

ⵜⴰⴳⴷⴰⵢⵜ ⵏ ⵎⵓⵔ
ⵜⴰⴷⵓⵏⵉⵜ ⵏ ⵉⵎⵓⵔ ⵏ ⵉⵎⵓⵔ ⵏ
ⵉⵎⵓⵔ ⵏ ⵉⵎⵓⵔ ⵏ ⵉⵎⵓⵔ



المملكة المغربية
وزارة التعليم العالي
والبحث العلمي والتكنولوجيا

Royaume du Maroc
Ministère de l'Enseignement Supérieur,
de la Recherche Scientifique et de l'Innovation



INSEA

CONCOURS NATIONAL COMMUN

d'Admission dans les Établissements de Formation d'Ingénieurs et
Établissements Assimilés

Session 2026

Épreuve de : **Informatique**

Filières : **MP**

Date : **15-05-2026** Durée **2 Heures** de **15H00 à 17H00**

Si, au cours de l'épreuve, un candidat repère ce qui lui semble être un point de clarification ou une erreur d'énoncé, il le signale sur sa copie et poursuit sa composition en expliquant les raisons de son initiative.

L'usage de tout document et de tout matériel électronique ou informatique (y compris la calculatrice)
est interdit

Cette épreuve comporte **12 pages** au format **A4** en plus de cette page de garde

Les candidats sont informés que la précision des raisonnements algorithmiques ainsi que le soin apporté à la rédaction et à la présentation des copies seront des éléments pris en compte dans la notation. Il convient en particulier de rappeler avec précision les références des questions abordées. Si, au cours de l'épreuve, un candidat repère ce qui peut lui sembler être une erreur d'énoncé, il le signale sur sa copie et poursuit sa composition en expliquant les raisons des initiatives qu'il est amené à prendre.

Remarques générales :

- ✓ Cette épreuve est composée d'un exercice et de trois parties tous indépendants ;
- ✓ Toutes les instructions et les fonctions demandées seront écrites en Python ;
- ✓ Les questions non traitées peuvent être admises pour aborder les questions ultérieures ;
- ✓ Toute fonction peut être décomposée, si nécessaire, en plusieurs fonctions.

Important : Le candidat doit impérativement commencer par traiter toutes les questions de l'exercice ci-dessous, et écrire les réponses dans les premières pages du cahier de réponses.

Exercice : (4 points)

Plus grand diviseur commun

Q.1- Écrire la fonction **existe** (L, x) qui reçoit en paramètres une liste de nombres L , et un nombre x . La fonction retourne **True** si x existe dans L , sinon, la fonction retourne **False**.

Q.2- Déterminer la complexité de la fonction **existe** (L, x), avec justification.

Q.3- Écrire la fonction **communs** (A, B) qui reçoit en paramètres deux listes de nombres A et B , et qui retourne une nouvelle liste contenant les nombres communs à A et B .

Exemple : $A = [35, 7, 1, 5, 14, 70, 2, 10]$ et $B = [28, 1, 4, 112, 7, 8, 14, 2, 16, 56]$

communs (A, B) retourne la liste $[7, 5, 1, 14, 2, 8]$

Q.4- Écrire la fonction **grand** (L) qui reçoit en paramètre une liste de nombres L , et qui retourne le plus grand élément dans L . (Ne pas utiliser la fonction prédéfinie `max()`).

Q.5- Écrire la fonction **diviseurs** (x) qui reçoit en paramètre un nombre entier strictement positif x , et qui retourne la liste des diviseurs de x .

Exemple : **diviseurs** (70) retourne la liste $[1, 2, 5, 7, 10, 14, 35, 70]$

Q.6- Un nombre entier strictement positif est premier s'il possède exactement deux diviseurs : 1 et lui-même.

Écrire la fonction **premier** (x) qui reçoit en paramètre un nombre entier strictement positif x , et qui retourne **True** si x est premier, sinon, la fonction retourne **False**.

Exemples :

- **premier** (23) retourne **True**
- **premier** (15) retourne **False**

Q.7- Écrire la fonction **pgcd** (x, y) qui reçoit en paramètres deux nombres entiers strictement positifs x et y . La fonction retourne le plus grand diviseur commun à x et y .

