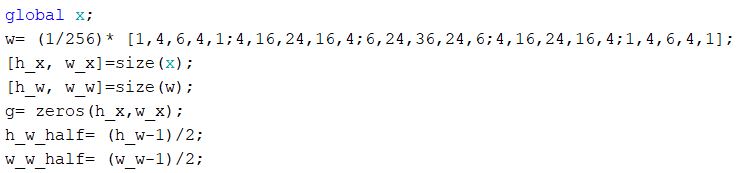
1st gaussian blurring:-

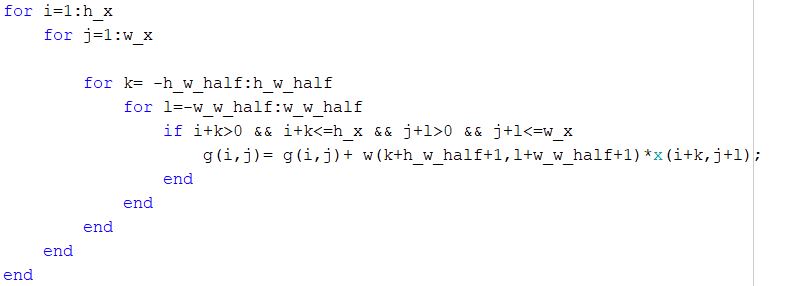


I used Kernel 5x5 and stored it in Variable W.

Stored the size of original image in x and the size of the kernel in variables.

created empty matrix to store the output image.

To Start the loop and remove the borders of the image to centralize the mask on the image I took the half of the kernel rows and columns and stored it in variables.

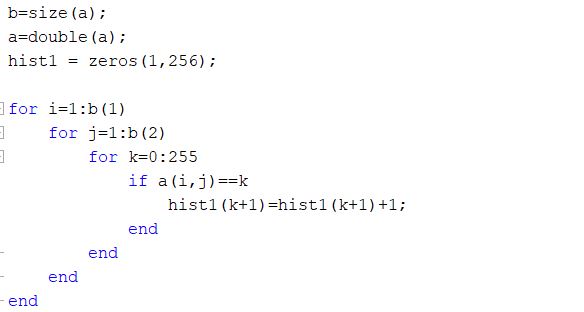


1st loop and 2nd one looping on the original image second two loops looping on the kernel halves.

The if is checking for skipping the borders so for example in this case, I won’t enter the IF unless the I and j is equal to 3.

Then the line under the IF is storing the output image by adding the pixel in some index of g and the kernel multiplying by the original image after centralizing the mask on each pixel of the image.

2nd Histogram Equalization: -



First, I stored size of the image in b and doubled it to avoid any mathematical mistakes

Then created empty matrix to store the histogram of the original image in

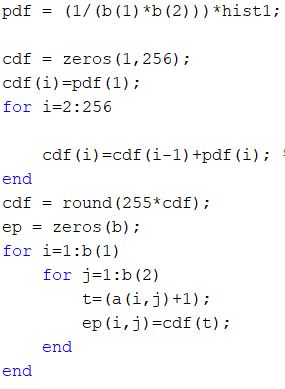
First loop is for tracing rows.

Second loop is for tracing columns.

Third loop checking image gray level.

The If checking each pixel of the original image is equal to which gray level.

Then shifting values to next position after increasing them by 1 to avoid any 0 value.

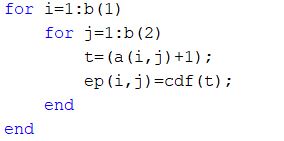


Generating PDF by normalizing the histogram by dividing the histogram over the multiplication of number of rows and number of columns PDf=histogram / total of number of pixels.

Then generating empty matrix called CDF.

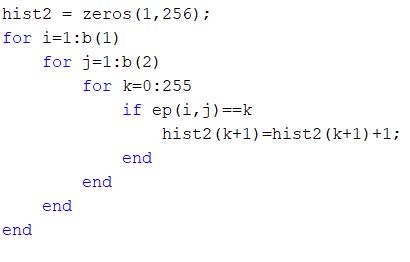
Then I will Generate the CDF Out of The PDF for example by adding cdf (1) to pdf (2).

Then I will round it up because there is no gray level in fraction then it will be the linear transformation function.



