Problem Set 3: Image Manipulation with Python, OpenCV, and Numpy
Computational Perception and Artificial Intelligence

Description:

We have learned how to use Python and Open Computer Vision to load, manipulate, and display images. This problem set has you practice manipulating images with Python in order to gain facility with the API and arrays in Numpy. You will submit your work in a zipped file called `ps03.zip` emailed to Mr. Michaud by the due date.

Setup

A. Download and unzip the following folder into your Perception Lastname Directory:

`http://nebomusic.net/perception/ps03.zip`

B. Open the `ps03.py` file with IDLE and place your name in the comments as indicated in the code.

C. The `ps03.py` file has the stubs for the functions you need to define. Again, take note of the comment style and content of comments. The 'pass' call under the function stubs is a placeholder for Python. When coding your functions, delete the 'pass' command.

Requirements:

1. Find four images with dimensions of 512 x 512 pixels. (They can be .png or .jpg). Save these into your 'input' folder.

2. In the `ps03.py` python module, read the data from your pictures according to these requirements:

Name of array in Python	Source	Color
img	lisa.png	Color
img1	Your Picture 1	Color
img2	Your Picture 2	Color
img3	Your Picture 3	Color
img4	Your Picture 4	Color

3. Create an image array `img_green` where the red and blue pixels of `img` are set to 0.

4. Use the cv2.cvtColor() function and create an image array `img1_g` that is a grayscale image from `img1`.

5. Create the following functions that implement lightness, average, and luminosity techniques for creating grayscale images. These are defined as stubs in the ps03.py module.

Function Name:	Formula

`def makeGrayLightness(colorImage):`
`    return grayImage`
	`(max(R,G,B)+min(R,G,B))/2`

`def makeGrayAverage(colorImage):`
`    return grayImage`
	`(R+G+B)/3`

`def makeGrayLumin(colorImage):`
`    return grayImage`
	`0.21*R+0.72*G+0.07*B`

6. Using the color `img` array, use the functions from requirement 6 to create the following:

```
img_grayLight = makeGrayLightness(img)

img_grayAverage = makeGrayAverage(img)

img_grayLumin = makeGrayLumin(img)
```

7. Use `img3` and `img4` for following.
   a. Make Luminosity Grayscale images `img3_gray`, `img4_gray`. (Use the `makeGrayLumin()` function from step 5)

   b. Write a function called `averageImages(a, b)` that will return an image consisting of averaging the values between inputs a and b.

   c. Use the `averageImages(img3_gray, img4_gray)` and to create an image `img34_average`.

8. Take the center 100x100 pixels of your `img1` and place them into the center 100x100 pixels of `img2`. Name this new array `img_center_patch`.

9. Use the `imshow()` function and display the following images onto the screen with the assigned Window names.

Window Name	Image array to display
Lisa	`img`
Lisa Green	`img_green`
Lisa Gray	`img1_g`
GrayLight	`img_grayLight`
GrayAverage	`img_grayAverage`
GrayLumin	`img_grayLumin`
Img3 Gray	`img3_gray`
Img 4 Gray	`img4_gray`
Average	`img34_average`
Patch	`img_center_patch`

10. Make sure you have an exit sequence in the Python code to keep the program from crashing out when the images are closed.