Programmation impérative et fonctionnelle avec OCaml

TP 3

Objectifs :

- Fonctions récursives :
  - récursivité directe ;
  - récursivité terminale.
- Factorisation et abstraction :
  - itérateur ;
  - ordre supérieur (passage de fonction en paramètre) ;
  - polymorphisme ;
  - généricité.

Aggrégation

Dans un nouveau fichier, on va écrire la fonction `sigma` (et des généralisations) qui correspond au symbole Σ dans une sommation :

$$\sum_{i=a}^{b} f(i)$$

1. On suppose que a et b sont des entiers ; $\sum_{i=a}^{b} f(i)$ signifie alors $f(a)+f(a+1)+\ldots+f(b)$. Écrire cette première version de la fonction `sigma` avec trois arguments (a, b et f) dans le style **récursif direct**.
2. Écrire une fonction équivalente en style **récursif terminal**.
3. Généraliser [1] (votre version préférée) avec un incrément quelconque (i.e. pas nécessairement 1) pris en paramètre.
4. Généraliser en prenant en paramètre une fonction s qui calcule l'argument suivant : $f(a) + f(s(a)) + f(s(s(a))) + \cdots + f(b)$
5. Généraliser en prenant en paramètre une condition quelconque `pred` pour les éléments : $f(a) + f(s(a)) + \cdots + f(x)$ où x est le dernier argument tel que `pred(x)` soit vrai (i.e. `pred` est un *prédicat*).
6. Généraliser en permettant le remplacement de la somme par n'importe quelle fonction binaire (possédant un élément neutre à prendre également en paramètre).
7. Écrire la fonction factorielle à l'aide de la fonction `sigma`.
8. Écrire le calcul d'une approximation de π avec une précision `epsilon` donnée en utilisant la série alternée :
$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \ldots$$
9. Écrire une fonction `integrale` qui calcule une approximation de $\int_a^b f(x)dx$ en prenant pour dx un « petit » flottant.

1. On pourra vérifier expérimentalement que la conjecture suivante est bien un théorème :

« On oublie toujours de renommer l'appel récursif quand on fait du copier-coller d'une fonction récursive. »

ainsi que son corollaire :

« Le copier-coller est rarement profitable en programmation. »