# Binaire

## Écriture décimale

L'écriture habituelle des entiers se fait dans le système décimal (en base 10). Par exemple, 70 685 c'est $7 \times 10\,000 + 0 \times 1\,000 + 6 \times 100 + 8 \times 10 + 5 \times 1$ :

| 7 | 0 | 6 | 8 | 5 |
|---|---|---|---|---|
| 10000 | 1000 | 100 | 10 | 1 |
| $10^4$ | $10^3$ | $10^2$ | $10^1$ | $10^0$ |

Puissances de 10 : on note $10^k$ pour $10 \times 10 \times \cdots \times 10$ (avec $k$ facteurs).

| $d_{p-1}$ | $d_{p-2}$ | $\cdots$ | $d_i$ | $\cdots$ | $d_2$ | $d_1$ | $d_0$ |
|---|---|---|---|---|---|---|---|
| $10^{p-1}$ | $10^{p-2}$ | $\cdots$ | $10^i$ | $\cdots$ | $10^2$ | $10^1$ | $10^0$ |

On calcule donc l'entier correspondant aux chiffres $[d_{p-1}, d_{p-2}, \ldots, d_2, d_1, d_0]$ par la formule :

$$n = d_{p-1} \times 10^{p-1} + d_{p-2} \times 10^{p-2} + \cdots + d_i 10^i + \cdots + d_2 \times 10^2 + d_1 \times 10^1 + d_0 \times 10^0$$

## Écriture décimale

**`decimale_vers_entier()`**

Usage : `decimale_vers_entier(liste_decimale)`
Entrée : une liste de chiffres entre 0 et 9
Sortie : l'entier dont l'écriture décimale est la liste

Exemple : si l'entrée est `[1,2,3,4]`, la sortie est `1234`.

*Indications.* Faire la somme d'éléments du type

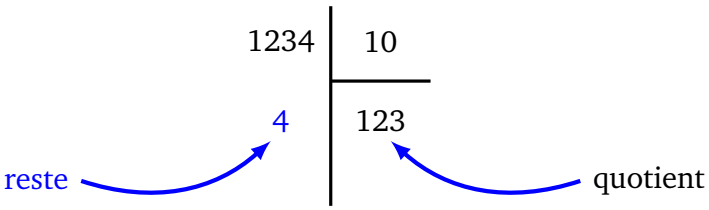
$$d_i \times 10^i \qquad \text{pour } 0 \leqslant i < p$$

où $p$ est la longueur de la liste et $d_i$ est le chiffre en position $i$ *en comptant à partir de la fin*.

# Écriture décimale (bis)

Comment associer à l'entier 1234 la liste de ses chiffres $[1, 2, 3, 4]$ ?
On aura besoin de la division euclidienne par 10 :

- on calcule par `n % 10` le ***reste***, on l'appelle aussi *n* ***modulo*** 10
- on calcule par `n // 10` le ***quotient*** de cette division.




Les commandes `Python` sont simplement `n % 10` et `n // 10`.
Exemple : `1234 % 10` vaut `4` ; `1234 // 10` vaut `123`.

## Écriture décimale (bis)

- Le chiffre des unités de $n$ s'obtient comme le reste modulo 10 : c'est $n\%10$. Exemple $1234\%10 = 4$.
- Le chiffre des dizaines s'obtient à partir du quotient de la division de $n$ par 10, puis en prenant le chiffre des unités de ce nombre : c'est donc $(n//10)\%10$. Exemple : $1234//10 = 123$, puis on a $123\%10 = 3$ ; le chiffre des dizaines de 1234 est bien 3.
- Pour le chiffre de centaines, on calcule le quotient de la division de $n$ par 100, puis on prend le chiffre des unités. Exemple $1234//100 = 12$ ; 2 est le chiffre des unités de 12 et c'est le chiffre des centaines de 1234.
- Pour le chiffre des milliers on calcule le quotient de la division par 1000 puis on prend le chiffre des unités…

# Écriture décimale (bis)

Objectifs : décomposer un entier en la liste de ses chiffres (en base 10).

> **Algorithme.**
> Entrée : un entier $n > 0$
> Sortie : la liste de ses chiffres
> - Partir d'une liste vide.
> - Tant que $n$ n'est pas nul :
>   — ajouter $n\%10$ au début de la liste,
>   — faire $n \leftarrow n//10$.
> - Le résultat est la liste.

