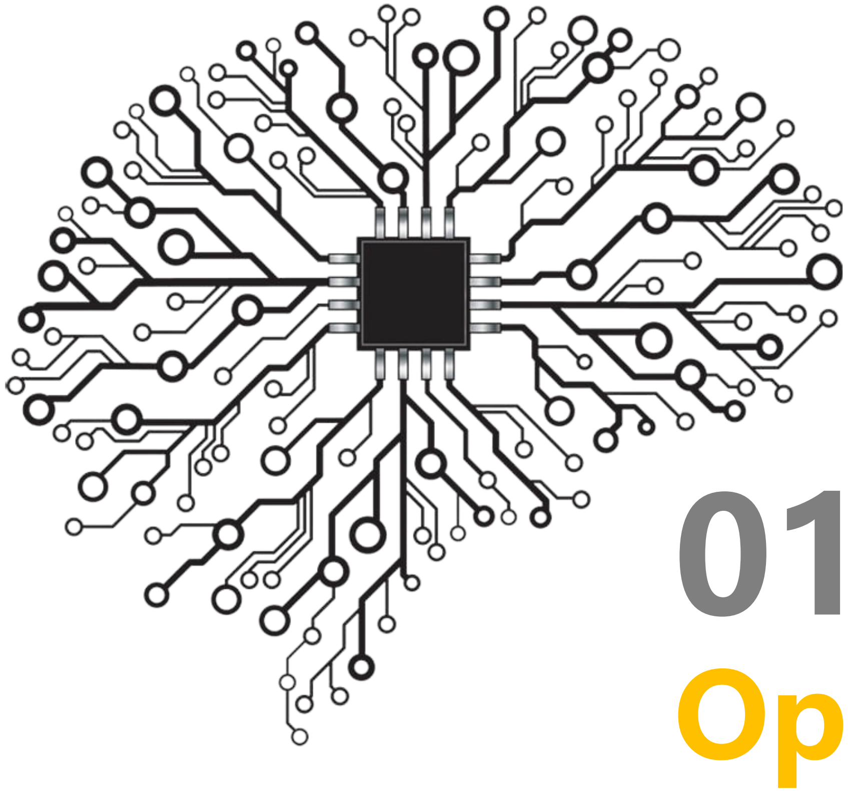
影像特徵的研究

目標任務

利用影像特徵進行影像處理操作。先對影像做(1)**二值化**(Image Binarization)，(2)**標籤化**(Labeling)並計算特徵參數，再由特徵參數分割影像

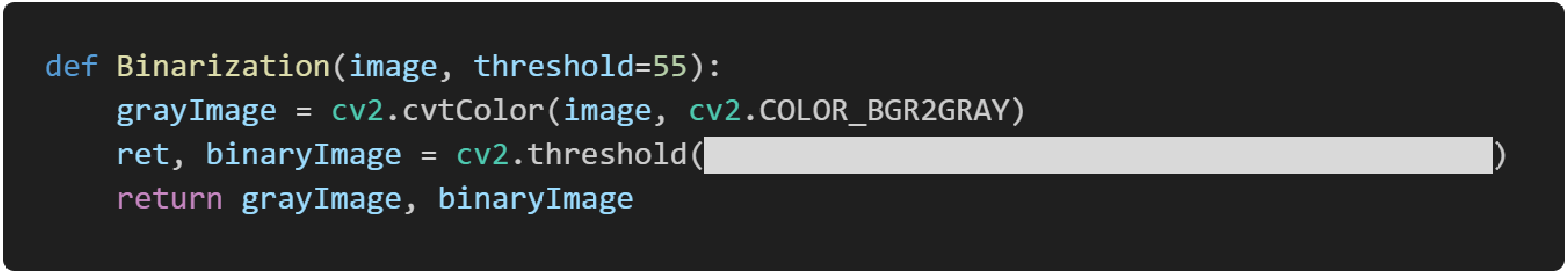
1. 先對目標影像進行**二值化**，門檻值設為**55**
2. 影像**標籤化**並計算特徵參數
3. 將**面積小於100**的標籤連通元件去除
4. 將剩餘的標籤連通元件透過輪廓偵測計算**周長**和**真圓度**
5. 將**真圓度小於0.5**的標籤連通元件去除，為了留下圓形的物體



