TP: tableau et matrice
Informatique pour tous

Comme toujours, toutes les fonctions doivent être testées.

Rappels sur les tableaux:

- `array` (tableau) est un type similaire à `list`, défini dans le module `numpy`. Il faut donc écrire `import numpy as np` pour l'utiliser.
- `np.array(L)` créé un tableau à partir d'une liste `L`. Par exemple `T = np.array([1, 7, 4])` créé un tableau `T` contenant les éléments 1, 7 et 4.
- Un tableau doit contenir des éléments tous de même type.
- Il est impossible de modifier la taille d'un tableau (sa taille est fixée à sa création).
- **On peut effectuer n'importe qu'elle opération arithmétique (+, -, *, /, ...) sur deux tableaux (de même taille), ce qui donne le tableau obtenu en appliquant l'opération case par case.**
  Exemple: `np.array([7, 1]) + np.array([2, 5])` donne le tableau `np.array([9, 6])`.
- On peut accéder/modifier le $i$ème élément d'un tableau `T` en écrivant `T[i]` et connaître la taille de `T` avec `len(T)` (comme pour les listes).

I Calcul vectoriel

Dans cette partie, on stocke un vecteur sous forme d'un tableau. Par exemple, le vecteur $\begin{pmatrix}1\\4\\2\end{pmatrix}$ est représenté par `np.array([1, 4, 2])`.

1. Écrire une fonction `addition` renvoyant la somme de deux vecteurs représentés par des tableaux.
   Par exemple $\begin{pmatrix}1\\4\\2\end{pmatrix} + \begin{pmatrix}2\\0\\5\end{pmatrix} = \begin{pmatrix}3\\4\\7\end{pmatrix}$.
2. Écrire une fonction calculant la somme des éléments d'un tableau.
3. En utilisant la fonction précédente, écrire une fonction calculant le produit scalaire de deux vecteurs représentés par des tableaux. On rappelle que le produit scalaire de $\begin{pmatrix}u_1\\\vdots\\u_n\end{pmatrix}$ et $\begin{pmatrix}v_1\\\vdots\\v_n\end{pmatrix}$ est $u_1v_1+...+u_nv_n$.
4. Écrire une fonction `norme` renvoyant la longueur d'un vecteur.
   Par exemple, la longueur de $\begin{pmatrix}0\\4\\3\end{pmatrix}$ est $\sqrt{0^2+4^2+3^2}$.
5. Écrire une fonction `perpendiculaire` déterminant si deux vecteurs sont perpendiculaires. On rappelle que c'est le cas si et seulement si leur produit scalaire est nul.

II Calcul matriciel

Une ligne d'une matrice est représentée par un tableau. On représente une matrice par un tableau de ses lignes (donc un tableau de tableaux).

1. Définir la matrice $\begin{pmatrix} 1 & 2 \\ 4 & 3 \\ 2 & 0 \end{pmatrix}$ en Python.
2. Écrire une fonction calculant le maximum des éléments d'une matrice.
3. Écrire une fonction effectuant le produit de deux matrices. On pourra utiliser `np.zeros((n, p))` pour créer une matrice de taille $n \times p$ avec que des zéros.
4. Écrire une fonction effectuant le produit d'une matrice par un vecteur.
5. Écrire une fonction `puissance` telle que `puissance(M, k)` renvoie un tableau représentant `M` à la puissance `k`.
   On pourra utiliser `M.copy()` qui renvoie une copie d'un tableau `M`.
6. Quelle est la complexité de la fonction précédente ? L'améliorer avec un algorithme d'exponentiation rapide, c'est à dire en utilisant :

$$\begin{cases} M^n = (M^{\frac{n}{2}})^2 & \text{si } n \text{ est pair} \\ M^n = M \times (M^{\frac{n-1}{2}})^2 & \text{sinon} \end{cases}$$

   Quelle est la nouvelle complexité ?
7. Mettre l'équation de récurrence $u_{n+2} = u_{n+1} + u_n$ sous forme matricielle, puis en déduire u_n en fonction de n en utilisant la fonction précédente. On prendra $u_0 = u_1 = 1$.