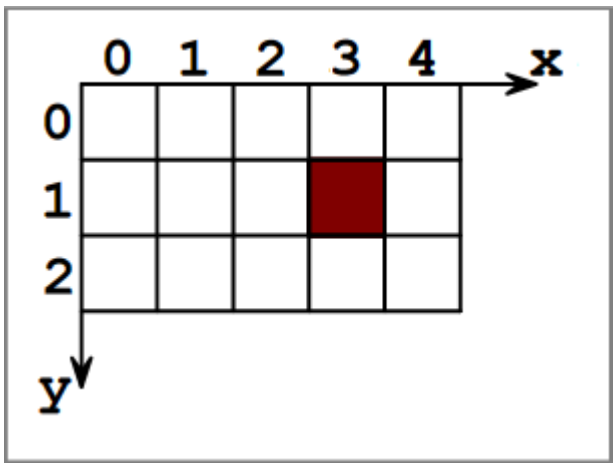
1 TP 12 - Les images

TD n°12 : Les IMAGES - INTRODUCTION	Thème : Transversal
	COURS et EXERCICES

I. Notion d'images

1.1 ⏩ VOCABULAIRE

- Un carré est appelé pixel (abréviation de picture element, élément d'image).
- Chaque pixel est repéré par ses coordonnées dans un repère dont l'origine est en haut à gauche de l'image :



- Plus la densité de pixels (résolution de l'image) sera élevée, plus l'image sera de bonne qualité mais plus le poids de cette image en octet sera important.
- La taille de l'image indique le nombre de pixels disposés sur la largeur et sur la hauteur.

- Le format de l'image correspond au standard utilisé pour coder l'image. Parmi les formats d'images bitmap les plus connus, on trouve .jpg, .png ou encore .gif.

II. Couleurs d'une image

A chaque pixel est associée une couleur, représentée sous la forme d'un code chiffré : on dit que l'image est numérisée.
Parmi les différents encodages de couleur, on distinguera :

- Le **noir et blanc** avec un encodage binaire : 0 ou 1 ;
- Le **monochrome** (aussi appelé «**nuances de gris**») avec un encodage sur une échelle de valeurs de 0 à n
  (généralement $n = 255$ pour coder une nuance sur un octet) ;
- La **couleur** (sans transparence) à partir d'un mélange des trois couleurs Red, Green, Blue (généralement abrégé RGB). Dans ce cas, chacune des trois composantes peut prendre des valeurs de 0 à 255 ce qui permet d'obtenir plus de 16 millions de couleurs (256 x 256 x 256).

III. Matricielle vs Vectorielle

Matricielle

Comme indiquée ci-dessus, une image créée pixel par pixel est appelée image bitmap ou ma- tricielle.

- **Avantage** : c'est ainsi qu'un appareil photo ou que l'oeil humain voit l'image.
- **Inconvénient** : zoomer sur l'image revient à grossir la taille des pixels et donc à voir apparaitre des carrés.

Vectorielle

Une image créée à l'aide d'une description géométrique des éléments qui la composent est appelée image vectorielle. Le format .svg est un format usuel d'image vectorielle.

- **Avantage** : zoomer sur l'image revient à changer d'échelle pour la représentation des éléments qui la composent sans corrompre

l'esthétique.
- **Inconvénient** : il est difficile de modéliser mathématiquement une photo issue de la réalité.

IV. Créer une nouvelle image avec le module PIL

Le module PIL permet de manipuler un grand nombre de formats d'image. Ce module n'est pas inclus d'office dans Python, il faut télécharger puis installer la bibliothèque pillow si on ne travaille pas dans le noteboook capytale.

Pour plus d'informations sur les possibilités de ce module, on pourra consulter Le site de référence.
Cette page présente comment concevoir une nouvelle image avec PIL en construisant ligne par ligne un programme. On commence par importer le module en en-tête du fichier :

```
from PIL import Image
```

On pourra sauvegarder les images souhaitées en rajoutant en début de programme :

```
import basthon
```

et en fin de programme

```
basthon.download("nom de l'image")
```

1.2 ⏩ Mise en pratique du module PIL

Exercice n°1 :

Le module PIL permet de créer très facilement l'image représentée ci-contre qui est un rectangle de 100 pixels de largeur et 256 pixels de hauteur, de couleur dégradée du noir (en haut) vers le rouge.
Pour créer une nouvelle image, il faut indiquer :

- le nom de la variable qui « contiendra » cette image ;

- l’encodage des couleurs de cette image
    - ’RGBA’ : couleurs avec gestion de la transparence (4 octets)
    - ’RGB’ : couleurs sans gestion de la transparence
    - ’L’ : nuances de gris
    - ’1’ : noir et blanc
- les dimensions en pixels (largeur et hauteur) de cette image (sous forme de tuple - liste non modifiable).

