Tableaux

I) Comment on implémente des tableaux avec des lignes et des colonnes ?

On représente un tableau avec des lignes (sens horizontal) et des colonnes (sens vertical) par une liste de liste. Si je veux représenter $M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$ j'entre la première ligne comme une liste, puis la seconde ligne comme une liste, puis la troisième comme une liste. Toutes ces listes sont dans une liste séparée (donc) par une virgule.

In [ ]:
```
M = [[1,2,3],[4,5,6],[7,8,9]]
print(M)
```

II) Afficher et Acceder au tableau

On remarque que quand on affiche notre tableau on obtient pas le jolie tableau $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$. On va essayer de l'afficher de façon "jolie". Pour cela on va afficher tout les éléments du tableau

In [ ]:
```
for ligne in M:
    print(ligne)
```

On obtient donc la représentation en tableau de notre tableau. Cependant ce qui est intéressant sera d'avoir accès à toute les valeur du tableau. On va chercher la valeur de la ligne i et de la colonne j. Donc on va taper `M[i][j]`. En effet `M[i]` est une liste et donc a des éléments qui sont ordonnées par les colonnes.

In [ ]:
```
print(M[1][0])
```

Pourquoi `M[1][0]` est 4 ?

Si on veut afficher toutes les valeurs on fera ainsi une double boucle. Vous pouvez remplacer les len(.) par les tailles du tableau.

In [ ]:
```
for i in range(len(M)):
    for j in range(len(M[i])):
        print(M[i][j])
```

III) Exercice filé : Sudoku

Voici une fonction pour afficher les tableaux.

In [ ]:
```
def afficher_tableau(tableau):
    for i in tableau:
        print(i)
```

1. Faire la variable suivante M représentant le tableau suivant :

1	2	3
4	5	6
7	8	9

In [ ]:

Vérifier votre tableau avec la fonction que je vous ai donné.

In [ ]:

2. Faire une fonction qui prend en argument un tableau M, un couple (t-uplet avec 2 élements) indices de ligne i et j et qui renvoie un tableau où les lignes i et j ont été permutées.

In [ ]:

3. Faire une fonction qui prend en argument un tableau M, deux indices de colonne i et j et qui renvoie un tableau où les colonnes i et j ont été permutées.

In [ ]:

4. Faites une fonction avec le module random qui renvoit un couple (t-uplet à 2 éléments) de nombres compris entre 0 et 2.

In [ ]:

5. Combiner la fonction de la question 4 et les fonctions des questions 2 et 3, pour faire deux fonctions :

- Une qui prend en argument un tableau qui mélange deux lignes choisies au hasard et qui renvoie le tableau mélangé
- Une qui prend en argument un tableau qui mélange deux colonnes choisies au hasard et qui renvoie le tableau mélangé

In [ ]:

6. Combiner les deux fonctions précédentes pour faire une fonction qui prends en entrée un tableau et qui renvoit un tableau avec 2 lignes choisies au hasard échangées et 2 colonnes choisies au hasard échangées.

In [ ]:

7. Faire une fonction qui prend en entrée un entier n et qui fait tourner n fois la fonction précédente.

In [ ]:

8. Vous venez de faire une grille de sudoku réponse. Maintenant faisons la grille question. Tester plusieurs fois la fonction suivante avec plusieurs valeurs de p compris entre 0 et 1. Que comprenez vous ?

In [ ]:
```
def random_hide(p,k):
    if rd.random()<p:
        return ""
    else:
        return k

p = 0.7
print(random_hide(p,3))
```

Réponse : ...

9. Faire une fonction qui prend en entrée un tableau qui affiche valeur par valeur toutes les valeurs du tableau. (différent de `afficher_tableau` qui affiche les valeurs 3 par 3)

In [ ]:

10. Adapter cette fonction pour qu'elle cache aléatoirement des valeurs du tableau. Elle prendra en argument un p compris entre 0 et 1.

In [ ]: