Bilan révisions S1 - Les essentiels de l'oral

Tous les exercices de cette fiche sont issus de sujets d'oral du concours Agro-Véto. Cet oral se compose d'un exercice de maths avec temps de préparation, puis d'un exercice non préparé de Python : les exercices suivants sont directement tirés des exercices Python proposés par le concours.

1. Suites et sommes

Exercice 1. Pour tout $\lambda > 0$ et tout $n \in \mathbb{N}$, on note $S_{n,\lambda} = \sum_{k=0}^{n} e^{-\lambda} \frac{\lambda^k}{k!}$.

1. Écrire une fonction `factorielle(n)` qui prend en argument un entier naturel n et qui renvoie la valeur de $n!$.
2. Écrire une fonction `S(n,lambda)` qui prend en argument un entier naturel n et un flottant λ strictement positif, et qui renvoie la valeur de $S_{n,\lambda}$.
3. On admet que $S_{n,\lambda} \underset{n\to+\infty}{\longrightarrow} 1$. Pour tout $\lambda > 0$, on définit une fonction G_λ sur $[0,1[$ par :

$$\forall \in [0,1[, \quad G_\lambda(x) = \min\{n \in \mathbb{N} \mid S_{n,\lambda} > x\}$$

Écrire en Python une fonction `G(lambda,x)` qui renvoie la valeur de $G_\lambda(x)$.

Exercice 2. Écrire une fonction `somme_cumul(L)` qui prend en entrée une liste `L` d'entiers et qui renvoie une liste `M` des sommes cumulées, c'est à dire une liste de même longueur que `L`, telle que $M[i] = \sum_{k=0}^{i} L[k]$ pour tout indice i de `L`.

2. Un peu d'aléatoire

Vous verrez plus d'aléatoire plus tard dans l'année, et surtout en deuxième année.
Cependant, vous connaissez les deux fonctions essentielles du module `random` :

- `random()` renvoie un flottant aléatoire de l'intervalle $[0,1]$
- `randint(n,p)` renvoie un entier aléatoire de l'intervalle $[\![n,p]\!]$.

Exercice 3.
1. Écrire une fonction `pile_ou_face()` qui ne prend pas d'argument, et renvoie 0 ou 1, de manière équiprobable.
2. Écrire une fonction `suite_pile_face(n)` qui prend en argument un entier n et qui simule une série de n lancers d'une pièce équilibrée, en renvoyant la liste des résultats de ces lancers ("Pile" est codé par 1, et "Face" par 0).
3. Écrire une fonction Python qui simule une série de lancers d'une pièce équilibrée jusqu'à l'obtention de la configuration "Pile, Pile, Face", et renvoie le nombre de lancers nécessaires à l'apparition de cette configuration.

Exercice 4. Écrire une fonction python, d'arguments d'entrée i et N, qui simule une marche aléatoire sur $\mathbb{Z}$: le marcheur démarre sur l'entier i et à chaque pas, il avance de 1 ou recule de 1 avec probabilité $\frac{1}{2}$. La marche s'arrête après le N-ième pas et la fonction renvoie l'entier sur lequel le marcheur s'est arrêté.

Exercice 5.
1. Écrire une fonction `gene(n)` qui prend en entrée un entier naturel non nul n et renvoie une chaîne de n caractères formée aléatoirement, de façon équiprobable, des caractères `'A'`, `'C'`, `'G'`, `'T'`.
2. Écrire une fonction `nbAC(g)` qui prend en entrée une chaîne de caractères formée des caractères `'A'`, `'C'`, `'G'`, `'T'` et qui renvoie le nombre de fois où la séquence `'AC'` est présente.
   Par exemple, `nbAC('GAGCACCCTACTTGGCGCGA')` renverra 2.

3. Listes

Exercice 6. 1. Écrire une fonction `strict_croissante` prenant une liste d'entiers en paramètre et renvoyant `True` si elle est strictement croissante et `False` sinon.
Par exemple, `strict_croissante([1, 7, 99])` renvoie `True`.
Mais `strict_croissante([1, 7, 5])` et `strict_croissante([1, 7, 7])` renvoient `False`.

2. Écrire une fonction `strict_monotone` prenant une liste d'entiers en paramètre et renvoyant `True` si elle est strictement monotone et `False` sinon.
Par exemple, `strict_monotone([5, 2, 1])` renvoie `True`.
Mais `strict_monotone([1, 2, 1])` renvoie `False`.

Exercice 7. Une grille de Takuzu, ou Binairo, est une grille carrée à n lignes et autant de colonnes avec n pair. Au départ, chaque case peut contenir 0, 1 ou être vide.
En Python, une grille est codée par une liste de listes, les cases vides étant représentées par `None`. Par exemple, la grille :

0	1		0
1		0	1
0	0	1	1
1	1	0	

est codée par la liste `[[0, 1, None, 0], [1, None, 0, 1], [0, 0, 1, 1], [1, 1, 0, None]]`.
Dans la suite, on identifie la grille et la liste qui la représente.

1. Écrire une fonction prenant en argument une grille de taille non précisée et renvoyant le nombre de cases non remplies.
2. Écrire une fonction prenant en argument une grille de taille non précisée et renvoyant un booléen testant s'il existe deux lignes identiques.

Exercice 8. Soient a et b deux entiers naturels avec $a < b$.

1. Écrire une fonction `verif(L,a,b)` qui a pour paramètres une liste `L` d'entiers naturels et 2 entiers `a` et `b`, et qui renvoie `True` si tous les éléments de `L` sont dans l'intervalle $[\![a, b]\!]$ et `False` sinon.
2. Soit L une liste dont chaque élément est dans l'intervalle d'entiers $[\![a, b]\!]$.
Écrire une fonction `denombre(L,a,b)` qui détermine le nombre d'apparitions de chacune des valeurs possibles (entre a et b) de la liste L et qui renvoie le résultat dans une liste dont le premier élément représentera le nombre de a dans L, le deuxième le nombre de $a + 1$ dans L etc.
Exemple : pour $a = 0$ et $b = 4$, `denombre([1,3,0,4,1,3,1],0,4)` renverra `[1,3,0,2,1]`.
3. En déduire une fonction qui vérifie si une liste L contient exactement une fois tous les entiers entre 0 et $n - 1$, où n est la longueur de la liste.

Exercice 9. On souhaite exploiter le suivi d'une randonnée en effectuant des mesures lors de différents points de passage : latitude, longitude, altitude et temps (date).
Ces données sont contenues dans une liste `coords`, où `coords[i]` désigne la liste des 4 mesures effectuées au point de passage numéro i. Ainsi, `coords[i][2]` renvoie par exemple l'altitude relevée au point de passage numéro i.

1. Écrire une fonction `temps(coords)` qui renvoie la liste des temps relevés lors de la randonnée `coords`.
2. ans utiliser la fonction max, écrire une fonction `plus_haut(coords)` qui renvoie la liste `[lat, long]` du point le plus haut lors de la randonnée `coords`.