파이썬 프로그래밍

(데이터 분석 및 시각화)

• 아나콘다 환경 설정 및 주피터 노트북 사용법

아나콘다란 수학, 과학관련 패키지들, 머신러닝이나 데이터 분석 등에 사용하는 여러 가지 패키지가 기본적으로 포함된 파이썬 배포판이다. 또한 아나콘다는 파이썬 가상 환경을 구축하는데도 유용하게 사용할 수 있다. 내부적으로 conda라는 환경/패키지 관리자가 존재하며 이 conda를 통해 패키지를 설치하거나 가상 환경을 관리할 수 있다.

• 아나콘다 설치

```
https://www.anaconda.com/
```

• 주피터 노트북 정의

웹브라우저(크롬) 환경에서 파이썬 코드를 작성하고 단계적으로 실행 가능하도록 하는 개발자 도구이다.

• 주피터 노트북 실행 및 폴더 생성

[Anaconda]-[Jupyter Notebook] 프로그램 실행한 후 [PythonDataWorkspace]-[Pandas] 폴더를 생성한다.

• 주피터 노트북 사용법

1) 초록색 : 편집모드
2) 파랑색 : 명령모드 (편집모드 상태에서 ESC)
3) 각 Cell의 Line Numbers : [View]-[Toggle Line Numbers]

- **'01.주피터 노트북 사용법.ipynb' 파일을 생성한다.**

```
print('Hello World')
print('환영합니다.')
name = '연탄이'
print(name)
print('Ctrl + Enter : 현재 Cell 실행')
print('Shift + Enter : 현재 Cell 실행 후 다음 Cell 선택')
print('Alt + Enter : 현재 Cell 실행 후 다음 위치에 Cell 삽입')
```

```
Hello World
환영합니다.
연탄이
Ctrl + Enter : 현재 Cell 실행
Shift + Enter : 현재 Cell 실행 후 다음 Cell 선택
Alt + Enter : 현재 Cell 실행 후 다음 위치에 Cell 삽입
```

- **클래스 + 함수 + '.' 입력한 후 'Tab' 키을 누르면 사용 가능한 명령어 목록이 나온다.**

```
import random
print(random.randint(1, 45))
```

- **함수 + '?' 입력한 후 'Shit' + 'Enter' 키를 누르면 설명이 출력된다.**

```
random.randint?
```

```
Signature: random.randint(a, b)
Docstring:
Return random integer in range [a, b], including both end points.
File:      /opt/conda/envs/anaconda-panel-2023.05-py310/lib/python3.11/random.py
Type:      method
```

- **함수 + '??' 입력한 후 'Shit' + 'Enter' 키를 누르면 source code와 설명이 출력된다.**

```
random.randint??
```

```
Signature: random.randint(a, b)
Docstring:
Return random integer in range [a, b], including both end points.
Source:
    def randint(self, a, b):
        """Return random integer in range [a, b], including both end points.
        """
        return self.randrange(a, b+1)
File:      /opt/conda/envs/anaconda-panel-2023.05-py310/lib/python3.11/random.py
Type:      method
```

- **문장 앞에 '#'을 입력하거나 'Ctrl' + '/' 키를 누르면 주석처리를 할 수 있다.**

```
# random.randint??
```

- **실행중인 경우 Number 대신 '*' 표시되고 [중지] 아이콘으로 실행중지가 가능하다.**

```
import time
for i in range(10):
    print(i)
    time.sleep(1)
```

- **마크다운 사용법**

마크다운이란 간단한 서식을 이용하여 일반 텍스트 문서의 양식을 편집하는 문법으로 HTML의 형태로 변환이 가능하다.

- **'1라인 이동' + '명령모드' + 'A' 키를 입력하여 한 행을 추가한 후 'M' 키를 입력하여 Mark Down을 변경한다.**

```
# Jupyter Notebook
주피터 노트북은 웹 브라우저 상에서 개발할 수 있는 도구이며, 코드를 Cell 단위로 묶어서 실행하고 그래프나 표, 그리고 이미지나 영상 등을 쉽게 볼 수 있어서 특히 데이터 관련 작업을 할 때 많이 활용됩니다.
> 교육을 위한 강의 노트로도 아주 훌륭해요!
```

Jupyter Notebook

주피터 노트북은 웹 브라우저 상에서 개발할 수 있는 도구이며, 코드를 Cell 단위로 묶어서 실행하고 그래프나 표, 그리고 이미지나 영상 등을 쉽게 볼 수 있어서 특히 데이터 관련 작업을 할 때 많이 활용됩니다.

> 교육을 위한 강의 노트로도 아주 훌륭해요!

- **마크다운으로 이미지 입력하기1 : [구글]-[Jupyter Notebook] 홈페이지를 화면 캡처하여 붙여넣기 한다.**

```
![image.png](attachment:image.png)
```

- **마크다운으로 이미지 입력하기2 : [Edit]-[Insert Image] 메뉴에서 입력할 이미지를 선택한다.**

```
![a.png](attachment:a.png)
```

- **'#'의 개수에 따라 글자 크기가 결정되고 목록 작성을 작성하려면 숫자 '1' 키를 입력한 후 문장을 작성한다.**

```
## 특징
1. 코드를 Cell 단위로 작성 및 실행
1. 마크다운을 통한 문서화
1. 그래프나 표 등을 실시간으로 확인
1. html, pdf등 파일로 저장
```

특징

1. 코드를 Cell 단위로 작성밑 실행
2. 마크다운을 통한 문서화
3. 크래프나 표 등을 실시간으로 확인
4. html, pdf등 파일로 저장

- **'---' 키를 누르면 수평선이 출력되며 '-' 키를 입력하면 목록을 작성 한다.**

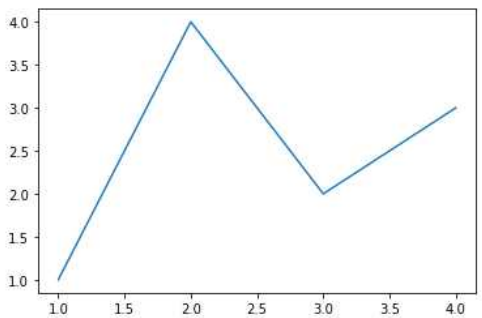
```
---
## 시각화 예제 1: 그래프
- matplotlib를 이용하면 다양한 그래프를 그릴 수 있으며 수행 결과를 바로 확인 할 수 있다.
```

시각화 예제 1: 그래프

- matplotlib를 이용하면 다양한 그래프를 그릴수 있으며 수행 결과를 바로 확인 할 수 있다.

- **[구글]-[matplotlig 검색]-[Tutorials]-[Basic Usage] 기본 그래프 코드를 복사하여 그래프를 출력해 본다.**

```
import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np
fig, ax = plt.subplots()  # Create a figure containing a single axes.
ax.plot([1, 2, 3, 4], [1, 4, 2, 3]);  # Plot some data on the axes.
```



- [구글]-[pandas 검색]-[User Guide]-[Table Visualization] 테이블 작성 코드를 복사하여 테이블을 출력해 본다.

```
import pandas as pd
import numpy as np
import matplotlib as mpl

df = pd.DataFrame([[38.0, 2.0, 18.0, 22.0, 21, np.nan],[19, 439, 6, 452, 226,232]],
                  index=pd.Index(['Tumour (Positive)', 'Non-Tumour (Negative)'], name='Actual Label:'),
                  columns=pd.MultiIndex.from_product(
          [['Decision Tree', 'Regression', 'Random'],['Tumour', 'Non-Tumour']], names=['Model:', 'Predicted:']))
df.style
```

Model:	Decision Tree		Regression		Random	
Predicted:	Tumour	Non-Tumour	Tumour	Non-Tumour	Tumour	Non-Tumour
Actual Label:						
Tumour (Positive)	38.000000	2.000000	18.000000	22.000000	21	nan
Non-Tumour (Negative)	19.000000	439.000000	6.000000	452.000000	226	232.000000

- 하이퍼 링크 텍스트를 이용하려면 '[링크단어]' + '(링크주소)'를 입력한다.

```
## 사전 학습 자료
다음 링크에서 파이썬 기본편에 대한 학습을 하실 수 있습니다. space 두 칸 입력
- [나도 코딩 블로그](https://nadocoding.tistory.com)
```

> **사전 학습 자료**
>
> 다음 링크에서 파이썬 기본편에 대한 학습을 하실 수 있습니다.
> 나도 코딩 블로그

- [유튜브]-[나도코딩 python 기본편 검색]-[오른쪽 마우스]-[소스코드복사] 메뉴를 선택하여 복사하여 'Code' 상태로 변경(Markdown인 경우 명령모드에서 후 'Y')한 후 상단에 '%%HTML' 입력 후 실행해 본다.

```
%%HTML
<iframe width="640" height="360"
src="https://www.youtube.com/embed/kWiCuklohdY"
title="YouTube video player" frameborder="0"
allow="accelerometer; autoplay;clipboard-write; encrypted-media;gyroscope;picture-in-picture" allowfullscreen>
</iframe>
```

- Line Number 재 정렬 : [Kernel]-[Restart & Run All]-[Restart and Run All Cells] 버튼 선택
- HTML 저장 : [File]-[Download as] - [HTML(*.htm)] 다운로드 폴더에 저장된다.
- Jupyter 파일 저장 : [c:]-[사용자]-[사용자 계정]에 확장자 IPYNB(Interactive PYthon NoteBook)로 저장된다.
- Jupyter 파일 열기 : [File]-[Open] 메뉴에서 파일을 선택한가.
- Jupyter 종료 : [Ctrl + S]로 저장-[File]-[Close and Halt]-[명령창]-[Ctrl + C, C]

• 판다스(pandas)

판다스(pandas)란 데이터 조작과 분석을 위한 파이썬(Python) 소프트웨어 라이브러리이다.

