Experiment 7

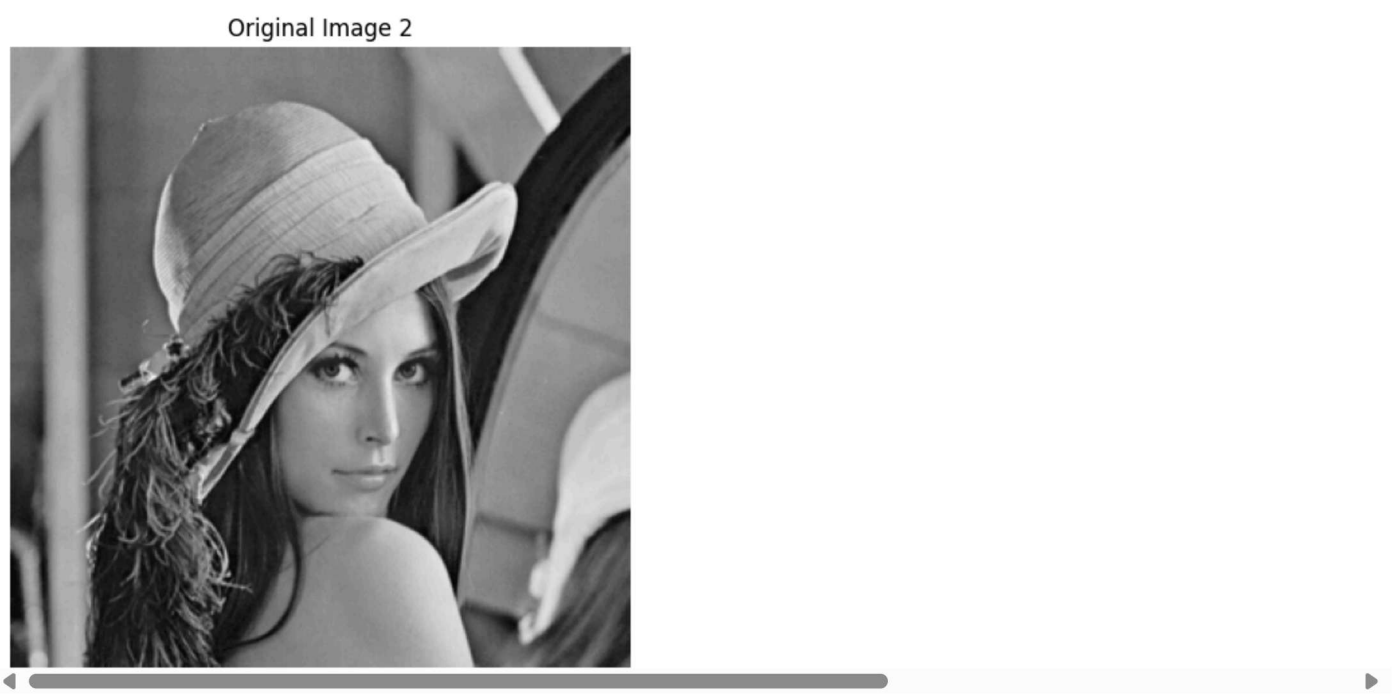
Write a program to improve the contrast of an image using histogram equalization. The prototype of the function is as below: histogram_equalisation(input_Image, no_of_bins); The function should return the enhanced image. Consider two low-contrast input images. Study the nature of the output image quality in each case by varying the number of bins.

```
import cv2
import numpy as np
import matplotlib.pyplot as plt


# Load the grayscale images
image1 = cv2.imread('OIP (2).jpeg', cv2.IMREAD_GRAYSCALE)
image2 = cv2.imread('Lenna.png', cv2.IMREAD_GRAYSCALE)


plt.figure(figsize=(8, 6))
plt.imshow(image1, cmap='gray')
plt.title('Original Image 1')
plt.axis('off')
plt.show()
```


```
plt.figure(figsize=(8, 6))
plt.imshow(image2, cmap='gray')
plt.title('Original Image 2')
plt.axis('off')
plt.show()
```




```
# Perform Histogram Equalization
def histogram_equalisation(input_Image, no_of_bins):
    hist, bins = np.histogram(input_Image.flatten(), no_of_bins, [0, 256])
    cdf = hist.cumsum()
    cdf_normalized = cdf * (255 / cdf[-1])
    equalized_image = np.interp(input_Image.flatten(), bins[:-1], cdf_normalized)
    return equalized_image.reshape(input_Image.shape).astype(np.uint8)


for bins in [32, 64, 128, 256]:
    enhanced_image1 = histogram_equalisation(image1, bins)
    enhanced_image2 = histogram_equalisation(image2, bins)
    plt.figure(figsize=(8, 6))

    plt.subplot(1, 2, 1)
    plt.imshow(enhanced_image1, cmap='gray')
    plt.title(f'Image 1 (Bins={bins})')
    plt.axis('on')

    plt.subplot(1, 2, 2)
    plt.imshow(enhanced_image2, cmap='gray')
    plt.title(f'Image 2 (Bins={bins})')
    plt.axis('on')

    plt.show()
```

