Université Mohammed V
Faculté des Sciences
Département d'Informatique

Cours M6 pour SMIA
Introduction à l'Informatique

M. El Marraki
N. El Khattabi

2020 – 2021

Cours N°7

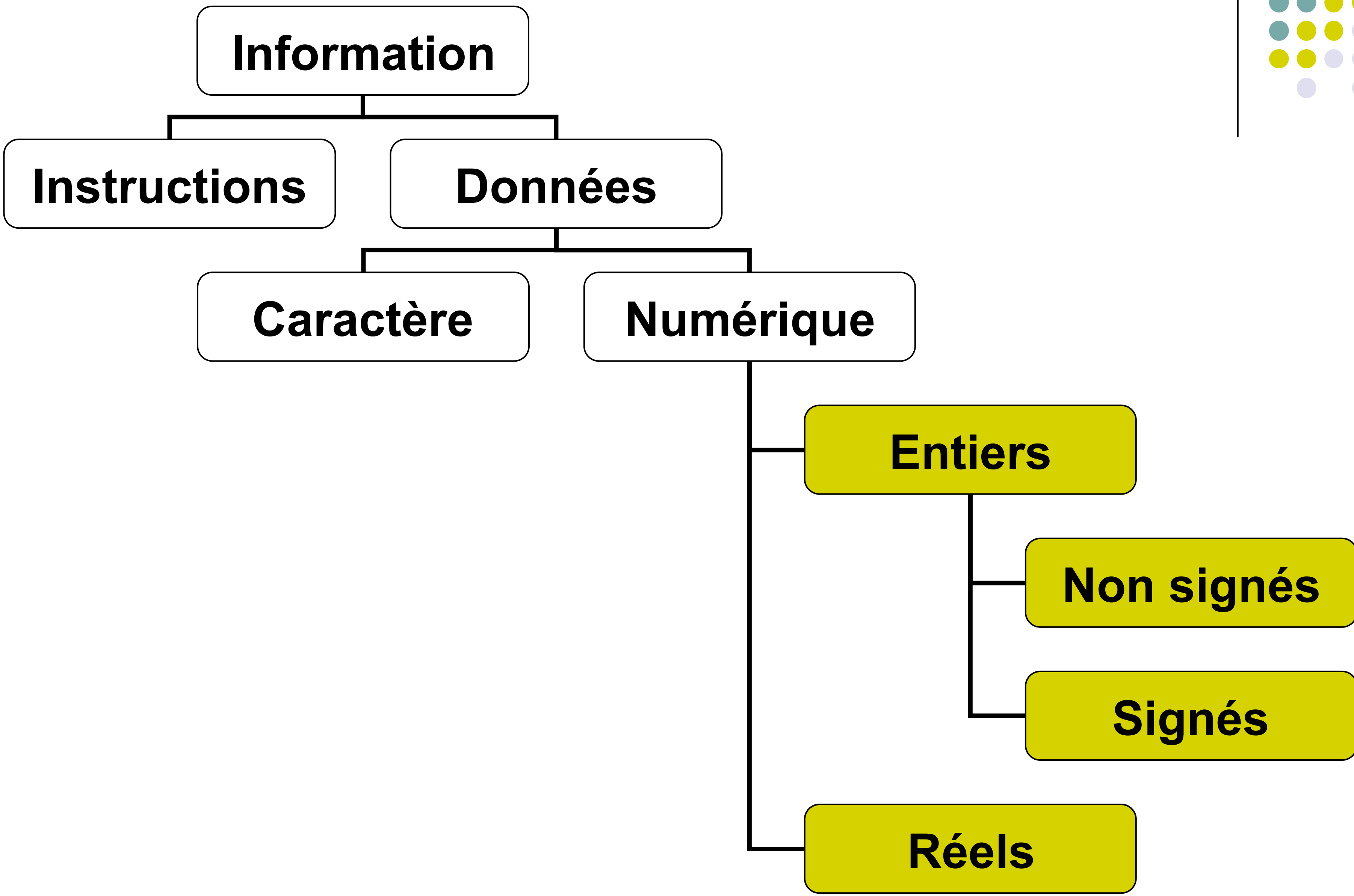
Sommaire

I. Le codage

Introduction

Système d’énumération

Codage des nombres entiers


Information
Instructions
Données
Caractère
Numérique
Entiers
Non signés
Signés
Réels

Codage des entiers naturels (1)

Utilisation du **code binaire pur** :

- L'entier naturel (positif ou nul) est représenté en base 2,
- Les **bits** sont **rangés** selon leur **poids**, on **complète** à **gauche** par des **0.**

Exemple : sur un octet, $10_{(10)}$ se code en binaire pur?

$$0\ 0\ 0\ 0\ 1\ 0\ 1\ 0_{(2)}$$

Codage des entiers naturels (1)

Exercice : Sur combien d'octet minimum, on peut coder le nombre 498 en binaire pur ?

