TP 11 : travailler avec des images en Python

Préambule :

1. Ouvrir une session sur l'ordinateur en entrant Identifiant / Mot de passe ;
2. Lancer Spyder en cherchant Spyder dans la barre de recherche ;
3. Effacer le texte dans la fenêtre de gauche et écrire :

```
#NOM
#PRENOM
#TP11
```

Objectifs de ce TP :

- *Ouvrir des images matricielles ;*
- *Accéder aux valeurs des pixels des images ;*
- *Modifier des images matricielles ;*
- *Créer de nouvelles images matricielles.*

Toutes les réponses aux questions doivent écrites dans le programme qui sera à rendre dans le dossier Restitution de devoirs de votre classe avant 18h.

L'objectif de ce TP est de découvrir comment travailler avec des images en Python. Nous pourrons ouvrir une image, regarder son contenu mais aussi la modifier (changer les couleurs, la taille, l'orientation, le format). Nous serons aussi capable de créer des images. Pour tout cela, nous utiliserons la bibliothèque `Pillow` dans laquelle un ensemble de méthodes permet de travailler sur des images.

Importer la bibliothèque `Pillow` :

```
from PIL import Image
```

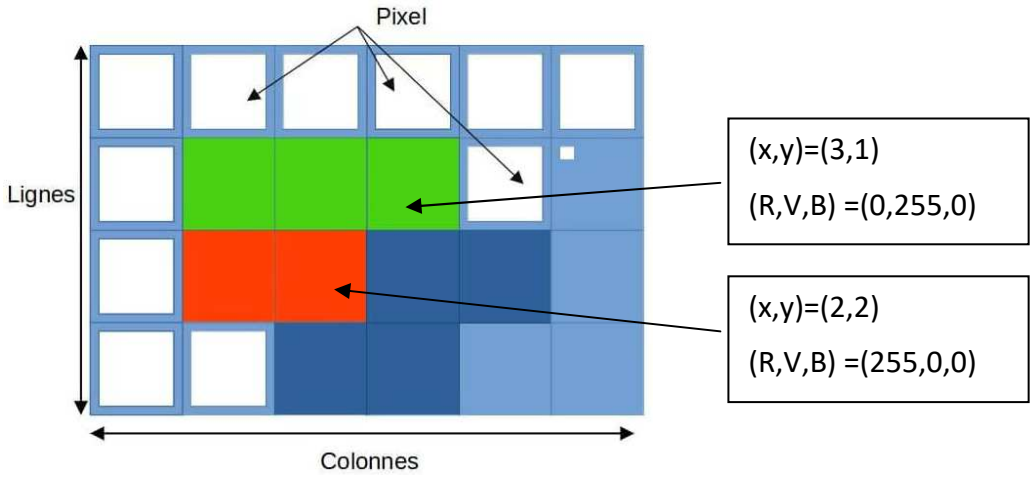
Partie 1 : qu'est-ce qu'une image au matricielle ?

Les images que vous connaissez bien de type .jpeg, .bmp, .gif sont des images matricielles.
Elles sont composées de pixels de différentes couleurs qui forment une matrice, un tableau, constituée de plusieurs lignes et plusieurs colonnes.
Chaque pixel est repéré par sa position dans la matrice et est défini par sa couleur.
La position est définie par les coordonnées (x,y) du pixel **en partant du coin supérieur gauche de l'image**:

- x : numéro de la colonne ou abscisse du pixel en partant de 0 ;
- y : numéro de la ligne ou ordonnée du pixel en partant de 0.

La définition de la couleur est différente selon le type d'image :

- image noir et blanc : 0 pour noir, 1 pour blanc ;
- image en niveau de gris : un entier entre 0 et 255 ;
- image en couleur 8 bits : un entier entre 0 et 255 ;
- image en couleurs vraies 24 bits : un triplet (R,V,B) de (0,0,0) à (255,255,255).



Dans tout le TP, nous travaillerons avec 2 images de perroquets disponibles dans le dossier Espace d'échanges de la TSI1 :

- parrots.gif : couleurs 8 bits ;
- parrots24.jpg : couleurs vraies 24 bits.

Copier les 2 images dans le même dossier que votre programme.

