Image processing in Python: assignment

Part 1.

i. Write some useful information like a title and your names in some comments at the top of the code.

ii. Load in the file "CT.bmp" and display it

iii. Plot a histogram of the image.

iv. Write a function that displays the CT image with different window levels. The input variables should be the window and the level.

```
def yourfunction(window, level):
    #YOUR CODE HERE
```

v. Identify which parts of the histogram relate to which parts of the image.

Part 2.

vi. Write some useful information like a title and your names in some comments at the top of the code.

vii. Import the libraries that you will need. You will need to import `matplotlib.pyplot` and `numpy`. You will also need `misc` from `scipy` and `interpolation` and `rotate` from `scipy.ndimage`.

viii. Load in the 2 images "lungs.jpg" and "lungs2.jpg"

ix. Display the images.

x. Make a plot where you can see both images on the same axes using transparency.

We will keep one of these images fixed (lungs.jpg) and make one of them able to move (lungs2.jpg). So that you can reference this moving image specify a name for the axis when you plot it. e.g.

```
floating = ax.imshow(…)
```

Later you can use:
```
floating.set_data(…)
```
to set the image in this axes to something else (like the image after we have manipulated it)

xi. Write a function that shifts your second image given an input argument called "shifts" which is a list of shifts vertically and horizontally. When your function is called the command
```
shiftImages([10,20])
```
should shift the image down 10 pixels and to the right 20 pixels. This will get you started.

```
def shiftImages(shift):
    global image2

    # YOUR CODE HERE

    floating.set_data(image2)
    fig.canvas.draw()
```

By defining image2 as a global here, changes we make to it inside the function have an affect outside the function too.

If you called the axes showing the second image “floating” then the figure will be updated with the second to last line.

The last line will update the figure if it is already displayed. This will make more sense later when our figure becomes interactive.

xii. Evaluate your function by calling it. What does `shiftImage([10,20])` do?

xiii. What shifts are needed to align the images?

xiv. Modify your code to include rotations. Load in “Lungs3.jpg” as your secondary image instead.

xv. Let's make it so that we can shift our image using keyboard presses.
To do that we will need to connect our matplotlib figure to a `key_press_event`.
We do that using the code below:

```
fig.canvas.mpl_connect('key_press_event', eventHandler)
```

This connects the figure "fig" to key presses. When a key is pressed the specified function (here it is called "`eventHandler`") is called. The first argument of this function is the event.
We haven't defined this function yet though.
You will need to write it. Below is an example of how it works to get you started.

```
def eventHandler(event):
    """
    This function handles deciphering what the user wants us
    to do, the event knows which key has been pressed.
    """

    up = 0

    whichKey = event.key
    if whichKey == "up":
        up = 1

    print up
```

What happens if you press the up key? what happens if you press any other key? Try printing whichKey to show what each key is called.
Your `eventHandler` should call the function `shiftImages` that you wrote before and send it the required arguments.

After you have written the function you can connect the event handler to the plot

```
fig.canvas.mpl_connect('key_press_event', eventHandler)
plt.show()
```