Université de Caen **Année universitaire 2009-2010**
U.F.R Sciences **L3 - Licence Informatique**

Programmation fonctionelle

Travaux dirigés n°1 : Mise à niveau

Exercice 1. Nous rappelons que la suite de Fibonacci est donnée par

$$u_0 = 0,\ u_1 = 1 \text{ et } u_n = u_{n-2} + u_{n-1} \text{ pour } n \geqslant 2.$$

Définir les fonctions suivantes:
- 1. `fibR` calculant le $n$ème terme de la suite de Fibonacci grâce à des appels récursifs naturels;
- 2. `fibT` calculant le $n$ème terme de la suite de Fibonacci de façon terminale;

Exercice 2. Écrire une fonction `long` qui calcule le nombre de chiffres nécessaires à l'écriture d'un nombre en base 10. Exemple : `long 2009` retourne `4`.

Exercice 3. Écrire une fonction `nbOcc` qui retourne le nombre de fois qu'apparaît un élément dans une liste.

Exercice 4. Le but de cet exercice est de simuler un monnayeur permettant de distribuer une somme donnée en pièces de 1, 2, 5 et 10 et minimisant le nombre de pièces nécessaires. Le type de pièces contenues dans le monnayeur sera donné en argument sous forme de liste de valeurs décroissantes. Exemple `monnayeur 23 [10, 5, 2, 1]` retourne `[10, 10, 2, 1]`.

Exercice 5. Le but de cet exercice est la manipulation d'ensembles. On représente un ensemble d'éléments de type `a` à l'aide d'une liste `[a]` : `data ensemble a = E [a]`. Définir les fonctions suivantes :
- 1. `estVide` qui teste si un ensemble est vide;
- 2. `appartient` qui test l'appartenance d'un élément à un ensemble;
- 3. `ajoute` qui ajoute un élément à un ensemble;
- 4. `intersection` qui retourne l'intersection de deux ensembles;
- 5. `union` qui retourne l'union de deux ensembles.