영상처리 프로그래밍

4주차

소개

- 조교 : 김형훈
  - 연구실 : A1409 (운영체제 연구실)
  - 이메일 : khh8996@naver.com
  - 전화번호 : 010-5316-7953

- 담당 교수 : 김백섭 교수님

목차

- Matplotlib 라이브러리
- 이미지 파일 불러오기, 보여주기, 저장하기
- 실습
- 과제

Matplotlib 라이브러리

- Matplotlib
  - Python에서 데이터를 그래프로 시각화 하는 라이브러리

- 다양한 시각화 기능을 제공
  - 선 그래프 (line plot)
  - 막대 그래프 (bar plot)
  - 산점도 (scatter plot)
  - 원 그래프 (pie plot)
  - 히스토그램 (histogram)
  - …

Matplotlib 라이브러리

- import를 이용하여 matplotlib 라이브러리 중 pyplot 모듈을 불러와서 plt라는 명칭으로 지정하여 사용함
  - pyplot : matplotlib에서 지원하는 모듈 중 하나이며 사용 환경 인터페이스를 제공
  - %matplotlib inline : Jupyter Notebook 내부에 그림을 표시하도록 하기 위해 작성

```
import matplotlib.pyplot as plt
%matplotlib inline
```

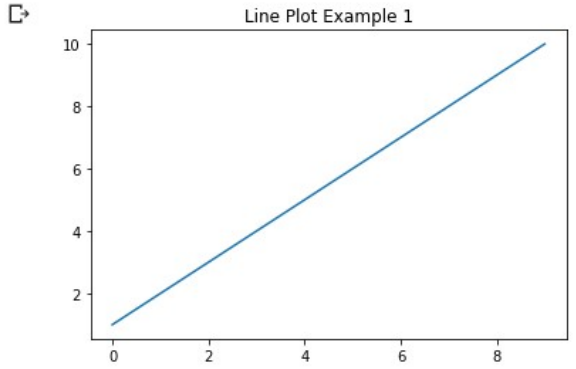
Matplotlib 라이브러리

- 선 그래프 (line plot)
  - title() : 그래프의 제목을 표현할 때 사용
  - plot() : 선 그래프 그릴 때 사용
    - plot(리스트) or plot(배열) : plot 함수에 리스트 또는 배열을 넘김
    - 1차원 리스트 또는 배열을 넘겼을 때, 요소 값이 y 축에 표시됨
    - 2차원 리스트 또는 배열을 넘겼을 때, 왼쪽 데이터는 x 축, 오른쪽 데이터는 y 축에 표시됨
  - show() : 그래프를 화면에 보여줄 때 사용
  - xlabel() : x 축 레이블을 표현할 때 사용
  - ylabel() : y 축 레이블을 표현할 때 사용

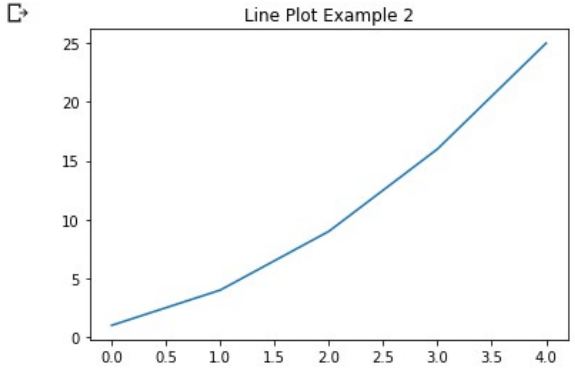
Matplotlib 라이브러리

- 선 그래프 (line plot) - 예시

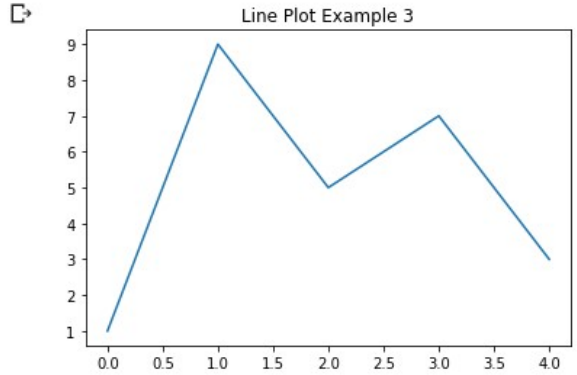
```
import matplotlib.pyplot as plt
%matplotlib inline

plt.title("Line Plot Example 1")
plt.plot([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

data = [1, 4, 9, 16, 25]

plt.title("Line Plot Example 2")
plt.plot(data)
plt.show()
```



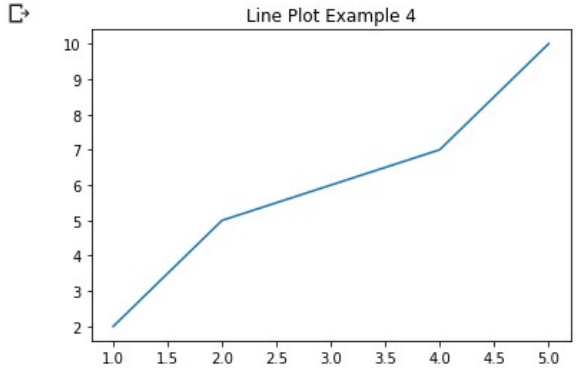

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

data = np.array([1, 9, 5, 7, 3])

plt.title("Line Plot Example 3")
plt.plot(data)
plt.show()
```



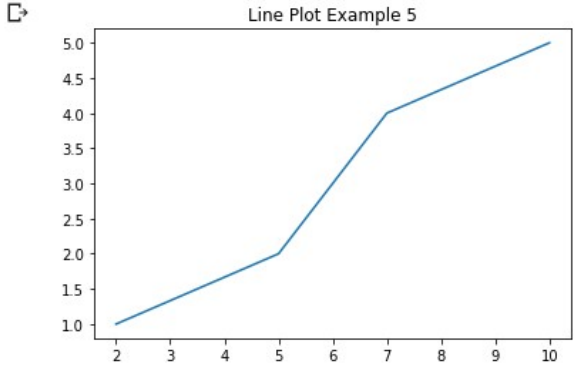
Matplotlib 라이브러리

- 선 그래프 (line plot) - 예시

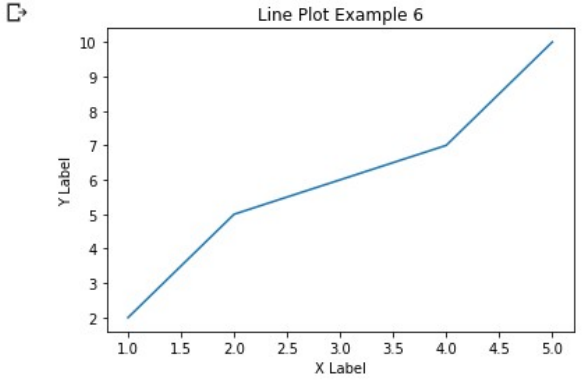
```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([1, 2, 3, 4, 5])
y_data = np.array([2, 5, 6, 7, 10])

plt.title("Line Plot Example 4")
plt.plot(x_data, y_data)
plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([2, 5, 6, 7, 10])
y_data = np.array([1, 2, 3, 4, 5])

plt.title("Line Plot Example 5")
plt.plot(x_data, y_data)
plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([1, 2, 3, 4, 5])
y_data = np.array([2, 5, 6, 7, 10])