01. 시리즈(Series)

Series는 Indexing된 데이터의 1차원 배열이다. Numpy는 명시적 인덱스 없이 묵시적으로 '0'부터 차례로 정수형 인덱스를 사용하여 접근하지만 Series는 명시적으로 정의된 인덱스가 존재하기 때문에 이 명시적 인덱스를 사용할 수 있다.

- **새로운 파일 '01.Series.ipynb'를 생성한다.**

- **1월부터 4월까지 평균 온도 데이터(-20, -10, 10, 20)를 입력 후 출력을 한다. print문은 생략이 가능하다.**

```
import pandas as pd
temp = pd.Series([-20, -10, 10, 20])
print(temp)
```

```
0   -20
1   -10
2    10
3    20
dtype: int64
```

- **index값 '0'에 저장되어 있는 데이터를 출력한다.**

```
temp[0]
```

```
-20
```

- **Series 객체를 생성할 경우 특정 index를 지정할 수 있다.**

```
temp = pd.Series([-20, -10, 10, 20], index=['Jan', 'Feb', 'Mar', 'Apr'])
temp
```

```
Jan   -20
Feb   -10
Mar    10
Apr    20
dtype: int64
```

- **index Apr(4월)에 해당하는 데이터를 출력한다.**

```
temp['Apr']
```

```
20
```

- **존재하지 않는 Index에 접근을 시도할 경우 에러가 발생 한다.**

```
temp['Jun']
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
...
KeyError: 'Jun'
The above exception was the direct cause of the following exception:
...
```

02. 데이터프레임(DataFrame)

데이터프레임(DataFrame)은 2차원 배열 데이터 즉 시리즈(Series)들의 모임을 의미한다.

- **새로운 파일 '02.DataFrame.ipynb'를 생성한다.**

- **슬램덩크 주요 인물 8명에 대한 데이터 (dictionary 타입으로 저장 한다.)**

```
data = {
    '이름' : ['채치수', '정대만', '송태섭', '서태웅', '강백호', '변덕규', '황태산', '윤대협'],
    '학교' : ['북산고', '북산고', '북산고', '북산고', '북산고', '능남고', '능남고', '능남고'],
    '키' : [197, 184, 168, 187, 188, 202, 188, 190],
    '국어' : [90, 40, 80, 40, 15, 80, 55, 100],
    '영어' : [85, 35, 75, 60, 20, 100, 65, 85],
    '수학' : [100, 50, 70, 70, 10, 95, 45, 90],
    '과학' : [95, 55, 80, 75, 35, 85, 40, 95],
    '사회' : [85, 25, 75, 80, 10, 80, 35, 95],
    'SW특기' : ['Python', 'Java', 'Javascript', '', '', 'C', 'PYTHON', 'C#']
}
data
```

```
{'이름': ['채치수', '정대만', '송태섭', '서태웅', '강백호', '변덕규', '황태산', '윤대협'],
 '학교': ['북산고', '북산고', '북산고', '북산고', '북산고', '능남고', '능남고', '능남고'],
 '키': [197, 184, 168, 187, 188, 202, 188, 190],
 '국어': [90, 40, 80, 40, 15, 80, 55, 100],
 '영어': [85, 35, 75, 60, 20, 100, 65, 85],
 '수학': [100, 50, 70, 70, 10, 95, 45, 90],
 '과학': [95, 55, 80, 75, 35, 85, 40, 95],
 '사회': [85, 25, 75, 80, 10, 80, 35, 95],
 'SW특기': ['Python', 'Java', 'Javascript', '', '', 'C', 'PYTHON', 'C#']}
```

```
data['키'], type(data)
```

```
([197, 184, 168, 187, 188, 202, 188, 190], dict)
```

```
data['이름'], type(data['이름'])
```

```
(['채치수', '정대만', '송태섭', '서태웅', '강백호', '변덕규', '황태산', '윤대협'], list)
```

- **판다스(pandas) 라이브러리를 import한 후 dictionary(data) 객체를 DataFrame 객체(df)로 생성한다.**

```
import pandas as pd
df = pd.DataFrame(data)
df
```

	이름	학교	키	국어	영어	수학	과학	사회	SW특기
0	채치수	북산고	197	90	85	100	95	85	Python
1	정대만	북산고	184	40	35	50	55	25	Java
2	송태섭	북산고	168	80	75	70	80	75	Javascript
3	서태웅	북산고	187	40	60	70	75	80	
4	강백호	북산고	188	15	20	10	35	10	
5	변덕규	능남고	202	80	100	95	85	80	C
6	황태산	능남고	188	55	65	45	40	35	PYTHON
7	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)에서 '이름' 칼럼 데이터는 시리즈(Series) 타입으로 출력된다.**

```
df['이름']
```

```
0    채치수
1    정대만
2    송태섭
3    서태웅
4    강백호
5    변덕규
6    황태산
7    윤대협
Name: 이름, dtype: object
```

```
type(df['이름'])
```

```
pandas.core.series.Series
```

```
df['이름'].values
```

```
array(['채치수', '정대만', '송태섭', '서태웅', '강백호', '변덕규', '황태산', '윤대협'],
      dtype=object)
```

- **DataFrame(df)에서 '키' 칼럼 데이터는 시리즈(Series) 타입으로 출력된다.**

```
df['학교']
```

```
0    북산고
1    북산고
2    북산고
3    북산고
4    북산고
5    능남고
6    능남고
7    능남고
Name: 학교, dtype: object
```

- **DataFrame(df)에서 '학교', '이름', '키' 세 개의 칼럼은 2차원 배열로 출력한다.**

```
df[['학교', '이름', '키']]
```

	학교	이름	키
0	북산고	채치수	197
1	북산고	정대만	184
2	북산고	송태섭	168
3	북산고	서태웅	187
4	북산고	강백호	188
5	능남고	변덕규	202
6	능남고	황태산	188
7	능남고	윤대협	190

```
df[['학교', '이름', '키']].shape
```

```
(8, 3)
```

- **DataFrame(df)에서 Index를 '1번', '2번', '3번' ... 으로 변경하여 저장한다.**

```
df = pd.DataFrame(data, index=['1번', '2번', '3번', '4번', '5번', '6번', '7번', '8번'])
df
```

	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	
5번	강백호	북산고	188	15	20	10	35	10	
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **data 중에서 원하는 column만 선택하여 순서를 변경한 후 DataFrame(df)에 새로 저장한다.**

```
df = pd.DataFrame(data, columns=['이름', '학교', '키'])
df
```

	이름	학교	키
0	채치수	북산고	197
1	정대만	북산고	184
2	송태섭	북산고	168
3	서태웅	북산고	187
4	강백호	북산고	188
5	변덕규	능남고	202
6	황태산	능남고	188
7	윤대협	능남고	190

```
df = pd.DataFrame(data, columns=['이름', '키', '학교'])
df
```

	이름	키	학교
0	채치수	197	북산고
1	정대만	184	북산고
2	송태섭	168	북산고
3	서태웅	187	북산고
4	강백호	188	북산고
5	변덕규	202	능남고
6	황태산	188	능남고
7	윤대협	190	능남고

03. 인덱스(Index)

인덱스(index)란 데이터에 접근할 수 있는 주소 값을 의미한다.

