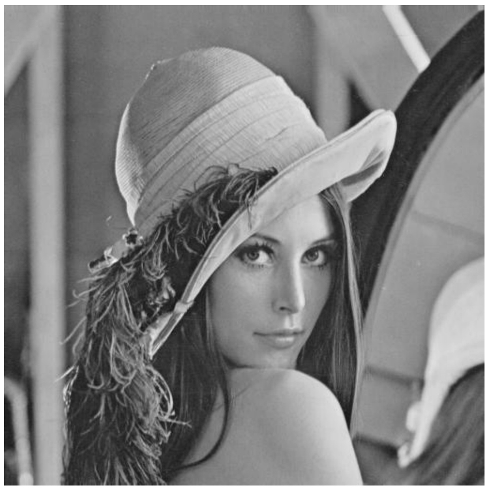
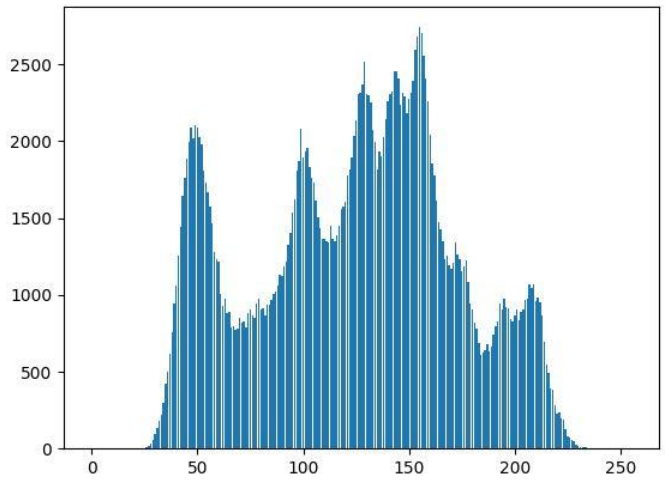
Computer Vision HW3 Report
蕭恩慈 / B07902095

The task is to write a program that generates images with certain specifications and its histogram.
To complete the task, I use Python as my program language with PIL, matplotlib, numpy, and csv modules.
First open the original image (lena.bmp) and get the image height and width.
Part a is to generate the original image, and its histogram. Since we just want the image output to be the same as the original, we can just save it as an output image (original-lena.bmp) then create its histogram.
To create the image histogram, first create an array using zeros for the histogram data, then "chop" the image into pixels then store the count result into the histogram array that we just created, then create the csv file from the array so it can be plot into histogram. After plotting the histogram, save the image as an output file.

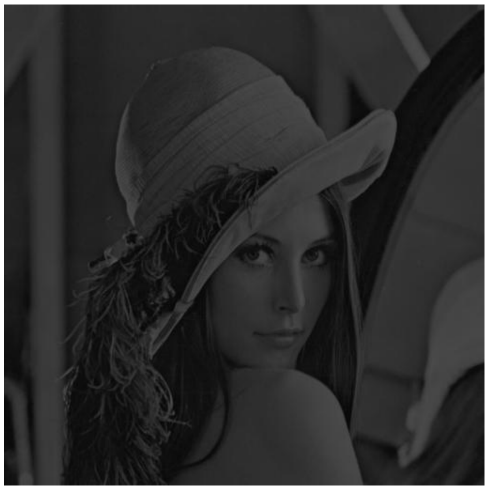
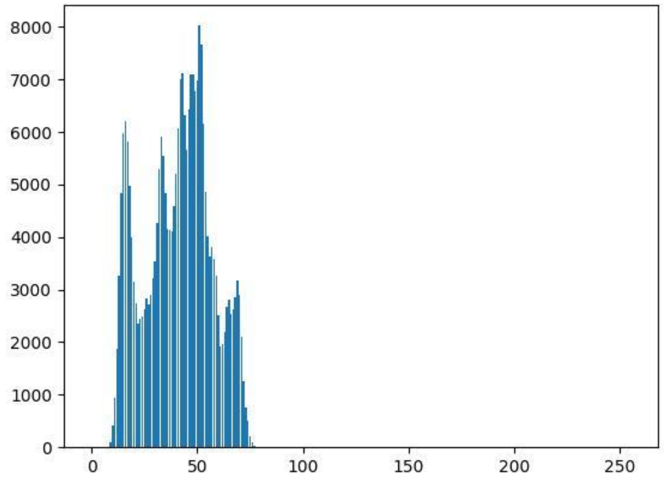
Result:

