TP4_2024

February 6, 2024

Calcul scientifique – TP4 – numpy

Nom :
Prénom:

Ce TP reprend la fin du TP3 mais propose une suite (qui n'est pas une option!). Comme au TP3, il n'y a pas de boucles à écrire.

Attention: exercice noté à la fin du TP5 (sur sympy et numpy, contenus des TP 1,2,3 et 4).

Pour commencer, on importe la librairie `numpy`

[1]:
```
import numpy as np
```

0.0.1 1) Utilisation des index

Creer un tableau nommé tab à une dimension de taille 19 avec des valeurs aléatoires puis l'afficher. Ensuite , afficher les valeurs dont les **indices** sont pairs (attention, les indices pas les valeurs). L'instruction doit être valable pour n'importe quel tableau à 1 dimension, quel que soit le nombre de valeurs.

Pour comparer les méthodes, afficher les valeurs paires du tableau

[2]:
```
# A compléter ...
```

```
tab [6 3 8 6 1 6 9 7 1 6 6 3 1 1 1 5 2 9 1]
val indices pairs : [6 8 1 9 1 6 1 1 2 1]
val paires [6 8 6 6 6 6 2]
```

Afficher la somme des valeurs de tab dont les **indices** sont pairs puis la somme des éléments qui sont à des places multiples de 3 plus 1 (index: 1, 4, 7 etc…)

[3]:
```
# A compléter ...

print('val indices pairs :',s1)
print('val indices 1,4,7 etc :',s2)
```

```
val indices pairs : 36
val indices 1,4,7 etc : 20
```

0.0.2 2) Modifications dans un tableau à 2D

Créer un tableau `a` de taille 8x8 (8 lignes et 8 colonnes) rempli de valeurs aléatoires entières de 1 à 9 inclus.

Pour toutes les questions suivantes, nous travaillerons avec un tableau de taille 8x8 mais vos instructions devraient fonctionner pour un tableau de taille quelconque

On fera un nouveau tableau par question pour mieux observer les résultats produits.

Au moyen du sélecteur approprié, remplacer toutes les valeurs du tableau par la valeur 0. Attention le nombre de lignes / colonnes doit être conservé. Afficher le résultat. On doit remplacer les valeurs, pas créer un nouveau tableau.

[4]:
```
# A compléter ...
print(a)
# A compléter ...
print("\nrésultat après modifications\n",a)
```

```
[[7 7 7 9 8 9 9 4]
 [9 7 6 8 2 4 5 8]
 [8 5 2 6 2 4 2 3]
 [7 3 5 9 3 6 4 8]
 [3 7 7 1 6 5 1 1]
 [6 9 9 9 3 3 4 8]
 [2 1 9 7 6 3 5 1]
 [3 5 9 5 7 4 2 7]]

résultat après modifications
 [[0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]]
```

Créer un tableau `a` de taille 8x8 (8 lignes et 8 colonnes) rempli de valeurs aléatoires entières de 1 à 9 inclus, puis mettre la première ligne du tableau à 0.

[5]:
```
# A compléter ...
```

```
[[0 0 0 0 0 0 0 0]
 [2 8 2 6 6 2 1 4]
 [2 6 2 4 5 1 2 6]
 [3 7 1 8 7 6 9 4]
 [8 9 5 5 7 3 1 3]
 [7 4 6 5 2 5 4 3]
 [7 2 2 2 2 2 3 5]
 [2 2 8 2 1 3 3 3]]
```

Créer un tableau `a` de taille 8x8 (8 lignes et 8 colonnes) rempli de valeurs aléatoires entières de 1 à 9 inclus puis mettre la colonne numéro 2 du tableau à 0.

[6]:
```
# A compléter ...
```

```
[[3 1 0 4 8 8 6 7]
 [1 6 0 6 7 7 6 9]
 [5 8 0 6 5 1 7 8]
 [1 8 0 6 6 5 3 1]
 [9 2 0 8 5 7 7 3]
 [4 8 0 5 3 8 7 9]
 [7 8 0 8 3 9 1 2]
 [2 2 0 1 1 4 5 9]]
```

Créer un tableau `a` de taille 8x8 (8 lignes et 8 colonnes) rempli de valeurs aléatoires entières de 1 à 9 inclus, puis mettre la première et la dernière colonne à 0 (en une seule instruction)

