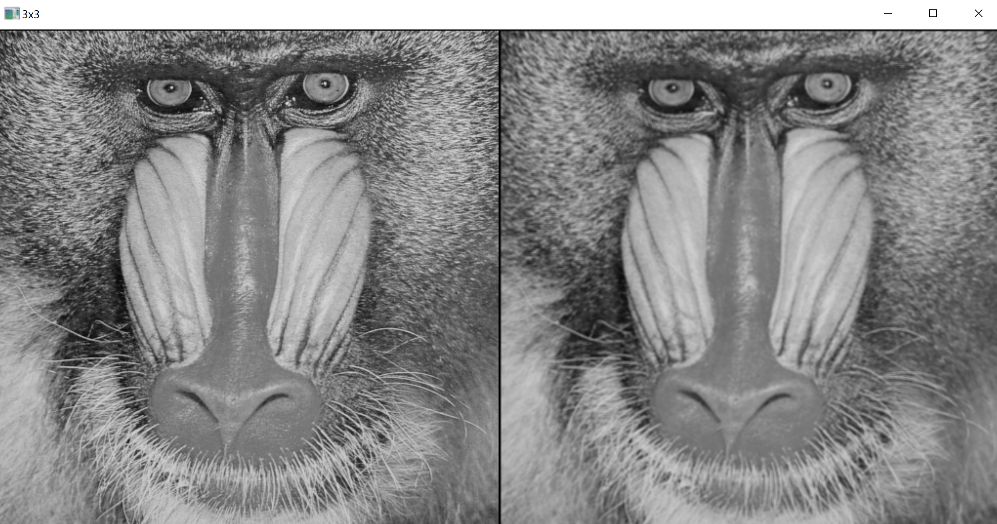
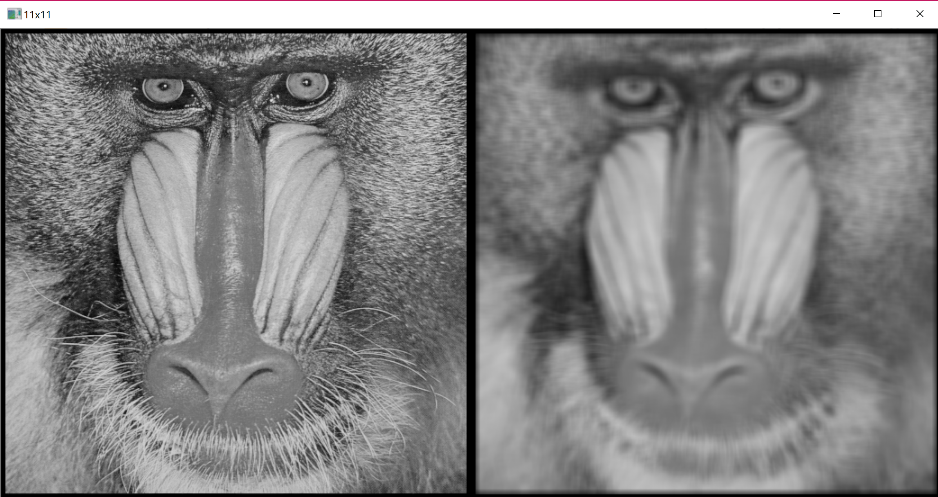
Homework1

