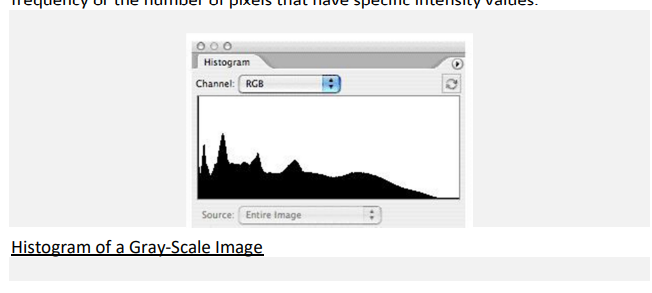
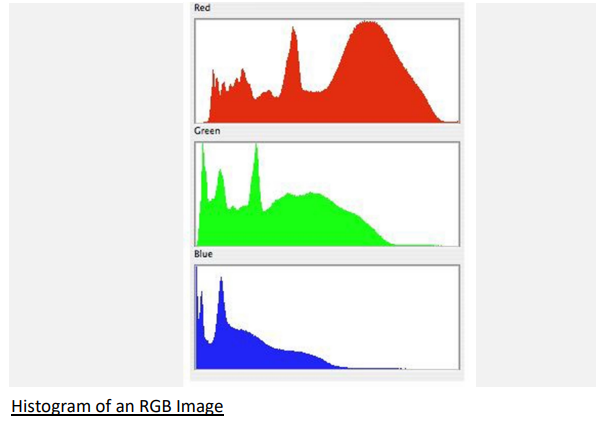


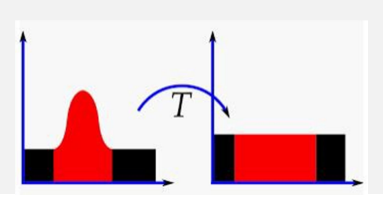
**Fig 1**

****

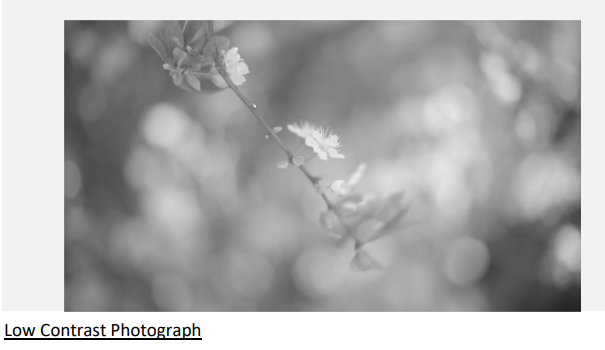
**Fig 2**

****

**Fig 3**

****

**Fig 4**

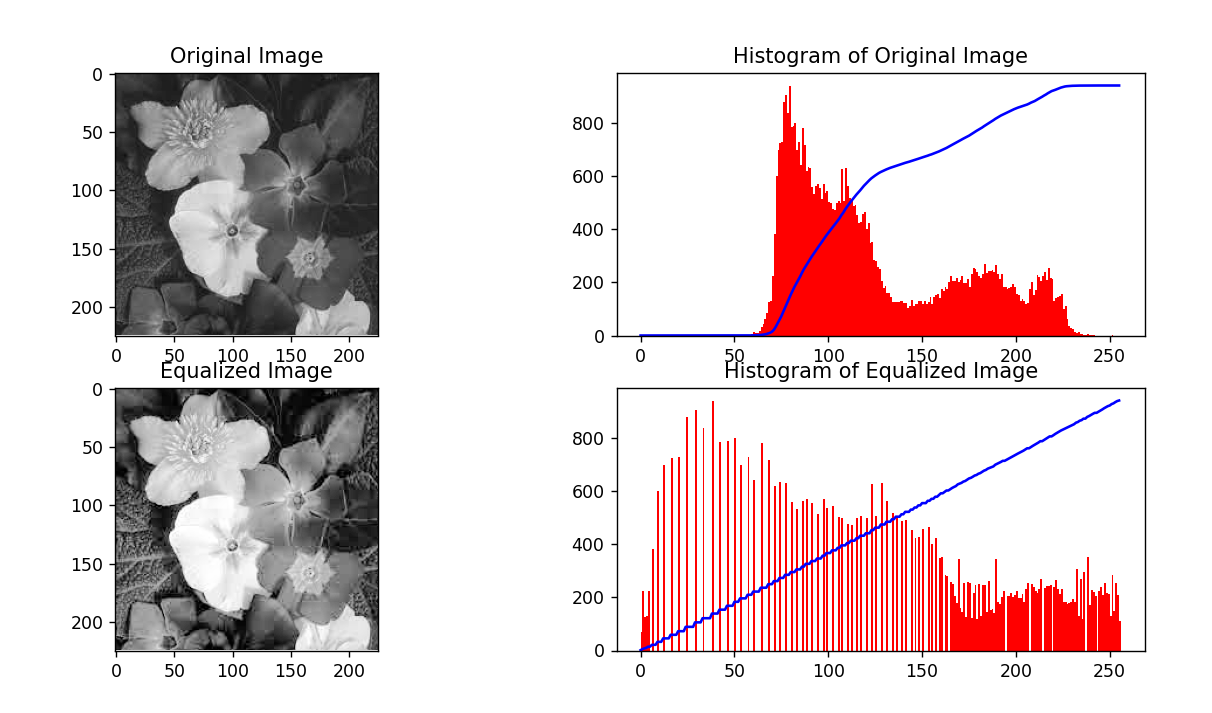
****

**Fig 5**

**Source Code :**

import cv2  
import numpy as np  
import matplotlib.pyplot as plt  
  
# Read the image  
image = cv2.imread('practical\_a2\_input.jpeg', cv2.IMREAD\_GRAYSCALE)  
  
# Calculate the histogram of the original image  
hist\_original, bins\_original = np.histogram(image.flatten(), 256, [0, 256])  
  
# Compute the CDF of the original histogram  
cdf\_original = hist\_original.cumsum()  
cdf\_normalized\_original = cdf\_original \* hist\_original.max() / cdf\_original.max()  
  
# Histogram equalization  
equ = cv2.equalizeHist(image)  
  
# Calculate the histogram of the enhanced image  
hist\_enhanced, bins\_enhanced = np.histogram(equ.flatten(), 256, [0, 256])  
  
# Compute the CDF of the enhanced histogram  
cdf\_enhanced = hist\_enhanced.cumsum()  
cdf\_normalized\_enhanced = cdf\_enhanced \* hist\_enhanced.max() / cdf\_enhanced.max()  
  
# Plotting  
plt.figure(figsize=(12, 6))  
  
plt.subplot(2, 2, 1)  
plt.imshow(image, cmap='gray')  
plt.title('Original Image')  
  
plt.subplot(2, 2, 2)  
plt.plot(cdf\_normalized\_original, color='b')  
plt.hist(image.flatten(), 256, [0, 256], color='r')  
plt.title('Histogram of Original Image')  
  
plt.subplot(2, 2, 3)  
plt.imshow(equ, cmap='gray')  
plt.title('Equalized Image')  
  
plt.subplot(2, 2, 4)  
plt.plot(cdf\_normalized\_enhanced, color='b')  
plt.hist(equ.flatten(), 256, [0, 256], color='r')  
plt.title('Histogram of Equalized Image')  
  
plt.show()

**Output :**

****