**3. Python KODLARI**

**3.1 Log Transform ve Power-Law kodları**

#Log Transform

import cv2  
import numpy as np  
import matplotlib.pyplot as plt  
  
# Gri seviye bir örnek görüntü yükleme  
inimage = cv2.imread('fourirespectrum.jpg', cv2.IMREAD\_GRAYSCALE)  
  
row, col = inimage.shape  
log\_transformed = np.zeros((row, col), dtype=np.uint8)  
# Logaritmik dönüşüm  
  
for x in range(row):  
 for y in range(col):  
 log\_transformed[x, y] = 1 \* (np.log(inimage[x, y] + 1))  
#c = 255 / np.log(1 + np.max(inimage))  
#log\_transformed = 2 \* (np.log(inimage + 1))  
  
# Görüntüleri gösterme  
plt.subplot(1, 2, 1), plt.imshow(inimage, cmap='gray'), plt.title('Original Image')  
plt.subplot(1, 2, 2), plt.imshow(log\_transformed, cmap='gray'), plt.title('Log Transformed Image')  
plt.show()

#PowerLaw

import cv2  
import numpy as np  
import matplotlib.pyplot as plt  
  
# Gri seviye bir örnek görüntü yükleme  
inimage = cv2.imread('fourirespectrum.jpg', cv2.IMREAD\_GRAYSCALE)  
  
gamma = 0.6 # Güç yasası parametresi  
  
# Power-Law Transformation işlemi  
power\_law\_transformed = np.power(inimage, gamma)  
  
# Görüntüleri gösterme  
plt.subplot(1, 2, 1), plt.imshow(inimage, cmap='gray'), plt.title('Original Image')  
plt.subplot(1, 2, 2), plt.imshow(power\_law\_transformed, cmap='gray'), plt.title('Kuvvet Alınmış Image')  
plt.show()

**3.2 Gray level Slicing Kodu**

#graylevelSlicing

import numpy as np  
import matplotlib.pyplot as plt  
import cv2  
  
# Görüntüyü oku  
inimage = cv2.imread("siginak.jpg", cv2.IMREAD\_GRAYSCALE)  
  
# Eşikleme fonksiyonu  
def esikleme(goruntu):  
 h, w = goruntu.shape  
 esikleme\_zeros = np.zeros((h, w), dtype=np.uint8)  
 for i in range(h):  
 for j in range(w):  
 if goruntu[i, j] >= 81: #eşik değeri  
 esikleme\_zeros[i, j] = 255  
 else:  
 esikleme\_zeros[i, j] = 0  
 return esikleme\_zeros  
  
# fonksiyon çağırılıyor  
cikis\_goruntu = esikleme(inimage)  
  
# Görüntüleri göster  
plt.subplot(1, 2, 1)  
plt.imshow(inimage, cmap='gray')  
plt.title("Orijinal Görüntü")  
  
plt.subplot(1, 2, 2)  
plt.imshow(cikis\_goruntu, cmap='gray')  
plt.title("Eşiklenmiş Görüntü")  
plt.show()

**3.3 Bitplane Slicing Kodu**

#BitplaneSlicing

import cv2  
import numpy as np  
import matplotlib.pyplot as plt  
  
# Gri seviye bir örnek görüntü yükleme  
inimage = cv2.imread('siginak.jpg', cv2.IMREAD\_GRAYSCALE)  
  
# Görüntü boyutları  
height, width = inimage.shape  
print(height, width )  
bit\_planes = np.zeros((height, width,8))  
# Bit Plane Slicing işlemi  
for kaydirma in range(8):  
 mask = 1 << kaydirma  
 for i in range(height):  
 for ii in range(width):  
 bit\_value = (inimage[i, ii] & mask) >> kaydirma  
 bit\_planes[i,ii,kaydirma] = bit\_value

# Görüntüleri gösterme  
plt.figure(figsize=(12, 6))  
plt.subplot(2, 4, 1), plt.imshow(inimage, cmap='gray'), plt.title('Original Image')  
  
for i in range(7, -1, -1):  
 plt.subplot(2, 4, 8 - i), plt.imshow(bit\_planes[:,:,i], cmap='gray'), plt.title(f'Bit Plane {i}')  
  
plt.show()

