Université de Pau et des Pays de l'Adour **Année 2017-2018**
Département de Mathématiques

L2-Info–Calcul scientifique
TD n^o 12

L'objectif de ce TD est de développer des outils pour le traitement d'images au format .ppm. (On pourra tester avec les différentes images fournies sur le site du cours.)
Ce format permet l'écriture en ascii du codage des couleurs en RGB. Pour chaque pixel, le code rgb se présente sous la forme d'un triplet de nombres entiers compris entre 0 et 255. Le code 255 255 255 correspond au blanc, 0 0 0 au noir, 255 0 0 au rouge etc. Les fichiers ppm que vous aurez à traiter se présentent sous la forme suivante:

P3	signifie que les couleurs sont en ASCII, suivant le principe rgb
# toto.ppm	ligne de commentaire
3 2	taille en pixels, dans chaque direction
255	valeur max utilisée pour le codage rgb
255 255 0	couleur pixel 0
0 0 255	couleur pixel 1
0 0 0	couleur pixel 2
255 255 255	couleur pixel 3
0 255 255	couleur pixel 4
255 0 0 0	couleur pixel 5

Vous avez trois fonctions à coder dans la classe PPMFile à récupérer sur le site du cours:

1. `void writeNegFile(const std::string& outfilename) const;`
   Passer au négatif d'une image.
   Vous devez parcourir chaque pixel et passer de sa couleur à son négatif. Si $r\ g\ b$ est la couleur d'un pixel alors $\_vmax - r\ \_vmax - g\ \_vmax - b$ est la couleurs du négatif. Le résultat devra être écrit, au fur et à mesure, suivant le format ci-dessus, dans un fichier ppm dont le nom est passé en paramètre.

2. `void writeContourFile(const std::string& outfilename, int seuil) const;`
   Repérer les contours dans une image.
   Tout d'abord il faut déclarer et dimensionner un tableau de couleurs de la taille de l'image à traiter. On pourra utiliser le même type que le tableau _pixels. On initialise tous les pixels à la couleur blanche. On parcourt ensuite les pixels dans chaque direction de 1 à _n-1 en y et de 1 à _m-1 en x et on estime les variations de la couleurs. Pour cela, on calcule les différences en valeurs absolues de la couleur du pixel par rapport à ses quatre voisines. (différence des r + différence des g + différences des b). Si la somme des différences en valeur absolue, est supérieure au seuil passé en paramètre, alors le pixel prend la couleur 0 0 0. Il faut alors produire le fichier image ppm correspondant. Le nom du fichier de sortie est passé en paramètre.

3. `void writeExtractColor(int r, int g, int b, int seuil, const std::string& outfilename);`
   Passer l'image en niveau de gris en ne conservant qu'une couleur.
   On parcourt tous les pixels de l'image d'origine et au fur et à mesure, on écrit dans le fichier de sortie, les couleurs de la nouvelle image. Si la différence entre la couleur du pixel et celle passée en paramètre est inférieure au seuil alors, on écrit la couleur du pixel. Sinon, on passe le pixel en niveau de gris. Pour cela, on affecte à r, g, b la même valeur égale à la moyenne des trois, à savoir, ((r+g+b)/3). La nouvelle image obtenue devra être écrite, toujours au format ppm, dans un fichier dont le nom est passé en paramètre.