plt.title("Line Plot Example 6")
plt.plot(x_data, y_data)
plt.xlabel('X Label')
plt.ylabel('Y Label')
plt.show()
```



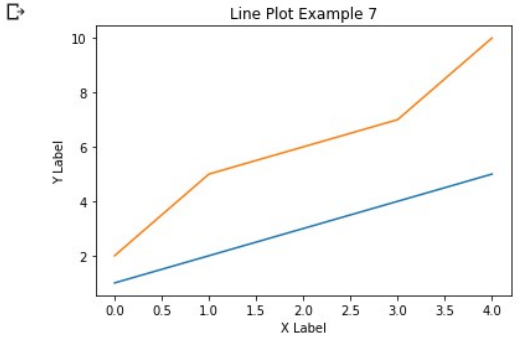
Matplotlib 라이브러리

- 선 그래프 (line plot)
  - plot()의 호출 횟수에 따라 그래프에 선이 표시됨
  - grid() : 그래프의 격자를 표시할 때 사용
  - legend() : 그래프의 선에 대한 레이블을 표시할 때 사용
    - loc를 사용하여 그래프에 레이블의 위치를 직접 설정할 수 있음
    - 위치 종류 : best, upper right, upper left, lower left, lower right, right, center left, center right, lower center, upper center, center

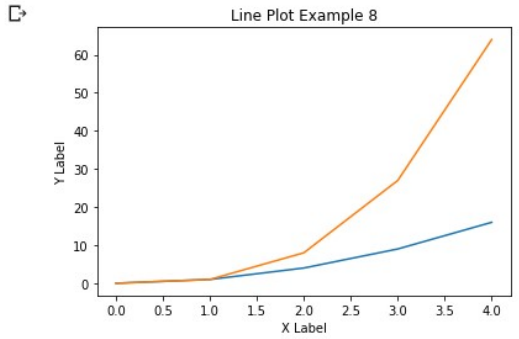
Matplotlib 라이브러리

- 선 그래프 (line plot) - 예시

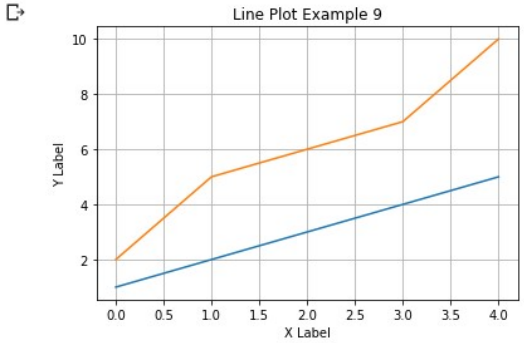
```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([1, 2, 3, 4, 5])
y_data = np.array([2, 5, 6, 7, 10])

plt.title("Line Plot Example 7")
plt.plot(x_data)
plt.plot(y_data)
plt.xlabel('X Label')
plt.ylabel('Y Label')
plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([0, 1, 2, 3, 4])
y_data_1 = np.array([0, 1, 4, 9, 16])
y_data_2 = np.array([0, 1, 8, 27, 64])

plt.title("Line Plot Example 8")
plt.plot(x_data, y_data_1)
plt.plot(x_data, y_data_2)
plt.xlabel('X Label')
plt.ylabel('Y Label')
plt.show()
```



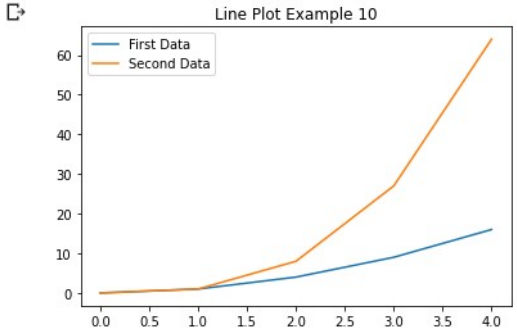

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([1, 2, 3, 4, 5])
y_data = np.array([2, 5, 6, 7, 10])

plt.title("Line Plot Example 9")
plt.plot(x_data)
plt.plot(y_data)
plt.xlabel('X Label')
plt.ylabel('Y Label')
plt.grid()
plt.show()
```



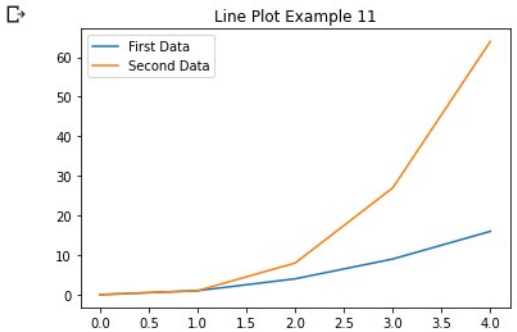
Matplotlib 라이브러리

- 선 그래프 (line plot) - 예시

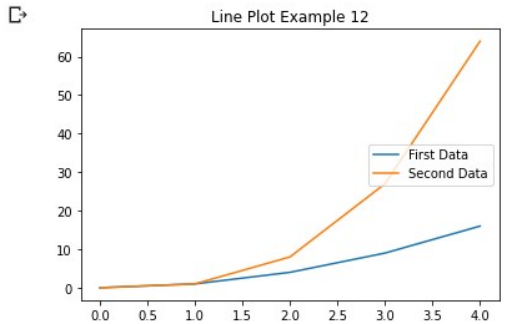
```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([0, 1, 2, 3, 4])
y_data_1 = np.array([0, 1, 4, 9, 16])
y_data_2 = np.array([0, 1, 8, 27, 64])

plt.title("Line Plot Example 10")
plt.plot(x_data, y_data_1)
plt.plot(x_data, y_data_2)
plt.legend(['First Data', 'Second Data'])
plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([0, 1, 2, 3, 4])
y_data_1 = np.array([0, 1, 4, 9, 16])
y_data_2 = np.array([0, 1, 8, 27, 64])

plt.title("Line Plot Example 11")
plt.plot(x_data, y_data_1)
plt.plot(x_data, y_data_2)
plt.legend(['First Data', 'Second Data'], loc='best')
plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([0, 1, 2, 3, 4])
y_data_1 = np.array([0, 1, 4, 9, 16])
y_data_2 = np.array([0, 1, 8, 27, 64])

plt.title("Line Plot Example 12")
plt.plot(x_data, y_data_1)
plt.plot(x_data, y_data_2)
plt.legend(['First Data', 'Second Data'], loc='right')
plt.show()
```



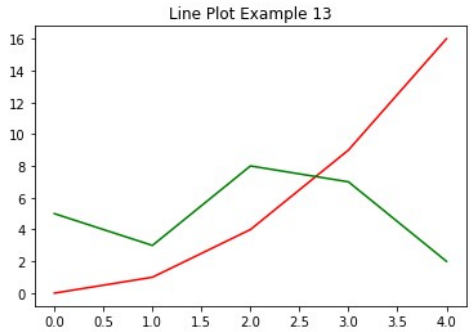
Matplotlib 라이브러리

- 선 그래프 (line plot)
  - 다양한 스타일 기능을 제공함
    - 색깔(color), 마커(marker), 선 종류(line style), …
    - color를 이용하여 선의 색깔을 변경할 수 있음
      - color에 대한 더 많은 정보는 아래 링크 참고 바람
      - https://matplotlib.org/examples/color/named_colors.html
    - marker를 이용하여 데이터 위치를 나타내는 기호를 변경할 수 있음
      - marker에 대한 더 많은 정보는 아래 링크 참고 바람
      - https://matplotlib.org/api/markers_api.html
    - linestyle을 사용하여 선의 종류를 변경할 수 있음
      - linestyle에 대한 더 많은 정보는 아래 링크 참고 바람
      - https://matplotlib.org/3.2.1/gallery/lines_bars_and_markers/linestyles.html