Question
Lancez la celulle suivante. Quelle remarque peut-on faire ?

```
##-- - - - Importation des modules - - - - -##
from PIL import Image

##-- - - - Variables et constantes - - - - -##
largeur = 100
hauteur = 256
couleur = 'RGB'

##-- - - - - Nouvelle image - - - - - -##
im = Image.new(couleur, (largeur, hauteur))

##-- - - - Fermeture et affichage - - - - -##
im.save('Degrade.jpg')
im.show()
```

Exercice n°2 :

Le module PIL permet «d’atteindre» et de modifier directement n’importe quel pixel de l’image grâce à ses coordonnées.
Les coordonnées d’un pixel de l’image sont sous la forme d’un tuple (x, y).
Selon l’encodage des couleurs du fichier, il faudra saisir une valeur entière différente pour le pixel correspondant :

- ’RGBA’ : un quadruplet (r, g, b, a) de valeurs comprises entre 0 et 255

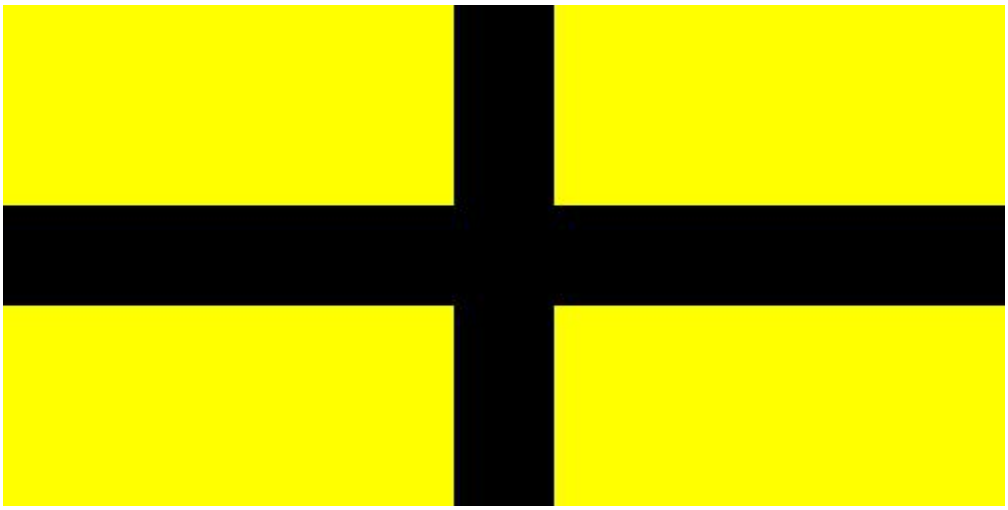
- ’RGB’ : un triplet (r, g, b) de valeurs comprises entre 0 et 255
- ’L’ : une valeur g comprise entre 0 et 255
- 1’ : une valeur nb comprise entre 0 et 1

Question :
Compléter le programme précédent afin d’obtenir l’image d’un rectangle de 100 pixels de largeur et 256 pixels de hauteur, de couleur dégradée du noir (en haut) vers le rouge.

```
##-- - - - Importation des modules - - - - -##
from PIL import Image
##-- - - - Variables et constantes - - - - -##
largeur = 100
hauteur = 256
couleur = 'RGB'
##-- - - - - Nouvelle image - - - - - -##
im = Image.new(couleur, (largeur, hauteur))
for x in range(......):
    for y in range(.......):
        im.putpixel((x, y), (.....,.....,....)) # Composante R
##-- - - - Fermeture et affichage - - - - -##
im.save('Degrade.jpg')
im.show()
```

Exercice n°3 :

Créer une image représentant le drapeaux suivant :



```
##-- - - - Importation des modules - - - - -##
from PIL import Image
##-- - - - Variables et constantes - - - - -##
largeur = 500
hauteur = 250
couleur = 'RGB'

##-- - - - - Nouvelle image - - - - - -##
im = Image.new(couleur, (largeur, hauteur),(255,255,0))  # on c
for x in range(......):
    for y in range(.......):
        im.putpixel((x, y), (.....,.....,....))

for x in range(......):
    for y in range(.......):
        im.putpixel((x, y), (.....,.....,....))




##-- - - - Fermeture et affichage - - - - -##
im.save('image1.jpg')
im.show()
```

Exercice n°4 :

Écrire un code Python pour dessiner le drapeau de la Norvège.

