TP : Introduction à la récursivité

1 Exponentiation

1. Écrire une fonction Caml qui calcule un entier x à une puissance entière n. La fonction sera donc de type `int -> int -> int`.
2. Combien d'appels récursifs cette fonction fait-elle pour calculer x^n ?
3. Trouver une fonction en faisant moins.

2 Factorielle

1. Écrire une fonction qui calcule la factorielle d'un entier.

Le problème de la fonction précédente est qu'elle n'est pas *récursive terminale*. Une fonction est dite *récursive terminale* si la dernière instruction de la fonction est un appel pur à la fonction, ou directement une valeur.

2. En utilisant une fonction auxiliaire à deux variables, donner une définition récursive terminale de la factorielle.

3 La suite de Fibonacci

La suite de Fibonacci est la suite définie par $F_0 = 0$, $F_1 = 1$ et

$$\forall n \in \mathbb{N},\ F_{n+2} = F_{n+1} + F_n.$$

1. Écrire une fonction de type `int -> int` qui calcule le n-ième terme de la suite.
2. Que se passe-t-il si on veut calculer F_{40} ? Combien fait-on d'appels récursifs pour calculer F_n ?
3. Écrire une fonction de type `int -> int*int` qui calcule les n-ième et $n+1$-ième termes de la suite, qui ne fait qu'un appel récursif.
4. On peut même calculer la suite en temps logarithmique :
   (a) Montrer que pour tous p et q
   $$F_{p+q} = F_{p+1}F_q + F_pF_{q-1}.$$
   (b) Exprimer F_{2m}, F_{2m+1} et F_{2m+2} en fonction de F_m et F_{m+1}.
   (c) En déduire une fonction qui calcule (F_n, F_{n+1}) encore plus rapide. Combien fait-elle d'appels récursifs ?