
Représentation des entiers naturels

Christophe Viroulaud

Première - NSI

DonRep 01

Représentation des entiers naturels

Christophe Viroulaud

Première - NSI

DonRep 01

Un ordinateur n'interprète que des signaux électriques :
- impulsion électrique → 1,
- pas d'impulsion → 0.

Nous parlons de **BInary DigiTS** ou plus simplement la contraction **bits**.

Un ordinateur n'interprète que des signaux électriques :

- impulsion électrique $\rightarrow 1$,
- pas d'impulsion $\rightarrow 0$.

Nous parlons de **BInary DigiTS** ou plus simplement la contraction **bits**.

Comment représenter les nombres entiers dans la mémoire de l'ordinateur ?

Comment représenter les nombres entiers dans la mémoire de l'ordinateur ?

Sommaire
1. Cellules mémoires
2. Encodage des entiers naturels

Sommaire

Cellules mémoires

Figure 1 – Le bit est la plus petite unité informatique.

Cellules mémoires

Figure 1 – Le bit est la plus petite unité informatique.

FIGURE 2 – 8 bits représentent un **octet**.

FIGURE 2 – 8 bits représentent un **octet**.

À retenir
Un ordinateur manipule des **mots mémoires** de 2, 4 ou 8 octets.

FIGURE 3 – Une machine *32 bits* manipule des mots de 4 octets (4×8 = 32 bits) quand elle effectue des opérations.

À retenir

Un ordinateur manipule des **mots mémoires** de 2, 4 ou 8 octets.

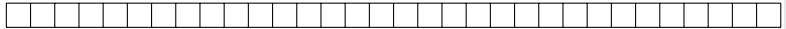


FIGURE 3 – Une machine *32 bits* manipule des mots de 4 octets (4×8 = 32 bits) quand elle effectue des opérations.

Activité 1 : Chaque bit accepte 2 valeurs possibles : 0 ou 1. Avec 1 bit nous pouvons donc avoir 2 combinaisons possibles.
1. Combien de combinaisons peut-on réaliser avec 1 octet ?
2. Même question pour 1 mot-mémoire 32 bits ?

Activité 1 : Chaque bit accepte 2 valeurs possibles : 0 ou 1. Avec 1 bit nous pouvons donc avoir 2 combinaisons possibles.
1. Combien de combinaisons peut-on réaliser avec 1 octet ?
2. Même question pour 1 mot-mémoire 32 bits ?

1. 4 milliards
2. PC récent = 64 bits

Correction
- 1 octet $\rightarrow 2^8 = 256$ combinaisons.
- 1 mot 32 bits $\rightarrow 2^{32} = 4\ 294\ 967\ 296$ combinaisons.

Correction

- 1 octet $\rightarrow 2^8 = 256$ combinaisons.
- 1 mot 32 bits $\rightarrow 2^{32} = 4\ 294\ 967\ 296$ combinaisons.

Sommaire

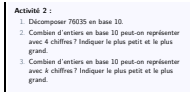
Sommaire

Écriture en base 10

$6103 = 6\times10^3 + 1\times10^2 + 0\times10^1 + 3\times10^0$

Écriture en base 10

$$6103 = 6\times10^3 + 1\times10^2 + 0\times10^1 + 3\times10^0$$


Activité 2 :
1. Décomposer 76035 en base 10.
2. Combien d'entiers en base 10 peut-on représenter avec 4 chiffres ? Indiquer le plus petit et le plus grand.
3. Combien d'entiers en base 10 peut-on représenter avec k chiffres ? Indiquer le plus petit et le plus grand.

Activité 2 :

1. Décomposer 76035 en base 10.
2. Combien d'entiers en base 10 peut-on représenter avec 4 chiffres ? Indiquer le plus petit et le plus grand.
3. Combien d'entiers en base 10 peut-on représenter avec k chiffres ? Indiquer le plus petit et le plus grand.

Correction
- Nous pouvons représenter 10^4 entiers avec 4 chiffres : de 0 à 9999.
- Nous pouvons représenter 10^k entiers avec k chiffres : de 0 à 10^k-1.

Correction

- Nous pouvons représenter 10^4 entiers avec 4 chiffres : de 0 à 9999.
- Nous pouvons représenter 10^k entiers avec k chiffres : de 0 à 10^k-1.

Sommaire

Sommaire

Et si on avait que 2 doigts ?

Écriture en base 2

$5 = 1\times2^2 + 0\times2^1 + 1\times2^0$
$5_{10} = 101_2$

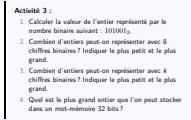
Remarque
Afin d'éviter les ambiguïtés, il est possible d'écrire un nombre en précisant sa base : 1001_2.

