Nom :

Prénom :

Traitement d'image

Lorsque l'on prend une photographie, l'appareil corrige lui-même l'image finale, mais le résultat n'est pas toujours satisfaisant. On peut alors utiliser des logiciels de traitement d'image afin d'améliorer le résultat. Il est possible de faire subir à l'image toutes sortes de transformations : changement de format, de définition, modification des couleurs, trucages variés.

Nous allons observé ces différents traitements grâce 2 activités.

Remarque :

Vous trouverez dans le public de la classe un fichier vous présentant les bases de Gimp, il pourrait vous être utile, pensez à y jeter un œil.

Activité 1 : Introduction au traitement d'image avec GIMP

Modifier les couleurs

1. Télécharger sur le bureau le dossier d'images que vous trouverez dans le public de la classe.
2. Ouvrir la photo de votre choix avec le logiciel Gimp (clic droit + ouvrir avec Gimp).
3. Dans le menu "couleurs", tester les réglages des niveaux (dans le menu "Niveaux").
4. Modifier l'image avec les premiers outils de ce menu jusqu'à la commande "courbe". On observe particulièrement les effets produits par les changements dans la balance et la température des couleurs et par ceux produits par "Luminosité - contraste "

définition :

Contraste = Opposition entre deux valeurs, chacune faisant ressortir l'autre. L'oeil humain est particulièrement sensible au contraste, il est meilleur comparateur qu'analyseur.

Faible contraste

Fort contraste

Utiliser des filtres

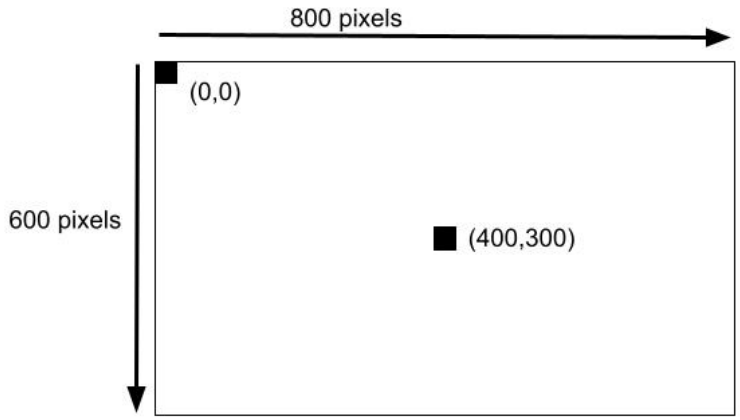
1. Ouvrir la photo de votre choix avec le logiciel Gimp (clic droit + ouvrir avec Gimp).
2. Ouvrir le menu "Filtres".
3. Explorer les différents types de filtres proposés par Gimp.
4. Dans le menu "Flou", tester le flou gaussien, puis le flou médian.

5. Dans le menu "distorsions", tester les filtres "Mosaïque" et "Kaléidoscope". en faisant varier la rotation du miroir et la taille des tuiles
6. Appliquer les différents filtres du menu "Artistiques"

Activité 2 : Traitement d'image avec Python

Nous allons utiliser le langage de programmation Python afin de directement travailler sur les pixels d'une image. Par travailler sur les pixels, j'entends déterminer la valeur du canal rouge, la valeur du canal et la valeur du canal bleu pour un pixel donné ou bien encore modifier carrément la couleur d'un pixel.

Avant de commencer à écrire un programme qui nous permettra de travailler sur les pixels d'une image, il est nécessaire de préciser que chaque pixel a des coordonnées x,y.

