

Fiche 7 - Représentation des entiers

I. SYSTÈME DE NUMÉRATION

I.1. NUMÉRATION EN BASE b

Théorème – Soit $b \in \mathbb{N}$ et $b \geq 2$. Pour tout $n \in \mathbb{N}$, il existe une unique suite $(a_i) \in \llbracket 0, b-1 \rrbracket^{\mathbb{N}}$ nulle à partir d'un certain rang $n_0 \in \mathbb{N}$ telle que

$$n = \sum_{i=0}^{n_0} a_i b^i$$

NOTATION – On dit que la suite (a_i) est la décomposition de n en base b notée

$$n = \overline{a_{n_0} \dots a_2 a_1 a_0}^b = (a_{n_0} \dots a_2 a_1 a_0)_b$$

Exemples

- $(475)_{10} = 4 \times 10^2 + 7 \times 10^1 + 5 \times 10^0$
- $(10011)_2 = 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 16 + 2 + 1 = 19$

Remarque – La numération en base 10 :

- 10 s'écrit avec les chiffres $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
- 2 (dit en *binnaire*) avec les chiffres $\{0, 1\}$
- 16 (dit en *hexadécimal*) avec les chiffres $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$

Exercice 1 Compléter le tableau :

en base10	en base2	en base16
10		
	10	
		10
		BB
	10111	
2025		

Démonstration –

■ **Unicité** : On suppose qu'il existe deux suites $(a_i), (c_i) \in \llbracket 0, b-1 \rrbracket^{\mathbb{N}}$ et $n_0, n_1 \in \mathbb{N}$ tel que

$$n = \sum_{i=0}^{n_0} a_i b^i = \sum_{i=0}^{n_1} c_i b^i$$

Procédons par récurrence forte sur \mathbb{N} :

- Initialisation** : pour $i = 0, n \equiv a_0 \equiv c_0 [b]$ donc $a_0 = c_0 \in \llbracket 0, b-1 \rrbracket$
- Hérédité** : soit $i \geq 0$, on suppose pour tout $k \in \llbracket 0, i \rrbracket, a_k = c_k$.

Il vient :

$$n - \sum_{k=0}^i a_k b^k = n - \sum_{k=0}^i c_k b^k$$

Il vient : $\sum_{k=i+1}^{n_0} a_k b^k = \sum_{k=i+1}^{n_1} c_k b^k$

En divisant par b^{i+1} et après le changement d'indice $j = k - i - 1$, le quotient est :

$$\sum_{j=0}^{n_0-i-1} a_{i+1+j} b^j = \sum_{j=0}^{n_1-i-1} c_{i+1+j} b^j$$

ce qui donne modulo b : $a_{i+1} \equiv c_{i+1} [b]$ et donc $a_{i+1} = c_{i+1}$.
Conclusion : les deux décompositions sont égales.

■ **Existence** : L'algorithme induit lors de l'unicité donne le développement :

- $a_0 \equiv n [b]$
- $a_1 \equiv \frac{n - a_0}{b} [b]$

$$n - \sum_{k=0}^{i-1} a_k b^k$$

pour $i \in \mathbb{N}^*, a_i \equiv \frac{n - \sum_{k=0}^{i-1} a_k b^k}{b^i} [b]$
 Comme la suite (b^k) est strictement croissante et tend vers $+\infty$, il existe $n_0 \in \mathbb{N}$ tel que $n < b^{n_0+1}$ et donc pour $i > n_0, a_i = 0$. Alors la relation obtenue pour $i = n_0$ est

$$\begin{aligned} \frac{n - \sum_{k=0}^{n_0} a_k b^k}{b^{n_0+1}} &\equiv 0 [b] \Leftrightarrow n \equiv \sum_{k=0}^{n_0} a_k b^k [b^{n_0+2}] \\ &\Leftrightarrow n = \sum_{k=0}^{n_0} a_k b^k \in \llbracket 0, b^{n_0+1} - 1 \rrbracket \end{aligned}$$

Proposition – La longueur de l'écriture de n en base b est

$$\left\lfloor \frac{\ln(n)}{\ln(b)} \right\rfloor + 1$$

Démonstration –

La suite (b^k) est strictement croissante et tend vers $+\infty$. Alors l'ensemble $\{k \in \mathbb{N}; b^k \leq n\}$ est fini ; notons

$$n_0 = \max\{k \in \mathbb{N}; b^k \leq n\}$$

La longueur de l'écriture de n en base b est de $n_0 + 1$ chiffres. De plus :

$$\begin{aligned} b^{n_0} \leq n < b^{n_0+1} &\Leftrightarrow n_0 \ln(b) \leq \ln(n) < (n_0 + 1) \ln(b) \\ &\Leftrightarrow n_0 \leq \frac{\ln(n)}{\ln(b)} < n_0 + 1 \quad \text{avec } \ln(b) > 0 \\ &\Leftrightarrow n_0 = \left\lfloor \frac{\ln(n)}{\ln(b)} \right\rfloor \end{aligned}$$

Le nombre de chiffres est $n_0 + 1$.