Écriture en base 2

$$5 = 1\times2^2 + 0\times2^1 + 1\times2^0$$
$$5_{10} = 101_2$$

Remarque

Afin d'éviter les ambiguïtés, il est possible d'écrire un nombre en précisant sa base : 1001_2.


Activité 3 :
1. Calculer la valeur de l'entier représenté par le nombre binaire suivant : 101001_2.
2. Combien d'entiers peut-on représenter avec 8 chiffres binaires ? Indiquer le plus petit et le plus grand.
3. Combien d'entiers peut-on représenter avec k chiffres binaires ? Indiquer le plus petit et le plus grand.
4. Quel est le plus grand entier que l'on peut stocker dans un mot-mémoire 32 bits ?

Activité 3 :

1. Calculer la valeur de l'entier représenté par le nombre binaire suivant : 101001_2.
2. Combien d'entiers peut-on représenter avec 8 chiffres binaires ? Indiquer le plus petit et le plus grand.
3. Combien d'entiers peut-on représenter avec k chiffres binaires ? Indiquer le plus petit et le plus grand.
4. Quel est le plus grand entier que l'on peut stocker dans un mot-mémoire 32 bits ?

Correction
▶ $1\times2^5+0\times2^4+1\times2^3+0\times2^2+0\times2^1+1\times2^0=41$

Correction

- $1\times2^5+0\times2^4+1\times2^3+0\times2^2+0\times2^1+1\times2^0=41$

Correction
- $1\times2^5+0\times2^4+1\times2^3+0\times2^2+0\times2^1+1\times2^0=41$
- Avec un octet nous pouvons représenter 2^8 entiers : de 0 à 255

Correction

- $1\times2^5+0\times2^4+1\times2^3+0\times2^2+0\times2^1+1\times2^0=41$
- Avec un octet nous pouvons représenter 2^8 entiers : de 0 à 255

Correction
- $1\times2^5+0\times2^4+1\times2^3+0\times2^2+0\times2^1+1\times2^0=41$
- Avec un octet nous pouvons représenter 2^8 entiers : de 0 à 255
- Avec k chiffres nous pouvons représenter 2^k entiers : de 0 à 2^k-1

Correction

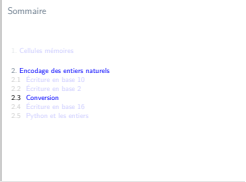
- $1\times2^5+0\times2^4+1\times2^3+0\times2^2+0\times2^1+1\times2^0=41$
- Avec un octet nous pouvons représenter 2^8 entiers : de 0 à 255
- Avec k chiffres nous pouvons représenter 2^k entiers : de 0 à 2^k-1

Correction
- $1\times2^5+0\times2^4+1\times2^3+0\times2^2+0\times2^1+1\times2^0=41$
- Avec un octet nous pouvons représenter 2^8 entiers : de 0 à 255
- Avec k chiffres nous pouvons représenter 2^k entiers : de 0 à 2^k-1
- $2^{32}-1=4294967295$

Correction

- $1\times2^5+0\times2^4+1\times2^3+0\times2^2+0\times2^1+1\times2^0=41$
- Avec un octet nous pouvons représenter 2^8 entiers : de 0 à 255
- Avec k chiffres nous pouvons représenter 2^k entiers : de 0 à 2^k-1
- $2^{32}-1=4294967295$

Sommaire



Sommaire

Conversion

Chaque entier est converti en base 2 avant d'être stocké en mémoire.

$1\times2^5+0\times2^4+1\times2^3+0\times2^2+0\times2^1+1\times2^0=41$

- Dans 41 il y a **1** fois $2^5=32$
- Dans 9 $(41-32)$ il y a **0** fois $2^4=16$
- Dans 9 il y a **1** fois $2^3=8$
- Dans 1 $(9-8)$ il y a **0** fois $2^2=4$
- Dans 1 il y a **0** fois $2^1=2$
- Dans 1 il y a **1** fois $2^0=1$

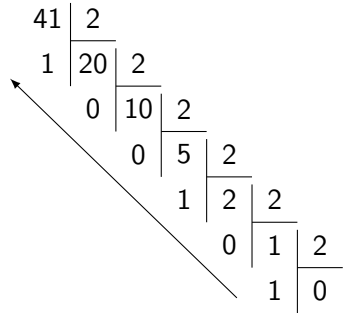
Conversion

Chaque entier est converti en base 2 avant d'être stocké en mémoire.