**3.4 Histogram Eşitleme, Ortalama değer ve Standart sapmaları**

#

import cv2  
import matplotlib.pyplot as plt  
import numpy as np  
  
# Görüntüyü oku  
inimage = cv2.imread('monalisa.jpg', cv2.IMREAD\_GRAYSCALE)  
height, width = inimage.shape  
  
esitlenmis\_image = np.zeros((height, width), dtype=np.uint8)  
  
# Histogramı hesapla  
hist = cv2.calcHist([inimage], [0], None, [256], [0, 256])  
new\_deger = np.zeros((256, 1))  
  
for i in range(256):  
 new\_deger[i] = sum(hist[:i]) \* 255 / (height \* width)  
  
for i in range(height):  
 for ii in range(width):  
 esitlenmis\_image[i, ii] = new\_deger[inimage[i, ii]]  
  
# Ortalama hesaplama  
def ortalama(goruntu):  
 toplam = 0  
 h, w = goruntu.shape  
 for i in range(h):  
 for j in range(w):  
 toplam += goruntu[i, j]  
 return toplam / (h \* w)  
  
# Standart sapma hesaplama  
def standart\_sapma(goruntu, ort):  
 toplam = 0  
 h, w = goruntu.shape  
 for i in range(h):  
 for j in range(w):  
 fark = goruntu[i, j] - ort  
 toplam += fark \* fark  
 return (toplam / (h \* w)) \*\* 0.5  
  
# Giriş ve çıkış için ortalama ve standart sapma hesapla  
giris\_ort = ortalama(inimage)  
giris\_std = standart\_sapma(inimage, giris\_ort)  
  
cikis\_ort = ortalama(esitlenmis\_image)  
cikis\_std = standart\_sapma(esitlenmis\_image, cikis\_ort)  
  
# Sonuçları yazdır  
print("Giriş Görüntüsü Ortalama:", giris\_ort)  
print("Giriş Görüntüsü Standart Sapma:", giris\_std)  
print("Çıkış Görüntüsü Ortalama:", cikis\_ort)  
print("Çıkış Görüntüsü Standart Sapma:", cikis\_std)  
  
# Görüntüleri göster  
plt.subplot(1, 2, 1), plt.imshow(inimage, cmap='gray'), plt.title('Orijinal Görüntü')  
plt.subplot(1, 2, 2), plt.imshow(esitlenmis\_image, cmap='gray'), plt.title('Eşitlenmiş Görüntü')  
plt.show()  
  
hist1 = cv2.calcHist([esitlenmis\_image], [0], None, [256], [0, 256])  
  
plt.plot(hist)  
plt.title('Giriş Görüntü Histogramı')  
plt.xlabel('Piksel Değeri')  
plt.ylabel('Piksel Sayısı')  
plt.show()  
  
plt.plot(hist1)  
plt.title('Çıkış Görüntü Histogramı')  
plt.xlabel('Piksel Değeri')  
plt.ylabel('Piksel Sayısı')  
plt.show()

**3.5 Contrast stretching**

#kontrast germe  
  
import cv2  
import numpy as np  
import matplotlib.pyplot as plt  
  
# Gri seviye bir örnek görüntü yükleme  
inimage = cv2.imread('monalisa.jpg', cv2.IMREAD\_GRAYSCALE)  
  
def KontrastGerme(image):  
  
 row, col = inimage.shape  
 # Predefine the output image  
 normalized\_image = np.zeros((row, col), dtype=np.uint8)  
  
 for y in range(row):  
 for x in range(col):  
 normalized\_image[y, x] = (inimage[y,x] - np.min(inimage))\*(255/(np.max(inimage)- np.min(inimage)))  
  
 return normalized\_image  
  
Outimage = KontrastGerme(inimage)  
  
# Görüntüleri gösterme  
plt.subplot(1, 2, 1), plt.imshow(inimage, cmap='gray'), plt.title('Original Image')  
plt.subplot(1, 2, 2), plt.imshow(Outimage, cmap='gray'), plt.title('Kontrast Gerilmiş Image')  
plt.show()