Exemple

Écriture de 147 en base 2 :

$$\begin{array}{r} 147 \mid 2 \\ 1 \mid 73 \mid 2 \\ \downarrow \mid 1 \mid 36 \mid 2 \\ a_0 \mid \downarrow \mid 0 \mid 18 \mid 2 \\ a_1 \mid \downarrow \mid 0 \mid 9 \mid 2 \\ a_2 \mid \downarrow \mid 1 \mid 4 \mid 2 \\ a_3 \mid \downarrow \mid 0 \mid 2 \mid 2 \\ a_4 \mid \downarrow \mid 0 \mid 1 \mid 2 \\ a_5 \mid \downarrow \mid 1 \mid 0 \\ a_6 \mid \downarrow \\ a_7 \end{array}$$

Alors $147 = \overline{10010011}^2 (= 2^7 + 2^4 + 2^1 + 2^0 = 128 + 16 + 2 + 1)$.

Attention ! Il faut renverser la suite des restes successifs pour obtenir l'écriture du nombre.

Algorithme de numération en base b :

Entrée : $n, b \in \mathbb{N}$ avec $b \geq 2$

L une liste vide

Tant que $n > 0$ faire

$q, r \leftarrow$ quotient et reste de la DE de n par b

 insérer r à gauche de L

$n \leftarrow q$

FinTantque

Si L est vide alors L $\leftarrow [0]$

FinSi

Sortie : L la liste des chiffres de l'écriture de n en base b

Exercice 2 Écrire une fonction $\text{base}(n, b)$ qui retourne l'écriture de n en base b .

I.2. FONCTIONS PYTHON

Notation	Explication
$\text{int}(c, b)$	retourne en base 10 le nombre dont l'écriture, en chaîne de caractères, est c en base b .
$\text{bin}(n)$	retourne l'écriture binaire de l'entier n en base 10
$\text{hex}(n)$	retourne l'écriture hexadécimale de l'entier n en base 10

```
>>> int('1001', 2)
9
>>> int('321', 5)
86
>>> int('ABC', 16)
2748
>>> bin(10)
'0b1010'
>>> hex(30)
'0x1e'
```

I.3. EXERCICES

Exercice 3 Montrer que, dans la base $b \geq 3$, $2(b-1)$ et $(b-1)^2$ s'écrivent avec les mêmes chiffres.

Exercice 4 Soient b et n deux entiers. Montrer que si $\forall k, b > \binom{n}{k}$, alors, on a :

$$(11)_b^n = \left(\binom{n}{0} \binom{n}{1} \cdots \binom{n}{n} \right)_b$$

Exercice 5

1. Déterminer les trois nombres réels a, b, c tels que :

$$\forall x \in \mathbb{R} \quad 8x^4 + 6x^2 + 2 = (2x^2 + x + 1)(ax^2 + bx + c)$$

2. En déduire qu'en base 9, $(80602)_9$ est divisible par $(211)_9$ et écrire, dans cette base, le quotient du premier nombre par le second.

Exercice 6 Quel est la base du système de numération dans lequel le nombre 12551 du système décimal s'écrit 30407 ?

Exercice 7 Un nombre s'écrit xyz en base 7 et zyx en base 9. Déterminer ce nombre.

Exercice 8 Déterminer l'entier qui s'écrit $abca$ en base 11 et $bbac$ en base 7.

Exercice 9 Montrer que dans toute base, le nombre 11211 n'est pas premier.

Exercice 10 Démontrer le résultat suivant :

- Soit a un nombre s'écrivant avec trois chiffres différents,
- on note \bar{a} le nombre s'écrivant avec les mêmes chiffres, dans l'ordre inverse,
- posons $b = |a - \bar{a}|$,
- alors $b + \bar{b} = 1089$.

Exercice 11 Critères de divisibilité

Pour chacun des cas suivant, donner les valeurs éventuelles de x et exprimer, dans le cadre général, le critère de divisibilité utilisé.

$2 1234x$	$7 1070x$	$4 (3x3)_5$
$3 1x43x$	$8 1x52$	$10 (2x10)_3$
$4 32xx2$	$9 x345$	$2 (6x5)_7$
$5 2x30$	$11 1x43x$	$16 (x3x1)_7$
$6 1x1x$	$3 (1010110x)_2$	$4 (3xx)_6$

II. REPRÉSENTATION (EN MACHINE) DES ENTIERS

II.1. LES ENTIERS NATURELS

Les nombres sont stockés en binaire, avec des 0 et des 1.