[7]:
```
# A compléter ...
```

```
[[0 5 9 9 3 3 4 0]
 [0 3 3 5 1 6 6 0]
 [0 3 9 3 9 7 8 0]
 [0 5 1 2 7 8 1 0]
 [0 5 6 4 7 8 3 0]
 [0 5 7 3 9 2 3 0]
 [0 9 6 4 4 7 8 0]
 [0 4 9 4 4 9 5 0]]
```

Créer un tableau `a`de taille 8x8 (8 lignes et 8 colonnes) rempli de valeurs aléatoires entières de 1 à 9 inclus puis mettre les 9 cases de la partie supérieure gauche du tableau à 0 en une seule instruction.

[8]:
```
# A compléter ...
```

```
[[0 0 0 7 4 9 3 6]
 [0 0 0 5 9 3 1 5]
 [0 0 0 1 3 9 5 8]
 [5 9 6 4 6 9 2 2]
 [8 6 9 8 9 8 8 2]
 [7 8 6 3 4 1 1 7]
 [5 3 4 7 1 3 9 8]
 [4 4 8 9 7 5 1 8]]
```

Créer un tableau `a` de taille 8x8 (8 lignes et 8 colonnes) rempli de valeurs aléatoires entières de 1 à 9 inclus puis mettre les cases qui sont sur des lignes **et** des colonnes paires à 0.

[9]:
```
# A compléter ...
```

```
[[0 5 0 2 0 6 0 5]
 [2 1 4 1 9 4 3 7]
 [0 5 0 2 0 2 0 8]
 [4 2 4 9 1 8 7 6]
```

```
 [0 4 0 8 0 3 0 3]
 [3 3 1 7 2 3 3 7]
 [0 7 0 3 0 2 0 6]
 [5 2 6 9 2 2 3 5]]
```

Créer un tableau `a` de taille 8x8 (8 lignes et 8 colonnes) rempli de valeurs aléatoires entières de 1 à 9 inclus puis mettre les cases $[1, 3]$ et $[3, 2]$ à 0 (en une seule instruction).

[10]:
```
# A compléter ...
```

```
[[3 5 9 3 9 9 5 4]
 [2 6 1 0 4 4 2 2]
 [2 3 6 2 4 8 6 2]
 [4 3 0 5 3 3 6 3]
 [5 6 1 7 1 4 7 8]
 [6 5 3 4 5 4 9 6]
 [7 7 6 3 1 6 6 3]
 [6 2 1 3 2 4 5 3]]
```

Créer un tableau `a` de taille 8x8 (8 lignes et 8 colonnes) rempli de valeurs aléatoires entières de 1 à 9 inclus puis mettre à 0 toutes les lignes pour lesquelles le vecteur `vect= np.array([False, True, True, False,False, 'True',False, 'True'])` vaut `True` (en une seule instruction).

[11]:
```
# A compléter ...
```

```
[False  True  True False False  True False  True]
[[6 6 5 3 3 3 8 1]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [8 6 8 4 5 6 5 1]
 [9 4 6 4 6 4 1 1]
 [0 0 0 0 0 0 0 0]
 [5 4 2 5 8 7 4 1]
 [0 0 0 0 0 0 0 0]]
```

A l'aide de la fonction np.ones, créer un tableau carré de taille 7 ne contenant que des 1. Puis en maximum deux lignes, faire un "damier" en remplaçant certains 1 par des 0 (voir résultat ci-dessous)

[12]:
```
# A compléter ...
```

```
creation de  a
 [[1. 1. 1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1. 1. 1.]]
apres transformation
 [[0. 1. 0. 1. 0. 1. 0.]
 [1. 0. 1. 0. 1. 0. 1.]
```

```
 [0. 1. 0. 1. 0. 1. 0.]
 [1. 0. 1. 0. 1. 0. 1.]
 [0. 1. 0. 1. 0. 1. 0.]
 [1. 0. 1. 0. 1. 0. 1.]
 [0. 1. 0. 1. 0. 1. 0.]]
```

0.0.3 3. Partage de valeurs dans un tableau

a) " Vues" (extrait avec des slices) Créer un tableau `a` de taille 4x4 rempli de valeurs aléatoires entières de 1 à 9 inclus que l'on nommera a.