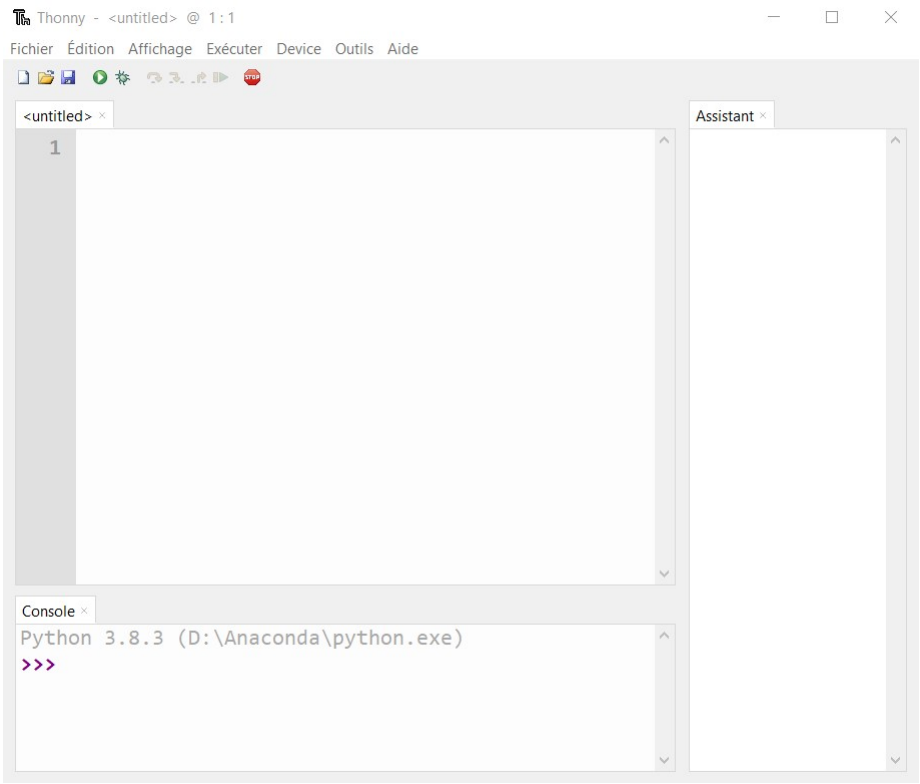


Comme vous pouvez le constater sur le schéma ci-dessus, le pixel de coordonnées (0,0) se trouve en haut à gauche de l'image. Si l'image fait 800 pixels de large et 600 pixels de haut, le pixel ayant pour coordonnées (400,300) sera au milieu de l'image.

Dans un premier temps nous allons utiliser une simple photo de pomme pour faire nos premiers essais, ensuite, vous pourrez travailler avec l'image de votre choix. L'image de la pomme se trouve dans le dossier d'images que vous avez télécharger dans l'activité 1. Cette image devra se trouver dans le même dossier que vos programmes Python.

-> Ouvrir Thonny

Vous devez arriver sur une fenêtre semblable à celle-ci :



La partie basse appelée `console` sert en général à visualiser le résultat d'un programme python ainsi que les éventuels problèmes rencontrés.

La partie droite appelée `Assistance` sert d'aide en cas de problème, espérons ne pas en avoir besoin !

La partie centrale est la zone ou on entre du code python.

Le bouton rond vert avec une petite flèche blanche à intérieur sert à tester le code python.

1. Copier et tester le code suivant dans thonny :

```
from PIL import Image
img = Image.open("pomme.jpg")
r,v,b=img.getpixel((100,250))
print("canal rouge : ",r,"canal vert : ",v,"canal bleu : ",b)
```

Ce programme vous donne le canal rouge, le canal vert et le canal bleu du pixel de coordonnées (100,250) de l'image "pomme.jpg"

Voici une analyse ligne par ligne du programme ci-dessus :

- "from PIL import Image" : pour travailler sur les images nous avons besoin d'une extension de Python (appelé bibliothèque). Cette bibliothèque se nomme PIL.
- "img = Image.open("pomme.jpg")" c'est grâce à cette ligne que nous précisons que nous allons travailler avec l'image "pomme.jpg". Pour travailler avec une autre image, il suffit de remplacer "pomme.jpg" par un autre nom (attention, votre fichier image devra se trouver dans le même dossier que le ficher de votre programme Python).
- "r,v,b=img.getpixel((100,250))" cette ligne récupère les valeurs du canal rouge (r), du canal vert (v) et du canal bleu (b) du pixel de coordonnées (100,250). Dans la suite du programme, r correspondra à la valeur du canal rouge, v correspondra à la valeur du canal vert et b correspondra à la valeur du canal bleu
- "print("canal rouge : ",r,"canal vert : ",v,"canal bleu : ",b)" permet d'imprimer le résultat

2. Modifiez le programme de la question 1 pour qu'il affiche les valeurs du canal rouge, du canal vert et du canal bleu du pixel de coordonnées (250,300), notez votre réponse.

   r = v = b =

Il est possible de modifier les canaux RVB d'un pixel :

3. Copier et tester le code suivant dans thonny :

```
from PIL import Image
img = Image.open("pomme.jpg")
img.putpixel((250,250),(255,0,0))
img.show()
```

   Regardez attentivement le centre de l'image, vous devriez voir un pixel rouge à la place d'un pixel vert.

Voici une analyse ligne par ligne du programme ci-dessus :

Les deux premières lignes sont identiques que celles à la question 1.

- "img.putpixel((250,250),(255,0,0))" permet de colorier le pixel de coordonnées (250,250) en rouge (255,0,0).
- "img.show()" permet d'afficher l'image modifiée

4. Modifiez le programme de la question 3 afin de colorier le pixel de coordonnées (100,250) en bleu.

   Appeler l'enseignant afin de valider cette question.

Modifiez un pixel c'est déjà bien, mais comment faire pour modifier plusieurs pixels ? La réponse est simple, nous allons utiliser des boucles "for". Le but ici n'est pas de détailler le fonctionnement des boucles "for" en Python, vous devez juste comprendre que grâce à ces boucles nous allons pouvoir balayer toute l'image et ne plus nous contenter de modifier les pixels un par un.