Codage des entiers naturels (1)

Exercice : Sur combien d'octet minimum, on peut coder le nombre 498 en binaire pur ?

Réponse :

On remarque que $496/16 = 31$

Donc $496 = 11111_2 * 10000_2 = 111110000_2$.

Par conséquent il faut au moins 9 bits pour coder le nombre 498 en binaire pur. On a $\mathbf{498 = 111110010_2}$.

Sur 2 octets le nombre 498 s'écrit en binaire pur :

$$\mathbf{0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 0_{(2)}}$$

Codage des entiers naturels (2)

Etendu du codage binaire pur :

Codage sur n bits : représentation des nombres de 0 à $2^n - 1$

- **sur 1 octet (8 bits): codage des nombres de 0 à $2^8 - 1 = 255$**
- **sur 2 octets (16 bits): codage des nombres de 0 à $2^{16} - 1 = 65\,535$**
- **sur 4 octets (32 bits) : codage des nombres de 0 à $2^{32} - 1 = 4\,294\,967\,295$**

Nombre de valeurs possibles

Avec des mots de n bits, il est possible de représenter 2^n valeurs différentes.

Par exemple :

- pour n=1, nous pouvons représenter deux valeurs 0 et 1.
- Avec deux bits, nous pouvons représenter 4 valeurs codées, 00, 01, 10 et 11.
- Avec trois bits, nous pouvons représenter 8 valeurs codées, 000, 001, 010, 011, 100, 101, 110 et 111.
- Avec n bits, nous pouvons représenter 2^n valeurs différentes.

Nombre de valeurs possibles

Inversement, pour coder n valeurs différentes, il faudra [$\log_2(n)$] bits, où $\log_2(n)$ est le plus petit entier k tel que .

$$2^{k-1} < n \leq 2^k$$

Ainsi, pour coder 11 valeurs différentes, il faudra 4 bits car $2^3 < 11 \leq 2^4$.

(avec 4 bits on peut coder $2^4 = 16$ valeurs différentes)

Nombre de valeurs possibles

- Combien de bits faut-il pour coder n=560 ?

  $2^9=512 < 560 < 2^{10} = 1024$.

  Donc, Il faut 10 bits pour coder n=560

- Combien de bits faut-il pour coder n=2160 ?

  $2^{11} = 2048 < 2160 < 2^{12} = 4096$.

  Donc, Il faut 12 bits pour coder n=2160

Codage des entiers relatifs

Il existe au moins trois façons pour coder :

- code binaire signé (*par signe et valeur absolue*)
- code complément à 1
- code complément à 2 (Utilisé sur ordinateur)

Codage des entiers relatifs -Binaire signé-

- Le bit le plus significatif est utilisé pour représenter **le signe** du nombre :
  - si le bit le plus fort **= 1** alors **nombre négatif**
  - si le bit le plus fort **= 0** alors **nombre positif**
- Les autres bits codent **la valeur absolue** du nombre

Exemple : coder le nombre **-44** sur **8** bits :

44 = 4 x 11, le codage de **11** en binaire est $\mathbf{1011_2}$ et celui de 44 est $\mathbf{44=101100_2}$ par conséquent le codage de **-44** en binaire signé est :

$$-44=10101100_{(bs)}$$

Codage des entiers relatifs -Binaire signé-

- Le bit le plus significatif est utilisé pour représenter **le signe** du nombre :
  - si le bit le plus fort **= 1** alors **nombre négatif**
  - si le bit le plus fort **= 0** alors **nombre positif**
- Les autres bits codent **la valeur absolue** du nombre

Codage des entiers relatifs
-Binaire signé-

Etendu de codage :

Avec **n bits**, on code tous les nombres entre

$$-(2^{n-1} - 1) \quad \text{et} \quad (2^{n-1} - 1)$$

- Avec **4 bits** : **-7** et **+7**
- Avec **5 bits** : **-15** et **+15**
- Avec **8 bits** : **-127** et **+127**
- Avec **11 bits** : **-1023** et **+1023**

Codage des entiers relatifs -Binaire signé-

Limitations du binaire signé:

- **Deux représentations du zéro : + 0 et - 0**
- Sur 4 bits : $+0 = 0000_{(bs)}$, $-0 = 1000_{(bs)}$
- Sur 8 bits : $+0 = 00000000_{(bs)}$, $-0 = 10000000_{(bs)}$

Multiplication et l'addition sont moins évidentes.

