Informatique - TP 6

Listes - parcours par indice, modification de listes

M. Marmorat, M. Morel

11 décembre 2024

Exercice 1 **Proche du cours** Une sympathique classe de BCPST1 souhaite étudier la croissance d'une nouvelle espèce de haricot. On imagine que la classe dispose de 60 spécimens de haricots qui ont poussés en laboratoire, sous lumière artificielle, pendant plusieurs jours.

A la fin de l'étude, les étudiants mesurent la taille de chacun des spécimens et rassemblent les résultats mesurés en centimètres au sein d'une liste Python.

Pour simuler cette mesure, taper les lignes suivantes :

```
from numpy.random import normal

L = [normal(14, 3) for _ in range(60)]
```

Ceci crée une liste L qui contient 60 nombres réels aléatoires (c'est l'intérêt du module random), (choisis selon une distribution *normale*, de moyenne 14 et d'écart-type 3) : le k-ème élément de la liste représente la mesure de la taille du spécimen numéro k.

On peut tracer ces mesures grâce aux instructions

```
import matplotlib.pyplot as plt

plt.close() #on ferme la fenetre graphique courante, si elle est ouverte
plt.plot(L, "*") # on prepare la courbe
plt.show() #affiche la courbe
```

Q1 On souhaite vérifier les mesures des 5 premiers spécimens, car il semble y avoir eu un problème avec l'arrosage de ces plants. Créer la sous-liste de L qui contient ses 5 premiers éléments et l'afficher dans la console Python.

Q2 En réalité, les 10 derniers spécimens de haricots sont également suspects et méritent d'être examinés, suite à un problème de sur-acidité du sol. Comment créer la sous-liste de L contenant ses 5 premiers ainsi que ses 10 derniers éléments ? Tester.

Q3 Certaines mesures semblent aberrantes. En effet, la taille d'un haricot à maturité est généralement comprise entre 12 et 15 centimètres. On considère que les mesures hors de l'intervalle $[12, 15]$ sont des erreurs de mesure.

Créer une fonction `nettoie` qui prend en argument une liste de nombres M et qui crée une copie de M, dans laquelle toute valeur inférieure à 12 a été remplacée par 12, et toute valeur supérieure à 15 a été remplacée par 15.

Tester sur votre liste de résultats, et tracer les éléments de la nouvelle liste dans un graphique.

Exercice 2 **Encore du cours**

Q1 Écrire une fonction prenant en argument une liste contenant les nombres $x_1, x_2, \ldots, x_n$ et renvoyant la valeur de $\sum_{k=1}^{n} k\,x_k$. Testez votre fonction avec la liste `L = [1,3,5,7,...,101]`, on vérifiera que la somme demandée vaut alors $89\,726$.

Q2 Écrire une fonction `positiver` prenant en argument une liste de nombres réels et la renvoyant après avoir transformé tous les nombres négatifs qu'elle contient en des 0. Testez votre fonction.

Q3 Écrire une fonction prenant en argument une liste `L` et renvoyant la liste des éléments de `L` placés à un indice impair. Testez votre fonction.

Exercice 3 Numéro de sécurité sociale

Le numéro de sécurité sociale en France (officiellement appelé numéro d'inscription au répertoire des personnes physiques ou NIRPP) est un numéro à 15 chiffres[1] servant à identifier les individus.

Les 13 premiers chiffres de ce numéro ont une signification bien précise :

— le premier chiffre est un 1 pour les hommes et un 2 pour les femmes,

— les deux chiffres suivants sont les deux derniers chiffres de l'année de naissance,

— les chiffres en positions 6 et 7 correspondent au département de naissance,

— etc.

A contrario, les deux derniers chiffres du code ne correspondent à rien de concret et servent uniquement de *clé de vérification*. Ils peuvent être obtenus à partir des 13 autres, et permettent ainsi de détecter une erreur qui aurait pu être faite lors de la recopie du NIRPP. Ces deux derniers chiffres s'obtiennent par la formule : $97 - r$ où r est le reste dans la division euclidienne par 97 du nombre n formé par les 13 premiers chiffres du NIPPR.

Par exemple, si les 13 premiers chiffres d'un NIPPR forment le nombre $n = 1234567890124$ alors, comme $n = 1234567890124 = 12727504021 \times 97 + 87$, on a $r = 87$ donc $97 - r = 10$ et donc les deux derniers chiffres du NIPPR sont 1 et 0.

Dans cet exercice, on souhaite écrire une fonction prenant en argument une liste à 13 éléments `L` contenant les 13 premiers chiffres d'un NIPPR, et complétant la liste `L` en un NIPPR complet à 15 chiffres.

Q1 Dans un premier temps, il faut convertir la liste à 13 éléments `L` en un nombre n à 13 chiffres. Commençons avec moins de chiffres. Si `L = [1,3,5]`, on souhaite obtenir $n = 135$. Quelles opérations sur les éléments de `L` doit-on faire pour cela ? Écrire alors une fonction `list_to_number` prenant en argument la liste `L` et renvoyant le nombre n.

Q2 Dans un deuxième temps, il faut calculer le nombre m formé par les deux derniers chiffres du NIPPR selon la formule expliquée dans l'énoncé. Écrire une fonction `dernier` prenant en argument un nombre à 13 chiffres n et renvoyant le nombre à 2 chiffres (au plus) m correspondant. On vérifiera que `dernier(1234567890124)` vaut bien 10.