- **새로운 파일 '03.Index.ipynb'를 생성한다.**

- **변수(list) 세 개의 값을 저장한 후 데이터와 데이터의 type를 출력한다.**

```
list = ['유재석', '하하', '정형돈']
list, type(list)
```

```
(['유재석', '하하', '정형돈'], list)
```

```
list[1]
```

```
'하하'
```

```
list[0:2]
```

```
['유재석', '하하']
```

```
list[-1]
```

```
'정형돈
```

- **Pandas 라이브러리를 import하여 data에 딕션너리(dictionary) 형태로 저장한 후 DataFrame 객체로 생성한다.**

```
import pandas as pd
data = {
    '이름' : ['채치수', '정대만', '송태섭', '서태웅', '강백호', '변덕규', '황태산', '윤대협'],
    '학교' : ['북산고', '북산고', '북산고', '북산고', '북산고', '능남고', '능남고', '능남고'],
    '키' : [197, 184, 168, 187, 188, 202, 188, 190],
    '국어' : [90, 40, 80, 40, 15, 80, 55, 100],
    '영어' : [85, 35, 75, 60, 20, 100, 65, 85],
    '수학' : [100, 50, 70, 70, 10, 95, 45, 90],
    '과학' : [95, 55, 80, 75, 35, 85, 40, 95],
    '사회' : [85, 25, 75, 80, 10, 80, 35, 95],
    'SW특기' : ['Python', 'Java', 'Javascript', '', '', 'C', 'PYTHON', 'C#']
}
df = pd.DataFrame(data, index = ['1번', '2번', '3번', '4번', '5번', '6번', '7번', '8번'])
df
```

	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	
5번	강백호	북산고	188	15	20	10	35	10	
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)의 index 이름들을 출력해 본다.**

```
df.index
```

```
Index(['1번', '2번', '3번', '4번', '5번', '6번', '7번', '8번'], dtype='object')
```

- **DataFrame(df)의 Index 칼럼 이름을 '지원번호'로 새로 설정한다.**

```
df.index.name = '지원번호'
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript

- **DataFrame(df)의 Index 칼럼 값들을 원리 값으로 초기화 한다.**

```
df.reset_index()
```

	지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
0	1번	채치수	북산고	197	90	85	100	95	85	Python
1	2번	정대만	북산고	184	40	35	50	55	25	Java
2	3번	송태섭	북산고	168	80	75	70	80	75	Javascript
3	4번	서태웅	북산고	187	40	60	70	75	80	
4	5번	강백호	북산고	188	15	20	10	35	10	
5	6번	변덕규	능남고	202	80	100	95	85	80	C
6	7번	황태산	능남고	188	55	65	45	40	35	PYTHON
7	8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)에 추가되었던 칼럼(저원번호)을 원래대로 삭제한다.**

```
df.reset_index(drop=True)
```

	이름	학교	키	국어	영어	수학	과학	사회	SW특기
0	채치수	북산고	197	90	85	100	95	85	Python
1	정대만	북산고	184	40	35	50	55	25	Java
2	송태섭	북산고	168	80	75	70	80	75	Javascript
3	서태웅	북산고	187	40	60	70	75	80	
4	강백호	북산고	188	15	20	10	35	10	
5	변덕규	능남고	202	80	100	95	85	80	C
6	황태산	능남고	188	55	65	45	40	35	PYTHON
7	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFram(df)을 출력하면 index(지원번호)가 다시 출력된다.**

```
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	
5번	강백호	북산고	188	15	20	10	35	10	
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df) 데이터에 바로 적용하려면 inplace 옵션을 사용한다.**

```
df.reset_index(drop=True, inplace=True)
df
```

	이름	학교	키	국어	영어	수학	과학	사회	SW특기
0	채치수	북산고	197	90	85	100	95	85	Python
1	정대만	북산고	184	40	35	50	55	25	Java
2	송태섭	북산고	168	80	75	70	80	75	Javascript
3	서태웅	북산고	187	40	60	70	75	80	
4	강백호	북산고	188	15	20	10	35	10	
5	변덕규	능남고	202	80	100	95	85	80	C
6	황태산	능남고	188	55	65	45	40	35	PYTHON
7	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)의 지정한 column('이름')으로 Index를 설정 한다.**

```
df.set_index('이름')
```

이름	학교	키	국어	영어	수학	과학	사회	SW특기
채치수	북산고	197	90	85	100	95	85	Python
정대만	북산고	184	40	35	50	55	25	Java
송태섭	북산고	168	80	75	70	80	75	Javascript
서태웅	북산고	187	40	60	70	75	80	
강백호	북산고	188	15	20	10	35	10	
변덕규	능남고	202	80	100	95	85	80	C
황태산	능남고	188	55	65	45	40	35	PYTHON
윤대협	능남고	190	100	85	90	95	95	C#

```
df.set_index('이름', inplace=True)
df
```

이름	학교	키	국어	영어	수학	과학	사회	SW특기
채치수	북산고	197	90	85	100	95	85	Python
정대만	북산고	184	40	35	50	55	25	Java
송태섭	북산고	168	80	75	70	80	75	Javascript
서태웅	북산고	187	40	60	70	75	80	
강백호	북산고	188	15	20	10	35	10	
변덕규	능남고	202	80	100	95	85	80	C
황태산	능남고	188	55	65	45	40	35	PYTHON
윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)에서 Index('이름')을 기준으로 오름차순 정렬을 한다.**

```
df.sort_index()
```

이름	학교	키	국어	영어	수학	과학	사회	SW특기
강백호	북산고	188	15	20	10	35	10	
변덕규	능남고	202	80	100	95	85	80	C
서태웅	북산고	187	40	60	70	75	80	
송태섭	북산고	168	80	75	70	80	75	Javascript
윤대협	능남고	190	100	85	90	95	95	C#
정대만	북산고	184	40	35	50	55	25	Java
채치수	북산고	197	90	85	100	95	85	Python
황태산	능남고	188	55	65	45	40	35	PYTHON

- **DataFrame(df)에서 Index('이름')를 기준으로 내림차순 정렬을 한다.**

```
df.sort_index(ascending=False)
```

이름	학교	키	국어	영어	수학	과학	사회	SW특기
황태산	능남고	188	55	65	45	40	35	PYTHON
채치수	북산고	197	90	85	100	95	85	Python
정대만	북산고	184	40	35	50	55	25	Java
윤대협	능남고	190	100	85	90	95	95	C#
송태섭	북산고	168	80	75	70	80	75	Javascript
서태웅	북산고	187	40	60	70	75	80	
변덕규	능남고	202	80	100	95	85	80	C
강백호	북산고	188	15	20	10	35	10	

04. 파일 저장 및 열기

- 새로운 파일 '04.파일 저장 및 열기.ipynb'를 생성한다.

```
import pandas as pd
data = {
    '이름' : ['채치수', '정대만', '송태섭', '서태웅', '강백호', '변덕규', '황태산', '윤대협'],
    '학교' : ['북산고', '북산고', '북산고', '북산고', '북산고', '능남고', '능남고', '능남고'],
    '키' : [197, 184, 168, 187, 188, 202, 188, 190],
    '국어' : [90, 40, 80, 40, 15, 80, 55, 100],
    '영어' : [85, 35, 75, 60, 20, 100, 65, 85],
    '수학' : [100, 50, 70, 70, 10, 95, 45, 90],
    '과학' : [95, 55, 80, 75, 35, 85, 40, 95],
    '사회' : [85, 25, 75, 80, 10, 80, 35, 95],
    'SW특기' : ['Python', 'Java', 'Javascript', '', '', 'C', 'PYTHON', 'C#']
}

df = pd.DataFrame(data, index = ['1번', '2번', '3번', '4번', '5번', '6번', '7번', '8번'])
df.index.name = '지원번호'
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	
5번	강백호	북산고	188	15	20	10	35	10	
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- DataFrame(df)를 csv 파일로 저장 한다.

```
df.to_csv('score.csv', encoding='utf-8-sig')
```

- DataFrame(df)를 지원번호를 생략하고 csv 파일로 저장한다.

```
df.to_csv('score.csv', encoding='utf-8-sig', index=False)
```

- DataFrame(df)을 텍스트(.txt) 파일로 저장한다. (칼럼을 'Tab'으로 구분한다.)

```
df.to_csv('score.txt', sep='\t')
```

- DataFrame(df)을 엑셀(.xlsx) 파일로 저장한다.

```
df.to_excel('score.xlsx')
```

- 'score.csv' 파일을 읽어 DataFrame(df)에 저장한다.

```
df = pd.read_csv('score.csv')
df
```

	이름	학교	키	국어	영어	수학	과학	사회	SW특기
0	채치수	북산고	197	90	85	100	95	85	Python
1	정대만	북산고	184	40	35	50	55	25	Java
2	송태섭	북산고	168	80	75	70	80	75	Javascript
3	서태웅	북산고	187	40	60	70	75	80	NaN
4	강백호	북산고	188	15	20	10	35	10	NaN
5	변덕규	능남고	202	80	100	95	85	80	C
6	황태산	능남고	188	55	65	45	40	35	PYTHON
7	윤대협	능남고	190	100	85	90	95	95	C#

- 'score.csv' 파일을 읽어 상위 1개의 ROW를 제외(Skip)하고 DataFrame(df)에 저장한다.

