Laboratory 1

1.1

Execute code in ‘exercise1\_1.m’

1.2

Execute code in ‘exercise1\_2.m’

Explain the difference between pixel values?

1.3

Execute code in ‘exercise1\_3.m’

1. The histogram does change: the vertical axis range decreases (obviously, as the number of pixels decreases) and it becomes crispier, with a less smooth contour, than in the original image. When returning the reduced image to its original size, it recovers the initial vertical range, but the crispy contour is maintained.
2. The bigger the mask, the blurrier the image. If the mask has a vertical form, the resulting image becomes blurry with a vertical deformation (all forms in the image look like they have been vertically stretched), and the same effect appears if the mask has a horizontal form, but this time, the deformation appears, logically, in the horizontal axis. In order to be convoluted, an image must be in a 2D form, each cell, or pixel, containing a single value (like in the grayscale). Therefore, a RGB image cannot, technically speaking, receive a filter, as it is a concatenation of 3 matrices. However, Matlab simulates the effect by applying the filter to each layer, or matrix, individually. The final effect is the same that appears in a gray filtered image. If we do not normalize the mask, the values of the pixels in the filtered image are the result of the direct sum of other pixels, which may result in many (even all) values being over 255 and, therefore, “burning” the image. That’s why it is vital to normalize the mask, in order to keep the resulting values in the [0:255] range.

1.4

Execute code in ‘exercise1\_4.m’

1.5

Execute code in ‘exercise1\_5.m’

1. The higher the threshold, the more pixels will become a 0, as they will be below the given threshold. Therefore, the image will be generally darker and it will gradually lose detail
2. Multiplying an image by its binarized will “paint” black all pixels in black in the binarized image (as they will be multiplied by 0), and will keep the rest of the pixels as they are in the original (as they will be multiplied by 1).
3. Multiplying an image by it inverted binarized will paint black all pixels in white in the “simply binarized” image (as they will be multiplied by 0), and will keep the rest of the pixels as they are in the original (as they will be multiplied by 1).

1.6

Execute code in ‘exercise1\_6.m’