

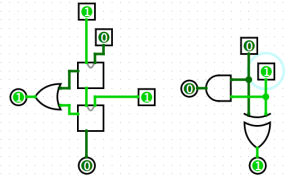
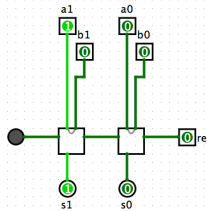
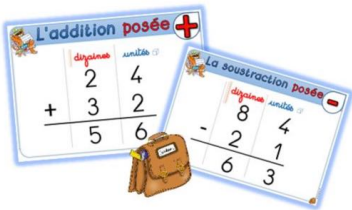
Concepts Informatiques I

2025–2026

Matthieu Picantin



- ◆ systèmes de numération & arithmétique associée
- ◆ représentation des nombres & arithmétique machine
- ◆ codes, codages, entropie, compression
- ◆ contrôle d'erreur (détection, correction)
- ◆ crypto (confidentialité, authenticité, intégrité)
- ◆ logique et calcul propositionnel
- ◆ circuits combinatoires





Numération égyptienne

- ♦ il y a 5000 ans environ
- ♦ numération additive
- ♦ sept hiéroglyphes

1	1
10	10
100	100
1000	1000
10,000	10,000
100,000	100,000
1,000,000	1,000,000



Numération babylonienne

- ♦ il y a 4000 ans environ
- ♦ numération positionnelle sexagésimale (base 60)
- ♦ 59 *chiffres* basés eux sur un système additif décimal

𐎶 1	𐎶𐎵 11	𐎶𐎶𐎵 21	𐎶𐎶𐎶𐎵 31	𐎶𐎶𐎶𐎵𐎶 41	𐎶𐎶𐎶𐎵𐎶𐎵 51
𐎶𐎶 2	𐎶𐎶𐎵 12	𐎶𐎶𐎶𐎵 22	𐎶𐎶𐎶𐎶𐎵 32	𐎶𐎶𐎶𐎶𐎵𐎶 42	𐎶𐎶𐎶𐎶𐎵𐎶𐎵 52
𐎶𐎶𐎶 3	𐎶𐎶𐎶𐎵 13	𐎶𐎶𐎶𐎶𐎵 23	𐎶𐎶𐎶𐎶𐎶𐎵 33	𐎶𐎶𐎶𐎶𐎶𐎵𐎶 43	𐎶𐎶𐎶𐎶𐎶𐎵𐎶𐎵 53
𐎶𐎶𐎶𐎵 4	𐎶𐎶𐎶𐎶𐎵 14	𐎶𐎶𐎶𐎶𐎶𐎵 24	𐎶𐎶𐎶𐎶𐎶𐎶𐎵 34	𐎶𐎶𐎶𐎶𐎶𐎶𐎵𐎶 44	𐎶𐎶𐎶𐎶𐎶𐎶𐎵𐎶𐎵 54
𐎶𐎶𐎶𐎶𐎵 5	𐎶𐎶𐎶𐎶𐎶𐎵 15	𐎶𐎶𐎶𐎶𐎶𐎶𐎵 25	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 35	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵𐎶 45	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵𐎶𐎵 55
𐎶𐎶𐎶𐎶𐎶𐎵 6	𐎶𐎶𐎶𐎶𐎶𐎶𐎵 16	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 26	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 36	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵𐎶 46	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵𐎶𐎵 56
𐎶𐎶𐎶𐎶𐎶𐎶𐎵 7	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 17	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 27	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 37	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵𐎶 47	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵𐎶𐎵 57
𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 8	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 18	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 28	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 38	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵𐎶 48	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵𐎶𐎵 58
𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 9	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 19	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 29	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 39	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵𐎶𐎵 49	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 59
𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 10	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 20	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 30	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 40	𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎶𐎵 50	

Numération romaine

- ♦ il y a 2800 ans environ
- ♦ additive et partiellement soustractive
- ♦ combinaison de sept *chiffres romains*
- ♦ utilisation moderne pour les ordinaux



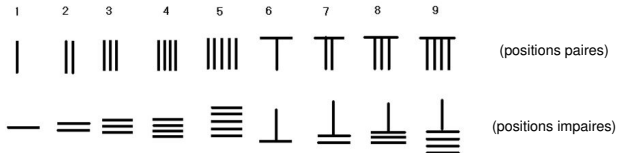
I = 1	C = 100
V = 5	D = 500
X = 10	M = 1000
L = 50	