Binaire signé

Malheureusement, avec cette représentation, une soustraction de deux nombres a-b ne pourra pas se faire par l'addition de a et -b.

En effet, avec ce codage, -5 serait codé sur 8 bits par 10000101 et si on effectue l'opération 9+(-5), on obtient comme résultat

	00001001	9
+	10000101	+(-5)
=	10001110	soit -14 et non 4 !

Binaire signé (exercice)

Coder 100 et -100 en binaire signé sur 8 bits.
Avec 8 bits on peut coder les nombres de -127 et 127

- $100_{(bs)}$:
  Comme le nombre 100 est positif, son code en binaire signé et le même que celui en binaire pure :

  $$100_{(10)} = 96 + 4 = 3*32 + 4$$
  $$= 01100000_{(2)} + 00000100_{(2)} = 01100100_{(bs)}$$

- $-100_{(bs)}$:
  Comme le nombre 100 est négatif, son code en binaire signé est :

  $$-100_{(10)} = 11100100_{(bs)}$$

Binaire signé (exercice)

Décoder en décimal $11000111_{(bs)}$ et $00001111_{(bs)}$

- $11000111_{(bs)}$: le nombre commence par 1, donc il est négatif sa valeur absolue est
  $1000111_2 = 1000000_2 + 0000111_2 = 2^6+7 = 64+7=71$,
  donc $11000111_{(bs)} = -71_{(10)}$

- $00001111_{(bs)}$: le nombre commence par 0, donc il est positif, sa valeur est 15,
  donc $00001111_{(bs)} = 15_{(10)}$

Binaire signé (exercice)

Calculer : **1– 2** en binaire signé sur 8 bits

$1 = 0000\ 0001_{bs}$

$-2 = 1000\ 0010_{bs}$

$$
\begin{array}{lrl}
1-2 = 1+(-2) : & & 0000\ 0001_{bs} \\
 & + & 1000\ 0010_{bs} \\
 & = & 1000\ 0011_{bs} = -3_{10}
\end{array}
$$

On obtient -3 au lieu de -1 !!

Codage des entiers relatifs code complément à 1 (cà1)

Aussi appelé Complément Logique (CL) ou Complément Restreint (CR) ou Complément 1 (Cà1) :

- les nombres positifs sont codés de la même façon qu'en binaire pure.
- un nombre négatif est codé en inversant chaque bit de la représentation de sa valeur absolue

Le bit le plus significatif est utilisé pour représenter le signe du nombre :

- si le bit le plus fort = 1 alors nombre négatif
- si le bit le plus fort = 0 alors nombre positif

Codage des entiers relatifs code complément à 1 (cà1)

Exemple : -24 en complément à 1 sur 8 bits

- |-24| en binaire pur → $0\,0\,0\,1\,1\,0\,0\,0_{(2)}$ puis
- on inverse les bits → $1\,1\,1\,0\,0\,1\,1\,1_{(cà1)}$

Limitation :

- deux codages différents pour 0 (+0 et -0)
- Sur 8 bits : $+0=0\,0\,0\,0\,0\,0\,0\,0_{(cà1)}$ et $-0=1\,1\,1\,1\,1\,1\,1\,1_{(cà1)}$
- Multiplication et l'addition sont moins évidentes.

Codage des entiers relatifs -Binaire signé-

Etendu de codage :

Avec **n bits**, on code tous les nombres entre

$$-(2^{n-1}-1) \quad \text{et} \quad (2^{n-1}-1)$$

- Avec **4 bits** : **-7** et **+7**
- Avec **5 bits** : **-15** et **+15**
- Avec **8 bits** : **-127** et **+127**
- Avec **11 bits** : **-1023** et **+1023**

Codage des entiers relatifs code complément à 1 (cà1)

Coder 100 et -100 par complément à 1 (cà1) sur 8 bits

$100_{(10)} = (01100100)_{(cà1)}$ (100 = 96 + 4 = 3*32 + 4)

$-100_{(10)} = (10011011)_{(cà1)}$

Décoder en décimal $(11000111)_{(cà1)}$ et $(00001111)_{(cà1)}$

$(11000111)_{(cà1)} = -56_{(10)}$

$(00001111)_{(cà1)} = 15_{(10)}$

Calculer : 1 – 2 en complément à 1 sur 8 bits

Codage des entiers relatifs code complément à 1 (cà1)

Calculer : -1-2 en complément à 1 sur 8 bits

$-1 = 1111\ 1110_{(cà1)}$

$-2 = 1111\ 1101_{(cà1)}$

$-1-2 = -1+(-2) :$

$$\begin{array}{rl} & 1111\ 1110_{(cà1)} \\ + & 1111\ 1101_{(cà1)} \\ = & 1111\ 1011_{(cà1)} = -\ 4_{10} \end{array}$$

Car $1111\ 1011_{(cà1)} \rightarrow 0000\ 0100_{(2)} = 4_{10}$

On obtient - 4 au lieu de -3 !!

