TP1
Image File Manipulation

The objective of this practical work is to implement basic image manipulation tools, e.g. read, write and convert. To this aim, and without loss of generality, the simple image file formats PBM (Portable Map) will be considered.

"Portable Map" image file formats

The netpbm file formats PBM, PGM and PPM are respectively: portable bitmap, portable grayscalemap and portable pixmap (also called PNM for portable anymap). They were originally designed in the early 80's to ease image exchange between platforms. They offer a simple and pedagogical solution to develop image manipulation tools. In these formats, an image is a matrix of pixels where values represent the illumination in each pixel: white and black (PBM), grayscale (PGM) or 3 values RGB (PPM).

Definition

The PNM file content is as follows:

1. A 'magic number' that identifies the file type. A pbm image's magic number is the two characters 'P1' (ASCII) or 'P4' (binary).
2. Whitespace (blanks, TABs, CRs, LFs).
3. The width and height (separated with a whitespace) in pixels of the image, formatted as ASCII characters in decimal.
4. Only for PGM and PPM: The maximum intensity value between 0 and 255, again in ASCII decimal, followed by a whitespace.
5. Width $\times$ Height numbers. Those numbers are either decimal values coded in ASCII et separated by a whitespace for the formats P1, P2, P3, or directly binary values (usually 1 byte) in the case of P4, P5, P6. In the latter case, there is no whitespace between values.

Remarks:

- Lines beginning with "#" are ignored.
- In principle lines should not be longer than 70 characters.

sergi.pujades-rocamora@inria.fr, julien.pansiot@inria.fr

Examples

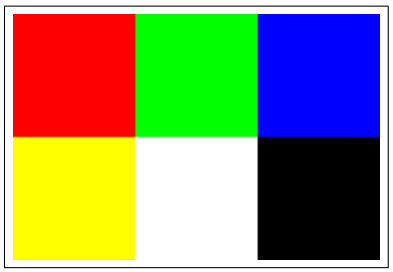

```
P1
# feep.pbm
24 7
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0
0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0
0 1 1 1 0 0 0 1 1 1 0 0 0 1 1 1 0 0 0 1 1 1 1 0
0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0
0 1 0 0 0 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

PBM file of a 24×7 image for which values are coded in ASCII decimal


```
P2
# feep.pgm
24 7
15
0 0 0 0 0 0 0 0 0 0 0 0 0 0  0  0  0  0 0 0  0  0  0  0
0 3 3 3 3 0 0 7 7 7 7 0 0 11 11 11 11 0 0 15 15 15 15 0
0 3 0 0 0 0 0 7 0 0 0 0 0 11 0  0  0  0 0 15 0  0  15 0
0 3 3 3 0 0 0 7 7 7 0 0 0 11 11 11 0  0 0 15 15 15 15 0
0 3 0 0 0 0 0 7 0 0 0 0 0 11 0  0  0  0 0 1  0  0  0  0
0 3 0 0 0 0 0 7 7 7 7 0 0 11 11 11 11 0 0 15 0  0  0  0
0 0 0 0 0 0 0 0 0 0 0 0 0 0  0  0  0  0 0 0  0  0  0  0
```

PGM file of a 24×7 image . Intensity values are coded in ASCII decimal with 15 levels between black and white.

sergi.pujades-rocamora@inria.fr, julien.pansiot@inria.fr



```
P3
# 6 colors RGB
3 2
255
255   0   0     0 255   0     0   0 255
255 255   0   255 255 255     0   0   0
```

PPM file of a 3×2 image. Intensity values are coded in ASCII decimal

Exercise 1

Download the archive that contains the code of TP1.

1. Compile the programs using the Makefile and test *pgmtopgm* on mandrill.pgm by running

   ```
   % make
   % ./pgmtopgm images/mandrill.pgm results/mandril1.pgm
   ```

   What kind of conversion is performed ? Check the contents of mandrill.pgm and mandrill1.pgm with a text editor.

2. How is the image stored within the program ?

3. What is the purpose of the functions pm_getint and pm_getrawbyte in the file Util.c ?

4. What data types are involved when manipulating intensities ? in the case of ASCII decimals (P1, P2, P3) ? In the case of binary values (P4, P5, P6) ?

5. Which color is associated to the maximal value for each format?

6. Comment on the sizes of files in the two formats.

Exercise 2

The PPM format allows to store images in RGB format. Check the file ppmtorgb.c defining the program *ppmtorgb*. Its goal is to convert a PPM image into three images with respect to the RGB channels.

sergi.pujades-rocamora@inria.fr, julien.pansiot@inria.fr

1. First implement the read and write functions (*read_ppm* and *write_ppm*) in `Utils/Util.c`.

2. Then each channel should be stored in a PPM image with zeros on the other channels. Implement the body of the function *split_channels* in TP1/TP1_funcs.c to complete the program. Test it by running

```
% ./ppmtorgb images/mandrill.ppm results/mandrillR.ppm
results/mandrillG.ppm results/mandrillB.ppm
```

3. Check the file `ppmtopgm.c` defining the program *ppmtopgm*. Its goal is to convert a PPM image into a grayscale image. Implement the body of the function *rgb_to_gray* in *TP1 TP1_funcs.c* . Test it by running

```
% ./ppmtopgm images/mandrill.ppm results/mandrill_gray.pgm
```

4. Use the variable "brightness" in the conversion process to obtain a brighter or darker image. Test it by running

```
% ./ppmtopgm images/mandrill.ppm
results/mandrill_gray_bright.pgm 1.2

% ./ppmtopgm images/mandrill.ppm
results/mandrill_gray_dark.pgm 0.8
```

Exercise 3

An extension of the netpbm formats to transparent images exist, i.e. images for which there is an additional alpha channel encoding transparency. It is called PAM (extension .pam for files) for Portable Arbitrary Map.

1. Look over the web and give the main principles of this extension, e.g. file header and pixel format.

2. **Bonus 1:** write a program `ppmtopam` that converts a PPM image into a PAM image with the alpha value (transparency) set to 255 for all pixels.

```
% ./ppmtopam images/mandrill.ppm results/mandrill.pam
```

3. **Bonus 2:** improve the `ppmtopam` program so that it also takes a PGM image and creates a PAM image with the colors of the PPM image and the alpha value (transparency) from the PGM image.

```
% ./ppmtopam images/mandrill.ppm images/mask.pgm
results/mandrill_masked.pam
```

The Portable Network Graphic (PNG) format, which supports compression, has become the de facto format for images with transparency. You can convert the generated PAM file into PNG with open software such as ffmpeg (`https://ffmpeg.org/`)

```
% ffmpeg -i results/mandrill_masked.pam results/mandrill_masked.png
```

or ImageMagick (`https://imagemagick.org/`)

```
% convert results/mandrill_masked.pam results/mandrill_masked.png
```

This way you can share and use your generated images in all standard pipelines.