TP de *Logique et Circuits* n° 9

Notions Mathématiques

Dans ce TP, comme dans le TP4, on représente un ordre donné comme une fonction de type `'a*'a->int`. On aura `ordre x y >0` si `x` est supérieur à `y`, `ordre x y =0` si `x` est égal à `y`, et `ordre x y <0` si `x` est inférieur à `y`.

On reprend du TP4 les fonctions `ordre_standard` qui définit l'ordre standard ($\leq_1$) sur les entiers, et `ordre_deux` qui définit l'ordre $0 \leq_2 1 \leq_2 -1 \leq_2 2 \leq_2 -2 \leq_2 3 \leq_2 -3 \leq_2 4 \leq_2 -4 \leq_2 \ldots$.

On cherche à représenter des multi-ensembles d'entiers en OCaml. Pour cela, on représente un multi-ensemble par une liste d'entiers.

1. Définir un type `multiensemble` pour représenter des multi-ensembles.
2. Définir une fonction `carac` qui associe à un multiensemble `multiens` une fonction f t.q. pour tout entier x, $f(x)$ est le nombre d'occurrences de x dans le multiensemble `multiens`.
3. Définir une fonction `intersection` qui prend deux multi-ensembles et renvoie leur intersection.
4. Définir de même une fonction `union`.
5. Ecrire une fonction `comparer ordre multiensemble1 multiensemble2` qui renvoie `true` si `multiensemble1` est inférieur à `multiensemble2` selon l'ordre multi-ensemble associé à l'ordre `ordre`.
6. Tester les fonctions `comparer ordre_standard` et `comparer ordre_deux` pour comparer deux multi-ensembles d'entiers.
7. Ecrire une fonction qui prend trois ordres et renvoie l'ordre lexicographique associé.
8. Définir alors une fonction `lex` qui identifie l'ordre lexicographique $\leq_{1,2,1}$.
9. Tester la fonction `comparer lex` pour comparer selon $\leq_{1,2,1}$ deux multi-ensembles contenant des triplets d'entiers.