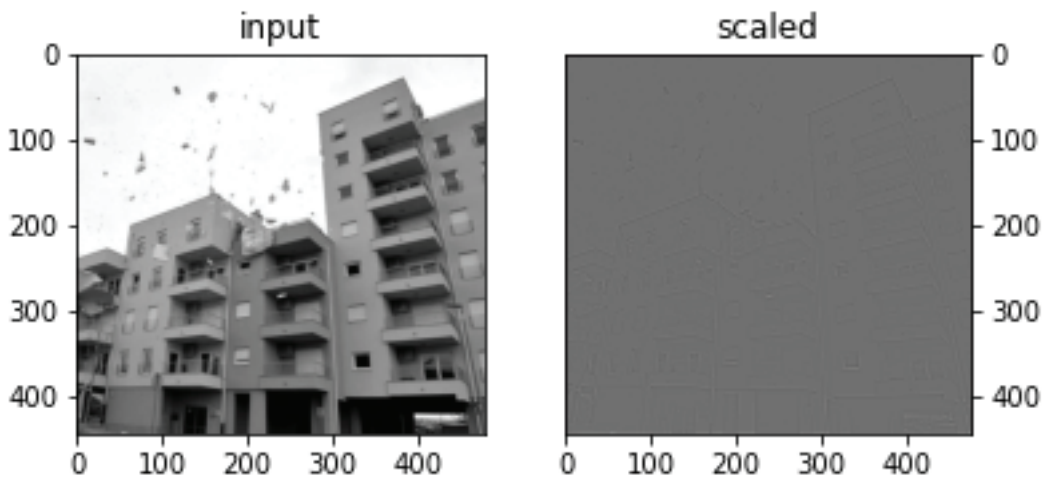
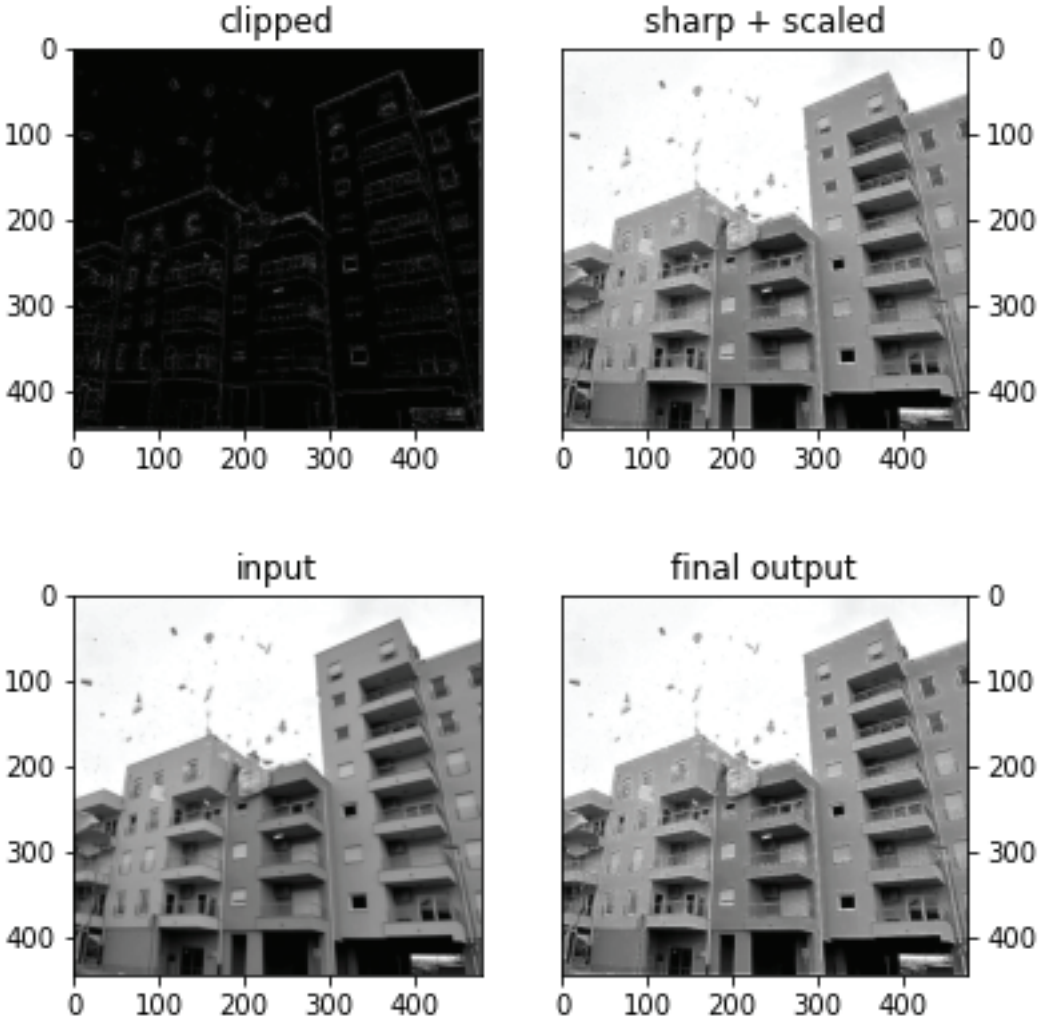
```python
import cv2 as cv
import matplotlib.pyplot as plt
import numpy as np
import math

# path to the input img
# path = "C:/Users/Raiyan/Desktop/img/03/Image-Processing-and-Computer-Vision-Lab/Lab 2/Average filter/Input.png"
path = 'C:/Users/Raiyan/Desktop/building.jpg'

# reading img + converting from BGR to GRAY
img = cv.imread(path)
img = cv.cvtColor(img, cv.COLOR_BGR2GRAY)

k_h = int(input("Enter kernel height: "))
k_w = k_h
k_size = (k_h,k_w)

# kernel with neg center value
kernel = np.array(([0,-1,0],
                   [-1,-4,-1],
                   [0,-1,0]), np.float32)

# img height
img_h = img.shape[0]
# img width
img_w = img.shape[1]
# kernel height // 2
a = kernel.shape[0] // 2
# kernel width // 2
b = kernel.shape[1] // 2

# empty op img
output = np.zeros((img_h,img_w), np.float32)

# conv
# visiting each pixel in the img
# m ta row img e ... for each row ...
for i in range(img_h):
    # n ta coln img e ... for each coln ...
    for j in range(img_w):
        # empty var for storing all the values
        values = []
        # visiting each pixel in the kernel
        # a ta row img e ... for each row ...
        for x in range(-a,a+1):
            # b ta coln img e ... for each coln ...
            for y in range(-b,b+1):
                if 0 <= i-x < img_h and 0 <= j-y < img_w:
                    output[i][j] += kernel[a+x][b+y] * img[i-x][j-y]
                else:
                    output[i][j] += 0

out_conv = output

# scaled
def scaled(image):
    g_m = image - image.min()
    g_s = 255*(g_m / g_m.max())
    return g_s.astype(np.float32)

scaled = scaled(out_conv)

plt.imshow(scaled, 'gray')
plt.title('scaled')
plt.show()

output = out_conv

# val capping or clipping from 0 - 255
for i in range(img_h):