At this loop i am making the output image using cdf as transformation function by taking the values which is calculated by multiplying the scale and rounding it up for the array

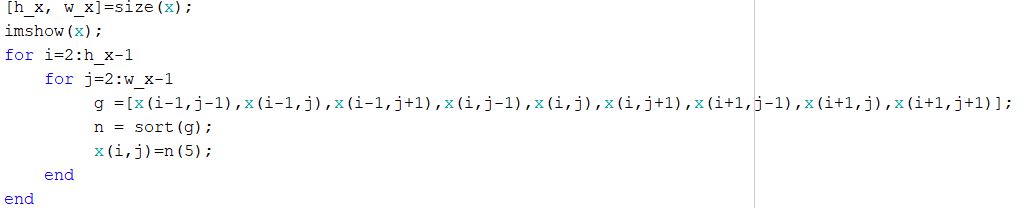
And storing it in empty matrix called ep.



here I am creating empty matrix to store the Equalized histogram in it called hist2.

Then, I loop again on rows and columns and checking gray level as before and storing values in hist2.

3rd Median Filter:-



In median filter is all about talking 9 neighbor’s pixels and sorting them then take number 5.

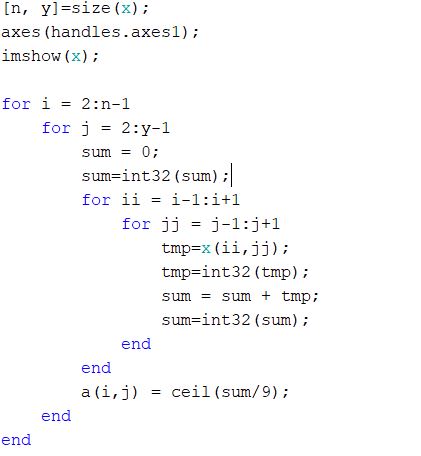
Here I stored size of x

Then, I loop on the rows and columns storing the pixels in array g.

Then, I sort them in array n.

Finally, I take the pixel number 5 and replace it with the first pixel in the original image.

4th Averaging filter: -



First, I stored size of x.

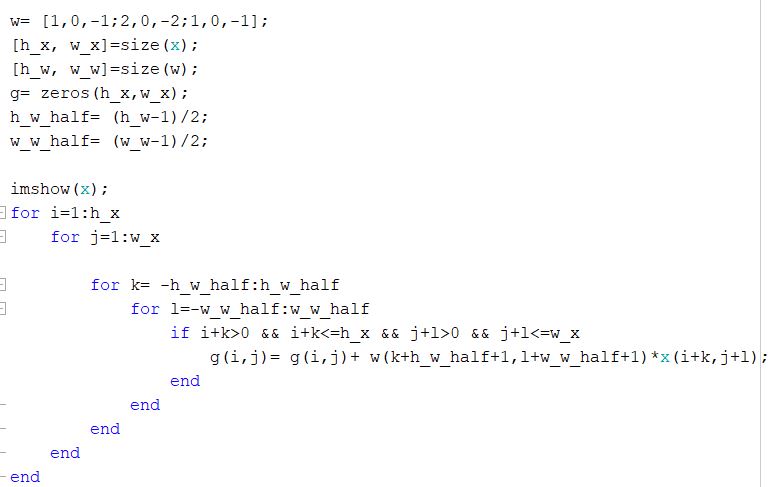
Then I looped on rows and columns.

Then, Declared variable sum equal to 0.

Then, I looped on pixels and adding them into the sum.

Finally, storing the averaged pixels into their indexes Ceil (sum/9) rounds the elements of sum to the nearest integers greater than or equal to sum. For complex sum, the imaginary and real parts are rounded independently.

5th sobel Filter: -



I used Kernel 3x3 and stored it in Variable W.

Stored the size of original image in x and the size of the kernel in variables.

created empty matrix to store the output image.

To Start the loop and remove the borders of the image to centralize the mask on the image I took the half of the kernel rows and columns and stored it in variables.

1st loop and 2nd one looping on the original image second two loops looping on the kernel halves.

The if is checking for skipping the borders so for example in this case, I won’t enter the IF unless the I and j is equal to 3.

Then the line under the IF is storing the output image by adding the pixel in some index of g and the kernel multiplying by the original image after centralizing the mask on each pixel of the image.