Exercice n°5 : Avec des maths - Approfondissement

Vous allez tracer le drapeau de l'Union Jack

Dessiner en diagonale

```
import matplotlib.pyplot as plt
from PIL import Image

#Creation d'une image blanche RGB 10x10
dessin5=Image.new("RGB",(600,400),(0,0,102))
# Definitions des couleurs
blanc = (255, 255, 255)
#On parcourt tous les pixels de l'image:
for col in range(0,600):
    for ligne in range(0,400):
```

```
        if ligne >= 0.67*col - 50 and ligne <= 0.67*col + 50:
            dessin5.putpixel((col,ligne),blanc)
plt.imshow(dessin5)
plt.show()
plt.close()
dessin5.show()
```

Explication :

1. Sur le graphique, placer les points A(0; 0) et B(600; 400).
2. Calculer le coefficient directeur de la droite (AB) :
3. Combien vaut son ordonnée à l'origine?
4. En déduire une équation de la droite (AB) :
5. Tracer en vert la droite d'équation : $\mathrm{y} = 0,67\mathrm{x} - 50$
6. Tracer en vert la droite d'équation : $\mathrm{y} = 0,67\mathrm{x} + 50$

Vous l'avez remarqué ! **Les ordonnées vont de haut en bas ici**. Ce n'est pas le cas en Maths!

Étude de la seconde diagonale :

1. Sur le résultat, placer les points C(0; 400) et D(600; 0).
2. Calculer le coefficient directeur de la droite (CD).
3. Combien vaut son ordonnée à l'origine ?
4. En déduire une équation de la droite (CD).
5. Quelles sont les conditions pour dessiner la deuxième bande blanche entourant la seconde diagonale :

```
if ligne >= ............. and ligne <= .................:
    dessin5.putpixel((col,ligne),blanc)
```

6. Compléter le code Python pour dessiner l'Union Jack. (Commencer par les bandes en diagonale)

```

```

7 Pour les plus rapides, dessiner en Python le drapeau des Seychelles : (Commencer par déterminer les équations des droites en jeu.)

V. Modifier des images avec le module PIL

1.3 ⏩ Ouvrir un fichier image

Pour ouvrir un fichier image au format .jpg (par exemple) située dans le répertoire courant, on utilise :

```
im = Image.open('perroquetmulticolore.jpg')
```

Afin d'être "manipulée", cette image est affectée à la variable nommée `im` dans le programme.

1.4 ⏩ Obtenir des informations sur cette image

Une fois le fichier ouvert, on peut connaître et utiliser :

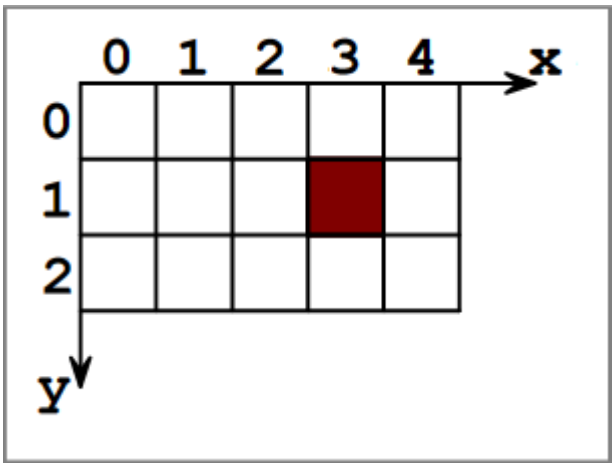
- L'encodage des couleurs du fichier : 'RGBA' (couleurs avec gestion de la transparence), 'RGB', 'L' (gris) ou '1' (noir et blanc).

```
couleur = im.mode
print("L'encodage des couleurs est ", couleur)
```

Les dimensions de l'image (en pixels) sous forme d'un tuple (liste non modifiable (largeur, hauteur)) :

```
l, h = im.size
print(f'largeur : {l} et hauteur : {h}')
```

Les composantes couleur du pixel de coordonnées (x, y) dans l'image. L'illustration ci-dessous présente comment PIL « lit » une image.



