

# BACCALAURÉAT

SESSION 2025

---

Épreuve de l'enseignement de spécialité

## NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

---

Sujet n°43

---

DURÉE DE L'ÉPREUVE : 1 heure

**Le sujet comporte 3 pages numérotées de 1 / 3 à 3 / 3  
Dès que le sujet vous est remis, assurez-vous qu'il est complet.**

*Le candidat doit traiter les 2 exercices.*

## EXERCICE 1 (10 points)

Écrire une fonction `couples_consecutifs` qui prend en paramètre un tableau de nombres entiers `tab` non vide (type `list`), et qui renvoie la liste Python (éventuellement vide) des couples d'entiers consécutifs successifs qu'il peut y avoir dans `tab`.

Exemples :

```
>>> couples_consecutifs([1, 4, 3, 5])
[]
>>> couples_consecutifs([1, 4, 5, 3])
[(4, 5)]
>>> couples_consecutifs([1, 1, 2, 4])
[(1, 2)]
>>> couples_consecutifs([7, 1, 2, 5, 3, 4])
[(1, 2), (3, 4)]
>>> couples_consecutifs([5, 1, 2, 3, 8, -5, -4, 7])
[(1, 2), (2, 3), (-5, -4)]
```

## EXERCICE 2 (10 points)

Soit une image binaire représentée dans un tableau à 2 dimensions. Les éléments  $M[i][j]$ , appelés pixels, sont égaux soit à 0 soit à 1.

Une composante d'une image est un sous-ensemble de l'image constitué uniquement de 1 et de 0 qui sont côte à côte, soit horizontalement soit verticalement.

Par exemple, les composantes de

M =	<table border="1"><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr></table>	0	0	1	0	0	1	0	1	1	1	1	0	0	1	1	0
0	0	1	0														
0	1	0	1														
1	1	1	0														
0	1	1	0														

sont

0	0	1	0
0	1	0	1
1	1	1	0
0	1	1	0

On souhaite, à partir d'un pixel égal à 1 dans une image  $M$ , donner la valeur  $val$  à tous les pixels de la composante à laquelle appartient ce pixel.

La fonction `colore_comp1` prend pour paramètre une image  $M$  (représentée par une liste de listes), deux entiers  $i$  et  $j$  et une valeur entière  $val$ . Elle met à la valeur  $val$  tous les pixels de la composante du pixel  $M[i][j]$  s'il vaut 1 et ne fait rien sinon.

M =	<table border="1"><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>3</td><td>0</td><td>1</td></tr><tr><td>3</td><td><b>3</b></td><td>3</td><td>0</td></tr><tr><td>0</td><td>3</td><td>3</td><td>0</td></tr></table>	0	0	1	0	0	3	0	1	3	<b>3</b>	3	0	0	3	3	0
0	0	1	0														
0	3	0	1														
3	<b>3</b>	3	0														
0	3	3	0														

Par exemple, `colore_comp1(M, 2, 1, 3)` donne

Compléter le code récursif de la fonction `colore_comp1` donné ci-dessous :

```
def colore_comp1(M, i, j, val):
    if M[i][j] != 1:
        return

    M[i][j] = val

    if i-1 >= 0: # propage en haut
        colore_comp1(M, i-1, j, val)
    if ... < len(M): # propage en bas
        colore_comp1(M, ..., j, val)
    if ...: # propage à gauche
        colore_comp1(M, ..., ..., val)
    if ...: # propage à droite
        ...
```

Exemple :

```
>>> M = [[0, 0, 1, 0], [0, 1, 0, 1], [1, 1, 1, 0], [0, 1, 1, 0]]
>>> colore_comp1(M, 2, 1, 3)
>>> M
[[0, 0, 1, 0], [0, 3, 0, 1], [3, 3, 3, 0], [0, 3, 3, 0]]
```