Créer ensuite une vue (appelée `b`) de la matrice `a` correspondant aux sélecteurs précédents en prenant pas ar exemple :

```
b = a[0,:]
```

Modifier le contenu de `b` et regarder l'effet sur `a`. Puis modifier `a` et regarder l'effet sur `b`

Exemple de ce qui est attendu :

```
a= [[6 3 8 1]
 [3 9 4 4]
 [9 5 4 5]
 [4 2 9 8]]
 b= [6 3 8 1]
Après modification de la case 0,3 de a (mise à -1) et la case 0,0 de b (mise à 0)     on obtie
a= [[ 0  3  8 -1]
 [ 3  9  4  4]
 [ 9  5  4  5]
 [ 4  2  9  8]]
 b= [ 0  3  8 -1]
```

[13]:
```
# A compléter ...
```

```
a= [[9 8 4 1]
 [2 1 6 5]
 [3 5 6 1]
 [8 8 4 2]]
première ligne b= [9 8 4 1]
Après modification de la case 0,3 de a (mise à -1) et la case 0,0 de b (mise à
0) on obtient :
a= [[ 0  8  4 -1]
 [ 2  1  6  5]
 [ 3  5  6  1]
 [ 8  8  4  2]]
b= [ 0  8  4 -1]
```

b) Copies Reproduire la même expérience, mais cette fois-ci en définissant `b` comme une copie de `a` (et non plus une vue) à l'aide de la fonction copy avec l'instruction a[0,:].copy(). Comparer les résultats.

[14]:
```
# A compléter ...
```

```
a= [[6 5 8 1]
 [4 5 5 6]
 [3 2 6 3]
 [2 7 9 6]]
b= [6 5 8 1]
Après modification de la case 0,3 de a (mise à -1) et la case 0,0 de b (mise à
0) on obtient :
a= [[ 6  5  8 -1]
 [ 4  5  5  6]
 [ 3  2  6  3]
 [ 2  7  9  6]]
b= [0 5 8 1]
```

0.0.4 4) Tableaux de tailles différentes

Créer un array numpy nommé `a` de taille 3x3 contenant des valeurs aléatoires entières entre 1 et 9 comprises.

Créer un second array nommé `b` de taille 1x3 également rempli de valeurs aléatoires entières entre 1 et 9 comprises.

Créer un troisième array nommé `c` de taille 3x1 également rempli de valeurs aléatoires entières entre 1 et 9 comprises.illes différentes

[15]:
```
# A compléter ...
```

```
a = [[2 5 6]
 [7 8 1]
 [4 3 9]]
b = [[5 8 8]]
c= [[8]
 [8]
 [5]]
```

Ajouter la constante 5 à a, afficher le résultat :

[16]:
```
# A compléter ...
```

```
[[ 7 10 11]
 [12 13  6]
 [ 9  8 14]]
```

Calculer (a+b). Que représente ce calcul?

Calculer (a+c). Que représente ce calcul?

Afficher b.T

En déduire une façon d'ajouter b à chaque colonne de a

[17]:
```
# A compléter ...
```

```
a = [[2 5 6]
 [7 8 1]
 [4 3 9]]
b = [[5 8 8]]
c= [[8]
 [8]
 [5]]
a+b [[ 7 13 14]
 [12 16  9]
 [ 9 11 17]]
a+c [[10 13 14]
 [15 16  9]
 [ 9  8 14]]
b.T [[5]
 [8]
 [8]]
a+b.T [[ 7 10 11]
 [15 16  9]
 [12 11 17]]
a = [[2 5 6]
 [7 8 1]
 [4 3 9]]
b = [[5 8 8]]
c= [[8]
 [8]
 [5]]
```

Calculer la somme b+c, afficher le résultat et le commenter.

[18]:
```
# A compléter ...
```

```
b:
 [[5 8 8]]
c:
 [[8]
 [8]
 [5]]
b+c:
 [[13 16 16]
 [13 16 16]
 [10 13 13]]
```

0.0.5 5. Exercice récapitulatif

On regroupe dans une liste de listes les notes de 50 étudiants. Il y a 20 notes par étudiant.