01

OpenCV：Segmentation

影像二值化

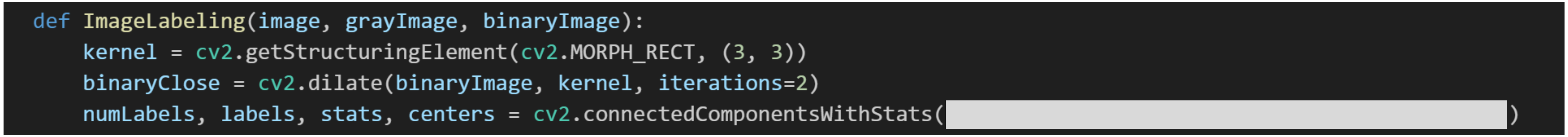


```
def Binarization(image, threshold=55):
    grayImage = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    ret, binaryImage = cv2.threshold(                              )
    return grayImage, binaryImage
```

- ✓ **門檻值：** 55
- ✓ **Threshold Type：** cv2.THRESH_BINARY

影像標籤化

- ■ 將影像進行**標籤化**，並且取得連通元件
- ■ 影像標籤化
  - □ *cv2.connectedComponentsWithStats(image[, labels[, stats[, centroids[, connectivity[, ltype]]]]])*

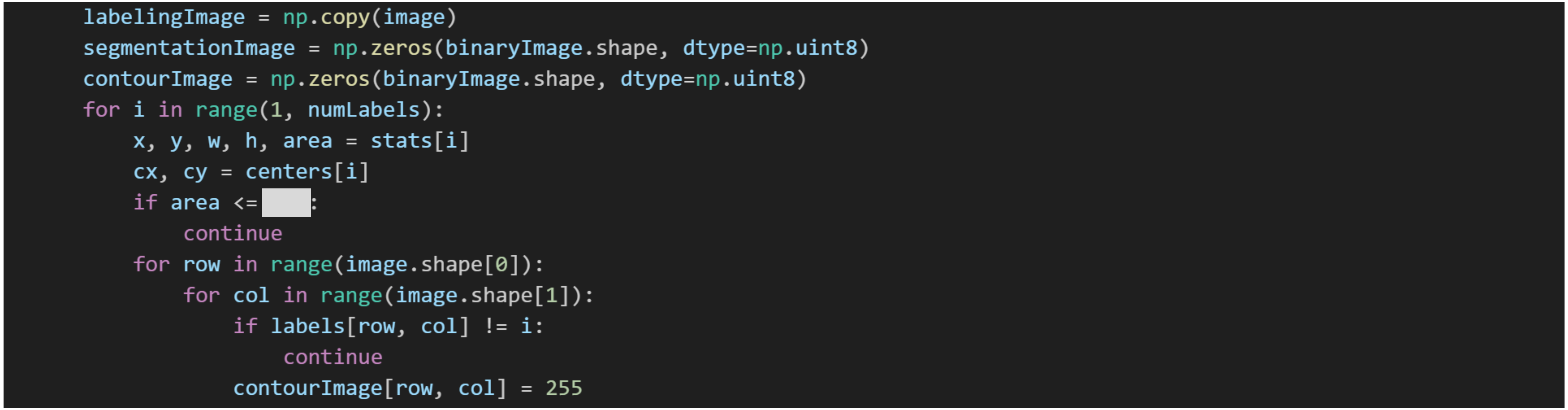


```
def ImageLabeling(image, grayImage, binaryImage):
    kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (3, 3))
    binaryClose = cv2.dilate(binaryImage, kernel, iterations=2)
    numLabels, labels, stats, centers = cv2.connectedComponentsWithStats(                    )
```