Exemple : **pgcd** (70, 112) retourne le nombre 14

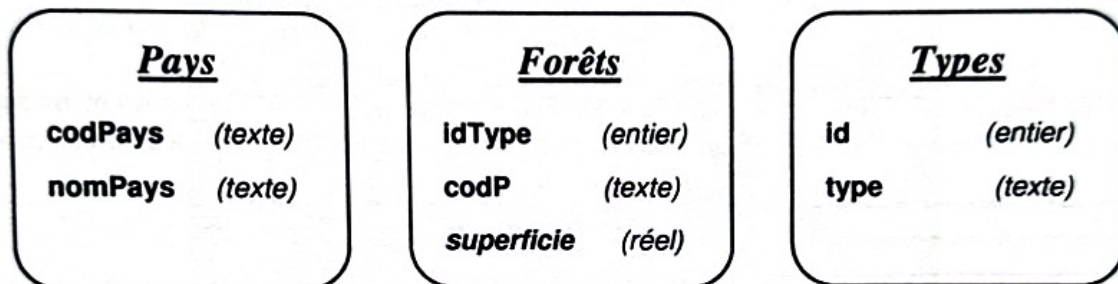
Partie I : Base de données et langage SQL**Les forêts du monde**

Les forêts couvrent environ **31 %** des terres émergées de la planète, représentant un élément fondamental de l'écosystème mondial. Elles jouent un rôle crucial dans la régulation du climat, la préservation de la biodiversité et le maintien des cycles biogéochimiques. Mais elles sont menacées par la déforestation et le changement climatique.

Certains types de forêts s'étendent sur plusieurs pays. Par exemple, on trouve des forêts tropicales :

- En Amérique du sud (forêt amazonienne) : *Brésil, Pérou, Colombie, Venezuela, Équateur, Bolivie, Guyana, Suriname et Guyane française.*
- En Afrique (forêt bassin du Congo) : *Cameroun, République centrafricaine, République démocratique du Congo, Gabon, et la Guinée équatoriale.*
- ...

Pour explorer les différents types de forêts à travers le monde, leur répartition géographique, on utilise une base de données composée des tables suivantes :



La table '**Pays**' est composée des champs suivants :

- Le champ **codPays** contient un code qui identifie de façon unique chaque pays ;
- Le champ **nomPays** contient le nom de pays.

Exemples :

<i>codPays</i>	<i>nomPays</i>
BRA	Brésil
CNG	Congo
RUS	Russie
CAN	Canada
IDN	Indonésie
...	...

La table '**Types**' est composée des champs suivants :

- Le champ **Id** contient un entier qui identifie de façon unique chaque type de forêt ;
- Le champ **type** contient le type de la forêt.

Exemples :

<i>id</i>	<i>type</i>
17	Tropicale
38	Boréale
29	Tempérée
14	Méditerranéenne
25	Subtropicale
...	...

La table 'Forêts' est composée des champs suivants :

- Le champ **idTyp** contient l'identifiant du type de la forêt ;
- Le champ **codP** contient le code du pays sur lequel s'étend la forêt ;
- Le champ **superficie** contient la superficie de la forêt dans chaque pays, en Km².

Exemples :

<i>idTyp</i>	<i>codP</i>	<i>superficie</i>
17	BRA	5 500 000
17	CNG	3 700 000
38	RUS	7 700 000
38	CAN	3 000 000
17	IDN	920 000
...	...	...

Q.8- Écrire, en langage SQL, une requête SQL qui ajoute dans la table 'Types', le type 'Mangrove', son identifiant est 45.

Q.9- Écrire, en langage SQL, une requête SQL qui donne pour résultat, les codes des pays et les superficies des trois premières forêts, ayant les plus grandes superficies. (Les superficies des forêts sont toutes différentes)

Q.10- Écrire, en langage SQL, une requête SQL qui donne pour résultat, le compte des forêts, la plus petite superficie, la plus grande superficie et la moyenne des superficies des forêts.

Q.11- Écrire, en algèbre relationnelle, une requête qui donne pour résultat, les noms des pays sur lesquels s'étendent les forêts de type 'Boréale'.

Q.12- Écrire la requête précédente (Q.11) en langage SQL. Le résultat doit être trié dans l'ordre alphabétique des noms des pays.

Q.13- Écrire, en langage SQL, une requête SQL qui donne pour résultat, les types des forêts et la superficie totale de chaque type de forêt, telles que la superficie totale est supérieure à 5 millions de Km². Le résultat doit être trié dans l'ordre décroissant de la superficie totale.

Q.14- Écrire, en langage SQL, une requête SQL qui donne pour résultat, les codes des pays sur lesquels s'étendent les forêts de type 'Tropicale', et le pourcentage de la superficie de la forêt dans chaque pays par rapport à la superficie totale de toutes les forêts de type 'Tropicale'. Le résultat doit être trié dans l'ordre croissant des pourcentages des superficies.

