Systèmes de numération

1 Généralités

Voir feuille à part

2 Les systèmes de numération positionnels

2.1 Définitions générales

Système positionnel. Un système de numération est *positionnel* si la valeur d'un chiffre dépend de sa position.

Base. Un système positionnel est caractérisé par sa *base*, c'est à dire le nombre de symboles de «chiffres» utilisés pour écrire ses nombres.

Exemples.

- Le système que nous utilisons couramment est un système en base 10, dit *décimal*. Il s'est introduit dans l'histoire par correspondance avec les 10 doigts de la main. Les 10 chiffres sont $0, 1, 2, 3, 4, 5, 6, 7, 8$ et 9.
- Le *système binaire* a comme base 2. Ses chiffres sont 0 et 1. Rôle fondamental en informatique.
- Système *octal*, base 8, chiffres $0, 1, 2, 3, 4, 5, 6, 7$.
- Système *hexadécimal*, base 16, chiffres $0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F$.
- Système *sexagésimal*, base 60. Système utilisé par les babyloniens 2000 ans av. JC, pratique car 60 admet de nombreux diviseurs.

Les systèmes octal et hexadécimal sont utilisés souvent en informatique, pratiques car les bases sont des puissances de 2. Par exemple, un octet permet de stocker n'importe quel nombre de $(0)_{16}$ à $(FF)_{16}$.

Remarque. Quand on écrit $(23)_{12}$, dans quelle base est écrit 12 ? Ici, on fixe la convention que la base est 10 quand elle n'est pas explicitement spécifiée. D'autres préfèrent lever l'ambiguïté en écrivant la base en toutes lettres ($(23)_{douze}$).

Base b. Considérons un nombre entier positif N dont la représentation en base b est :

$$a_n a_{n-1} \dots a_2 a_1 a_0$$

Si le contexte est ambigu, on peut expliciter la base utilisée par la notation :

$$(a_n a_{n-1} \dots a_2 a_1 a_0)_b$$

Par définition, sa valeur est alors :

$$a_n \times b^n + a_{n-1} \times b^{n-1} + \cdots + a_2 \times b^2 + a_1 \times b^1 + a_0 \times b^0$$

On rappelle que $b^1 = b$ et $b^0 = 1$, pour tout entier b non nul.

Unicité de l'écriture. Dans les systèmes positionnels, il n'y a qu'une seule façon d'écrire chaque entier.

Théorème 1. *Soit n en entier positif et $b > 2$, alors l'écriture de n en base b est unique.*

Notons que ce n'est pas le cas des systèmes additifs. Par exemple, avec les chiffres romains, on peut noter 1999 de plusieurs façons :

$$MCMXCIX = MCMXCVIIII$$

Exercices. Donner la représentation décimale des nombres suivants :

1. $(201)_3 = 19$;
2. $(201)_7 = 99$;
3. $(201)_8 = 129$;
4. $(201)_{10} = 201$;
5. $(201)_{16} = 513$

Remarque. Le chiffre de droite de la représentation d'un nombre n en base b n'est rien d'autre que le reste de la division de n par b.

2.2 Conversion d'un nombre décimal en sa représentation dans une base b

Méthode directe. Si on connaît les puissances de b, on peut procéder empiriquement comme suit. On veut passer en base 2 le nombre 3317 :

$$\begin{array}{rcccl} 3317 & = & 2048 + 1269 & = & 2^{11} + 1269 \\ 1269 & = & 1024 + 245 & = & 2^{10} + 245 \\ 245 & = & 128 + 117 & = & 2^7 + 117 \\ 117 & = & 64 + 53 & = & 2^6 + 53 \\ 53 & = & 32 + 21 & = & 2^5 + 21 \\ 21 & = & 16 + 5 & = & 2^4 + 5 \\ 5 & = & 4 + 1 & = & 2^2 + 2^0 \end{array}$$

Donc, le nombre 3317 peut se décomposer en :

$$2^{11} + 2^{10} + 2^7 + 2^6 + 2^5 + 2^4 + 2^2 + 2^0 = (110011110101)_2$$

Cependant, cette méthode repose sur la connaissance des puissances de b et est difficilement automatisable.

