TD 6 :

1 Induction sur les listes

Définir les fonctions suivantes sur les listes [1] :
- `sum`, qui prend la somme de tous les éléments d'une liste d'entiers.
- `length`, qui calcule la longueur d'une liste.
- `append`, qui concatène deux listes.
- `concat`, qui prend une liste de listes et concatène tous ses éléments.
- `modulo`, qui prend une liste et renvoie la liste des éléments congrus à 0 modulo 2
- `filter`, qui prend un prédicat et une liste, et renvoie tous les éléments de la liste qui vérifient le prédicat. Exemple :

```
# filter (fun x -> x mod 2 = 0) [1; 2; 3; 4; 5] ;;
- : int list = [2; 4]
```

2 Multi-ensembles

Un *multi-ensemble* est un ensemble qui peut contenir plusieurs occurrences d'un élément. Autrement dit, un multi-ensemble est une liste où l'ordre des éléments n'importe pas. Par exemple, $\{1,2,3,1,3,3\}$ et $\{3,2,3,1,1,3\}$ sont les mêmes multi-ensembles.

Une manière convenable de représenter un multi-ensemble fini en Caml est comme instance du type

```
type 'a multiset = 'a -> int
```

qui pour chaque élément du type `'a` donne le nombre d'occurrences dans l'ensemble. Donc, les deux ensembles d'en haut sont représentés par la fonction $1 \mapsto 2, 2 \mapsto 1, 3 \mapsto 3$ (parce qu'il y a deux occurrences de 1, une occurrence de 2, trois occurrences de 3).

Écrivez les fonctions suivantes :
- `m_empty` représente le multi-ensemble vide
- `m_add` ajoute un élément à un multi-ensemble
- `multi_of_list` convertit une liste en multi-ensemble. Cette fonction est utile pour tester les fonctions suivantes.
- `m_remove_one` supprime une occurrence d'un élément donné e d'un multi-ensemble m
- `m_remove_all` supprime *toutes* les occurrences d'un élément donné e d'un multi-ensemble m
- `m_union` est l'union de deux multi-ensembles m_1 et m_2 (prendre, pour chaque élément, la somme des occurrences dans m_1 et m_2). Par exemple, $\{1,2,3,1,3,3\} \cup \{2,5,1,7\} = \{1,1,1,2,2,3,3,3,5,7\}$.
- `m_intersection` est l'intersection de deux multi-ensembles m_1 et m_2 (prendre, pour chaque élément, le minimum des occurrences dans m_1 et m_2). Par exemple, $\{1,2,3,1,3,3\} \cup \{2,5,1,7\} = \{1,2\}$.
- `m_diff` est la différence de deux multi-ensembles m_1 et m_2 (prendre, pour chaque élément, la différence (non-négative !) des occurrences dans m_1 et m_2). Par exemple, $\{1,2,3,1,3,3\} - \{2,5,1,7\} = \{1,3,3,3\}$, tandis que $\{2,5,1,7\} - \{1,2,3,1,3,3\} = \{5,7\}$

1. À l'exception de `sum`, ces fonctions sont prédéfinies dans le module `List` de Caml et sont préfixées par le nom du module, par exemple `List.length`. La fonction `append` s'écrit habituellement avec infixe `@`.