- ✓ **Connectivity：** 8
- ✓ **ltype：** cv2.CV_32S

影像標籤化

- 計算特徵參數，並且去除面積小於100的連通元件



```
labelingImage = np.copy(image)
segmentationImage = np.zeros(binaryImage.shape, dtype=np.uint8)
contourImage = np.zeros(binaryImage.shape, dtype=np.uint8)
for i in range(1, numLabels):
    x, y, w, h, area = stats[i]
    cx, cy = centers[i]
    if area <=    :
        continue
    for row in range(image.shape[0]):
        for col in range(image.shape[1]):
            if labels[row, col] != i:
                continue
            contourImage[row, col] = 255
```

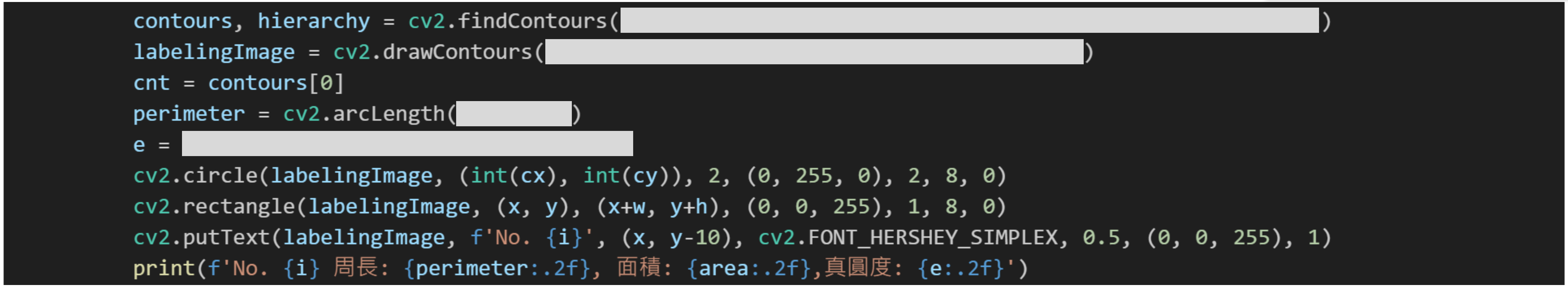
影像標籤化

- ■ **計算剩餘標籤連通元件的周長和真圓度**
  - □ **取得輪廓資訊用:**
    - ✓ ***cv2.findContours(image, mode, method[, contours[, hierarchy[, offset]]])***
  - □ **取得物體輪廓資訊用(預設物體為白色):**
    - ✓ ***cv2.drawContours(image, contours, contourIdx, color[, thickness[, lineType[, hierarchy[, maxLevel[, offset]]]]])***
  - □ **計算周長:**
    - ✓ ***cv2.arclength(curve, closed)***
  - □ **計算真圓度用:**
    - ✓ ***4 * numpy.pi * 面積 / (周長)^2***

- ✓ **image**：contourImage
- ✓ **mode**：cv2.RETR_EXTERNAL
- ✓ **method**：cv2.CHAIN_APPROX_SIMPLE