      for j in range(img_w):
          if output[i][j] <0 :
              output[i][j] = 0
          elif output[i][j] >255 :
              output[i][j] = 255

```

```
clipped = output.astype(np.float32)


# center of kernel is (-)
sharpened = img - out_conv

output = sharpened
# sharpened + clipping from 0 - 255
for i in range(img_h):
      for j in range(img_w):
          if output[i][j] <0 :
              output[i][j] = 0
          elif output[i][j] >255 :
              output[i][j] = 255

sharpened_and_clipped = output.astype(np.float32)

# sharpened + scaled
g_m = sharpened - sharpened.min()
g_s = (g_m / g_m.max()) * 255
sharpened_and_scaled = g_s.astype(np.float32)


def show_images(images, image_title):
    # displaying multiple images side by side
    # https://stackoverflow.com/questions/41793931/plotting-images-side-by-side-using-matplotlib

    # err : was giving weird colormap due to diff in the mechanism of reading img of cv2 & matplotlib
    # https://stackoverflow.com/questions/3823752/display-image-as-grayscale-using-matplotlib
    # running this once in the code will ALWAYS give gray op
    plt.gray()

    no_of_imgs = len(images)
    f = plt.figure()
    for i in range(no_of_imgs):

        # Debug, plot figure
        axes = f.add_subplot(1, no_of_imgs, i + 1)
        # the last img will show y axis on the RHS instead of LHS(which is by default)

        if i==no_of_imgs-1:
            axes.yaxis.tick_right()

        plt.title(image_title[i])
        plt.imshow(images[i])
        # plt.rc('font', size=8)
    plt.show(block=True)


show_images([img,scaled], ['input', 'scaled'])
show_images([clipped, sharpened_and_scaled], ['clipped', 'sharp + scaled'])
show_images([img,sharpened_and_scaled], ['input', 'final output'])
```




clipped
sharp + scaled
input
final output
0
100
200
300
400