Assignment 0 Questions

Instructions:

- Assignments are to be completed **individually**
- Ensure you have the required software installed (see the guide on LEARN)
- Read through the Python tutorial on LEARN
- This assignment is composed of 2 questions (and their sub-parts) make sure to answer them all
- Use the image files included in this zip file to run your experiments
- In your Jupyter notebook, use 'Markdown' blocks to respond to written/comprehension portions of the questions and 'Code' blocks to write and run your code (Note that Jupyter notebooks have the option to toggle between code blocks and 'markdown' blocks. This is done via a dropdown in the top toolbar next to the 'Run' button).
- Have at least 1 block per subquestion, and it's okay(and encouraged when it improves neatness) to have several alternating blocks per subquestion that illustrate your point better.
- Include all your code and code outputs in the final submission documents (i.e. save the notebook after you've finished running and printing everything)
- Make sure your code is well commented and uses readable variable names (e.g. image_mean and identity_matrix are much better variable names than aaa and $5\pi\S$).
- Feel free to include images or equations (Jupyter's markdown blocks accept Latex formatting when put in between $ $)
- Your final submission should contain two files in a zip file named YOUR_NAME_Assignment_0.zip:
    1. File1: Your_Name_Assignment_0.ipynb
    2. File2: Your_Name_Assignment_0.pdf (which for Assignment 0 is just File1 exported as a PDF)
- Note: If it helps you to debug your code in another IDE do so. For example I use VSCODE and/or PYCHARM, both are open source and free to use. Once you are done, copy your codes to a Jupyter notebook and make sure to comment and explain what you did before submitting.

Questions

Q1: We wish to set all pixels that have a value of 10 or less to 0, to remove camera sensor noise. However, our code is slow when run on a database with 1000 grayscale images.

Image: grizzlypeakg.png (gray image)

```
from skimage import io

im_gray = io.imread('grizzlypeakg.png')
(m1,n1) = im_gray.shape
for i in range(m1):
    for j in range(n1):
        if im_gray[i,j] <= 10 :
            im_gray[i,j] = 0
```

Q1.1: How could we speed it up? Please include your code.

Q1.2: What factor speedup would we receive over 1000 images? Please measure it and include your code.

Ignore file loading; assume all images are equal resolution; don't assume that the time taken for one image $\times 1000$ will equal 1000 image computations, as single short tasks on multitasking computers often take variable time.

Note: We found that running the slow code on 1000 images might take 45 minutes or more! You can compute the factor on just 10 images. For the sped-up version, you might need to compute the time on more images to receive a more reliable estimate of the average time. Be sure to compensate for the number of images computed against in your speedup factor.

Q1.3: Next, we wish to operate on color images. How does your speeded-up version from Q1.2 change for color images? Please implement and measure it, report the speed factor change, and include your code.

Note: We will change the value in each color channel; we do not wish to convert the image to grayscale.

Image: grizzlypeak.jpg (colored image)

Q2: We wish to reduce the brightness of an image by editing the values in its matrix. But, when trying to visualize the result, we see some "errors".

Image: gigi.jpg

```
from skimage import io
import matplotlib.pyplot as plt
import numpy as np

I =  io.imread('gigi.jpg').astype(np.float32)
I = I - 50
plt.imshow( I )
plt.show()
```

Q2.1: What is incorrect with this approach? How can it be fixed while maintaining the same intended brightness reduction? Please include your code and result image.