Lycée Louis-Le-Grand, Paris
MPSI 4 – Informatique commune
G. Dewaele, A. Troesch

2015/2016

TP n° 5 : Manipulations de listes

Pour tout le TP, on rappelle qu'on accède à l'élément d'indice `i` d'une liste `l` par `l[i]`. On rappelle également qu'on peut extraire des tranches : `l[i:j]` extrait la liste des éléments d'indice i (inclus) à j (exclus). On peut rajouter un pas en 3e paramètre : `l[i:j:p]`. On peut aussi se servir d'une indexation négative permettant d'accèder d'abord aux derniers éléments. Par exemple `l[-1]` est le dernier élément de la liste.

Exercice 1 – Manipulations élémentaires de listes

Les questions suivantes sont à faire dans l'interpréteur Python.

1. Créer la liste `l=[1,2,3,4,5]`, et extraire le terme d'indice 4. Que remarquez-vous ?
2. Redéfinissez le terme d'indice 4 comme étant la somme des deux précédents.
3. Essayez de sommer, de multiplier deux listes entre elles ; de sommer, de multiplier une liste par un entier. Observations ?
4. Comparer l'effet sur l'adressage mémoire des opérations `l = l + [1]`, `l += [1]` et `l.append(1)`.
5. À l'aide des fonctions `random` et `sample` du module `random`, dont on pourra consulter la page d'aide, générer une liste `li` de longueur aléatoire comprise entre 10 et 10000, et contituée d'entiers distincts choisis au hasard entre 1 et 10000.
6. Extraire le dernier terme de la liste `li`, sans modifier la liste, et sans chercher à déterminer la longueur de la liste.
7. Utiliser la fonction `help` (pour cette question et celles qui suivent) pour trouver comment déterminer la longueur de la liste `li`.
8. Déterminer le rang de l'éventuelle occurrence de 1. S'il n'y en a pas, rajouter l'élément en fin de liste. Déplacer la valeur 1 en début de liste.
9. Stockez dans une variable x la valeur d'indice 5 de la liste, et supprimez cette valeur du tableau (en une seule instruction).
10. (a) À l'aide des méthodes associées aux listes, triez le tableau en ordre décroissant.
    (b) Triez le tableau dans l'ordre croissant des sinus de ses éléments.
    (c) Triez le tableau dans l'ordre décroissant des valeurs de $x^2 - x$.
11. (a) À l'aide de la fonction `sum`, déterminer la somme des éléments de la liste `li`
    (b) Calculer la somme des carrés des éléments du tableau.

Exercice 2 – Techniques de slicing sur les listes

On rappelle (voir le cours) qu'étant donné une liste `li`, `li[m:n]` *renvoie un tableau contitué des termes d'indices m à $n-1$ du tableau* `li`. *Les questions suivantes sont à faire dans l'interpréteur Python.*

1. (a) Définir une liste `liste` d'entiers, de longueur 10.
   (b) Extraire la tranche `[2:6]`, visualisez le résultat, triez cette tranche par ordre décroissant, replacez-la dans la liste initiale. Ce faisant, contrôlez l'effet de ces opérations sur les adresses.
2. (a) Insérez la liste `liste` comme élément d'elle-même, à l'indice 3.
   (b) Supprimez l'insertion que vous venez de faire.
3. Créer deux listes, insérer la première au milieu de la seconde.

Exercice 3 – Mutabilité et copies

On vérifie dans cet exercice les comportements relatifs à la mutabilité des listes étudiés en cours.

1. Créer une liste `liste1`, la copier dans `liste2`. Comparez les identifiants de `liste1` et `liste2`.
2. Modifiez un attribut de `liste1`. Quel est l'effet sur `liste2` ?