Les coordonnées du pixel rouge de cette image sont donc (3, 1) : à nouveau, c'est un tuple (il doit être entouré de parenthèses...).

```
x, y = 120, 120 # Coordonnées choisies au hasard
triplet = im.getpixel((x, y))
print("Ce pixel a pour composantes : ", triplet)
```

Remarque importante

Dans le cas d'une image encodée en 'RGB', triplet prend pour valeur le triplet d'entier (r, v, b) correspondant aux composantes couleurs du pixels de coordonnées x en colonne et y en ligne. Par conséquent,

- Ou bien on récupère l'intégralité du triplet puis on récupère les composantes r,v,b une à une avec les instructions

```
triplet = im.getpixel((x, y))
r = triplet[0]
v = triplet[1]
b = triplet[2]
```

- Ou bien on récupère directement composante par composante avec l'instruction

```
r, v, b = im.getpixel((x, y))
```

```
from PIL import Image
issoire = Image.open('issoire_tour_horloge.jpg')

couleur = issoire.mode
print(f"L'encodage des couleurs est {couleur}")

l, h = issoire.size
print(f'largeur : {l} et hauteur : {h}')

issoire.show()
```

1.4.1 ⏩ Pour éclaircir une image, on peut augmenter de 50% les composantes couleurs de chaque pixel.

```
def eclaircissement(im_originale):
    """
    fonction qui renvoie l'image im_originale éclaircie
    """

    # récupération des dimensions de l'image originale
    L,H = im_originale.size

    # création d'une image vierge, de même format et même dimension que l'
    im_modifiee = Image.new( mode=im_originale.mode , size=(L,H) )

    # ouverture de l'accès aux pixels des deux images
    pix_origine = im_originale.load()
    pix_modifie = im_modifiee.load()

    # on parcourt tous les pixels des images
    for x in range(L):
        for y in range(H):
            R,G,B = pix_origine[x,y] #récupération des composantes R,G,B du

            R = int(R*1.5)
            G = int(G*1.5)
            B = int(B*1.5)

            pix_modifie[x,y] = R,G,B #écriture des composantes R,G,B du pix

    # on renvoie l'image modifiée
    return im_modifiee
```

```
eclaircissement(issoire)
```

1.4.2 ⏩ Pour assombrir une image, on peut diminuer de 50% les composantes couleurs de l'image d'origine.

Question 1 :

Écrire une fonction Python assombrissement qui permet d'assombrir une image, puis tester cette fonction.

Aide

On pourra commencer par effectuer un copier/coller de la fonction eclaircissement puis l'adapter.

```
# Ecrire ici la fonction assombrissement
```

```
# Tester ici un appel à la fonction assombrissement
```

1.4.3 ⏩ Filtre

Le principe du filtre rouge consiste à conserver les composantes rouges des pixels de l'image et à remplacer les autres par 0.

Question 2

a. Écrire une fonction Python filtre_R qui applique le filtre rouge à une image, puis tester cette fonction.

```
# Écrire ici la fonction filtre_R
```

```
# Tester ici un appel à la fonction filtre_R
```

b. Écrire sur le même principe des fonctions Python filtre_G et filtre_B pour les filtres vert et bleu.

```
# Écrire ici la fonction filtre_G
```

```
# Tester ici un appel à la fonction filtre_G
```

```
# Écrire ici la fonction filtre_B
```

```
# Tester ici un appel à la fonction filtre_B
```

Question 3 : Niveau de gris

Dans cette question, on souhaite convertir une image couleur en image en niveau de gris.
En RGB, on obtient un pixel en niveau de gris lorsque ses 3 composantes couleur sont égales.

a. Une première méthode consiste à remplacer chacune des composantes couleur d'un pixel par la moyenne des trois composantes initiales.
Écrire une fonction Python gris_rapide qui effectue cette conversion.
Tester cette fonction.

```
# Écrire ici la fonction gris_rapide
```

```
# Tester ici un appel à la fonction gris_rapide
```

b. La perception des composantes couleur par l'oeil humain diffère selon ces couleurs.
Ainsi, l'organisme CIE (Commission Internationale de l'Éclairage) préconise la méthode suivante pour convertir une image en niveau de gris :

Conversion en niveau de gris :

On attribue à chaque composante couleur la valeur obtenue en additionnant :

- 21,25% de la valeur de la composante rouge initiale;
- 71,54% de la valeur de la composante verte;
- 7,21% de la valeur de la composante bleue.

