Feuille d'exercices n°8 de l'option d'informatique.

1. Dans Maple, pour pouvoir travailler avec des entiers très grands, les entiers sont représentés par des listes d'entiers de la façon suivante :

$[a_0; a_1; \ldots; a_n]$ représente l'entier $\sum_{k=0}^{n} a_k * 10000^k$ avec $\forall k \in [\![0, n]\!],\ a_k \in [\![0, 9999]\!]$.

On peut définir le type entier long de la manière suivante :

```
type entier_long == int list;;
```

Bien remarquer qu'ici on utilise un double signe égal (==) car il s'agit de renommer un type de base de caml est non de créer un nouveau type. On peut forcer le typage pour créer des entiers longs :

```
let un = ([1] : entier_long);;
```

À partir d'une telle représentation, imaginez les fonctions suivantes (on se limitera au cas des entiers positifs) :

- **somme** qui fait la somme de deux entiers longs,
- **produit** qui fait le produit de deux entiers longs,
- **affiche** qui affiche à l'écran l'entier long passé en paramètre,
- **string_of_big_int** qui convertit un entier long en chaîne de caractères,
- **big_int_of_int** qui convertit un entier en entier long,
- **fact** qui pour un entier n calcule $n!$ (en entrée on donne un entier Caml et en sortie un entier long),
- **compare a b** qui retourne **true** si l'entier long **a** est plus petit que **b** et **false** sinon,
- **moins** qui calcule la différence entre deux entiers longs si le résultat est positif,
- **div_eucl a b** retourne le quotient et le reste de la division euclidienne de l'entier long **a** par l'entier long **b**.

S'il vous reste du temps, vous pouvez créer un entier long signé

```
type signe = plus | moins;;
type entier_long_signé = entier of signe*entier_long;;
```

puis les fonctions d'addition, de soustraction, de multiplication,....

Et si vous êtes vraiment très rapide, vous pouvez créer le type rationnel de la façon suivante

```
type rationnel = rat of entier_long_signé * entier_long;;
```

puis les fonctions d'addition, de soustraction, de multiplication,....