```
df = pd.read_csv('score.csv', skiprows=1)
df
```

	채치수	북산고	197	90	85	100	95	85.1	Python
0	정대만	북산고	184	40	35	50	55	25	Java
1	송태섭	북산고	168	80	75	70	80	75	Javascript
2	서태웅	북산고	187	40	60	70	75	80	NaN
3	강백호	북산고	188	15	20	10	35	10	NaN
4	변덕규	능남고	202	80	100	95	85	80	C
5	황태산	능남고	188	55	65	45	40	35	PYTHON
6	윤대협	능남고	190	100	85	90	95	95	C#

- 'score.csv' 파일의 1, 3, 5 ROW들을 제외(Skip)하고 DataFrame(df)에 저장한다.

```
df = pd.read_csv('score.csv', skiprows=[1, 3, 5])
df
```

	이름	학교	키	국어	영어	수학	과학	사회	SW특기
0	정대만	북산고	184	40	35	50	55	25	Java
1	서태웅	북산고	187	40	60	70	75	80	NaN
2	변덕규	능남고	202	80	100	95	85	80	C
3	황태산	능남고	188	55	65	45	40	35	PYTHON
4	윤대협	능남고	190	100	85	90	95	95	C#

- 'score.csv' 파일의 지정된 수만큼의 ROW만 가져와서 DataFrame(df)에 저장한다.

```
df = pd.read_csv('score.csv', nrows=4)
df
```

	이름	학교	키	국어	영어	수학	과학	사회	SW특기
0	채치수	북산고	197	90	85	100	95	85	Python
1	정대만	북산고	184	40	35	50	55	25	Java
2	송태섭	북산고	168	80	75	70	80	75	Javascript
3	서태웅	북산고	187	40	60	70	75	80	NaN

- 'score.csv' 파일의 처음 2 ROW는 무시하고 이후에 4 ROW를 가져와서 DataFrame(df)에 저장한다.

```
df = pd.read_csv('score.csv', skiprows=2, nrows=4)
df
```

	정대만	북산고	184	40	35	50	55	25	Java
0	송태섭	북산고	168	80	75	70	80	75	Javascript
1	서태웅	북산고	187	40	60	70	75	80	NaN
2	강백호	북산고	188	15	20	10	35	10	NaN
3	변덕규	능남고	202	80	100	95	85	80	C

- 'score.txt' 텍스트 파일을 'TAB'을 구분자로 읽어 DataFrame(df)에 저장한다.

```
df = pd.read_csv('score.txt', sep="\t")
df
```

	지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
0	1번	채치수	북산고	197	90	85	100	95	85	Python
1	2번	정대만	북산고	184	40	35	50	55	25	Java
2	3번	송태섭	북산고	168	80	75	70	80	75	Javascript
3	4번	서태웅	북산고	187	40	60	70	75	80	NaN

- 'score.txt' 파일을 '지원번호' 칼럼을 인덱스 칼럼으로 지정하여 DataFrme(df)에 저장한다.

```
df = pd.read_csv('score.txt', sep="\t", index_col='지원번호')
df
```

	이름	학교	키	국어	영어	수학	과학	사회	SW특기
지원번호									
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN

- 'score.txt' 파일을 읽어 DataFrame(df)에 저장한 후 지원번호를 인덱스를 지정하면 앞의 결과와 같다.

```
df = pd.read_csv('score.txt', sep="\t")
df.set_index('지원번호', inplace=True)
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- 'score.xlsx' 엑셀 파일을 읽어 DataFrame(df)에 저장한다.

```
df = pd.read_excel('score.xlsx')
df
```

	지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
0	1번	채치수	북산고	197	90	85	100	95	85	Python
1	2번	정대만	북산고	184	40	35	50	55	25	Java
2	3번	송태섭	북산고	168	80	75	70	80	75	Javascript
3	4번	서태웅	북산고	187	40	60	70	75	80	NaN
4	5번	강백호	북산고	188	15	20	10	35	10	NaN
5	6번	변덕규	능남고	202	80	100	95	85	80	C
6	7번	황태산	능남고	188	55	65	45	40	35	PYTHON
7	8번	윤대협	능남고	190	100	85	90	95	95	C#

- 'score.xlsx' 엑셀 파일을 읽어 index를 지원번호로 지정한 후 데이터를 읽어온다.

```
df = pd.read_excel('score.xlsx', index_col='지원번호')
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

05. 데이터 확인

- 새로운 파일 '05.데이터 확인.ipynb'를 생성한 후 DataFrame(df)에 저장한다.

```
import pandas as pd
df = pd.read_excel('score.xlsx', index_col='지원번호')
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- 계산 가능한 데이터에 대해 Column 별로 데이터의 수, 평균, 표준편차, 최솟값, 최댓값 등의 정보를 보여줌

```
df.describe()
```

	키	국어	영어	수학	과학	사회
count	8.000000	8.000000	8.000000	8.000000	8.000000	8.000000
mean	188.000000	62.500000	65.625000	66.250000	70.000000	60.625000
std	9.985704	29.519969	26.917533	30.325614	23.754699	32.120032
min	168.000000	15.000000	20.000000	10.000000	35.000000	10.000000
25%	186.250000	40.000000	53.750000	48.750000	51.250000	32.500000
50%	188.000000	67.500000	70.000000	70.000000	77.500000	77.500000
75%	191.750000	82.500000	85.000000	91.250000	87.500000	81.250000
max	202.000000	100.000000	100.000000	100.000000	95.000000	95.000000

- DataFrame(df)의 각 칼럼별로 데이터 타입을 확인할 수 있다.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 8 entries, 1번 to 8번
Data columns (total 9 columns):
#   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   이름        8 non-null      object
 1   학교        8 non-null      object
 2   키         8 non-null      int64
 3   국어        8 non-null      int64
 4   영어        8 non-null      int64
 5   수학        8 non-null      int64
 6   과학        8 non-null      int64
 7   사회        8 non-null      int64
 8   SW특기      6 non-null      object
dtypes: int64(6), object(3)
memory usage: 640.0+ bytes
```

- **DataFramd(df)의 처음 5개의 ROW를 가지고 온다.**

```
df.head()
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN

- **DataFrame(df)의 처음 4개의 ROW를 자기고 온다.**

```
df.head(4)
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN

- **DataFrame(df)의 마지막 5개의 ROW를 가지고 온다.**

```
df.tail()
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)의 마지막 3개의 ROW를 가지고 온다.**

```
df.tail(3)
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)의 값들을 2차원 배열로 출력한다.**

```
df.values
```

```
array([
       ['채치수', '북산고', 197, 90, 85, 100, 95, 85, 'Python'],
       ['정대만', '북산고', 184, 40, 35, 50, 55, 25, 'Java'],
       ['송태섭', '북산고', 168, 80, 75, 70, 80, 75, 'Javascript'],
       ['서태웅', '북산고', 187, 40, 60, 70, 75, 80, nan],
       ['강백호', '북산고', 188, 15, 20, 10, 35, 10, nan],
       ['변덕규', '능남고', 202, 80, 100, 95, 85, 80, 'C'],
       ['황태산', '능남고', 188, 55, 65, 45, 40, 35, 'PYTHON'],
       ['윤대협', '능남고', 190, 100, 85, 90, 95, 95, 'C#']
], dtype=object)
```

- **DataFrame(df)의 index 정보가 출력된다.**

```
df.index
```

```
Index(['1번', '2번', '3번', '4번', '5번', '6번', '7번', '8번'], dtype='object', name='지원번호')
```

- **DataFrame(df)의 칼럼(Column) 정보가 출력된다.**

```
df.columns
```

```
Index(['이름', '학교', '키', '국어', '영어', '수학', '과학', '사회', 'SW특기'], dtype='object')
```

- **DataFrame(df)가 몇 개의 Row와 Column이 있는지 출력된다.**

```
df.shape
```

```
(8, 9)
```

- **1차원 Series 확인(일차원 배열) 키에 대한 모든 정보를 출력한다.**

```
df['키'].describe()
```

```
count    8.000000
mean     188.000000
std      9.985704
min      168.000000
25%      186.250000
50%      188.000000
75%      191.750000
max      202.000000
Name: 키, dtype: float64
```

- DataFrame(df)의 '키' 칼럼 데이터 값 중 최솟값을 출력한다.

```
df['키'].min()
```

```
168
```

- DataFrame(df)의 '키' 데이터 값 중 최댓값을 출력한다.

```
df['키'].max()
```

```
202
```

- DataFrame(df)의 '키'가 큰사람 순서대로 3명의 데이터를 출력한다.

```
df['키'].nlargest(3)
```

```
지원번호
6번    202
1번    197
8번    190
Name: 키, dtype: int64
```

- DataFrame(df)의 '키' 칼럼들의 평균값을 출력한다.

```
df['키'].mean()
```

```
188.0
```

- DataFramd(df)의 모든 '키'에 대한 합계를 출력한다.

```
df['키'].sum()
```

```
1504
```

- DataFrame(df) 데이터들 중 'SW특기'가 있는 사람들이 몇 명인지 출력한다.

```
df['SW특기'].count()
```

```
6
```

- DataFrame(df)의 데이터들을 학교이름 중복을 제거하고 출력한다.

