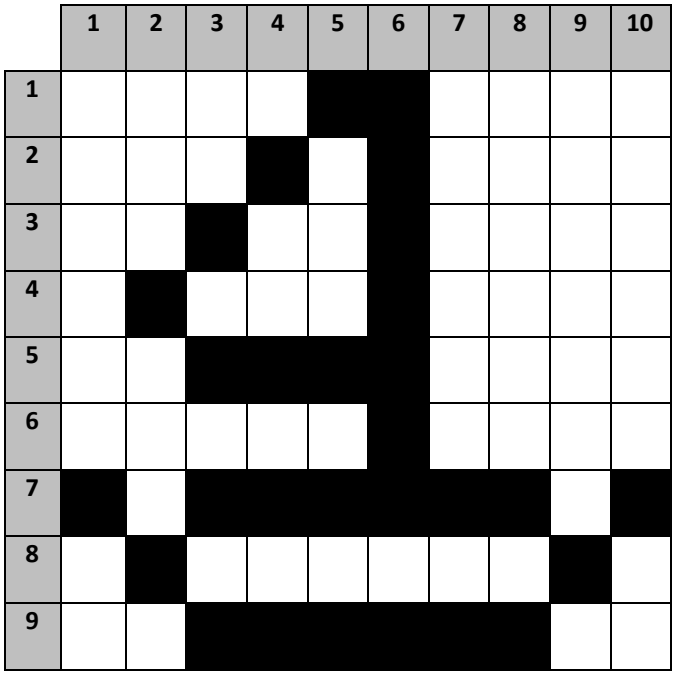
Jouer avec les images

1. Généralités sur les images numériques

Une **image numérique** est constituée d'un **ensemble de points appelés pixels** (*abréviation de Picture Element*) pour former une image. Le pixel est le plus petit élément constituant une image numérique.



Définition d'une image
La définition d'une image est le nombre de pixels qui la composent.

Exemple
Cette image possède 10 colonnes et 9 lignes.
Sa définition est de $10 \times 9 = 90$ pixels.

Résolution d'une image
La résolution d'une image est le nombre de pixels par unité de longueur qui est le pouce (1 pouce = 2,54 cm)

Relation entre définition, taille et résolution :

$$Résolution = \frac{définition}{dimension}$$

La taille d'une image peut se définir par :
Sa définition en pixels (nombre de colonnes × nombre de lignes)
Ses dimensions en pouces (exemple : 21". *Au fait, ça fait combien en cm ?* ……)
Sa résolution en dpi ou ppp

Exercice 1
Quelle serait la définition d'une feuille scannée d'une largeur de 8,5 pouces sur une hauteur de 11 pouces en 300 dpi ?

Poids d'une image en noir et blanc ou en couleur
le poids d'une image est le nombre de bits nécessaires à l'affichage de l'image.

- **En noir et blanc**
  Chaque pixel de l'image est codé à l'aide d'un bit : 0 (noir) 1 (blanc)
- **En nuances de gris** (exemple 256 nuances)
  Chaque pixel de l'image est codé à l'aide d'un octet (au total 8 bits) : **un entier compris entre 0 et 255**
- **En couleur** (exemple RVB : 3 couches de 256 couleurs chacune)
  Chaque pixel est codé par 3 octets : 1 pour le rouge, 1 pour les vert, 1 pour le bleu (au total 24 bits)
  Chaque pixel sera représenté par **un tuple** (liste non modifiable) **de 3 entiers**, chacun compris entre 0 et 255

Exercice 2
Combien peut-on obtenir de couleurs avec 3 couches de 256 couleurs chacune ?
Quel est le poids d'une image couleur RVB de définition 6 000×4 000 ?

