LIFAP1 – TD 13 : Les structures

Objectifs : Manipulation des structures.
Application aux tableaux

Définition de la structure	Structure Nom_Structure champ1 : type champ2 : type … Fin structure
Déclaration d'une variable de type structure	Ma_structure : Nom_Structure
Accès à un champ de la structure	Ma_structure.champ1

On souhaite développer un logiciel permettant d'effectuer des traitements simples sur les images : extraction de valeurs caractéristiques, seuillage, miroir horizontal, addition.


image initiale

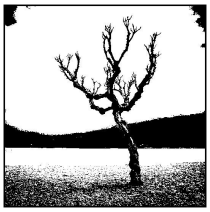
image seuillée


miroir image


initiale + miroir

Une image est constituée d'une grille (tableau) de pixels ayant une intensité lumineuse (niveau de gris). Le niveau de gris d'un pixel est une valeur entière comprise entre 0 (noir) et 255 (blanc).

1. Afin de pourvoir gérer des images de tailles variables, on définira une taille maximale MAX_X et MAX_Y pour les tableaux utilisés. Définir en langage C / C++ deux constantes ayant pour valeur 256.

```
const int MAX_X = 256;
const int MAX_Y = 256;
```

2. Écrire en langage algorithmique puis en langage C / C++ la structure de données permettant de stocker une image. Cette image sera identifiée par sa taille effective en nombre de pixels (taille_x et taille_y) et un tableau contenant les intensités lumineuses de chacun des pixels.

```
struct image
{
    int taille_x;
    int taille_y;
    int tab_img[MAX_X][MAX_Y];
};
```

```
Structure image
    taille_x, taille_y : entiers
    tab_img[MAX_X][MAX_Y] : entiers
Fin structure image
```

3. Écrire en C / C++ une fonction qui permet de remplir la structure image.

```
struct image saisir_image()
{
   int i,j;
   struct image im;
   cout<<"quelle est la taille de l'image ?"<<endl;
   cin>>im.taille_x>>im.taille_y;
   for (i=0;i<im.taille_x;i++)
    {
```

```
            for (j=0;j<im.taille_y;j++)
            {
                cout<<"niveau de gris du pixel"<<i<<","<<j<<endl;
                cin>>im.tab_img[i][j];
            }
        }
        return im;
}
```

4. Écrire en langage C / C++ un sous-programme qui permet d'afficher les valeurs des différents pixels de l'image.

```
void afficher_image(struct image im)
{
   int i,j;

   for (i=0;i<im.taille_x;i++)
    {
            for (j=0;j<im.taille_y;j++)
            {
                cout<<im.tab_img[i][j];
            }
            cout<<endl;
    }
}
```

5. Écrire en langage C / C++ un sous-programme permettant d'extraire, en un seul parcours de l'image :
   - le niveau de gris minimum de l'image,
   - le niveau de gris maximum de l'image,
   - le niveau de gris moyen de l'image.

```
void extraire (struct image im, int &min, int &max, int &moy)
{
    int i,j,som ;
    min=255 ;
    max=0 ;
    som=0 ;
    for (i=0 ;i<im.taille_x ;i++)
    {
            for (j=0 ;j<im.taille_y ;j++)
            {
                som+=im.tab_img[i][j] ;
                if (im.tab_img[i][j] > max)
                   max= im.tab_img[i][j] ;
                if (im.tab_img[i][j] < min)
                   min = im.tab_img[i][j] ;
            }
    }
    moy= som / (im.taille_x * im.taille_y) ;
}
```

6. L'opération dite de « seuillage simple » consiste à mettre à zéro tous les pixels ayant un niveau de gris inférieur à une certaine valeur (appelée seuil) et à mettre à la valeur maximale les pixels ayant une valeur supérieure à ce seuil. Ainsi le résultat du seuillage est une image binaire ne contenant que des pixels noirs et blancs. Écrire en langage C / C++ un sous-programme permettant d'effectuer le seuillage d'une image. Le seuil choisi par l'utilisateur sera passé en paramètre.

```
void seuillage (struct image &im, int seuil)
{
    int i,j;
    for (i=0;i<im.taille_x;i++)
    {
          for (j=0;j<im.taille_y;j++)
          {
              if (im.tab_img[i][j] > seuil)
                im.tab_img[i][j]=255;
             else
                im.tab_img[i][j]=0;
          }
    }
}
```