Vocabulaire :

- un bit (de *binary digit*) est l'unité la plus simple d'un système de numération (0 ou 1, et parfois True ou False)
- un octet : $1\text{o} = 8$ bits
- un byte : c'est l'adressage mémoire la plus petite, en pratique $1\text{ byte} = 1\text{o}$

Certains système enregistre les entiers sur un nombre fixe de bits. Exemple :

- 32 bits dit *entiers courts* (ou *simple précision*)
- 64 bits dit *entiers longs* (ou *double précision*)

⇒ Une variable de type *entier* enregistrée sur n bits peut prendre 2^n valeurs de 0 à $2^n - 1$.

Attention ! le dépassement de capacité se produit lorsque la taille limite de stockage est dépassée, cela génère des inattendus.

Exemples

- L'explosion du premier lanceur d'Ariane 5 en 1996 : le copie-coller du système d'Ariane 4 ne prend par en compte que le système de numération fixe ne peut gérer la grandeur des forces d'accélération de ce modèle plus puissant ; le système de guidage se met hors service ...

- Le bug du compteur de vues de youtube pour *Gangnam style* : le compteur était de 32 bits ce qui limitait le nombre de vues à $2^{31} - 1 = 2147483647$ (en tenant compte d'un bit bannalisé pour le signe). Il a fallu quatre année, mais Psy est venu à bout du compteur ! Youtube utilise maintenant un compteur de 64 bits.

- Le bug de l'an 2038 : Certains systèmes (*POSIX*) représentent le temps écoulé en nombre de seconde depuis le 1er janvier 1970 à 0h00. Pour ceux qui travaille en 32 bits, alors en janvier 2038 le limite de $2^{31} - 1$ secondes sera atteint et l'horloge basculera à -2^{31} , donc en décembre 1901

→ En PYTHON les entiers n'ont pas de limitation dans la représentation (excepté celle imposée par la machine). Cependant, dans certains types comme les tableaux array, PYTHON tronque les entiers sur 32 bits par défaut.

```
import numpy as np
n=2*3*4*5*6*7*8*9
T=np.array([n])
print(2**31-1)
for k in range(10,18):
    n,T[0]=n*k,T[0]*k
    print(k, '! : ', n, ' ', T)
```

Ce qui donne :

```
2147483647
10 ! : 3628800 [3628800]
11 ! : 39916800 [39916800]
12 ! : 479001600 [479001600]
__main__:20: RuntimeWarning: overflow
        encountered in long_scalars
13 ! : 6227020800 [1932053504]
14 ! : 87178291200 [1278945280]
15 ! : 1307674368000 [2004310016]
16 ! : 20922789888000 [2004189184]
17 ! : 355687428096000 [-288522240]
```

II.2. LES ENTIERS RELATIFS

Dans cette parties, les entiers sont représentés sur un nombre $n \in \mathbb{N}^*$ de bits :

- 1 bit pour le signe
- les $n - 1$ suivants donne la taille du nombre

→ **Méthode 1** : pour $a \in \llbracket -2^{n-1} + 1, 2^{n-1} - 1 \rrbracket$

- si $a \geq 0$ alors son écriture est celle de a sur n bits ;
- si $a < 0$ alors on écrit un 1 puis $-a$ sur $n - 1$ bits.

On note que $(100\dots 0)_2$ peut être une seconde représentation de zéro.

Exemple

Sur 8 bits, les nombres représentés sont $\llbracket -127, 127 \rrbracket$

$$\begin{array}{l} 10 \rightarrow \overbrace{00001010}^{10} \\ -10 \rightarrow \underbrace{10001010}_{10} \end{array}$$

→ **Méthode 2** : dite du **complément à 2**

Pour $a \in \llbracket -2^{n-1}, 2^{n-1} - 1 \rrbracket$

- si $a \geq 0$ alors son écriture est celle de a sur n bits ;
- si $a < 0$ alors son écriture est celle de $2^n + a$ sur n bits.

Exemple

Sur 8 bits, les nombres représentés sont $\llbracket -128, 127 \rrbracket$

$$\begin{array}{l} 10 \rightarrow \overbrace{00001010}^{10} \\ -10 \rightarrow \underbrace{11110110}_{246} \end{array}$$

car $2^8 - 10 = 256 - 10 = 246$ et les quotients et les restes successifs de la DE de 246 par 2 sont

q	123	61	30	15	7	3	1	0
r	0	1	1	0	1	1	1	1

Exercice 12

1. Quels sont les nombres que l'on peut écrire sur 6 bits avec la méthode du complément à 2 ?
2. Compléter le tableau suivant des représentations binaires sur 6 bits :

en base 10	méthode 1	complément à 2
19		
-19		
-14		
25		
23		
-1		
	111001	X
	101011	X
	X	111001
	X	101011

Exercice 13 Effectuer l'opération $19 + (-14)$ dans les deux représentations binaires à 6 bits. Qu'en déduire ?

Exercice 14 Montrer que la représentation du complément à 2 est compatible avec l'addition.