STEGANOGRAPHIE

FICHE PROFESSEUR

- **Niveau et durée :** 2nde (SNT), 3h pour la totalité (1h30 pour une des deux parties)

- **Objectifs pédagogiques :** Utiliser un langage de programmation (python) pour traiter une image. Manipuler la notion de composantes RGB, se repérer sur une image, utiliser des instructions conditionnelles et des boucles « for » pour traiter une image.

- **La situation-problème :** On veut transmettre un message secret. Pour cela, on va le « cacher » dans une image. Il faut donc élaborer deux algorithmes : un pour coder le message, un pour le décoder.

- **Prérequis :**

Les élèves doivent avoir vu les affectations, les instructions conditionnelles et la boucle *for* sur Python.
Pour le traitement d'images, les élèves doivent avoir vu les instructions : o*pen, getpixel, show* et doivent savoir ce que sont les composantes RGB d'un pixel.
Les élèves doivent également connaître l'écriture binaire des nombres.

Nous avons réalisé préalablement deux séances de découverte de traitement de l'image (environ 3 h).
Lien Capitale vers l'activité : https://capytale2.ac-paris.fr/web/c/0457-570552

- **Déroulement :**

Cette activité est composée de deux parties indépendantes.

Partie A : Insertion d'un message TEXTE dans une photo

I - Codage du message à dissimuler :

On commence par cacher un mot de 5 lettres dans un morceau de l'image (un rectangle de 6×5 pixels).
Pour cela, on modifie la composante B du code (R, G, B) de chaque pixel.

- **Étape 1 :** On code chaque caractère du mot en base 10.
  Exemple avec le mot OUEST : 14 ; 20 ; 4 ; 18 ; 19.

- **Etape 2 :** On convertit en binaire le codage décimal.
  Exemple : Le code binaire de 14 est 001110 ;
  Le code binaire de 20 est 010100 ;
  Le code binaire de 4 est 000100 ;
  Le code binaire de 18 est 010010 ;
  Le code binaire de 19 est 010011.

- **Étape 3 :** On récupère la composante B de chaque pixel de la grille qu'on écrit en binaire.

54 00110110	**78** 01001110	**93** 01011101	**176** 10110000	**154** 10011010	**47** 00101111
76 01001100	**115** 01110011	**129** 10000001	**165** 10100101	**171** 10101011	**84** 01010100
51 00110011	**109** 01101101	**120** 01111000	**146** 10010010	**121** 01111001	**120** 01111000
54 00110110	**35** 00100011	**93** 01011101	**108** 01101100	**116** 01110100	**118** 01110110

48 0 0 1 1 0 0 0 **0**	**49** 0 0 1 1 0 0 0 **1**	**58** 0 0 1 1 1 0 **1** **0**	**92** 0 1 0 1 1 1 0 **0**	**81** 0 1 0 1 0 0 0 **1**	**110** 0 1 1 0 1 1 1 **0**

- **Étape 4 :** On dissimule le codage binaire du mot dans la grille de pixels : on modifie le dernier chiffre (de l'écriture binaire) de chaque pixel. La composante B du code (R,G,B) a été modifiée d'au plus 1.

Exemple :

54 0 0 1 1 0 1 1 **0**	**78** 0 1 0 0 1 1 1 **0**	**93** 0 1 0 1 1 1 0 **1**	**177** 1 0 1 1 0 0 0 **1**	**155** 1 0 0 1 1 0 1 **1**	**46** 0 0 1 0 1 1 1 **0**
7 6 0 1 0 0 1 1 0 **0**	**115** 0 1 1 1 0 0 1 **1**	**128** 1 0 0 0 0 0 0 **0**	**165** 1 0 1 0 0 1 0 **1**	**170** 1 0 1 0 1 0 1 **0**	**84** 0 1 0 1 0 1 0 **0**
50 0 0 1 1 0 0 1 **0**	**108** 0 1 1 0 1 1 0 **0**	**120** 0 1 1 1 1 0 0 **0**	**147** 1 0 0 1 0 0 1 **1**	**120** 0 1 1 1 1 0 0 **0**	**120** 0 1 1 1 1 0 0 **0**
54 0 0 1 1 0 1 1 **0**	**35** 0 0 1 0 0 0 1 **1**	**92** 0 1 0 1 1 1 0 **0**	**108** 0 1 1 0 1 1 0 **0**	**117** 0 1 1 1 0 1 0 **1**	**118** 0 1 1 1 0 1 1 **0**
48 0 0 1 1 0 0 0 **0**	**49** 0 0 1 1 0 0 0 **1**	**58** 0 0 1 1 1 0 **1** **0**	**92** 0 1 0 1 1 1 0 **0**	**81** 0 1 0 1 0 0 0 **1**	**111** 0 1 1 0 1 1 1 **1**

Observation à faire noter :

On observe que les valeurs en base 10 des composantes B' de la nouvelle image sont obtenues ainsi :

- **Si on veut dissimuler un 0 : Si B est pair, alors B' = B ;**
  Si B est impair, alors B' = B – 1 ;
- **Si on veut dissimuler un 1 : Si B est pair, alors B' = B + 1 ;**
  Si B est impair, alors B' = B.