Partie II :

Modélisation d'un feu de forêt **Automate cellulaire**

Dans cette partie, on suppose que la méthode **randint** est importée :

```
from random import randint
```

La méthode **randint** reçoit en paramètres deux entiers x et y tels que $x \leq y$, et elle retourne un entier aléatoire compris entre x et y . Par exemple, **randint(0, 100)** retourne **61**

Le réchauffement climatique est à l'origine d'un grand nombre de phénomènes climatiques catastrophes. Les sécheresses sont de plus en plus nombreuses et provoquent entre autres des feux de forêt aux conséquences désastreuses.



*Pour modéliser un feu dans une forêt d'arbres, on utilise un **automate cellulaire**.*

*Un **automate cellulaire** est un modèle mathématique et informatique qui simule l'évolution d'un ensemble de cellules disposées sur une grille régulière, et contenant chacune un « état » choisi parmi un ensemble fini. L'état d'une cellule évolue par pas de temps discrets, à l'aide d'une règle locale, laquelle prend en compte l'état d'un nombre fini de cellules appelé « voisinage » de la cellule. Classiquement, à chaque pas de temps, la même règle est appliquée simultanément à toutes les cellules de la grille, produisant une nouvelle « génération » de cellules, ou configuration, à partir de la configuration précédente.*

Les automates cellulaires sont à la fois des systèmes dynamiques discrets et des modèles de calcul, ce qui les rend fascinants pour les chercheurs en mathématiques, en informatique, et en sciences

Représentation d'une forêt d'arbres :

La forêt est représentée par une matrice **R** de n lignes et p colonnes. Les cellules **R_{i,j}** sont de deux types :

- Sol nu : représenté par le nombre **0**
- Arbre : représenté par le nombre **1**

Exemple :

La matrice suivante représente une forêt d'arbres de 10 lignes et 15 colonnes :

```

[[0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1],
[1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1],
[1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0],
[1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1],
[1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1]]

```

Si N est le nombre d'arbres dans la forêt, on notera $d=100*N/(n*p)$ la densité d'arbres dans la forêt.

La densité de l'arbre de l'exemple ci-dessus est : 85%

Création d'une forêt aléatoire d'arbres :

Q.15 - Écrire la fonction `matrice_zeros(n, p)` qui reçoit en paramètres deux entiers strictement positifs n et p , et qui retourne une matrice de 0, de n lignes et p colonnes.

Exemple : `matrice_zeros(3, 5)` retourne la matrice `[[0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]]`

Q.16 - Écrire la fonction `indices(R)` qui reçoit en paramètre une matrice R , et qui retourne une liste de tuples : chaque tuple contient les indices (ligne et colonne) d'une cellule dans la matrice R .

Exemple : `R = [[0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]]`

`indices(R)` retourne la liste : `[(0, 0), (0, 1), (0, 2), (0, 3), (0, 4), (1, 0), (1, 1), (1, 2), (1, 3), (1, 4), (2, 0), (2, 1), (2, 2), (2, 3), (2, 4)]`

Q.17 - Écrire la fonction `foret_aleatoire(n, p, d)` qui reçoit en paramètres deux entiers strictement positifs n et p , et un réel d qui représente la densité des arbres. La fonction retourne une matrice R de n lignes et p colonnes, qui représente une forêt d'arbres de densité d . Les arbres sont répartis aléatoirement dans la forêt.

Exemple : `foret_aleatoire(15, 25, 80)` retourne la matrice suivante.

```

[[1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1],
[1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1],
[0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1],
[1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1],
[1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0],
[1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1],
[1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0],
[1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1],
[1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1]]

```

Dans la suite de cette partie, on utilise cette matrice dans les exemples des fonctions.

Dans la matrice R , une cellule a au plus quatre cellules voisines : au nord, à l'est, au sud ou à l'ouest.

Q.18 - Écrire la fonction **voisines** (R, i, j) qui reçoit en paramètres une matrice R qui représente une forêt d'arbres, et deux entiers i et j qui représentent les indices d'une cellule dans R . La fonction retourne la liste des tuples qui représentent les indices des cellules voisines à la cellule d'indices i et j dans R .