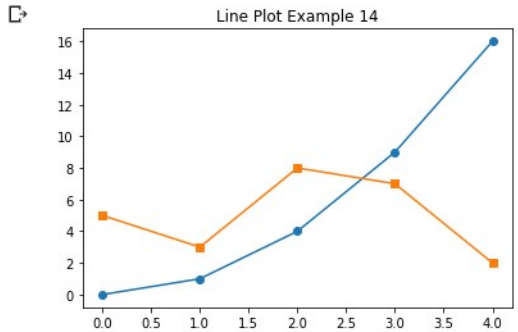
Matplotlib 라이브러리

- 선 그래프 (line plot) - 예시

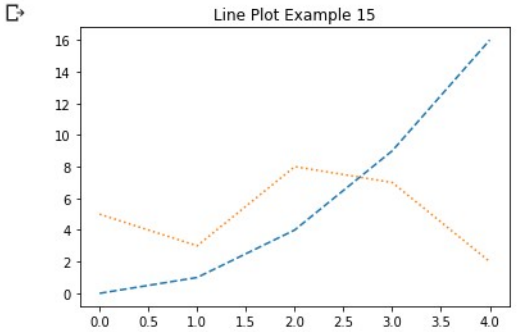
```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([0, 1, 2, 3, 4])
y_data_1 = np.array([0, 1, 4, 9, 16])
y_data_2 = np.array([5, 3, 8, 7, 2])

plt.title("Line Plot Example 13")
plt.plot(x_data, y_data_1, color='red')
plt.plot(x_data, y_data_2, color='green')
plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([0, 1, 2, 3, 4])
y_data_1 = np.array([0, 1, 4, 9, 16])
y_data_2 = np.array([5, 3, 8, 7, 2])

plt.title("Line Plot Example 14")
plt.plot(x_data, y_data_1, marker='o')
plt.plot(x_data, y_data_2, marker='s')
plt.show()
```



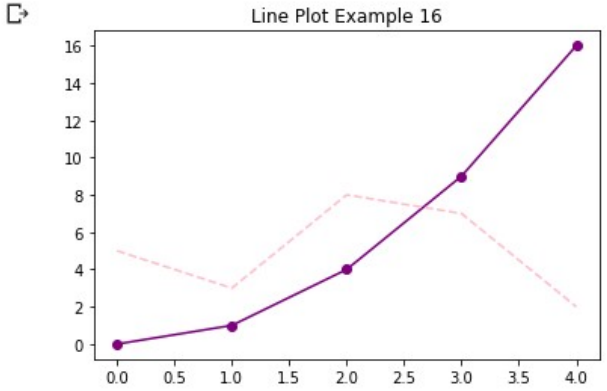

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([0, 1, 2, 3, 4])
y_data_1 = np.array([0, 1, 4, 9, 16])
y_data_2 = np.array([5, 3, 8, 7, 2])

plt.title("Line Plot Example 15")
plt.plot(x_data, y_data_1, linestyle='--')
plt.plot(x_data, y_data_2, linestyle=':')
plt.show()
```



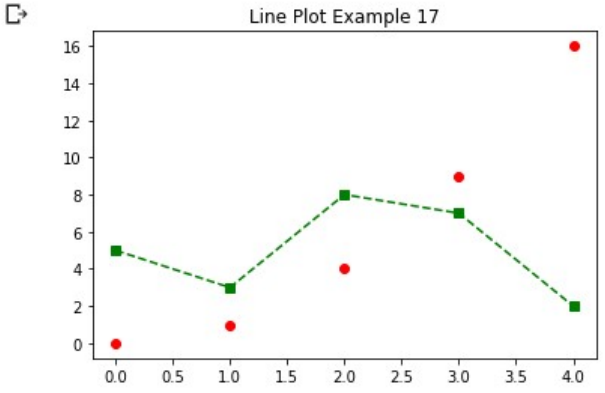
Matplotlib 라이브러리

- 선 그래프 (line plot) - 예시

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([0, 1, 2, 3, 4])
y_data_1 = np.array([0, 1, 4, 9, 16])
y_data_2 = np.array([5, 3, 8, 7, 2])

plt.title("Line Plot Example 16")
plt.plot(x_data, y_data_1, color='purple', marker='o')
plt.plot(x_data, y_data_2, color='pink', linestyle='--')
plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([0, 1, 2, 3, 4])
y_data_1 = np.array([0, 1, 4, 9, 16])
y_data_2 = np.array([5, 3, 8, 7, 2])

plt.title("Line Plot Example 17")
plt.plot(x_data, y_data_1, color='red', marker='o', linestyle='')
plt.plot(x_data, y_data_2, color='green', marker='s', linestyle='--')
plt.show()
```



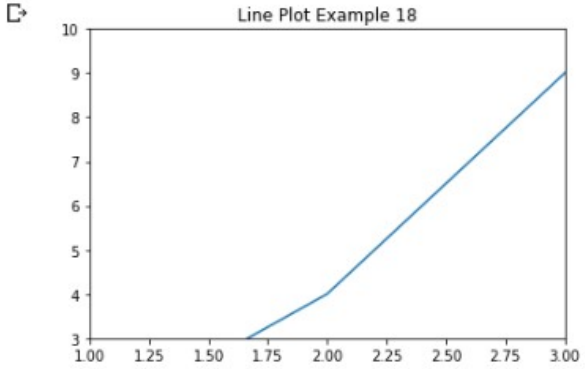
Matplotlib 라이브러리

- 선 그래프 (line plot)
  - 그래프에 표시되는 범위를 지정할 수 있음
    - xlim() : x 축의 최솟값과 최댓값을 지정
    - ylim() : y 축의 최솟값과 최댓값을 지정
    - axis() : x 축, y 축 각각의 최솟값과 최댓값을 지정
      - axis([x 축 최솟값, x 축 최댓값, y 축 최솟값, y 축 최댓값])
    - xticks() : x 축의 표시 지점을 지정
    - yticks() : y 축의 표시 지점을 지정

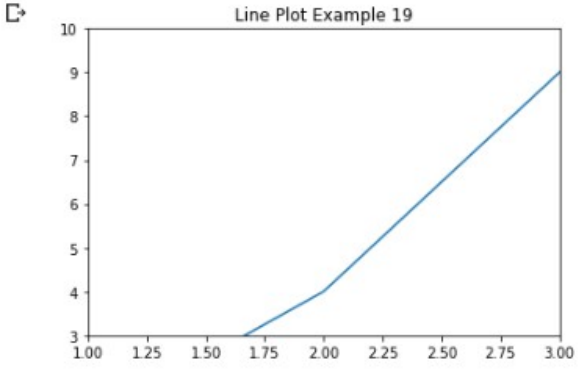
Matplotlib 라이브러리

- 선 그래프 (line plot) - 예시

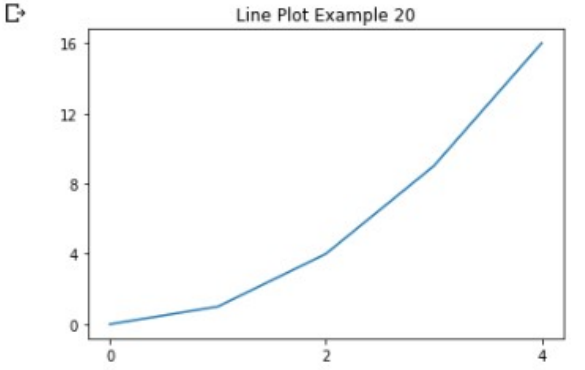
```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([0, 1, 2, 3, 4])
y_data = np.array([0, 1, 4, 9, 16])

plt.title("Line Plot Example 18")
plt.plot(x_data, y_data)
plt.xlim(1, 3)
plt.ylim(3, 10)
plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([0, 1, 2, 3, 4])
y_data = np.array([0, 1, 4, 9, 16])

plt.title("Line Plot Example 19")
plt.plot(x_data, y_data)
plt.axis([1, 3, 3, 10])
plt.show()
```