Programme l'algorithme en une fonction `entier_vers_decimale()`.

**`entier_vers_decimale()`**

Usage : `entier_vers_decimale(n)`
Entrée : un entier positif
Sortie : la liste de ses chiffres

Exemple : si l'entrée est `1234`, la sortie est `[1,2,3,4]`.

# Binaire

**Puissances de 2.** On note $2^k$ pour $2 \times 2 \times \cdots \times 2$ (avec $k$ facteurs). Par exemple, $2^3 = 2 \times 2 \times 2 = 8$.

| $2^7$ | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ |
|---|---|---|---|---|---|---|---|
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |

**Écriture binaire : un exemple.**
Tout entier admet une écriture binaire, c'est-à-dire une écriture où les seuls chiffres sont des 0 ou des 1. En binaire, les chiffres sont appelés des ***bits***. Par exemple, 1.0.1.1.0.0.1 (prononce les chiffres un par un) est l'écriture binaire de l'entier 89. Comment faire ce calcul ? C'est comme pour la base 10, mais en utilisant les puissances de 2 !

| 1 | 0 | 1 | 1 | 0 | 0 | 1 |
|---|---|---|---|---|---|---|
| 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ |

Donc l'écriture 1.0.1.1.0.0.1 représente l'entier :

$$1 \times 64 + 0 \times 32 + 1 \times 16 + 1 \times 8 + 0 \times 4 + 0 \times 2 + 1 \times 1 = 64 + 16 + 8 + 1 = 89.$$

## Binaire

**`Python` et le binaire.** `Python` accepte que l'on écrive directement les entiers en écriture binaire, il suffit d'utiliser le préfixe « `0b` ». Exemples :

- avec `x = 0b11010`, alors `print(x)` affiche 26,
- avec `y = 0b11111`, alors `print(y)` affiche 31,
- et `print(x+y)` affiche 57.

Calcule les entiers dont l'écriture binaire est donnée ci-dessous. Tu peux le faire à la main ou t'aider de `Python`. Par exemple 1.0.0.1.1 vaut $2^4 + 2^1 + 2^0 = 19$ ce que confirme la commande `0b10011` qui renvoie 19.

- 1.1, 1.0.1, 1.0.0.1, 1.1.1.1
- 1.0.0.0.0, 1.0.1.0.1, 1.1.1.1.1
- 1.0.1.1.0.0, 1.0.0.0.1.1
- 1.1.1.0.0.1.0.1

# Binaire

**Écriture binaire : formule.**

| $b_{p-1}$ | $b_{p-2}$ | $\cdots$ | $b_i$ | $\cdots$ | $b_2$ | $b_1$ | $b_0$ |
|---|---|---|---|---|---|---|---|
| $2^{p-1}$ | $2^{p-2}$ | $\cdots$ | $2^i$ | $\cdots$ | $2^2$ | $2^1$ | $2^0$ |

On calcule donc l'entier correspondant aux *bits* $[b_{p-1}, b_{p-2}, \ldots, b_2, b_1, b_0]$ comme une somme de termes $b_i \times 2^i$, par la formule :

$$n = b_{p-1} \times 2^{p-1} + b_{p-2} \times 2^{p-2} + \cdots + b_i 2^i + \cdots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

# Binaire

Objectifs : à partir de l'écriture binaire, retrouver l'entier (en écriture décimale usuelle).
Écris une fonction `binaire_vers_entier(liste_binaire)` qui à partir d'une liste représentant l'écriture binaire calcule l'entier correspondant.

**`binaire_vers_entier()`**

Usage : `binaire_vers_entier(liste_binaire)`
Entrée : une liste de *bits* 0 et 1
Sortie : l'entier dont l'écriture binaire est la liste

Exemples :

- entrée `[1,1,0]`, sortie `6`
- entrée `[1,1,0,1,1,1]`, sortie `55`
- entrée `[1,1,0,1,0,0,1,1,0,1,1,1]`, sortie `3383`

## Binaire avec Python

`Python` calcule très bien l'écriture binaire d'un entier grâce à la fonction `bin()`.