(left: original-lena.bmp; right: original-lena-histogram.jpg)

Part b is to generate an image with intensity divided by 3 and its histogram. To put it simply, make every pixel from the original image into a third of each pixel. To get the result, after chopping the original image, save the pixel into a newly created image file with // 3 for each pixel, which means divided by 3 and get the lower bound of the result, then store accordingly to each pixel position (intensify-lena.bmp). After getting the image result we wanted, we can now create its histogram by using the same method from the first part. (save file name: intensify-lena-histogram.jpg)

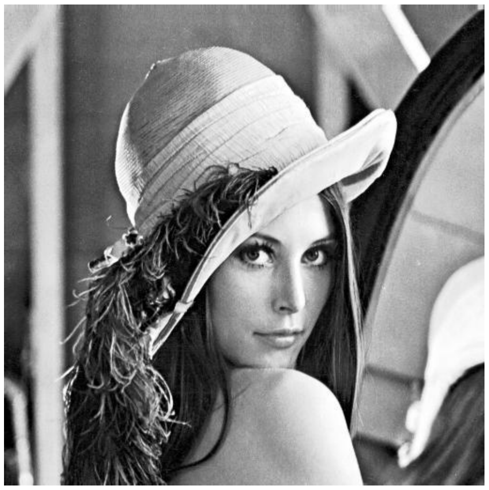
Result:

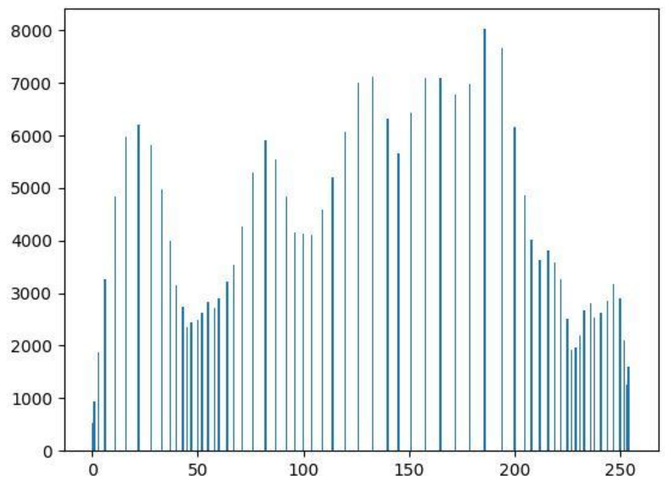



(left: intensify-lena.bmp; right: intensify-lena-histogram.jpg)

Part c is to generate an image after applying histogram equalization to (b) and its histogram. So we want to apply the histogram equalization to the image result from part b, then create its histogram. First we want to create the transformation table, before applying it to each pixel to get the equalization. First create an array to store the data later, then using the formula s = T(r) from the slide, count each pixel by implementing histogram equalization histogram linearization from the slide's formula (sk), then store it to the array we just created. Now chop the (b) image into pixels, then use the result of the transformation into each pixel, creating the new image. (equalized-lena.bmp). Lastly, after getting the image result we wanted, we can now create its histogram by using the same method from the first part. (save file name: equalized-lena-histogram.jpg)

Result:





(left: equalized-lena.bmp; right: equalized-lena-histogram.jpg)