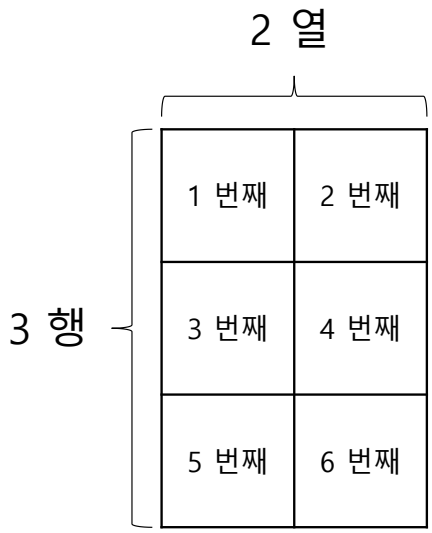



```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([0, 1, 2, 3, 4])
y_data = np.array([0, 1, 4, 9, 16])

plt.title("Line Plot Example 20")
plt.plot(x_data, y_data)
plt.xticks([0, 2, 4])
plt.yticks([0, 4, 8, 12, 16])
plt.show()
```



Matplotlib 라이브러리

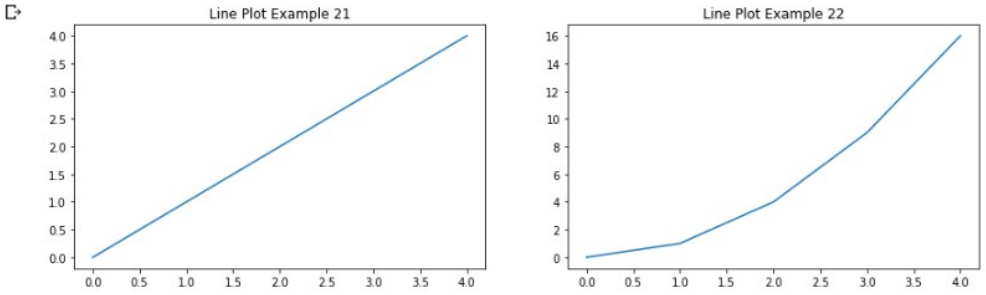
- 선 그래프 (line plot)
  - subplot을 사용하여 그래프를 여러 개 그릴 수 있음
    - subplot(m 행, n 열, i 번째 그래프)
    - subplots_adjust() : 그래프와 그래프 사이의 간격을 조정
      - left : 왼쪽 공간
      - right : 오른쪽 공간
      - bottom : 아래쪽 공간
      - top : 위쪽 공간
      - wspace : 가로로 인접한 그래프 사이의 공간
      - hspace : 세로로 인접한 그래프 사이의 공간



Matplotlib 라이브러리

- 선 그래프 (line plot) - 예시

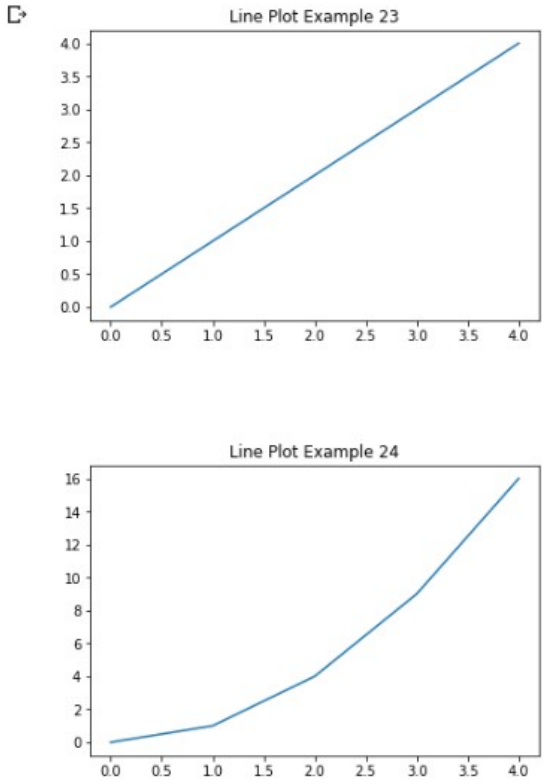
```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

first_data = np.array([0, 1, 2, 3, 4])
second_data = np.array([0, 1, 4, 9, 16])

# 1행 2열의 1 번째 그래프
plt.subplot(1, 2, 1)
plt.title("Line Plot Example 21")
plt.plot(first_data)

# 1행 2열의 2 번째 그래프
plt.subplot(1, 2, 2)
plt.title("Line Plot Example 22")
plt.plot(second_data)

plt.subplots_adjust(right=2)
plt.show()
```



Matplotlib 라이브러리

- 선 그래프 (line plot) - 예시

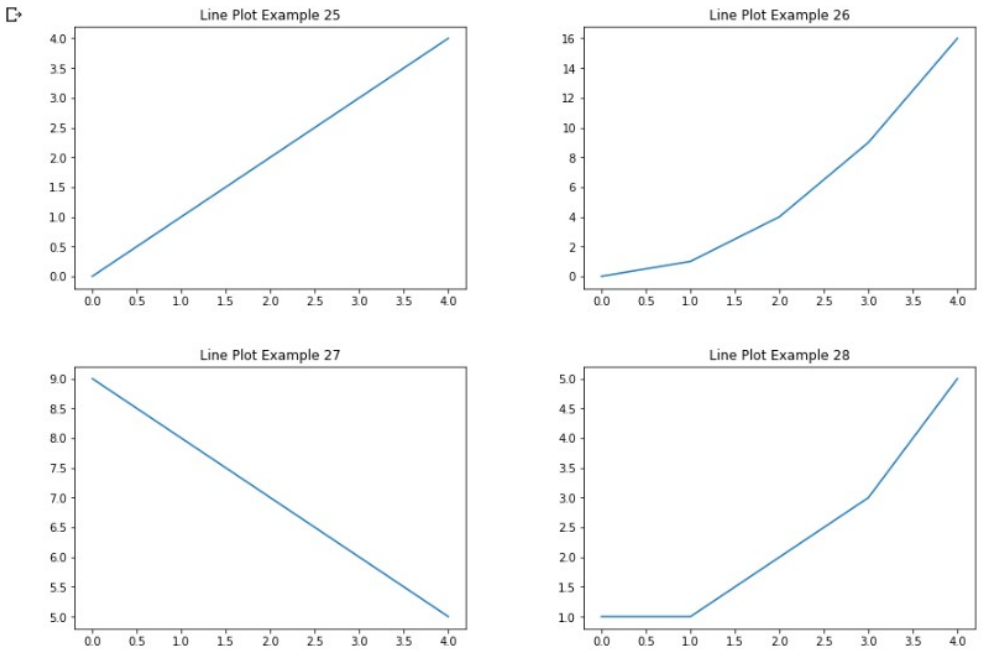
```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

first_data = np.array([0, 1, 2, 3, 4])
second_data = np.array([0, 1, 4, 9, 16])

# 2행 1열의 1 번째 그래프
plt.subplot(2, 1, 1)
plt.title("Line Plot Example 23")
plt.plot(first_data)

# 2행 1열의 2 번째 그래프
plt.subplot(2, 1, 2)
plt.title("Line Plot Example 24")
plt.plot(second_data)

plt.subplots_adjust(top=2, hspace=0.5)
plt.show()
```