2. Jouer avec des images en nuances de gris

Enregistrer une photo au format JPG ou PNG en niveau de gris dans un dossier appelé : jouer_avec_images
Vous appellerez votre image « image_gris »
Attention : *Une photo en noir et blanc n'est pas forcément une image en niveau de gris ; chaque pixel peut avoir une composante R, V et B. Bien lire les informations de l'image.*
Taper le code ci-dessous et l'enregistrer dans le même dossier jouer-avec_images.

Dans un premier temps, on va chercher à tester ce code et à comprendre les différentes fonctions rencontrées.

Notes sur les différentes instructions rencontrées :

```
import os
from PIL import Image

def mystere(i):
    (l, h) = i.size
    for y in range(h):
        for x in range(l):
            c=i.getpixel((x,y))
            magie=255-c
            i.putpixel((x, y), magie)

i=Image.open("image_gris.extension")
i.show()

mystere(i)
i.show()
os.system ("pause")
```

Image et cryptographie
La cryptographie est la science qui consiste à crypter des informations afin qu'elles ne puissent être comprises que par ceux qui connaissent la clé pour les déchiffrer.

1er exemple : le codage de César
Évidemment à l'époque de César, on ne cherchait à coder que des messages écrits. Ce codage consistait uniquement à décaler les lettre d'un certain rang.
Pour un décalage de 3, le A était remplacé par un D, le B par un E, ...
Imaginons ce même type de codage sur une image en niveau de gris.
Pour un décalage de 10, un niveau de gris de 40 devient un niveau de gris de 50 ...
Si le nouveau niveau de gris dépasse 255, on prend alors le reste de la division euclidienne de ce nouveau niveau de gris par 225.
Exemple : Pour un niveau de gris de 250, qui passerait à 260, on prend 5 (reste de la division euclidienne de 260 par 255)
Instruction python : **260 % 255** (donne le reste de la division euclidienne de 260 par 255)

Exercice 3
Essayer de modifier le code initial afin de faire subir un codage de César à notre image en niveau de gris.
Choisir plusieurs exemples de décalage.

2ème exemple : le codage affine
Choisir deux nombres a et b, avec a et 256 premiers entre eux
(*C'est à dire que leur plus grand diviseur commun est* 1).
On fait subir à chaque pixel x une transformation affine : $ax + b$
Évidemment, ce résultat risque de dépasser 255.
On remplace alors chaque pixel x par le reste de la division euclidienne de $ax + b$ par 256.

Exercice 4
Essayer de modifier le code initial afin de faire subir un codage affine à notre image en niveau de gris.
Choisir plusieurs exemples pour a et b :

1. $a = 3$ et $b = 45$
2. $a = 3$ et $b = 100$
3. $a = 77$ et $b = 15$

Défi : mélanger des images
Enregistrer deux images en niveau de gris de même taille.
Imaginer un mélange de pixels entre les deux images.

Exercice 5
On définit un premier mélange de la façon suivante :
On compare chaque niveau de gris de chaque pixel de chaque image.
Si le niveau de gris du pixel est supérieur dans la première image, alors on remplace le niveau de gris du pixel de la deuxième image par celui du pixel correspondant de la première image.

À vous d'imaginer d'autres mélanges ...

3. Jouer avec des images en couleur

Enregistrer une photo au format JPG ou PNG en couleur dans un dossier appelé : jouer_avec_images
Vous appellerez votre image « image_couleur »

Taper le code ci-dessous et l'enregistrer dans le même dossier jouer-avec_images.

Dans un premier temps, on va chercher à tester ce code et à comprendre les différences avec le code précédent.

Notes sur les différences observées :

```
import os
from PIL import Image

def mystere(i):
    (l, h) = i.size
    for y in range(h):
        for x in range(l):
            c=i.getpixel((x,y))
            i.putpixel((x, y), (255-c[0],255-c[1],255-c[2]))

i=Image.open("image_couleur.extension")
i.show()

mystere(i)
i.show()
os.system ("pause")
```

On peut imaginer reprendre les différents exercices faits ave l'image en niveaux de gris.