Question 1 : Ouvrir l'image et voir ses propriétés

Commande	Résultats

Ouvrir l'image 8 bits et l'afficher :
 	
`perroquets8=Image.open("parrots.gif")`
`perroquets8.show()`	

Récupérer la taille de l'image en pixels :
 	
`print(perroquets8.size)`
`L8=perroquets8.size[0]`
`H8=perroquets8.size[1]`	

Calculer le poids de l'image en octets : `poids8=L8*H8`	

Question 2 : Accéder aux valeurs des pixels

Commandes	Résultats
Charger la matrice de pixels : `matrice8=perroquets8.load()`	

Afficher les valeurs des pixels :
 	
`print(matrice8[0,0])`
`print(matrice8[100,200])`	

Question 3 : Ouvrir l'image 24 bits et voir ses propriétés

Commande	Résultats

Ouvrir l'image 24 bits et l'afficher :
 	
`perroquets24=Image.open("parrots.gif")`
`perroquets24.show()`	

Récupérer la taille de l'image en pixels :
 	
`print(perroquets24.size)`
`L24=perroquets24.size[0]`
`H24=perroquets24.size[1]`	

Calculer le poids de l'image en octets : `poids24=L24*H24*24/8`	

Question 4 :

Charger la matrice de pixels de l'image 24 bits et afficher la valeur du pixel (100,200).

Résultats : _____

Conclure sur les différences entre l'image 8 bits et l'image 24 bits :

_____

Question 5 : Créer une nouvelle image de la couleur d'un pixel (R,V,B)

Tester les lignes suivantes :

```
(R,G,B)=eval(input("quel triplet RGB ?:"))
couleurRGB=Image.new("RGB",(50,50),(R,G,B))
couleurRGB.show()
```

Que remarquez-vous ?

Question 6 : Parcourir la matrice de pixels

Pour parcourir la matrice de pixel il suffit de faire 2 boucles `for` imbriquées afin de parcourir lignes et colonnes, par exemple :

Commandes	Résultats

Parcourir la matrice de pixels :
 	
`for x in range(L24):`
`  for y in range(H24):`
`  print(matrice24[x,y])`	

Question 7 : Remplacer le rouge par du bleu (enfin à peu près) !

En parcourant la matrice du pixel, remplacer le rouge par du bleu. Pour cela, nous considérerons comme « rouge » tous les pixels vérifiant cette condition :

```
matrice24[x,y][0]>100 and matrice24[x,y][1]<50 and matrice24[x,y][2]<50
```

A la fin du remplacement, il faut sauvegarder la nouvelle image sans écraser l'ancienne :

```
perroquets24.save("perroquets24_bleu.jpg")
```

Observer le résultat.

Partie 2 : comment modifier une image matricielle ?

Dans cette partie, nous allons modifier une image en la transformant :

- **symétrie d'axe vertical ;**
- **rotation de 90°;**
- **effet négatif ;**
- **conversion en niveau de gris ;**
- **effet photomaton.**

Le principe sera toujours le même : ne jamais travailler sur l'image originale mais partir d'une image vide et la remplir au fur et à mesure.

Pour cela, la structure du code sera toujours de la forme suivante :

```
# créer une image noire appelée new du bon type et de la bonne taille
new=Image.new("RGB",(L24,H24),(0,0,0))
```

type dimensions couleur

```
# sauvegarder la nouvelle image
new.save("new.jpg")
# charger la matrice
matrice_new=new.load()

# faire la transformation en parcourant la matrice
… à vous de jouer

# sauvegarder l'image finale modifiée
new.save("new_finale.jpg")
```

Question 8 : Symétriser l'image

Créer le code permettant de symétriser l'image. Il suffit d'affecter au pixel de coordonnée `(x,y)` de la nouvelle image, le pixel de coordonnées `(L24-1-x,y)` de l'ancienne image.

Question 9 : Tourner l'image de 90°

Créer le code permettant de tourner l'image de 90°. Il suffit d'affecter au pixel de coordonnée `(x,y)` de la nouvelle image, le pixel de coordonnées `(L24-1-y,x)` de l'ancienne image.

Attention aux dimensions de la nouvelle image !

Question 10 : Convertir l'image en niveau de gris

Créer le code permettant de convertir l'image en niveau de gris.

Pour cela, nous respecterons la norme qui indique que le niveau de gris G du pixel est calculé à partir des valeurs R,V et B :

$$G=R*0.2125+V*0.7154+B*0.0721$$

Question 11 : Inversion des couleurs

Créer le code permettant d'inverser les couleurs de l'image pour créer un effet de négatif.

Question 12 : Effet photomaton

Créer le code permettant de créer un effet photomaton à partir de la photo originale.

Attention aux dimensions de la nouvelle image !