Écrire une fonction Python gris_precis qui réalise cette conversion.

```
# Écrire ici la fonction gris_precis
```

```
# Tester ici un appel à la fonction gris_precis
```

Question 4.

Pour obtenir le négatif d'une image, on remplace chaque composante de chaque pixel par la différence entre 255 et cette composante initiale.

Écrire une fonction Python negatif qui renvoie le négatif d'une image.

```
# Écrire ici la fonction negatif
```

```
# Tester ici un appel à la fonction negatif
```

Question 5

Dans cette question, on souhaite convertir l'image en noir et blanc (sans niveaux de gris).

a. Rappeler les codes R,G,B correspondant respectivement au noir et au blanc.
b. On décide qu'un pixel sera blanc si et seulement si la moyenne de ses composantes initiales est supérieure ou égale à 128.
Ecrire une fonction Python NB qui réalise cette transformation.

```
# Écrire ici la fonction NB
```

```
# Tester ici un appel à la fonction NB
```

Question 6.

Pourriez vous écrire un programme qui change la couleur du rouge à lèvre de cette dame.

```
from PIL import Image
red = Image.open('redn.jpg')

couleur = red.mode
print(f"L'encodage des couleurs est {couleur}")

l, h = red.size
print(f'largeur : {l} et hauteur : {h}')

red.show()
```

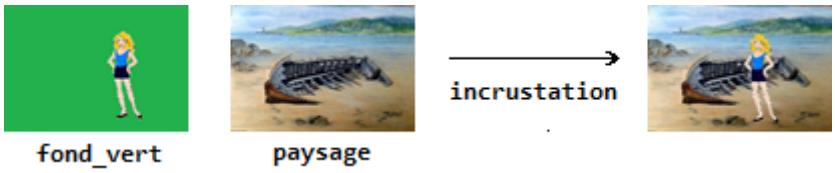
```
L'encodage des couleurs est RGB
largeur : 400 et hauteur : 239
```

VI. Incrustation sur fond vert

Le procédé d'incrustation sur fond vert consiste à remplacer tous les pixels de l'image sur fond vert dont la composante verte est « dominante » par rapport aux deux autres par les pixels de l'image du fond à incruster.

1.4.4 ⏩ On souhaite réaliser l'incrustation sur fond vert de la façon suivante :

- On fixe des coefficients de seuils $\mathrm{C_R} = 1,3$ et $\mathrm{C_B} = 1,3$.
- Pour chaque pixel de l'image sur fond vert, si ses composantes R, G, B vérifient $\mathrm{G} > \mathrm{C_R} \times \mathrm{R}$ et $\mathrm{G} > \mathrm{C_B} \times \mathrm{B}$, alors on récupère le pixel du paysage, sinon on conserve le pixel de l'image sur fond vert.



Incrustation

a. Écrire la fonction Python incrustation qui convient.

On utilisera les images : - personnage_test.jpg - paysage_test.jpg

```
# Écrire ici la fonction incrustation
```

b. Tester la fonction incrustation avec les photos suivantes : personnage_test.jpg et paysage_test.jpg.
Si nécessaire, on pourra modifier les valeurs des coefficients de seuil C_R et C_B.

```
# Tester ici la fonction incrustation
```

VII. Réaliser des montages et transformations d'images

1.5 ⏩ Exercice : Bordure

Bordure

Programmer une fonction def bordure(im,coul) qui crée une bordure de couleur autour de votre image.

```

```

1.6 ⏩ Exercice : Andy Warhol

Andy Warhol

Maintenant testez le code qui permet de coller l'image de pomme dans une image de fond :

