Image Processing and Artificial Vision
Final exam, June 13th, 2019

Remarks:

a) Write a single Python script named `yourname_yoursurname.py` containing the code developed for the exercise. Important: **Do not use symbols** such as spaces, accents, ñ, ç, et cetera. Upload the script to campusvirtual2.
b) Do not change the name of the files provided
c) Note that **the code should run without errors**. Otherwise, it will not be considered.
d) In case of any ambiguity, decide what is more convenient.

1. Hash functions are used to label binary information of arbitrary size. They are calculated in such a way that the produced hash is always of a fixed size. Specialized hash algorithms for images (ImageHash) are used on images (of arbitrary size). The output of ImageHash is usually a set of 64 binary numbers (alternatively, eight hexadecimal numbers) that identify the image. Interestingly, similar images display similar hashes.
   Nowadays, ImageHash-based algorithms are used in on-line social networks to determine whether an uploaded image fulfills the terms of use of the network or not. The ImageHash of the target picture is compared with the hashes of a large image dataset. If the image falls in a category of banned images, it is removed from the user profile. These algorithms are very successful because ImageHash calculations and comparisons are very fast.

   In this exe3rcise you are going to implement the Difference Imagehash (dHash) of a picture: is calculated as follows:
   i. Convert the image to gray level (this step is not necessary for gray level images).
   ii. Resize the image to 8x9 pixels [`imr`].
   iii. Produce an 8x8 binary image `imb`. For each row `ii`,
   `imb[ii,jj]= imr[ii, jj] < imr[ii, jj+1];` `ii` and `jj` vary from 0 to 7.
   Note that this step can be calculated without loops. Not a big deal though.
   iv. Reshape `imb` as a 1d-array. This is the ImageHash of the image.
   v. In order to compare the hashes of two images, the Hamming distance (d_H) has to be calculated: simply count the number of bits that are different (at the same position). When $d_H < 10$, it is very likely that both images are almost the same.

   - Use the `skimage.data.astronaut` image. Calculate the Hamming distance between this image and its corresponding red, green and blue channels.
   - Calculate the Hamming distance between the astronaut and the Chelsea image `skimage.data.chelsea`.

2. The Lucy-Richadson (LR) algorism is an iterative method designed to restore defocused images affected by noise. A defocused image *d*, recorded by an optical system with point spread function (PSF) *h* and affected by noise *n*, is described by $d = i * h + n$, where *i* is the ideal image to be recovered. The image after $k+1$ iterations, i_{k+1} is described by

$$i_{k+1} = i_k \left| \left[\frac{d}{i_k * h} \right] * h \right| \qquad \text{with } i_0 = d$$

where symbols * and | | stand for convolution and modulus, respectively.
The image provided (`ecolidn.jpg`), show a group of E. coli bacteria. The image is defocused and affected by additive noise. The PSF can be modeled by a circle with a diameter of 37 pixels.

a) Implement the LR algorithm. Display the recovered image after 50 iterations: i_{50}.
b) Calculate the entropy for d and i_{50}; use an appropriate neighborhood.
   Display the entropy using the rainbow `colormap`. Add a `colorbar`. Provide some insight regarding the entropy. Include a comment in your code.

Show the four images in a single 2x2 figure.