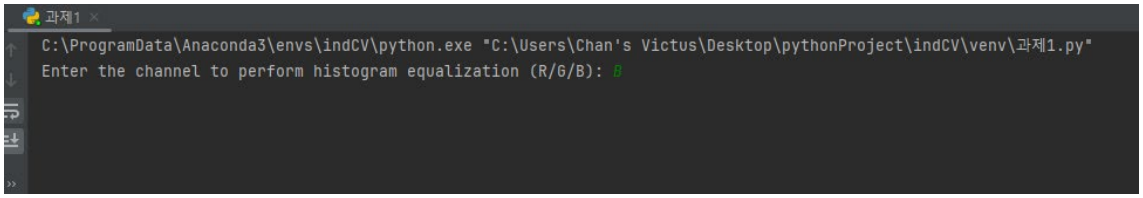
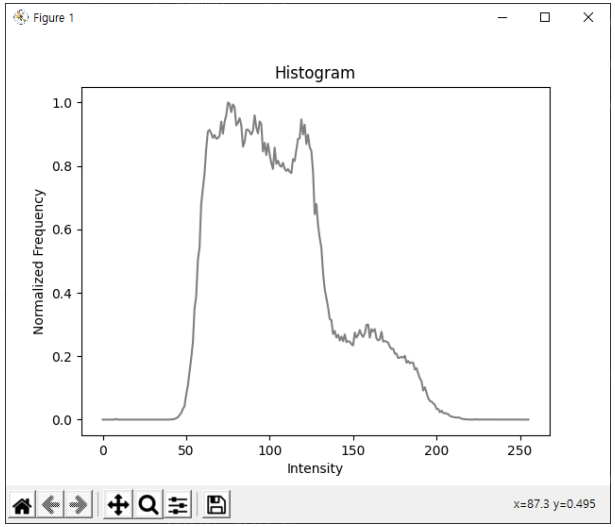
1-1. 1번 과제 코드

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

def histogram_equalization(image):
    # Check if image is color (3 channels) or grayscale
    if len(image.shape) == 3 and image.shape[2] == 3:
        # Convert image to grayscale
        gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    else:
        gray_image = image  # Assuming input is already grayscale

    # Equalize histogram
    equalized_image = cv2.equalizeHist(gray_image)

    # Calculate histogram
    hist = cv2.calcHist( images: [gray_image],  channels: [0],  mask: None,  histSize: [256],  ranges: [0, 256])

    # Normalize histogram
    hist_normalized = hist.ravel() / hist.max()

    return equalized_image, hist_normalized

def hsv_value_equalization(image):
    # Check if image is color (3 channels)
    if len(image.shape) == 3 and image.shape[2] == 3:
        # Convert image to HSV color space
        hsv_image = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

        # Split HSV image into channels
        h, s, v = cv2.split(hsv_image)

        # Equalize V channel
        equalized_v = cv2.equalizeHist(v)

        # Merge channels back together
        equalized_hsv_image = cv2.merge([h, s, equalized_v])

        # Convert back to BGR color space
        equalized_color_image = cv2.cvtColor(equalized_hsv_image, cv2.COLOR_HSV2BGR)

        # Calculate histogram
        hist = cv2.calcHist( images: [v],  channels: [0],  mask: None,  histSize: [256],  ranges: [0, 256])

        # Normalize histogram
        hist_normalized = hist.ravel() / hist.max()

        return equalized_color_image, hist_normalized
    else:
        print("Error: Input image does not have 3 channels (BGR).")
        exit()

# Read input image
input_image_path = '../data/Lena.png'  # Replace "input_image.jpg" with your image path
input_image = cv2.imread(input_image_path)

if input_image is None:
    print("Error: Unable to load image.")
    exit()

# Ensure image has 3 channels
if input_image.shape[2] != 3:
    print("Error: Input image does not have 3 channels (BGR).")
    exit()

# Get user input for channel selection
channel = input("Enter the channel to perform histogram equalization (R/G/B): ").upper()