- ✓ **Curve**：cnt
- ✓ **Closed**：True

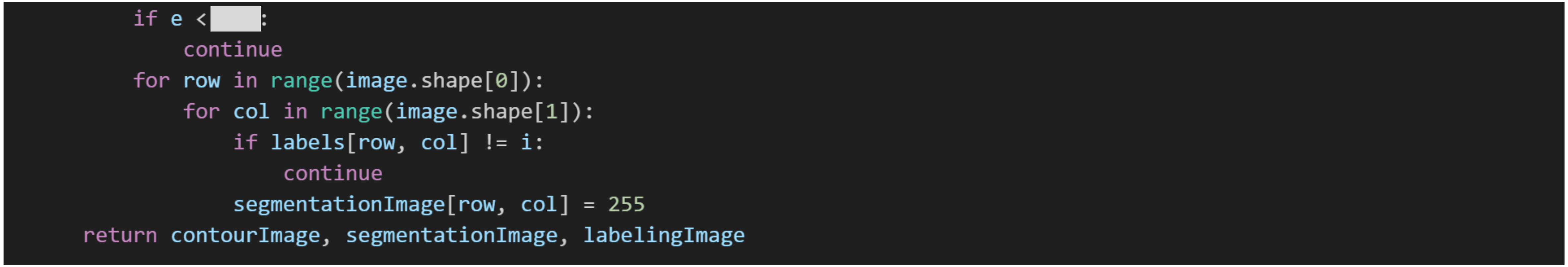
- ✓ **image**：labelingImage
- ✓ **contours**：contours
- ✓ **contourIdx**：-1
- ✓ **color**：(255, 0, 0)
- ✓ **thickness**：1



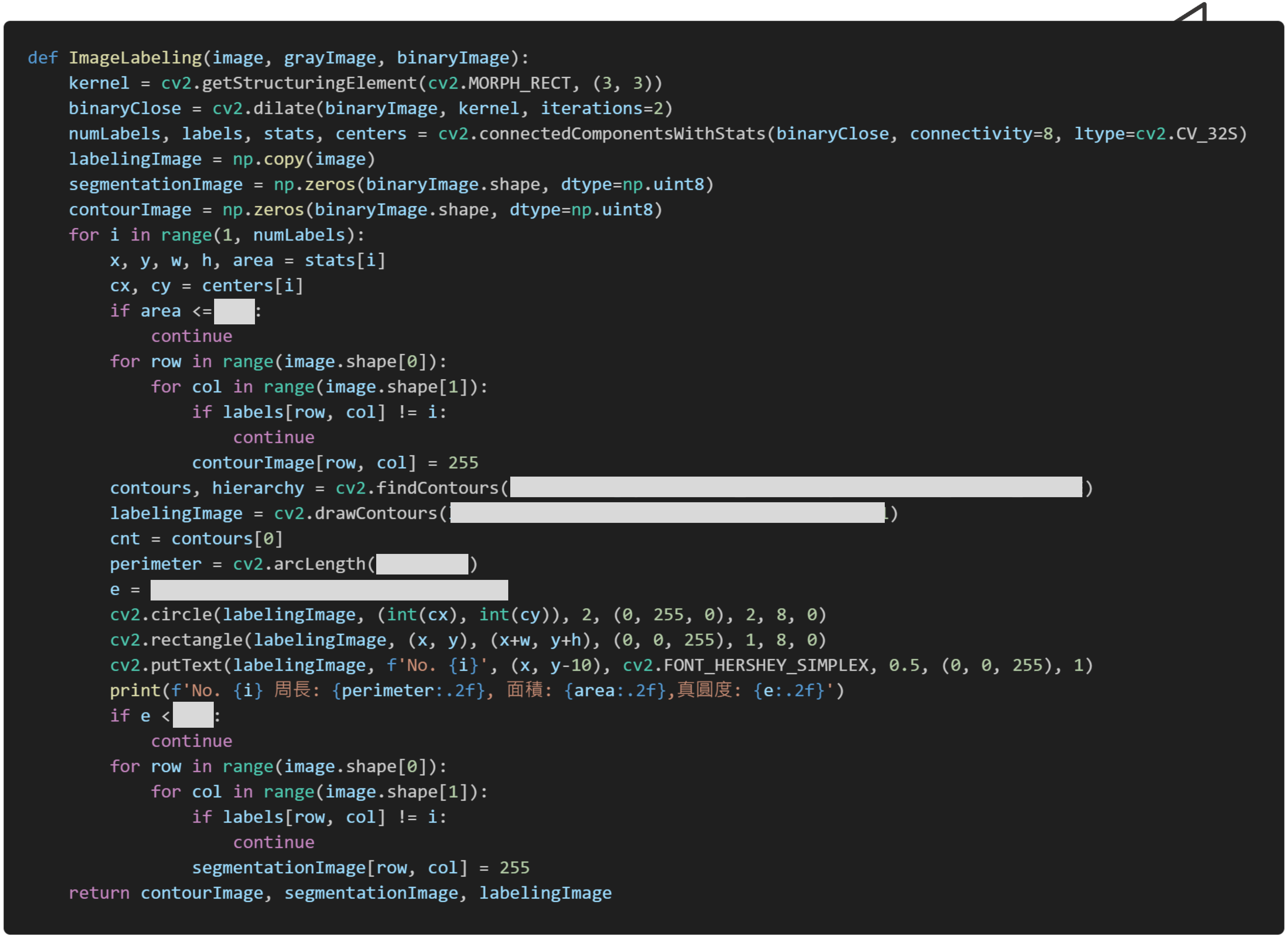
```
contours, hierarchy = cv2.findContours(                                        )
labelingImage = cv2.drawContours(                                  )
cnt = contours[0]
perimeter = cv2.arcLength(          )
e = 
cv2.circle(labelingImage, (int(cx), int(cy)), 2, (0, 255, 0), 2, 8, 0)
cv2.rectangle(labelingImage, (x, y), (x+w, y+h), (0, 0, 255), 1, 8, 0)
cv2.putText(labelingImage, f'No. {i}', (x, y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 1)
print(f'No. {i} 周長: {perimeter:.2f}, 面積: {area:.2f},真圓度: {e:.2f}')
```

