Lycée Chateaubriand
Informatique MP 2002/03
Mathieu Giraud

TP Caml n^o 1
Listes et polynômes

Pour ce premier TP de l'année, nous révisons les listes. Chaque partie peut se servir des fonctions définies dans les parties précédentes. Les questions étoilées peuvent être sautées dans un premier temps.

À chaque fois que vous écrivez du code, réfléchissez à la complexité de vos fonctions. Pour les questions plus complexes, pensez à utiliser des fonctions auxiliaires.

L'énoncé de ce TP est disponible à l'adresse `http://magiraud.free.fr/tpcaml`

1 Opérations de base sur les listes

a.* Écrire la définition d'un type `'a liste` muni des fonctions `tete`, `queue`, et `cons` pour simuler les listes. L'appel de `tete` et de `queue` sur la liste vide doit renvoyer une erreur.

Dans la suite, on utilisera le type `list` fourni par Caml. Il est cependant interdit de faire appel aux fonctions Caml prédéfinies.

b. Réaliser les fonctions suivantes :
- `longueur l` (nombre d'éléments de `l`)
- `est_element x l` (vérifie si `x` appartient à `l`)
- `enleve x l` (enlève `x` de `l`, erreur si `x` est non présent dans `l`)
- `ajoute_a_la_fin l x` (ajoute `x` à la fin `l`)
- `miroir l` (liste des éléments de `l` dans l'ordre inverse)

c. Écrire les fonctions `reunion`, `intersection`, `difference` de type `'a list -> 'a list -> 'a list` effectuant la réunion, l'intersection, et la différence de deux listes `l1` et `l2` vues comme des ensembles (chaque élément des listes ne figure qu'une fois, idem pour la liste résultat).

d. Écrire les fonctions `reunion_t`, `intersection_t`, `difference_t` qui s'appliquent cette fois-ci sur des listes triées d'entiers. Leur résultat doit être lui aussi trié (ici, chaque élément peut apparaître plusieurs fois).

e. Écrire la fonction `tri_fusion : int list -> int list` triant une liste d'entiers.

f.* Réaliser les fonctions `somme : int list -> int`, `produit : int list -> int` (somme et produit des éléments d'une liste d'entiers), et `concatenation : string list -> string` (concaténation d'un ensemble de chaînes) à partir d'une même fonction `itere_operateur` que l'on définira.

g.* Comment obtenir la liste de toutes les permutations d'une liste donnée ?

2 Polynômes d'entiers

On considère les polynômes de $\mathbb{Z}[X]$ que l'on représente par des listes d'entiers en commençant par les coefficients de plus petit degré. Ces polynômes sont normalisés : le coefficient de plus grand degré doit être non nul. Ainsi $5X^4 - 4X^3 + X^2 - 1$ est représenté par la liste `[ 5; -4; 1; 0; -1]`. Le polynôme nul est représenté par la liste vide `[]`.

h. Écrire la fonction `degre : int list -> int` donnant le degré d'un polynôme. On pourra renvoyer `-1` pour le polynôme nul.

i. Écrire la fonction `horner : int list -> int -> int` qui calcule la valeur d'un polynôme en un point par la méthode de Hörner.

j. On définit le type `type limite = moins_infini | zero | plus_infini`. Écrire la fonction `limite_poly : limite -> int list -> limite` telle que `limite_poly t P` calcule la limite en `t` du polynôme `P`.

k. Écrire une fonction `addition`. Attention à maintenir les polynômes normalisés.

l.* Écrire une fonction `multiplication`.

m.* Et si on avait choisi l'ordre inverse pour les coefficients ?

3 Entiers longs

n. Effectuer l'opération `123456789 * 987654321`. Que se passe-t-il ?

Afin de réaliser des calculs exacts sur des entiers "arbitrairement longs", on fixe une base $b \in \mathbb{N}$ représentable en machine. Tout entier k peut alors se décomposer canoniquement sous la forme

$$k = \sum_{i=0}^{n} \alpha_i b^i \quad \text{avec} \quad \forall i \quad 0 \leq \alpha_i < n$$

On représente k par la liste de ses coefficients $\alpha_1, \alpha_2, ..., \alpha_n$. Ici, nous prendrons $b = 10^8$. L'entier

1099114353243864027

se représente donc par la liste `[ 3864027 ; 1435324 ; 10991 ]`.

o.* Écrire les fonctions `add_long`, `mult_long`, et `puiss_long` qui traitent les entiers longs.

Sources

- Listes : un de mes TPs de spé
- Polynômes : voir par exemple la partie 5 de E3A 1999