CSCI-160 *Image Project*

ImageFrame.java (revisited)

1 Reminder: Project Handin Rules

1. All project due dates are specified on Brightspace and on the syllabus, but the Brightspace due dates will always be correct, while the syllabus dates may need to be adjusted.
2. Projects may be turned in up to 7 days late. You will lose 5% of the grade for each day late.
3. If you hand in a semi-functional project on time, you are then allowed to hand it in a second time for an average of the two grades. Typically you would wait to receive your project grade and suggestions for improvement, and then use those suggestions to improve your project for your second handin. Handing in a project that will earn 0 points on time will not help your grade much. Do the math. Handing in a perfect project one day late would earn 95 points. A project that earns 0 followed by a perfect project a day later, would earn 50 points.
4. If you hand in a project late, you are only allowed to hand it in once for credit.
5. For projects, you will hand in what is specified by individual assignments. Sometimes this will be .java files sometimes it will be an entire zipped project archive.

2 Introduction

You must have finished both the homework and the lab before starting on this project. Another way to enhance an image is to modify its histogram and then use this to create the modified image. The histogram of an image keeps a tally, over each possible gray value in the image, of the number of pixels that have that gray value.

For example, suppose that we have a camera that only permits us to take photos with gray values 0-10, and that we have taken a 3X3 pixel image with the following pixel values:

```
 0      0      3
 2      2      2
10      8      5
```

The histogram would be:

Pixel value(index):	0	1	2	3	4	5	6	7	8	9	10
Number of pixels with value:	2	0	3	1	0	1	0	0	1	0	1

That means there are 2 pixels with value 0, 0 pixels with value 1, 3 pixels with value 2, etc.

Histogram equalization is a technique that increases the dynamic range of the histogram of an image. Histogram equalization modifies the gray values of each pixel in an image so that the image contains a uniform distribution of intensities. This is another method to improve the contrast of the image.

There are three steps to histogram equalization:

1. Create the histogram for the image as described above.

2. Create a set of scaled gray values for each gray value by creating a uniform histogram. This is easiest done in two steps. First calculate the sums, then scale them. This is done using the following formula:

   $$E_i = \sum_{j=0}^{i} H_j * MaxGV/TotalNumberOfPixels$$

   where E is the value of the new, uniform histogram at gray value i. H_j is the value of the original histogram value at gray value j. MaxGV is the largest gray value in the image (for the example, 10 because there is one pixel with value 10 in our image.)

   The sums before scaling would be these:

Pixel value(index):	0	1	2	3	4	5	6	7	8	9	10
Number of pixels with value:	2	2	5	6	6	7	7	7	8	8	9

   This shows us that the sum of the pixel counts 0 and below is 2. The sum of pixels with values 1 and below is also 2 because there were 0 pixels with value 1. The sum of the pixels with values 2 and below is 5 because there were 3 pixels with value 2. Etc. Once the sums are calculated, you can scale them

   The scaled value for pixels with 0 as the original value would be
   2 * 10 / 9 = 20 / 9 = 2.2 = 2 (because it has to be an int.)

3. For each pixel in the image, replace the original gray value with the new gray value (obtained from the new uniform histogram.) That is, if the original gray value of a pixel is i, replace it with the value E_i . The new hEnhancedImage.image would be this:

   ```
    2    2    6
    5    5    5
   10    8    7
   ```

Write a design before starting to code. The method that you need to write begins on line 232 of the ImageFrame.java file that you worked on in lab.

3 Hand in Your Lab

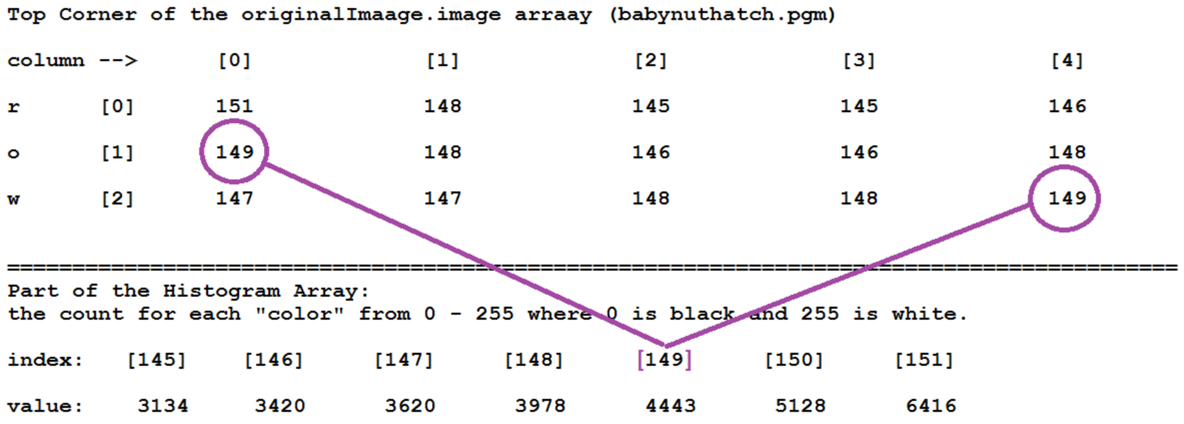
If you are in a classroom setting, you can show your working program to your instructor during class or office hours. If you are an online student or can't show it to your instructor, you will need to upload it. **Upload only the ImageFrame.java file to Brightspace**. DO NOT zip it, DO NOT rename it.

