DM_2048_Corrige

December 29, 2019

0.1 Question 2

```
In [36]: from random import randint, random
```

0.2 Question 3

```
In [9]: def grille_remplie(val):
            grille = []
            for lig in range(4):
                ligne = []
                for col in range(4):
                    ligne.append(val)
                grille.append(ligne)
            return grille
```

```
In [67]: grille_remplie(734)
```

```
Out[67]: [[734, 734, 734, 734],
          [734, 734, 734, 734],
          [734, 734, 734, 734],
          [734, 734, 734, 734]]
```

```
In [10]: def afficher_grille(grille):
             for lig in range(4):
                 for col in range(4):
                     print('|{:^6d}'.format(grille[lig][col]),end='')
                 print('|')
```

```
In [69]: afficher_grille(grille_remplie(2048))
```

```
| 2048 | 2048 | 2048 | 2048 |
| 2048 | 2048 | 2048 | 2048 |
| 2048 | 2048 | 2048 | 2048 |
| 2048 | 2048 | 2048 | 2048 |
```

```
In [70]: afficher_grille([[randint(0,2048) for _ in range(4)] for _ in range(4)])
```

```
| 1955 | 846  | 1651 | 299  |
| 373  | 764  | 1769 | 1009 |
| 1995 | 895  | 1710 | 1729 |
| 767  |  37  | 787  | 124  |
```

0.3 Questions 4 et 5

```
In [8]: def deplacer_gauche(grille, nlig, verbose = True):
            """Décale vers la gauche tous les nombres non nuls
            de la ligne nlig de grille"""
            fin_zero = 0
            debut_zero = 0
            k = 1
            while fin_zero < 4:
                if verbose:
                    print("Tour ", k)
                    #Point d'arret 1
                    print("Point d'arret 1", "debut_zero=", debut_zero, "fin_zero=", fin_zero)
                while debut_zero < 4 and grille[nlig][debut_zero] != 0:
                    debut_zero = debut_zero + 1
                if debut_zero == 4:
                    fin_zero = 4
                else:
                    fin_zero = debut_zero + 1
                #Point d'arret 2
                if verbose:
                    print("Point d'arret 2", "debut_zero=", debut_zero, "fin_zero=", fin_zero)
                while fin_zero < 4 and grille[nlig][fin_zero] == 0:
                    fin_zero = fin_zero + 1
                if fin_zero < 4:
                    grille[nlig][fin_zero], grille[nlig][debut_zero] = grille[nlig][debut_zero],
                #Point d'arret 3
                if verbose:
                    print("Point d'arret 3", "debut_zero=", debut_zero, "fin_zero=", fin_zero)
                k = k + 1
```

```
In [72]: # Test
         grille_test = [[0,4,0,2],[0,0,0,0,],[0,0,0,0], [0,0,0,0]]
         deplacer_gauche(grille_test, 0)
         print(grille_test)
```

```
Tour  1
Point d'arret 1 debut_zero= 0 fin_zero= 0
Point d'arret 2 debut_zero= 0 fin_zero= 1
Point d'arret 3 debut_zero= 0 fin_zero= 1
Tour  2
Point d'arret 1 debut_zero= 0 fin_zero= 1
```

```
Point d'arret 2 debut_zero= 1 fin_zero= 2
Point d'arret 3 debut_zero= 1 fin_zero= 3
Tour  3
Point d'arret 1 debut_zero= 1 fin_zero= 3
Point d'arret 2 debut_zero= 2 fin_zero= 3
Point d'arret 3 debut_zero= 2 fin_zero= 4
[[4, 2, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
```

