Do **not** copy code from colleagues, internet or other sources. Do **not** post the code template or any part of the assignment material on the internet in any form on any site. You may discuss approaches together with your peers, but all code must be your own. Presenting work other than your own during lab sessions or at a viva is plagiarism under university policy and will be regarded as such. Doing so is a recipe for disaster.

Aims:

- Understand how an image is stored internally, and how to manipulate the image
- Translate useful graphics algorithms into working code
- Improve your programming skills
- Task based learning – seek out information sources to carry out the assignment
- Understand that graphics can be useful to users
- Combine interaction with visual feedback
- Practice presenting your work in a viva situation

Files:

The supporting framework is written in Java. You may build on this framework. If you wish to carry out the coursework in a different programming language you may do so, but there will be no provided framework. For any choice of programming language, the algorithms must be of your own implementation. Reliance on external library code to do the core implementation questions will **not** constitute a valid submission. The assignment is to be demonstrated in a viva (you may do so on your own laptop). You will be required to demonstrate your working program to me or a post-graduate demonstrator in the week of the deadline at times to be arranged.

You will require the following to start the exercise:

1. A copy of the Java template – downloadable from blackboard. This demonstrates how to display and manipulate images, and also functions and procedures which will help you with the exercise.

[Note: You must **not** redistribute the provided material anywhere - you do **not** have permission to do that]

Doing so may qualify as unfair practice.

Exercise – Implement the following:

1. Gamma Correction:
    a. A fast implementation of gamma correction (details in lecture).
    b. An interface to provide the value to gamma correction.
2. Contrast stretching:
    a. Contrast stretching algorithm.
    b. An interface to provide the (r1, s1) and (r2, s2) points to the algorithm.
3. Histogram:
    a. Calculate the histogram(s) for the image.
    b. Output the histogram(s).
    c. Full marks are awarded to calculating and displaying histograms for red, green, blue and brightness.
    d. Histogram equalisation carried out on the image.
4. Cross-correlation
    a. Implement cross-correlation using the 5x5 Laplacian Matrix.

Computer Graphics Assignment - Marking scheme [100 marks total]

Gamma Correction [20 marks]

1. Has gamma correction been implemented correctly? [15 marks]
2. Can a value for gamma be entered by some interface rather than in code constant? [5 marks]
3. If the student has done at least 1 thing from above, does the student answer a suitable set of questions about their implementation and gamma correction? – e.g. What is the purpose of gamma correction, what is the equation for gamma correction, how is this implemented, how are the other parts implemented? **Deduct marks** from above if the student cannot or hesitantly describes their own code.

Contrast Stretching [25 marks]

1. Has contrast stretching been implemented correctly? [10 marks]
2. Can values for (r1, s1) and (r2, s2) be entered by a complex interface (e.g. via mouse, creating a line chart like Photoshop)? [15 marks]. (7 marks here if using text boxes or sliders, 0 if using in code constants).
4. If the student has done at least 1 thing from above, does the student answer a suitable set of questions about their implementation and contrast stretching? – e.g. What is the purpose of contrast stretching, what is the equation for contrast stretching, how is this implemented, how are the other parts implemented? **Deduct marks** from 1-3 above if the student cannot or hesitantly describes their own code.

Histogram [25 marks]

1. Has histogram calculation been implemented correctly? (Do not copy my previous histogram code which you may find, and which does not answer this question). [4 marks]
2. Can the histograms for red, green and blue and brightness be displayed using a fancy interface (e.g. using a chart library in Java)? [14 marks] Just [4/14 marks] if only one histogram is displayed by printing values. Just [6/14 marks] if four (RGB and Brightness) can be displayed by printing values.
3. Has histogram equalisation been implemented correctly? [7 marks]

If the student has done at least 1 thing from above, does the student answer a suitable set of questions about their implementation and histograms? – e.g. What is the purpose of histogram equalisation, what is the algorithm for histogram equalisation, how is this implemented, how are the other parts implemented? **Deduct marks** from 1-3 above if the student cannot or hesitantly describes their own code.

Cross-correlation [30 marks]

Has cross-correlation been implemented correctly? Hard coding a 5x5 Laplacian is all that is needed for this assignment. Edges may be set to black. [30 marks]

If the student has done this, does the student answer a suitable set of questions about their implementation and cross-correlation? – e.g. What is the purpose of cross-correlation, what is the algorithm, how is this implemented, what are high pass and low pass filters, what does the Laplacian do? **Deduct marks** if the student cannot or hesitantly describes their own code.

Requirements for submission:
You will demonstrate your program working to me in the lab slots (2pm-4pm Monday and Tuesday) or assigned viva slot. Therefore, in order to submit your work:

- You must have a comment with your name and student number at the top of all your java files along with the declaration that it is your own work.
- Place all your java files in a single ZIP file named your student number (i.e. **600134.zip**) and submit this ZIP file to blackboard.
- **Do not** include the image(s) used to demonstrate your code in your ZIP file submission as this will unnecessarily increase the ZIP file size to be stored on blackboard.
- For the viva, please arrive early before your assigned slot and prepare by setting up and opening your code, being ready to demonstrate it before the grader walks over to mark you.

Mark distribution:
See later for a breakdown. You will get 0% if you do not submit anything. You will get 0% if you do not give a viva (even if you submit some code via blackboard).

5x5 Laplacian Matrix (for reference):

```
-4 -1  0 -1 -4
-1  2  3  2 -1
 0  3  4  3  0
-1  2  3  2 -1
-4 -1  0 -1 -4
```

Java and Bytes:
Java does not support the unsigned types (e.g. unsigned byte). Images store the red, green and blue values as unsigned bytes (0 to 255), therefore some problems can arise - e.g. a value of 255 can be interpreted as -1. In order to work around this problem, one needs to carry out a logical AND with a bit mask of 255 and correctly typecast the result to int. e.g. int red = image_byte[index]&255; Examples are contained in the code template.

Discrepancies:
If you find a discrepancy between any information you receive about this course, first please tell me, and then note that **Blackboard** should be regarded as being correct (since I will resolve the discrepancy by making sure Blackboard is correct, and creating a clarification announcement on Blackboard).

Feedback:
Feedback will be sent to university email addresses and home email addresses (make sure these are up-to-date). It will not be sent via blackboard but grades will be visible in the module grade centre.