```
df['학교'].unique()
```

```
array(['북산고', '능남고'], dtype=object)
```

- DataFrame(df)의 데이터들이 전체 몇 개의 학교에 속해있는지를 출력한다.

```
df['학교'].nunique()
```

```
2
```

06. 데이터 선택(기본)

- **새로운 파일 '06.데이터 선택(기본).ipynb'를 생성한 후 'score.xlsx' 파일을 읽어 DataFrame에 저장한다.**

```
import pandas as pd
df = pd.read_excel('score.xlsx', index_col='지원번호')
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)에서 모든 인덱스에 대하여 칼럼이 '이름'인 데이터를 출력한다.**

```
df['이름']
```

```
지원번호
1번    채치수
2번    정대만
3번    송태섭
4번    서태웅
5번    강백호
6번    변덕규
7번    황태산
8번    윤대협
Name: 이름, dtype: object
```

- **DataFrame(df) 모든 인덱스에 대한 '키' 칼럼의 데이터를 출력한다.**

```
df['키']
```

```
지원번호
1번    197
2번    184
3번    168
4번    187
5번    188
6번    202
7번    188
8번    190
Name: 키, dtype: int64
```

- **DataFrame(df)의 모든 인덱스에 대한 '이름'과 '키' 칼럼을 2차원 배열로 출력한다.**

```
df[['이름','키']]
```

	이름	키
지원번호		
1번	채치수	197
2번	정대만	184
3번	송태섭	168
4번	서태웅	187
5번	강백호	188
6번	변덕규	202
7번	황태산	188
8번	윤대협	190

- **DataFrame(df)의 칼럼 이름들을 출력한다.**

```
df.columns
```

```
Index(['이름', '학교', '키', '국어', '영어', '수학', '과학', '사회', 'SW특기'], dtype='object')
```

- **DataFrame(df)에서 인덱스가 0인 칼럼의 이름을 출력한다.**

```
df.columns[0]
```

```
'이름'
```

- **DataFrame(df)에서 인덱스가 2인 칼럼의 이름을 출력한다.**

```
df.columns[2]
```

```
'키'
```

- **DataFrame(df)의 인덱스0의 모든 칼럼 데이터들을 출력한다. df['이름']과 동일한 결과가 출력된다.**

```
df[df.columns[0]]
```

```
지원번호
1번    채치수
2번    정대만
3번    송태섭
4번    서태웅
5번    강백호
6번    변덕규
7번    황태산
8번    윤대협
Name: 이름, dtype: object
```

- **DataFrame(df)의 인덱스 마지막의 모든 칼럼을 출력하고 칼럼수가 많은 경우 유용하다.**

```
df[df.columns[-1]]
```

```
지원번호
1번          Python
2번            Java
3번      Javascript
4번             NaN
5번             NaN
6번               C
7번          PYTHON
8번              C#
Name: SW특기, dtype: object
```

- **DataFrame(df)의 '영어' 칼럼에서 0~4 ROW 데이터를 출력한다.**

```
df['영어'][0:5]
```

```
지원번호
1번    85
2번    35
3번    75
4번    60
5번    20
Name: 영어, dtype: int64
```

- **DataFrame(df)의 '이름', '키' 칼럼들을 인덱스 처음부터 5개의 데이터를 출력한다.**

```
df[['이름','키']][:5]
```

	이름	키
지원번호		
1번	채치수	197
2번	정대만	184
3번	송태섭	168
4번	서태웅	187
5번	강백호	188

- **DataFrame(df)의 인덱스 3(4번째)~인덱스 마지막 데이터까지 출력한다.**

```
df[3:]
```

	이름	학교	키	국어	영어	수학	과학	사회	SW특기
지원번호									
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

07. 데이터 선택(loc)

- 새로운 파일 '07.데이터 선택(loc).ipynb'를 생성하고 DataFrame(df)에 index를 '지원번호'로 지정해 저장한다.

```
import pandas as pd
df = pd.read_excel('score.xlsx', index_col='지원번호')
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- DataFrame(df)의 인덱스가 '1번'에 해당하는 모든 칼럼 데이터를 출력한다.

```
df.loc['1번']
```

```
이름        채치수
학교        북산고
키         197
국어         90
영어         85
수학        100
과학         95
사회         85
SW특기    Python
Name: 1번, dtype: object
```

- DataFrame(df)의 인덱스 '5번'에 해당하는 모든 칼럼 데이터를 출력한다.

```
df.loc['5번']
```

```
이름        강백호
학교        북산고
키         188
국어         15
영어         20
수학         10
과학         35
사회         10
SW특기     NaN
Name: 5번, dtype: object
```

- **DataFrame(df)의 인덱스가 '1번'에 해당하는 '국어' 칼럼 데이터를 출력한다.**

```
df.loc['1번', '국어']
```

```
90
```

- **DataFrame(df)의 인덱스가 '1번', '2번'에 해당하는 '영어' 칼럼 데이터를 출력한다.**

```
df.loc[['1번', '2번'], '영어']
```

```
지원번호
 1번    85
 2번    35
Name: 영어, dtype: int64
```

```
type(df.loc[['1번', '2번'], '영어'])
```

```
pandas.core.series.Series
```

- **DataFrame(df)의 인덱스가 '1번', '2번'에 해당하는 '영어', '수학' 칼럼 데이터를 출력한다.**

```
df.loc[['1번', '2번'], ['영어','수학']]
```

지원번호	영어	수학
1번	85	100
2번	35	50

```
type(df.loc[['1번', '2번'], ['영어','수학']])
```

```
pandas.core.frame.DataFrame
```

- **DataFrame(df)의 인덱스가 '1번'~'5번'까지 칼럼이 '국어'~'사회'까지 데이터를 출력한다.**

```
df.loc['1번':'5번', '국어':'사회']
```

지원번호	국어	영어	수학	과학	사회
1번	90	85	100	95	85
2번	40	35	50	55	25
3번	80	75	70	80	75
4번	40	60	70	75	80
5번	15	20	10	35	10

08. 데이터 선택(iloc)

- 새로운 파일 '08.데이터 선택(iloc).ipynb'를 생성하고 파일을 읽어 DataFrame(df)에 저장한다.

```
import pandas as pd
df = pd.read_excel('score.xlsx', index_col='지원번호')
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- DataFrame(df)에서 인덱스가 0번째 위치의 모든 칼럼 데이터를 출력한다.

```
df.iloc[0]
```

```
이름          채치수
학교          북산고
키            197
국어           90
영어           85
수학          100
과학           95
사회           85
SW특기       Python
 Name: 1번, dtype: object
```

- DataFrame(df)의 인덱스가 0~4번째(5직전) 위치의 모든 칼럼 데이터를 출력한다.

```
df.iloc[0:5]
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN

- **DataFrame(df)의 index 0번째 위치의 label 1번째 데이터를 출력한다.**

```
df.iloc[0, 1]
```

```
'북산고'
```

- **DataFrame(df)의 인덱스가 4번째(5번) 학생의 2번째(키) 칼럼 데이터를 출력한다.**

```
df.iloc[4, 2]
```

```
188
```

- **DataFramd(df)의 인덱스가 0, 1번째 학생의 2번째(키) 칼럼 데이터를 출력한다.**

```
df.iloc[[0,1], 2]
```

```
지원번호
 1번    197
 2번    184
 Name: 키, dtype: int64
```

```
type(df.iloc[[0,1], 2])
```

```
pandas.core.series.Series
```

- **DataFrame(df)의 인덱스가 0, 1번째 학생에 대한 3, 4번째 칼럼 데이터를 출력한다.**

```
df.iloc[[0,1], [3,4]]
```

	국어	영어
지원번호		
1번	90	85
2번	40	35

- **DataFramd(df)의 인덱스가 0~4번째 학생에 대한 3~7번째 칼럼 데이터를 출력한다.**

```
df.iloc[0:5, 3:8]
```

	국어	영어	수학	과학	사회
지원번호					
1번	90	85	100	95	85
2번	40	35	50	55	25
3번	80	75	70	80	75
4번	40	60	70	75	80
5번	15	20	10	35	10

09. 데이터 선택(조건)

- **새로운 파일 '09.데이터 선택(조건).ipynb'를 생성하고 'score.xlsx'파일을 읽어 DataFrame에 저장한다.**

```
import pandas as pd
df = pd.read_excel('score.xlsx', index_col='지원번호')
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)의 학생들의 '키' 칼럼이 185 이상인지 여부를 True/False로 출력한다.**

```
df['키'] >= 185
```

```
지원번호
1번    True
2번    False
3번    False
4번    True
5번    True
6번    True
7번    True
8번    True
Name: 키, dtype: bool
```

- **DataFrame(df)의 학생들의 '키' 칼럼이 185 이상인 학생들의 정보를 2차원 배열로 출력한다.**

```
filt = (df['키'] >= 185)
df[filt]
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **filter를 역으로 적용하여 키가 185 미만의 데이터를 출력한다.**