II- Décodage d'un message

On réalise l'opération inverse : à partir des composantes B (écrites en base 10), on retrouve le mot codé :

→ D'abord la décomposition en base 2 du code de chaque caractère ;
→ En suite la décomposition en base 10 du code de chaque caractère ;
→ Le mot dissimulé.

Partie B : Programmation en Python du codage et décodage stéganographique

I - Codage du message à dissimuler :

On dissimule une image en noir et blanc dans une photo en couleurs.

Étape 1 :

Lorsqu'un pixel est blanc, la composante B est égale à 255 ;
Lorsqu'un pixel est noir, la composante B est égale à 0.

Ainsi la composante B des pixels de cette image est :

0	0	0	0	0
0	255	255	255	255
0	255	255	255	255
0	255	255	255	255
0	255	255	255	255

STEGANOGRAPHIE

Étape 2 : Algorithme de codage
On utilise l'algorithme suivant :

Si un pixel de l'image code1.bmp est blanc alors :

On remplace le chiffre des unités du code binaire de la composante B du pixel associé par 0

sinon :

On remplace le chiffre des unités du code binaire de la composante B du pixel associé par 1

On reformule l'observation faite à l'étape 4 précédente pour pouvoir le programmer avec Python :

Si la composante B d'un pixel de code1.bmp est égal à 255 alors :

la composante B' du pixel associé sera égale à B – B%2 (B%2 donne le reste de la division euclidienne de B par 2)

sinon :

la composante B' du pixel associé sera égale à B – B%2 +1

Les premières composantes B' de photocodee.bmp seront :

55	79	93	177	155	47
77	114	128	164	170	84
51	108	120	146	120	120
55	34	92	108	116	118
49	48	58	92	80	110

Étape 3 : On complète un script qui va permettre d'automatiser ce travail.
Lien vers un notebook Capytale qui peut être utilisé : https://capytale2.ac-paris.fr/web/c/0c79-616415
Corrigé de l'activité : https://capytale2.ac-paris.fr/web/c/3191-614225

On complète les parties bleues du script qui permet de cacher le message **code.bmp** dans l'image **photo.bmp**.

```
from PIL import Image

photo=Image.open('photo3.bmp')  #On charge la photo à coder que tu as choisie
code=Image.open('code2.bmp')    #On charge le code que tu as choisi

photocodee=Image.new('RGB',(300,150))  #La photo codée aura pour dimensions 300*150
for x in range(300):
    for y in range(150):
        (R1,G1,B1) = photo.getpixel((x,y))
        (R2,G2,B2) = code.getpixel((x,y))
        if B2==255:
            photocodee.putpixel((x,y),(R1,G1,B1-B1%2))
        else :
            photocodee.putpixel((x,y),(R1,G1,B1-B1%2+1))

photo.show()   #On affiche la photo initiale
photocodee.show()   #On affiche la photo avec le code dissimulé
```

Aide pour les lignes 12 et 14 : on fait observer les valeurs de `a%2,a-a%2,a-a%2+1` pour a un nombre entier.

STEGANOGRAPHIE

II - Décodage de photo-codee.jpg :

On décode des photos dans lesquelles on a caché des images en noir et blanc.
Lien vers un notebook Capytale : https://capytale2.ac-paris.fr/web/c/fe1b-616419
Corrigé de l'activité : https://capytale2.ac-paris.fr/web/c/5b33-614229

On complète les parties bleues du script ci-dessous.

```
from PIL import Image

photocodee=Image.open('photocodee1.bmp') #Remplacer 1 par la valeur de ton choix

decodage=Image.new('RGB',(300,150))
for x in range(300):
    for y in range(150):
        (R,G,B) = photocodee.getpixel((x,y))
        if B%2==0 :
            decodage.putpixel((x,y),(255,255,255))
        else :
            decodage.putpixel((x,y),(0,0,0))

decodage.show()   #On affiche le message qui était dissimulé
```

- **Analyse du dispositif :**

Les élèves ont apprécié l'échange de messages codés dans la partie A et le décodage de photos dans la partie B. Les parties A et B étant bien séparées, on peut choisir de n'en traiter qu'une.

La partie A nécessite que l'écriture des nombres en base 2 soit comprise.

Dans la partie B, les élèves ont eu plus de facilité à décoder des images qu'à les coder. Lorsque leur programme contenait une erreur, ils ne le visualisaient pas : en effet l'image de départ et l'image codée sont indiscernables à l'œil nu.
Il est assez judicieux de commencer par l'activité « Décodage d'une photo ». Le script utilisé pour le décodage peut alors être intégré à la page « Codage d'une photo » pour vérifier le script.

Si l'enseignant le souhaite, il peut ajouter les images codées par les élèves dans le notebook pour les utiliser dans la partie « Décodage d'une photo ».

- **Dans les programmes du cycle :**

En SNT :

Contenus	Capacités attendues
Traitement d'image	Traiter par programme une image pour la transformer en agissant sur les trois composantes de ses pixels.

- **Les six compétences majeures**

Compétences pour le lycée
- Modéliser Domaines du socle : 1, 2, 4 • Utiliser, comprendre, élaborer une simulation numérique ou géométrique prenant appui sur la modélisation et utilisant un logiciel.
- Représenter Domaines du socle : 1, 5 • Passer d'un mode de représentation à un autre.