0.4 Question 6

```
In [7]: def deplacer_droite(grille, nlig, verbose = True):
            """Décale vers la droite tous les nombres non nuls
            de la ligne nlig de grille"""
            fin_zero = 3
            debut_zero = 3
            k = 1
            while fin_zero >= 0:
                if verbose:
                    print("Tour ", k)
                    #Point d'arret 1
                    print("Point d'arret 1", "debut_zero=", debut_zero, "fin_zero=", fin_zero)
                while debut_zero >= 0 and grille[nlig][debut_zero] != 0:
                    debut_zero = debut_zero - 1
                if debut_zero == -1:
                    fin_zero = -1
                else:
                    fin_zero = debut_zero - 1
                #Point d'arret 2
                if verbose:
                    print("Point d'arret 2", "debut_zero=", debut_zero, "fin_zero=", fin_zero)
                while fin_zero >= 0 and grille[nlig][fin_zero] == 0:
                    fin_zero = fin_zero - 1
                if fin_zero >= 0:
                    grille[nlig][fin_zero], grille[nlig][debut_zero] = grille[nlig][debut_zero],
                #Point d'arret
                if verbose:
                    print("Point d'arret 3", "debut_zero=", debut_zero, "fin_zero=", fin_zero)
                k = k + 1

In [74]: # Test
         grille_test = [[4,0,2,0],[0,0,0,0],[0,0,0,0], [0,0,0,0]]
         deplacer_droite(grille_test, 0)
         print(grille_test)

Tour  1
Point d'arret 1 debut_zero= 3 fin_zero= 3
Point d'arret 2 debut_zero= 3 fin_zero= 2
```

```
Point d'arret 3 debut_zero= 3 fin_zero= 2
Tour  2
Point d'arret 1 debut_zero= 3 fin_zero= 2
Point d'arret 2 debut_zero= 2 fin_zero= 1
Point d'arret 3 debut_zero= 2 fin_zero= 0
Tour  3
Point d'arret 1 debut_zero= 2 fin_zero= 0
Point d'arret 2 debut_zero= 1 fin_zero= 0
Point d'arret 3 debut_zero= 1 fin_zero= -1
[[0, 0, 4, 2], [0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
```

0.5 Question 7

```
In [6]: def transpose(grille):
            grille_trans = grille_remplie(0)
            for lig in range(4):
                for col in range(4):
                    grille_trans[lig][col] = grille[col][lig]
            return grille_trans
```

0.6 Question 8

```
In [5]: def decaler_haut(grille, ncol):
            """Retourne une nouvelle grille où tous les nombres non nuls
            de la colonne ncol sont décalés versla haut"""
            grille_trans = transpose(grille)
            deplacer_gauche(grille_trans, ncol, verbose = False)
            return transpose(grille_trans)
```

```
In [77]: #Test
         grille1 = [[2,0,4,4], [0,2,0,0], [4,2,0,4], [0,4,2,8]]
         print("avant", grille1)
         grille2 = decaler_haut(grille1, 0)
         print("après décalage vers le haut de la colonne 0", grille2)
```

```
avant [[2, 0, 4, 4], [0, 2, 0, 0], [4, 2, 0, 4], [0, 4, 2, 8]]
après décalage vers le haut de la colonne 0 [[2, 0, 4, 4], [4, 2, 0, 0], [0, 2, 0, 4], [0, 4, 2,
```

0.7 Question 9

```
In [4]: def decaler_bas(grille, ncol):
            """Retourne une nouvelle grille où tous les nombres non nuls
            de la colonne ncol sont décalés versla haut"""
            grille_trans = transpose(grille)
            deplacer_droite(grille_trans, ncol, verbose = False)
            return transpose(grille_trans)
```

```
In [79]: #Test
         grille1 = [[2,0,4,4], [0,2,0,0], [4,2,0,4], [0,4,2,8]]
         print("avant", grille1)
         grille2 = decaler_bas(grille1, 0)
         print("après décalage vers le bas de la colonne 0", grille2)
```

```
avant [[2, 0, 4, 4], [0, 2, 0, 0], [4, 2, 0, 4], [0, 4, 2, 8]]
après décalage vers le bas de la colonne 0 [[0, 0, 4, 4], [0, 2, 0, 0], [2, 2, 0, 4], [4, 4, 2,
```