Algorithme général. On s'appuie sur les deux propositions suivantes :

Proposition 1. *Soit n un nombre entier positif, et $b > 2$. On suppose que n s'écrit :*

$$n = (a_p a_{p-1} \dots a_2 a_1 a_0)_b$$

Par ailleurs, on considère la division euclidienne de n par b :

$$n = q \times b + r$$

Alors :

- *Le reste r est le dernier chiffre de l'écriture de n en base b :*

$$r = a_0$$

- *Le quotient q correspond aux autres chiffres. Plus précisément,*

$$q = (a_p a_{p-1} \dots a_1)_b$$

Exemple. Illustrons la proposition suivante en base dix. Considérons le nombre $(543)_{10}$.

- Si on divise $(543)_{10}$ par 10, on obtient le quotient $(54)_{10}$ et le reste 3 ;
- Si on divise $(54)_{10}$ par 10, on obtient le quotient $(5)_{10}$ et le reste 4 ;
- Si on divise $(5)_{10}$ par 10, on obtient le quotient $(0)_{10}$ et le reste 5.

On en déduit l'algorithme suivant pour construire la représentation de n en base b :

- On calcule la division euclidienne de n par b :

$$n = q \times b + r$$

  On fixe r comme étant le dernier chiffre de la représentation de n.
- On recommence avec q.

Exemple. Calcul de la représentation binaire de 317.

Exercice. Calculer la représentation binaire de 402.

2.3 Conversion d'un nombre écrit en base b en sa représentation décimale

Méthode 1 : Algorithme de changement de base. On peut appliquer le même algorithme : faire des divisions successives du nombre exprimé en base b par 10 exprimé en base b ($10 = (13)_7$).

Problème : Necessité de savoir faire des opérations arithmétiques en base b. Par exemple, voici les tables d'addition et de multiplication en base 7 :

+	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	10
2	2	3	4	5	6	10	11
3	3	4	5	6	10	11	12
4	4	5	6	10	11	12	13
5	5	6	10	11	12	13	14
6	6	10	11	12	13	14	15

×	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	11	13	15
3	0	3	6	12	15	21	24
4	0	4	11	15	22	26	33
5	0	5	13	21	26	34	42
6	0	6	15	24	33	42	51

Sous Unix/Linux, l'utilitaire `bc` peut aider à effectuer ces opérations. Imaginons par exemple qu'on veuille calculer la représentation décimale de $(1363)_7$:

```
ibase=7
obase=7
1363/13
104
1363%13
5
104/13
5
104%13
3
```

L'interprétation de ces résultats est que en base 7, on a les égalités :

$$\begin{aligned} 1363 &= 104 * 13 + 5 \\ 104 &= 5 * 13 + 3 \\ 5 &= 0 * 13 + 5 \end{aligned}$$

Donc :

$$(1363)_7 = 535$$

Remarque. Méthode peu utilisable à cause des difficultés de calcul dans les bases non décimales.

Méthode 2 : Calcul direct. Dire que $n = (a_n a_{n-1} \dots a_0)_b$, c'est dire :

$$n = a_n \times b^n + \dots + a_1 \times b^1 + a_0 \times b^0$$

On peut donc utiliser cette formule pour déterminer la représentation décimale de n par calcul brutal. Notons qu'on peut accélérer en pré-calculant les puissances de b.

Méthode 3 : méthode de Horner. On part de la remarque suivante : on peut réécrire l'expression

$$a_n \times b^n + \dots + a_1 \times b + a_0$$

en :

$$(a_n \times b^{n-1} + \dots + a_1) \times b + a_0$$

De plus, on peut refaire cette opération à l'intérieur des parenthèses, pour obtenir :

$$((\dots(((a_n)\times b + a_{n-1})\times b + a_{n-2})\times b + \dots + a_2)\times b + a_1)\times b + a_0$$

Cette méthode est celle qui demande le moins d'opérations.