# Perform histogram equalization based on user input
if channel == 'R':
    equalized_image, hist_normalized = histogram_equalization(input_image[:, :, 2])
elif channel == 'G':
    equalized_image, hist_normalized = histogram_equalization(input_image[:, :, 1])
elif channel == 'B':
    equalized_image, hist_normalized = histogram_equalization(input_image[:, :, 0])
else:
    print("Invalid channel selection.")
    exit()

# Display histogram
plt.plot( *args: hist_normalized, color='gray')
plt.xlabel('Intensity')
plt.ylabel('Normalized Frequency')
plt.title('Histogram')
plt.show()

# Display histogram equalized image
cv2.imshow( winname: "Histogram Equalized Image", equalized_image)
cv2.waitKey(0)
cv2.destroyAllWindows()

# Perform HSV value equalization
hsv_equalized_image, hist_normalized_hsv = hsv_value_equalization(input_image)

# Display HSV value equalized image
cv2.imshow( winname: "HSV Value Equalized Image", hsv_equalized_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

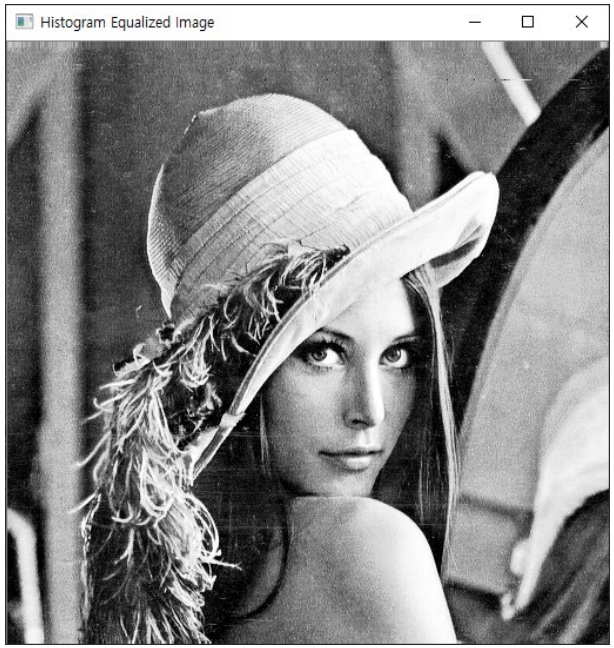
1-2. 1번 과제 결과



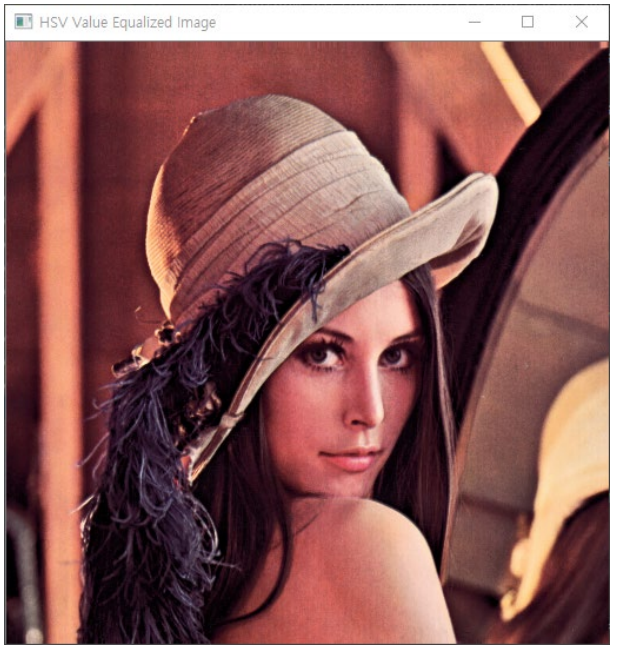
R,G,B 선택



histogram



Histogram
Equalized image

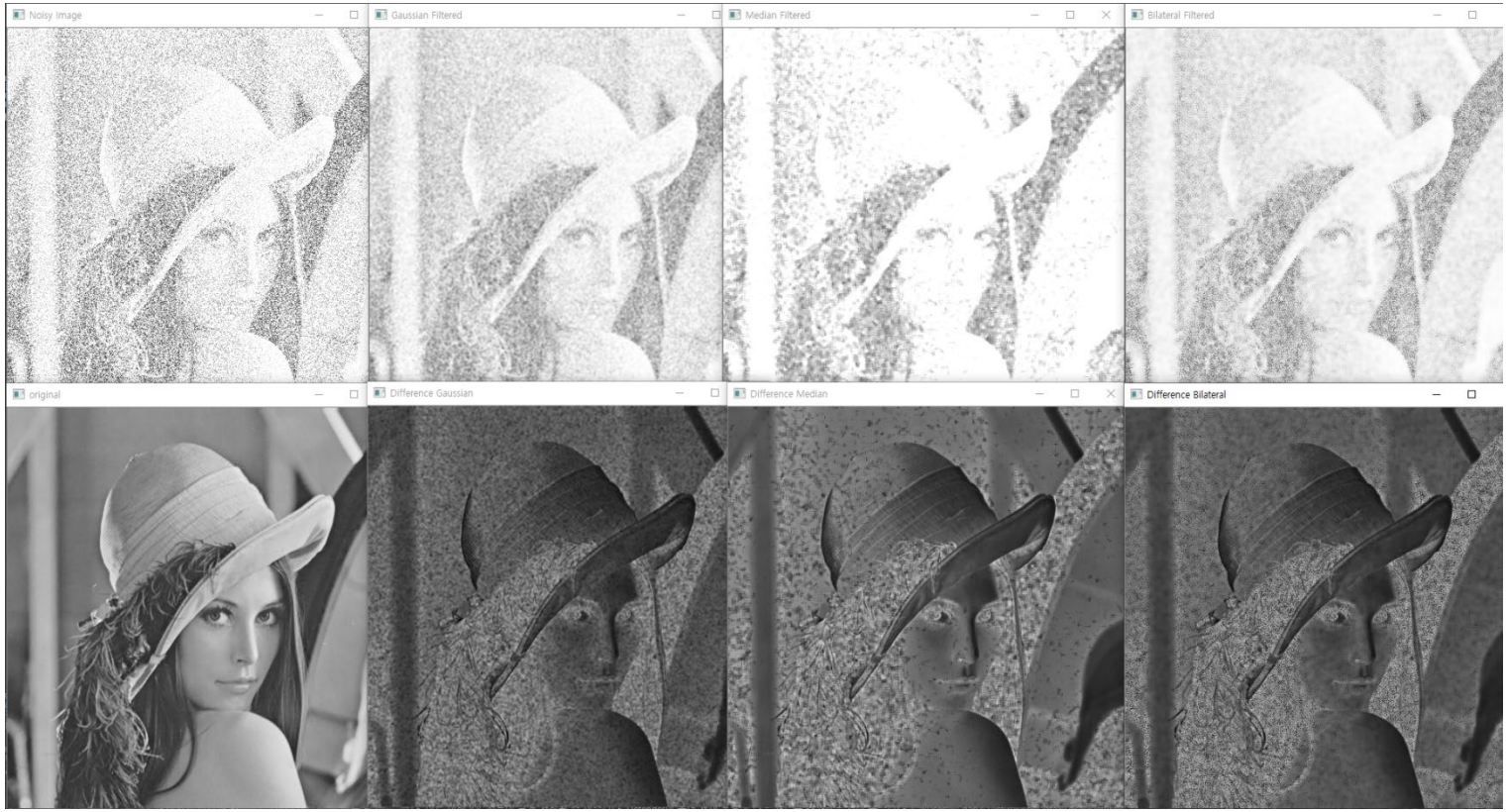


HSV Equalized
image

2-1. 2번 과제 코드

```python
import cv2
import numpy as np

# 이미지 불러오기
image = cv2.imread( filename: '../data/Lena.png', cv2.IMREAD_GRAYSCALE)