4 Grading Criteria

Declaration of variables with meaningful names	10%
Correct calculation of all values	45%
Complete and correct output	25%
Programming style:	20%
Total	100%

Count the number of each value in the original image

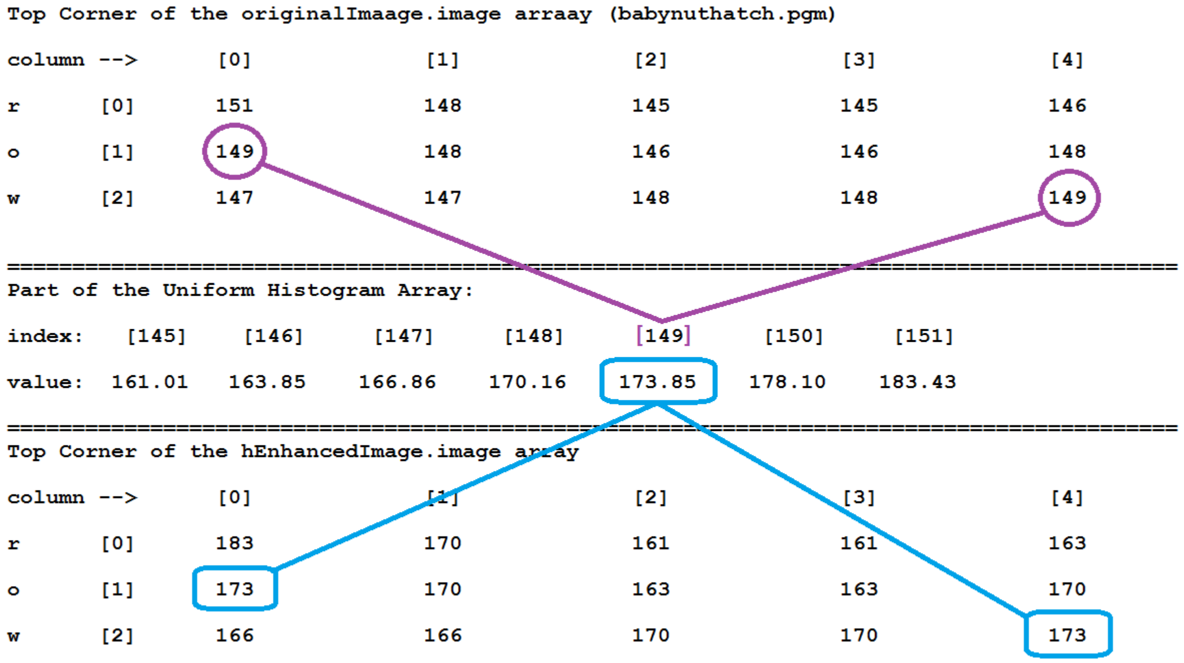
The value of an element in the original image is the index in the histogram array.



```
Top Corner of the originalImaage.image arraay (babynuthatch.pgm)

column -->      [0]         [1]         [2]         [3]         [4]

r       [0]     151         148         145         145         146

o       [1]     149         148         146         146         148

w       [2]     147         147         148         148         149

=============================================================================
Part of the Histogram Array:
the count for each "color" from 0 - 255 where 0 is black and 255 is white.

index:   [145]    [146]    [147]    [148]    [149]    [150]    [151]

value:    3134     3420     3620     3978     4443     5128     6416
```

Use the scaled values associated with the original values in the new image

The value in the original image gives you the index of the new scaled value that goes into that location in the new image.



```
Top Corner of the originalImaage.image arraay (babynuthatch.pgm)

column -->      [0]         [1]         [2]         [3]         [4]

r       [0]     151         148         145         145         146

o       [1]     149         148         146         146         148

w       [2]     147         147         148         148         149

=============================================================================
Part of the Uniform Histogram Array:

index:   [145]    [146]    [147]    [148]    [149]    [150]    [151]

value:  161.01   163.85   166.86   170.16   173.85   178.10   183.43

=============================================================================
Top Corner of the hEnhancedImage.image array

column -->      [0]         [1]         [2]         [3]         [4]

r       [0]     183         170         161         161         163

o       [1]     173         170         163         163         170

w       [2]     166         166         170         170         173
```