[19]:
```
tabNotes=[[ 4,  4,  13, 9, 11,  8, 11, 8,  0,  6, 6,  12,  5,  9,  6,  8,
         3, 11, 10, 11],
       [5, 11, 9,  2,  6,  7, 6,  6, 11, 11,  1, 10, 11, 7, 14, 10,
         9, 12,  7,  5],
       [15, 16,  12,  10, 13, 10, 11, 14,  6,  6,  13,  8, 11, 12, 10,  9,
        10,  12, 12,  12],
       [12, 17, 14,  13, 12, 17,  13, 14,  15, 16, 15,  15,  12,  16,  13, 20,
         12, 14,  14, 19],
       [6,  7,  5, 13,  6,  9,  10,  10,  9,  8, 7,  0, 6,  6, 13, 13,
        10,  4, 16, 13],
       [ 5,  4,  6,  9, 11, 13,  6,  7, 10,  8,  6, 13, 11, 11, 11, 15,
        10, 10,  9,  8],
       [ 11, 7, 12, 10, 10,  6,  5, 8,  10,  3, 9, 11,  8,  7, 11,  4,
         6, 12,  4,  7],
       [10, 10, 11,  5,  9, 10, 16,  5,  5,  4, 13, 14, 12,  2, 16,  5,
        11,  8,  9,  6],
       [ 2, 5, 7,  3,  4, 16, 12,  8,  0, 9, 10, 10,  9,  3,  4,  3,
        10, 7, 12, 10],
       [ 4, 10, 13,  8,  8,  7, 10,  3,  8,  6,  0, 15, 12, 11, 12, 15,
        11, 13, 15,  4],
       [15,  11,  10, 16, 14,  17, 19, 17, 16,  16, 17, 14, 12,  17,  12, 14,
         17, 17,  13,  12],
       [6, 7,  4, 14,  8, 11,  9,  7, 12,  6,  8, 11,  10, 9,  13, 9,
         12,  9, 10, 10],
       [20, 14, 17, 17, 16,  13, 17, 15,  17, 14,  10, 17,  14,  13,  19, 15,
         13, 19, 15,  18],
       [ 3, 14, 11,  3,  11,  8,  2, 11,  4,  5, 12, 17,  3,  3,  9, 13,
        12,  3,  5, 10],
       [ 5, 10, 10, 11, 9, 12,  0,  5,  2, 12, 9,  9,  1,  4, 10,  9,
        10,  1, 10, 13],
       [10,  8,  8, 19, 13,  6,  5, 16, 16,  10, 17,  10,  3, 15, 13, 14,
         5, 14, 12,  9],
       [ 9, 10, 10, 10, 13, 10, 12,  7, 10, 10, 11, 15,  5,  7,  0,  7,
        12, 10, 13, 13],
       [ 6,  7,  2, 11,  1, 11,  11, 11,  6,  0, 10, 14, 12,  12,  9,  13,
        18,  10, 18, 20],
       [14,  13, 19,  10,  17, 15,  10,  15,  12, 13,  7,  8,  13, 11,  9, 18,
        13, 13,  5, 18],
       [4,  1,  6, 17,  4,  7,  5, 19, 11,  5,  7, 18, 16,  8,  9, 11,
        10, 10,  1, 12],
       [12, 12,  12,  6,  6, 15, 11, 12, 12, 10, 11, 11, 12, 11,  9, 11,
         5, 7,  12, 12],
       [ 7, 5, 13, 11,  9, 11,  6, 10, 12, 10, 17, 10,  5,  4,  9, 11,
        10,  3, 11,  5],
       [10,  6, 10,  4, 8,  4,  5,  4,  4, 8, 7, 9,  4, 9, 5, 8,
         4,  2,  7,  3],
