LEÇON 22 : ENCADREMENT PAR DES NOMBRES DÉCIMAUX

Rappel : un nombre décimal est une fraction dont le dénominateur est une puissance de dix : $14,25 = \frac{1425}{100}$.

① Troncatures

Avec les parties entières on a vu que pour tout réel x on a $\lfloor x \rfloor \leqslant x < \lfloor x \rfloor + 1$; généralisons ceci. Soit $x \in \mathbb{R}$ et soit $n \in \mathbb{N}$. On applique la propriété à $10^n \times x$:

$$\lfloor 10^n \times x \rfloor \leqslant 10^n \times x < \lfloor 10^n \times x \rfloor + 1 \overset{\div 10^n}{\Longleftrightarrow} \frac{\lfloor 10^n \times x \rfloor}{10^n} \leqslant x < \frac{\lfloor 10^n \times x \rfloor}{10^n} + \frac{1}{10^n}.$$

Ceci s'appelle la troncature (à n chiffres après la virgule) de x.

Ex : $\pi = 3,1415926538\ldots$

$n=0$: $\frac{\lfloor 10^0 \times \pi \rfloor}{10^0} = \frac{\lfloor 1 \times \pi \rfloor}{1} = 3$

$n=0$ chiffre après la virgule

$n=1$: $\frac{\lfloor 10^1 \times \pi \rfloor}{10^1} = \frac{\lfloor 10 \times \pi \rfloor}{10} = \frac{\lfloor 31,41\ldots \rfloor}{10} = \frac{31}{10} = 3,\underset{\smile}{1}$

$n=1$ chiffre après la virgule

$n=6$: $\frac{\lfloor 10^6 \times \pi \rfloor}{10^6} = \frac{\lfloor 3141592,653\ldots \rfloor}{1000\,000} = \frac{3\,141\,592}{1\,000\,000} = 3,\underbrace{141\,592}.$

$n=6$ chiffres après la virgule

② Cas des nombres rationnels

On prend $x = \frac{a}{b}$ avec $a \in \mathbb{Z}$ et $b \in \mathbb{N}^*$. On rappelle que $\left\lfloor 10^n \times \frac{a}{b} \right\rfloor = \left\lfloor \frac{10^n \times a}{b} \right\rfloor$ est le quotient dans la division euclidienne de $10^n \times a$ par b. En Python, il se note `10**n *a // b`.

quotient de la division euclidienne.

Ex : essayons avec $\frac{a}{b} = \frac{355}{113}$ et $n = 6$. Qu'en dit Python ?

```
>>> 10 ** 6 * 355 // 113
3141592
```

Reste à insérer un « , » entre 3 et 141592 pour obtenir la troncature $\frac{\left\lfloor 10^6 \times \frac{355}{113} \right\rfloor}{10^6}$.

Problème : Python (ou la calculatrice) ne peut pas utiliser des nombres à virgule ayant plus de seize chiffres ; donc on ne peut pas brutalement diviser par 10^n. En revanche on peut utiliser des entiers ou des chaînes de caractères arbitrairement longs.

```
def Troncature(a, b, n):
    q = 10**n * a // b
    (e, f) = divmod(q, 10**n)
    E = str(e) ; F = str(f)
    print(E + "." + "0"*(n - len(F)) + F)
```

divmod : effectue la division euclidienne de q par 10^n. Donc e = 3 et f = 14159292035... dans l'exemple ci dessous.

str : convertit un entier en chaîne.

On affiche E (la partie entière), un « . », éventuellement des 0, puis F (la partie fractionnaire).

```
>>> Troncature(355, 113, 50)
3.14159292035398230088495575221238938053097345132743
```

③ Un exemple plus sophistiqué : $\sqrt{2}$

Petit calcul : $(a-b\sqrt{2})^2 = a^2 - 2ab\sqrt{2} + b^2\times 2 = (a^2+2b^2) - 2ab\sqrt{2}$.

On part de $(a,b) = (-1,-1)$ et on effectue K fois la transformation

$$(a,b) \longmapsto (a^2+2b^2,\ 2ab).$$

Ceci permet d'obtenir $\underbrace{\left(\left(\left((x^2)^2\right)^2\cdots\right)^2\right.}_{K\text{ fois}} = x^{2^K}$ avec $x = \underbrace{-1}_{a=-1} - \underbrace{(-1)}_{b=-1}\sqrt{2} = \underbrace{-1+\sqrt{2}}_{\simeq 0,414\ldots}$.

Or $0 \leq x < \frac{1}{2} \Rightarrow 0 \leq x^{2^K} < \frac{1}{2^{2^K}}$

et $0 \leq a - b\sqrt{2} < \frac{1}{2^{2^K}} \overset{(-a)}{\Longleftrightarrow} -a \leq -b\sqrt{2} < -a + \frac{1}{2^{2^K}}$

$\underset{\text{si } -b \text{ est négatif}}{\overset{\div(-b)}{\Longleftrightarrow}} \frac{a}{b} \geqslant \sqrt{2} > \frac{a}{b} - \frac{1}{b\times 2^{2^K}}$

donc $\frac{a}{b}$ est une valeur approchée à $\frac{1}{b\times 2^{2^K}}$ près de $\sqrt{2}$.

```
def RacineDeDeux(K):
    (a, b) = (-1, -1)
    for i in range(K):
        (a, b) = (a**2 + 2*b**2, 2*a*b)
    n = len(str(b * 2**2**K)) - 1
    Troncature(a, b, n)
```

for : permet de répéter K fois la transformation

on compte le nombre de chiffres qu'on est sûr d'avoir justes

on affiche le résultat avec le programme précédent

```
>>> RacineDeDeux(6)
1.4142135623730950488016887242096980785696718
```