3. Effectuez une copie `liste3` de `liste1` par saucissonnage. Quel est l'effet sur les adresses ? Modifier un attribut de `liste1`. Quel est l'effet sur `liste3` ?
4. Créer une liste `liste4`, dont l'attribut `liste4[0]` est elle-même une liste. Créer une copie `liste5` de `liste4` par saucissonnage.
5. Comparez les adresses de `liste4` et `liste5`. Comparer les adresses des attributs `liste4[0]` et `liste5[0]`. Modifier un des attributs de la liste `liste4[0]`, et vérifier l'effet de cette modification sur `liste5`.
6. Créer le tuple `couple=([1,2],3)`. Peut-on modifier l'attribut d'indice 1 de `couple` ? Modifier un des attributs de `couple[0]`, et voir l'effet sur `couple`.
   Ce n'est pas parce que le contenu de `couple` *ne peut pas être modifié en place que les contenus des attributs ne peuvent pas être modifiés, si ces attributs sont mutables.*

Exercice 4 –

1. Dans le module `numpy.random` se trouve une fonction nommée `randint`. Utilisez-la pour créer une liste de 100 entiers tirés au hasard entre 0 et 99.
2. Calculer le nombre d'éléments de $[\![0, 99]\!]$ qui n'appartiennent pas à cette liste.
3. Recommencer cette expérience un grand nombre de fois et calculer la moyenne du nombre d'absents.
4. Comparer à la valeur théorique.

Exercice 5 – In and Out Shuffle
Une méthode traditionnelle pour mélanger un paquet de cartes consiste à couper le paquet en deux puis à entrelacer ces deux parties. Lorsque les deux parties sont égales et que l'entrelacement se fait carte par carte, le mélange est dit parfait.
Il y a deux types de mélanges parfaits :

- le *out shuffle* lorsqu'on reconstitue le jeu en commençant par la première carte de la première moitié ;
- le *in shuffle* lorsqu'on reconstitue le jeu en commençant par la première carte de la seconde moitié.

On rappelle que `li[a:b:p]` extrait les termes d'une liste de p en p, à partir de l'indice a jusqu'à l'indice $b-1$.

1. Écrire une fonction `out_shuffle` effectuant le mélange *out shuffle* d'une liste contenant un nombre pair d'éléments.
   Pour un jeu de 52 cartes, combien de mélanges doit-on effectuer pour retrouver la liste dans son état initial ?
2. Mêmes questions avec le *in shuffle*.

Exercice 6 – Le problème de Josephus
Josephus faisait partie d'un groupe de 41 rebelles juifs, à l'époque romaine. Cernés par les romains, le groupe décida d'un suicide collectif. Le déroulement devait être le suivant : le groupe se dispose en cercle, muni d'une origine (une personne numérotée 1). Cette personne se suicide en premier, puis une personne sur trois (parmi les survivants) en faisant le tour du cercle autant de fois qu'il faut pour qu'il ne reste personne. Josephus et un de ses compagnons ne voulaient pas de ce suicide. Josephus calcula les places que lui et son compagnon devaient prendre sur le cercle pour être les 2 derniers rescapés (et donc faire à leur guise pour terminer).
Écrire un programme répondant au problème de Josephus. On traitera le cas plus général où le groupe est constitué de n personnes, le nombre de survivants souhaités est k et le pas de l'expérience est p (ainsi on procède par élimination de p en p en commençant par la personne numérotée 1).

Exercice 7 – À l'aide du crible d'Ératosthène, consistant à barrer les multiples des nombres premiers au fur et à mesure qu'on les découvre, créer une liste constitué des entiers premiers inférieurs ou égaux à 1000.

Exercice 8 –

1. Créer une liste `entiers` de tous les entiers de 2 à 100.
2. Définir par compréhension une liste `diviseurs` contituée des couples (n, diviseurs de n), pour tout n élément de la liste `entiers`, les diviseurs de n étant donnés sous forme d'une liste.
3. En déduire la liste `premiers` de tous les entiers premiers de 2 à 100.
4. Que pensez-vous de cette méthode par rapport au crible d'Ératosthène ?