Task 4 – FT Magnitude & phase Mixer

Introduction: Fourier transform is a fundamental tool in the signal processing field.

UI Mockup: https://ninjamock.com/s/DDLFTGx

Design and implement a piece of software that explains the relative importance of the magnitude and phase components. We will do this task on a 2D signal (i.e. image) just for clarity but the same concept applies to any signal. Your software should have the following features:

1. Ability to open and show two images. For each image, the software should have two displays (one fixed display for the image, while the second display can show several components based on a combo-box/drop-menu selection of 1) FT Magnitude, 2) FT Phase, 3) FT Real component, 4) FT Imaginary component.
   Note: the software should impose that the two images have the same size. i.e. when opening the second image, the sw should check it has the same size of the one previously opened. Otherwise, give an error message.
2. A mixing panel where an output will be formed based on the mix of two components. Each one of these components should be determined from:
   a. which image (via a combo or drop-menu). Available images are image 1, and image 2.
   b. Which component of the image FT (via a combo or drop-menu). Available components are: Magnitude, Phase, Real, Imaginary, uniform magnitude (i.e. all magnitude values are set to 1), uniform phase (i.e. all phase values are set to 0).
   c. the mixing ratio (via a slider ranging from 0 to 100%).
3. Based on the mixing panel, an output image should be generated and displayed for the user. There should be two available displays, each for one output. The mixing panel should be sending the output to either display-Output 1 or 2. The display is determined using a drop-menu in the mixing panel. And the output image should be updated whenever the user makes a change in the mixing options. There should not be a button for generating the output. The output is generated on the fly whenever the user changes a mixing option.

Code practice:

- Same practices from previous tasks (i.e. proper variable names & No code repetition) will continue with task 4.
- Apply OOP Concepts:
  - Think of part A in terms of OOP. If you start implementing each image and its components, the code will turn out to have a lot of repetitions.
  - Apply the encapsulation concepts of OOP. Do not create a class for the image and its display and then start implementing everything outside! I'm expecting to see very few code in your main function and most of the code inside your image/display class! No mathematical manipulation for the image should be handled outside the image class!
- Logging: Logging is a very important concept you should get used to. It helps you to track how problems happen and figure out their resolution much quicker. Python has a logging library. All you need to do is to import it and start logging the main user interactions and main steps into some text file. I need to see how logging has helped you to debug the problems you faced during your development. i.e. Do NOT just throw any trash variables to the log file to show you did it. You will be asked why you logged this variable or that.