Corrigé du Devoir à la Maison n°2

Partie A. Des chiffres au nombre

1. Le chiffre a_p est en position 0 dans la liste L, le chiffre a_{p-1} est en position 1, etc jusqu'au chiffre a_0 qui est en position $m-1$ où m est la longueur de la liste L.
   Ainsi $m = p+1$ et a_k est en position $m-1-k$ dans la liste L.

```
1 def Nombre(L,b=10):
2     """Donne le nombre dont les chiffres en base b (décimale par
3     défaut) sont les composantes de la liste L."""
4     n=0
5     m=len(L)
6     for k in range(len(L)):
7         n=n+L[m-1-k]*b**k
8     return n
```

2. Soit $C(m)$ le coût en temps de la fonction ci-dessus. On va constater qu'il dépend de m, la longueur de la liste d'entrée L.
   Les opérations sont
   - deux affectations lignes `4` et `5`
   - une affectation ligne `6`
   - une affectation, une addition, deux soustractions, une multiplication, et une puissance k, soit $5+(k-1) = k+4$ opérations ligne `7`.

   Les lignes `6` et `7` sont parcourues pour k allant de 0 à $m-1$ donc le coût est

$$C(m) = 2 + \sum_{k=0}^{m-1}(k+5) = 2 + \sum_{k=0}^{m-1} k + \sum_{k=0}^{m-1} 5 = 2 + \frac{m(m-1)}{2} + 5m$$

   Soit finalement :

$$C(m) = \frac{m^2}{2} + \frac{9m}{2} + 2$$

   Ainsi $C(m) = O(m^2)$, il s'agit d'une complexité quadratique.

3. Pour tout $i = 0 \dots p+1$ on note $\mathcal{P}_i$ la propriété : $u_i = \sum_{j=0}^{i-1} b^{i-1-j} x_j$.
   On démontre par récurrence finie que cette propriété est vraie pour tout $i = 0 \dots p+1$.
   Initialisation. La propriété $\mathcal{P}_0$ est vraie car elle s'écrit $u_0 = 0$. En effet u_0 est une somme vide.
   Hérédité. Supposons que pour un certain $i \in \{0, \dots p\}$ la propriété $\mathcal{P}_i$ est vraie.
   Ceci donne

$$u_{i+1} = bu_i + x_i = b\left(\sum_{j=0}^{i-1} b^{i-1-j} x_j\right) + x_i = \sum_{j=0}^{i-1} b^{i-j} x_j \ + x_i$$

Si $j = i$ alors $b^{i-j}x_j = x_i$ donc $u_{i+1} = \sum_{j=0}^{i} b^{i-j}x_j$. La propriété $\mathcal{P}_{i+1}$ est vraie.

L'hérédité est prouvée : pour tout $i = 0 \dots p$, si $\mathcal{P}_i$ est vraie alors $\mathcal{P}_{i+1}$ est vraie.

Conclusion. Par récurrence finie la propriété $\mathcal{P}_i$ est vraie pour tout $i = 0 \dots p+1$.

En particulier pour $i = p+1$ on obtient $u_{p+1} = \sum_{j=0}^{p} b^{p-j}x_j$.

Si $(x_0, \dots, x_p) = (a_p, \dots, a_0)$ alors $x_j = a_{p-j}$ pour tout $j = 0 \dots p$ donc :

$$u_{p+1} = \sum_{j=0}^{p} b^{p-j}a_{p-j}$$

Le changement d'indice $k = p - j$ donne $u_{p+1} = \sum_{k=0}^{p} b^k a_k$.

On retrouve la formule $(*)$, donc u_p est le nombre dont les chiffres en base b sont $a_p, \dots, a_0$.

4.

```
def Nombre(L,b=10):
    """Donne le nombre dont les chiffres en base b (décimale par défaut)
    sont les composantes de la liste L."""
    n=0
    for x in L:
        n=b*n+x
    return n
```

5. Soit $D(m)$ le coût en temps de la fonction définie ci-dessus, où m est la longueur de la liste L. On obtient :

$$D(m) = 1 + m * (1 + 3) = 4m + 1$$

On constate que $D(m) = O(m)$, il s'agit d'une complexité linéaire.

Ce nouvel algorithme est donc plus rapide que le précédent.

Remarque. Si on avait compté la complexité du premier algorithme en considérant que le coût en temps de calcul de x^k est logarithmique en k (par exemple grâce à l'exponentiation rapide), alors on aurait obtenu $C(n) = O(n \ln n)$, ce qui est aussi supérieur à une complexité linéaire.

Partie B. Du nombre aux chiffres

6.

```
def Chiffres(n,b=10):
    """Donne la liste des chiffres de n en base b décimale par défaut"""
    L=[]
    m=n
    while m>0:
        m,r=m//b,m%b
        L=[r]+L
    return L
```

On utilisera ci-dessous les numéros de lignes de l'algorithme en pseudo-code de l'énoncé, et non ceux de la fonction ci-dessus.

7. D'après la ligne `5` :

$$m_{k+1} = \left\lfloor \frac{m_k}{b} \right\rfloor$$

Tant que l'algorithme n'est pas fini m_k est strictement positif, et comme $b > 1$ alors :

$$m_{k+1} \leqslant \frac{m_k}{b} < m_k$$

La suite (m_k) est donc strictement décroissante.