5. Copier et tester le programme suivant (ATTENTION : l'exécution de ce programme n'est pas très intéressante en soi, vous pouvez l'arrêter à tout moment en appuyant simultanément sur la touche Ctrl et sur la touche C):

```
from PIL import Image
img = Image.open("pomme.jpg")
largeur_image=500
hauteur_image=500
for y in range(hauteur_image):
    for x in range(largeur_image):
        r,v,b=img.getpixel((x,y))
        print("rouge : ",r,"vert : ",v,"bleu : ",b)
print("fin")
```

Quelques commentaires sur ce programme :

- Nous commençons par définir les variables "largeur_image" et "hauteur_image" ("largeur_image=500" et "hauteur_image=500"). Je pense que vous aurez compris que notre

image "pomme.jpg" fait 500 pixels de large et 500 pixels de haut. Si vous désirez travailler avec une autre image, il faudra veiller à bien modifier la valeur de ces deux variables.
- Les 2 boucles "for" nous permettent de parcourir l'ensemble des pixels de l'image :

```
for y in range(hauteur_image):
    for x in range(largeur_image):
                ...
```

- Le plus important ici est de bien comprendre que dans la suite du programme, les variables x et y vont nous permettre de parcourir l'ensemble des pixels de l'image : nous allons commencer avec le pixel de coordonnées (0,0), puis le pixel de coordonnées (1,0), puis le pixel de coordonnées (2,0)...jusqu'au pixel de coordonnées (499,0). Ensuite, nous allons changer de ligne avec le pixel de coordonnées (0,1), puis le pixel de coordonnées (1,1)...bref, le dernier pixel sera le pixel de coordonnées (499,499), tout cela grâce à la double boucle "for" !
- "r,v,b=img.getpixel((x,y))" cette ligne ne devrait pas poser de problème, nous avons juste remplacé les coordonnées des pixels par (x,y) afin de considérer l'ensemble des pixels de l'image.
- "print("rouge : ",r,"vert : ",v,"bleu : ",b)" nous imprimons les valeurs des canaux RVB pour chaque pixel de l'image.
- "print("fin")" ATTENTION cette ligne n'est pas dans la double boucle (pas de décalage), le mot "fin" ne sera donc affiché qu'une seule fois (après avoir parcouru l'ensemble des pixels).

Compliquons un peu la chose en modifiant tous les pixels de l'image :

6. Copier et tester le code suivant :

```
from PIL import Image
img = Image.open("pomme.jpg")
largeur_image=500
hauteur_image=500
for y in range(hauteur_image):
    for x in range(largeur_image):
        r,v,b=img.getpixel((x,y))
        n_r=v
        n_v=b
        n_b=r
        img.putpixel((x,y),(n_r,n_v,n_b))
img.show()
```

   Expliquez en quelques mots ce que fait ce programme.

7. En vous inspirant de ce qui a été fait au à la question 6, écrire un programme qui inverse les valeurs des canaux bleu et rouge sans changer la valeur du canal vert

Appeler l'enseignant pour valider cette question.

8. Après avoir fait quelques recherches sur le "négatif d'une image", écrivez un programme qui donne le négatif d'une image.

Appeler l'enseignant pour valider cette question.

9. Après avoir fait quelques recherches sur les "images en niveau de gris", écrivez un programme qui transforme une "image couleur" en une "image en niveau de gris".

   Petite astuce qui pourrait vous aider : en Python pour avoir une division entière (le résultat est un entier), il faut utiliser l'opérateur // à la place de l'opérateur /.

Appeler l'enseignant pour valider cette question.

10. Testez les programmes écrient dans les questions 8 et 9 avec une image de votre choix (attention aux variables "largeur_image" et "hauteur_image").
11. Copier et tester le programme suivant:

```
from PIL import Image
img = Image.open("pomme.jpg")
largeur_image=500
hauteur_image=500
for y in range(hauteur_image):
    for x in range(largeur_image):
        r,v,b=img.getpixel((x,y))
        if b<200:
            n_b=255-b
        else :
            n_b=b
        img.putpixel((x,y),(r,v,n_b))
img.show()
```

   Expliquer en quelques mots ce que fait le programme.

12. Copier et tester le programme suivant :

```
from PIL import Image
img = Image.open("pomme.jpg")
largeur_image=500
hauteur_image=500
for y in range(hauteur_image):
    for x in range(largeur_image):
        r,v,b=img.getpixel((x,y))
        if v>100 and y>250:
            n_v=0
        else :
            n_v=255
        img.putpixel((x,y),(r,n_v,b))
img.show()
```

Expliquer en quelques mots ce que fait le programme.

JHermilier