CHAPITRE 4

INF 4 - Tableau Numpy - partie 1

On a déjà vu les listes Python, seulement elles ne sont pas très efficaces. Les listes qu'on manipulera par la suite sont celles codées dans la bibliothèque **Numpy**. On rappelle qu'on doit importer la bibliothèque Numpy en tapant

```
import numpy as np
```

Le type du tableau numpy est le type **np.array**.

1. TABLEAUX UNIDIMENSIONNELS

Création et méthodes. On peut créer un tableau uni-dimensionnel (ou liste) en tapant

```
X = np.array([1,2,3,4,5])
```

On obtient une liste et on peut faire plusieurs opérations entre listes

- `np.size` prend en entrée la liste et renvoie sa taille
- les opérations usuelles fonctionnent sur les listes : le `+` additionne les listes terme à terme, etc.
- on peut appliquer les fonctions usuelles aux listes : par exemple `np.cos(X)` renverra la liste des cosinus des éléments de `X`
- la commande `np.append` permet de rajouter un élément en queue de tableau. Par exemple, regardez ce que fait

  ```
  X = np.array([1,2,3,4,5])
  X = np.append(X,6)
  ```

  Cela permet de rajouter des éléments dans une liste sans savoir quelle sera la taille finale. On peut aussi appeler `np.append(X,Y)` qui concatènera deux listes.

On a le droit d'appliquer une même opération logique à un tableau ! Par exemple

```
X = np.array([1,2,3,4,7,6])
X > 3
```

renverra le tableau de booléen

```
array([False, False, False, True, True, True, True])
```

Exercice 1 Créer deux tableaux de booléens de même taille `X` et `Y`. Essayer les commandes `X and Y`, `X or Y`, etc. Cela ne marche pas, que conclure ?

Remarque 1.1 — Pour "vectoriser" les opérations booléennes on utilisera les commandes `np.logical_and`,`np.logical_or`, etc.

Accès et modification. On peut accéder aux éléments des tableaux Numpy et les modifier. Les commandes sont les suivantes :

- l'accès se fait par la commande `X[i]` où `i` est un entier. Attention, les éléments d'un tableau de n nombres sont numérotés de 0 à $n-1$.
- la modification se fait par la commande `X[5] = 1000` qui ici va remplacer le sixième élément du tableau par 1000.

Attention

Si on tape

```
X = np.array([0,0,1,1,2,2,3,3])
Y = X
Y[0] = 1998
X
```

```
import numpy as np

X = np.array([0,0,1,1,2,2,3,3])
Y = np.zeros(np.size(X))
for k in range(np.size(X)):
        Y[k] = X[k]

Y[0] = 1000
X
```

ou, encore plus simple

```
import numpy as np

X = np.array([0,0,1,1,2,2,3,3])
Y = X[:]
```

```
Y[0] = 1000
X
```

Quelques méthode de création de tableaux. On ne pourra pas toujours remplir des tableaux directement en rentrant la ligne de façon explicite. Pour cela on a des méthodes à connaître.

1. `np.zeros(n)` va créer un tableau de taille n avec uniquement des zéros.
2. `np.ones(n)` va créer un tableau de taille n rempli de 1.
3. `np.arange` est la commande qui sera utile pour les tracés de suites. Elle fonctionne exactement comme la commande `range` que l'on a vue dans les boucles For. Elle a l'avantage de prendre en entrée des arguments non entiers. Que fait par exemple ?

   ```
   np.arange(0, np.pi, 0.1*np.pi)
   ```

4. `np.linespace` est la commande qui sera utile pour les tracés de fonction. En effet crée un tableau de premier terme a (flottant), de dernier termes b (flottant) et de taille n (entier) dont les éléments sont réparties régulièrement entre a et b. Par exemple, que fait le code suivant ?

   ```
   X = np.linspace(0,10,21)
   ```

Appliquer une fonction mathématique à un tableau. Les fonctions mathématiques de numpy peuvent s'appliquer en une fois à tous les éléments d'un tableau. Par exemple le code suivant renvoie les valeurs de sinus entre 0 et π sur un ensemble de valeurs discrétisées.

```
X = np.linspace(0,np.pi ,25)
Y = np.cos(X)
Y
```

2. EXERCICES DE CRÉATIONS DE TABLEAU

Exercice 2 Créer les tableaux suivants :

1. $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{pmatrix}$
2. $\begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$
3. $\begin{pmatrix} 2 & 2 & 2 & 2 & 2 & 2 & 0 & 2 \end{pmatrix}$

Exercice 3 Créer une fonction `basecanonique` qui prend en entrée un entier n et un entier i entre 1 et n et renvoie un tableau de taille n qui ne contient que des zéros sauf le coefficient $i-1$ qui vaut 1.

Exercice 4 Créer un tableau qui prend en entrée un entier n et renvoie le tableau des valeurs $\cos(k)$ entre 0 et n.

Exercice 5 Créer une fonction qui prend en entrée un tableau représentant une suite $(u_0, u_1, \dots, u_n)$ et renvoie la suite $(v_0, v_1, \dots, v_n)$ avec

$$v_k = \frac{1}{n+1} \sum_{k=0}^{n} u_k.$$

Exercice 6 Créer une fonction qui prend en entrée un tableau et renvoie le tableau qui contient un terme sur deux.

Exercice 7 Crée une fonction qui prend en entrée un "pas" dx et renvoie le tableau des valeurs de la fonction **sinus** pour des flottants compris entre 0 et 2π uniformément répartis et séparés de dx.

Combien d'éléments contient ce tableau ?