

Exercices sur les chapitres D1, D11, D12, D2, D3 : représentation des nombres entiers, entiers signés et fractionnaires, norme IEEE 754, ainsi que des TP sur les bases en Python.

Partie 1

## Entiers

### 1.1 Nombre de bits nécessaires

1. Rappeler les valeurs des puissances de 2, de  $2^8$  à  $2^{10}$ .
2. Le nombre de lignes d'un fichier de l'ancienne version d'un tableur très connu était limité à 65535. Combien de bits étaient nécessaires pour stocker le numero de ligne ? Et combien d'octets ? (issu du cahier NSI p 21)

### 1.2 Numération hexadecimale (cahier NSI p 24)

1. Convertir en hexadecimale les nombres 64, 224, 264
2. Deux entiers positifs ont pour écriture en base hexadecimale A8 et 94. Quelle est l'écriture en base 16 de leur somme ?

### 1.3 Représentations

1. Comment reconnaître qu'un nombre représenté en binaire est divisible par 2 ? Par 4 ?
2. Si la représentation d'un nombre positif  $n$  en base 2 est :  $b_k \dots b_0$  quelle est la représentation de  $2 \times n$  ?

Partie 2

## Entiers signés

### 2.1 Norme du complément à 2

1. Quels sont le nombre le plus grand et le nombre le plus petit que l'on peut représenter sur 16 bits en complément à 2 ?
2. Même question sur 32 bits
3. Sur 64 bits

### 2.2 Quel est l'entier relatif codé en complément à 2 sur un octet par le code binaire 1111 1111 ?

- |     |      |     |     |
|-----|------|-----|-----|
| (1) | -127 | (2) | 127 |
| (3) | -1   | (4) | 1   |

### 2.3 Représentations

1. Quelle est la représentation binaire d'un nombre de la forme  $2^k - 1$  ?
2. Quelle est la représentation binaire sur 16 bits du nombre 2023 ? On pourra observer que  $2047 = 2^{11} - 1$
3. En déduire la représentation du complément à 2 du nombre -2023.
4. 2. Si la représentation d'un nombre signé  $n$  en complément à 2 est :  $b_k, b_{k-1} \dots b_0$  quelle est la représentation de  $2 \times n$  ?

## 2.4 Association

Associer chacun des codes binaires suivants en complément à deux sur 8 bits au nombre qu'il représente :

(1)	00001111	*	*	(a)	-127
(2)	10000001	*	*	(b)	-86
(3)	11110000	*	*	(c)	-16
(4)	01010101	*	*	(d)	-15
(5)	11110001	*	*	(e)	15
(6)	10101010	*	*	(f)	85

## 2.5 Débordement

Quelles additions des nombres suivants provoquent un dépassement de capacité lorsque l'on utilise un codage sur 8 bits ?

1. 111 - 240
2. 113 + 15
3. 112 - 240
4. 112 + 15
5. -128 + 128
6. 256 - 200
7. -113 - 15

Partie 3

# Nombres fractionnaires

## 3.1 Conversions

1. Convertir  $3,375_{10}$  en binaire (cahier NSI p 23)
2. Association

Associer le code binaire de la partie décimale d'un nombre fractionnaire représenté par un codage à virgule fixe utilisant 8 bits de partie fractionnaire avec le nombre correspondant.

(1)	1000 0000	*	*	(a)	0,9375
(2)	1111 0000	*	*	(b)	0,875
(3)	0101 0000	*	*	(c)	0,5
(4)	1110 0000	*	*	(d)	0,375
(5)	0110 0000	*	*	(e)	0,3125

## 3.2 Norme IEEE 754 simplifiée

voir l'exercice en ligne sur la page *codage des nombres* du site allophysique

Partie 4

## Algorithmes

4.1 Ex 4.1 : Utiliser l'algo de multiplication et suivre l'évolution des variables pour vérifier que celui-ci réalise bien la multiplication de 7 par 4

Rappel de l'algorithme de multiplication

```

1 r <- 0
2 faire b fois:
3     r <- r + a
4 # le resultat est r

```

avancée	a	b	r	itération n°
debut	7	4	0	avant iteration
dans la boucle	7	4	7	fin de la 1ere

4.2 Ex 4.2 : Utiliser l'algorithme de division pour a = 39 et b = 8

Rappel de l'algorithme de division :

```

1 r <- a
2 i <- 0
3 tant que r >= b, faire:
4     r <- r - b
5     i <- i + 1
6 # le resultat est i

```

avancée	a	b	r	i
debut	39	8	39	0
dans la boucle (1ere it)	39	8	31	1

...

Partie 5

## Corrections

Ex 4.1 :

avancée	a	b	r	itération n°
debut	7	4	0	avant iteration
dans la boucle	7	4	7	fin de la 1ere
	7	4	14	2
	7	4	21	3
	7	4	28	4

le résultat est 28.

Ex 4.2 :

avancée	a	b	r	i
debut	39	8	39	0
dans la boucle (1ere it)	39	8	31	1
dans la boucle (2)	39	8	23	2
dans la boucle (3)	39	8	15	3
dans la boucle (4)	39	8	7	4