Exemples :

- **voisines** ($R, 0, 0$) retourne la liste : [(0, 1), (1, 0)]
- **voisines** ($R, 2, 0$) retourne la liste : [(2, 1), (3, 0), (1, 0)]
- **voisines** ($R, 1, 1$) retourne la liste : [(0, 1), (2, 1), (1, 0), (1, 2)]

Propagation du feu dans une forêt d'arbres :

Pour déclencher le feu dans la forêt, on met en feu un arbre aléatoire dans la forêt : on choisit un arbre aléatoire, et on change son état en un arbre en feu.

Q.19 - Écrire la fonction **miseEnFeu** (R) qui reçoit en paramètre une matrice R qui représente une forêt d'arbres. La fonction met en feu un arbre aléatoire dans la forêt R .

Lors de la propagation du feu, les cellules de la forêt sont de quatre types :

- Sol nu : représenté par le nombre 0
- Arbre vivant : représenté par le nombre 1
- Arbre en feu : représenté par le nombre 2
- Arbre en cendre : représenté par le nombre 3

L'évolution du feu dans la forêt sera le reflet de l'évolution de la matrice dont les cellules changent d'état, selon les deux règles suivantes :

- Si la cellule est un arbre en feu, alors à l'état suivant, elle devient cellule d'un arbre en cendre.
- Si la cellule est un arbre en feu, et qu'une cellule de ses voisines dans les quatre directions est un arbre vivant, alors cette cellule voisine devient cellule d'un arbre en feu à l'état suivant.

Q.20 - Écrire la fonction **evolution** (R) qui reçoit en paramètre une matrice R qui représente une forêt d'arbres. La fonction met à jour la matrice selon l'état des arbres de la forêt R :

- Les arbres en feu deviendront des arbres en cendres
- Les arbres vivants voisins à des arbres en feu deviendront des arbres en feu.

Attention, si une cellule est un arbre en feu, et qu'une cellule voisine est un arbre vivant, il ne faut pas mettre à jour tout de suite cette cellule, en la mettant en feu, sinon on risque de modifier le voisinage de certaines cellules non encore examinées.

Q.21 - Écrire la fonction **propagation** (R) qui reçoit en paramètre une matrice R qui représente une forêt d'arbres. La fonction effectue le traitement suivant :

- Déclenche le feu dans un arbre aléatoire de la forêt ;
- Répéter l'évolution du feu dans la forêt, jusqu'à ce qu'il n'y ait plus d'arbre en feu.

Exemple :

```

[[3, 3, 3, 3, 3, 3, 3, 0, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3],
 [3, 3, 0, 3, 3, 3, 0, 3, 0, 3, 3, 3, 3, 3, 0, 3, 3, 3, 3, 0, 3, 3],
 [3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 0, 3, 3, 0, 3, 3, 3, 0, 3, 3, 3],
 [3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 0, 3, 3, 3, 0, 3, 3, 3, 3],
 [3, 3, 3, 0, 3, 3, 3, 0, 3, 0, 3, 3, 0, 3, 3, 0, 0, 3, 3, 3, 3],
 [0, 3, 3, 3, 3, 3, 3, 3, 0, 0, 3, 3, 3, 3, 3, 3, 3, 0, 3, 3, 3],
 [3, 3, 0, 0, 0, 3, 3, 3, 0, 3, 0, 3, 0, 3, 3, 0, 3, 3, 0, 0, 0],
 [3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 0, 3, 3, 3, 3, 3, 3, 3],
 [0, 3, 3, 0, 3, 3, 3, 3, 3, 3, 0, 0, 3, 3, 3, 3, 3, 3, 0, 3, 3],
 [3, 3, 0, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 0, 0, 3, 0],
 [3, 3, 0, 3, 3, 0, 3, 3, 3, 3, 0, 3, 0, 3, 3, 3, 0, 0, 0, 3, 0],
 [3, 3, 0, 3, 3, 3, 3, 0, 3, 3, 3, 3, 0, 3, 3, 3, 3, 3, 3, 3, 0],
 [3, 3, 3, 3, 3, 3, 3, 0, 0, 3, 0, 3, 3, 0, 3, 3, 3, 3, 3, 0, 0],
 [3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 0, 3, 3, 3, 1],
 [3, 3, 0, 3, 3, 0, 3, 0, 3, 3, 3, 0, 3, 3, 0, 3, 3, 0, 3, 0, 1]]

```

Percolation dans une forêt d'arbres :

La propagation du feu va dépendre essentiellement de la densité des arbres dans la forêt. Les arbres sont supposés répartis aléatoirement. Il est concevable que les dégâts sont plus étendus si la densité est élevée. En fait, en jouant avec la densité, on se rend compte qu'elle admet une valeur critique, dite **seuil de percolation** :

- En deçà, l'incendie reste très limité ;
- Au-delà, et presque immédiatement, la propagation est beaucoup plus importante.