```
df[~filt]
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript

- **조건식을 바로 배열에 입력하여 적용할 수 있다.**

```
df[df['키'] >= 185]
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)의 키가 185이상인 학생들의 수학 점수 데이터를 출력한다.**

```
df.loc[df['키'] >= 185, '수학']
```

```
지원번호
1번    100
4번     70
5번     10
6번     95
7번     45
8번     90
Name: 수학, dtype: int64
```

- **DataFrame(df)의 키가 185이상인 학생들의 이름, 수학, 과학 데이터를 출력한다.**

```
df.loc[df['키'] >= 185, ['이름','수학','과학']]
```

지원번호	이름	수학	과학
1번	채치수	100	95
4번	서태웅	70	75
5번	강백호	10	35
6번	변덕규	95	85
7번	황태산	45	40
8번	윤대협	90	95

- **DataFrame(df)의 키가 185 이상인 '북산고' 학생 데이터를 출력한다.(& 그리고)**

```
df.loc[(df['키'] >=185) & (df['학교'] == '북산고')]
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN

- **DataFrame(df)의 키가 170보다 작거나, 200보다 큰 학생 데이터를 출력한다. (또는 |)**

```
df.loc[(df['키'] < 170) | (df['키'] > 200)]
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
6번	변덕규	능남고	202	80	100	95	85	80	C

- **DataFrame(df)의 '송'씨 성을 가진 데이터 출력한다.**

```
filt = df['이름'].str.startswith('송')
df[filt]
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
3번	송태섭	북산고	168	80	75	70	80	75	Javascript

- **DataFrame(df)의 이름에 '태'가 들어가는 데이터 출력한다.**

```
filt = df['이름'].str.contains('태')
df[filt]
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
7번	황태산	능남고	188	55	65	45	40	35	PYTHON

- **DataFrame(df)에서 이름에 '태'가 들어가는 사람을 제외하고 출력한다.**

```
df[~filt]
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)에서 'SW특기'가 'Python'이거나 'Java'인 데이터를 출력한다.**

```
langs = ['Python', 'Java']
filt = df['SW특기'].isin(langs)
df[filt]
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java

- **DataFrame(df)에서 'SW특기'를 모두 소문자로 바꾼 후 소문자끼리 비교하여 데이터를 출력한다.**

```
langs = ['python', 'java']
filt = df['SW특기'].str.lower().isin(langs)
df[filt]
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
7번	황태산	능남고	188	55	65	45	40	35	PYTHON

- **DataFrame(df)에서 'SW특기'에 'Java'가 들어가는 학생들의 True, False 값 출력한다.**

```
df['SW특기'].str.contains('Java')
```

```
지원번호
1번     False
2번     True
3번     True
4번     NaN
5번     NaN
6번     False
7번     False
8번     False
Name: SW특기, dtype: object
```

- **DataFrame(df)에서 'NaN'를 False로 설정한다.**

```
df['SW특기'].str.contains('Java', na=False)
```

```
지원번호
1번    False
2번    True
3번    True
4번    False
5번    False
6번    False
7번    False
8번    False
Name: SW특기, dtype: bool
```

- **DataFrame(df)에서 'SW특기'에 'Java'가 들어가는 학생들 검색하여 출력한다.**

```
filt = df['SW특기'].str.contains('Java', na=False)
df[filt]
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript

10. 결측치(비어있는 데이터)

- **새로운 파일 '10.결측치.ipynb'를 생성하여 'score.xlsx' 파일을 읽어 DataFrame(df)에 저장한다.**

```
import pandas as pd
df = pd.read_excel('score.xlsx', index_col='지원번호')
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)에 'NaN' 데이터를 빈 칸으로 채운다.**

```
df.fillna('')
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	
5번	강백호	북산고	188	15	20	10	35	10	
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)에서 'NaN' 데이터를 '없음'으로 채운다.**

```
df.fillna('없음')
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	없음
5번	강백호	북산고	188	15	20	10	35	10	없음
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)에서 '학교' 데이터 전체를 'NaN'으로 채운다.**

```
import numpy as np
df['학교'] = np.nan
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	NaN	197	90	85	100	95	85	Python
2번	정대만	NaN	184	40	35	50	55	25	Java
3번	송태섭	NaN	168	80	75	70	80	75	Javascript
4번	서태웅	NaN	187	40	60	70	75	80	NaN
5번	강백호	NaN	188	15	20	10	35	10	NaN
6번	변덕규	NaN	202	80	100	95	85	80	C
7번	황태산	NaN	188	55	65	45	40	35	PYTHON
8번	윤대협	NaN	190	100	85	90	95	95	C#

- **DataFrame(df)에서 '학교' 데이터 전체를 모두 모름으로 채운다.**

```
df.fillna('모름', inplace=True)
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	모름	197	90	85	100	95	85	Python
2번	정대만	모름	184	40	35	50	55	25	Java
3번	송태섭	모름	168	80	75	70	80	75	Javascript
4번	서태웅	모름	187	40	60	70	75	80	모름
5번	강백호	모름	188	15	20	10	35	10	모름
6번	변덕규	모름	202	80	100	95	85	80	C
7번	황태산	모름	188	55	65	45	40	35	PYTHON
8번	윤대협	모름	190	100	85	90	95	95	C#

- **새로운 데이터를 다시 불러와 DataFrame(df)에 저장한다.**

```
df = pd.read_excel('score.xlsx', index_col='지원번호')
df
```

- **DataFrame(df)에서 'SW특기' 데이터가 'NaN'에 대해서 '확인중'으로 채운다.**

```
df['SW특기'].fillna('확인 중', inplace=True)
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	확인 중
5번	강백호	북산고	188	15	20	10	35	10	확인 중
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **새로운 데이터를 다시 불러와 DataFrame(df)에 저장한다.**

```
df = pd.read_excel('score.xlsx', index_col='지원번호')
df
```

- **DataFrame(df)에서 전체 데이터 중에서 'NaN'을 포함하는 데이터를 삭제한다.**

```
df.dropna(inplace=True)
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **새로운 데이터를 다시 불러와 DataFrame(df)에 저장한다.**

```
df = pd.read_excel('score.xlsx', index_col='지원번호')
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **axis 속성: (index or columns) / how 속성 (any or all)**
- **DataFrame(df)에서 'NoN'이 하나라도 있는 인덱스 행을 삭제한다.**

```
df.dropna(axis='index', how='any')
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)에서 'NaN'이 하나라도 있는 칼럼을 삭제한다.**

```
df.dropna(axis='columns')
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회
1번	채치수	북산고	197	90	85	100	95	85
2번	정대만	북산고	184	40	35	50	55	25
3번	송태섭	북산고	168	80	75	70	80	75
4번	서태웅	북산고	187	40	60	70	75	80
5번	강백호	북산고	188	15	20	10	35	10
6번	변덕규	능남고	202	80	100	95	85	80
7번	황태산	능남고	188	55	65	45	40	35
8번	윤대협	능남고	190	100	85	90	95	95

- **DataFrame(df)에서 '학교' 데이터를 모두 'NaN'으로 채운다.**

```
df['학교'] = np.nan
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	NaN	197	90	85	100	95	85	Python
2번	정대만	NaN	184	40	35	50	55	25	Java
3번	송태섭	NaN	168	80	75	70	80	75	Javascript
4번	서태웅	NaN	187	40	60	70	75	80	NaN
5번	강백호	NaN	188	15	20	10	35	10	NaN
6번	변덕규	NaN	202	80	100	95	85	80	C
7번	황태산	NaN	188	55	65	45	40	35	PYTHON
8번	윤대협	NaN	190	100	85	90	95	95	C#

- **DataFrame(df)에서 데이터 전체가 'NaN'인 경우에만 Column 삭제한다.**

```
df.dropna(axis='columns', how='all')
```

지원번호	이름	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	197	90	85	100	95	85	Python
2번	정대만	184	40	35	50	55	25	Java
3번	송태섭	168	80	75	70	80	75	Javascript
4번	서태웅	187	40	60	70	75	80	NaN
5번	강백호	188	15	20	10	35	10	NaN
6번	변덕규	202	80	100	95	85	80	C
7번	황태산	188	55	65	45	40	35	PYTHON
8번	윤대협	190	100	85	90	95	95	C#

11. 데이터 정렬

- **새로운 파일 '11.데이터 정렬.ipynb'를 생성하고 'score.xlsx' 파일을 생성하고 DataFrame에 저장한다.**

```
import pandas as pd
df = pd.read_excel('score.xlsx', index_col='지원번호')
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)에서 '키' 기준으로 '오름차순' 정렬을 한다.**

```
df.sort_values('키')
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
2번	정대만	북산고	184	40	35	50	55	25	Java
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#
1번	채치수	북산고	197	90	85	100	95	85	Python
6번	변덕규	능남고	202	80	100	95	85	80	C