影像標籤化

■ 將真圓度小於0.5的標籤連通元件去除，以留下圓形的物體

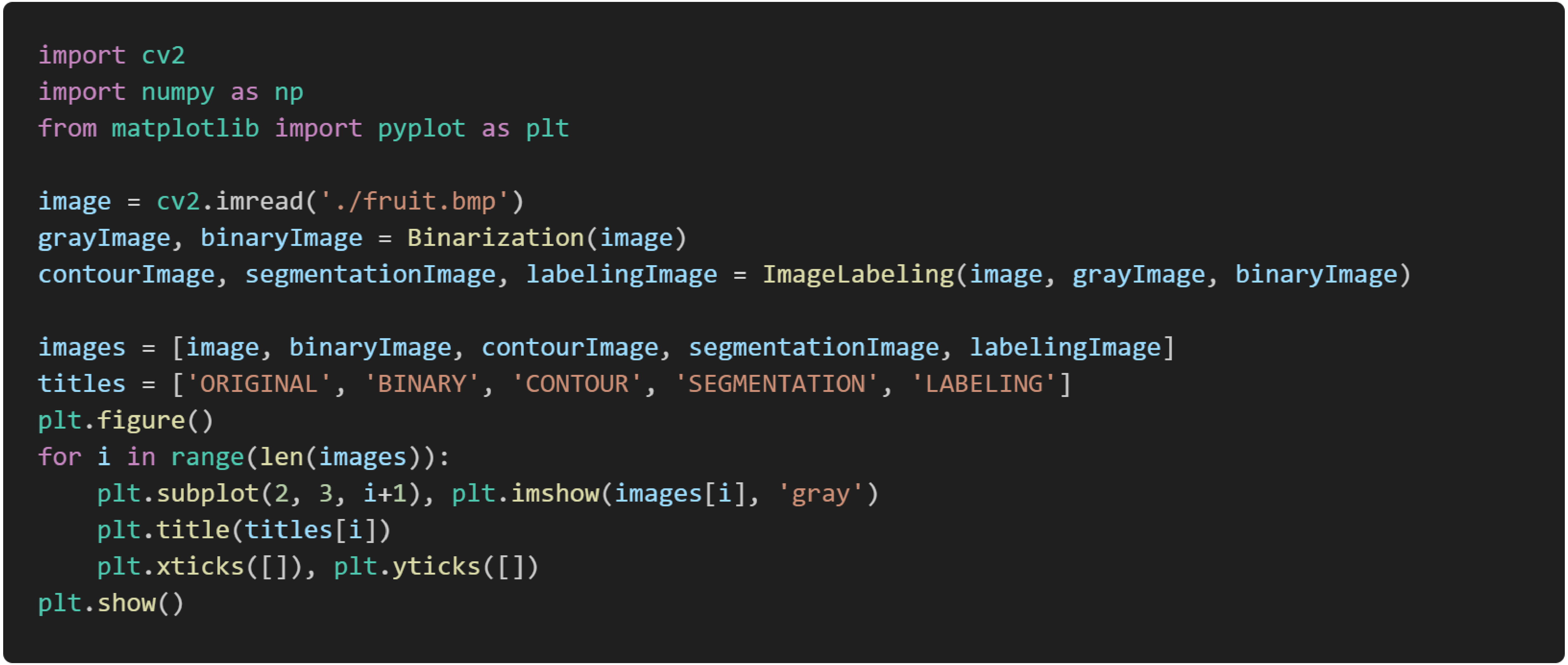


```
        if e <     :
            continue
        for row in range(image.shape[0]):
            for col in range(image.shape[1]):
                if labels[row, col] != i:
                    continue
                segmentationImage[row, col] = 255
    return contourImage, segmentationImage, labelingImage
```



```
def ImageLabeling(image, grayImage, binaryImage):
    kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (3, 3))
    binaryClose = cv2.dilate(binaryImage, kernel, iterations=2)
    numLabels, labels, stats, centers = cv2.connectedComponentsWithStats(binaryClose, connectivity=8, ltype=cv2.CV_32S)
    labelingImage = np.copy(image)
    segmentationImage = np.zeros(binaryImage.shape, dtype=np.uint8)
    contourImage = np.zeros(binaryImage.shape, dtype=np.uint8)
    for i in range(1, numLabels):
        x, y, w, h, area = stats[i]
        cx, cy = centers[i]
        if area <=    :
            continue
        for row in range(image.shape[0]):
            for col in range(image.shape[1]):
                if labels[row, col] != i:
                    continue
                contourImage[row, col] = 255
        contours, hierarchy = cv2.findContours(                                                  )
        labelingImage = cv2.drawContours(                                          )
        cnt = contours[0]
        perimeter = cv2.arcLength(         )
        e = 
        cv2.circle(labelingImage, (int(cx), int(cy)), 2, (0, 255, 0), 2, 8, 0)
        cv2.rectangle(labelingImage, (x, y), (x+w, y+h), (0, 0, 255), 1, 8, 0)
        cv2.putText(labelingImage, f'No. {i}', (x, y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 1)
        print(f'No. {i} 周長: {perimeter:.2f}, 面積: {area:.2f},真圓度: {e:.2f}')
        if e <    :
            continue
        for row in range(image.shape[0]):
            for col in range(image.shape[1]):
                if labels[row, col] != i:
                    continue
                segmentationImage[row, col] = 255
    return contourImage, segmentationImage, labelingImage
```