Q.22 - Écrire la fonction `taux_brule (R)` qui reçoit en paramètre une matrice `R` qui représente une forêt d'arbres. La fonction déclenche le feu et effectue la propagation du feu dans la forêt `R`. La fonction retourne le pourcentage des arbres en cendres dans la forêt `R`.

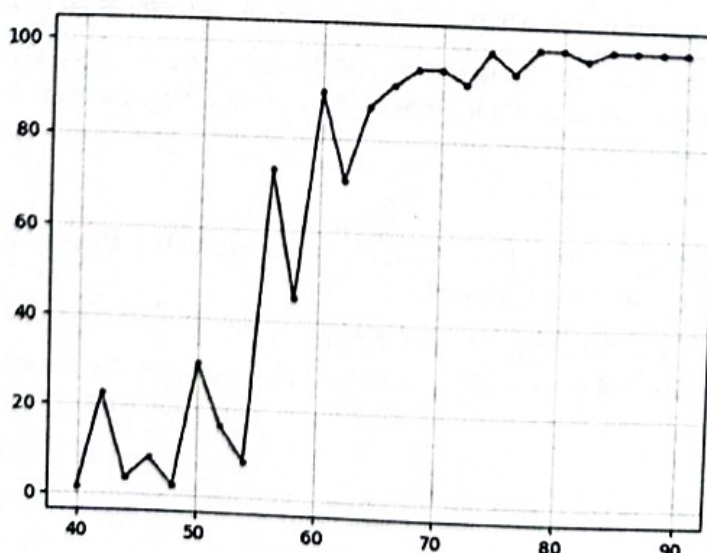
Exemple : `taux_brule (R)` retourne le pourcentage 98.67%

Q.23 – On suppose que le module suivant est importé :

```
import matplotlib.pyplot as plt
```

Écrire la fonction `percolation (n, p)` qui reçoit en paramètres deux entiers strictement positifs `n` et `p`. La fonction trace la courbe qui représente le taux d'arbres brûlés en fonction de la densité `d` des arbres dans la forêt représentée par une matrice de `n` lignes et `p` colonnes, pour `d=40, 42, 44, 46, ..., 88, 90`.

Exemple :



Partie III :

Détection d'objets dans une image
Histogramme des gradients orientés (HOG)

Dans cette partie, on suppose que le module *numpy* est importé : `import numpy as np`

L'**histogramme des gradients orientés (HOG)** est un descripteur de caractéristiques utilisé pour extraire les caractéristiques et les formes essentielles d'un objet particulier dans une image, telles que les bords et les textures. Les fonctionnalités extraites par *HOG* peuvent être utilisées pour alimenter le modèle d'apprentissage automatique.

HOG convertit une image en un vecteur caractéristique, dont la dimension spatiale est relativement plus petite que l'image d'origine. Par conséquent, *HOG* peut être utilisé pour détecter des images à petite échelle avec moins de puissance de calcul. Cependant, *HOG* est avantageux pour la détection des objets dans l'industrie de la vision par ordinateur en raison de sa vitesse de calcul et de sa précision. *HOG* est largement utilisé pour la détection de piétons.



Une image numérique en **niveaux de gris** est représentée par une matrice à deux dimensions. Chaque élément de la matrice est appelé **pixel**. La valeur de chaque pixel est un entier compris entre 0 et 255.

Le nombre 0 correspond au noir, le nombre 255 correspond au blanc, et chacun des autres nombres 1, 2, 3, ..., 253, 254 correspond à un niveau de gris entre le noir et le blanc. Ainsi l'image est codée sur 256 niveaux de gris.

Exemple :

Matrice qui représente l'image en niveaux de gris, du piéton ci-dessus.

[108	107	105	105	105	...]
[118	119	120	120	120	...]
[120	122	124	125	125	...]
[122	124	127	128	128	...]
...	...	...	...	...	...
[99	99	100	100	101	...]]

Dans la suite de cette partie, on utilise cette image dans les exemples des fonctions.

Somme des produits des éléments de deux matrices

Q.24 - Écrire la fonction **som_prods** (*A*, *B*) qui reçoit en paramètres deux matrices carrées *A* et *B* de même dimension (3 lignes, 3 colonnes), et qui retourne la somme des produits élément par élément de *A* et *B* :

$$\sum A_{i,j} * B_{i,j}$$

Matrice des voisins d'un pixel :

M est une matrice qui représente une image en niveaux de gris. On considère un pixel *M_{i,j}* qui n'appartient pas ni à la première ligne, ni à la dernière ligne, ni à la première colonne, ni à la dernière colonne de *M*.