# 임의의 노이즈 생성
noise = np.random.normal(loc=0, scale=150, size=image.shape).astype(np.uint8)
noisy_image = cv2.add(image, noise)

# Gaussian 필터링
gaussian_filtered = cv2.GaussianBlur(noisy_image,  ksize: (5, 5),  sigmaX: 0)

# Median 필터링
median_filtered = cv2.medianBlur(noisy_image,  ksize: 5)

# Bilateral 필터링
bilateral_filtered = cv2.bilateralFilter(noisy_image,  d: 9,  sigmaColor: 75,  sigmaSpace: 75)

# 결과 출력
cv2.imshow( winname: 'Noisy Image', noisy_image)
cv2.imshow( winname: 'Gaussian Filtered', gaussian_filtered)
cv2.imshow( winname: 'Median Filtered', median_filtered)
cv2.imshow( winname: 'Bilateral Filtered', bilateral_filtered)

# 입력 영상과의 차이 계산
diff_gaussian = np.abs(image.astype(np.float32) - gaussian_filtered.astype(np.float32)).astype(np.uint8)
diff_median = np.abs(image.astype(np.float32) - median_filtered.astype(np.float32)).astype(np.uint8)
diff_bilateral = np.abs(image.astype(np.float32) - bilateral_filtered.astype(np.float32)).astype(np.uint8)

cv2.imshow( winname: 'original', image)
cv2.imshow( winname: 'Difference Gaussian', diff_gaussian)
cv2.imshow( winname: 'Difference Median', diff_median)
cv2.imshow( winname: 'Difference Bilateral', diff_bilateral)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

2-2. 2번 과제 결과

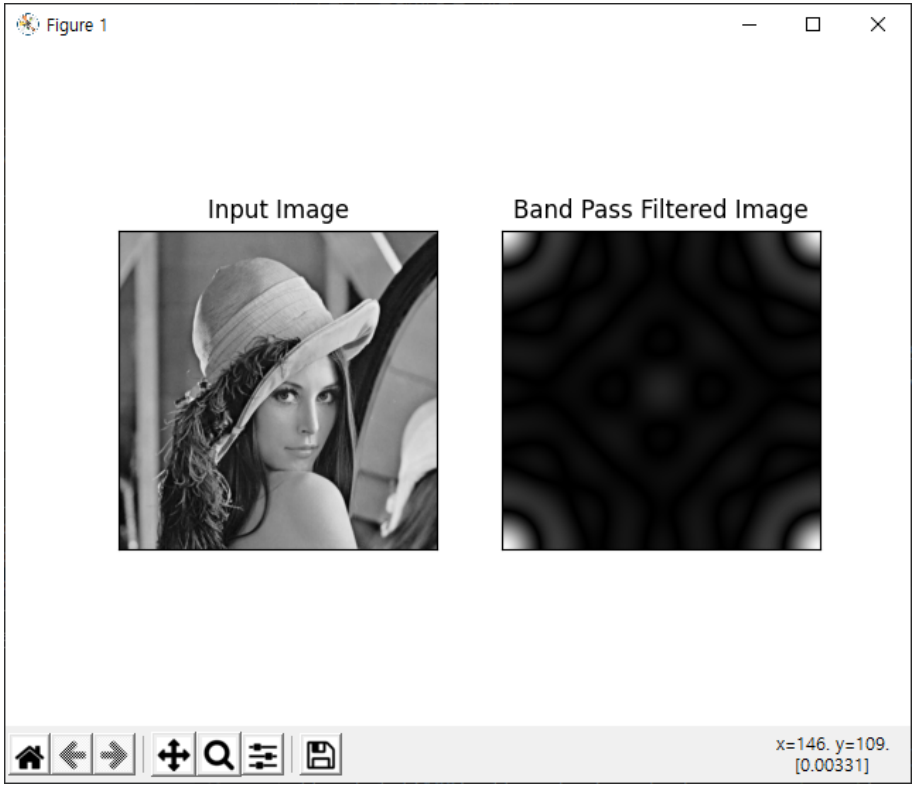