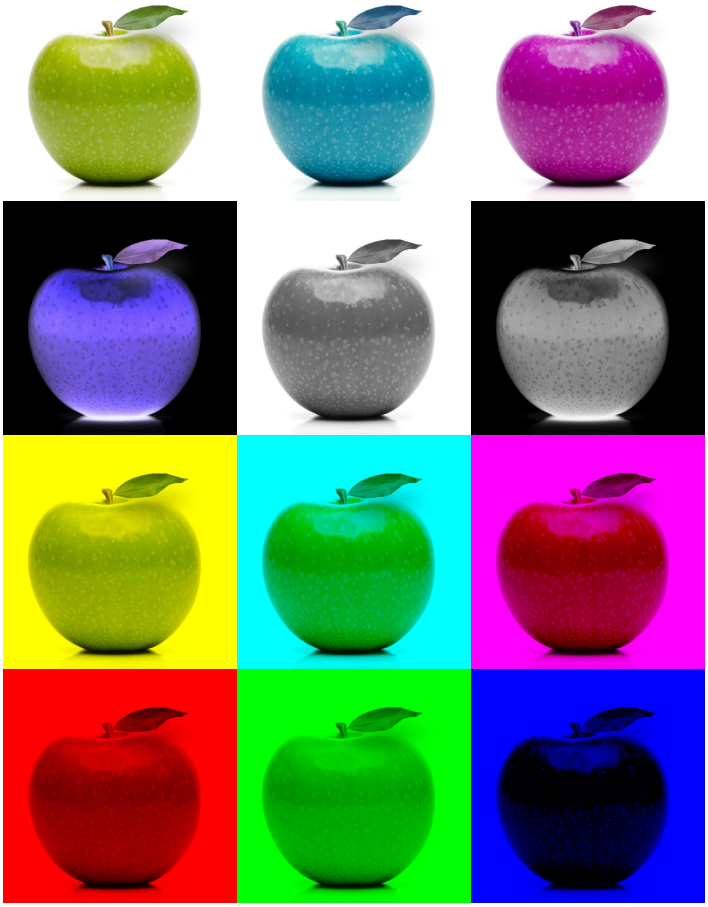
```
from PIL.Image import *
# On ouvre l ' image de pomme
im = open("pomme.jpg")
# On récupère la dé finition de l ' image de pomme
(L, H) = im.size
couleur = im.mode
# Je crée une nouvelle image pouvant contenir 4 fois l ' image
fond = new(couleur, (L * 2, H * 2))
# Je colle l ' image en (0, 0) ( coin en haut à gauche)
fond.paste(image, (0, 0))
fond.save("resultat2.jpg", "JPEG")
fond.show()
```

Maintenant modifiez ce code pour obtenir l'image suivante :

```

```

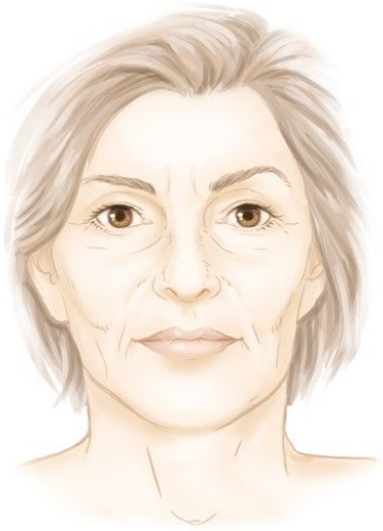
puis en réutilisant tout ce que vous avez vu dans ce TP, écrivez un programme permettant d'obtenir l'image suivante :



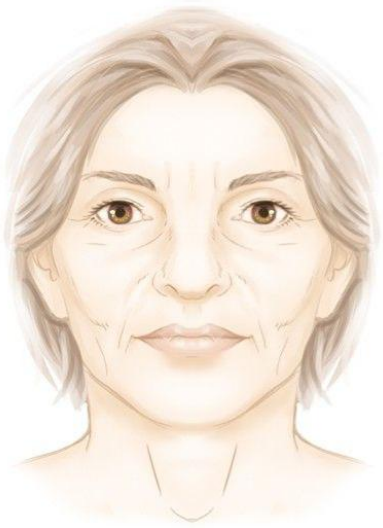
1.7 ⏩ Exercice : Symétrie

Symétrie

Créer une fonction qui affiche une image ou la partie de droite est remplacée par sa symétrie. (A tester avec une photo d'identité pour voir la symétrie de votre visage)



- >



1.8 ⏩ A vous de jouer

A vous de jouer

Voici quelques exemples de transformation possibles :