Représentation des nombres réels

« En mathématiques, un nombre réel est un nombre qui peut être représenté par une partie entière munie d'un signe positif ou négatif, et une liste finie ou infinie de décimales »

in `https://fr.wikipedia.org/wiki/Nombre_r%C3%A9el`

. . . .

Capacités attendues

- ✓ Représentation approximative des nombres réels : notion de nombre flottant ;
- ✓ Calculer sur quelques exemples la représentation de nombres réels : 0.1, 0.25 ou 1/3 ;
- ✓ 0.2 + 0.1 n'est pas égal à 0.3. Il faut éviter de tester l'égalité de deux flottants ;
- ✓ Aucune connaissance précise de la norme IEEE-754 n'est exigible.

. . . .

1 Nombres à virgule

Tout comme dans le système de numération décimal, le système binaire autorise la présence d'une virgule.

Par exemple : $1001,1011_{\,2} = 2^3 + 2^0 + 2^{-1} + 2^{-3} + 2^{-4} = 8 + 1 + \frac{1}{2} + \frac{1}{8} + \frac{1}{16} = 9,6875_{\,10}$

Réciproquement, pour déterminer l'écriture binaire d'un nombre :

- on écrit la partie entière en binaire ;
- les chiffres de la partie fractionnaire s'obtiennent par des divisions par $\frac{1}{2}$, donc par des multiplications par 2.

Explication sur un exemple : soit à convertir en binaire le nombre $9,6875_{\,10}$.

On a d'une part $9_{\,10} = 1001_{\,2}$, et d'autre part :

$$\begin{array}{rcccl} 0,6875 & \times & 2 & = & \mathbf{1} + 0,375 \\ 0,375 & \times & 2 & = & \mathbf{0} + 0,75 \\ 0,75 & \times & 2 & = & \mathbf{1} + 0,5 \\ 0,5 & \times & 2 & = & \mathbf{1} + 0 \end{array}$$

On retrouve bien que $9,6875_{\,10} = 1001,1011_{\,2}$.

Exercice 1

Déterminer l'écriture binaire des nombres $5,84375_{\,10}$ et $0,1_{\,10}$.

Réponse : $5,84375_{\,10} = 101,11011_{\,2}$ **et** $0,1_{\,10} = 0,0001\,1001\,1001\,1001\,1001\ldots_{\,2}$.

On démontre que l'écriture binaire d'un nombre x ne se termine que si x est une fraction dont le dénominateur est une puissance de 2. Il s'agit du même phénomène que dans l'écriture décimale de $\frac{1}{3} = 0,333\ldots$

2 Les flottants

Si on souhaite réaliser des calculs exacts avec des nombres décimaux, il suffit de fixer une précision (par exemple 10^{-4}) et d'utiliser des entiers. Par exemple, $1,234 = 12340 \times 10^{-4}$: on a simplement changé d'unité, on compte en dix-millèmes. On parle de représentation des nombres réels à virgule fixe , le nombre de décimales après la virgule étant fixé.

S'il est utile dans certaines situations (précision des calculs financiers), l'inconvénient de ce procédé est qu'il ne permet pas de représenter à la fois des nombres très grands (le nombre d'Avogadro $\approx 6 \times 10^{23}$) et très petits (la constante de Planck $\approx 6,6 \times 10^{-34}$).

La solution à ce problème consiste à faire varier la virgule en fonction des besoins, à l'aide d'un exposant comme dans la notation scientifique : c'est la représentation des nombres réels en virgule flottante.

Exercice 2

Déterminer la notation scientifique de $123\,000\,000$ et $-0,000\,321$.

Réponse : $123\,000\,000 = 1,23 \times 10^8$ et $-0,000\,321 = -3,21 \times 10^{-4}$.

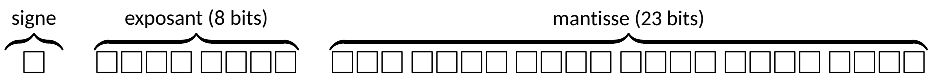
> **Définition**
> Dans la notation scientifique d'un nombre, $\pm m \times 10^e$:
> $\pm$ est le signe ; e s'appelle l'exposant . m s'appelle la mantisse .

Exercice 3

1. Écrire le nombre $14,25$ sous la forme $m \times 2^e$ où m est un nombre réel compris dans l'intervalle $[1;2[$.

   Réponse : $14,25 = 1,78125 \times 2^3$ donc $m = 1,78125$ et $e = 3$

2. Déterminer l'écriture binaire de : $e + 127 =$ $1000\,0010$ et $m =$ $1,11001$.

> **Définition**
> La norme IEEE 754 spécifie deux formats en virgule flottante en base 2. Sur 32 bits (simple précision), la structure est la suivante (chaque ☐ représente un bit) :
>
> 
>
>
> Si un nombre réel a, en base 2, pour notation scientifique $\pm m \times 2^e$:
> - le signe est codé par 0 si le nombre est positif, 1 sinon ;
> - l'exposant codé est l'écriture binaire de $e + 2^7 - 1 = e + 127$ (biais égal à 127);
> - la mantisse codée est l'écriture binaire de m, privée du 1 devant la virgule et tronquée si nécessaire.

Exercice 4

Coder le nombre décimal 128 selon la norme IEEE 754 sur 32 bits (simple précision).

Réponse : $0\ 1000\ 0110\ 000\ 0000\ 0000\ 0000\ 0000\ 0000$

Exercice 5

En utilisant les résultats de l'exercice 3, coder le nombre décimal $14,25$ selon la norme IEEE 754 sur 32 bits.

Réponse : $0\ 1000\ 0010\ 110\ 0100\ 0000\ 0000\ 0000\ 0000$

Exercice 6

Donner la valeur décimale des nombres flottants suivants codés sur 32 bits :

a) $1\,0111\,1110\,11110000000000000000000$ Réponse : $-0,96875$

b) $0\,1000\,0011\,11100000000000000000000$ Réponse : 30

Exercice 7

On considère le script Python ci-dessous.

```python
x = 10

while x != 0 :
    x = x - 0.1
    print(x)
```

1. Ce script ne se termine pas : pourquoi ?
2. Comment le modifier pour qu'il se termine ?

Exercice 8

Comparer et expliquer la différences d'affichage entre les instructions `print(2**1024)` et `print(2.0**1024)`.

Valeurs spéciales

- Le nombre zéro est représenté par une mantisse et un exposant tous à zéro.
  Selon la valeur du bit de poids fort, on a donc deux codages de zéro : $+0$ et -0.
- Si l'exposant est $1111\,1111$ et la mantisse à 0, la valeur représentée correspond à $\pm\infty$ selon le signe. Ces « nombres » sont notés `inf` et `-inf` en Python.
- Si l'exposant est $1111\,1111$ et la mantisse est non nulle, la valeur représentée est « Not a Number », notée `nan` en Python : cette valeur est parfois renvoyée lors de calculs impossibles.

À retenir

- Les nombres flottants sont une représentation approximative des nombres réels dans un ordinateur ;
- Une norme internationale (IEEE 754) définit un encodage en simple ou double précision (32 ou 64 bits) ;
- Les opérations arithmétiques sur les nombres flottants n'ont pas toujours les mêmes propriétés que ces mêmes opérations sur les réels.