

TP2 : Boulier

Objectifs : Implémenter une addition sur un boulier Japonais.

1 Le boulier japonais

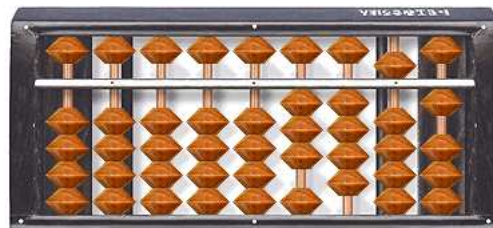
Le but de ce TP est de réaliser un programme qui simule l'utilisation d'un boulier japonais.

Le boulier japonais (appelé également *soroban*) est formé de tiges séparées en deux par une barre transversale ; la partie inférieure compte 4 boules, la partie supérieure 1 boule. Ce type de boulier convient parfaitement pour calculer en base 10. Le boulier ci-dessous n'a que 9 tiges mais rien n'interdit d'avoir plus de tiges (au contraire, cela permet de représenter des nombres plus grands).



Représentation d'un nombre Les boules du haut valent 5 unités et celles du bas 1 unité. On dit que l'on *active* une boule quand on la rapproche de la barre transversale. La colonne la plus à droite représente le chiffre des unités, celle juste à sa gauche représente le chiffre des dizaines, etc.

Par exemple, le nombre 3451 sera représenté par la configuration suivante :



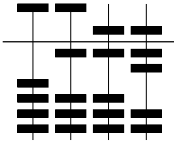
Opérations arithmétiques Les deux opérations de base que l'on peut effectuer avec un boulier sont l'addition et la soustraction¹. Nous nous intéresserons uniquement à l'addition par la suite.

Manuellement, l'addition de deux nombres s'effectue en initialisant le boulier avec le plus grand des deux nombres et en ajoutant le plus petit de gauche à droite. Cependant, afin de simplifier les algorithmes à réaliser, l'addition sera effectuée tige par tige de la droite vers la gauche (comme une addition à la main) avec un boulier initialisé arbitrairement avec l'un des deux nombres.

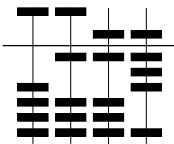
¹les opérations de multiplication ou de division consistent à se ramener à ces deux opérations

Pour additionner 2 tiges ayant respectivement (h_1, b_1) et (h_2, b_2) boules actives en haut et en bas (avec $h_i \in \{0, 1\}$ et $b_i \in \{0, 1, 2, 3, 4\}$) et une éventuelle boule de retenue r (avec $r \in \{0, 1\}$), on commence par ajouter les boules du bas et la retenue. On obtient un nombre entre 0 et 9 que l'on peut représenter sur une seule tige par un couple (h_3, b_3) de boules actives en haut et en bas. On ajoute ensuite les boules du haut h_1 , h_2 et h_3 . On obtient un nombre de boules entre 0 et 3 (donc un chiffre valant 0, 5, 10 ou 15) que l'on peut représenter par une boule haute (active ou non) ainsi qu'une éventuelle boule de retenue.

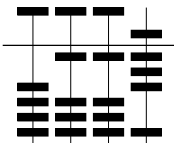
Par exemple, l'addition de 451 à un boulier initialisé avec 167 donne les bouliers suivants :



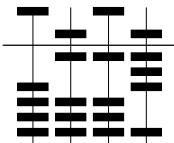
Le boulier de départ initialisé avec 167



En ajoutant les deux tiges des unités on obtient 168



L'addition des tiges des dizaines donne le boulier représentant le nombre 118, avec une retenue qui n'est pas représentée sur le schéma.



Enfin, on ajoute les tiges des centaines en tenant compte de la retenue des dizaines et on obtient 618.

Programmation Nous allons représenter les bouliers par des listes de tiges à l'aide des deux types suivants :

```
type tige = { b : int ; h : int }
type boulier = tige list
```

Ici, les étiquettes **b** et **h** du type **tige** représentent les boules *actives* respectivement du bas et du haut. Par convention, le premier élément d'une liste de tiges représentera le chiffre des unités, le deuxième élément celui des dizaines, etc.

Question 1

Reprendre les types précédents dans un programme `boulier.ml`, et construire à la main sous forme d'une liste de tige les bouliers correspondant aux nombres 18, 43 et 124.

Question 2

Écrire la fonction `print_tige` de type `tige -> unit` qui prend une tige en paramètre et l'affiche à l'écran selon la convention suivante :

- La tige est affichée horizontalement avec la partie haute vers la gauche. Les boules sont représentées par le caractère `o`, les trous par le caractère `-` et la barre de séparation par `|`.

- Par exemple, la représentation demandée de la tige avec la boule de la partie supérieure levée et deux boules de la partie inférieure levées (le chiffre 2) sera :
o-|oo-oo. De même, la représentation de la tige avec la boule de la partie supérieure baissée et toutes les boules de la partie inférieure baissées (le chiffre 5) sera :
-o|-oooo.
- L'affichage d'une tige sera suivi d'un retour à la ligne (`print_newline()`).

Question 3

Écrire la fonction `print_boulier` de type `boulier -> unit` qui prend un boulier en paramètre et l'affiche à l'écran. On affiche une tige par ligne. La tige des unités est affichée en premier, puis la tige des dizaines etc. Par exemple, le boulier représentant le nombre 4512 sera affiché de la façon suivante :

```
o-|oo-oo
o-|o-ooo
-o|-oooo
o-|oooo-
```

Question 4

Écrire la fonction `boulier_of_int` de type `int -> boulier` qui initialise un boulier à partir d'un entier passé en paramètre.

Question 5

À écrire la fonction `int_of_boulier` de type `boulier -> int` qui retourne l'entier associé à un boulier passé en paramètre.

Question 6

Écrire la fonction `ajoute_tige` de type `int -> tige -> tige -> tige * int` telle que `ajoute_tige r t1 t2` retourne la tige et l'éventuelle retenue correspondant à l'addition des tiges `t1` et `t2` et de la retenue `r`.

Question 7

En utilisant la fonction précédente, écrire la fonction `addition` de type `boulier -> boulier -> boulier` qui effectue l'addition de deux bouliers.

Question 8

Réutilisant une fonction préalablement définie, construire une fonction `liste_bouliers` qui prend en argument une suite d'entiers et crée la suite de bouliers qui lui est associée. Tester cette fonction sur la suite `[4;85;12;63;457;695]`.

Question 9

Créer la fonction `addition_liste` qui prend en argument une liste de bouliers et renvoie l'entier correspondant à la somme de tous les bouliers de la liste.

Rappel. Nous rappelons que `a mod b` retourne `a modulo b`.