TP 1 : Remise en forme

Exercice 1:
Pour chaque élément du tableau ci-dessous, préciser son type (on a importé la bibliothèque `numpy` avec l'alias `np`) :

15	2.4	1.0	'bonjour'	'3.7'	'True'	True	[1,2,3]	np.zeros([3])	(1,2)

Vérifier votre intuition à l'aide de la syntaxe `type()`.

Exercice 2:
Soit $L = [1, 5, -4]$ et $M = [2, 0, -1, 1]$.

1. Introduire les listes L et M dans le shell. Que renvoie $L + M$? $2 * L$? Comment atteindre l'élément 5 de L ?
2. Mêmes questions après avoir introduit L et M comme des matrices lignes, c'est-à-dire des tableaux `numpy`.

Pour les syntaxes des commandes/bibliothèques, utiliser l'aide mémoire des concours AgroVeto

Exercice 3:
Soit $A = \begin{pmatrix} 2 & 4 \\ -1 & 1 \end{pmatrix}$ et $B = \begin{pmatrix} 1 & -1 \\ 1 & 0 \end{pmatrix}$.

1. Introduire A et B dans le shell comme des tableaux `numpy`.
2. Comment atteindre l'élément -1 de A ? Comment atteindre la première ligne de A ? La dernière colonne ?
3. Que renvoie $A + B$? $2 * A$?
4. Comprendre la différence entre `A*B` et `np.dot(A,B)`.
5. Comparer : `A*A, A**2, np.dot(A,A), la.matrix_power(A,2)`
6. Comprendre la différence entre `1/A` et `la.inv(A)`

Exercice 4:
L'objectif est de créer un tableau à deux dimensions de taille (n, p) avec que des 0.
Commenter les deux syntaxes suivantes :
`T=np.zeros([n,p])` `T=[p*[0] for i in range(n)]`
Par quelle syntaxe peut-on atteindre un coefficient dans chacun des cas ?

Exercice 5:

1. Créer une fonction d'entrée n qui renvoie la liste des n premiers entiers non-nuls.
   On proposera 3 syntaxes différentes.
2. A l'aide de la syntaxe `np.linspace()`, créer une fonction d'entrée n qui renvoie le tableau unidimensionnel avec les n premiers entiers non-nuls.

Exercice 6:
Ecrire une fonction d'en-tête `def compt(L,elt):` qui renvoie le nombre d'occurences de `elt` dans la liste `L`.

N'oubliez pas de tester votre programme ou votre fonction avant de passer à l'exercice suivant.

Exercice 7:
Ecrire une fonction d'en-tête `def premier1(L):` qui renvoie la position (= l'indice python) du premier 1 dans la liste `L` si il y en a au moins un et -1 sinon.

Exercice 8:
Ecrire une fonction qui prend en argument une liste, et qui renvoie la valeur du maximum de la liste ainsi que sa position dans la liste. Si le maximum n'est pas unique, le programme renvoie la première position.

Exercice 9:
Ecrire une fonction d'en-tête `def egale(M,N):` qui prend deux matrices en entrée et qui testera si ces deux matrices sont égales. *On commencera par vérifier qu'elles ont même taille.*

Exercice 10:

1. Créer une liste de liste L dans votre shell. A quoi correspond `len(L)` ?
2. Comment atteindre la 2^e liste de L ? Le 1^{er} élément de la première liste de L ?

Exercice 11:
Une liste L répertorie différents points de localisation à l'aide de 3 paramètres : la latitude, la longitude, et l'altitude. L est une liste de listes.

1. Créer une fonction `alt` qui prend en entrée la liste L et qui retourne une liste uniquement avec les valeurs de l'altitude pour tous les points étudiés.
2. Ecrire alors une fonction qui renvoie les dénivelés successifs que l'on a entre les points successifs étudiés.

