Traitement des images
Opérations élémentaires

Objectif : Rendre l'élève capable :

- de lire et écrire des images avec **Python**
- de modifier les couleurs d'une images :
  - niveaux de gris
  - noir et blanc
  - inversion des couleurs, ...
- d'obtenir les images miroirs vertical ou horizontal
- de faire tourner l'image de 90° ou 180°
- réduire ou agrandir une image d'un facteur r entier

1 Lecture/écriture d'une image

1.1 Différentes méthodes d'ouverture du fichier image

- avec matplotlib.pyplot

```
import matplotlib.pyplot as plt
image = plt.imread('carnot.jpg')
print(len(image), len(image[0]), len(image[0][0]))
```

- avec pillow

```
import numpy as np
from PIL import Image as im
img = im.open('carnot.jpg')
image = np.array(img)
print(len(image), len(image[0]), len(image[0][0]))
```

1.2 Obtenir des informations sur une image

```
>>> image = plt.imread('carnot.bmp')
>>> print(image.shape, image.dtype)
(306, 408, 4) uint8
```

```
>>> img = im.open('carnot.png')
>>> print(img.format, img.size, img.mode)
PNG (408, 306) RGBA
```

Q - 1 : *Charger l'image* `carnot.jpg` *avec* `plt`.

1.3 Visualiser une image à partir d'un tableau exploitable

```
plt.imshow(image)
plt.show()
```

```
image_pil = im.fromarray(image)
image_pil.show()
```

Q - 2 : *Déterminer les caractéristiques de l'image* `carnot.jpg` *avec* `plt`.

1.4 Créer/modifier une image

La partie 1.3 montre qu'à partir d'une image stockée sous forme de liste de listes de listes telle que définie en partie 1.1 il est possible de la visualiser directement avec matplotlib.pyplot. Pour la lire avec pillow reste à la convertir en tableau numpy (si ce n'est pas déjà le cas) puis en image pillow.

On utilise donc les outils numpy pour créer des tableaux de listes à 3 éléments. Le plus rapide pour obtenir une image noire et une image blanche de `nl` lignes et `nc` colonnes est :

```
image_noire = np. zeros ((nl , nc,3) ,dtype =np. int8 )
image_blanche = np. ones ((nl , nc,3) ,dtype =np. int8 )*255
```

Q - 3 : *Créer un échiquier dont les cases sont des carrés larges de 50 pix.*

1.5 Sauvegarder une image

A partir d'une image `image` stockée sous forme de liste de listes de listes telle que définie en partie 1.1 il est possible de l'enregistrer sous différents formats :

```
plt.imsave('nomdelimage.jpg',  image)
```

```
image_pil = im.fromarray(image)
image_pil.save('nomdelimage.jpg')
```

Attention ! avec `PIL`, il faut adapter l'extension en fonction de mode de l'image : par exemple `jpg` pour RGB ; `png` pour RGBA.

Q - 4 : *Sauvegarde l'échiquier au format `jpg`.*

2 Opérations élémentaires

2.1 Opérations sur les couleurs

Q - 5 : *Écrire une fonction `extraction(T)` qui prend en entrée une image `T` sous forme de tableau et renvoie 3 listes contenant chacune les canaux de l'image.*

Q - 6 : *Écrire une fonction `niveau_gris_geom(T)` qui prend en entrée une image `T` sous forme de tableau et renvoie l'image en nuance de gris.*

Q - 7 : *Écrire une fonction `inverse(T)` qui prend en entrée une image `T` sous forme de tableau et renvoie l'image en couleurs inverses.*

2.2 Opérations de symétrie

Q - 8 : *Écrire une fonction `miroir_v(T)` qui prend en entrée une image `T` sous forme de tableau et renvoie l'image en miroir vertical.*

Q - 9 : *Écrire une fonction `miroir_h(T)` qui prend en entrée une image `T` sous forme de tableau et renvoie l'image en miroir horizontal.*

Q - 10 : *Écrire une fonction* `rotation_180(T)` *qui prend en entrée une image* `T` *sous forme de tableau et renvoie l'image tournée de 180°.*

Q - 11 : *Écrire une fonction* `rotation_90(T)` *qui prend en entrée une image* `T` *sous forme de tableau et renvoie l'image tournée de 90° dans le sens trigonométrique.*

2.3 Changement d'échelle

Q - 12 : *Écrire une fonction* `reduction(T, r)` *qui prend en entrée une image* `T` *sous forme de tableau et un entier* `r` *et qui renvoie l'image réduite d'un facteur* `r`.

Q - 13 : *Écrire une fonction* `agrandissement(T, r)` *qui prend en entrée une image* `T` *sous forme de tableau et un entier* `r` *et qui renvoie l'image agrandie d'un facteur* `r`.