Statistique pour la SAé 2.04 - TP0
Listes, Numpy.Array et Pandas.DataFrame

A. Révisions sur les Numpy.Array

Création de Numpy.Array

1. Exécuter avec Python les commandes suivantes et noter à côté ce qu'elles renvoient :

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

B=np.zeros((2,3))

C=np.ones((3,2))

D=np.eye(4)
```

2. Vous connaissez maintenant les principales manières de créer de nouveaux numpy.array. Définir en Python les tableaux suivants :

$$M1 = \begin{pmatrix} 5 & 6 & 7 & 8 & 9 \end{pmatrix}; \quad M2 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \end{pmatrix};$$

$$M3 = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}; \quad M4 = \begin{pmatrix} 1 & 2 & 4 \\ -1 & 2 & 3 \\ 1 & 8 & 9 \end{pmatrix}.$$

3. A l'aide de `.tolist()`, créer une liste `L3` à partir du tableau `M3`. Comparer et comprendre les résultats renvoyés par la fonction `len()` et la méthode `np.shape()` sur `M3` et `L3`.

Concaténation, slicing

4. Exécuter les commandes suivantes, et noter ce qu'elles renvoient :

```
np.concatenate((A,B),axis=0)

np.concatenate((A,B),axis=1)

np.concatenate((B,C),axis=0)
M4[1, 0]
M4[0, :]
M4[0:1, :]
M4[ :, 1]
M4[1:3, 0 :2]
```

5. Comparer `np.shape(M4[0, :])` avec `np.shape(M4[0:1, :])` , puis `np.shape(A[:,1])` avec `np.shape(A[ :, 1 :2])`. Que se passe-t-il ?
6. Construire les tableaux suivants à partir des tableaux $M1, M2, M3$, et $M4$ définis précédemment :

$$M5 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 5 & 6 & 7 & 8 & 9 \end{pmatrix}, \quad M6 = \begin{pmatrix} 5 & 6 & 7 & 8 & 9 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix},$$

$$M7 = \begin{pmatrix} -1 & 2 \\ 1 & 8 \end{pmatrix}, \quad M8 = \begin{pmatrix} 1 & 8 & 9 \\ 1 & 2 & 4 \\ -1 & 2 & 3 \end{pmatrix}$$

7. Effectuer les modifications suivantes (en un minimum de commandes) :
   - M9 : Remplacer la 1e et la 3e cases de $M1$ par 8,
   - M10 : Ajouter 2 au début de $M2$,
   - M11 : Ajouter 8 en bas de $M3$ et 9 en haut de $M3$,
   - M12 : Supprimer la 2e ligne de $M4$. On pourra utiliser l'instruction `np.delete()`,
   - M13 : Supprimer la 2e colonne de $M4$.

Pour les + rapides. **Utilisation d'opérations matricielles**

8. Commencer par vérifier que vous vous rappelez du produit matriciel, en faisant les produits suivants sur papier :

$$\begin{pmatrix} -1 & 2 \\ 1 & 8 \end{pmatrix} \times \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, \qquad \begin{pmatrix} 1 & 2 & 3 \end{pmatrix} \times \begin{pmatrix} 1 & 8 \\ 1 & 2 \\ -1 & 2 \end{pmatrix}$$

   Si besoin de rappels sur le produit matriciel, relire le Polycopié de la ressource Outils Fondamentaux (sur Moodle) p11.

9. Vérifiez vos calculs précédents à l'aide de Python. Vous pourrez utiliser les matrices des questions précédentes, ainsi que
   - le produit matriciel : `np.dot(A,B)` renvoie le produit $A \times B$.
   - la transposition de matrice : `np.transpose(A)`.

10. Utiliser un produit matriciel pour créer une matrice à 10 lignes et 10 colonnes de la forme suivante (la "table de multiplication" !) :

$$M14 = \begin{pmatrix} 1 & 1\times 2 & 1\times 3 & \dots & 1\times 10 \\ 2 & 2\times 2 & 2\times 3 & \dots & 2\times 10 \\ 3 & 3\times 2 & 3\times 3 & \dots & 3\times 10 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 10 & 10\times 2 & 10\times 3 & \dots & 10\times 10 \end{pmatrix}$$

11. Utiliser un produit matriciel pour calculer a, un autre pour calculer b, et 2 produits pour calculer c :
    - $a = 1\times 1 + 2\times 2 + 3\times 3 + 4\times 4 + 5\times 5$,
    - $b = 1\times 2 + 2\times 3 + 3\times 4 + 4\times 5 + 5\times 6$,

12. Soit $X = [3, 2, 1, 6, 4, 5, 2, 5]$ une série de lancers de dés de moyenne 3,5. Calculer la variance de X à l'aide d'un produit matriciel.

B. Premiers pas avec les Pandas.DataFrame

Dans les TP de Statistiques pour la SAé, on utilisera la librairie Pandas, essentiellement pour importer nos fichiers .csv et manipuler des DataFrames. Commencer par l'importer :

```
import pandas as pd
```

Création d'un DataFrame de toutes pièces

On a vu dans la partie cours qu'on peut créer un DataFrame à partir d'un tableau numpy.array.

13. Créer un DataFrame `PremierDF` similaire au DataFrame `df5` du cours, en remplaçant les noms des 3 lignes par le vôtre et ceux de vos 2 voisin.es les plus proches dans cette salle, et en remplissant les colonnes par l'Age, l'Année de Naissance, l'Année d'obtention du Brevet des collèges, et un chiffre entre 0 et 5 décrivant son humeur du jour (0 étant une très mauvaise humeur, 5 une très bonne humeur).
14. Dans la console, tester les commandes suivantes. Que renvoie chacune d'elles ?

```
PremierDF.columns
PremierDF.index
PremierDF.shape
PremierDF['Age']
```

Création d'un DataFrame à partir d'un fichier .csv

15. Télécharger sur Moodle le fichier Sangliers.csv. Importer le fichier `Sangliers.csv` dans Python sous forme d'un DataFrame `SangliersDF`.
16. Dans la console, tester les commandes suivantes.

```
SangliersDF.index
SangliersDF.shape
SangliersDF['Annees']
```

Quels sont les noms donnés aux lignes de ce DataFrame ?

DataFrame et Numpy.Array

Comme on a travaillé avec des tableau Numpy.Array en Python jusque là, il sera intéressant de savoir passer du format DataFrame au format Numpy.Array et inversement.
Dans les questions suivantes, on va s'entrainer à passer d'un DataFrame à des np.array et de np.array à DataFrame. L'objectif de ces questions est de créer un DataFrame `SangliersODF` dans lequel le nom des lignes sera les années.

17. A partir du DataFrame `SangliersDF`, créer un np.array `SangliersAr` contenant toutes les valeurs numériques du DataFrame `SangliersDF`.

18. En utilisant des commandes de slicing, créer un np.array `AnneesAr` contenant uniquement les années, puis un autre np.array `SangliersOAr` contenant toutes les données sauf les années.

19. A l'aide d'une des commandes vues dans la question 15 et de la commande `.to_numpy()`, créer un np.array `ColAr` contenant les noms des colonnes de `SangliersDF`.

20. En utilisant `SangliersOAr`, `AnneesAr` et `ColAr`, créer un DataFrame `SanglierODF` contenant les données sangliers sauf les années, et ayant les années comme noms de lignes.