Noisy Image
Gaussian Filtered
Median Filtered
Bilateral Filtered
original
Difference Gaussian
Difference Median
Difference Bilateral

3-1. 3번 과제 코드

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

# 이미지 불러오기
image_path = '../data/Lena.png'
image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)

# DFT를 위한 함수 정의
def dft(img):
    f = np.fft.fft2(img)
    fshift = np.fft.fftshift(f)
    magnitude_spectrum = 20 * np.log(np.abs(fshift))
    return magnitude_spectrum

# 주파수 도메인으로 변환
dft_img = dft(image)

# 입력 받은 반지름
r1 = int(input("첫 번째 원의 반지름 입력: "))
r2 = int(input("두 번째 원의 반지름 입력: "))

# 중심 좌표 및 크기 계산
rows, cols = image.shape
center_row, center_col = rows // 2, cols // 2
x, y = np.ogrid[:rows, :cols]
mask1 = np.logical_and( *args: np.sqrt((x - center_row)**2 + (y - center_col)**2) > r1,
                        np.sqrt((x - center_row)**2 + (y - center_col)**2) < r2)

# 필터링
dft_img_filtered = dft_img * mask1

# 역 DFT
f_ishift = np.fft.ifftshift(dft_img_filtered)
img_back = np.fft.ifft2(f_ishift)
img_back = np.abs(img_back)

# 결과 출력
plt.subplot(121), plt.imshow(image, cmap = 'gray')
plt.title('Input Image'), plt.xticks([]), plt.yticks([])
plt.subplot(122), plt.imshow(img_back, cmap = 'gray')
plt.title('Band Pass Filtered Image'), plt.xticks([]), plt.yticks([])
plt.show()
```

