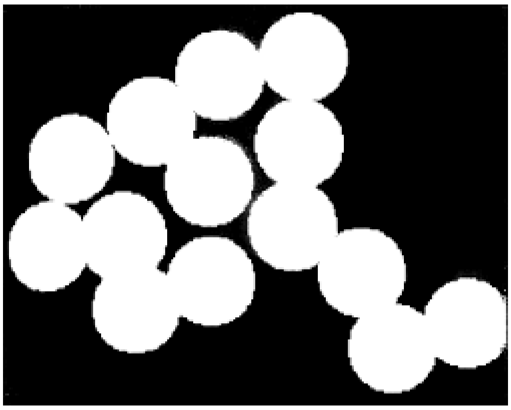
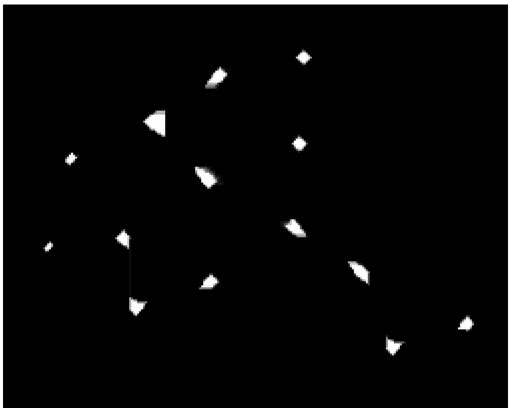
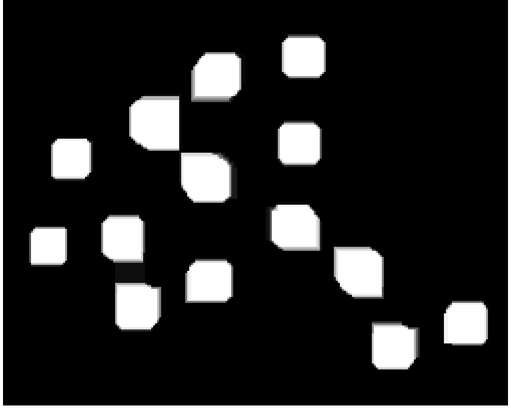
```
import cv2
import numpy as np
from matplotlib import pyplot as plt
from google.colab.patches import cv2_imshow
```

```
image = cv2.imread('img.png',0)
cv2_imshow(image)
```



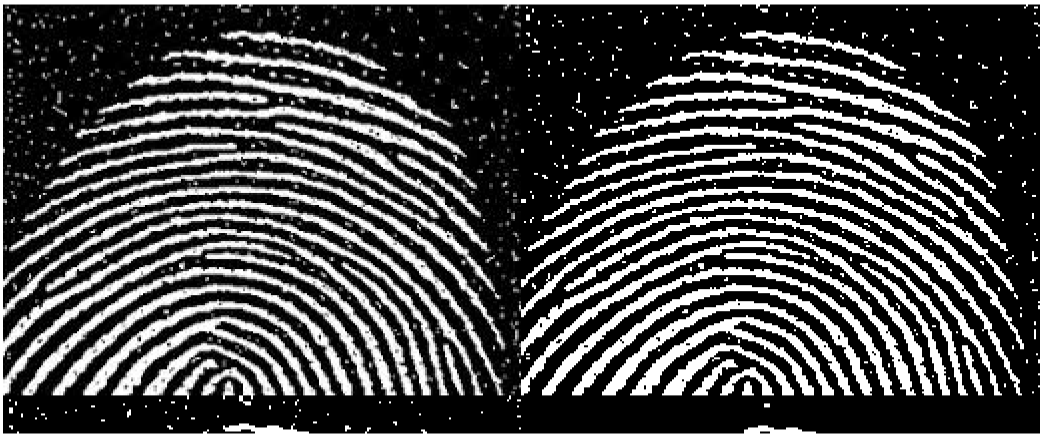
```
kernel = np.ones((15,15),np.uint8)
erosion = cv2.erode(image,kernel,iterations = 2)
dialtion = cv2.dilate(erosion,kernel,iterations = 1)
cv2_imshow( erosion)
cv2_imshow( dialtion)
```





```
# Lab task 2
image1 = cv2.imread('fingerprint.jpeg',0)
cv2_imshow(image1)
```

```
#Lab task 3

# Load the image
img = cv2.imread('fingerprint.jpeg', 0)  # Read as grayscale

# Apply thresholding to create a binary image
ret, thresh_img = cv2.threshold(img, 127, 255, cv2.THRESH_BINARY)

# Perform morphological closing to connect broken ridges and fill small gaps
kernel = np.ones((2, 2), np.uint8)  # Structuring element similar to ridge width
closed_img = cv2.morphologyEx(thresh_img, cv2.MORPH_CLOSE, kernel)

# Apply morphological opening to remove small protrusions or noise pixels
smaller_kernel = np.ones((2, 3), np.uint8)  # Smaller kernel for minimum gap size
opened_img = cv2.morphologyEx(closed_img, cv2.MORPH_OPEN, smaller_kernel)

# Visualize the results
# cv2_imshow(img)  # Original Image
# cv2_imshow(thresh_img)  # Thresholded Image
# cv2_imshow(closed_img)  # Closed Image
# cv2_imshow(opened_img)  # Opened Image (Final)
h1=np.hstack([img,thresh_img])
h2=np.hstack([closed_img,opened_img])
v=np.vstack([h1,h2])
cv2_imshow(v)
```



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

# Load the image (Replace 'your_image_path.jpg' with the actual path or image file)
image_path = 'head.jpg'
image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)

# Check if the image is loaded successfully
if image is None:
    print(f"Error: Unable to load the image from {image_path}")
else:
    # Grayscale dilation with a 3x3 kernel
    dilated_image = cv2.dilate(image, np.ones((3, 3), np.uint8), iterations=1)

    # Grayscale erosion with a 3x3 kernel
    eroded_image = cv2.erode(image, np.ones((3, 3), np.uint8), iterations=1)

    # Morphological gradient (difference between dilation and erosion)
    morph_gradient = cv2.subtract(dilated_image, eroded_image)

    # Display original, dilated, eroded, and gradient images
    plt.figure(figsize=(12, 4))

    plt.subplot(1, 4, 1), plt.imshow(image, cmap='gray'), plt.title('Original')
    plt.subplot(1, 4, 2), plt.imshow(dilated_image, cmap='gray'), plt.title('Dilated')
    plt.subplot(1, 4, 3), plt.imshow(eroded_image, cmap='gray'), plt.title('Eroded')
    plt.subplot(1, 4, 4), plt.imshow(morph_gradient, cmap='gray'), plt.title('Morpholog

    plt.show()
```