0.8 Question 10

```
In [14]: def fusion_ligne_gauche(grille, nlig):
             """Balaie la ligne nlig de grille à partir de la dernière colonne à droite(position
             et fusionne les cases voisines identiques.
             Travaille sur une ligne où les zéros et les autres valeurs sont déjà séparées"""
             deplacer_gauche(grille, nlig, verbose = False)
             ncol = 0
             while ncol < 3:
                 if grille[nlig][ncol] != 0 and grille[nlig][ncol + 1] == grille[nlig][ncol]:
                     grille[nlig][ncol + 1] = 2 * grille[nlig][ncol]
                     grille[nlig][ncol] = 0
                     ncol = ncol + 1
                 ncol = ncol + 1
             deplacer_gauche(grille, nlig, verbose = False)
```

```
In [15]: #Tests
         grille = [[4,2,2,0], [8,4,4,2], [8,8,8,2], [4,4,2,0]]
         print("Grille initiale :", grille)
         for k in range(4):
             print("Fusion ligne gauche", k)
             fusion_ligne_gauche(grille, k)
             print(grille)
```

```
Grille initiale : [[4, 2, 2, 0], [8, 4, 4, 2], [8, 8, 8, 2], [4, 4, 2, 0]]
Fusion ligne gauche 0
[[4, 4, 0, 0], [8, 4, 4, 2], [8, 8, 8, 2], [4, 4, 2, 0]]
Fusion ligne gauche 1
[[4, 4, 0, 0], [8, 8, 2, 0], [8, 8, 8, 2], [4, 4, 2, 0]]
Fusion ligne gauche 2
[[4, 4, 0, 0], [8, 8, 2, 0], [16, 8, 2, 0], [4, 4, 2, 0]]
Fusion ligne gauche 3
[[4, 4, 0, 0], [8, 8, 2, 0], [16, 8, 2, 0], [8, 2, 0, 0]]
```

0.9 Question 11

```
In [21]: def fusion_ligne_droite(grille, nlig):
             """Balaie la ligne nlig de grille à partir de la dernière colonne à gauche (positio
```

```
            et fusionne les cases voisines identiques.
            Travaille sur une ligne où les zéros et les autres valeurs sont déjà séparées"""
            ncol = 3
            deplacer_droite(grille, nlig, verbose = False)
            while ncol > 0:
                if grille[nlig][ncol] != 0 and grille[nlig][ncol  - 1] == grille[nlig][ncol]:
                    grille[nlig][ncol - 1] = 2 * grille[nlig][ncol]
                    grille[nlig][ncol] = 0
                    ncol = ncol - 1
                ncol = ncol - 1
            deplacer_droite(grille, nlig, verbose = False)
```

```
In [22]: #Tests
         grille = [[0, 4,2,2], [0,4,4,8], [0, 8,8,8], [0, 2,4,4]]
         print("Grille initiale :", grille)
         for k in range(4):
             print("Fusion ligne droite", k)
             fusion_ligne_droite(grille, k)
             print(grille)
```

```
Grille initiale : [[0, 4, 2, 2], [0, 4, 4, 8], [0, 8, 8, 8], [0, 2, 4, 4]]
Fusion ligne droite 0
[[0, 0, 4, 4], [0, 4, 4, 8], [0, 8, 8, 8], [0, 2, 4, 4]]
Fusion ligne droite 1
[[0, 0, 4, 4], [0, 0, 8, 8], [0, 8, 8, 8], [0, 2, 4, 4]]
Fusion ligne droite 2
[[0, 0, 4, 4], [0, 0, 8, 8], [0, 0, 8, 16], [0, 2, 4, 4]]
Fusion ligne droite 3
[[0, 0, 4, 4], [0, 0, 8, 8], [0, 0, 8, 16], [0, 0, 2, 8]]
```

