RD01 REPRÉSENTATION DES ENTIERS POSITIFS

I. Origine du binaire en informatique

Pour communiquer, deux êtres humains doivent parler la même langue. L'échange d'informations entre les composants d'un ordinateur repose sur le même principe.

Dans les années 40, lorsqu'il a fallu passer de l'idée abstraite de machine de Turing (modèle mathématique) à la réalisation concrète du premier ordinateur (machine physique), la question des composants matériels à choisir s'est posée (transistor, lampe, DEL, aimant, tube, bande, ...). Pour ces composants, il est possible de réduire leur fonctionnement à 2 états exclusifs l'un de l'autre : état 0 et état 1.

- le transistor est bloqué (état 0) ou passant (état 1),
- l'interrupteur est ouvert (état 0) ou fermé (état 1),
- la lampe est éteinte (état 0) ou allumée (état 1),
- le ruban est trouée (état 0) ou non (état 1),

Le langage de base finalement choisie pour l'informatique, et encore utilisé aujourd'hui, est le binaire dont l'unité est le **bit** : binary digit (ou nombre binaire), pouvant prendre comme seules valeurs **0 ou 1**.

Exercice 1

Combien d'états différents peut-on coder avec 1 bit ? 2 bits ? 8 bits ? n bits ? .....

II. Représentation binaire

Exercice 2

Représentation binaire d'un entier naturel
Sur 8 bits, donner la représentation des premiers entiers naturels :

V. décimale	V. binaire		V. décimale	V. binaire		V. décimale	V. binaire		V. décimale	V. binaire
0			8			16			24	
1			9			17			25	
2			10			18			26	
3			11			19			27	
4			12			20			28	
5			13			21			29	
6			14			22			30	
7			15			23			31	

Propriété 1 : A retenir

- un entier naturel est représenté en binaire par une suite de bits (0 et 1).
- un entier naturel codé sur n bits peut prendre 2^n valeurs différentes, de 0 à $2^n - 1$.
- avec 8 bits, on peut représenter les entiers naturels de 0 à 255.
- le **bit de poids faible** est le bit de droite. C'est le seul multiplié par une valeur impaire. C'est donc le bit qui donnera la **parité**. Si la valeur se termine par 0, le nombre sera pair, sinon il sera impair
- le **bit de poids fort** est le bit de gauche.
- un octet est un groupe de 8 bits.

III. Conversion Binaire – Décimal : convertir $1101\,1011_2$ en décimal

Rang du bit	7	6	5	4	3	2	1	0
Puissance de 2	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Bit	1	1	0	1	1	0	1	1
Valeur	128	64	0	16	8	0	2	1

Ainsi $1101\,1011_2 = 128 + 64 + 16 + 8 + 2 + 1 = 219_{10}$

Exercice 3 : Convertir les valeurs suivantes en décimal : $1001\,0100_2$, 1010_2, $110\,1101_2$

IV. Conversion Décimal – Binaire

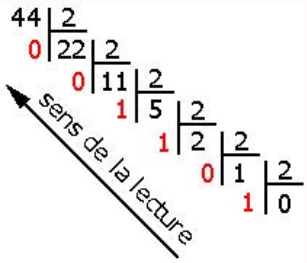
Méthode 1 : Par la méthode des divisions successives

Par exemple pour convertir 44_{10} en binaire :

- On effectue des divisions par 2 jusqu'à ce que le quotient soit nul.
- On lit ensuite les restes de la droite vers la gauche.

On a donc :

$$44_{10} = 0010\,1100_2$$



Méthode 2 : Par la méthode des soustractions successives

Lorsqu'on cherche à convertir 222_{10}, on remarque que la plus grande puissance de 2 inférieure à 222 est $2^7 = 128$. Nous aurons donc besoin de 8 bits.

- $222 \geqslant 128$. On peut donc décomposer $222 = 2^7 + 94$
- $94 \geqslant 64$. On peut donc décomposer $94 = 2^6 + 30$
- $30 < 36$
- $30 \geqslant 16$. On peut donc décomposer $36 = 2^4 + 14$
- $14 \geqslant 8$. On peut donc décomposer $14 = 2^3 + 6$
- $6 \geqslant 4$. On peut donc décomposer $6 = 2^2 + 2$
- $2 \geqslant 2$. On peut donc décomposer $2 = 2^1 + 0$
- $0 < 1$

Conversion de la valeur décimale 222 en binaire

Exercice 4 : En utilisant les deux méthodes, convertir 234 **et** 86 **sur 8 bits**

V. Opérations binaires

Propriété 1 : Addition en binaire

L'addition se pratique de la même façon que dans le calcul décimal usuel, et repose sur la table d'addition binaire.