3-2. 3번 과제 결과

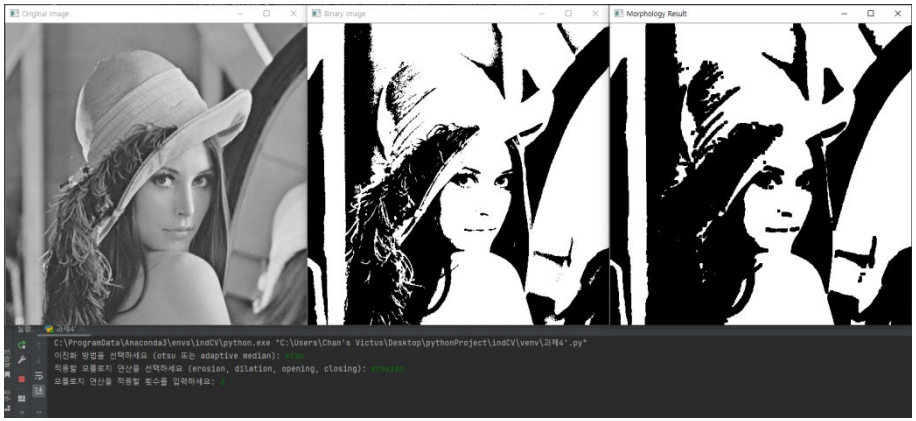
```
과제3
C:\ProgramData\Anaconda3\envs\indCV\python.exe "C:\Users\Chan's Victus\Desktop\pythonProject\indCV\venv\과제3.py"
첫 번째 원의 반지름 입력: 1
두 번째 원의 반지름 입력: 5
```



