

---

# BTS SIO

## SOUS-ÉPREUVE E22

### ALGORITHMIQUE APPLIQUÉE

### CONTRÔLE EN COURS DE FORMATION

---

#### Déroulement de l'épreuve

Cette épreuve de Contrôle en cours de Formation (CCF) se déroule en trois étapes :

– **Écrit (30 minutes)**

Vous devez traiter l'étape 1' du sujet. Pour cette partie, l'ordinateur est interdit mais la calculatrice est autorisée.

**Vous inscrirez vos réponses dans le document réponse à la fin du sujet.**

Les algorithmes à écrire peuvent être rédigés en **langage naturel** ou en PYTHON mais ni en C# ni en VB.NET.

**À la fin de l'étape 1, votre document réponse doit être remis à la personne surveillant l'épreuve.** Vous garderez le sujet.

– **Machine (30 minutes)**

Vous devez traiter l'étape 2 du sujet à l'aide d'un ordinateur. Le langage utilisé est celui travaillé dans l'année, à savoir PYTHON. Vous sauvegarderez votre travail sur la clé USB fournie.

La durée totale pour effectuer les deux premières étapes est exactement d'une heure.

– **Oral (20 minutes au maximum)**

Cette partie se déroule en deux temps. Tout d'abord, vous disposez de 10 minutes pour présenter votre travail de l'étape 2 puis, au cours des 10 minutes suivantes, un entretien permet de préciser votre démarche.

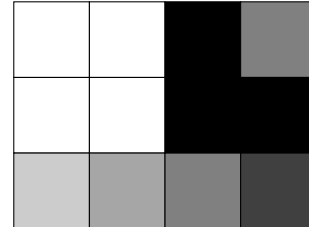
**À la fin de l'épreuve le sujet devra être rendu à l'examineur.**

Une image rectangulaire en noir et blanc peut être représentée par une liste de lignes qui sont des listes d'entiers compris entre 0 (pour le noir) et 255 (pour le blanc).

Par exemple la liste suivante :

```
[[255, 255, 0, 128],  
 [255, 255, 0, 0],  
 [204, 165, 128, 64]]
```

correspond à l'image ci-contre.



## Étape 1

### Question 1

Si `lst` représente une image de  $n$  lignes par  $p$  colonnes.

1. Quel est le nombre d'éléments de `lst` ?
2. Quel est le nombre d'éléments de `lst[0]` ?

### Question 2

Compléter sur votre copie le code de la fonction `binarise` qui

- en entrée prend une liste représentant une image;
- modifie chaque pixel en mettant à 0 ceux qui sont inférieurs à 128 et à 255 ceux qui sont supérieurs.

```
fonction binarise(lst)  
    variables  
        i, j, n, p : entiers  
    n ← longueur(lst)  
    p ← longueur(...)  
    pour i allant de 0 à ..... faire  
        .....  
            si lst[i][j] < ... alors  
                ...  
            sinon  
                ...
```

```
    fin pour
fin pour
```

### Question 3

Compléter sur votre copie le code de la fonction `cree_image_vide` qui

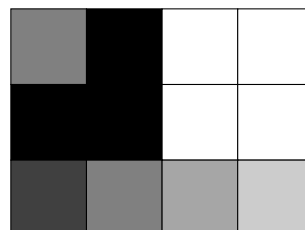
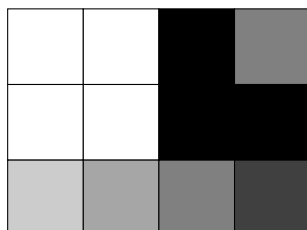
- en entrée prend deux entiers `n` et `p`;
- renvoie une liste représentant une image «toute noire» de `n` lignes et `p` colonnes.

```
fonction cree_image_vide(n, p)
    variables
        i, j : entiers
        resultat, ligne : liste
    resultat ← liste vide
    pour i allant de 0 à ... faire
        ligne ← ...
        pour j allant de 0 à ... faire
            ajouter 0 à la fin de ligne
        fin pour
    ...
fin pour
...
```

On aimerait coder une fonction `miroir_vertical` qui

- en entrée prend une liste `lst` représentant une image;
- crée une liste vide aux mêmes dimensions que `lst` et remplit ses pixels en «renversant chaque ligne»;
- renvoie cette liste.

Par exemple, en appliquant cette fonction à l'image de gauche, on doit obtenir celle de droite



#### Question 4

Compléter sur votre copie le code de la fonction `miroir_vertical`.

```
fonction cree_image_vide(lst)
    variables
        i, j : entiers
        resultat: liste
    n ← ...
    p ← ...
    resultat ← cree_image_vide(...)
    pour i allant de 0 à ... faire
        pour j allant de 0 à ... faire
            resultat[i][j] ← ...
        fin pour
    fin pour
    renvoyer resultat
```

## Étape 2

#### Question 5

Ouvrir le fichier `traitement_image.py` et coder les fonctions manquantes.

#### Question 6

Coder la fonction `miroir_horizontal` qui

- en entrée prend une liste `lst` qui représente une image;
- renvoie une liste qui est l'image de `list` par une symétrie horizontale.