SENEJKO Morgane TD F

RAPPORT

Etant seule pour réaliser ce projet, j'ai beaucoup travaillé en dehors des séances de TD, environ 3 à 5h à chaque fois. Ce qui fait un total de 35h de travail.
Voici la description chronologique de mon avancée au cours des 5 TD :

TD1 :
-compréhension du sujet
-classe Pixel
-classe MyImage (attributs, constructeur, propriétés)
-méthode de conversion Endian_to_Int

Entre TD1 et TD2 :
-méthode de conversion Int_to_Endian
-méthode From_Image_To_File

TD2 :
-passage d'une photo couleur à une photo en nuance de gris
-passage d'une photo couleur à une photo en noir et blanc
-rotations (90, 180, 270)
-effet miroir

Entre TD2 et TD3 :
-agrandir une image

TD3 :
-compréhension d'une matrice de convolution
-matrice de convolution filtre contour

Entre TD3 et TD4 :
-filtres renforcement des bords, flou et repoussage
-rétrécir une image

TD4 :
-finalisation filtre flou
-compréhension d'une fractale
-début du code d'une fractale

Entre TD4 et TD5 :
-création du 2ème constructeur de la classe MyImage
-finalisation fractale
-cacher une image dans une image (1 pixel sur 2)
-quelques tests unitaires

TD5 :
-création du menu
-cacher une image dans une image avec la stéganographie
-retrouver une image dans une image
-ma création : inversion des couleurs

Afin de réaliser ce projet, j'ai construit deux classes : une classe Pixel et une classe MyImage.

1) La classe Pixel

Attributs :

Un pixel possède 3 attributs. Ce sont 3 bytes dont la valeur sera comprise entre 0 et 255 : red, green et blue.

Méthodes :

Les méthodes crées dans la classe Pixel servent à simplifier les méthodes de la classe MyImage. En effet, ces méthodes sont utilisées dans celles de la classe MyImage lorsqu'il faut modifier les pixels de la matrice de pixels.

- Public void Transforme_En_Gris()

Pour un pixel donné, cette méthode fait la somme des valeurs de ses 3 attributs et en fait la moyenne. Cette moyenne constitue alors la valeur de chacun des 3 attributs.

- Public void Transforme_En_Noir_Ou_Blanc()

Pour un pixel donné, cette méthode fait la somme des valeurs de ses 3 attributs. Si cette somme dépasse 382 le pixel est transformé en blanc sinon il est transformé en noir.

- Public void Clone_Pixel()

Cette fonction clone le pixel.

- Public void Pixel_inverse()

Chaque pixel est remplacé par son inverse au niveau des couleurs.

- Méthodes de conversion

Public int [] ConvertToBase2(byte couleur)
Public int [,] ConvertPixelToBase2()
Ces méthodes servent à convertir un pixel en une matrice d'entier. Chaque ligne correspond à la conversion d'un attribut en base 2.
Public byte ConvertToByte(int [] couleur)
Public Pixel ConvertBase2ToPixel(int [,] pixelbase2)
Ces méthodes servent à convertir une matrice d'entie, correspondant au pixel écrit en base 2, en un Pixel dont les attributs sont des bytes.

- Public Pixel Mixer2Pixels(Pixel p2)

Cette méthode sera utilisée pour cacher une image dans une autre image. En effet, chaque pixel est transformé en utilisant les pixels de l'image original et les pixels p2 de l'image à cacher. On réalise un mixage des 2 pixels. Pour cela on utilise les méthodes de conversion décrite précédemment.

- Méthodes pour retrouver une image dans une image

Public byte RetrouverCouleurReference(byte couleur)
Public Pixel RetrouverPixelReference()
Ces méthodes servent à retrouver à peu près le pixel référent lorsqu'on lui donne un pixel mixé.
Public byte RetrouverCouleurImageCachee (byte couleur)
Public Pixel RetrouverPixelImageCachee()
Ces méthodes servent à retrouver à peu près le 2ème pixel utilisé lorsqu'on lui donne un pixel mixé. Il s'agit du pixel de l'image qu'on a caché.
Je n'ai pas réussi à faire fonctionner cette méthode.

2) La classe MyImage

Attributs :

Une image possède 9 attributs : son type (string typeImage), la taille du fichier (int tailleFichier), la taille offset (int tailleOffset), la taille du header (int tailleHeader), la taille de l'image (int tailleImage), sa largeur (int largeur), sa hauteur (int hauteur), le nombre de bits par couleur (int nbBitsParCouleur), une matrice de Pixels qui représentent l'image (Pixel [,] image).

Constructeurs :

- Public MyImage (string myfile)

Ce 1er constructeur permet de créer une instance de la classe MyImage à partir d'un fichier donné en paramètre. Ce fichier va être lu afin qu'on puisse en créer un tableau. Tous les attributs de l'image vont être initialisés grâce aux données trouvées dans ce tableau.

- Public MyImage(int largeur, int hauteur, int nbBitsParCouleur, byte red, byte green, byte blue)

Ce 2ème constructeur permet de créer une image à partir de rien. En effet, les attributs de l'image lui sont donnés en paramètre. Ce constructeur nous servira pour la création d'une fractale.

Méthodes :
Voir commentaires dans le projet