Codage des entiers relatifs code complément à 2 (cà2)

Aussi appelé Complément Vrai (CV) :

- les nombres **positifs** sont codés de la même manière qu'en **binaire pure.**
- un nombre **négatif** est codé en **ajoutant** la valeur **1** à son **complément à 1**

Le **bit le plus significatif** est utilisé pour représenter le **signe** du nombre

Codage des entiers relatifs code complément à 2 (cà2)

Exemple : - 24 en complément à 2 sur 8 bits

- 24 est codé par $0\,0\,0\,1\,1\,0\,0\,0_{(2)}$
- - 24 → $1\,1\,1\,0\,0\,1\,1\,1_{(cà1)}$

donc - 24 est codé par

$$
\begin{array}{r}
1\,1\,1\,0\,0\,1\,1\,1_{(cà1)} \\
+ \qquad\qquad\qquad 1 \\
\hline
1\,1\,1\,0\,1\,0\,0\,0_{(cà2)}
\end{array}
$$

Codage des entiers relatifs code complément à 2

Un seul codage pour 0. Par exemple sur 8 bits :

- **+ 0 est codé par $00000000_{(cà2)}$**
- **- 0 est codé par $11111111_{(cà1)}$**

Donc - 0 sera représenté par $00000000_{(cà2)}$

Codage des entiers relatifs code complément à 2

Etendu de codage :

Avec n bits, on peut coder de -2^{n-1} à $(2^{n-1}-1)$

Sur 1 octet (8 bits), codage des nombres de -128 à 127

+0 = 00000000	-0=00000000
+1 = 00000001	-1=111111111
…	…
+127= 01111111	-128=10000000

Complément à 2 (exercice)

Coder $100_{(10)}$ et $-100_{(10)}$ par complément à 2 sur 8 bits

$$100_{(10)} = 01100100_{(Cà2)}$$

$$-100_{(10)} = 10011011_{(Cà1)} + 00000001_{(Cà2)}$$
$$= 10011100_{(Cà2)}$$

Complément à 2 (exercice)

Décoder en décimal $11001001_{(Cà2)}$ et $01101101_{(Cà2)}$

$11001001_{(Cà2)} = -55_{(10)}$

car $11001001_{(Cà2)} \rightarrow 00110110_2 + 1_2 = 00110111_2 = 55_{(10)}$

$01101101_{(Cà2)} = 109_{(10)}$

Calculer : 1-2 en complément à 2 sur 8 bits

Complément à 2 (exercice)

Calculer : 1– 2 en complément à 2 sur 8 bits

$1 = 0000\ 0001_{(cà2)}$

$-2 = 1111\ 1110_{(cà2)}$

$1\text{-}2 = 1+(-2)$:

$$
\begin{array}{rl}
 & 0000\ 0001_{(cà2)} \\
+ & 1111\ 1110_{(cà2)} \\
= & 1111\ 1111_{(cà2)} = -1_{10}
\end{array}
$$

On obtient -1

Exercice de l'examen 2016/2017

1. Coder les nombres entiers du tableau suivant sur **1 octet** :

Nombre	Binaire signé	Complément à 2
$72_{(10)}$		
$0_{(10)}$		
$-96_{(10)}$		

Exercice de l'examen 2016/2017

$$72 = 64 + 8 = 2^6 + 2^3$$
$$= 01000000_2 + 00001000_2$$
$$= 01001000_{(bs)}$$
$$= 01001000_{(cà2)}$$

Exercice de l'examen 2016/2017

$96 = 3*32 \;\; = 3*2^5$

$= 11_2 * 100000_2 = 1100000_2$

$= 01100000_2$

$-96 = 11100000_{(bs)}$

$-96 = 10011111_{(cà1)} + 00000001$

$-96 = 10100000_{(cà2)}$

Exerice de l’examen 2016/2017

Nombre	Binaire signé	Complément à 2
$72_{(10)}$	01001000	01001000
$0_{(10)}$	10000000 et 00000000	00000000
$-96_{(10)}$	11100000	10100000

Fin du cours