**`python : bin()`**

Usage : `bin(n)`
Entrée : un entier
Sortie : l'écriture binaire de $n$ sous la forme d'une chaîne de caractères commençant par `'0b'`

Exemple :

- `bin(37)` renvoie `'0b100101'`
- `bin(139)` renvoie `'0b10001011'`

## Calcul de l'écriture binaire

Pour calculer l'écriture binaire d'un entier $n$, c'est la même méthode que pour calculer l'écriture décimale mais en remplaçant les divisions par 10 par des divisions par 2.
Nous avons donc besoin :

- de $n\%2$ : le reste de la division euclidienne de $n$ par 2 (appelé aussi $n$ modulo 2) ; le reste vaut soit 0 soit 1.
- de $n//2$ : le quotient de cette division.

Note que le reste $n\%2$ vaut soit 0 (quand $n$ est pair) soit 1 (quand $n$ est impair).
Voici la méthode générale pour calculer l'écriture binaire d'un entier :

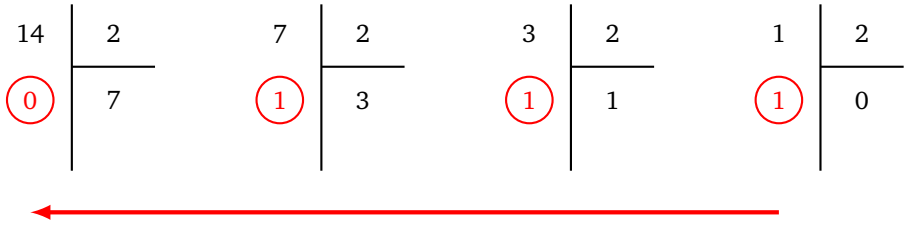
- On part de l'entier dont on veut l'écriture binaire.
- On effectue une suite de divisions euclidiennes par 2 :
  — à chaque division, on obtient un reste qui vaut 0 ou 1 ;
  — on obtient un quotient que l'on divise de nouveau par 2… On s'arrête quand ce quotient est nul.
- On lit l'écriture binaire comme la suite des restes, mais en partant du dernier reste.

## Calcul de l'écriture binaire

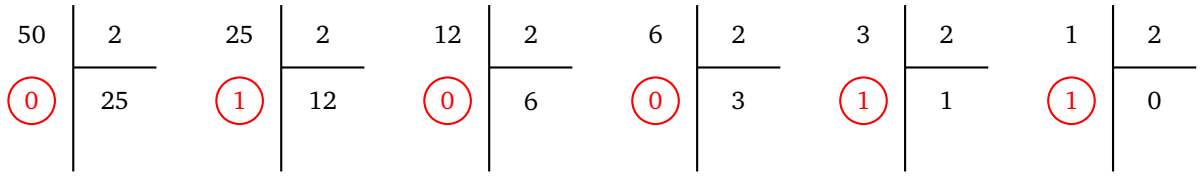
**Exemple.**
Calcul de l'écriture binaire de 14.

- On divise 14 par 2, le quotient est 7, le reste est 0.
- On divise 7 (le quotient précédent) par 2 : le nouveau quotient est 3, le nouveau reste est 1.
- On divise 3 par 2 : quotient 1, reste 1.
- On divise 1 par 2 : quotient 0, reste 1.
- C'est terminé (le dernier quotient est nul).
- Les restes successifs sont 0, 1, 1, 1. On lit l'écriture binaire à l'envers c'est 1.1.1.0.

Les divisions se font de gauche à droite, mais on lit les restes de droite à gauche.



## Calcul de l'écriture binaire

> **Exemple.**
> Écriture binaire de 50.
>
> 
> 
>
> Les restes successifs sont 0, 1, 0, 0, 1, 1, donc l'écriture binaire de 50 est 1.1.0.0.1.0.

Calcule à la main l'écriture binaire des entiers suivants. Vérifie tes résultats à l'aide de la fonction `bin()` de `Python`.

- 13, 18, 29, 31,
- 44, 48, 63, 64,
- 100, 135, 239, 1023.

## Calcul de l'écriture binaire

Programme l'algorithme suivant en une fonction `entier_vers_binaire()`.

**Algorithme.**
Entrée : un entier $n > 0$
Sortie : son écriture binaire sous la forme d'une liste

- Partir d'une liste vide.
- Tant que $n$ n'est pas nul :
  — ajouter $n\%2$ au début de la liste,
  — faire $n \leftarrow n//2$.
- Le résultat est la liste.

`entier_vers_binaire()`

Usage : `entier_vers_binaire(n)`
Entrée : un entier positif
Sortie : son écriture binaire sous forme d'une liste

Exemple : si l'entrée est 204, la sortie est `[1,1,0,0,1,1,0,0]`.