## Préparer l’activité

Pour cette activité, je récupère les images et les exemples dans le dossier de la classe et je les enregistre dans mon dossier personnel.

## Lire les couleurs d’un pixel

Je vais utiliser l’image **pomme.jpg**.

**Exercice 1** : Quelles sont les dimensions (nombre de pixels en largeur et en hauteur) de l’image **pomme.jpg** ?

................................................................................................................................

**Exercice 2** : Je lance le logiciel **EduPython** puis j’ouvre le fichier **lecturePixel.py**. Je lance son exécution en cliquant sur la flèche verte .

J’écris le résultat obtenu ci-dessous :

................................................................................................................................

**Exercice 3** : Je regarde le code du programme pour rechercher les **coordonnées**   
(x, y) du pixel dont le programme a lu **l’intensité** des couleurs rouge, verte et bleue ?

x : ............ et y : .........

## Ecrire les couleurs d’un pixel

Je saisis le code ci-dessous et j’enregistre ce fichier dans mon dossier en prenant comme nom de fichier "**ecriturePixel.py**". Puis j’exécute le programme en cliquant sur la flèche verte.

from PIL import Image

img = Image.open("pomme.jpg")

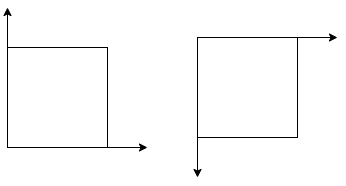
img.putpixel((5,5),(255,0,0))

img.show()

img.save("pommeAvecPoint.jpg")

La première ligne permet d’utiliser la **bibliothèque PILLOW** qui facilite la gestion des images.

**Exercice 4** : Je **cherche** dans quelle partie de l’image le pixel a été dessiné. Je **complète** le schéma pour placer le point et les coordonnées x et y des axes.



Je peux **dessiner d’autres points** en **modifiant** le programme.

## Changer les couleurs des pixels

J’ouvre le fichier **mystere.py**. qui utilise la **même** image **pomme.jpg** que les programmes précédents. Je lance son exécution en cliquant sur la flèche verte.

**Exercice 5** :

J’analyse le code du programme et le résultat obtenu pour écrire ce qu’il fait :

................................................................................................................................

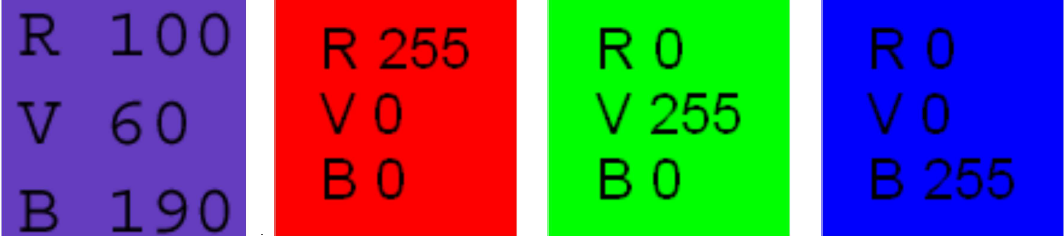
................................................................................................................................

## Changer les couleurs de l’image en niveaux de gris

Pour modifier les couleurs rouge, verte et bleue d’un pixel en niveaux de gris, on calcule la **moyenne** de ces trois valeurs. Le résultat obtenu doit être un **entier** et pour cela il faut utiliser la **division entière** (le résultat est un entier) en utilisant l'opérateur // à la place de l'opérateur /.

Cette valeur est appelée **luminance** et il faut alors remplacer chaque composante rouge, verte et bleue par cette même valeur.

**Exercice 6** : Je calcule la moyenne non pondérée des couleurs des quatre pixels pour déterminez la luminance de ces pixels :



1 – L = ……………….. 2 – L = …………………. 3 – L = ………………… 4 – L = ………………..

J’obtiens le **même résultat** pour les 3 dernières couleurs en niveaux de gris. Il n’est donc **pas possible** de les **distinguer**. Pour éviter cela, je calcule une **moyenne pondérée** qui tient compte de la **sensibilité** de l’œil humain aux couleurs primaires et du **support** utilisé pour afficher l'image (ici l’écran de l’ordinateur).

Voici la formule couramment utilisée : L=0,21R + 0,71V + 0,08 B.

**Exercice 7** : Je **recalcule la moyenne pondérée** des couleurs des quatre pixels pour déterminer la luminance de ces pixels :

1 – L = ……………….. 2 – L = …………………. 3 – L = ………………… 4 – L = ………………..

**Exercice 7 :** Je **saisis** le code ci-dessous et je le **complète** pour **calculer la moyenne non pondérée** des couleurs **rvb** de tous les pixels de l’image. J’enregistre ce fichier dans mon dossier sous le de fichier "**pommeGrise.py**".

from PIL import Image

img = Image.open("pomme.jpg")

largeur\_image=240

hauteur\_image=240

for y in range(hauteur\_image):

for x in range(largeur\_image):

rouge,vert,bleu=img.getpixel((x,y))

nouveau\_rouge = ( ………………….. ) // 3

nouveau\_vert = ( …………………… ) // ……

nouveau\_bleu = ( …………………... ) // ……

img.putpixel((x,y),(nouveau\_rouge,nouveau\_vert,nouveau\_bleu))

img.show()

La bibliothèque PILLOW propose un **mode L** pour calculer la **luminance** de tous les pixels d’une image afin de **convertir** une image en nuances de gris.

Je saisis le code ci-dessous et j’enregistre ce fichier dans mon dossier en prenant comme nom de fichier " **pommeGriseL.py** ". Puis j’exécute le programme en cliquant sur la flèche verte.

from PIL import Image

img = Image.open("pomme.jpg").convert("L")

img.show()

img.save("pommegriseL.jpg")

**Exercice 8 :** Je **modifie** mon programme "**pommeGrise.py**" pour utiliser une moyenne pondérée de calcul de la luminance.

**Exercice 9 :** Je **modifie** mon programme "**pommeGrise.py**" pour ne mettre en nuance de gris seulement la moitié de l’image : en vertical, en horizontal ou en diagonale.

**Exercice 10 :** Je recherche une autre image et je **modifie** mon programme "**pommeGrise.py**" pour utiliser cette image et la transformer en négatif uniquement pour les pixels dont la composante rouge est plus grande que les composantes vertes et bleue .

## Mettre l’image en négatif

Le négatif d’une couleur est obtenu en changeant chaque couleur par son complément à 255.

Voici un exemple : nouveau\_rouge = 255 - rouge

**Exercice 11 :** Je **modifie** mon programme "**pommeGrise.py**" pour transformer l’image en négatif.

**Exercice 12 :** Je **modifie** mon programme "**pommeGrise.py**" pour utiliser une transformation des couleurs des pixels de mon invention.