On appelle **matrice des voisins** du pixel *M_{i,j}*, la sous matrice carrée d'ordre 3, composée des 8 voisins du pixel *M_{i,j}*. (Le pixel *M_{i,j}* se trouve au centre de la sous matrice)

Exemple : La **matrice des voisins** du pixel *M_{1,1}* (=119) est la sous matrice suivante.

```
[ [108  107  105]
  [118  119  120]
  [120  122  124] ]
```

Q.25 - Écrire la fonction **voisins** (*M*, *i*, *j*) qui retourne la matrice des voisins du pixel *M_{i,j}*.

Pour construire le descripteur **HOG** d'une image, on utilise l'algorithme suivant :

- Calculer les gradients de l'image dans chaque pixel
- Calculer les orientations des gradients
- Diviser l'image en secteurs
- Construire un histogramme des orientations des gradients dans chaque secteur,
- Construire un histogramme des orientations des gradients dans chaque bloc (= 4 secteur),
- Construire le vecteur de fonctionnalité HOG de l'image par la concaténation de tous les histogrammes des blocs.

Pour calculer les gradients de l'image, on calcule d'abord les gradients horizontaux et les gradients verticaux. Ceci est réalisé en utilisant le filtre de **Sobel**. Ce filtre permet de déterminer les contours d'objets dans une image.

Calcul des gradients horizontaux :

Les gradients horizontaux permettent de détecter les contours horizontaux d'objets dans une image.

Q.26 - Écrire la fonction **gradH** (*M*) qui reçoit en paramètre la matrice *M* qui représente une image en niveau de gris. La fonction retourne la matrice des gradients horizontaux *X* de même dimension que *M*. Les éléments de la matrice *X* sont calculés selon la formule suivante :

- Si le pixel *M_{i,j}* se trouve sur la première ou la dernière ligne de *M*, ou sur la première ou la dernière colonne de *M*, alors *X_{i,j}* ← *M_{i,j}* ;
- Sinon, *X_{i,j}* ← La valeur de la somme des produits élément par élément de la matrice des voisins du pixel *M_{i,j}* et de la matrice suivante :

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Calcul des gradients verticaux

Les gradients verticaux permettent de détecter les contours verticaux d'objets dans une image.

Q.27 - Écrire la fonction **gradV (M)** qui reçoit en paramètre la matrice **M** qui représente une image en niveau de gris. La fonction retourne la matrice des gradients verticaux **Y** de même dimension que **M**. Les éléments de la matrice **Y** sont calculés selon la formule suivante :

- Si le pixel $M_{i,j}$ se trouve sur la première ou la dernière ligne de **M**, ou sur la première ou la dernière colonne de **M**, alors $Y_{i,j} \leftarrow M_{i,j}$;
- Sinon, $Y_{i,j} \leftarrow$ La valeur de la somme des produits élément par élément de la matrice des voisins du pixel $M_{i,j}$ et de la matrice suivante :

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Calcul des gradients

Les gradients permettent de détecter les contours d'objets dans l'image.

Q.28 - Écrire la fonction **gradients (M)** qui reçoit en paramètre la matrice **M** qui représente une image en niveau de gris. La fonction retourne la matrice des gradients **G** de même dimension que **M**. Les éléments de **G** sont calculés en fonction des gradients horizontaux et des gradients verticaux, selon la formule suivante :

$$G_{i,j} \leftarrow \lfloor \sqrt{X_{i,j}^2 + Y_{i,j}^2} \rfloor$$

Exemple :

La matrice des gradients représente l'image suivante.



Calcul des orientations des gradients

Pour calculer les orientations des gradients, on utilise la fonction prédéfinie du module numpy : **atan2**

La fonction **atan2(x, y)** est une variante de la fonction **arc tangente** qui calcule l'angle entre l'axe des abscisses positifs et le point de coordonnées (x, y) , en radians avec une plage de $-\pi$ et π . Elle prend en compte les signes de ses deux paramètres pour déterminer correctement le quadrant de l'angle.

Q.29 - Écrire la fonction **orientations** (*M*) qui reçoit en paramètre la matrice *M* qui représente une image en niveau de gris. La fonction retourne la matrice des orientations des gradients *D* de même dimension que *M*. Les éléments de *D* sont calculés en fonction des gradients horizontaux et des gradients verticaux, selon la formule suivante :

$$D_{i,j} \leftarrow \text{atan2}(X_{i,j}, Y_{i,j})$$

NB : Les éléments de *D* doivent être calculés en degrés, avec une plage de 0 et 360.

