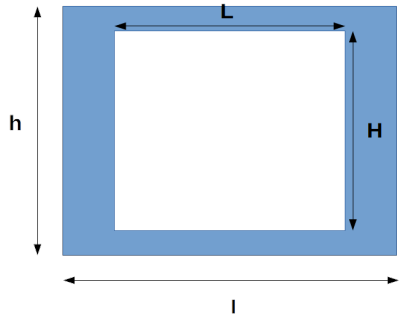
# TD informatique du chapitre 6 : Traitement d'images 2

**Exercice 1 : Stéganographie**

1. Ecrire une fonction decodage prenant en argument une image im renvoyant l'image cachée dans im selon le principe vu en cours.
   Tester cette fonction sur les images mystere1.png et mystere2.png.
2. On souhaite maintenant faire l'opération inverse, c'est-à-dire cacher une image dans une autre image.
   (a) Afin de cacher une image dans une autre, il faut que les 2 images aient la même taille. Ecrire une fonction rogner prenant en argument une image im et deux entiers H et L qui renvoie une image de taille H × L obtenue en rognant et centrant im.




   (b) Utiliser la fonction rogner pour transformer les images intitulées lena.png et trigo.png en deux images de même taille.
   (c) Ecrire une fonction cacher prenant en argument deux images de même taille im1 et im2 renvoyant l'image obtenue en cachant im2 dans im1.
   (d) Cacher les formules de trigonométrie dans l'image de Lena.

**Exercice 2 : Modification d'une image par convolution**

1. Ecrire une fonction convolution prenant en argument une image im et un noyau de convolution N et renvoyant l'image obtenue par convolution.
   On testera cette fonction avec les exemples de noyaux du cours.
2. On souhaite améliorer le résultat obtenu pour la détection des contours. On va utiliser le filtre de Sobel. Le principe est d'introduire deux noyaux de convolution :

$$N_1 = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \text{ et } N_2 = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$

   Le noyau $N1$ permet de détecter les contours verticaux et $N2$ les contours horizontaux.
   Si on considère une valeur $p$ d'une composante d'un pixel. On note $s_1$ la valeur obtenue par $N_1$ et $s_2$ la valeur obtenue par $N_2$. On combine alors cette deux valeurs en utilisant la moyenne quadratique : $\sqrt{s_1^2 + s_2^2}$. Enfin, on affiche le négatif de l'image obtenue.
   Ecrire une fonction Sobel prenant en argument une image im et renvoyant l'image obtenue par filtrage de Sobel.

**Exercice 3 : Niveaux de gris**

Dans une image en niveaux de gris, les trois composantes R,V,B de chaque pixel ont la même valeur. L'oeil étant plus sensible à certaines couleurs qu'à d'autres, on ne considère pas la moyenne des composantes R,V,B mais la moyenne pondérée suivante :

$$gris = 0.299 \times R + 0.587 \times V + 0.114 \times B.$$

Ecrire une fonction niveau_gris prenant en argument une image im et renvoyant l'image en niveaux de gris
Afficher niveau_gris(lena).

**Exercice 4 : Insertion d'un fond**

On considère une image fond et une image im de taille inférieure à celle de fond et ayant un fond blanc.
On considèrera que le fond blanc n'est pas pur et on introduira une précision p pour détecter le blanc. Ainsi, un pixel blanc ne sera pas de la forme $(R,V,B)$ avec $R = V = B = 255$ mais avec $R,V,B \geq 255 - p$.
Ecrire une fonction Fond prenant en argument deux images im et fond et un entier p et renvoyant l'image ayant pour fond fond sur laquelle est insérée, au centre, l'image im.
On testera cette fonction en insérant une panthère ( panthere.png) dans le lycée Saint-Louis ( SaintLouis.png).

**Exercice 5 : Pixellisation**

La pixellisation est souvent utilisée pour flouter une image. La zone pixellisée fait apparaître des blocs carrés de couleur uniforme. Pour arriver à ce résultat, on considère des blocs carrés de taille $n \times n$, c'est-à-dire de $n^2$ pixels et on remplace chaque composante R,V,B par la moyenne des $n^2$ composantes R,V,B.

Afin de traiter tous les pixels, il faudrait que la taille de l'image soit un multiple de $n$ ce qui n'est pas toujours le cas. Ici, on ne traitera pas les bords mais uniquement les blocs carrés contenus dans l'image.

Ecrire une fonction `pixellisation` prenant en argument une image `im` et un entier `n` et renvoyant l'image pixellisée avec des blocs de taille $n$.
Afficher `pixellisation(lena,8)`.