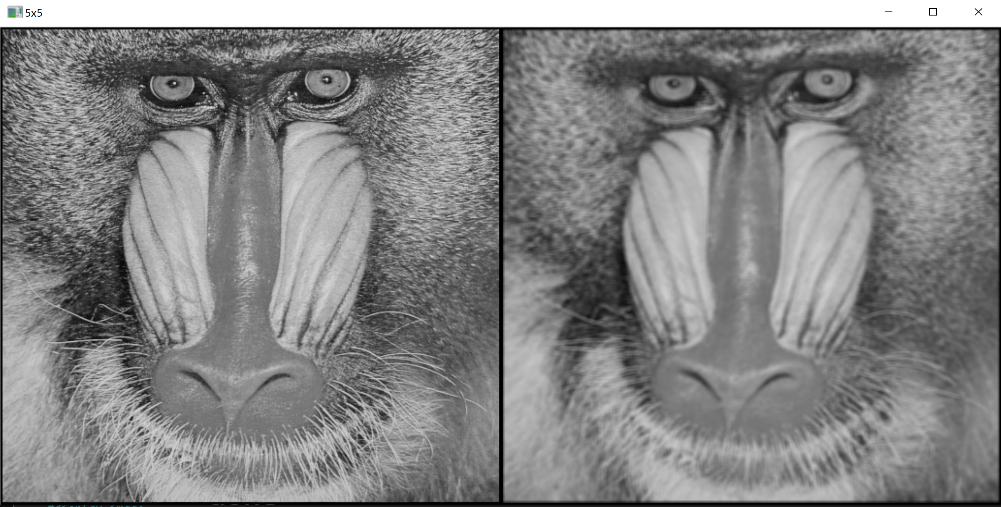
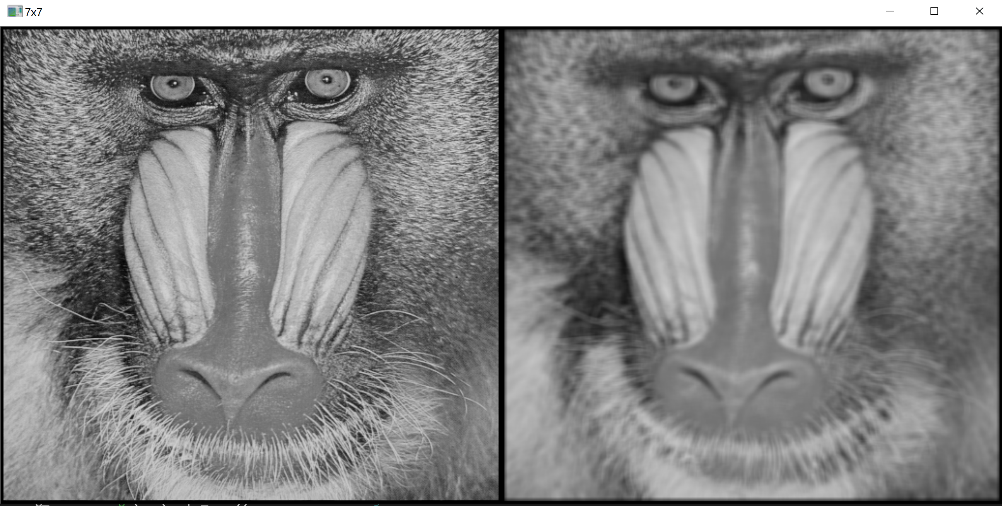
Sinem Ozden – 150150202

In this homework I implemented a filtering method for images. First, I read images from a directory and put them on a list. Then, add a padding to images to prevent them get smaller because of the filtering process and I placed the padded images to a list to access them later. I created 4 kernels for different filtering processes. I also created empty padded kernels for filtered images.

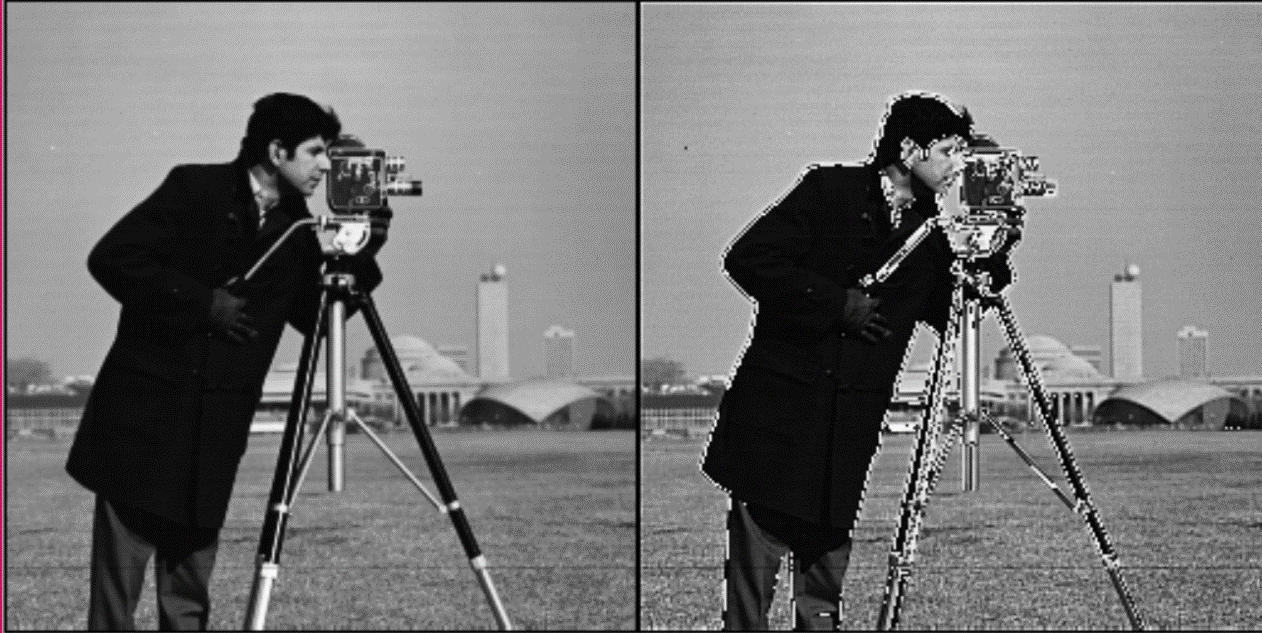
All blurring kernels filled with ones and divided with a value squared of their size. As the matrix get bigger the image get more blurred.