- **DataFrame(df)에서 '키'가 큰 순으로 '내림차순' 정렬을 한다.**

```
df.sort_values('키', ascending=False)
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
6번	변덕규	능남고	202	80	100	95	85	80	C
1번	채치수	북산고	197	90	85	100	95	85	Python
8번	윤대협	능남고	190	100	85	90	95	95	C#
5번	강백호	북산고	188	15	20	10	35	10	NaN
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
4번	서태웅	북산고	187	40	60	70	75	80	NaN
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript

- **DataFrame(df)에서 '수학', '영어' 점수 기준으로 '오름차순' 정렬을 한다.**

```
df.sort_values(['수학', '영어'])
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
5번	강백호	북산고	188	15	20	10	35	10	NaN
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
2번	정대만	북산고	184	40	35	50	55	25	Java
4번	서태웅	북산고	187	40	60	70	75	80	NaN
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
8번	윤대협	능남고	190	100	85	90	95	95	C#
6번	변덕규	능남고	202	80	100	95	85	80	C
1번	채치수	북산고	197	90	85	100	95	85	Python

- **DataFrame(df)에서 '수학', '영어' 점수 기준으로 '내림차순' 정렬을 한다.**

```
df.sort_values(['수학', '영어'], ascending=False)
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
6번	변덕규	능남고	202	80	100	95	85	80	C
8번	윤대협	능남고	190	100	85	90	95	95	C#
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
2번	정대만	북산고	184	40	35	50	55	25	Java
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
5번	강백호	북산고	188	15	20	10	35	10	NaN

- **DataFrame(df) '수학' 점수는 '오름차순'으로,' 영어' 점수는 '내림차순'으로 정렬한다.**

```
df.sort_values(['수학', '영어'], ascending=[True, False])
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
5번	강백호	북산고	188	15	20	10	35	10	NaN
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
8번	윤대협	능남고	190	100	85	90	95	95	C#
6번	변덕규	능남고	202	80	100	95	85	80	C
1번	채치수	북산고	197	90	85	100	95	85	Python

- **DataFrame(df)에서 '키'순으로 '오름차순' 정렬을 한다.**

```
df['키'].sort_values()
```

```
지원번호
3번    168
2번    184
4번    187
5번    188
7번    188
8번    190
1번    197
6번    202
Name: 키, dtype: int64
```

- **DataFrame(df)에서 '키'순으로 '내림차순' 정렬을 한다.**

```
df['키'].sort_values(ascending=False)
```

```
지원번호
6번    202
1번    197
8번    190
5번    188
7번    188
4번    187
2번    184
3번    168
Name: 키, dtype: int64
```

- **DataFrame(df)에 'index' 기준(지원번호)로 '오름차순' 정렬을 한다.**

```
df.sort_index()
```

	이름	학교	키	국어	영어	수학	과학	사회	SW특기
지원번호									
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)에 'index' 기준(지원번호)로 '내림차순' 정렬을 한다.**

```
df.sort_index(ascending=False)
```

	이름	학교	키	국어	영어	수학	과학	사회	SW특기
지원번호									
8번	윤대협	능남고	190	100	85	90	95	95	C#
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
6번	변덕규	능남고	202	80	100	95	85	80	C
5번	강백호	북산고	188	15	20	10	35	10	NaN
4번	서태웅	북산고	187	40	60	70	75	80	NaN
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
2번	정대만	북산고	184	40	35	50	55	25	Java
1번	채치수	북산고	197	90	85	100	95	85	Python

12. 데이터 수정

- 새로운 파일 '12.데이터 수정.ipynb'를 생성하고 'score.xlsx' 파일을 읽어 DataFrame에 저장한다.

```
import pandas as pd
df = pd.read_excel('score.xlsx', index_col='지원번호')
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- DataFrame(df)에서 학교 이름의 '북산고'를 '상북고'로, '능남고'는 '무슨고'로 수정한다.

```
df['학교'].replace({'북산고':'상북고', '능남고':'무슨고'})
```

```
지원번호
1번    상북고
2번    상북고
3번    상북고
4번    상북고
5번    상북고
6번    무슨고
7번    무슨고
8번    무슨고
Name: 학교, dtype: object
```

- 앞에서 수정된 데이터를 DataFrame(fd)에 실제로 반영한다.

```
df['학교'].replace({'북산고':'상북고'}, inplace=True)
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	상북고	197	90	85	100	95	85	Python
2번	정대만	상북고	184	40	35	50	55	25	Java
3번	송태섭	상북고	168	80	75	70	80	75	Javascript
4번	서태웅	상북고	187	40	60	70	75	80	NaN
5번	강백호	상북고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)에서 'SW특기'를 모두 소문자로 출력한다.**

```
df['SW특기'].str.lower()
```

```
지원번호
1번         python
2번           java
3번     javascript
4번            NaN
5번            NaN
6번              c
7번         python
8번             c#
Name: SW특기, dtype: object
```

- **DataFrame(df)에서 'SW특기'를 모두 소문자로 바꾸어 새로 DataFrame(df)에 저장한다.**

```
df['SW특기'] = df['SW특기'].str.lower()
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	상북고	197	90	85	100	95	85	python
2번	정대만	상북고	184	40	35	50	55	25	java
3번	송태섭	상북고	168	80	75	70	80	75	javascript
4번	서태웅	상북고	187	40	60	70	75	80	NaN
5번	강백호	상북고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	c
7번	황태산	능남고	188	55	65	45	40	35	python
8번	윤대협	능남고	190	100	85	90	95	95	c#

- **DataFrame(df)에서 'SW특기'를 모두 대문자로 바꾸어 새로 DataFrame(df)에 저장한다.**

```
df['SW특기'] = df['SW특기'].str.upper()
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	상북고	197	90	85	100	95	85	PYTHON
2번	정대만	상북고	184	40	35	50	55	25	JAVA
3번	송태섭	상북고	168	80	75	70	80	75	JAVASCRIPT
4번	서태웅	상북고	187	40	60	70	75	80	NaN
5번	강백호	상북고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)에서 '학교' 칼럼에 '등학교' 단어를 추가해 저장한다.**

```
df['학교'] = df['학교'] + '등학교'
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	상북고등학교	197	90	85	100	95	85	PYTHON
2번	정대만	상북고등학교	184	40	35	50	55	25	JAVA
3번	송태섭	상북고등학교	168	80	75	70	80	75	JAVASCRIPT
4번	서태웅	상북고등학교	187	40	60	70	75	80	NaN
5번	강백호	상북고등학교	188	15	20	10	35	10	NaN
6번	변덕규	능남고등학교	202	80	100	95	85	80	C
7번	황태산	능남고등학교	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고등학교	190	100	85	90	95	95	C#

- **DataFrame(df)에서 모든 과목의 합계를 구하여 새로운 '총합' 칼럼을 새로 추가한다.**

```
df['총합'] = df['국어'] + df['영어'] + df['수학'] + df['과학'] + df['사회']
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기	총합
1번	채치수	상북고등학교	197	90	85	100	95	85	PYTHON	455
2번	정대만	상북고등학교	184	40	35	50	55	25	JAVA	205
3번	송태섭	상북고등학교	168	80	75	70	80	75	JAVASCRIPT	380
4번	서태웅	상북고등학교	187	40	60	70	75	80	NaN	325
5번	강백호	상북고등학교	188	15	20	10	35	10	NaN	90
6번	변덕규	능남고등학교	202	80	100	95	85	80	C	440
7번	황태산	능남고등학교	188	55	65	45	40	35	PYTHON	240
8번	윤대협	능남고등학교	190	100	85	90	95	95	C#	465

- **DataFrame(df)에 '결과' 칼럼을 추가하고 전체 데이터는 'Fail'로 초기화 한다.**

```
df['결과'] = 'Fail'
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기	총합	결과
1번	채치수	상북고등학교	197	90	85	100	95	85	PYTHON	455	Fail
2번	정대만	상북고등학교	184	40	35	50	55	25	JAVA	205	Fail
3번	송태섭	상북고등학교	168	80	75	70	80	75	JAVASCRIPT	380	Fail
4번	서태웅	상북고등학교	187	40	60	70	75	80	NaN	325	Fail
5번	강백호	상북고등학교	188	15	20	10	35	10	NaN	90	Fail
6번	변덕규	능남고등학교	202	80	100	95	85	80	C	440	Fail
7번	황태산	능남고등학교	188	55	65	45	40	35	PYTHON	240	Fail
8번	윤대협	능남고등학교	190	100	85	90	95	95	C#	465	Fail

- **DataFrame(df) '총합'이 400보다 큰 결과를 'Pass'로 수정해 준다.**