+	0	1
0	00	01
1	01	10

Exercice 5

Poser et effectuer une addition en binaire : $1\,0101_2 + 1110_2$

Propriété 2 : Multiplication en binaire

La multiplication se pratique de la même façon que dans le calcul décimal usuel, et repose sur la table de multiplication suivante :

X	0	1
0	00	00
1	00	01

Exercice 6 : Poser et effectuer une multiplication en binaire : $1\,0101_2 \times 100_2$

Propriété 3 : A retenir

- **Multiplier** une valeur décimale par 2 revient à décaler tous les bits d'un cran vers la gauche et à ajouter un 0 sur le bit de poids faible.
- **Diviser** une valeur décimale par 2 revient à décaler tous les bits d'un cran vers la droite. Le bit de poids faible est perdu. Cela revient à récupérer le quotient de la division par 2.
- **Multiplier** une valeur décimale par 2^n revient à décaler tous les bits de n crans vers la gauche et à ajouter des 0 sur les bits de poids faible.
- **Diviser** une valeur décimale par 2^n revient à décaler tous les bits de n crans vers la droite. Les bit de poids faible sont perdus. Cela revient à récupérer le quotient de la division par 2^n.

VI. L'Hexadécimal

Propriété 1 : Représentation d'un entier en hexadécimal

Si le binaire est une base comprenant 2 éléments, l'hexadécimal est une base comprenant 16 éléments. Il est donc également possible de coder les valeurs décimales suivantes en hexadécimal.
Puisqu'il nous faut 16 symboles. On utilise les 10 premiers chiffres et on rajouter les symboles $\{A, B, C, D, E, F\}$

Valeur décimale	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Valeur hexa																
Valeur décimale	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Valeur hexa																

Propriété 2 : Conversion hexadécimale

Les méthodes utilisées sont les memes que précédemment en remplaçant la valeur 2 par 16

Exercice 7 : Convertir en décimal : $A3_{16}$, BCF_{16}, 42_{16}

Exercice 8 : Convertir en hexadécimal : 364_{10}, 23_{10}, 44_{10}

Propriété 3 : Conversion binaire – hexadécimal

Dans le cas de conversion binaire - hexadécimal (et inversement) il n'est pas utile de repasser par le décimal. En effet un symbole hexadécimal peut directement etre codé par 4 bits.
Pour convertir des entiers naturels de la représentation binaire à la représentation hexadécimale : Il suffit de regrouper les bits par quatre et de leur associer leur chiffre hexadécimal.
$01101000_2 = 68_{16}$

Exercice 9 : Convertir en hexadécimal :

- 110110100011_2
- 001111111001_2

Exercice 10 : Convertir en binaire :

- $A3E_{16}$
- $8DC_{16}$

Propriété 4 : Conversion en base b

Dans le même ordre d'idée il est donc possible de faire des conversion depuis n'importe quelle base b.

Pour tout entier naturel b non nul appelé base, tout nombre entier naturel n peut se décomposer sous la forme :

$$n = \sum_{k=0}^{N} a_k b^k = a_N b^N + \ldots + a_1 b^1 + a_0 b^0$$

L'écriture en base b de n sera alors $\overline{a_N a_{N-1} \ldots a_1 a_0}$

Méthode 1 : Conversion en python

Code Python

```python
a_bin = 0b1101    # le 0b veut dire que la valeur est écrite en binaire
a_dec = int(a_bin)    # conversion en décimal - valeur entière égale à 13
a_hex = hex(a_dec)    # conversion en hexadécimal - valeur égale à 0xd

b_dec = 44
b_bin = bin(b_dec)    # conversion en binaire - valeur égale à 0b101100
b_hex = hex(b_dec)    # conversion en hexadécimal - valeur égale à 0x2c

# de manière générale, pour convertir un entier n en base b, on peut utiliser:
a_dec = int('1101', 2)     # conversion binaire en décimal égale à 13
a_hex = hex(int('1101', 2))    # conversion binaire en hexadécimal égale à 0xd
a_bin = bin(int('A3', 16))    # conversion hexadécimal en binaire égale à 0b10100011
```