1 Lecture et écriture dans des fichiers

En langage C, la lecture et l'écriture dans un fichier font appel à deux familles de fonctions

- `fprintf` et `fscanf`
- `fwrite` et `fread`

La première famille correspond à de l'écriture/lecture dite *formatée* alors que la deuxième correspond à de l'écriture/lecture dite *binaire.*

Pour rappel, voici les syntaxes des deux fonctions en écriture :

```
fprintf(flot, "chaine", expression-1,expression-2,...)
fwrite(adresse du tableau de données, taille d'une donnée,
        nombre de données à écrire, flot)
```

Le but de cet exercice est de créer un programme qui va inscrire le nombre $n = 201$ dans un fichier. Vous verrez alors que chaque famille de fonctions donne un résultat bien différent en mémoire.

Question 1 : Écrivez n sous forme binaire puis hexadécimale (à la main !).

Question 2 : Trouvez le type C le plus adapté pour stocker n. Utilisez ensuite la fonction `fprintf` pour écrire ce dernier dans un fichier que vous appellerez `n_format`.

Question 3 : Utilisez la fonction `fwrite` pour écrire n dans un fichier que vous appellerez `n_binaire`.

Question 4 : Comparez la taille des deux fichiers et affichez leur contenu avec la commande

```
$ hexdump -C fichier
```

Que remarquez vous ?

2 Structures pour les images

Un fichier PPM (portable pixmap) est un format très simple pour stocker une image de petite taille.

Un fichier .PPM à la forme suivante :

```
P3
2 2
255
0 0 0 255 255 255
255 255 255 0 0 0
```

- `P3` signifie qu'il s'agit d'un fichier PPM en couleur (RGB) et que les valeurs de chaque pixel seront écrites sous forme de chaînes de caractères dans un fichier non binaire. En d'autres termes, vous pouvez ouvrir un fichier PPM de type `P3` avec un éditeur de texte et vous verrez directement ce qu'il contient.
- `2 2` est la taille de l'image, d'abord 2 colonnes, puis 2 lignes.
- `255` est la valeur maximale que peut prendre chaque composante couleur pour un pixel.
- Enfin, le reste du fichier contient la description de chacun des pixels, une ligne de texte correspond à une ligne de l'image.

Remarquez que les caractères d'échappement pour les fichiers PPM sont simplement des espaces et des sauts de ligne.

Question 1 : Créez un programme dans lequel vous déclarerez une structure `pixel` qui contiendra un pixel (i.e. 3 champs, un par couleur).

Question 2 : Dans ce même programme, créez 2 fonctions `noir()` et `blanc()` qui renvoient respectivement un pixel noir et un pixel blanc.

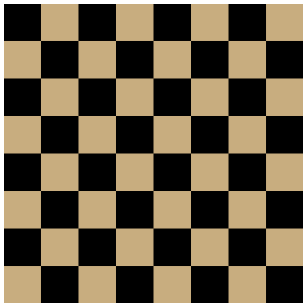
Question 3 : Dans votre `main`, créez un tableau à deux dimensions de taille 8×8 de pixels. Vous initialiserez alors ce tableau pour qu'il contienne un damier, c'est à dire des cases successivement noires et blanches.

Astuce : Pour obtenir un beau damier d'échecs, remplacer la couleur blanche par le beige ayant pour code RGB : `200 173 127`.

3 Applications

Question 1 : Maintenant que vous savez lire et écrire dans un fichier (en mode binaire ou formaté), vous pouvez reprendre l'exercice 3 du TP 2 et déchiffrer le fichier présent sur mon site web.

Question 2 : Modifiez le programme de l'exercice précédent pour qu'il crée un fichier image `damier.ppm`, vous pourrez alors l'ouvrir et découvrir quelque chose comme cela



Astuce : Si vous souhaitez visualiser votre fichier .PPM, ouvrez le avec un programme qui permet de zoomer très fortement sans avoir d'effet de "flou", par exemple `GIMP`.