主程式

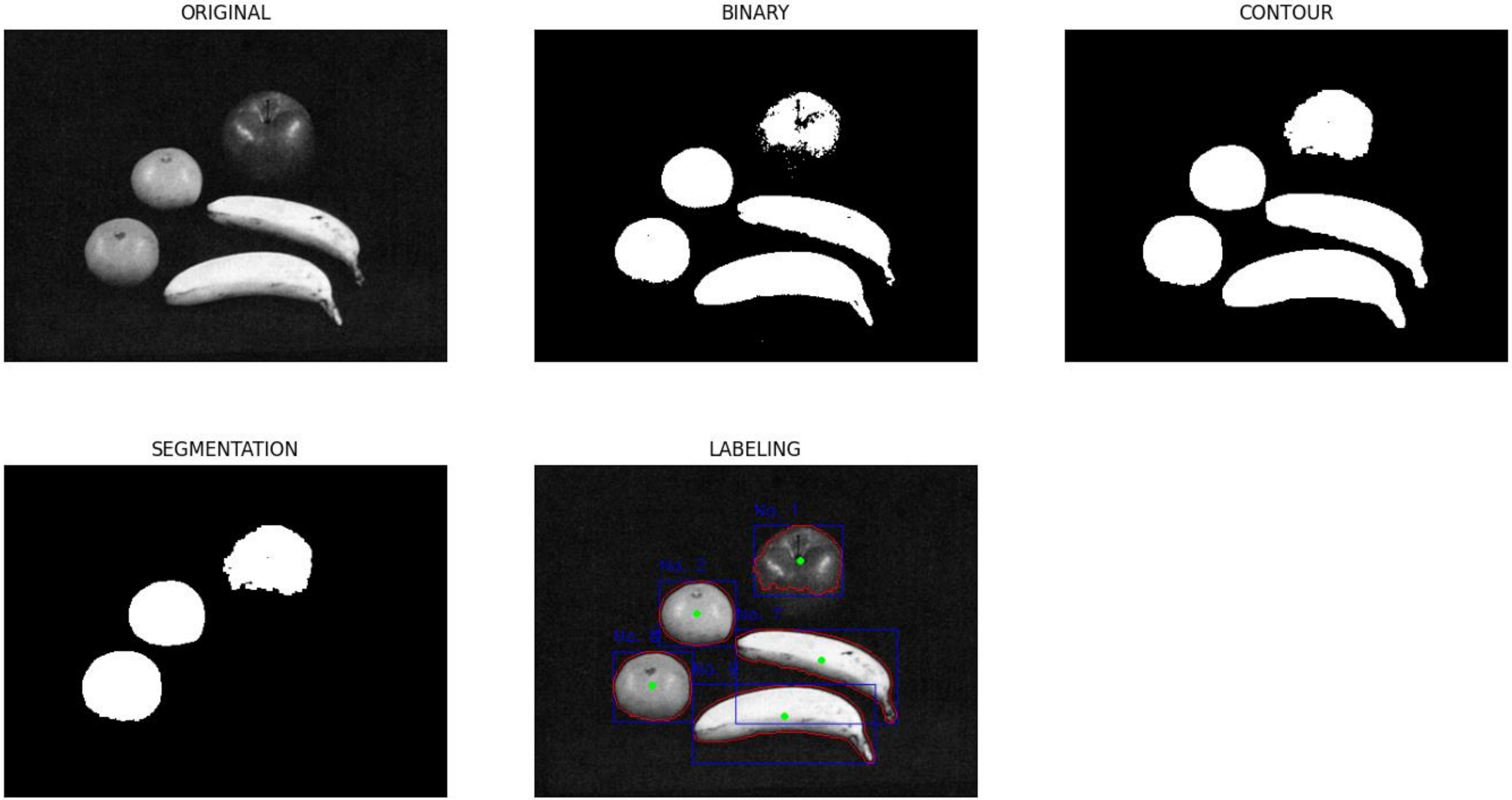


```python
import cv2
import numpy as np
from matplotlib import pyplot as plt

image = cv2.imread('./fruit.bmp')
grayImage, binaryImage = Binarization(image)
contourImage, segmentationImage, labelingImage = ImageLabeling(image, grayImage, binaryImage)

images = [image, binaryImage, contourImage, segmentationImage, labelingImage]
titles = ['ORIGINAL', 'BINARY', 'CONTOUR', 'SEGMENTATION', 'LABELING']
plt.figure()
for i in range(len(images)):
    plt.subplot(2, 3, i+1), plt.imshow(images[i], 'gray')
    plt.title(titles[i])
    plt.xticks([]), plt.yticks([])
plt.show()
```

需呈現之結果



ORIGINAL

BINARY

CONTOUR

SEGMENTATION

LABELING

Thanks for listening