TRAVAUX PRATIQUES VII

Récursivité en OCaml

On rappelle que les codes doivent être compilés avec `ocamlopt` et testés !

A Autour du cours

1. Écrire une fonction récursive `factorielle` qui prend en argument un entier `n` et calcule sa factorielle $n!$ (sans regarder dans le cours de préférence).
2. **a.** Écrire une fonction récursive `ligne_etoiles : int -> unit` qui prend en argument un entier `n`, s'évalue à unit et a pour effet de bord d'afficher `n` caractères `*` . Vous pourrez réduire le problème d'afficher `n` étoiles à celui d'afficher `n-1` étoiles.

   b. Écrire une fonction récursive `triangle_bas : int -> unit` qui affiche un triangle de base `n` pointe vers le bas comme ceci pour `n=4` :

```
* * * *
* * *
* *
*
```

   c. Écrire une fonction récursive `triangle_haut : int -> unit` qui comme effet secondaire affiche un triangle de base `n` pointe vers le haut comme ceci pour `n=4` :

```
*
* *
* * *
* * * *
```

   d. Écrire une fonction récursive `carre_creux : int -> unit` qui a pour effet secondaire d'afficher les contours d'un carré `n` $\times$ `n`. Voici un exemple pour `n=4` :

```
* * * *
*     *
*     *
* * * *
```

3. Écrire une fonction récursive `somme_f : (int->int) -> int -> int` qui prend en argument une fonction `f : int->int` et un entier positif `n` et qui caclule $\sum_{i=0}^{n} f(i)$.

Définition 1

Une fonction est dite récursive *terminale* si elle effectue un seul appel récursif, et si cet appel est la *dernière* chose qu'elle fait.

B Un peu de maths

B.1 Maths du supérieur

4. **a.** Les nombres de Catalan forment une suite d'entiers naturels définis par :

$$C_0 = 1, \quad \forall n \geq 1, C_{n+1} = \frac{2(2n+1)}{n+2} C_n$$

Écrire une fonction `catalan` qui prend en argument un entier naturel n et qui calcule C_n.

b. Modifier votre fonction pour qu'elle soit récursive terminale.

5. **a.** La suite de Collatz associée à un $N \in \mathbb{N}$ est définie de la façon suivante :

$$u_0 = N \quad \text{et} \quad \forall n \in \mathbb{N}, u_{n+1} = \begin{cases} \frac{u_n}{2} \text{ si } u_n \text{ est pair} \\ 3 * u_n + 1 \text{ sinon} \end{cases}$$

On appelle *temps de vol* le plus petit entier k tel que $u_k = 0$ (il dépend donc de N car ildépend de u_0). Écrire une fonction `temps_de_vol : int -> int` qui prend en argument N et renvoie son temps de vol.

b. (*Problème ouvert de recherche*) Prouver que votre fonction termine.

B.2 Maths de primaire

Les mathématiques du supérieur, c'est bien. Mais l'école primaire, c'est mieux [1] !

6. **a.** Écrire une fonction `prod : int -> int -> int` qui prend en argument deux entiers positifs `x` et `y` et calcule `x*y` sans utiliser `*` .

b. Écrire une fonction `div : int -> int -> int` qui prend en argument deux entiers positifs `x` et `y` et renvoie `x/y` (division entière) sans utiliser `/` ni `mod` .

c. Écrire une fonction `reste : int -> int -> int` qui prend en argument deux entiers positifs `x` et `y` et renvoie `x mod y` (le reste de `x` divisé par `y`) sans utiliser `/` ni `mod` .

1. Et c'est un peu plus de l'informatique.