Chapitre XLIX

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

















Numération chinoise à bâtons

- ♦ il y a 2300 ans environ
- ♦ numération positionnelle décimale
- ♦ pas de chiffre zéro
- ♦ deux séries de neuf *chiffres bâtons*



Numération maya

- ♦ il y a 2300 ans environ
- ♦ numération positionnelle vigécimale (base 20)
- ♦ 19 *chiffres* basés eux sur un système additif quinaire (base 5)
- ♦ chiffre zéro (deux *nombres zéros*: l'un cardinal, l'autre ordinal)

0	1	2	3	4
	•	• •	• • •	• • • •
5	6	7	8	9
	• 	• • 	• • • 	• • • • 
10	11	12	13	14
	• 	• • 	• • • 	• • • • 
15	16	17	18	19
	• 	• • 	• • • 	• • • • 

Numération positionnelle décimale

♦ dix chiffres, disons $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

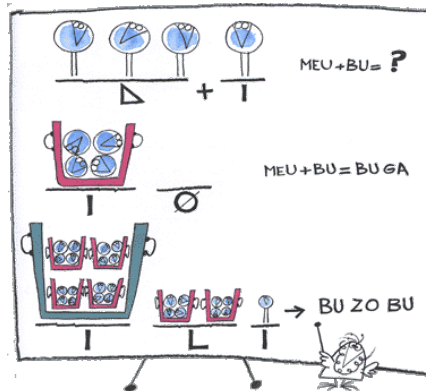
♦ $(a_p \cdots a_0)_{10}$ représente le nombre $\sum_{k=0}^p a_k 10^k$, soit

$$a_p \times 10^p + a_{p-1} \times 10^{p-1} + \cdots + a_2 \times 10^2 + a_1 \times 10 + a_0$$

Hindu-Arabic	1	2	3	4	5	6	7	8	9	0
Arabic	١	٢	٣	٤	٥	٦	٧	٨	٩	.
Devanagari (Hindi)	१	२	३	४	५	६	७	८	९	०
Tibetan	༡	༢	༣	༤	༥	༦	༧	༨	༩	༠
Bengali	১	২	৩	৪	৫	৬	৭	৮	৯	০
Thai	๑	๒	๓	๔	๕	๖	๗	๘	๙	๐

Numération positionnelle en base $b > 0$

- ♦ exactement b chiffres, disons $\{0, 1, \dots, b-1\}$
- ♦ $(a_p \dots a_0)_b$ représente le nombre $\sum_{k=0}^p a_k b^k$



Comment convertir une écriture en base b $(a_p \cdots a_0)_b$
vers une écriture en base d ? $(c_q \cdots c_0)_d$

Méthode par encadrements successifs (plutôt naïve)

- ♦ on encadre le nombre n entre deux facteurs successifs

$$c_k d^k \leq n < (c_k + 1) d^k \quad (\text{avec } 0 < c_k < d)$$

- ♦ on collecte le chiffre c_k (pour la position k)
- ♦ on recommence avec le nombre n auquel on a retranché $c_k d^k$
- ♦ on s'arrête quand le nombre est nul
et on renvoie l'écriture avec les chiffres collectés

Comment convertir une écriture en base b $(a_p \cdots a_0)_b$
vers une écriture en base d ? $(c_q \cdots c_0)_d$

Méthode par divisions successives (vrai algo)

- ♦ on divise n par la base d (calcul fait en base b)
- ♦ on collecte le reste r
- ♦ on recommence avec le quotient q
- ♦ on s'arrête quand le nombre est nul
et on renvoie la suite **inversée** des restes collectés

$$\begin{array}{r|l} n & d \\ r & q \end{array}$$

à privilégier quand
la base b est confortable

Comment convertir une écriture en base b $(a_p \cdots a_0)_b$
 vers une écriture en base d ? $(c_q \cdots c_0)_d$

Méthode par recomposition – méthode *de Horner*

à privilégier quand
la base d est confortable

- ♦ on utilise la définition $\sum_{k=0}^p a_k b^k$ en calculant dans la base d
- ♦ on peut diminuer le nombre d'opérations en factorisant:

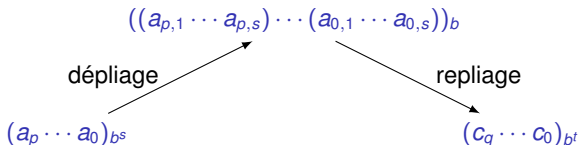
$$\begin{aligned}
 & a_p \times b^p + a_{p-1} \times b^{p-1} + \cdots + a_2 \times b^2 + a_1 \times b + a_0 \\
 = & (a_p \times b^{p-1} + a_{p-1} \times b^{p-2} + \cdots + a_2 \times b + a_1) \times b + a_0 \\
 & \vdots \\
 = & (\cdots (a_p) \times b + a_{p-1}) \times b + \cdots + a_2) \times b + a_1) \times b + a_0
 \end{aligned}$$