Exemple. Sur le nombre $(30624)_8$:

$$(30624)_8 = ((((3)\times 8 + 0)\times 8 + 6)\times 8 + 2)\times 8 + 4 = 12692$$

3 Arithmétique en bases $2, 8$ et 16

Exercices.

4 Nombres non entiers

4.1 Fractions

Écrites sous la forme m/n. On garde la même notation, mais m et n sont écrits dans la base choisie.

4.2 Nombres décimaux en virgule fixe

Définition. Un nombre *décimal* est la donnée d'une partie entière et d'une partie décimale, séparées par une virgule (ou un point pour les anglo-saxons).

Partie entière. On prend simplement la représentation de la partie entière dans la base b.

Partie décimale. On procède de la même façon que pour les entiers, mais avec des puissances négatives de b. Ainsi :

$$(0, a_1a_2\dots a_n)_b = a_1\times b^{-1} + a_2\times b^{-2} + \dots + a_n\times b^{-n}$$

Exemples.

$$\begin{aligned}(301,23)_4 &= 3\times 4^2 + 0\times 4^1 + 1\times 4^0 + 2\times 4^{-1} + 3\times 4^{-2}\\ &= 3\times 16 + 1 + 2\times 0,25 + 3\times 0,0625\\ &= 49,3875\end{aligned}$$

$$\begin{aligned}(0,36207)_8 &= 3\times 8^{-1} + 6\times 8^{-2} + 2\times 8^{-3} + 0\times 8^{-4} + 7\times 8^{-5}\\ &= 8^{-5}(7 + 8(0 + 8(2 + 8(6 + 8(3)))))\\ &= 15495/32768\\ &= 0,47286987\dots\end{aligned}$$

4.3 Conversion

Comment faire pour passer un nombre réel n en base b ?

Algorithme Pour passer en base b la partie entière, on procède comme auparavant. Pour la partie décimale, on fait comme suit :

1. On multiplie la partie décimale par b, on note la partie entière obtenue.
2. On recommence l'opération avec la partie décimale obtenue.

Exemple. Considérons la partie décimale $0,2734375$, qu'on veut passer en base 2. Les multiplications successives donnent :

$$\begin{aligned}
0,2734375 \times 2 &= \mathbf{0},5468750 \\
0,5468750 \times 2 &= \mathbf{1},0937500 \\
0,0937500 \times 2 &= \mathbf{0},1875000 \\
0,1875000 \times 2 &= \mathbf{0},3750000 \\
0,3750000 \times 2 &= \mathbf{0},7500000 \\
0,7500000 \times 2 &= \mathbf{1},5000000 \\
0,5000000 \times 2 &= \mathbf{1},0000000
\end{aligned}$$

Donc :

$$0,2734375 = (0,0100011)_2$$

En particulier, ce nombre est bien somme de puissances négatives de 2. Ce n'est pas toujours le cas :

Exercice. Convertir $0,1$ en base 2. Que se passe-t-il ?

4.4 Virgule flottante

Si on ne veut pas imposer une longueur fixe de partie entière, on passe en «virgule flottante» (types `float` et `double`) en Java. Elle repose sur le théorème suivant :

Théorème 2. *Si y est un réel positif et b une base, il existe un nombre $m \in [0,1]$ et un entier e tels que :*

$$y = m \times b^e$$

C'est la base de la représentation en virgule flottante.

- m est la *mantisse*, de la forme $0,m_1m_2m_3\ldots$. Chacun des m_i est un chiffre de la base b, et $m_1 \neq 0$.
- e est l'*exposant*.

Le couple (m,e) est alors la représentation de y en virgule flottante dans la base b. La contrainte $m_1 \neq 0$ assure l'unicité de cette représentation à base b fixée.