Matplotlib 라이브러리

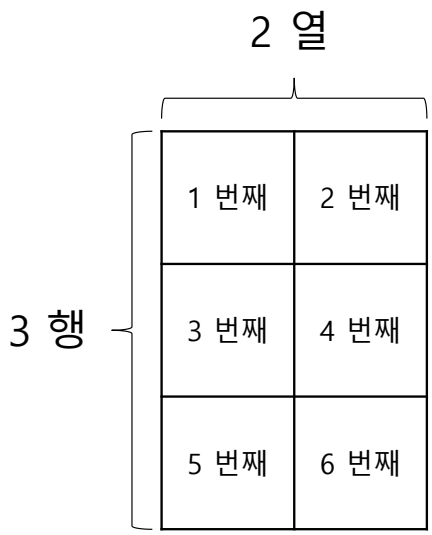
- 선 그래프 (line plot) - 예시

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

first_data = np.array([0, 1, 2, 3, 4])
second_data = np.array([0, 1, 4, 9, 16])
third_data = np.array([9, 8, 7, 6, 5])
fourth_data = np.array([1, 1, 2, 3, 5])

# 2행 2열의 1 번째 그래프
plt.subplot(2, 2, 1)
plt.title("Line Plot Example 25")
plt.plot(first_data)

# 2행 2열의 2 번째 그래프
plt.subplot(2, 2, 2)
plt.title("Line Plot Example 26")
plt.plot(second_data)

# 2행 2열의 3 번째 그래프
plt.subplot(2, 2, 3)
plt.title("Line Plot Example 27")
plt.plot(third_data)

# 2행 2열의 4 번째 그래프
plt.subplot(2, 2, 4)
plt.title("Line Plot Example 28")
plt.plot(fourth_data)

plt.subplots_adjust(top=2, right=2, wspace=0.3, hspace=0.3)
plt.show()
```



Matplotlib 라이브러리

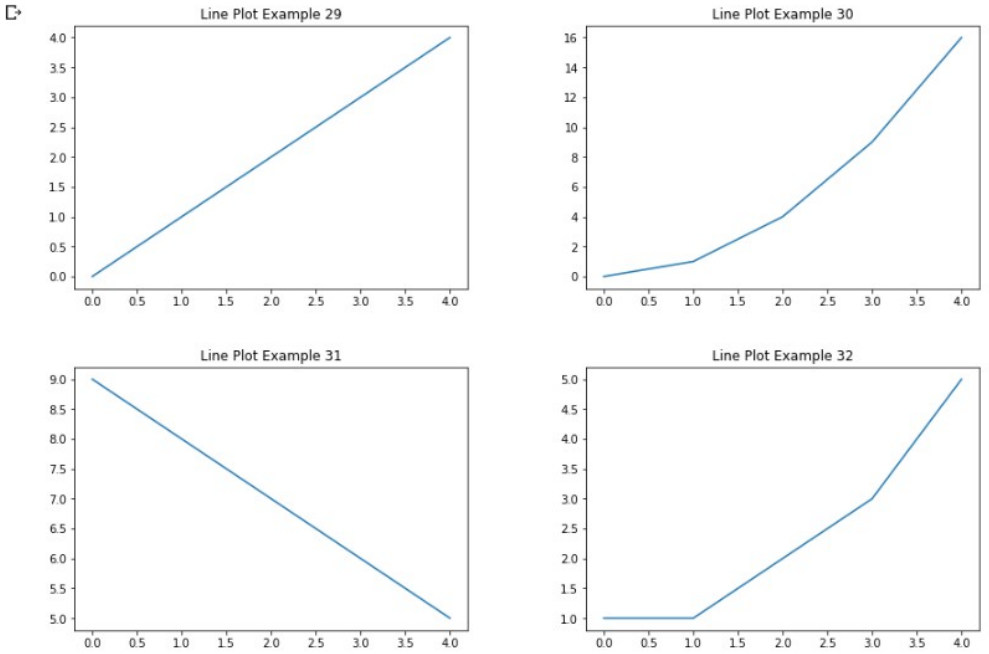
- 선 그래프 (line plot)
  - Figure 객체를 사용하여 그래프를 여러 개 그릴 수 있음
    - figure() : Figure 객체 생성
    - figsize() : 그래프의 크기를 설정
    - add_subplot(m 행, n 열, i 번째 그래프)



Matplotlib 라이브러리

- 선 그래프 (line plot) - 예시

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

first_data = np.array([0, 1, 2, 3, 4])
second_data = np.array([0, 1, 4, 9, 16])
third_data = np.array([9, 8, 7, 6, 5])
fourth_data = np.array([1, 1, 2, 3, 5])
data = [first_data, second_data, third_data, fourth_data]

fig = plt.figure()

# 2행 2열의 그래프
for i in range(1, 5):
  ax = fig.add_subplot(2, 2, i)
  ax.set_title('Line Plot Example {}'.format(i+28))
  ax.plot(data[i-1])

fig.subplots_adjust(top=2, right=2, wspace=0.3, hspace=0.3)
plt.show()
```



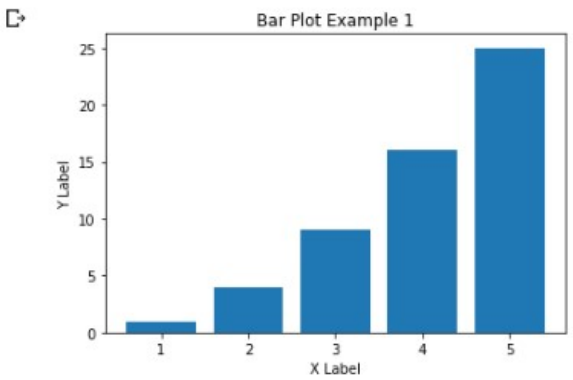
Matplotlib 라이브러리

- 막대 그래프 (bar plot)
  - bar([x 축 데이터], [y 축 데이터])
- 산점도 (scatter plot)
  - scatter([x 축 데이터], [y 축 데이터])
- 원 그래프 (pie plot)
  - pie([비율 데이터])
- 히스토그램 (histogram)
  - hist(리스트 또는 배열, bins=데이터를 집계할 구간 정보)