Comment convertir une écriture en base b^s $(a_p \cdots a_0)_{b^s}$
 vers une écriture en base b^t $(c_q \cdots c_0)_{b^t}$

Méthode par dépliage-repliage

à privilégier face
aux autres méthodes

- ♦ on écrit chaque chiffre de l'écriture en base b^s dans la base b
 $a_i = a_{i,1} \cdots a_{i,s}$
- ♦ on regroupe par paquet de t chiffres **en commençant par la droite**
- ♦ on convertit chaque paquet dans la base b^t

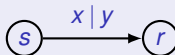


Comment convertir une écriture en base b $(a_p \cdots a_0)_b$
vers une écriture en base d ? $(c_q \cdots c_0)_d$

Méthode par transducteur

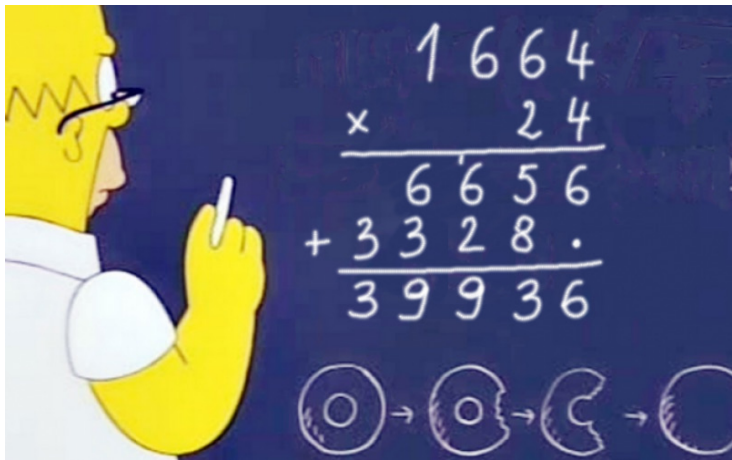
*hors programme!
(rendez-vous en S3)*

- ♦ on construit un transducteur avec
 - ▶ d états numérotés $\{0, 1, \dots, d-1\}$
 - ▶ pour chaque état s et chaque lettre x dans $\{0, 1, \dots, b-1\}$



avec r le reste et y le quotient de la division de $s \times b + x$ par d

- ♦ pour chaque écriture en base b à convertir en base d
 - ▶ on fait lire le mot depuis l'état de départ 0
 - ▶ on collecte l'état d'arrivée
 - ▶ on recommence avec le mot obtenu en sortie
 - ▶ on s'arrête quand le mot devient vide
et on renvoie la suite **inversée** des états d'arrivée collectés



Addition posée

- ♦ on s'appuie sur la *table d'addition* en base b
- ♦ on dispose les nombres en colonnes: chiffres des unités, chiffres des b -aines, chiffres des b^2 -aines, etc
- ♦ on effectue la somme des chiffres de la colonne la plus à droite:
on *pose* son chiffre des unités
on *reporte* la retenue sur la ou les colonnes à gauche
- ♦ on recommence sur la colonne immédiatement à gauche, etc

Multiplication posée

- ♦ on s'appuie sur les *tables d'addition et de multiplication* en base b
- ♦ on dispose les deux nombres $(a_p \cdots a_0)_b$ et $(c_q \cdots c_0)_b$ en colonnes: chiffres des unités, chiffres des b -aines, chiffres des b^2 -aines, etc
- ♦ on effectue le produit de $(a_p \cdots a_0)_b$ par le nombre $c_k b^k$ pour $0 \leq k \leq q$
- ♦ on conclut avec l'addition posée de ces $q + 1$ produits intermédiaires

La "preuve" par 9 pour un calcul en base 10

- ♦ n'est pas une *preuve* de correction, mais un moyen simple de vérification (avec un certain indice de confiance): c'est un test, une épreuve
- ♦ utilise les propriétés de l'arithmétique modulaire
- ♦ consiste à "refaire" le calcul mais en considérant les nombres *modulo 9* (obtenus en faisant la somme de leurs chiffres, etc)

$$\sum_{k=0}^p a_k 10^k \equiv \sum_{k=0}^p a_k \pmod{9}$$