       [14,  9,  3, 13,  5, 11,  6, 12, 10,  7, 12,  6, 13,  8,  0,  5,
```

```
 14, 13, 12, 11],
[10,  4,  5,  3,  7, 15,  8,  0,  7,  8,  0,  5,  4, 12,  8,  4,
 13, 12, 14, 11],
[11,  8, 11, 11,  10, 14,  2, 11, 10, 12,  10,  9, 10,  8,  9, 12,
 10, 10, 7,  7],
[ 5,  4, 5,  13, 7,  7,  9, 10,  6, 11, 13, 12,  8,  9,  3,  0,
 10,  9, 11, 14],
[11, 11, 11,  10,  10, 10,  7,  12,  7, 14,  6,  13, 11,  12, 8, 7,
 12,  12,  8, 11],
[11, 10,  8,  6, 11,  8,  8, 14,  11, 12,  12, 11,  11,  10, 10, 8,
  13,  9,  5, 11],
[17, 17, 14, 15,  13, 12, 15,  15, 15, 16, 16, 12,  14, 15, 14, 11,
  6, 13,  12, 10],
[16, 13, 15, 13,  11,  10, 14,  12,  10,  12, 16,  16,  16,  16, 10, 10,
  10,  11,  11,  11],
[12, 12,  8,  8,  9,  9, 15,  9,  9, 9,  9,  7, 13, 10, 9, 14,
  5, 12,  10,  5],
[16,  9,  10,  8, 17, 13, 16, 13,  7,  9, 17, 10, 16, 11,  7, 11,
 10,  6, 12, 12],
[10,  5,  5,  6,  7,  2, 10, 13, 17, 12, 10, 12, 16,  10, 13,  5,
 10,  7, 10,  8],
[ 3, 12,  8, 9, 9, 12,  2, 12,  3,  7,  1, 10,  6, 9,  5,  7,
  2,  5, 15, 10],
[14, 12,  9, 12,  9, 8, 12, 10,  9, 13, 13,  8,  2, 13,  11,  11,
  13, 7,  11,  11],
[ 11, 8,  9, 14, 16,  3, 16,  4,  5, 13, 12,  11, 9,  7, 7, 16,
 14, 10, 16,  6],
[11,  4,  5,  10,  4,  5, 10, 10, 14,  4, 12,  6,  7,  9, 16,  8,
 10, 9, 9,  13],
[16, 18, 10, 18, 11, 12, 19, 13,  4,  6,  16, 17, 17, 10,  7, 16,
 10, 10,  3,  1],
[ 7, 12,  9, 12, 9,  12,  4, 13, 13,  3,  12,  9, 14, 19, 12,  0,
  3, 6, 13, 13],
[14, 10, 10,  9,  6, 12, 18, 18, 14, 14, 10, 18,  10, 10,  10,  12,
 19,  5, 14,  0],
[20, 20,  1,  7,  5, 15,  5, 8, 9,  8,  7, 8, 8, 16,  0,  3,
  3, 13,  3,  6],
[ 4,  6,  1, 2, 13,  2, 9, 6, 10, 12, 12,  2,  4,  10, 14,  10,
  6, 15, 10,  5],
[ 3, 7,  3,  4, 10,  3, 10, 13,  7,  5,  3,  4, 2,  8, 11,  8,
  9,  6,  7,  3],
[20, 14,  9, 13,  6, 5, 8,  4,  4, 11,  6,  11,  12, 11, 10,  4,
  7, 8,  9,  11],
[ 8, 10,  2,  8,  5, 10,  10,  0, 7,  7,  6, 14, 13,  4, 7,  8,
  6, 8, 12,  3],
[ 8, 12,  7,  9, 14, 12, 12, 12, 13, 13,  9,  7,  4, 10,  8, 11,
  6, 15,  0, 15],
```

```
       [ 7, 11,  7, 11,  6,  7, 10,  8,  1, 14, 12, 13, 10,  8, 15, 12,
        7, 10, 14,  4],
       [18,  9, 14,  5, 14,  6, 12,  8,  12,  10, 9,  8, 9,  0,  4, 15,
        11, 10,  12,  7],
       [ 20,  6, 10,  12,  5,  4,  13,  8, 12, 12,  13, 12, 17,  8,  9,  0,
        10,  13,  12,  13]]
```