Matplotlib 라이브러리

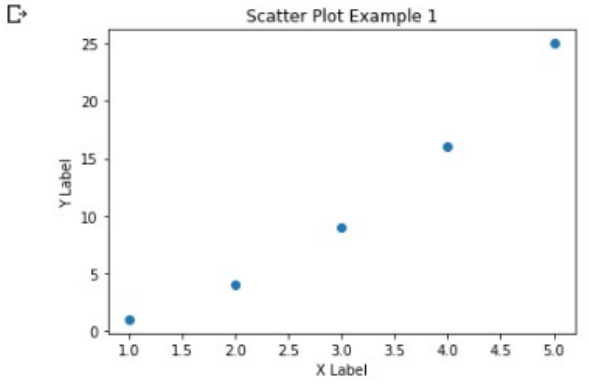
- 막대 그래프 - 예시

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([1, 2, 3, 4, 5])
y_data = np.array([1, 4, 9, 16, 25])

plt.title("Bar Plot Example 1")
plt.bar(x_data, y_data)
plt.xlabel('X Label')
plt.ylabel('Y Label')
plt.show()
```



- 산점도 - 예시

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x_data = np.array([1, 2, 3, 4, 5])
y_data = np.array([1, 4, 9, 16, 25])

plt.title("Scatter Plot Example 1")
plt.scatter(x_data, y_data)
plt.xlabel('X Label')
plt.ylabel('Y Label')
plt.show()
```



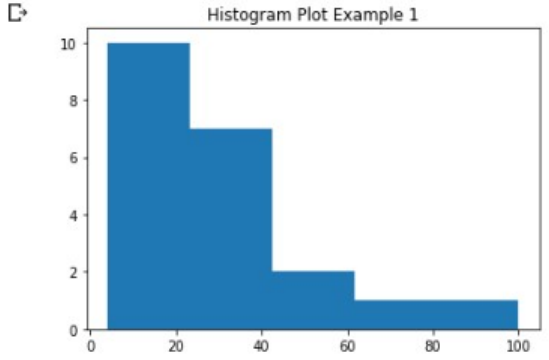
Matplotlib 라이브러리

- 원 그래프 - 예시

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

data = np.array([1, 2, 3, 4, 5])

plt.title("Pie Plot Example 1")
plt.pie(data)
plt.show()
```

- 히스토그램 - 예시

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

data = [21, 22, 23, 4, 5, 6, 77, 8, 9, 10, 31, 32, 33, 34, 35, 36, 37, 18, 49, 50, 100]

plt.title("Histogram Plot Example 1")
plt.hist(data, bins=5)
plt.show()
```



이미지 파일 불러오기, 보여주기, 저장하기

- OpenCV 라이브러리 이용
  - OpenCV(Open Source Computer Vision) : 다양한 영상 처리에 사용할 수 있는 오픈소스 라이브러리
  - import cv2

- imread()를 사용하여 이미지 파일 불러올 수 있음
  - cv2.imread(filename, flag)
  - filename : 이미지 파일의 경로 및 파일명
  - flag : 이미지 파일 불러올 때, 옵션
    - cv2.IMREAD_COLOR : 이미지 파일을 Color(BGR) 스케일로 불러옴, 기본값
    - cv2.IMREAD_UNCHANGED : 이미지 파일을 변함 없이 불러옴
    - cv2.IMREAD_GRAYSCALE : 이미지 파일을 Gray 스케일로 불러옴

이미지 파일 불러오기, 보여주기, 저장하기

- shape를 사용하여 이미지의 형태를 확인할 수 있음
  - 이미지.shape -> (행, 열, 색(BGR))

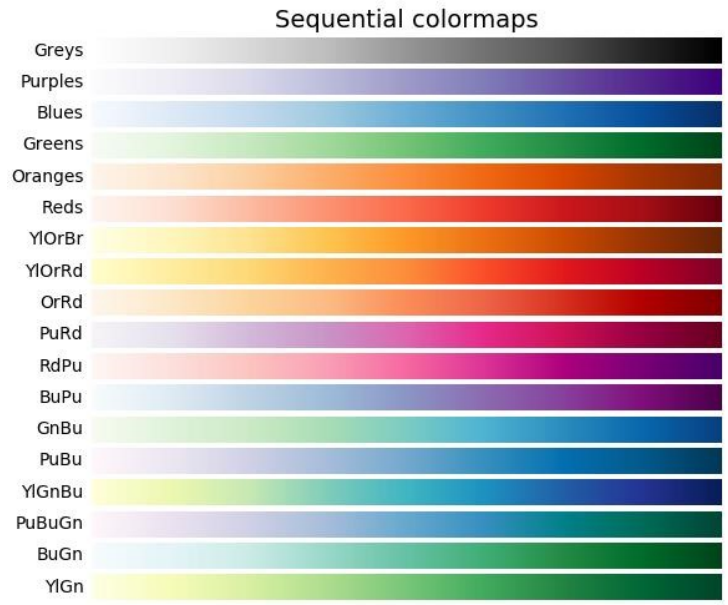
- cvtColor를 사용하여 이미지의 색상을 변경할 수 있음
  - cv2.cvtColor(image, code)
  - image : 이미지 파일
  - code : 변환 코드

- split()을 사용하여 이미지의 채널을 분리
  - cv2.split(image)
  - image : 이미지 파일

이미지 파일 불러오기, 보여주기, 저장하기

- imshow()를 사용하여 이미지 파일 화면에 출력할 수 있음
  - plt.imshow(image, cmap, vmin, vmax, …)
  - image : 이미지 파일
  - cmap : Matplotlib의 colormap
  - vmin : colorbar의 최솟값
  - vmax : colorvar의 최댓값
  - …
  - plt.show()

- imwrite()를 사용하여 이미지 파일을 저장할 수 있음
  - cv2.imwrite(filename, image)
  - filename : 저장될 이미지 경로 및 파일명
  - image : 저장할 이미지 파일



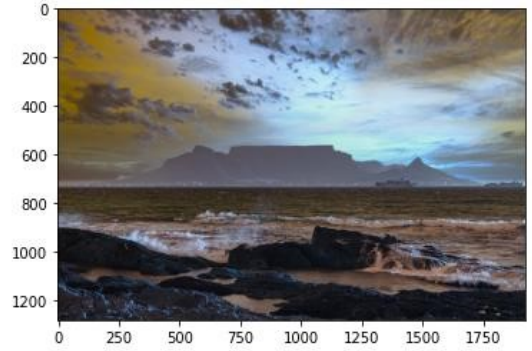
출처 : https://matplotlib.org/3.1.0/tutorials/colors/colormaps.html

이미지 파일 불러오기, 보여주기, 저장하기

- 예시 - 이미지 파일 불러와서 화면에 보여주기 (BGR, RGB)

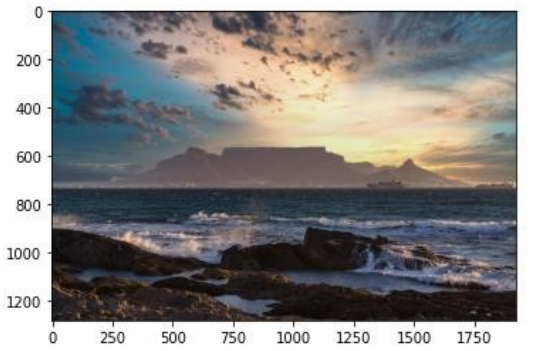
```
import cv2
import matplotlib.pyplot as plt
%matplotlib inline

# imread()를 이용하여 이미지 파일 불러오기
img = cv2.imread('./sunset.jpg')

print('이미지 파일 형태: {}'.format(img.shape))

# imshow()를 이용하여 이미지 파일 화면에 보여주기
plt.imshow(img)
plt.show()
```

```
이미지 파일 형태: (1280, 1920, 3)
```