De plus les m_k sont entiers car ce sont des quotients de divisions euclidiennes.

La suite (m_k) est une suite d'entier strictement décroissante, donc elle finit par prendre une valeur négative ou nulle.

Ceci montre que la boucle n'est pas infinie, puisque la condition d'arrêt est $m \leqslant 0$.

On peut ajouter que la suite (m_k) ne prend que des valeurs positives, donc il existe un entier q minimal tel que $m_q = 0$.

Cet entier est le nombre d'itérations de la boucle.

8. Les lignes `4` et `5` montrent que r_{k+1} et m_{k+1} sont le reste et le quotient de la division euclidienne de m_k par b, ce qui s'écrit :

$$m_k = bm_{k+1} + r_{k+1} \qquad \text{et} \qquad 0 \leqslant r_{k+1} < b$$

9. On calcule pour tout $k = 0 \dots q-1$:

$$\begin{aligned} v_{k+1} - v_k &= b^{k+1}m_{k+1} + \sum_{i=1}^{k+1} r_i b^{i-1} - b^k m_k - \sum_{i=1}^{k} r_i b^{i-1} \\ &= b^{k+1}m_{k+1} + r_{k+1}b^k - b^k m_k \\ &= b^k(bm_{k+1} + r_{k+1} - m_k) \end{aligned}$$

D'après la question précédente $v_{k+1} - v_k = 0$. Ceci montre que la suite $(v_k)_{0\leqslant k\leqslant q}$ est constante : v_k est un invariant de boucle.

10. On sait que $m_q = 0$, donc pour $k = q$ on obtient $v_q = \sum_{i=1}^{q} r_i b^{i-1}$.

Comme la suite (v_k) est constante alors $v_q = v_0$. Pour $k = 0$ on obtient $v_0 = m_0$, ce qui donne $v_0 = n$, puis $v_q = n$.

Ainsi $n = \sum_{i=1}^{q} r_i b^{i-1}$. Le changement d'indice $k = i - 1$ donne $n = \sum_{k=0}^{q-1} r_{k+1} b^k$.

À chaque itération la valeur de r est ajoutée à l'avant de la liste L, donc celle-ci prend les valeurs $[r_1]$, puis $[r_2, r_1]$, etc jusqu'à $[r_q, \dots, r_1]$.

Les r_k sont des restes de divisions euclidiennes par b, donc ce sont des entiers compris entre 0 et $b-1$.

SI on note $a_k = r_{k+1}$ alors $n = \sum_{k=0}^{q-1} a_k b^k$ et la liste renvoyée est $[a_{q-1}, \dots a_0]$.

Ceci montre que la liste renvoyée est la liste des chiffres de n en base b.

11. Dans la question 8 on a établi que pour tout $k = 0 \dots q-1$: $m_k = bm_{k+1} + r_{k+1}$.
Pour $k = q-1$ on obtient :

$$m_{q-1} = bm_q + r_q$$

Comme m_q est le premier terme nul de la suite (m_k), alors $m_q = 0$ et $m_{q-1} \neq 0$, ce qui donne $m_{q-1} = r_q$ puis montre que r_q est non-nul.
On considère ensuite l'égalité :

$$n = \sum_{k=0}^{q-1} r_{k+1} b^k$$

Comme les r_k et b sont positifs, alors cette somme est plus grande que son dernier terme :

$$n = \sum_{k=0}^{q-2} r_{k+1} b^k + r_q b^{q-1} \geqslant r_q b^{q-1} \geqslant b^{q-1}$$

En effet, comme $r_q > 0$ alors $r_q \geqslant 1$.
Ensuite, chaque r_k est inférieur ou égal à $b-1$, donc

$$n = \sum_{k=0}^{q-1} r_{k+1} b^k \leqslant \sum_{k=0}^{q-1} (b-1) b^k = \sum_{k=0}^{q-1} b^{k+1} - \sum_{k=0}^{q-1} b^k = \sum_{k=1}^{q} b^k - \sum_{k=0}^{q-1} b^k = b^q - 1$$

On a démontré que $b^{q-1} \leqslant n \leqslant b^q - 1$, ce qui donne :

$$b^{q-1} \leqslant n < b^q$$

Par application de la fonction logarithem népérien, qui est strictement croissante :

$$(q-1)\ln b \leqslant \ln n < q \ln b$$

On divise par $\ln b$, qui est strictement positif car $b \geqslant 2$:

$$q - 1 \leqslant \frac{\ln n}{\ln b} < q$$

Ceci montre que $q-1$ est la partie entière de $\frac{\ln n}{\ln b}$, puis que :

$$q = \left\lfloor \frac{\ln n}{\ln b} \right\rfloor + 1$$

12. Soit E le coût en temps de l'algorithme.
Après deux affectations ligne `2`, la boucle **Tant que** est itérée q fois. Chaque itération contient 7 opérations lignes `3` à `6`. En ajoutant le test de sortie de boucle on obtient :

$$E(q) = 3 + 7q$$

D'après la question précédente ceci donne :

$$E(n) = 10 + 7\left\lfloor \frac{\ln n}{\ln b} \right\rfloor$$

Comme b est une constante, alors $E(n) = O(\ln n)$.
Il s'agit d'une complexité logarithmique en n.