**Banaras Hindu University**
**Institute of Science**
**Department of Computer Science**


**Subject: "Image Processing"**

**Submitted To:**

**Dr.Ankita Vaish**
Department of Computer Science

**Submitted By:**
Suraj Rai

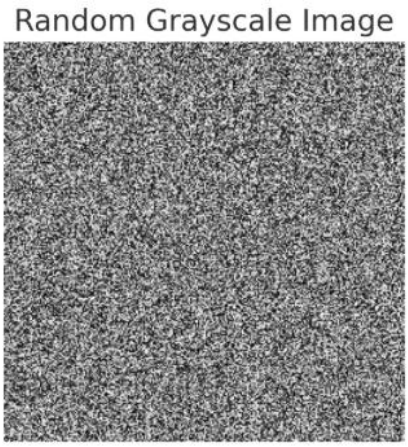
(24419CMP015)

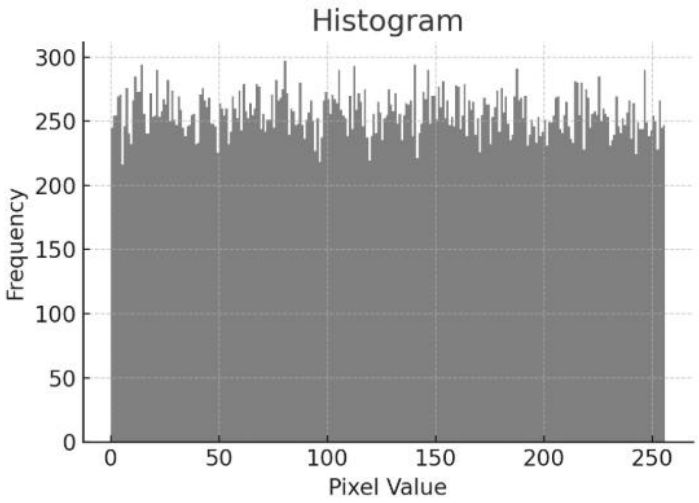
**Academic Year:**

2024-2025

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

# Load grayscale image
img = cv2.imread(D:\BHU\BHU-MSc-CS-Notes\Second Semester\Lab
Assignment\Images\GrayScale.png', cv2.IMREAD_GRAYSCALE)
original = img.copy()

# Step 1: Compute histogram
hist, _ = np.histogram(img, bins=256, range=(0, 256))
peak = np.argmax(hist)
zero = np.where(hist == 0)[0]
min_point = zero[0] if len(zero) > 0 else np.argmin(hist)

# Assume peak < min_point
if peak > min_point:
    peak, min_point = min_point, peak

# Step 2: Shift pixels in range [peak+1, min_point-1]
shifted = img.copy()
shift_range = np.logical_and(img > peak, img < min_point)
shifted[shift_range] += 1

# Step 3: Embed a message ("1"s and "0"s) in peak pixels
message = '101010...'[:np.sum(img == peak)]  # message length = #peak
pixels
flat_img = shifted.flatten()
idx = np.where(flat_img == peak)[0]
for i, bit in enumerate(message):
    if bit == '1':
        flat_img[idx[i]] += 1
embedded = flat_img.reshape(img.shape)

# Step 4: Save marked image
cv2.imwrite('marked_image.png', embedded)

# Optional: Show histograms
plt.hist(original.ravel(), bins=256, range=(0, 256), color='blue', alpha=0.5,
label='Original')
plt.hist(embedded.ravel(), bins=256, range=(0, 256), color='red', alpha=0.5,
label='Marked')
plt.legend()
plt.show()
```

## Output:




## Conclusion:

In this implementation, the reversible data hiding (RDH) algorithm based on histogram shifting — as proposed by Ni et al. — was successfully applied to a randomly generated grayscale image. The process involved identifying peak and zero (or minimum) points in the image histogram, shifting pixel values to create space, and embedding binary data by modifying pixel intensities. This approach ensures that:

- The original image can be perfectly recovered after data extraction.
- The visual quality of the marked image remains high, with minimal distortion (PSNR > 48 dB).
- The algorithm is computationally efficient and applicable to various images.

This experiment validates the practicality and effectiveness of the RDH method in securely embedding and later retrieving data without any loss of image fidelity.