Partie 1 : Image : manipulation des couleurs

Pour cette partie, il faut utiliser l'image `lena.png`, disponible sur le site www.touron.tech.

Exercice 1 : **Convertir une image : couleur vers nuance de gris**

Pour transformer une image en couleur en image en niveaux de gris, nous allons nous contenter de calculer la moyenne pixel par pixel des trois composantes du triplet (R,V,B).

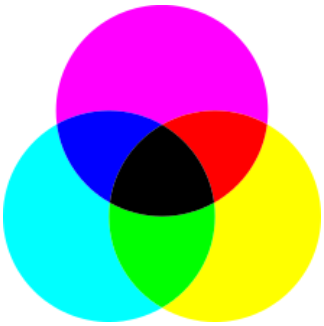
Q1. Ecrire une fonction `niveau_gris` ayant pour argument une chaîne de caractères étant le nom du fichier de l'image initiale. Cette fonction doit créer et afficher une nouvelle image en niveau de gris.

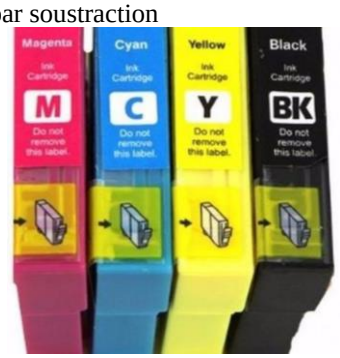
Exercice 2 : **Négatif d'une image**

Pour créer une image, on peut procéder par addition de lumières colorées sur un fond noir (principe du vidéo projecteur numérique) ou par soustraction de couleur à partir une lumière blanche (principe des cartouches d'encre).

Couleurs par addition

Couleurs par soustraction





(voir la version numérique du sujet en couleur)

Une image convertie d'un format dans l'autre est transformée par une « image en négatif ».

Une image négative est une image dont les couleurs ont été inversée par rapport à l'originale ; par exemple le rouge devient cyan, le vert devient magenta, le bleu devient jaune et inversement. Les régions sombres deviennent claires, le noir devient blanc.
Les couleurs primaires étant codées sur un octet (256 nuances), la couleur opposée à x se calcule par : $255 - x$

Q1. Ecrire une fonction `inverse` ayant pour argument une chaîne de caractères étant le nom du fichier de l'image initiale. Cette fonction doit créer et afficher une nouvelle image négative de la première.

Partie 2 : Image : Détection de bord

Pour cet exercice, il faut utiliser l'image `shapes.bmp`, disponible sur le site www.touron.tech.

Le bord d'une figure particulière, insérée dans une photo, est reconnaissable par une forte variation de l'intensité lumineuse. Cette variation est quantifiable en calculant la dérivée de l'intensité lumineuse $I(x, y)$, où x et y sont les coordonnées verticale et horizontale du pixel dans l'image.

Une approximation de la dérivée dans la direction x s'écrit :

$$D_x(x,y) = \frac{I(x+1,y) - I(x-1,y)}{2\Delta x}$$

($\Delta x = 1$ pixel)
En termes de traitement informatique, cela correspond à la convolution de la matrice équivalente à l'image par la matrice :

$$\frac{1}{2}[-1,0,1]$$

Une approximation plus fine est obtenue par l'opérateur de Sobel selon x :

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

Et selon y :

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

Q1. Ecrire un programme qui réalise et qui affiche les dérivées selon x et y de l'image originale par convolution par l'opérateur de Sobel (les dimensions de l'image produite auront 2 pixels de moins que celles de l'originale, horizontalement et verticalement).

Q2. On souhaite combiner les dérivées D_x et D_y en calculant pour chaque pixel la norme du vecteur formé par ces deux valeurs : $N(x,y) = \sqrt{D_x(x,y)^2 + D_y(x,y)^2}$. Ecrire un programme qui réalise et affiche l'image formée par la norme des deux dérivées.

Partie 3 : Tri par comptage

Le document `LesMiserables1.txt` contient le premier des cinq tomes d'un ouvrage classique de la littérature française.

On désire y compter la fréquence (au sens statistique) de chacune des 26 lettres de l'alphabet.

Q1. Ecrire un programme qui donnent en sortie un tableau des fréquences. On peut par exemple parcourir 26 fois le document.

Q2. Ecrire un programme plus performant où le calcul est fait en un seul parcours, en utilisant l'index de la lettre dans l'alphabet. On pourra utiliser avec astuce la fonction ord proposée par python.