Exercice 12:

Introduire deux chaînes de caractère de votre choix, et reprendre les questions de l'exercice 2.
Attention de ne pas confondre chaîne de caractère et liste, même si les opérations sont les mêmes !

Exercice 13:

1. Ecrire une fonction d'en-tête `def appartient(c,C):` qui teste si un caractère c se trouve dans une chaîne de caractère C en parcourant la chaîne.
   Soyez vigilents pour tester cette fonction avec une chaîne de caractère et non une liste !
2. Compléter la fonction précédente afin qu'elle renvoie également la position (=indice python) où se trouve le caractère en cas d'appartenance. (Si le caractère apparait plusieurs fois, une seule position suffit).

$\longrightarrow$ dorénavant, on pourra utiliser la syntaxe `c in C` valable également lorsque C est une liste ou une matrice.

Exercice 14:

On considère c une chaine de caractères correspondant à une séquence d'ADN.
Ecrire une fonction Python `mystere` comptant le nombre d'occurences de chaque nucléotide.

Exercice 15:

On considère deux chaînes de caractère $C1$ et $C2$ de même longueur. Ecrire une fonction permettant de compter le nombre de caractères communs entre $C1$ et $C2$ et étant à la même place.

Exercice 16:

Soit M une matrice ne contenant que des 0, 1 et 2.
Écrire une fonction qui compte le nombre de lignes de M comportant les 3 chiffres.

Exercice 17:

Ecrire une fonction donnant les indices ligne et colonne du premier 1 d'une matrice à 2 dimensions (en partant en haut à gauche).
*bonus** :* proposer une deuxième version qui n'utilise qu'une seule boucle while à la place des 2 boucles for.

Exercice 18:

Soit `L` une liste d'entiers compris entre 1 et 6.

1. Créer alors une liste de taille 6 qui contient le nombre d'occurences dans `L` de chaque entier.
   On pourra commencer par créer une liste de taille 6 remplie de 0, puis la faire évoluer en parcourant `L`
2. Compléter alors le script afin d'afficher l'entier qui est apparu le plus de fois. En cas d'égalité, on pourra n'en afficher qu'un. *Bonus : les faire afficher tous en cas d'égalité.*
3. Ecrire une variante lorsque L est un tableau (= matrice ligne) et que l'on crée également un tableau des occurences.

Pour les plus rapides uniquement

Exercice 19:

On considère deux chaînes de caractère $C1$ et $C2$ qui peuvent être de longueurs différentes. Ecrire une fonction permettant de compter le nombre de caractères communs entre $C1$ et $C2$, peu importe leur place (un même caractère, s'il apparaît plusieurs fois dans une des chaînes ne sera compté qu'une fois). Cette fonction renvoie également la chaîne de caractères contenant les caractères communs.

Exercice 20:

Dans cet exercice, on appelle liste descriptive d'une liste `L` d'entiers la liste `M` contenant les entiers qui, une fois lus, décrivent `L`. Par exemple : si `L=[1,3,2,2,1]` alors sa liste descriptive est `M=[1,1,1,3,2,2,1,1]`. (lire L de la façon suivante : un 1 puis un 3 puis deux 2 puis un 1)

1. Écrire une fonction `f` qui prend en argument une liste d'entiers `L` et qui renvoie sa liste descriptive. La fonction renverra `False` si la liste est vide.
2. On définit la suite de listes (L_n) par récurrence par : $L_0 = L$ et, pour $n \in \mathbb{N}$, $L_{n+1} = f(L_n)$. Afficher les 15 premiers termes de la suite lorsque `L=[1]`. Que constate-t-on ?

Exercice 21:

Ecrire une fonction qui prend en paramètre une liste de nombres et un nombre, et qui renvoie le plus grand nombre d'occurences successives de ce nombre dans la liste ou 0 si le nombre n'appartient pas à la liste.
Par exemple, si `liste=[1,8,5,9,9,7,5,8,9,9,9,8,5]` et `nombre=9` la fonction renvoie `3`.