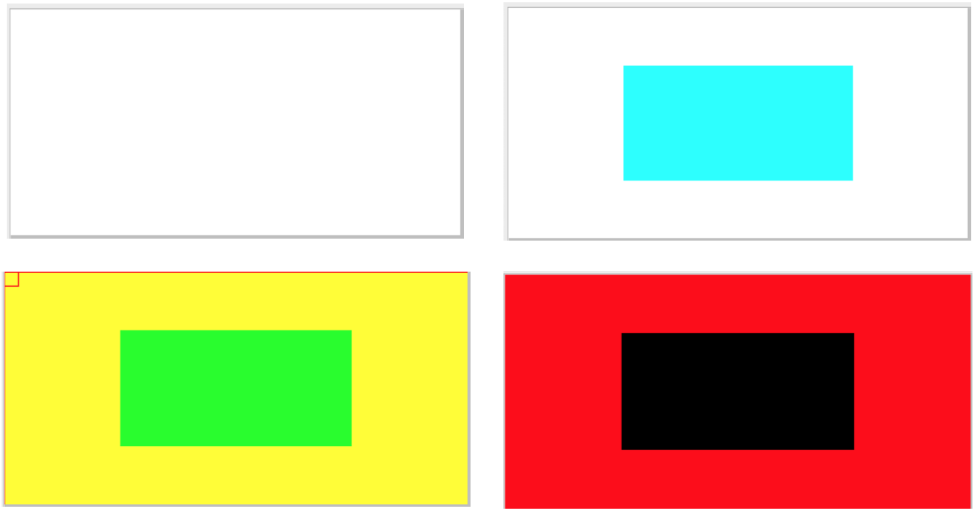
test-bcl <rows> <cols>

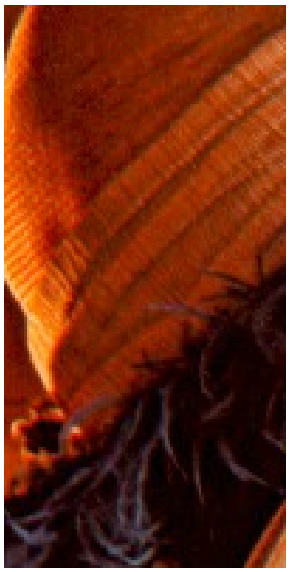
— compiler (avec le `Makefile` fourni) et exécuter le programme `test-bcl.c`
— le programme produit des images (`{a,b,c,d}.ppm`) dont les dimensions sont spécifiés par les paramètres `rows` et `cols`
— étudier et comprendre ce que fait le programme `test-bcl.c`



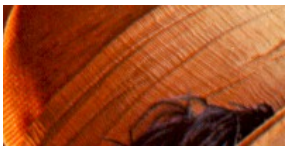
`./test-bcl 100 200`

extract-subimage <i> <j> <rows> <cols> <ims> <imd>

Écrire le programme `extract-subimage.c`
— le programme permet de charger une image donnée en paramètre `ims`
— le programme permet d'extraire une sous image de l'image source `ims` à partir du pixel positionné en (`i`, `j`) de taille `rows`×`cols`. Le résultat est sauvegardé dans l'image nommée `imd`
— visualiser les images produites via `pvisu`



```
./extract-subimage
100 100 200 100
../data/lena-color.ppm
a.ppm
```



```
./extract-subimage
100 100 100 200
../data/lena-color.ppm
b.ppm
```

extract-channel <num> <ims> <imd>

Écrire le programme `extract-channel.c`
— le programme permet de charger une image donnée en paramètre `ims`
— le programme permet d'extraire le canal de numéro `num` de l'image source `ims`. Le résultat est sauvegardé dans l'image en niveaux de gris nommée `imd`
— visualiser les images produites via `pvisu`

`./extract-channel 0 ../data/lena-color.ppm r.ppm`

`./extract-channel 1 ../data/lena-color.ppm g.ppm`

`./extract-channel 2 ../data/lena-color.ppm b.ppm`

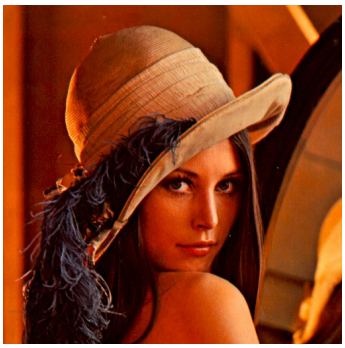
gray2color <ims0> <ims1> <ims2> <imd>

Écrire le programme `gray2color.c`
— le programme permet de composer une image couleur (sauvegardée dans `imd`) à partir de trois images en niveaux de gris (`ims0 ims1 ims2`)
— visualiser les images produites via `pvisu`

`r.ppm`

`g.ppm`

`b.ppm`



`./gray2color r.ppm g.ppm b.ppm a.ppm`

color2mean <ims> <imd>

Écrire le programme `color2mean.c`

— le programme permet de convertir une image couleur (`ims`) en une image d'intensité moyenne (`imd`) où chaque pixel de l'image résultat est la moyenne des trois canaux de l'image couleur (attention aux problèmes de type)

— visualiser les images produites via `pvisu`

```
./color2mean
../datat/lena-color.ppm gray.ppm
```

normalize <min> <max> <ims> <imd>

Écrire le programme `normalize.c`

— le programme permet de normaliser les valeurs des pixels de l'image source `ims` dans la nouvelle image `imd` en utilisant le nouvel intervalle $[\min, \max]$ et en appliquant la fonction de normalisation suivante

$$I'(i,j) = \frac{\max - \min}{\mathrm{Max}(I) - \mathrm{Min}(I)} \times I(i,j) + \frac{\min \times \mathrm{Max}(I) - \max \times \mathrm{Min}(I)}{\mathrm{Max}(I) - \mathrm{Min}(I)} \tag{1}$$

où I' est l'image résultat, $\mathrm{Min}(I)$ et $\mathrm{Max}(I)$ correspondent respectivement aux valeurs minimale et maximale de l'image source I

— pour une image couleur, la normalisation se fait de manière marginale (canal par canal)

— visualiser les images produites via `pvisu` et observer l'influence de la transformation sur les histogrammes des images via `pvisu`

— **attention, les calculs doivent se faire en nombres flottants**

```
./normalize 100 125
gray.ppm a.ppm
```

```
./normalize 100 125
../data/lena-color.ppm
a.ppm
```