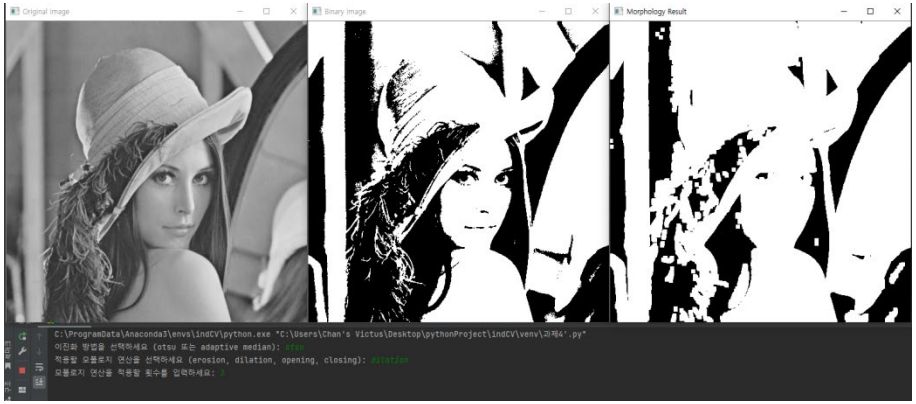
4-1. 4번 과제 코드

```python
import cv2
import numpy as np

# 이미지 불러오기
image_path = '../data/Lena.png'
image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)

# 사용자 입력 받기
binary_method = input("이진화 방법을 선택하세요 (otsu 또는 adaptive median): ")
morphology_operation = input("적용할 모폴로지 연산을 선택하세요 (erosion, dilation, opening, closing): ")
iterations = int(input("모폴로지 연산을 적용할 횟수를 입력하세요: "))

# 이진화
if binary_method == 'otsu':
    _, binary_image = cv2.threshold(image, thresh: 0, maxval: 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)
elif binary_method == 'adaptive median':
    binary_image = cv2.adaptiveThreshold(image, maxValue: 255, cv2.ADAPTIVE_THRESH_MEAN_C, cv2.THRESH_BINARY, blockSize: 11, C: 2)

# 모폴로지 연산 수행
kernel = np.ones( shape: (3,3),np.uint8)
if morphology_operation == 'erosion':
    result = cv2.erode(binary_image, kernel, iterations=iterations)
elif morphology_operation == 'dilation':
    result = cv2.dilate(binary_image, kernel, iterations=iterations)
elif morphology_operation == 'opening':
    result = cv2.morphologyEx(binary_image, cv2.MORPH_OPEN, kernel, iterations=iterations)
elif morphology_operation == 'closing':
    result = cv2.morphologyEx(binary_image, cv2.MORPH_CLOSE, kernel, iterations=iterations)

# 결과 출력
cv2.imshow( winname: 'Original Image', image)
cv2.imshow( winname: 'Binary Image', binary_image)
cv2.imshow( winname: 'Morphology Result', result)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

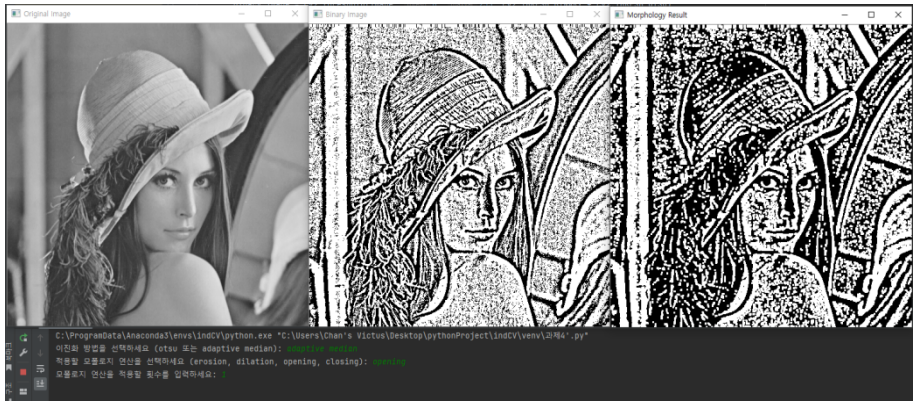
4-2. 4번 과제 결과



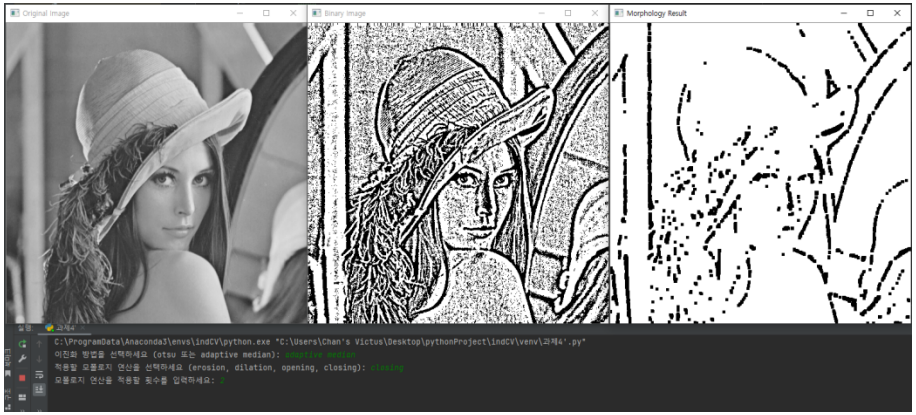
Otsu / erosion / 2회



Otsu / dilation / 3회



Adaptive median / opening / 1회



Adaptive median / closing / 2회