TP 1: Manipulation de fichiers

Système d'Exploitation

Image en niveau de gris

Le format pgm (Portable Gray Map) permet de représenter des images en niveaux de gris en utilisant des caractères pour coder les pixels.

Vous pouvez donc utiliser un éditeur d'images tel que `Gimp` et exporter une image au format pgm.

En cochant l'option RAW, le fichier résultant sera un fichier avec un contenu consultable avec un éditeur de texte. Ce 'texte' comporte une entête et le codage de votre image.

Structure d'un fichier pgm (ascii)

Un fichier pgm est composé de deux parties: une entête et une section de données.

L'entête comporte au moins trois lignes:

1. un "magic number" ici 'P5' pour indiquer le format pgm en mode RAW
2. une information sur la taille de l'image: sa largeur, au moins un espace, et sa longueur
3. les niveaux de gris, 255 par exemple.

L'entête peut comporter des lignes de commentaires commençant par un caractère '#'.

Il est important de noter que toutes les valeurs numériques (largeur, longueur et nombre de niveau de gris) sont données en décimal ASCII.

Par exemple, si notre image est de taille 128x128 alors nous trouverons la chaine de caracatères en ascii '128 128' à la ligne 2.

Pour la partie données, chaque point de l'image (pixel) est représenté par un caractère (ou un octet) correspondant à son niveau de gris (0:noir jusqu'à 255:blanc)

Voici un exemple de début de fichier pgm :

```
P5
# CREATOR: GIMP PNM Filter Version 1.1
256 256
255
+Wae?<G?QCRT.)1%-00=?=>:9DG:.,+(*/.06@KRSRHKHHK@7<730...
```

Etape 1: copie d'image

L'objectif de cette étape est de compléter le programme `tp1_copie.c` pour lire le contenu d'un fichier pgm et créer une image similaire (une copie) en ignorant les commentaires.

Par exemple la commande:

```
tp1_copie ./img/cat.pgm copie.pgm
```

Devra produire une image pgm nommée `copie.pgm` qui sera similaire à `./img/cat.pgm` et sans les commentaires.

Exercice 1:

- Consulter le code de la fonction `lecture_MagicNumber` dans le fichier `tp1_copie.c`
- Consulter le code de la fonction `ecriture_MagicNumber` dans le fichier `tp1_copie.c`
- Consulter le code des fonctions `ecriture_entier`, `ecriture_NL`, `ecriture_SPACE` dans le fichier `tp1_copie.c`

Exercice 2:

Écrire la fonction `int avance_jusqua_nouvelle_ligne(int descripteurLecture);` qui avance la tête de lecture jusqu'à la lecture du caractère `\n`

Exercice 3:

Écrire le code de la fonction `int test_si_ligne_commentaire(int descripteurLecture);` qui va tester si la ligne courante est une ligne de commentaire.

Pour rappel, une ligne de commentaire commence avec le caractère `#`

Vous pouvez considérer que la tête de lecture du fichier est déjà positionnée au début de la ligne. Attention, une fois le test terminé, vous devez remettre à la tête de lecture au début de la ligne.

Exercice 4:

Ecrire le code des fonctions suivantes:

- `ecriture_largeur_longueur`
- `ecriture_NiveauGris`

Qui vont écrire respectivement les lignes de largeur et longueur ainsi que le niveau de gris dans le fichier.

Les valeurs se trouvent dans les variables globales `largeur` , `longeur` et `niveau_gris` qui sont de type `int`. Vous devrez donc convertir ces valeurs en chaînes de caractères représentant leurs écritures décimales en ascii. Pour cela, vous pouvez utiliser la fonction `sprtinf(...)` après consultation de sa documentation.

Exercice 5:

Écrire le code la fonction `traitement_Contenu(int descripteurLecture, int descripteurSortie)` qui va copier tous les caractères de `descripteurLecture` vers `descripteurSortie`.

Synthèse :

Veuillez consulter maintenant la fonction `main` pour comprendre le traitement réaliser.

Vous pouvez ensuite compiler et tester votre programme de copie d'images.

Etape 2: réduction de la taille de moitié

Maintenant que vous avez terminé l'étape 1 avec la copie de l'image, nous allons passer aux choses sérieuses!

Nous vous demandons de réaliser un programme qui va réduire la taille d'une image de moitié en largeur et en longueur.

Un pixel dans la nouvelle image (taille réduire) va représenter 4 pixels de l'ancienne image (taille normale) Il aura pour valeur de niveau de gris la moyenne de ces 4 pixels.

Voici la relation entre un pixel de la nouvelle image `npixel` et les pixels de l'ancienne image `apixel`:

```
npixel[i,j] = (apixel[2*i,2*j] + apixel[2*i,2*j +1] + apixel[2*i
+1,2*j ] + apixel[2*i +1,2*j +1 ])/4
```

Pour simplifier le problème, vous pouvez considérer que la taille des images (largeur et longueur) sont des multiples de 2 (c'est le cas de l'image `cat.pgm`)

Exercice 7:

En partant du contenu de `tp1_copie.c`, créer un nouveau programme `tp1_shrink.c` qui va réduire de moitié l'image donnée en paramètre.

Recommandation: essayez d'utiliser `lseek` pour chercher les valeurs dans le fichier source sans avoir à le copier entièrement en mémoire.