Python et les images

Pour traiter les images numériques en Python nous allons utiliser le module PIL (Python Imaging Library). Ce module n'est pas préinstallé. Il nécessite une installation supplémentaire.

Voici quelques instructions utiles :
Ouverture d'un fichier image : *nom_image* = Image.open("*nom_fichier*")
La bibliothèque support plusieurs formats de fichiers parmi lesquels PNG, JPEG, GIF, TIFF, BMP

Sauvegarde d'une image dans un fichier : *nom_image*.save(*"nom_fichier"*)

Obtenir le type d'image (couleur ou niveau de gris) : *nom_image*.mode
Retourne "L" pour une image en niveau de gris et "RGB" pour une image en couleur
On trouve aussi le format "RGBA" : RGBA accepte **3 valeurs entières** comprises entre 0 et 255 ou 3 **valeurs entières en pourcentage** et une valeur entre 0 (transparent) et 1 opaque.
Si vous êtes en pourcentage 100% représente 255. Vous ne pouvez pas mélanger les unités.

Obtenir la taille de l'image : *nom_image*.size . On obtient un tuple sous la forme (L,H)

Créer une image de largeur L et de hauteur H (en pixels) :
nom_image = Image.new("RGB",(L,H)) ou *nom_image* = Image.new("L",(L,H))

Lecture d'un pixel de l'image : *nom_pixel* = *nom_image*.getpixel((x,y))
On obtient un tableau à 3 colonnes qui correspondent respectivement aux composantes rouge, verte et bleue du pixel pour une image en couleur ou juste un entier pour une image en noir et blanc.

Ecriture d'un pixel : *nom_image*.putpixel((x,y),(r,v,b)) où r, v et b correspondent respectivement aux composantes rouge, verte et bleue du pixel pour une image en couleur
nom_image.putpixel((x,y),g) où g correspond à la valeur du niveau de gris du pixel pour une image en niveau de gris.

Afficher une image : nom_*image*.show()

Exercices avec le module PIL de Python

Exercice 1 : Ouvre le fichier *decompositionPIL.py*

1) Essaie de comprendre à quoi correspondent les images image1, image2 et image3.
2) Exécute le programme en utilisant le fichier arbreBMP.bmp, lenaBMP.bmp, tortue.jpg et une autre image de ton choix.
3) Ouvre les fichiers obtenus avec Gimp ou ImageJ.
4) Essaye d'utiliser le fichier singemBMP.bmp. Explique ce qui se passe.

Exercice 2 :

Ecris un programme pour créer une image en niveau de gris à partir d'une image en couleur.
On donne la formule suivante :
gris = int(round(0.299 * red + 0.587 * green + 0.114 * blue)) où gris est la composante en niveau de gris et red, green et blue sont les trois composantes rouge, vert, bleu.

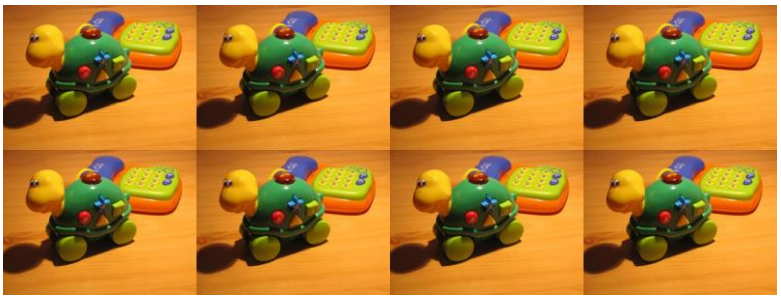
Exercice 3 :

1) Ecris un programme qui permet de retourner une image horizontalement.
2) Ecris un programme qui permet de retourner une image verticalement.
3) Ecris un programme qui permet de faire faire une rotation de 180° à une image.
4) Ecris un programme qui permet de faire faire une rotation de 90° à une image.

			
Image originale	Retournement horizontal	Retournement vertical	Rotation à 180°

Exercice 4 :

Ecris un programme Python qui permet créer une nouvelle image en dupliquant une image donnée.



Exercice 5 :

1) Ecrire un programme Python pour les fichiers en niveau de gris qui permet d'afficher le nombre de pixels pour chaque niveau de gris entre 0 et 255, puis détermine le niveau de gris minimal, puis le niveau de gris maximal.
   A l'aide de ces valeurs on peut tracer ce qu'on appelle l'histogramme de l'image.
   Ouvre avec Gimp un fichier en niveau de gris puis affiche l'histogramme correspondant.
2) Si les niveaux de gris ne prennent pas des valeurs entre 0 et 255 inclus on peut réaliser un étirement des contrastes. Ainsi chaque pixel d'intensité x est remplacé par un pixel d'intensité $\mathbf{f(x)=M(x-g_{min})//(g_{max}-g_{min})}$ avec M une valeur constante fixée (par ex M=255), g_{min} (resp g_{max}) le niveau de gris minimal (resp maximal)
   Ecris un programme permettant de réaliser un étirement de contraste pour une image en niveau de gris.