ACADÉMIE
DE REIMS
Liberté
Égalité
Fraternité

NUMERIQUE
et
SCIENCES
INFORMATIQUES

Épreuve de l'enseignement de spécialité

Sujet d'entraînement

Partie Pratique

Classe Terminale de la voie générale

Le candidat doit traiter les 2 exercices

Ce sujet comporte 4 pages

Exercice 1 (4 points)

Vous travaillez comme développeur dans une entreprise publicitaire. Vous avez à déterminer les jours de la semaine où le temps d'attente moyen au téléphone des "clients importants" dépasse 10 secondes.

Le dictionnaire `dis_qu_allo` suivant associe à chaque jour ouvré de la semaine la liste des temps d'attente au téléphone connus par les "clients importants" ce jour-là.

```
dis_qu_allo = {'lundi':[],'mardi':[12,6,8],'mercredi':[9,11,4,6],
               'jeudi':[13],'vendredi':[2,7,9,15],'samedi':[14,13,9,6,9]}
```

Écrire une fonction `liste_depassements` qui prend en argument un dictionnaire `dico` d'associations jour / liste de temps d'attente et qui renvoie la liste des jours où le temps moyenne moyen des "clients importants" dépasse 10 secondes.

Exemples :

```
>>> liste_depassements(dis_qu_allo)
    ['jeudi','samedi']
>>> liste_depassements({'lundi':[6],'mardi':[5,19,1],'mercredi':[10],
'jeudi':[],'vendredi':[4,4,4,4],'samedi':[6,11,12]})
    []
```

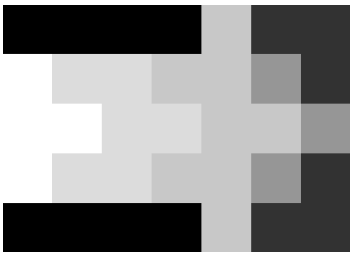
Exercice 2 (4 points)

L'entreprise publicitaire qui vous emploie veut faciliter la modification des teintes d'affiches publicitaires en nuances de gris.

Les affiches sont stockées comme des images où chaque pixel est codé par un nombre entier compris entre 0 et 255, nombre qui indique le niveau de gris du pixel, sachant que 0 correspond au noir et 255 au blanc.

Chaque image est représentée par un tableau à deux dimensions `img`. Chaque élément `img[i][j]` correspond au nombre entier compris entre 0 et 255 codant la nuance du gris d'un pixel ; ce pixel `img[i][j]` est celui décalé de `i` pixels vers le bas et de `j` pixels vers la gauche depuis le pixel du coin supérieur gauche.

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

Par exemple : L'image 

est représentée par le tableau suivant :

img =

0	0	0	0	200	50	50
255	220	220	200	200	150	50
255	255	220	220	200	200	150
255	220	220	200	200	150	50
0	0	0	0	200	50	50

dont les composantes sont :

img =

0	0	0	0	200	50	50
255	220	220	200	200	150	50
255	255	220	220	200	200	150
255	220	220	200	200	150	50
0	0	0	0	200	50	50

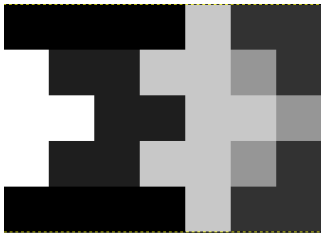
Pour modifier facilement l'image en nuances de gris, l'entreprise souhaite, à partir d'un pixel dans une image, donner la valeur `intensite` à tous les pixels de la composante à laquelle appartient ce pixel.

Pour cela, vous devez finaliser une procédure `propager` qui prend en paramètre un tableau à deux dimensions `img`, deux entiers `i` et `j` correspondant aux coordonnées d'un pixel et un entier `intensite`, compris entre 0 et 255. Cette procédure met à la valeur `intensite` tous les pixels de la composante `img[i][j]`.

img =

0	0	0	0	200	50	50
255	30	30	200	200	150	50
255	255	30	30	200	200	150
255	30	**30**	200	200	150	50
0	0	0	0	200	50	50

Par exemple, `propager(img,i=3,j=2,intensite=30)` donne :

ce qui correspond à l'image suivante : 

Remarque : Si `img[i][j]` est déjà égal à `intensite`, il n'y a rien à faire.

Compléter le code suivant, ainsi que les commentaires sous forme `...`, commencé par un de vos collègues de l'entreprise puis ajouter des tests pour vérifier son bon fonctionnement :

```
def propager(img, i, j, intensite):
    if img[i][j] == ...:
        # ...
        return

    intensite_origine = ...

    img[i][j] = intensite

    if (i - 1) >= 0 and img[i - 1][j] == ...:
        # l'élément au-dessus fait partie de la composante
        propager(img, i - 1, j, intensite)

    if (...) < len(img) and img[i + 1][j] == intensite_origine:
        # l'élément au-dessous fait partie de la composante
        propager(img, ..., j, intensite)

    if ... and img[...][...] == intensite_origine:
        # l'élément à gauche fait partie de la composante
        ...

    if ...:
        # ...
        ...
```

Exemple :

```
>>> img = [[0,0,0,0,200,50,50],
           [255,220,220,200,200,150,50],
           [255,255,220,220,200,200,150],
           [255,220,220,200,200,150,50],
           [0,0,0,0,200,50,50]]
>>> propager(img,3,2,30)
>>> img
[[0,0,0,0,200,50,50],[255,30,30,200,200,150,50],[255,255,30,30,200,200,150],
[255,30,30,200,200,150,50],[0,0,0,0,200,50,50]]
```