Students' name:
Phan Manh Tung
Yahaya Moussa Moubarak
We are students from CoDAS program, joining M1 MoSIG since our program was canceled.
We really hope to have your support in order to catch up with our class. Thank you so much for your consideration.

Exercise 1:
For this exercise, we write 3 functions including:
+ gray* binomial_filter_5x5(gray* new_image, gray* current_image, int cols, int rows)
+ gray* binomial_filter_3x3(gray* new_image, gray* current_image, int cols, int rows)
+ gray* median_filter(gray* new_image, gray* current_image, int cols, int rows)

which apply the filters to the current_image, returning the new_image as the result.

For the median filter, we use a snippet of code to sort the array from the following link:
https://stackoverflow.com/questions/1787996/c-library-function-to-perform-sort
to fast forwards the process.

To apply the functions, we simply change the filter's name and the number n times to apply the filter inside the main function.

Terminal commands for execution:
make
./pgmtopgm boat_noise1.pgm > bino_3_1.pgm

Method:
+ For binomial filters, we create the masking matrices (3x3 or 5x5) based on Pascal Triangle and apply convolution t o the original image.
+ For median filter, each pixel is replaced by the median of the surrouding 9 pixels.

Output: bino_3_1.pgm, bino_3_5.pgm, bino_5_1.pgm, median_1.pgm (bino_3_1.pgm -> binomial filter 3x3, n=1)
We also convert pgm into jpg to open images easier

Discussion:
+ Binomial filter makes the image smoother (bino_3_1, bino_5_1). If we apply it many times, the image will be blur red (bino_3_5)
+ Median filter is used for noise removal (eliminate intensity outliers).
It removes some random black or white pixels (noise) in the image, making it more smooth and clear (median_1).

Exercise 2:
We have no outside function in this exercise, all process is inside the main() function. There are 4 main steps to perf orm Histogram Equalization:
+ Step 1: build the histogram from the image
+ Step 2: calculate the cumulative sum histogram
+ Step 3: apply Histogram Equalization into the cumulative sum histogram
+ Step 4: replace each pixel in the original image with a new value from the cumsum result (step 3), using pixel's val ue as index
loop through image: new_image_arr[i] = cumsum[graymap[i]]

Terminal commands for execution:
make
./pgmtopgm boat_noise1.pgm > hist_equal.pgm

Supporting resource: https://medium.com/hackernoon/histogram-equalization-in-python-from-scratch-ebb9c8aa3f23

Method:
+ Find out the cumulative sum histogram of all pixels in image, normalize the values from 0 to 255 then reapply the normalized results back to the orginal image.

Discussion:
+ After implementation, the intensity contrast is enhanced, making darker gray pixels become black and lighter gray pixels become white.
Thus, the boat and the behind scene look more substantial.

Thank you so much for your time!