

## D) Procédé de Kaprekar... (16 points)

Les exercices de cette partie sont indépendants. Le dernier exercice utilise les fonctions des exercices précédents, mais il n'est pas obligatoire de les avoir écrites.

Pour toutes les fonctions, on supposera que les paramètres sont valides : il n'y a pas de test à faire, pas d'exception à déclencher.

### 1) Test ... (1 point)

Soit la fonction **test** suivante :

```
1 def test(x, L):
2     i = len(L) - 1
3     while i >= 0 and L[i] != x:
4         i = i - 1
5     return i >= 0
```

Que fait cette fonction ?

### 2) Entiers <-> liste ... (6 points)

- a) Écrire la fonction **int\_to\_list(n,p)** qui convertit le nombre **n** (entier naturel à au plus **p** chiffres) en une liste de ses chiffres éventuellement complétés par des 0 pour atteindre **p** chiffres.

*Exemples d'applications (l'ordre des chiffres dans le résultat importe peu) :*

```
>>> int_to_list(27972,5)
[2, 7, 9, 7, 2]

>>> int_to_list(42,5)
[2, 4, 0, 0, 0]

>>> int_to_list(258,4)
[8, 5, 2, 0]
```

- b) Écrire la fonction **list\_to\_ints(L)** qui, à partir de la liste **L** non vide ne contenant que des chiffres (de 0 à 9), retourne le couple (*Left*, *right*), avec :

- *left* le nombre construit avec les chiffres de **L** lus de gauche à droite,
- *right* le nombre construit avec les chiffres de **L** lus de droite à gauche.

Exemples d'applications :

```
>>> list_to_ints([1,2,3,4,5,6])
(123456, 654321)

>>> list_to_ints([2,4,0,0,0,])
(24000, 42)
```

### 3) Histogramme et tri ... (4 points)

Nous travaillons ici avec des listes qui ne contiennent que des chiffres (de 0 à 9).

- a) Écrire la fonction **hist(L)** qui retourne un histogramme (sous la forme d'une liste) des chiffres présents dans la liste **L**.

Rappel : l'histogramme  $H$  est une liste telle que  $H[i]$  est le nombre d'occurrences de la valeur  $i$ . Par exemple dans la première application ci-dessous, il y a 5 valeurs 0, 3 valeurs 1, 3 valeurs 2, ...

Exemples d'applications :

```
>>> hist([1,5,9,3,0,1,2,0,1,0,2,5,0,5,2,0])
[5, 3, 3, 1, 0, 3, 0, 0, 0, 1]

>>> hist([1,0,1,0,1,0,1,0,1])
[4, 5, 0, 0, 0, 0, 0, 0, 0]
```

- b) Utiliser la fonction **hist** pour écrire la fonction **sort(L)** qui trie la liste **L** en ordre croissant (la fonction construit une nouvelle liste qui doit être retournée).

Exemples d'application :

```
>>> sort([1,5,9,3,0,1,2,0,1,0,2,5,0,5,2,0])
[0, 0, 0, 0, 0, 1, 1, 1, 2, 2, 2, 3, 5, 5, 5, 9]
```

### 4) Kaprekar ... (5 points)

Procédé de Kaprekar :

Soit un entier  $n$  à  $p$  chiffres. Par exemple  $n = 6264$ .

- Prendre les  $p$  chiffres de  $n$  pour former deux nombres : le plus grand et le plus petit : 6642 et 2466,
- Les soustraire en complétant éventuellement par des 0 à gauche pour obtenir un nouveau nombre à  $p$  chiffres :  $n = 6642 - 2466 = 4176$ ,
- Recommencer le procédé avec le résultat :  $4176 \rightarrow 7641 - 1467 = 6174$ .

Le procédé de Kaprekar peut être utilisé avec un nombre quelconque. Dans tous les cas, quel que soit le nombre de chiffres, **on arrivera à une valeur déjà rencontrée** (dans les exemples donnés plus loin : on retrouve 6174 dans le premier, et 63 dans le deuxième), qui sera la valeur 0 si tous les chiffres sont identiques.

**Remarque :**

Nous travaillons ici toujours avec  $p$  chiffres. Cela signifie que si un résultat intermédiaire est inférieur à  $10^{p-1}$  (par exemple 999 lorsque  $p = 4$ ), les deux nombres seront construits à partir des chiffres du nombre complétés par des 0 (ici 999 et 9990).

La fonction à écrire **kaprekar( $n, p$ )** prend en paramètre un entier naturel  $n$  et son nombre de chiffres  $p$  et applique le procédé de Kaprekar. Elle construit et retourne la liste les différentes valeurs calculées  $c$ , comme dans les exemples suivants.

*Exemples d'applications :*

```
>>> kaprekar(1574,4)
[1574, 6084, 8172, 7443, 3996, 6264, 4176, 6174, 6174]
// 6174 rencontré deux fois

>>> kaprekar(42,2)
[42, 18, 63, 27, 45, 9, 81, 63] // 63 rencontré deux fois

>>> kaprekar(666,3)
[666, 0, 0] // 0 rencontré deux fois
```

Vous pouvez utiliser les fonctions des questions 1 à 3 de cet exercice, même si vous ne les avez pas écrites.

Fin de l'énoncé