TD4 – éléments de corrigé (algorithmes)

Exercice 2

Algorithme 1 : CreerImage2DEntiers
Principe : Parcourir les lignes (i) et colonnes (j) de l'image et mettre la valeur i*255 + j*255 / (hauteur + largeur) ou 255 si le résultat est plus grand.

```
Entrée :        - hauteur : entier
                - largeur : entier
Local :         - i : entier
                - j : entier
                - valeur : entier
Sortie :        image : tableau 2D de hauteur * largeur entiers
Début
|       Soit image = tableau 2D de hauteur * largeur entiers
|       Pour i de 0 à hauteur − 1 avec un pas de +1 Faire
|       |       Pour j de 0 à largeur − 1 avec un pas de +1 Faire
|       |       |       Soit valeur = i * 255 + j * 255 / (hauteur + largeur)
|       |       |       Si valeur > 255 Alors
|       |       |       |       image[i,j] = 255
|       |       |       Sinon
|       |       |       |       image[i,j] = valeur
|       |       |       Fin Si
|       |       Fin Pour
|       Fin Pour
|       Retourner image
Fin
```

Algorithme 2 : SeuillerImage2DEntiers
Principe : Créer une image vide de même dimensions que l'image d'entrée. Parcourir les lignes (i) et colonnes (j) de l'image d'entrée. Si la valeur de la case i,j est inférieure au seuil, mettre dans la même case de l'image vide la valeur 0, sinon recopier la valeur de l'image d'entrée.

```
Entrée :        - image : tableau 2D de hauteur * largeur entiers
                - hauteur : entier
                - largeur : entier
                - seuil : entier
Local :         - i : entier
                - j : entier
Sortie :        imageSeuillee : tableau 2D de hauteur * largeur entiers
Début
|       Soit imageSeuillee = tableau 2D de hauteur * largeur entiers
|       Pour i de 0 à hauteur − 1 avec un pas de +1 Faire
|       |       Pour j de 0 à largeur − 1 avec un pas de +1 Faire
|       |       |       Si image[i,j] < seuil Alors
|       |       |       |       imageSeuillee[i,j] = 0
|       |       |       Sinon
|       |       |       |       imageSeuillee[i,j] = image[i,j]
|       |       |       Fin Si
|       |       Fin Pour
|       Fin Pour
|       Retourner imageSeuillee
Fin
```

Algorithme 3 : BinariserImage2DEntiers
Principe : Créer une image vide de même dimensions que l'image d'entrée (booléens cette fois-ci). Parcourir les lignes (i) et colonnes (j) de l'image d'entrée. Si la valeur de la case i,j est inférieure au seuil, mettre dans la même case de l'image vide la valeur faux, sinon y mettre la valeur vrai.

```
Entrée :        - image : tableau 2D de hauteur * largeur entiers
                - hauteur : entier
                - largeur : entier
                - seuil : entier
Local :         - i : entier
                - j : entier
Sortie :        imageBinaire : tableau 2D de hauteur * largeur booléens
Début
|       Soit imageBinaire = tableau 2D de hauteur * largeur booléens
|       Pour i de 0 à hauteur − 1 avec un pas de +1 Faire
|       |       Pour j de 0 à largeur − 1 avec un pas de +1 Faire
|       |       |       Si image[i,j] < seuil Alors
|       |       |       |       imageBinaire[i,j] = faux
|       |       |       Sinon
|       |       |       |       imageBinaire[i,j] = vrai
|       |       |       Fin Si
|       |       Fin Pour
|       Fin Pour
|       Retourner imageBinaire
Fin
```

Algorithme 4 : CalculerHistogrammeImage2DEntiers
Précondition : l'image ne contient que des valeurs entre 0 et 255.
Principe : Créer un tableau 1D vide de 256 cases (initialisées à 0). Parcourir les lignes (i) et colonnes (j) de l'image d'entrée. Ajouter à la case du tableau 1D dont l'indice est la valeur de la case de l'image 2D indexée i,j la valeur + 1.

```
Entrée :        - image : tableau 2D de hauteur * largeur entiers
                - hauteur : entier
                - largeur : entier
                - seuil : entier
Local :         - i : entier
                - j : entier
Sortie :        histogramme : tableau 1D de 256 entiers
Début
|       Soit histogramme = tableau 1D de 256 entiers
|       // initialisation de l'histogramme
|       Pour i de 0 à 255 avec un pas de +1 Faire
|       |       histogramme[i] = 0
|       Fin Pour
|       // calcul de l'histogramme
|       Pour i de 0 à hauteur − 1 avec un pas de +1 Faire
|       |       Pour j de 0 à largeur − 1 avec un pas de +1 Faire
|       |       |       histogramme[image[i,j]] = histogramme[image[i,j]] + 1
|       |       Fin Pour
|       Fin Pour
|       Retourner histogramme
Fin
```