$$1\times2^5+0\times2^4+1\times2^3+0\times2^2+0\times2^1+1\times2^0=41$$

- Dans 41 il y a **1** fois $2^5=32$
- Dans 9 $(41-32)$ il y a **0** fois $2^4=16$
- Dans 9 il y a **1** fois $2^3=8$
- Dans 1 $(9-8)$ il y a **0** fois $2^2=4$
- Dans 1 il y a **0** fois $2^1=2$
- Dans 1 il y a **1** fois $2^0=1$

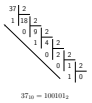
Méthode équivalente



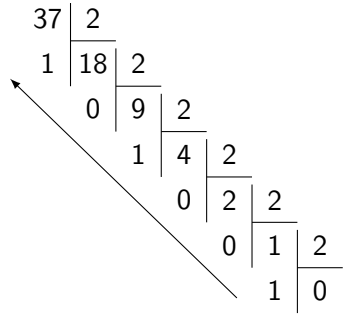
Activité 4 : Convertir 37_{10} en base 2.

> **Activité 4 :** Convertir 37_{10} en base 2.

Correction



Correction



$$37_{10} = 100101_2$$

Sommaire

Sommaire

Écriture en base 16

La base 16 est régulièrement utilisé pour représenter les nombres binaires plus facilement. Chaque chiffre hexadécimal est représenté par 4 bits.

À retenir

1 octet est représenté par 2 chiffres hexadécimaux.

Écriture en base 16

La base 16 est régulièrement utilisé pour représenter les nombres binaires plus facilement. Chaque chiffre hexadécimal est représenté par 4 bits.

À retenir

1 octet est représenté par 2 chiffres hexadécimaux.

Activité 5 : Compléter le tableau.

décimal	hexadécimal	bits
0	0	0000
1	1	0001
2	2	
3	3	
4	4	
5	5	
6	6	
7	7	
8	8	
9	9	
10	A	
11	B	
12	C	
13	D	
14	E	
15	F	

Correction

décimal	hexadécimal	bits
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Correction

décimal	hexadécimal	bits
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Sommaire

1. Cellules mémoires
2. Encodage des entiers naturels
2.1 Écriture en base 10
2.2 Écriture en base 2
2.3 Conversion
2.4 Écriture en base 16
2.5 Python et les entiers

Sommaire

Python et les entiers
- Par défaut les nombres entiers sont encodés en base 10 en Python.

Python et les entiers

- Par défaut les nombres entiers sont encodés en base 10 en Python.

Python et les entiers

- Par défaut les nombres entiers sont encodés en base 10 en Python.
- Pour utiliser des nombres binaires, il suffit d'ajouter le préfixe 0b.

Python et les entiers

- Par défaut les nombres entiers sont encodés en base 10 en Python.
- Pour utiliser des nombres binaires, il suffit d'ajouter le préfixe `0b`.

Python et les entiers

- Par défaut les nombres entiers sont encodés en base 10 en Python.
- Pour utiliser des nombres binaires, il suffit d'ajouter le préfixe 0b.
- Le préfixe 0x permet de manipuler des nombres en base hexadécimale.

Python et les entiers

- Par défaut les nombres entiers sont encodés en base 10 en Python.
- Pour utiliser des nombres binaires, il suffit d'ajouter le préfixe `0b`.
- Le préfixe `0x` permet de manipuler des nombres en base hexadécimale.

Python et les entiers

- Par défaut les nombres entiers sont encodés en base 10 en Python.
- Pour utiliser des nombres binaires, il suffit d'ajouter le préfixe 0b.
- Le préfixe 0x permet de manipuler des nombres en base hexadécimale.
- La fonction bin() convertit en base 2 n'importe quelle valeur.

Python et les entiers

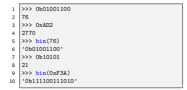
- Par défaut les nombres entiers sont encodés en base 10 en Python.
- Pour utiliser des nombres binaires, il suffit d'ajouter le préfixe `0b`.
- Le préfixe `0x` permet de manipuler des nombres en base hexadécimale.
- La fonction `bin()` convertit en base 2 n'importe quelle valeur.

Activité 6 : Dans la console, écrire :
1. 0b01001100
2. 0xAD2
3. bin(76)
4. Convertir le nombre binaire 10101 en décimal.
5. Convertir le nombre hexadécimal F3A en base 2.

Activité 6 : Dans la console, écrire :

1. `0b01001100`
2. `0xAD2`
3. `bin(76)`
4. Convertir le nombre binaire `10101` en décimal.
5. Convertir le nombre hexadécimal `F3A` en base 2.

Correction



Correction

```
>>> 0b01001100
76
>>> 0xAD2
2770
>>> bin(76)
'0b01001100'
>>> 0b10101
21
>>> bin(0xF3A)
'0b111100111010'
```