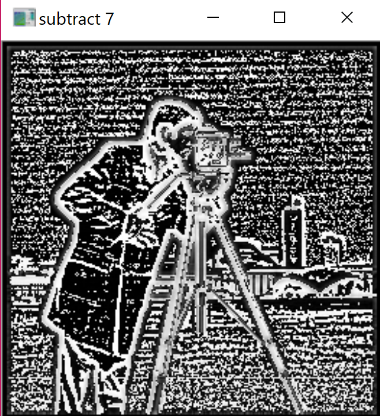
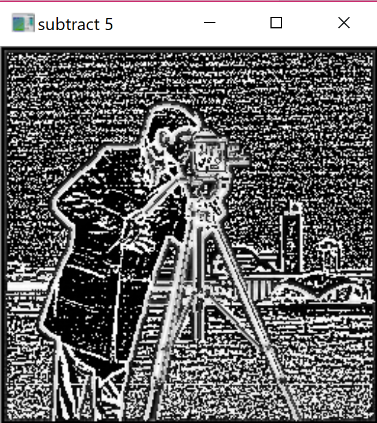
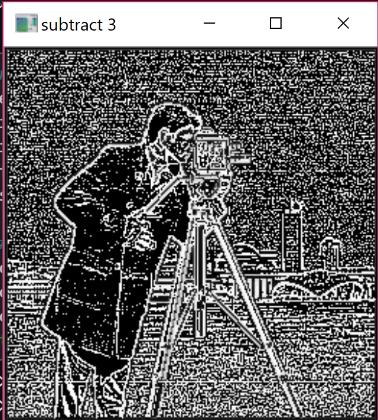
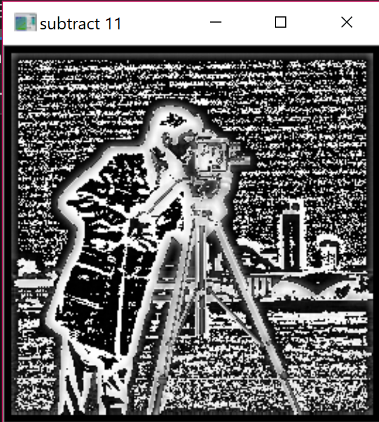




 To shift an image up, kernel has all zeroes except the top row and middle column value. To shift an image to right, kernel has all zeroes except most right column and middle row. To shift pixels more than one the siz of kernel should be enhanced.

To sharpen an image, first the original image should multiply by two, to do this the original image multiplied by two by using identity matrix which obtains all zeroes except the middle value is 2. Then we subtract the blurred image from 2 original image.

Subtracting blurred from only one original image will generate another type of image.



I write 2 different functions for filtering. In filter2d first I find right padded empty image. Then I traverse in the image’s row and column. In second for I called for the second function. In apply\_kernel function first I traversed in row and column of kernel. According to which filtering is requested first I found the item of the kernel in a certain index. Then I found the item in the padded image respect to kernel. I get the multiplication of image’s pixel value and kernel’s value and add them to summation and return the number sum. I placed the value of summation to middle of kernel respect to image. After I complete this process to all image, I replaced the changed image in new image list.

For salt and pepper filtering instead of multiplying kernel and image I found the median value of kernel respect to image and placed the median value in the middle of kernel respect to image. So, I could correct the intense shift to black or white without disrupting the image.