TD 6 d'Introduction à la programmation

Exercices sur la récursivité (suite)

1 Multiplication par un entier p

Dans cette question, nous cherchons à réaliser une multiplication sans avoir recours à l'opérateur de multiplication de *Caml*.

1. Écrire une fonction qui à partir de deux entiers positifs n et p calcule le produit $n \times p$ par une suite d'additions.
2. Est-ce que la fonction peut être utilisée pour deux entiers négatifs ?
3. Est-ce qu'elle peut être utilisée avec un entier positif et un entier négatif ?
4. Est-ce que l'algorithme peut être généralisé pour la multiplication entre flottants, ou entre un entier et un flottant ?

2 Renverser un nombre quelconque

Nous avons vu dans les TD précédents comment renverser un nombre compris entre 0 et 99. Vous avez vu en cours l'algorithme récursif qui permet de renverser un nombre quelconque. En déduire une fonction qui, à partir d'un entier positif n ne contenant aucun 0 dans ses chiffres, renverse les chiffres de cet entier. Par exemple l'image de 356457 par cette fonction serait 754653.

3 Un nombre *palindrome*

Rappelons qu'un palindrome est un nombre qui, lorsqu'on inverse l'ordre de ses chiffres ne change pas de valeur. Par exemple, 43134 est un palindrome. Nous avons vu en cours un algorithme permettant d'indiquer si un entier positif n ne contenant aucun 0 dans ses chiffres est un palindrome ou non. Cet algorithme ne s'appuie pas sur celui de la fonction renverser vue plus haut. Écrire en *Caml* une fonction qui, étant donnée un tel entier n indique s'il s'agit d'un palindrome ou non.

4 Les combinaisons C_n^p

Le nombre de combinaisons de p objets pris dans un ensemble de n objets différents est de :

$$C_n^p = \frac{n!}{p!(n-p)!}$$

Dans la suite, nous allons essayer plusieurs manières d'écrire une fonction qui, à partir de deux entiers positifs n et p, renvoie ce nombre de combinaisons.

1. On peut écrire cette fonction en utilisant la fonction factorielle ci-dessus.
2. En calculant C_{14}^7 par exemple, calculer le nombre de fois où le produit $7 \times 6 \times ... \times 2$ est calculé.
3. Exprimer C_n^p en fonction de C_{n-1}^{p-1} (sans faire intervenir C_{n-1}^p).
4. En déduire une fonction capable de calculer C_n^p à partir de p et de n.
5. Essayer cette fonction pour calculer C_4^3 : est-ce qu'on obtient la valeur attendue ?

6. Enfin on essaie d'écrire cette fonction grâce au triangle de Descartes :

$$\begin{cases} (\forall n \in \mathbb{N}) & C_n^0 = 1 \\ (\forall n \in \mathbb{N}) & C_n^n = 1 \\ (\forall (n,p) \in \mathbb{N}^2) \quad | \quad (n>1) and (p>1) and (n>p) & C_n^p = C_{n-1}^{p-1} + C_{n-1}^p \end{cases}$$

5 Le calcul de $(\cos(nx), \sin(nx))$

Écrire une fonction qui prend en entrée un entier n et une parie de valeurs réelles qui sont en fait les valeurs du cosinus et du sinus d'un certain angle x, et ;qui renvoie la paire $(cos(nx), sin(nx))$. Autrement dit, le deuxième argument de la fonction est une paire (a,b) telle que $a = cosx$ et $b = sinx$. Le schéma de calcul doit bien évidemment être récursif. On pourra se servir des formules de trigonométrie suivantes :

$$\cos(nx) = \cos\left((n-1)x\right)\cos(x) - \sin\left((n-1)x\right) sin(x)$$
$$\sin(nx) = \sin\left((n-1)x\right)\cos(x) - \cos\left((n-1)x\right) sin(x)$$