Correction du DS 1

Informatique pour tous, première année

Julien Reichert

Exercice 1

Convertissons 1912 en base 5.

Première méthode : $1912 = 119 \times 16 + 8$; $119 = 7 \times 16 + 7$; 7 est le dernier reste. La réponse est donc $\overline{778}^{16}$ en reprenant les restes de droite à gauche.

Deuxième méthode : $1912 = 7 \times 16^2 + 120$; $120 = 7 \times 16 + 8$; 8 est le chiffre des unités. La réponse est la même en écrivant les coefficients de gauche à droite.

Exercice 2

Pour rappel, l'écriture en virgule flottante sur 16 bits utilise un bit de signe, puis 5 bits d'exposant et finalement 10 bits de mantisse.

Il n'y a pas de raison de travailler sur $\frac{1}{9}$ autrement qu'en tant que fraction, ce qui facilite les calculs sans entraver les comparaisons à 1. On sait qu'en binaire notre nombre commence par $0, \ldots$ puisqu'il s'agit de la partie entière en décimal aussi.

Procédons donc aux multiplications par 2 successives.

$$\begin{aligned}
2\times \tfrac{1}{9} &= \mathbf{0} + \tfrac{2}{9} \\
2\times \tfrac{2}{9} &= \mathbf{0} + \tfrac{4}{9} \\
2\times \tfrac{4}{9} &= \mathbf{0} + \tfrac{8}{9} \\
2\times \tfrac{8}{9} &= \mathbf{1} + \tfrac{7}{9} \\
2\times \tfrac{7}{9} &= \mathbf{1} + \tfrac{5}{9} \\
2\times \tfrac{5}{9} &= \mathbf{1} + \tfrac{1}{9}
\end{aligned}$$

À ce stade, on a trouvé un cycle, et on arrête le calcul.

Une remarque pour les adeptes de l'arithmétique modulaire : 9 n'étant pas premier, la période du cycle est plus difficile à déterminer, mais on peut s'intéresser pour le plaisir à la puissance strictement positive minimale k telle que $x^k = 1$ modulo 9 pour x non multiple de 3. Cette puissance étant visiblement 6 pour 2, on déduit qu'elle est 3 pour 4 et 2 pour 8 (ce qui est normal vu que 8 est congru à -1 modulo 9) ; pour 5, elle est 6 et pour 7 elle est 3.

Une autre façon de retrouver ce cycle est de chercher une suite géométrique de raison une puissance de $\frac{1}{2}$ dont la somme fait $\frac{1}{9}$. Il s'avère qu'on a les mêmes puissances qui entrent en jeu. Précisément : $\sum_{i=0}^{\infty} \frac{1}{64^i} = \frac{1}{1-\frac{1}{64}} = \frac{64}{63}$.

Ayant trouvé une puissance de 2 telle qu'un de moins soit un multiple de 9 (d'où la correspondance des puissances), il suffit de trouver un premier terme adéquat, qui sera nécessairement une fraction dont le dénominateur sera 2 à la puissance en question et le numérateur sera positif et inférieur au dénominateur, soit ici $\frac{7}{64}$, car $\frac{7}{64} \times \frac{64}{63} = \frac{1}{9}$. Or, $\frac{7}{64}$ s'écrit exactement $\overline{0,000111}^2$, et en répétant la partie après la virgule, on forme bien la suite géométrique recherchée.

Quoi qu'il en soit, $\frac{1}{9}$ s'écrit en écriture scientifique binaire $1,11000111000\cdots \times 2^{-4}$.

L'exposant -4 se représente sur 5 bits comme $-4 + 2^{5-1} - 1$, soit $\overline{01011}^2$. L'écriture de la mantisse en virgule flottante omettant le 1 avant la virgule, on a donc les bits 1100011100 car, le bit suivant étant un 0, l'arrondi se fait par défaut.

La représentation finale est alors 0 01011 1100011100.

Exercice 3

Le plus petit nombre strictement positif représentable en virgule flottante (hors valeurs exceptionnelles) sur 32 bits s'écrit 0 00000001 $0\ldots0$ et correspond à 2^{-126}, le plus grand nombre s'écrit 0 11111110 $1\ldots1$ et correspond à $2^{127} \times (1 + 2^{-1} + 2^{-2} + \cdots + 2^{-23}) = 2^{127} \times (2 - 2^{-23})$. Le produit des deux fait donc $2 \times (2 - 2^{-23}) = 4 - 2^{-22}$.

Exercice 4

Entre autres possibilités, on retiendra la plus classique :

```
TP4exo5 = []
for i in range(11):
    for j in range(i,-1,-1):
        TP4exo5.append(j)
```

... et la plus courte :

```
l = [j for i in range(11) for j in range(i,-1,-1)]
```

Créer une fonction revient à reprendre ce code et à l'indenter, en utilisant également le mot-clé `return`. Pour le code court, cela donne :

```
def compte_a_rebours_10():
    return [j for i in range(11) for j in range(i,-1,-1)]
```

Et pour terminer, si on paramètre les comptes à rebours, il suffit de changer le `11` en la valeur souhaitée :

```
def compte_a_rebours(n):
    return [j for i in range(n+1) for j in range(i,-1,-1)]
```