1.Transformer ce tableau en array numpy. Vérifier qu'on a bien un tableau à 50 lignes et 20 colonnes.

Calculer la moyenne globale des toutes les notes puis la moyenne des toues les notes non nulles.

[20]:
```
# A compléter ...
```

```
moyenne globale :  9.714
moyenne sans les 0:  9.892057026476579
```

2.a. Calculer le tableau des 50 moyennes des des étudiants à chaque test.

b. Calculer le tableau des 20 notes moyennes pour chaque des 20 tests.

[21]:
```
# A compléter ...
```

```
moyennes par étudiants [ 7.75  8.   11.1  14.65  8.55  9.15  8.05  9.05  7.2
9.25 14.8   9.25
 15.65  7.95  7.6  11.15  9.7  10.1  12.65  9.05 10.45  8.95  6.05  9.2
  7.5   9.6   8.3  10.15  9.95 13.6  12.65  9.7  11.5   9.4   7.35 10.4
 10.35  8.8  11.7   9.75 11.65  8.25  7.65  6.3   9.15  7.4   9.85  9.35
  9.65 10.45]
moyennes par devoir [10.24  9.64  8.86  9.84  9.36  9.64  9.84 10.    9.08  9.4
9.94 10.84
  9.66  9.6   9.46  9.66  9.64  9.7  10.24  9.64]
```

3.a. Pour chacun des 20 tests, compter combien d'étudiants ont la moyenne et afficher les résultats

3.b. Compter pour chaque étudiant combien de fois il a eu la moyenne

[22]:
```
# A compléter ...
```

```
pour chaque test: nombre d'étudiants ayant la moyenne
  [29 27 24 28 23 28 29 29 26 26 29 33 29 26 25 27 33 29 33 30]

pour chaque étudiant: nombre de tests où il a eu la moyenne
 [ 7  8 16 20  8 10  8 10  7 11 20  9 20  9  9 13 14 13 16  9 15 11  2 11
  7 13  8 14 13 19 20  8 14 12  6 13 11  9 15 11 16  5  9  4  9  6 11 11
 10 13]
```

4. Afficher le numéro de l'étudiant qui a la meilleure moyenne sur les 20 tests et celui qui a la moins bonne (ainsi que les moyennes correspondantes).

[23]:
```
# A compléter ...
```

```
c'est l'étudiant  12 qui a la meilleure moyenne avec 15.65
c 'est l'étudiant  22 qui a la plus faible moyenne avec 6.05
```

5. Calculer et afficher le numéro de l'étudiant qui a le plus de notes supérieures à 14.

[24]:
```
# A compléter ...
```

```
c'est l'étudiant  12 qui a le plus de notes supérieures à 14 avec 16 notes
supérieures à 14
```

6. On veut corriger quelques erreurs de saisie

- l'étudiant de la ligne d'indice 5 a dix points de plus pour sa dernière note
- pour le test de la première colonne (colonne 0) , il faut ajouter 2 points à tous les étudiants
- ensuite il faut remettre à 20 les notes qui dépasseraient 20

