [1,5,7] [2,3,1,7,4] retourne 24 (ie: 1*2 + 5*3 + 7*1)

Exercice 3

Développez les fonctions suivantes (version récursive).

- length' l : retourne la longueur de la liste l (son nombre d'élément).
  exemple : length' [1,5,7] retourne 3
- deleteAll' e l : retourne la liste l sans toutes ses occurrences de e
  exemple : deleteAll' 5 [1,5,7,5,8] retourne [1,7,8]
- toUpperString ch : retourne la chaîne de caractères ch en majuscule
  exemple : toUpperString "hello" retourne "HELLO"

  note :
  vous pouvez utiliser la fonction prédéfinie `toUpper c` qui retourne le caractère c en majuscule. Cette fonction est déclarée dans le module Data.Char.
  Ajoutez `import Data.Char` au début de votre fichier source pour pouvoir utilisez les fonctions de ce module.

Exercice 4

Développez les fonctions suivantes (version récursive).

- countVowel ch : retourne le nombre de voyelles présents dans la chaîne de caractères ch
  exemple : countVowel "hello" retourne 2

- analyseStrings lch : lch est une liste de chaînes de caractères. Cette fonction retourne une liste de tuples formés des chaînes de caractères de lch et de leur taille.
  exemple : analyseStrings ["hello","him","char"] retourne la liste [("hello",5),("him",3),("char",4)]

- analyseStrings2 lch : lch est une liste de chaînes de caractères. Cette fonction retourne une liste de tuples formés des chaînes de caractères de lch, de leur taille et de leur nombre de voyelles.
  exemple :
  analyseStrings2 ["hello","him","char"] retourne la liste [("hello",5,2),("him",3,1),("char",4,1)]