Carleton University
Department of Systems and Computer Engineering
SYSC 1005 - Introduction to Software Development - Fall 2017

Prelab Exercises for Lab 4 - Introduction to Image Processing

Have your solutions to these exercises available when you attend Lab 4 (save them on a flash drive or a cloud-based file storage service, or email the Python file to yourself).

Getting Started

Step 1: Create a new folder named Lab 4 Prelab.

Step 2: Download lab_4_prelab.py, Cimpl.py and the image (JPEG) files from cuLearn to your Lab 4 Prelab folder.

Step 3: Launch Wing IDE 101. Check the message Python displays in the shell window, and verify that Wing is running Python version 3.6.

Part 1 - Learning to Use the `Cimpl` Module

Exercise 1

Step 1: Open lab_4_prelab.py in a Wing IDE editor window. This module contains the definitions of two functions that modify images, **`maximize_red`** and **`reduce_red_50`**.

Step 2: Click the Run button. This will load the module into the Python interpreter and check it for syntax errors.

Step 3: You're now going to use the **`Cimpl`** module to choose one of the image files, load it into memory and display it. You'll call a function to modify the image, then display the modified image.

Important: When you execute a command in the Python shell, the message

> Executing command. Please wait for result.

appears at the top of the Python Shell window. Functions that modify large images can take 3 or 4 seconds (or longer) to execute. **Do not type commands until the function has finished and the shell prompt (>>>) is displayed. If you type while a function is executing, there is a good chance that Wing IDE will hang.** If this happens, you will have to exit the IDE and relaunch it.

In the Python shell, type this statement to select an image file and load it into memory:

```
>>> image = load_image(choose_file())
```

You may have to move some of the windows on the screen in order to see the file chooser dialogue box. Select one of the JPEG files in your Lab 4 Prelab folder, then click the Open button. The contents of the selected image file will be decoded and stored in an object, which

will be bound to variable **image**.

What is the type of the object that is bound to **image**? To find out, we can call Python's **type** function. Try this:

```
>>> type(image)
```

Will evaluating **image** in the shell display the image that is bound the variable? Try this:

```
>>> image
```

Clearly, evaluating **image** does not cause the image to be displayed.

We can call **Cimpl**'s **show** function to open a window that displays the image. Try this:

```
>>> show(image)
```

The shell prompt won't reappear until you click the window's Close button.

Read the definition of **maximize_red** (the header, the docstring and the function body), then call the function:

```
>>> maximize_red(image)
```

Call **show** to verify that **maximize_red** modified the image:

```
>>> show(image)
```

Click the window's Close button.

Use Python's help facility to display information about the **Cimpl** functions used in this exercise. Type these statements:

```
>>> help(choose_file)

>>> help(load_image)

>>> help(show)

>>> help(create_color)

>>> help(set_color)
```

Exercise 2

Read the definition of **reduce_red_50**.

From the shell, use **Cimpl** to select an image file and load it into memory. Call **reduce_red_50** to modify this image, then display the modified image.

Part 2 - Developing Some Simple Image Manipulation Functions

All the functions that you write for this part must have a docstring containing:

- the function's type contract,
- a brief description of what the function does, and
- an example of how we can interactively test the function from the shell.

For an example of what we're looking for, see the docstrings for **`maximize_red`** and **`reduce_red_50`**.

Exercise 3

Step 1: In lab_4_prelab.py, define a filter named **`maximize_green`**. The function header is:

```
def maximize_green(image):
```

This filter has one parameter, **`image`**, which refers to the image that the function will modify. This function should set the green component of each pixel to its maximum value, leaving the red and blue components unchanged.

Step 2: Use the shell to test **`maximize_green`**. To do this:

- Save the edited file, then click the Run button. (If you forget to do this, you won't be able to call **`maximize_green`** from the shell, because you haven't saved the modified module and loaded it into the Python interpreter.)
- Notice that clicking Run resets the shell, which means that all variables and objects created in the previous shell session are lost. You'll have to choose an image file and load the image before calling **`maximize_green`**. Remember to bind the **`Image`** object returned by **`load_image`** to a variable; e.g., **`image`**.
- Call **`maximize_green`**.
- After **`maximize_green`** returns, display the modified image.

Step 3: In lab_4_prelab.py, define a function named **`maximize_blue`** that is passed an image. This function should set the blue component of each pixel to its maximum value, leaving the red and green components unchanged.

Step 4: Use the shell to test **`maximize_blue`**; that is, load an image file, call **`maximize_blue`** to modify the image, and display the modified image.

Exercise 4

Step 1: In lab_4_prelab.py, define a filter named **`reduce_green_50`**. The function header is:

```
def reduce_green_50(image):
```

This filter has one parameter, **`image`**, which will refer to the image that the function will modify. This function should set the green component of each pixel to 50% of its current value, leaving the red and blue components unchanged.

Step 2: Use the shell to test **`reduce_green_50`**.

Step 3: In lab_4_prelab.py, define a function named **`reduce_blue_50`** that is passed an image. This function should reduce the blue component to 50% of its current value, leaving the red and green components unchanged.

Step 4: Use the shell to test **`reduce_blue_50`**.

Wrapup

Your lab_4_prelab.py module should now have these six functions: **`maximize_red`**, **`maximize_green`**, **`maximize_blue`**, **`reduce_red_50`**, **`reduce_green_50`** and **`reduce_blue_50`**. You're now ready to do Lab 4.