The PPM format

The PPM format (defined on http://netpbm.sourceforge.net/doc/ppm.html) is similar to the already seen PGM format and allows you to store color images. As for the PGM we will use the following **simplified definition**. The PPM file is defined as:
1) A "magic number" to identify the type of file. The magic number of a PPM image is the sequence of two characters "P6".
2) The character "\n", that is the Line Feed (LF), that is the character 10 (0x0A), that is a "carriage return". Beware that this cannot be the pair "\r\n"
3) An optional comment identified by a "#" character followed by any sequence of characters ending with the "\n" character. During read it is possible to verify if the "#" character is present and if not, this field is not present.
4) The width of the image (W), formatted as a sequence of ASCII characters in decimal.
5) The " " character, that is, a space.
6) The height of the image (H), formatted as a sequence of ASCII characters in decimal.
7) The character "\n".
8) The string "255".
9) The character "\n".
10) A sequence of H rows, in order from top to bottom. Each row consists of W triplets of RGB colors, in order from left to right. Each triplet consists of three numbers from 0 to 255, with 0 to indicate the minimum value of a channel and 255 to indicate the maximum. Each channel is represented in binary with a single byte. The order is R, followed by G, then followed by B (we specify this because in some formats, such as BMP, this is not the case).

There is also the version of the PPM known as "plain". As for the PGM, all the values of point 10) (i.e. the pixels of the image) are represented as a sequence of ASCII characters in decimal, followed by space, tab, newline, or more spaces, tabs and newlines mixed. The magic number of this format is "P3".

Here is an example:

```
P3
# Test
5 5
255
255 0 0    255 0 0    255 0 0    255 0 0    255 0 0
192 0 0    192 0 0    192 0 0    192 0 0    192 0 0
128 0 0    128 0 0    128 0 0    128 0 0    128 0 0
64 0 0     64 0 0     64 0 0     64 0 0     64 0 0
0 0 0      0 0 0      0 0 0      0 0 0      0 0 0
```

This is a 5x5 image that changes from red to black with a vertical gradient.

Exercise 1

Write a program that generates a 256×256 color image, in which the first row is made up of 256 values with red set to 0, the second with 256 values with red set to 1, the third with 256 values with red set to 2 and so on. The other channels are all always at 0. Save the image in the simplified PPM and plain PPM formats, described above. Verify that the image is viewable in XnView. The image should appear as a black to red gradient from top to bottom.

Exercise 2

Write a program that accepts the following syntax:

```
flip <filename.ppm>
```

The program opens a PPM image whose name is given as argument on the command line file and creates an "upside down" version, that is, the first line at the top becomes the last at the bottom of the new image, the second becomes the penultimate and so on. Save the image in the PPM and plain PPM formats, described above, naming them `"filename_flip.pgm"` and `"filename_flip_plain.pgm"`. Verify that the images are viewable in XnView.

Exercise 3

Write a program that accepts the following syntax:

```
split <filename.ppm>
```

The program opens a PPM image whose name is given as argument on the command line file and saves three PGM images named `"filename_R.pgm"`, `"filename_G.pgm"` and `"filename_B.pgm"`, containing the separate R, G and B color planes. Verify that the images are viewable in XnView.

Exercise 4

Write a program that accepts the following syntax:

```
combine <filename>
```

The program opens the three files `"filename_R.pgm"`, `"filename_G.pgm"` and `"filename_B.pgm"`, containing three separate R, G and B color planes, and generates the file `"filename_reconstructed.pgm"` composed by combining the three color planes. Verify that the image is viewable in XnView.