TP4 - Listes

→ *Q.1) Récupérez le fichier* `TP4.py` *sur cahier de prépas.*

→ *Q.2) Ouvrez pyzo puis le fichier* `TP4.py`.

→ *Q.3) Corriger le code proposé pour qu'il corresponde à la spécification donnée.*

→ *Q.4)*
1. *Construire la liste* `L` *contenant les entiers de* 0 *à* 100.
2. *En utilisant des tranches, récupérer :*
   - *la liste contenant les 42 premiers termes ;*
   - *la liste contenant les 42 derniers termes ;*
   - *la liste contenant les 21 premiers et les 21 derniers termes ;*
   - *la liste contenant tous les nombres pairs de* 0 *à* 100.

→ *Q.5) La fonction* `random` *du module* `random` *renvoie un flottant au hasard entre* 0 *et* 1. *Créer une liste de 1000 nombres aléatoires compris entre* 0 *et* 1, *puis en extraire la liste de tous les éléments qui sont compris entre* $\frac{1}{\pi}$ *et* $\ln(2)$.

→ *Q.6) Écrire une fonction* `carres(n)` *qui renvoie la liste des n premiers carrés (en partant de* 0^2 *compris).*

→ *Q.7) Écrire une fonction* `non_div3(n)` *qui renvoie la liste des entiers de* $[\![0, n-1]\!]$ *qui ne sont pas divisibles par* 3.

→ *Q.8) Écrire une fonction* `ajouteUn(L)` *qui étant donnée une liste de nombres* `L`, *ajoute* 1 *à tous les éléments de* `L`.

→ *Q.9) Écrire une fonction d'entête* `def moyenne(L:list) -> float:` *qui retourne la moyenne d'une liste de flottants* `L`.

→ *Q.10)*
1. *Écrire une fonction* `u(n)` *qui renvoie la liste des n premiers termes de la suite définie par* $u_0 = 1$ *et* $u_{n+1} = u_n\left(1 + \frac{1}{e^{n+1}}\right)$.
2. *Tracer la représentation graphique des 100 premiers termes de la suite précédente.*

→ *Q.11) Tracer le graphique de la fonction exponentielle entre* 0 *et* 2 *en prenant* 5 *points, puis* 20 *points, puis* 200 *points et enfin* 2000 *points.*

→ *Q.12) On dit qu'une liste* `L` *de longueur n est croissante si pour tout i entre* 0 *et* $n-2$, *on a* $L[i+1] \geq L[i]$. *Écrire une fonction* `est_croissante(L)` *qui renvoie* `True` *si* `L` *est croissante et* `False` *sinon.*

→ *Q.13) Écrire une fonction* `evalPoly(L, x)` *qui prend en paramètre une liste de nombres* $a_0, a_1, \ldots, a_n$ *et un flottant* `x` *et qui renvoie la valeur de* $a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$.

→ *Q.14) Écrire une fonction* `renverse(L)` *qui prend en argument une liste et la modifie en inversant l'ordre de ses éléments.*

→ *Q.15) Écrire une fonction* `pop2(L)` *qui prend en argument une liste de taille au moins 2, supprime son avant-dernier élément et le renvoie. La liste* `L` *doit être modifiée.*

→ *Q.16) On donne le dictionnaire* `scrabble` *qui associe à chaque lettre son nombre de points au scrabble. Écrire une fonction qui étant donné un mot écrit en majuscule, renvoie le nombre de points de ce mot.*

→ *Q.17) On stocke les réponses d'un QCM dans un dictionnaire dont les clés sont de la forme* `'Qx'` *(avec* `'x'` *le numéro de la question) et les valeurs sont parmi* `'a'`, `'b'`, `'c'` *et* `'d'`.
Écrire une fonction `note(reponses_prof, reponses_eleve)` *qui renvoie la note de l'élève dont les réponses sont dans* `reponse_eleve` *en comparant au corrigé dans* `reponse_prof` *: une réponse correcte vaut* 3 *points, une erreur* -1 *et une réponse manquante* 0.

→ *Q.18)*
1. *Écrire une fonction qui renvoie le nombre d'occurences de chaque lettre* `'A'`, `'T'`, `'G'` *et* `'C'` *dans une chaîne de caractères ne comprenant que ces lettres. Le résultats doit être un dictionnaire dont les clés sont* `'A'`, `'T'`, `'G'` *et* `'C'`.
2. *En utilisant la fonction précédente, écrire une fonction qui renvoie la lettre qui apparaît le plus souvent dans une chaîne de caractères ne comprenant que* `'A'`, `'T'`, `'G'` *et* `'C'`.

→ *Q.19)*
1. *Écrire une fonction qui étant donnée une liste d'entiers entre* 0 *et* 99 *renvoie un dictionnaire dont les clés sont les entiers de* 0 *à* 99 *et les valeurs sont les nombres d'occurence de chaque entier dans la liste.*
On peut créer un dictionnaire en compréhension : par exemple `d = {i:0 for i in range(100)}`.
2. *En utilisant le fonction précédente, écrire une fonction qui à partir d'une liste* `L` *d'entiers de* 0 *à* 99, *renvoie une nouvelle liste contenant les même éléments que* `L` *mais triée par ordre croissant.*

→ *Q.20) Écrire une fonction* `fusion(L, M)` *qui prend en arguments deux listes triées et qui renvoie une liste triée contenant tous les éléments de* `L` *et* `M`, *sans modifier* `L` *et* `M`.

→ *Q.21) Écrire une fonction* `melange(L)` *qui prend en argument une liste et la mélange de la façon suivante : on parcours* `L` *et pour chaque indice* `i`, *on échange l'élément en position* `i` *avec celui situé à un indice choisi aléatoirement entre* 0 *et* `i`.