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

# Upload an image in Colab
from google.colab import files
uploaded = files.upload()

# Read Image
image_path = list(uploaded.keys())[0]
color_img = cv2.imread(image_path)
color_img = cv2.cvtColor(color_img, cv2.COLOR_BGR2RGB)

# Convert to Grayscale
gray_img = cv2.cvtColor(color_img, cv2.COLOR_RGB2GRAY)

# Function to compute and display histogram
def plot_histogram(image, title, method='count'):
    hist = cv2.calcHist([image], [0], None, [256], [0, 256])
    if method == 'probability':
        hist /= hist.sum()

    plt.figure(figsize=(8, 4))
    plt.plot(hist, color='black')
    plt.title(title)
    plt.xlabel('Gray Level')
    plt.ylabel('Pixel Count' if method == 'count' else 'Probability')
    plt.show()

# Compute and display histograms
plot_histogram(gray_img, "Grayscale Histogram (Count)", 'count')
plot_histogram(gray_img, "Grayscale Histogram (Probability)", 'probability')

# Color Histograms
colors = ('r', 'g', 'b')
plt.figure(figsize=(8, 4))
for i, color in enumerate(colors):
    hist = cv2.calcHist([color_img], [i], None, [256], [0, 256])
    plt.plot(hist, color=color)
plt.title("Color Histogram")
plt.xlabel("Intensity")
plt.ylabel("Pixel Count")
plt.show()

# Bright and Dark images
bright_img = cv2.add(gray_img, 50)
dark_img = cv2.subtract(gray_img, 50)

plot_histogram(bright_img, "Bright Image Histogram", 'count')
plot_histogram(dark_img, "Dark Image Histogram", 'count')

# Histogram Equalization
eq_img = cv2.equalizeHist(gray_img)
plot_histogram(eq_img, "Histogram Equalized Image", 'count')

# Display images
fig, axes = plt.subplots(1, 4, figsize=(15, 5))
axes[0].imshow(gray_img, cmap='gray'); axes[0].set_title("Original")
axes[1].imshow(bright_img, cmap='gray'); axes[1].set_title("Bright")
axes[2].imshow(dark_img, cmap='gray'); axes[2].set_title("Dark")
axes[3].imshow(eq_img, cmap='gray'); axes[3].set_title("Equalized")
for ax in axes: ax.axis('off')
plt.show()

# ------------------------------------------------
# TASK 2: FOURIER TRANSFORM (FFT) & IFFT
# ------------------------------------------------

def compute_fft(image):
    dft = np.fft.fft2(image)
    dft_shift = np.fft.fftshift(dft)
    magnitude_spectrum = 20 * np.log(np.abs(dft_shift) + 1)
    return dft_shift, magnitude_spectrum

def compute_ifft(dft_shift):
    dft_ishift = np.fft.ifftshift(dft_shift)
    img_reconstructed = np.fft.ifft2(dft_ishift)
    return np.abs(img_reconstructed)

# Compute FFT and Magnitude Spectrum
fft_shift, magnitude_spectrum = compute_fft(gray_img)
```

```
# Compute Inverse FFT (IFFT)
reconstructed_img = compute_ifft(fft_shift)

# Display results
fig, axes = plt.subplots(1, 3, figsize=(15, 5))
axes[0].imshow(gray_img, cmap='gray'); axes[0].set_title("Original Image")
axes[1].imshow(magnitude_spectrum, cmap='gray'); axes[1].set_title("Magnitude Spectrum")
axes[2].imshow(reconstructed_img, cmap='gray'); axes[2].set_title("Reconstructed Image")
for ax in axes: ax.axis('off')
plt.show()

# ------------------------------------------------
# TASK 3: VERIFY ROTATION PROPERTY OF FOURIER TRANSFORM
# ------------------------------------------------
# Rotate image by 45 degrees
(h, w) = gray_img.shape[:2]
center = (w // 2, h // 2)
M = cv2.getRotationMatrix2D(center, 45, 1.0)
rotated_img = cv2.warpAffine(gray_img, M, (w, h))

# Compute FFT for rotated image
fft_rotated_shift, magnitude_spectrum_rotated = compute_fft(rotated_img)

# Display results
fig, axes = plt.subplots(1, 3, figsize=(15, 5))
axes[0].imshow(rotated_img, cmap='gray'); axes[0].set_title("Rotated Image (45°)")
axes[1].imshow(magnitude_spectrum_rotated, cmap='gray'); axes[1].set_title("FFT Magnitude Spectrum (Rotated)")
axes[2].imshow(magnitude_spectrum, cmap='gray'); axes[2].set_title("FFT Magnitude Spectrum (Original)")
for ax in axes: ax.axis('off')
plt.show()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Screenshot 2025-02-20 112852.png to Screenshot 2025-02-20 112852 (1).png
```