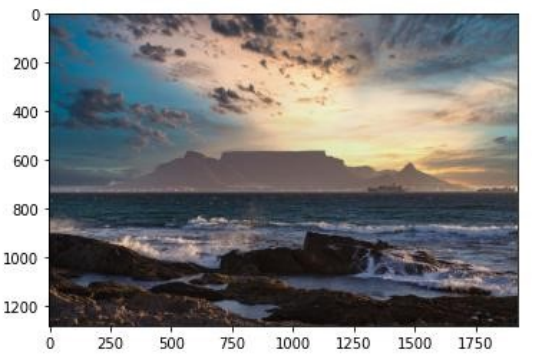

```
import cv2
import matplotlib.pyplot as plt
%matplotlib inline

img = cv2.imread('./sunset.jpg')

print('이미지 파일 형태: {}'.format(img.shape))

# BGR에서 RGB로 변경한 후, 화면에 보여주기
plt.imshow(img[:, :, ::-1])
plt.show()
```

```
이미지 파일 형태: (1280, 1920, 3)
```




```
import cv2
import matplotlib.pyplot as plt
%matplotlib inline

img = cv2.imread('./sunset.jpg')

# BGR에서 RGB로 변경
imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

print('이미지 파일 형태: {}'.format(imgRGB.shape))

plt.imshow(imgRGB)
plt.show()
```

```
이미지 파일 형태: (1280, 1920, 3)
```

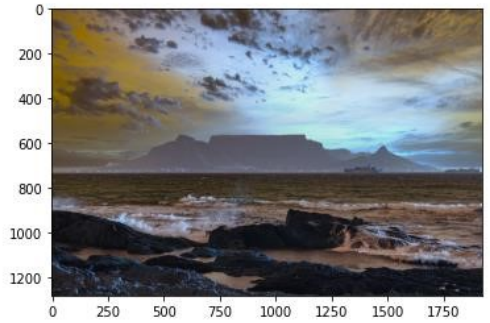


이미지 파일 불러오기, 보여주기, 저장하기

- 예시 - 이미지 파일 불러와서 화면에 보여주기 (Flag : Color, Unchanged, Grayscale)

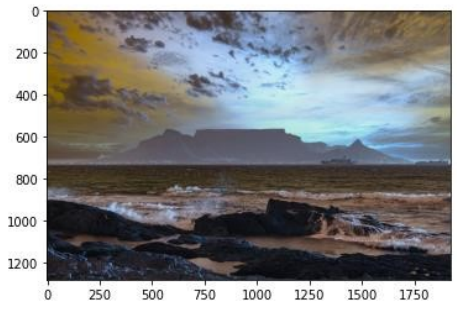
```
import cv2
import matplotlib.pyplot as plt
%matplotlib inline

# imread()를 이용하여 이미지 파일을 COLOR 옵션으로 불러오기
img = cv2.imread('./sunset.jpg', cv2.IMREAD_COLOR)

print('이미지 파일 형태: {}'.format(img.shape))

plt.imshow(img)
plt.show()
```

```
이미지 파일 형태: (1280, 1920, 3)
```



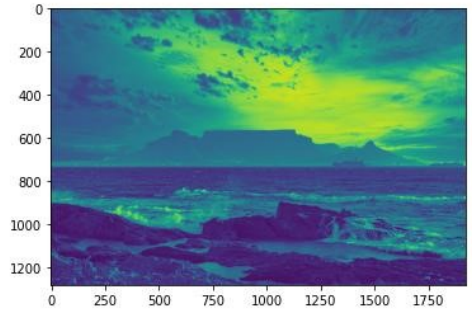

```
import cv2
import matplotlib.pyplot as plt
%matplotlib inline

# imread()를 이용하여 이미지 파일을 UNCHANGED 옵션으로 불러오기
img = cv2.imread('./sunset.jpg', cv2.IMREAD_UNCHANGED)

print('이미지 파일 형태: {}'.format(img.shape))

plt.imshow(img)
plt.show()
```

```
이미지 파일 형태: (1280, 1920, 3)
```




```
import cv2
import matplotlib.pyplot as plt
%matplotlib inline

# imread()를 이용하여 이미지 파일 GRAYSCALE 옵션으로 불러오기
img = cv2.imread('./sunset.jpg', cv2.IMREAD_GRAYSCALE)

print('이미지 파일 형태: {}'.format(img.shape))

plt.imshow(img)
plt.show()
```

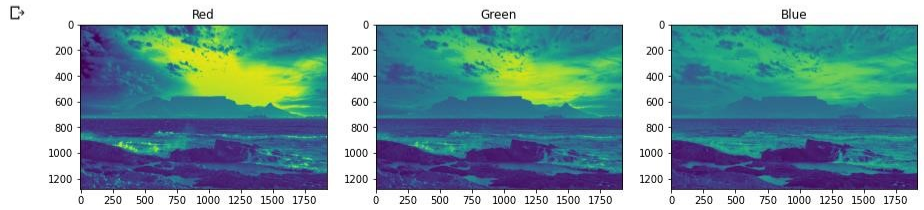
```
이미지 파일 형태: (1280, 1920)
```



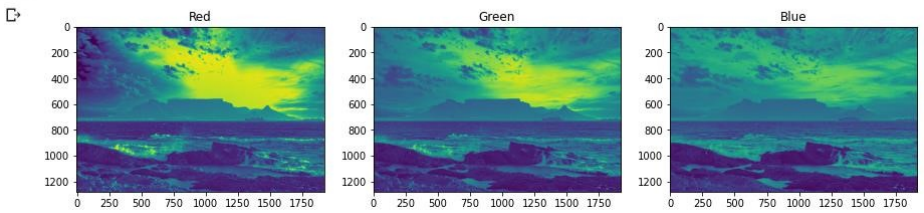
이미지 파일 불러오기, 보여주기, 저장하기

- 예시 - 이미지 파일 불러와서 화면에 보여주기 (BGR2RGB, Red, Green, Blue)

```python
import cv2
import matplotlib.pyplot as plt
%matplotlib inline

img = cv2.imread('./sunset.jpg')
imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

r = imgRGB[:,:,0]  # Red
g = imgRGB[:,:,1]  # Green
b = imgRGB[:,:,2]  # Blue

fig = plt.figure(figsize=(15, 10))

ax1 = fig.add_subplot(131)
ax1.set_title('Red')
ax1.imshow(r)

ax2 = fig.add_subplot(132)
ax2.set_title('Green')
ax2.imshow(g)

ax3 = fig.add_subplot(133)
ax3.set_title('Blue')
ax3.imshow(b)

plt.show()
```



