Image interpolation:

I was able to implement each process by iterating over each pixel. For shrinking, I just created a new image that was ½ as short and ½ as tall, and then the pixels in the image can be calculated by iterating over the pixels in the new image and getting the matching pixels from the original image. Nearest neighbour implementation was basically the same as shrinking, but reversed. Bilinear requires calculating the pixel value based on the 4 pixels closest to it, but can still be done by simply iterating over each pixel in the new image and calculating what the new value is. For bicubic I used the OpenCV function to resize the image.

Point operations

The negative was simple to implement, just flip the values based on the level range that is being used. The power law was also simple, just multiply each pixel value by the gamma value. Contrast stretching requires calculating the minimum and maximum values in the image, and using those values to set new high and low pixels, and increase the range of pixel values in the image

Histogram processing

To equalize a single image, first get the count of pixels at each value, then get the probability of each pixel value with: pixel value count / total pixel count. Then get the cumulative histogram by adding each probability to the previous cumulative value to get the cumulative probability for each value. After that you calculate the intensity of each by multiplying by the level range - 1, and can use these new intensities to get the new intensity of every pixel in the image. The histogram matching also uses cumulative histograms, but uses two images. Using the new intensity values for each image, find what new intensity value pixel level i is for image 1, then find the closest intensity value for image 2, and then the final value is whatever original intensity level that was for image 2.

For the image interpolations, nearest neighbour gave the most pixelated result, bilinear was still a bit pixelated, but better than nearest neighbour. Bicubic was smooth but looked quite blurry compared to the original.

For the power-law transformation I chose a gamma value of 1.4, I tested many values increasing by 0.1 from 0.5 to 2, and i found 1.4 was the best value