**CODE**

import numpy as np

import matplotlib.pyplot as plt

import random

import cv2

from google.colab import files

def SaltAndPepper(image, density):

    output = np.zeros(image.shape, np.uint8)

    threshhold = 1 - density

    for i in range(image.shape[0]):

        for j in range(image.shape[1]):

            possibility = random.random()

            if possibility < density:

                output[i][j] = 0

            elif possibility > threshhold:

                output[i][j] = 255

            else:

                output[i][j] = image[i][j]

    return output

def MedianFilter(image):

    output = np.zeros(image.shape, np.uint8)

    filter\_array = [image[0][0]] \* 9

    for j in range(1, image.shape[0]-1):

        for i in range(1, image.shape[1]-1):

            filter\_array[0] = image[j-1, i-1]

            filter\_array[1] = image[j, i-1]

            filter\_array[2] = image[j+1, i-1]

            filter\_array[3] = image[j-1, i]

            filter\_array[4] = image[j, i]

            filter\_array[5] = image[j+1, i]

            filter\_array[6] = image[j-1, i+1]

            filter\_array[7] = image[j, i+1]

            filter\_array[8] = image[j+1, i+1]

            filter\_array.sort()

            output[j][i] = filter\_array[4]

    return output

def histogram\_equalization(image):

    # Check if image is grayscale (1 channel)

    if len(image.shape) == 2:

        # Convert grayscale image to BGR (3 channels) for histogram equalization

        image = cv2.cvtColor(image, cv2.COLOR\_GRAY2BGR)

    # Convert to YUV color space

    img\_yuv = cv2.cvtColor(image, cv2.COLOR\_BGR2YUV)

    # Apply histogram equalization to the Y channel

    img\_yuv[:,:,0] = cv2.equalizeHist(img\_yuv[:,:,0])

    # Convert back to BGR color space

    img\_output = cv2.cvtColor(img\_yuv, cv2.COLOR\_YUV2BGR)

    return img\_output

def main():

    uploaded = files.upload()

    # Extract the uploaded image data

    for name, data in uploaded.items():

        image\_array = cv2.imdecode(np.frombuffer(data, np.uint8), -1)

        # Apply Salt and Pepper noise

        noise\_image = SaltAndPepper(image\_array, 0.01)

        # Apply Median Filter

        median\_filtered\_image = MedianFilter(noise\_image)

        # Apply Histogram Equalization

        equalized\_image = histogram\_equalization(image\_array)

        # Display images

        plt.figure(figsize=(16, 8))

        plt.subplot(2, 2, 1)

        plt.title('Original Image')

        plt.imshow(cv2.cvtColor(image\_array, cv2.COLOR\_BGR2RGB))

        plt.subplot(2, 2, 2)

        plt.title('Noisy Image')

        plt.imshow(cv2.cvtColor(noise\_image, cv2.COLOR\_BGR2RGB))

        plt.subplot(2, 2, 3)

        plt.title('Median Filtered Image')

        plt.imshow(cv2.cvtColor(median\_filtered\_image, cv2.COLOR\_BGR2RGB))

        plt.subplot(2, 2, 4)

        plt.title('Equalized Image')

        plt.imshow(cv2.cvtColor(equalized\_image, cv2.COLOR\_BGR2RGB))

        plt.show()

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**Code Link on Google Colab:**

<https://colab.research.google.com/drive/1M2jwAxVJVw1DybYMcwHu0-ROo4jTHvzY?usp=sharing>

1. **Input Image:**