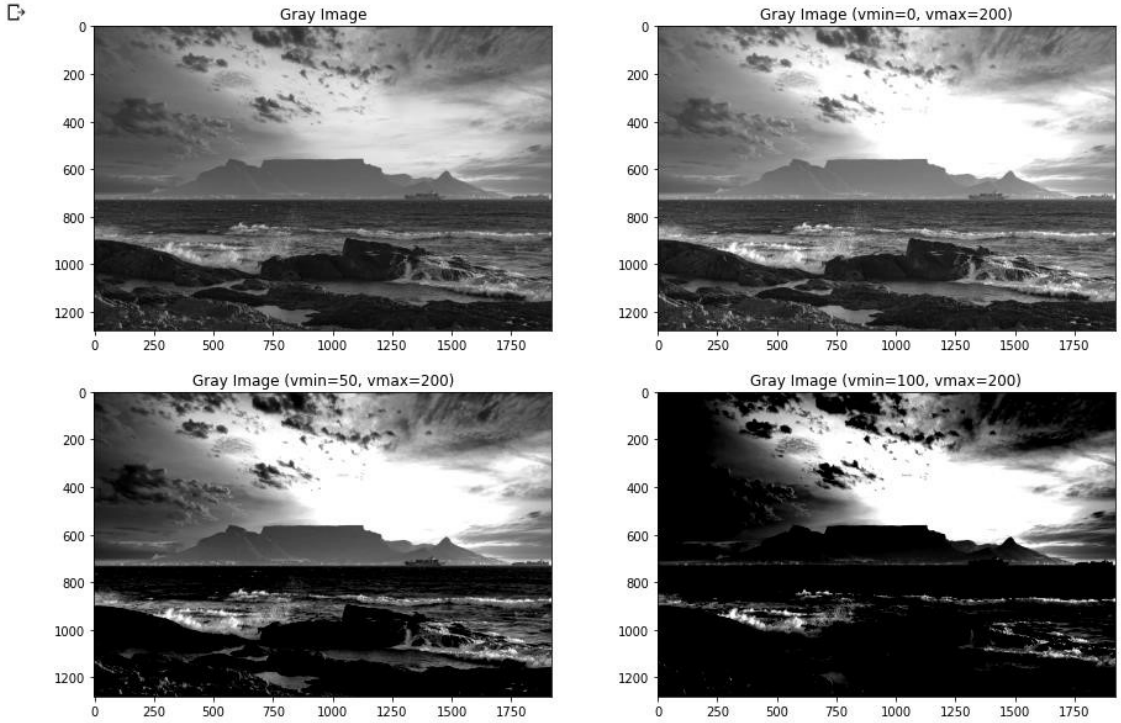

```python
import cv2
import matplotlib.pyplot as plt
%matplotlib inline

img = cv2.imread('./sunset.jpg')
imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

# split()을 사용하여 이미지의 채널을 r, g, b로 나눔
r, g, b = cv2.split(imgRGB)

fig = plt.figure(figsize=(15, 10))

ax1 = fig.add_subplot(131)
ax1.set_title('Red')
ax1.imshow(r)

ax2 = fig.add_subplot(132)
ax2.set_title('Green')
ax2.imshow(g)

ax3 = fig.add_subplot(133)
ax3.set_title('Blue')
ax3.imshow(b)

plt.show()
```



이미지 파일 불러오기, 보여주기, 저장하기

- 예시 - 이미지 파일 불러와서 화면에 보여주기 (BGR2GRAY, vmin, vmax)

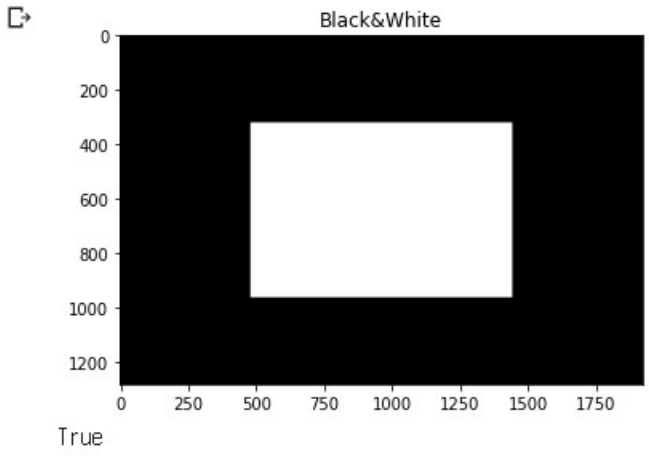
```
import cv2
import matplotlib.pyplot as plt
%matplotlib inline

img = cv2.imread('./sunset.jpg')

# BGR에서 GRAY로 변경
imgGRAY = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

fig = plt.figure(figsize=(15, 10))

ax1 = fig.add_subplot(221)
ax1.set_title('Gray Image')
ax1.imshow(imgGRAY, cmap='gray')

ax2 = fig.add_subplot(222)
ax2.set_title('Gray Image (vmin=0, vmax=200)')
ax2.imshow(imgGRAY, cmap='gray', vmin=0, vmax=200)

ax3 = fig.add_subplot(223)
ax3.set_title('Gray Image (vmin=50, vmax=200)')
ax3.imshow(imgGRAY, cmap='gray', vmin=50, vmax=200)

ax4 = fig.add_subplot(224)
ax4.set_title('Gray Image (vmin=100, vmax=200)')
ax4.imshow(imgGRAY, cmap='gray', vmin=100, vmax=200)

plt.show()
```



이미지 파일 불러오기, 보여주기, 저장하기

- 예시 - numpy을 이용하여 이미지 파일 생성 후, 저장하기 (검은색: 0, 흰색: 255)

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

col = 1280  # 열
row = 1920  # 행

# 모든 요소가 0(검은색)이고 (1280, 1920, 3) 형태인 배열 생성
img = np.zeros((col, row, 3), dtype=np.int)

for i in range(int(col * (1/4)), int(col * (3/4)) + 1):
  for j in range(int(row * (1/4)), int(row * (3/4)) + 1):
    # 0(검은색) -> 255(흰색)
    img[i][j] = 255

plt.title('Black&White')
plt.imshow(img)
plt.show()

# imwrite()를 이용하여 현재 폴더에 생성한 이미지 파일 저장
cv2.imwrite('./Black&White.jpg', img)
```



실습

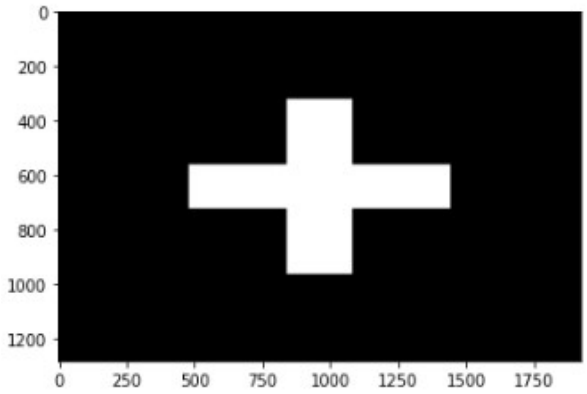
- 실습은 예시를 따라 하는 것으로 대체함

과제

- 소스 코드와 결과를 캡쳐하여 문서(워드, 한글)에 정리한 후, 스마트 캠퍼스에 제출
  - 제출 형식 : 교과목명_주차_학번_이름.docx or .hwp (제출 형식 안 맞을 시 감점)
  - 기간 내에 제출하지 못할 경우 이메일로 제출 (감점)
  - 주석 작성 필수 (주석 없을 시 감점)
  - 부정 행위 적발 시 감점 (컨닝)

과제1

- numpy를 이용하여 이미지 파일을 생성 후, 저장하시오.
  - (1280, 1920, 3) 형태의 배열
  - 십자가 위치는 중앙에 있도록 하고 적당한 크기로 생성 (검은색 : 0, 흰색 : 255)



과제2

- OpenCV를 이용하여 이미지 파일 불러오시오. (이미지 A)
  - 스마트 캠퍼스에 이미지 파일 (cat.jpg) 다운로드

- 이미지 A의 화소 값에서 최솟값과 최댓값을 구하시오.

과제2

- 이미지 A의 각 화소 값을 다음 식을 이용하여 값을 변환하시오. (이미지 B)
  - v = (u - min) / (max - min) * 255
  - u : 이미지 A 화소의 명암 값
  - v : 변환된 명암 값

- 이미지 B의 화소 값에서 최솟값과 최댓값을 구하시오.

과제2

- 아래 결과와 같이 1행 2열로 이미지 A와 B를 출력하시오.
  - 이미지 B 출력 시, vmin=0, vmax=255로 설정

감사합니다