Subdivision de l'image

On suppose que le nombre de ligne *n* et le nombre de colonnes *p* dans l'image sont des multiples de 8, et que $n \geq 16$ et $p \geq 16$.

L'image est subdivisée en secteurs de même dimension : chaque secteur est composé de 8 lignes et 8 colonnes

Exemple :

108	107	105	105	105	106	108	110	107	108	109	109	108	107	106	105	104	104	105	106	107	107	106	106
118	119	120	120	120	121	123	124	123	123	123	122	121	121	121	121	121	121	122	123	123	123	122	122
120	122	124	125	125	126	128	129	129	129	128	127	126	127	128	125	125	126	126	126	126	124	124	124
122	124	127	128	128	128	129	131	131	131	130	129	129	129	130	130	130	131	131	131	129	127	127	127
127	128	129	129	130	130	131	131	131	131	132	131	131	130	130	131	130	130	130	130	128	127	126	126
129	129	129	130	130	130	130	130	130	130	131	131	131	130	130	130	133	133	133	133	131	130	129	129
130	131	132	132	131	131	131	131	131	132	132	132	132	131	131	131	131	131	131	131	131	130	129	129
129	131	133	133	131	130	130	131	132	132	132	132	132	132	132	132	132	132	132	132	132	131	130	130
126	126	126	126	126	127	129	130	131	131	132	133	132	131	130	130	131	131	132	133	133	132	131	131
128	128	127	126	126	126	127	128	133	133	132	132	132	132	131	131	132	133	133	133	134	133	133	133
129	129	127	126	125	126	127	128	131	131	130	130	132	133	133	133	132	132	132	132	134	133	133	133
128	127	126	126	127	127	129	130	130	129	128	127	130	132	132	132	132	132	133	133	134	133	132	132
129	128	128	128	129	130	132	133	133	133	132	132	132	131	130	130	131	33	131	133	134	133	133	133
129	128	128	128	129	131	132	132	134	134	134	133	133	131	130	129	126	128	130	131	132	132	133	132
129	129	129	129	130	131	131	131	132	133	134	133	132	131	131	131	126	127	128	129	129	130	131	131
129	129	129	129	131	131	131	131	132	133	133	132	130	128	128	129	130	130	129	129	129	129	131	131
126	126	126	127	131	132	133	133	134	134	134	132	128	126	127	128	132	131	131	131	131	132	132	132
127	128	128	129	131	132	133	134	133	134	134	132	130	129	130	130	131	131	131	131	132	132	132	131
129	130	131	131	133	133	134	134	134	135	134	133	132	132	132	131	131	132	133	134	134	133	132	132
130	132	133	133	134	133	134	135	135	135	135	134	134	133	132	131	131	131	132	134	135	134	133	132
130	132	133	133	134	133	134	134	135	135	134	134	134	133	131	130	130	130	131	132	132	131	129	128
129	131	132	133	134	133	133	133	134	134	134	133	132	131	129	128	128	128	127	127	126	124	123	123
129	130	132	133	135	135	134	134	134	135	135	133	131	129	128	126	126	125	125	125	125	124	123	123
130	131	132	134	137	137	136	135	136	137	137	136	133	130	128	127	126	125	125	125	126	127	128	128

Calcul de l'histogramme dans un secteur

L'intervalle des angles $[0, 360]$ est subdivisé en 8 sous-intervalles :

$$K = [0, 45, 90, 135, 180, 225, 270, 315]$$

Chaque pixel de l'image participe au vote, à l'aide du gradient et de l'angle du gradient du pixel.

L'histogramme d'un secteur est un vecteur *H* de taille 8. Les éléments de *H* sont calculés en utilisant la formule suivante :

$$\text{Pour chaque angle } k \text{ de la subdivision } K : H(k) \leftarrow \sum G_{i,j} \text{ tel que } D_{i,j} \in [k, k + 45[$$

Exemple :

L'histogramme du secteur C1 est : $[980, 2454, 7, 72, 43, 0, 0, 19]$

Q.30 - Écrire la fonction `hist_secteur (M, i, j)` qui reçoit en paramètres la matrice **M** qui représente une image en niveau de gris, et **i, j** sont respectivement les indices de la ligne et de la colonne du pixel en haut à gauche d'un secteur. La fonction retourne l'histogramme de ce secteur.

