

# BACCALAURÉAT

SESSION 2025

---

Épreuve de l'enseignement de spécialité

## NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

---

Sujet n°14

---

DURÉE DE L'ÉPREUVE : 1 heure

**Le sujet comporte 5 pages numérotées de 1 / 5 à 5 / 5  
Dès que le sujet vous est remis, assurez-vous qu'il est complet.**

*Le candidat doit traiter les 2 exercices.*

## EXERCICE 1 (10 points)

Dans cet exercice les tableaux sont représentés par des listes Python (type `list`).

Écrire en python deux fonctions :

- `lancer` de paramètre `n`, un entier positif, qui renvoie un tableau de `n` entiers obtenus aléatoirement entre 1 et 6 (1 et 6 inclus) ;
- `paire_6` de paramètre `tab`, un tableau de `n` entiers compris entre 1 et 6 et qui renvoie un booléen égal à `True` si le nombre de 6 est supérieur ou égal à 2, `False` sinon.

On pourra utiliser la fonction `randint(a, b)` du module `random` pour laquelle la documentation officielle est la suivante :

```
random.randint(a, b)  
    Renvoie un entier aléatoire N tel que  $a \leq N \leq b$ .
```

Exemples :

```
>>> lancer1 = lancer(5)  
>>> lancer1  
[5, 6, 6, 2, 2]  
>>> paire_6(lancer1)  
True  
>>> lancer2 = lancer(5)  
>>> lancer2  
[6, 5, 1, 6, 6]  
>>> paire_6(lancer2)  
True  
>>> lancer3 = lancer(3)  
>>> lancer3  
[2, 2, 6]  
>>> paire_6(lancer3)  
False  
>>> lancer4 = lancer(0)  
>>> lancer4  
[]  
>>> paire_6(lancer4)  
False
```

## EXERCICE 2 (10 points)

On considère une image en 256 niveaux de gris que l'on représente par une grille de nombres, c'est-à-dire une liste composée de sous-listes toutes de longueurs identiques.

La largeur de l'image est donc la longueur d'une sous-liste et la hauteur de l'image est le nombre de sous-listes.

Chaque sous-liste représente une ligne de l'image et chaque élément des sous-listes est un entier compris entre 0 et 255, représentant l'intensité lumineuse du pixel.

Le négatif d'une image est l'image constituée des pixels  $x_n$  tels que  $x_n + x_i = 255$  où  $x_i$  est le pixel correspondant de l'image initiale.

Étant donné une valeur seuil, la binarisation d'une image est l'image constituée des pixels  $x_b$  valant 0 si  $x_i < \text{seuil}$  et 255 sinon, où  $x_i$  est le pixel correspondant de l'image initiale.

Compléter le programme ci-dessous :

```
def nombre_lignes(image):
    '''renvoie le nombre de lignes de l'image'''
    return ...

def nombre_colonnes(image):
    '''renvoie la largeur de l'image'''
    return ...

def negatif(image):
    '''renvoie le negatif de l'image sous la forme
    d'une liste de listes'''
    # on cree une image de 0 aux memes dimensions
    # que le parametre image
    nouvelle_image = [[0 for k in range(nombre_colonnes(image))]
                       for i in range(nombre_lignes(image))]

    for i in range(nombre_lignes(image)):
        for j in range(...):
            nouvelle_image[i][j] = ...
    return nouvelle_image

def binaire(image, seuil):
    '''renvoie une image binarisee de l'image sous la forme
    d'une liste de listes contenant des 0 si la valeur
    du pixel est strictement inferieure au seuil et 255 sinon'''
    nouvelle_image = [[0] * nombre_colonnes(image)
                       for i in range(nombre_lignes(image))]

    for i in range(nombre_lignes(image)):
        for j in range(...):
            if image[i][j] < ... :
                nouvelle_image[i][j] = ...
            else:
                nouvelle_image[i][j] = ...
    return nouvelle_image
```



## Exemples :

```
>>> img=[[20, 34, 254, 145, 6], [23, 124, 237, 225, 69],  
         [197, 174, 207, 25, 87], [255, 0, 24, 197, 189]]  
>>> nombre_lignes(img)  
4  
>>> nombre_colonnes(img)  
5  
>>> negatif(img)  
[[235, 221, 1, 110, 249], [232, 131, 18, 30, 186],  
 [58, 81, 48, 230, 168], [0, 255, 231, 58, 66]]  
>>> binaire(img,120)  
[[0, 0, 255, 255, 0],[0, 255, 255, 255, 0],  
 [255, 255, 255, 0, 0],[255, 0, 0, 255, 255]]
```