0.10 Question 12

```
In [24]: def fusion_col_haut(grille, ncol):
             """Balaie la colonne ncol vers le haut
             et fusionne les cases voisines identiques.
             Travaille sur une colonne où les zéros et les autres valeurs sont déjà séparées.
             Retourne une nouvelle grille"""
             grille_trans = transpose(grille)
             fusion_ligne_gauche(grille_trans, ncol)
             return transpose(grille_trans)
```

```
In [26]: #Tests
         grille3 = [[8,8,4,8], [0,2,0,8], [4,2,0,8], [0,4,2,0]]
         print("Grille initiale :", grille3)
         print("Fusion colonne 3 vers le haut", fusion_col_haut(grille3, 3))
```

```
Grille initiale : [[8, 8, 4, 8], [0, 2, 0, 8], [4, 2, 0, 8], [0, 4, 2, 0]]
Fusion colonne 3 vers le haut [[8, 8, 4, 16], [0, 2, 0, 8], [4, 2, 0, 0], [0, 4, 2, 0]]
```

```
In [27]: def fusion_col_bas(grille, ncol):
             """Balaie la colonne ncol vers le haut
             et fusionne les cases voisines identiques.
             Travaille sur une colonne où les zéros et les autres valeurs sont déjà séparées.
             Retourne une nouvelle grille"""
             grille_trans = transpose(grille)
             fusion_ligne_droite(grille_trans, ncol)
             return transpose(grille_trans)

In [28]: #Tests
         grille4 = [[8,8,4,0], [0,2,0,4], [4,2,0,4], [0,4,2,4]]
         print("Grille initiale :", grille4)
         print("Fusion colonne 3 vers le bas", fusion_col_bas(grille4, 3))

Grille initiale : [[8, 8, 4, 0], [0, 2, 0, 4], [4, 2, 0, 4], [0, 4, 2, 4]]
Fusion colonne 3 vers le bas [[8, 8, 4, 0], [0, 2, 0, 0], [4, 2, 0, 4], [0, 4, 2, 8]]
```

0.11 Question 13

```
In [32]: def liste_case_vide(grille):
             L = []
             for lig in range(4):
                 for col in range(4):
                     if grille[lig][col] == 0:
                         L.append((lig, col))
             return L
```

0.12 Question 14

```
In [33]: def maximum(grille):
             maxi = 0
             for lig in range(3):
                 for col in range(3):
                     n = grille[lig][col]
                     if n  > maxi:
                         maxi = n
             return maxi
```

0.13 Question 15

```
In [39]: def jeu_2048():
             #initialisation
             urne = [2, 4]
             grille = grille_remplie(0)
             case_vide = liste_case_vide(grille)
             while len(case_vide) > 0 and maximum(grille) < 2048:
                 (lig, col) = case_vide[randint(0, len(case_vide) - 1)]
                 grille[lig][col] = urne[randint(0, 1)]
```

```
                afficher_grille(grille)
                print('-'*9 + 'Nouveau tour' + '-'*9)
                deplacement = int(input("Déplacement 4 (gauche) 6 (droite)  8 (haut) 9 (bas) ? "))
                #vérification sommaire de l'entrée
                assert deplacement in [4,6,8,9]
                if deplacement == 4:
                    for lig in range(4):
                        deplacer_gauche(grille, lig, verbose = False)
                        fusion_ligne_gauche(grille, lig)
                elif deplacement == 6:
                    for lig in range(4):
                        deplacer_droite(grille, lig, verbose = False)
                        fusion_ligne_droite(grille, lig)
                elif deplacement == 9:
                    for col in range(4):
                        grille = decaler_bas(grille, col)
                        grille = fusion_col_bas(grille, col)
                else:
                    for col in range(4):
                        grille = decaler_haut(grille, col)
                        grille = fusion_col_haut(grille, col)
                case_vide = liste_case_vide(grille)
            afficher_grille(grille)
```

In [ ]:
```
jeu_2048()
```

```
|  0   |  0   |  0   |  0   |
|  0   |  2   |  0   |  0   |
|  0   |  0   |  0   |  0   |
|  0   |  0   |  0   |  0   |
---------Nouveau tour---------
Déplacement 4 (gauche) 6 (droite)  8 (haut) 9 (bas) ?  4
|  0   |  0   |  0   |  2   |
|  2   |  0   |  0   |  0   |
|  0   |  0   |  0   |  0   |
|  0   |  0   |  0   |  0   |
---------Nouveau tour---------
Déplacement 4 (gauche) 6 (droite)  8 (haut) 9 (bas) ?  9
|  2   |  0   |  0   |  0   |
|  0   |  0   |  0   |  0   |
|  0   |  0   |  0   |  0   |
|  2   |  0   |  0   |  2   |
---------Nouveau tour---------
Déplacement 4 (gauche) 6 (droite)  8 (haut) 9 (bas) ?  9
|  0   |  0   |  4   |  0   |
|  0   |  0   |  0   |  0   |
|  0   |  0   |  0   |  0   |
|  4   |  0   |  0   |  2   |
```

```
---------Nouveau tour---------
```