TP : représentation des entiers
Informatique pour tous

1 Changement de base

On rappelle que pour convertir un entier $< n_{p-1}...n_1n_0 >_b$ d'une base b en base 10 il suffit de calculer $n_0 + bn_1 + b^2n_2 + ... + b^{p-1}n_{p-1}$. Par exemple $< 121 >_3 = 1 \times 3^2 + 2 \times 3 + 1 = 16$ $(= < 16 >_{10})$.

1. Écrire une fonction `to_base10` ayant comme arguments une base `b` et une liste `L`, et renvoyant le nombre $<$ `L[len(L) - 1]`...`L[1]L[0]` $>_b$ converti en base 10.
   Par exemple, `to_base10(3, [0, 2, 1])` doit renvoyer $< 120 >_3$ en base 10, c'est à dire 15.

On rappelle que pour connaître les chiffres d'un entier n en base b on peut écrire sa division euclidienne par b : $n = bq + r$. Le reste r est alors le premier chiffre de n en base b. Il suffit alors d'appliquer à nouveau la méthode en remplaçant n par q, tant que $n \neq 0$.

Par ex., pour convertir 26 en base 2, on effectue des divisions par 2 jusqu'à obtenir 0 :

$$
\begin{array}{r|l}
26 & 2 \\ \hline
\mathbf{0} & 13 \;|\; 2 \\
 & \mathbf{1} \;|\; 6 \;|\; 2 \\
 & \quad \mathbf{0} \;|\; 3 \;|\; 2 \\
 & \quad\quad \mathbf{1} \;|\; 1 \;|\; 2 \\
 & \quad\quad\quad \mathbf{1} \;|\; 0
\end{array}
$$

On en déduit alors que $26 =< 11010 >_2$.

2. Écrire une fonction `from_base10` ayant une base `b` et un nombre `n` en arguments et renvoyant les chiffres de `n` en base `b`.
3. Donner une formule pour la valeur de $< \underbrace{11...11}_{k} >_2$ en base 10 et vérifier avec la fonction précédente.

2 Opérations

On rappelle que l'on peut additionner deux nombres écrits en base b en ajoutant les chiffres un par un, en gardant une retenue de 1 si la somme est supérieure ou égale à b. Voici deux exemples, en base 10 puis en base 2 :

$$
\begin{array}{rr}
 & {}^{1\,1\,1} \\
 & 3\,2\,8 \\
+ & 9\,7\,4 \\ \hline
 & 1\,3\,0\,2
\end{array}
\qquad\qquad
\begin{array}{rr}
 & {}^{1\,1} \\
 & <1\,0\,0\,1>_2 \\
+ & <1\,1>_2 \\ \hline
 & <1\,1\,0\,0>_2
\end{array}
$$

On représente un nombre $n =< n_{p-1}...n_1n_0 >_b$ en base b par la liste `L` de ses chiffres, où `L[i]` $= n_i$ est le ième chiffre de n. Par exemple $< 1010 >_2$ est représenté par `[0, 1, 0, 1]`.

1. Écrire une fonction `add(L1, L2, b)` renvoyant la liste des chiffres de la somme de `L1` et `L2`, en base b. Par exemple `add([1, 0, 0, 1], [1, 1], 2)` doit renvoyer `[0, 0, 1, 1]`. Vérifier votre fonction, éventuellement en utilisant `to_base10` et `from_base10`.

On rappelle que l'on peut multiplier deux nombres écrits en base b de la façon suivante :

$$
\begin{array}{rr}
 & <1\,0\,0\,1>_2 \\
\times & <1\,1>_2 \\ \hline
 & <1\,0\,0\,1>_2 \\
+ & <1\,0\,0\,1\,0>_2 \\ \hline
 & <1\,1\,0\,1\,1>_2
\end{array}
$$

2. Écrire une fonction `mult(L1, L2, b)` renvoyant la liste des chiffres de la multiplication de `L1` et `L2`, en base b. Par exemple `mult([1, 0, 0, 1], [1, 1], 2)` doit renvoyer `[1, 1, 0, 1, 1]`. On pourra réutiliser `add`.
3. Écrire aussi des fonctions pour soustraire/diviser deux nombres écrits en base b.