Exemple :

- ✓ `hist_secteur (M, 0, 0)` retourne l'histogramme `[980, 2454, 7, 72, 43, 0, 0, 19]`
- ✓ `hist_secteur (M, 0, 8)` retourne l'histogramme `[276, 1233, 6, 7, 7, 3, 3, 805]`

Les secteurs sont regroupés en **blocs** de 2x2 secteurs. *Un secteur peut participer à plusieurs blocs.*

108	107	105	105	105	106	108	110	107	108	109	109	108	107	106	105	104	104	105	106	107	107	106	106
118	119	120	120	120	121	123	124	123	123	123	123	122	121	121	121	121	121	122	123	123	123	122	122
120	122	124	124	124	126	128	129	129	129	128	127	126	127	128	125	125	126	126	126	126	124	124	124
122	124	127	127	128	128	129	131	131	131	130	129	129	129	130	130	130	131	131	131	129	127	127	127
127	128	129	129	130	130	131	131	131	132	131	131	130	130	131	130	130	130	130	130	128	127	126	126
129	129	129	130	130	130	130	130	130	131	131	131	130	130	130	133	133	133	133	131	130	129	129	129
130	131	132	132	131	131	131	131	131	132	132	132	132	131	131	131	131	131	131	131	131	130	129	129
129	131	133	133	131	130	130	131	132	132	132	132	132	132	132	132	132	132	132	132	132	131	130	130
126	126	126	126	126	127	129	130	131	131	132	133	132	131	130	130	131	131	132	133	133	132	131	131
128	128	127	126	126	126	127	128	133	133	132	132	132	132	131	131	132	133	133	133	134	133	133	133
129	129	127	126	125	126	127	128	131	131	130	130	132	133	133	133	132	132	132	132	134	133	133	133
128	127	126	125	125	127	129	130	130	129	128	128	130	132	132	132	132	132	133	133	134	134	133	132
129	128	128	128	129	130	132	133	133	133	132	132	132	132	131	130	130	131	33	134	133	134	133	133
129	128	128	128	129	131	132	132	134	134	134	133	133	131	130	129	128	128	130	133	132	132	133	132
129	129	129	129	130	131	131	131	132	133	134	133	132	131	131	131	126	127	128	129	129	130	131	131
129	129	129	129	131	131	131	131	132	133	133	132	130	128	128	129	130	130	129	129	129	129	131	131
126	126	126	127	131	132	133	133	134	134	134	132	128	126	127	128	132	131	131	131	131	132	132	132
127	128	128	129	131	132	133	134	133	134	134	132	130	129	130	130	131	131	131	131	132	132	132	131
129	130	131	131	133	133	134	134	134	135	134	133	132	132	132	131	131	132	132	132	134	133	133	133
130	132	133	133	134	133	134	135	135	135	135	134	133	132	131	131	131	132	132	132	134	133	133	133
130	132	133	133	134	133	134	134	135	135	134	134	133	133	131	130	130	130	131	132	132	131	129	128
129	131	132	133	134	133	133	133	134	134	134	133	133	131	129	128	128	128	127	127	126	124	123	123
129	130	132	133	135	135	134	134	135	135	135	133	131	129	128	126	126	125	125	125	125	124	123	123
130	131	132	134	137	137	136	135	136	137	137	136	133	130	128	127	126	125	125	125	126	127	128	128

Exemples :

- Les secteurs (C1, C2, C4, C5) forment un bloc ;
- Les secteurs (C2, C3, C5, C6) forment un bloc ;
- Les secteurs (C4, C5, C7, C8) forment un bloc ;
- Les secteurs (C5, C6, C8, C9) forment un bloc ;
- ...

Calcul de l'histogramme d'un bloc :

L'histogramme d'un bloc est la concaténation des histogrammes des secteurs qui composent ce bloc.

Exemple :

L'histogramme du bloc (C1, C2, C4, C5) est la concaténation des histogrammes des secteurs C1, C2, C4 et C5 :

`[980, 2454, 7, 72, 43, 0, 0, 19, 276, 1233, 6, 7, 7, 3, 3, 805, 169, 1543, 67, 77, 80, 28, 17, 40, 6, 5, 35, 65, 59, 122, 96, 32, 93]`

Le vecteur HOG de l'image est la concaténation des histogrammes de tous les blocs dans l'image.

Q.31 - Écrire la fonction `HOG (M)` qui reçoit en paramètre la matrice **M** qui représente une image en niveau de gris. La fonction retourne le vecteur HOG de l'image **M**.

***** FIN *****