S04 : Haskell (fonctions d'ordre supérieur)

Enseignant : Stéphane LE PEUTREC
Assistant : Jonathan LAUPER

Instructions

- Deadline : jeudi suivant à 11:00

Exercice 1
Développez les fonctions suivantes (avec leur type).

- **insert' e list** : cette fonction prend en paramètre une valeur et une liste triée par ordre croissant et insère l'élément à sa place dans la liste
  Exemple : insert' 7 [2,5,8,9] retourne la liste [2,5,7,8,9]

- **insertionSort list** : cette fonction trie la liste selon l'algorithme du tri par insertion.
  Rappel : Le tri par insertion consiste à trier la liste privée de son premier élément, puis à y insérer cet élément.

- **takeWhile' predicat list** : cette fonction est équivalente à la fonction prédéfinie takeWhile

- **zipWith' fct list1 list2** : cette fonction est équivalente à la fonction prédéfinie zipWith

- **intersect ' list1 list2** : cette fonction retourne l'intersection ensembliste des deux listes passées en paramètre. Développez trois versions de cette fonction : une version récursive, une version avec une fonction d'ordre supérieur, une version avec une liste en compréhension

- **divisorsList v** : cette fonction retourne la liste des diviseurs de v compris dans l'intervalle [1..v-1]. Développez trois versions de cette fonction : une version récursive, une version avec la fonction filter et une fonction anonyme, et une version avec une liste en compréhension
  Exemple : divisorsList 9 retourne la liste [1,3]

- **perfectNumber v** : cette fonction retourne True si v est un nombre parfait . Un nombre est parfait si la somme des diviseurs de x compris dans [1..x-1] est égale à x.
  Exemple : perfectNumber 6 retourne True car 6 = 1 + 2 + 3

- **perfectNumbers n** : cette fonction retourne les n premiers nombres parfaits.
  Exemple : perfectNumbers 4 retourne [6,28,496,8128]

Exercice 2
On choisit de représenter les polynômes par une liste de doublets de la forme :
[(c1,d1), (c2,d2), ..., (cn,dn)] où chaque doublet (ci,di) est un monône de coefficient ci et de degré di
Exemple : Le polynôme $2x + 4x^2 - 2x^4$ est représenté par la liste [(2,1), (4,2), (-2,4)]

Développez la fonction :
calculatePolynomial poly v qui retourne la valeur du polynôme poly pour x = v
Exemple : calculatePolynomial [(2,1), (4,2), (-2,4)] 3 retourne $2*3 + 4 * 3^2 - 2 * 3^4$ soit -120

Développez trois versions de cette fonction : une version récursive, une version avec fonction d'ordre supérieur et une fonction anonyme et une version avec une liste en compréhension