Q3 Dans un troisième temps, il faut reconvertir le nombre à deux chiffres m obtenu en une liste à deux éléments `M`. Le premier élément de `M` est le chiffre des dizaines de m, et son deuxième élément le chiffre des unités. Prenons l'exemple du nombre $m = 52$, par quelles opérations obtient-on les nombres 2 et 5 à partir de m ? Écrire alors une fonction `number_to_list` prenant en argument le nombre m et renvoyant la liste `M`.

Q4 Conclure alors en écrivant une fonction `complete` prenant en argument une liste à 13 éléments correspondant à un début de NIPPR et renvoyant la liste à 15 éléments correspondant au NIPPR complet. Testez votre fonction (avec votre propre numéro de sécurité sociale si vous le connaissez !).

1. Pour être exact, il peut aussi y avoir une lettre, A ou B, pour les personnes nées en Corse, mais nous excluons ce cas dans cet exercice.

Exercice 4 Mastermind

Le but de l'exercice est d'écrire un code simulant le jeu du Mastermind. Dans ce jeu, un joueur doit deviner par des essais successifs une combinaison secrète choisie par l'ordinateur. Pour chaque combinaison proposée, l'ordinateur indique le nombre d'éléments correctement placés ainsi que le nombre d'éléments présents dans la combinaison secrète mais à une autre place.

Q1 Écrire une fonction `liste_alea` prenant en argument un entier n et renvoyant une liste de n chiffres, choisis aléatoirement. *On rappelle qu'un chiffre est un entier compris entre* 0 *et* 9. *On utilisera la fonction* `choice` *de la bibliothèque* `random` *(qu'il faudra importer). Cette fonction permet de choisir un élément* x *aléatoirement parmi les éléments d'une liste* `L` *via la syntaxe suivante :*

```
x = random.choice(L)
```

Q2 Écrire une fonction `bonne_place` prenant en arguments deux listes de même taille `L` et `M` et renvoyant le nombre d'indices k tels que les éléments de `L` et `M` d'indice k sont égaux.

Par exemple, `bonne_place([1,3,7,5,2],[7,3,8,4,2])` doit renvoyer 2.

On ne demande pas de vérifier que les listes prises en arguments sont de même taille.

Q3 On souhaite définir une fonction `distrib` prenant en argument une liste de chiffres `L` et renvoyant une liste à 10 éléments dont l'élément d'indice k est le nombre de fois où le chiffre k apparait dans `L`. Par exemple, `distrib([3,2,6,1,3,5,3,1])` doit renvoyer la liste `[0,2,1,3,0,1,1,0,0,0]`.

1. *(Sans ordinateur).* Que doit renvoyer `distrib([0,3,7,3,1,2])` ?
2. Écrire la fonction `distrib`.

Q4 On souhaite définir une fonction `elt_communs` prenant en arguments deux listes de chiffres `L` et `M`, et renvoyant le nombre de chiffres communs aux deux listes. On souhaite compter les répétitions, mais ne pas tenir compte de la position des chiffres. Pour cela, on propose de :

— calculer les listes D_L et D_M, obtenues en appliquant `distrib` à `L` et `M`,

— calculer et renvoyer la somme $\sum_{k=0}^{9} \min(D_L[k], D_M[k])$.

1. *(Sans ordinateur).* En appliquant l'algorithme proposé, déterminer ce que doit renvoyer `elt_communs([2,3,2,7,1,4,2] , [1,5,3,3,7,2,2])`.
2. Écrire la fonction `elt_communs`.
   On rappelle que `min(x,y)` *renvoie le minimum entre les deux nombres* x *et* y.

Q5 L'ordinateur a choisi une liste `mystere` de 5 chiffres via la fonction `liste_alea` :

```
mystere = liste_alea(5)
```

Écrire une fonction `jeu` prenant en argument une liste `L` de 5 chiffres (correspondant à la proposition d'un joueur) et renvoyant le couple (a, b), où a est le nombre de chiffres bien placés dans `L`, et b le nombre de chiffres communs à `L` et à `mystere`, mais mal placés dans `L`. Par exemple, si la liste `mystere` vaut `[7,3,7,1,4]`, alors `jeu([1,3,7,5,2])` doit renvoyer le couple `(2,1)`.

Q6 Jouez contre l'ordinateur pour trouver la liste `mystere` !

Exercice 5 Nombres de Catalan

Les nombres de Catalan C_n sont définis par $C_0 = 1$ et : $\forall n \in \mathbb{N}^*,\ C_n = \sum_{k=0}^{n-1} C_k\, C_{n-1-k}$.

Q1 *(Sans ordinateur.)* Calculer à la main C_1, C_2, C_3 et C_4.

Q2 Écrire une fonction `next_cat` prenant en argument une liste `L` de longueur n qu'on supposera contenir les nombres de Catalan $C_0, C_1 \ldots, C_{n-1}$ et renvoyant le prochain nombre C_n. Vérifier votre fonction en calculant `next_cat([`C_0`,`C_1`,`C_2`,`C_3`])`.

Q3 Écrire une fonction `cat` prenant en argument $n \in \mathbb{N}$ et renvoyant la liste `[`C_0`,`C_1`,...,`C_n`]`. Testez votre fonction.