[25]:
```
# A compléter ...
```

```
[[ 6  4 13  9 11  8 11  8  0  6  6 12  5  9  6  8  3 11 10 11]
 [ 7 11  9  2  6  7  6  6 11 11  1 10 11  7 14 10  9 12  7  5]
 [17 16 12 10 13 10 11 14  6  6 13  8 11 12 10  9 10 12 12 12]
 [14 17 14 13 12 17 13 14 15 16 15 15 12 16 13 20 12 14 14 19]
 [ 8  7  5 13  6  9 10 10  9  8  7  0  6  6 13 13 10  4 16 13]
 [ 7  4  6  9 11 13  6  7 10  8  6 13 11 11 11 15 10 10  9 18]
 [13  7 12 10 10  6  5  8 10  3  9 11  8  7 11  4  6 12  4  7]
 [12 10 11  5  9 10 16  5  5  4 13 14 12  2 16  5 11  8  9  6]
 [ 4  5  7  3  4 16 12  8  0  9 10 10  9  3  4  3 10  7 12 10]
 [ 6 10 13  8  8  7 10  3  8  6  0 15 12 11 12 15 11 13 15  4]
 [17 11 10 16 14 17 19 17 16 16 17 14 12 17 12 14 17 17 13 12]
 [ 8  7  4 14  8 11  9  7 12  6  8 11 10  9 13  9 12  9 10 10]
 [20 14 17 17 16 13 17 15 17 14 10 17 14 13 19 15 13 19 15 18]
 [ 5 14 11  3 11  8  2 11  4  5 12 17  3  3  9 13 12  3  5 10]
 [ 7 10 10 11  9 12  0  5  2 12  9  9  1  4 10  9 10  1 10 13]
 [12  8  8 19 13  6  5 16 16 10 17 10  3 15 13 14  5 14 12  9]
 [11 10 10 10 13 10 12  7 10 10 11 15  5  7  0  7 12 10 13 13]
 [ 8  7  2 11  1 11 11 11  6  0 10 14 12 12  9 13 18 10 18 20]
 [16 13 19 10 17 15 10 15 12 13  7  8 13 11  9 18 13 13  5 18]
 [ 6  1  6 17  4  7  5 19 11  5  7 18 16  8  9 11 10 10  1 12]
 [14 12 12  6  6 15 11 12 12 10 11 11 12 11  9 11  5  7 12 12]
 [ 9  5 13 11  9 11  6 10 12 10 17 10  5  4  9 11 10  3 11  5]
 [12  6 10  4  8  4  5  4  4  8  7  9  4  9  5  8  4  2  7  3]
 [16  9  3 13  5 11  6 12 10  7 12  6 13  8  0  5 14 13 12 11]
 [12  4  5  3  7 15  8  0  7  8  0  5  4 12  8  4 13 12 14 11]
 [13  8 11 11 10 14  2 11 10 12 10  9 10  8  9 12 10 10  7  7]
 [ 7  4  5 13  7  7  9 10  6 11 13 12  8  9  3  0 10  9 11 14]
 [13 11 11 10 10 10  7 12  7 14  6 13 11 12  8  7 12 12  8 11]
 [13 10  8  6 11  8  8 14 11 12 12 11 11 10 10  8 13  9  5 11]
 [19 17 14 15 13 12 15 15 15 16 16 12 14 15 14 11  6 13 12 10]
 [18 13 15 13 11 10 14 12 10 12 16 16 16 16 10 10 10 11 11 11]
 [14 12  8  8  9  9 15  9  9  9  9  7 13 10  9 14  5 12 10  5]
 [18  9 10  8 17 13 16 13  7  9 17 10 16 11  7 11 10  6 12 12]
 [12  5  5  6  7  2 10 13 17 12 10 12 16 10 13  5 10  7 10  8]
 [ 5 12  8  9  9 12  2 12  3  7  1 10  6  9  5  7  2  5 15 10]]
```

```
[16 12  9 12  9  8 12 10  9 13 13  8  2 13 11 11 13  7 11 11]
[13  8  9 14 16  3 16  4  5 13 12 11  9  7  7 16 14 10 16  6]
[13  4  5 10  4  5 10 10 14  4 12  6  7  9 16  8 10  9  9 13]
[18 18 10 18 11 12 19 13  4  6 16 17 17 10  7 16 10 10  3  1]
[ 9 12  9 12  9 12  4 13 13  3 12  9 14 19 12  0  3  6 13 13]
[16 10 10  9  6 12 18 18 14 14 10 18 10 10 10 12 19  5 14  0]
[20 20  1  7  5 15  5  8  9  8  7  8  8 16  0  3  3 13  3  6]
[ 6  6  1  2 13  2  9  6 10 12 12  2  4 10 14 10  6 15 10  5]
[ 5  7  3  4 10  3 10 13  7  5  3  4  2  8 11  8  9  6  7  3]
[20 14  9 13  6  5  8  4  4 11  6 11 12 11 10  4  7  8  9 11]
[10 10  2  8  5 10 10  0  7  7  6 14 13  4  7  8  6  8 12  3]
[10 12  7  9 14 12 12 12 13 13  9  7  4 10  8 11  6 15  0 15]
[ 9 11  7 11  6  7 10  8  1 14 12 13 10  8 15 12  7 10 14  4]
[20  9 14  5 14  6 12  8 12 10  9  8  9  0  4 15 11 10 12  7]
[20  6 10 12  5  4 13  8 12 12 13 12 17  8  9  0 10 13 12 13]]
```

7. Extraire et afficher les résultats du test 5 (colonne d'indice 4)

[26]:
```
# A compléter ...
```

```
[11  6 13 12  6 11 10  9  4  8 14  8 16 11  9 13 13  1 17  4  6  9  8  5
  7 10  7 10 11 13 11  9 17  7  9  9 16  4 11  9  6  5 13 10  6  5 14  6
 14  5]
```

8. Utiliser np.unique pour voir quelle est la note la plus souvent obtenue à ce test et quelles notes ne sont pas obtenues.

[27]:
```
# A compléter ...
```

```
(array([ 1,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 16, 17]), array([1, 3,
4, 6, 3, 3, 7, 4, 6, 1, 5, 3, 2, 2]))
```

9. Utiliser np.where pour récupérer les numéros des étudiants qui ont eu plus de 12 ce test.

[28]:
```
# A compléter ...
```

```
(array([ 2,  3, 10, 12, 15, 16, 18, 29, 32, 36, 42, 46, 48]),)
```