

## Exercices : Représentation des nombres en machine

### Exercice 01 *Additions binaires*

Poser et effectuer les opérations binaires suivantes :

1.  $(1011)_2 + (101)_2$
2.  $(0101)_2 + (1101)_2$
3.  $(1010111)_2 + (1100010)_2$

### Exercice 02 *Overflow error*

1. Dans cette question, on code les entiers sur 4 bits.
  - (a) L'opération  $2 + 3$  provoque-t-elle un dépassement ?
  - (b) L'opération  $9 + 8$  provoque-t-elle un dépassement ?
2. Dans cette question, on code les entiers sur 1 octet.
  - (a) L'opération  $12 + 34$  provoque-t-elle un dépassement ?
  - (b) L'opération  $132 + 170$  provoque-t-elle un dépassement ?
3. Si on code les entiers sur  $n$  bits, quelle est la plus grande valeur possible pour la somme de deux entiers (sans dépassement) ?

### Exercice 03 *Entiers signés*

Un **entier signé** est le nom donné au type de donnée informatique permettant de représenter à la fois des nombres positifs et négatifs. Les nombres négatifs sont « marqués » par un bit de poids fort (le bit le plus à gauche) égal à 1.

1. On code les entiers sur 1 octet. Peut-on représenter 66 ? 130 ?
2. On admet que pour un entier signé codé sur  $n$  bits, on peut représenter tous les entiers compris entre  $-2^{n-1}$  (valeur minimale) et  $2^{n-1} - 1$  (valeur maximale).
  - (a) Sur 4 bits, quels sont les entiers signés que l'on peut représenter ?
  - (b) Donner les valeurs minimales et maximales des entiers signés que l'on peut former avec les différents types de variables suivants utilisés avec le langage Kotlin :

i. **Byte** (1 octet)

ii. **Short** (2 octets)

iii. **Int** (4 octets)

### Exercice 04 *Bug de l'an 2000*

Sur la majorité des systèmes UNIX, la date est stockée comme le nombre de secondes écoulées depuis le 1er janvier 1970 à minuit UTC ("Epoch Unix"). Ce nombre de secondes est appelé **timestamp**, généralement stocké sous la forme d'un entier signé sur 32 bits.

En quelle année verra-t-on un dépassement ? (considérer pour simplifier des années de 365 jours)

☞ *Plus précisément, ce sera un 19 Janvier à 3h14 du matin...*

**Exercice 05** *Complément à 2*

- Donner la représentation en complément à 2 des nombres négatifs suivants :
  - 27 (sur 1 octet)
  - 42 (sur 1 octet)
  - 1234 (sur 2 octets)
- Que représente le nombre  $(0101\ 1101)_2$  en complément à 2 ? et le nombre  $(1100\ 0101)_2$  ?

**Exercice 06** *Opérations avec des négatifs*

- Quelle est la représentation binaire de -12 sur 1 octet en complément à 2 ?
- Calculer  $23 + (-12)$  de deux façons différentes : en décimal puis en binaire.

**Exercice 07** *Inverses des puissances de 2*

- Compléter le tableau suivant :

$n$	1	2	3	4	5	6
$\frac{1}{2^n}$	$\frac{1}{2} =$ 0,5	$\frac{1}{4} =$ 0,25				
Binaire	0,1	0,01				

- Donner les valeurs décimales des nombres binaires suivants :

$$(0, 11)_2 \quad (0, 101)_2 \quad (0, 0101)_2$$

**Exercice 08** *Binaire à virgule*

Convertir en décimal :

$$(101, 001)_2 \quad (10, 101)_2 \quad (1001, 11)_2$$

**Exercice 09** *Dans l'autre sens !*

Convertir en binaire :

$$10,5 \quad 3,75 \quad 4,375$$

**Exercice 10** *L'homme plus fort que la machine*

- Essayer de convertir le nombre  $(0,1)_{10}$  en binaire. Quel est le problème ?
- Quelle séquence de bits semble se répéter dans l'écriture dyadique de  $(0,1)_{10}$  ?
- Donner les arrondis à 6 chiffres et à 11 chiffres (binaires) après la virgule de  $(0,1)_{10}$ .
- Déterminer les valeurs décimales de ces arrondis.
- Expliquer (sans calculs) pourquoi le programme Kotlin suivant affiche 0.30000000000000004 :

```

1 fun main() {
2     println(0.1 + 0.1 + 0.1)
3 }

```