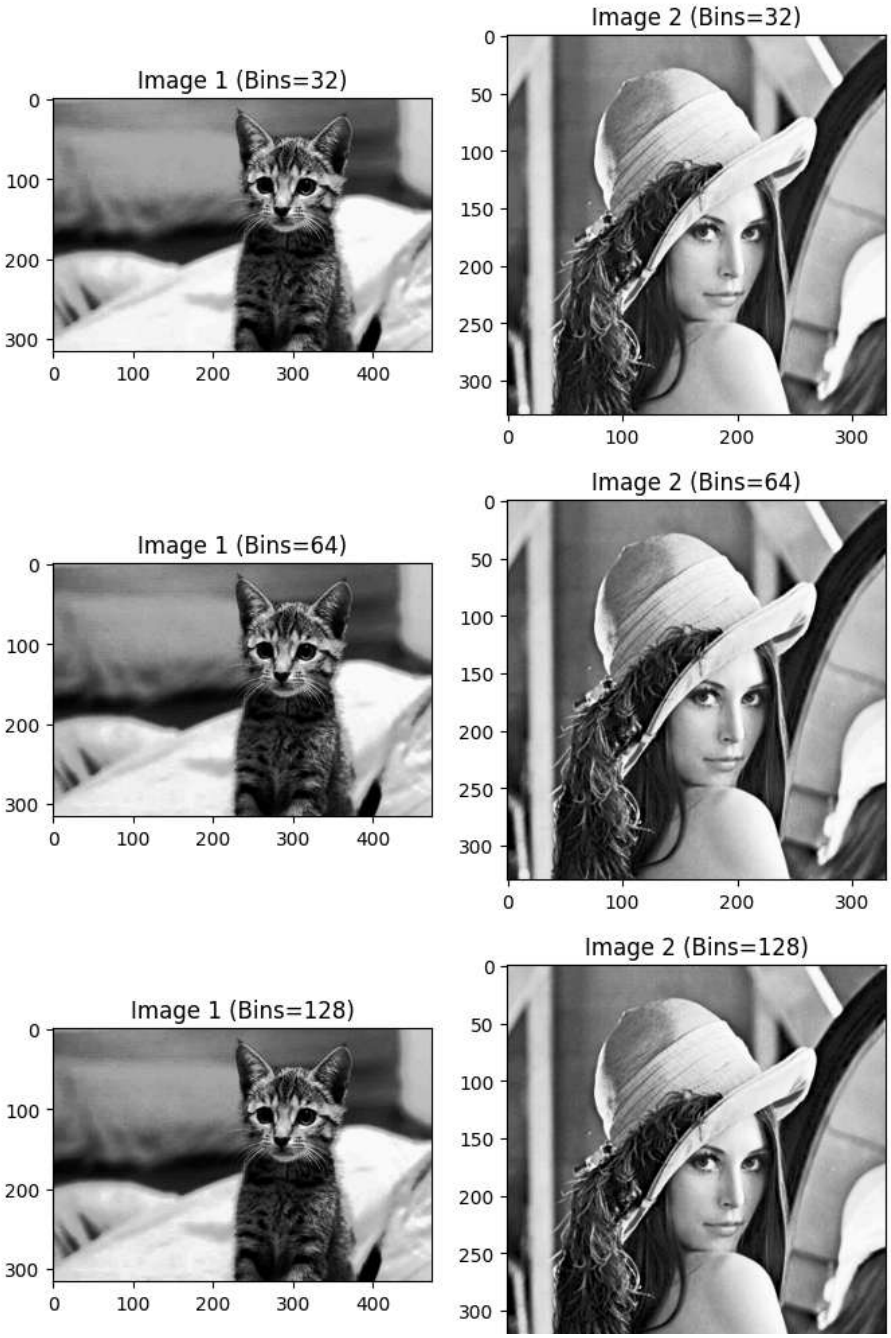


Image 1 (Bins=32)
Image 2 (Bins=32)
Image 1 (Bins=64)
Image 2 (Bins=64)
Image 1 (Bins=128)
Image 2 (Bins=128)