```
df.loc[df['총합'] > 400, '결과'] = 'Pass'
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기	총합	결과
1번	채치수	상북고등학교	197	90	85	100	95	85	PYTHON	455	Pass
2번	정대만	상북고등학교	184	40	35	50	55	25	JAVA	205	Fail
3번	송태섭	상북고등학교	168	80	75	70	80	75	JAVASCRIPT	380	Fail
4번	서태웅	상북고등학교	187	40	60	70	75	80	NaN	325	Fail
5번	강백호	상북고등학교	188	15	20	10	35	10	NaN	90	Fail
6번	변덕규	능남고등학교	202	80	100	95	85	80	C	440	Pass
7번	황태산	능남고등학교	188	55	65	45	40	35	PYTHON	240	Fail
8번	윤대협	능남고등학교	190	100	85	90	95	95	C#	465	Pass

- **DataFramd(df)에서 '총합' 칼럼을 삭제 한다.**

```
df.drop(columns=['총합'])
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기	결과
1번	채치수	상북고등학교	197	90	85	100	95	85	PYTHON	Pass
2번	정대만	상북고등학교	184	40	35	50	55	25	JAVA	Fail
3번	송태섭	상북고등학교	168	80	75	70	80	75	JAVASCRIPT	Fail
4번	서태웅	상북고등학교	187	40	60	70	75	80	NaN	Fail
5번	강백호	상북고등학교	188	15	20	10	35	10	NaN	Fail
6번	변덕규	능남고등학교	202	80	100	95	85	80	C	Pass
7번	황태산	능남고등학교	188	55	65	45	40	35	PYTHON	Fail
8번	윤대협	능남고등학교	190	100	85	90	95	95	C#	Pass

- **DataFrame(df)에서 '국어', '영어', '수학' 세 개의 칼럼을 삭제 한다.**

```
df.drop(columns=['국어', '영어', '수학'])
```

지원번호	이름	학교	키	과학	사회	SW특기	총합	결과
1번	채치수	상북고등학교	197	95	85	PYTHON	455	Pass
2번	정대만	상북고등학교	184	55	25	JAVA	205	Fail
3번	송태섭	상북고등학교	168	80	75	JAVASCRIPT	380	Fail
4번	서태웅	상북고등학교	187	75	80	NaN	325	Fail
5번	강백호	상북고등학교	188	35	10	NaN	90	Fail
6번	변덕규	능남고등학교	202	85	80	C	440	Pass
7번	황태산	능남고등학교	188	40	35	PYTHON	240	Fail
8번	윤대협	능남고등학교	190	95	95	C#	465	Pass

- **DataFrame(df)에서 인덱스 '4번' 학생 데이터를 삭제 한다.**

```
df.drop(index='4번')
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기	총합	결과
1번	채치수	상북고등학교	197	90	85	100	95	85	PYTHON	455	Pass
2번	정대만	상북고등학교	184	40	35	50	55	25	JAVA	205	Fail
3번	송태섭	상북고등학교	168	80	75	70	80	75	JAVASCRIPT	380	Fail
5번	강백호	상북고등학교	188	15	20	10	35	10	NaN	90	Fail
6번	변덕규	능남고등학교	202	80	100	95	85	80	C	440	Pass
7번	황태산	능남고등학교	188	55	65	45	40	35	PYTHON	240	Fail
8번	윤대협	능남고등학교	190	100	85	90	95	95	C#	465	Pass

- **DataFrame(df)에 '수학' 점수 80점미만 데이터만 추출한다.**

```
filt = df['수학'] < 80
df[filt]
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기	총합	결과
2번	정대만	상북고등학교	184	40	35	50	55	25	JAVA	205	Fail
3번	송태섭	상북고등학교	168	80	75	70	80	75	JAVASCRIPT	380	Fail
4번	서태웅	상북고등학교	187	40	60	70	75	80	NaN	325	Fail
5번	강백호	상북고등학교	188	15	20	10	35	10	NaN	90	Fail
7번	황태산	능남고등학교	188	55	65	45	40	35	PYTHON	240	Fail

- **위 조건을 만족하는 인덱스 값을 출력한다.**

```
df[filt].index
```

```
Index(['2번', '3번', '4번', '5번', '7번'], dtype='object', name='지원번호')
```

- **위 조건을 만족하는 데이터를 삭제한다.**

```
df.drop(index=df[filt].index)
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기	총합	결과
1번	채치수	상북고등학교	197	90	85	100	95	85	PYTHON	455	Pass
6번	변덕규	능남고등학교	202	80	100	95	85	80	C	440	Pass
8번	윤대협	능남고등학교	190	100	85	90	95	95	C#	465	Pass

- **DataFrame(df)에 새로운 학생 데이터를 추가 한다.**

```
df.loc['9번'] = ['이정환', '해남고등하고', 184, 90, 90, 90, 90, 90, 'Kotlin', 450, 'Pass']
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기	총합	결과
1번	채치수	상북고등학교	197	90	85	100	95	85	PYTHON	455	Pass
2번	정대만	상북고등학교	184	40	35	50	55	25	JAVA	205	Fail
7번	황태산	능남고등학교	188	55	65	45	40	35	PYTHON	240	Fail
8번	윤대협	능남고등학교	190	100	85	90	95	95	C#	465	Pass
9번	이정환	해남고등하고	184	90	90	90	90	90	Kotlin	450	Pass

- **DataFrame(ds)의 '4번' 학생의 'SW특기' 데이터를 'Python'으로 변경 한다.**

```
df.loc['4번', 'SW특기'] = 'Python'
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기	총합	결과
1번	채치수	상북고등학교	197	90	85	100	95	85	PYTHON	455	Pass
2번	정대만	상북고등학교	184	40	35	50	55	25	JAVA	205	Fail
3번	송태섭	상북고등학교	168	80	75	70	80	75	JAVASCRIPT	380	Fail
4번	서태웅	상북고등학교	187	40	60	70	75	80	Python	325	Fail

- **DataFrame(df)의 '5번' 학생의 학교는 '능남고등학교'로, 'SW특기'는 'C'로 변경 한다.**

```
df.loc['5번', ['학교', 'SW특기']] = ['능남고등학교', 'C']
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기	총합	결과
1번	채치수	상북고등학교	197	90	85	100	95	85	PYTHON	455	Pass
2번	정대만	상북고등학교	184	40	35	50	55	25	JAVA	205	Fail
3번	송태섭	상북고등학교	168	80	75	70	80	75	JAVASCRIPT	380	Fail
4번	서태웅	상북고등학교	187	40	60	70	75	80	Python	325	Fail
5번	강백호	능남고등학교	188	15	20	10	35	10	C	90	Fail
6번	변덕규	능남고등학교	202	80	100	95	85	80	C	440	Pass
7번	황태산	능남고등학교	188	55	65	45	40	35	PYTHON	240	Fail
8번	윤대협	능남고등학교	190	100	85	90	95	95	C#	465	Pass
9번	이정환	해남고등하고	184	90	90	90	90	90	Kotlin	450	Pass

- **DataFrame(df)의 칼럼 이름들을 목록으로 출력한다.**

```
cols = list(df.columns)
cols
```

```
['이름', '학교', '키', '국어', '영어', '수학', '과학', '사회', 'SW특기', '총합', '결과']
```

- **맨 뒤에 있는 결과 Column을 앞으로 가져오고, 나머지 Column들과 합쳐서 순서를 변경 한다.**

```
df = df[[cols[-1]] + cols[0:-1]]
df
```

	결과	이름	학교	키	국어	영어	수학	과학	사회	SW특기	총합
지원번호											
1번	Pass	채치수	상북고등학교	197	90	85	100	95	85	PYTHON	455
2번	Fail	정대만	상북고등학교	184	40	35	50	55	25	JAVA	205

- **DataFrame(df)에서 '결과', '이름', '학교' 세 개의 칼럼으로 새로운 DataFrame(df)을 만든다.**

```
df = df[['결과', '이름', '학교']]
df
```

	결과	이름	학교
지원번호			
1번	Pass	채치수	상북고등학교
2번	Fail	정대만	상북고등학교
3번	Fail	송태섭	상북고등학교

- **DataFrame(df)의 칼럼 이름 목록을 출력해 본다.**

```
df.columns
```

```
Index(['결과', '이름', '학교'], dtype='object')
```

- **DataFrame(df)의 칼럼의 이름을 'Result', 'Name', 'School'로 변경해 본다.**

```
df.columns = ['Result', 'Name', 'School']
df
```

	Result	Name	School
지원번호			
1번	Pass	채치수	상북고등학교
2번	Fail	정대만	상북고등학교
3번	Fail	송태섭	상북고등학교

13. 함수 적용

- 새로운 파일 '13.함수 적용.ipynb'를 생성하고 'score.xlsx' 파일을 읽어 DataFrame(df)에 저장한다.

```
import pandas as pd
df = pd.read_excel('score.xlsx', index_col='지원번호')
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- DataFrame(df)의 '키' 칼럼에 'cm'를 연결하는 함수를 작성해서 '키' 칼럼에 적용한다.

```
def add_cm(height):
    return str(height) + 'cm'
df['키'] = df['키'].apply(add_cm)
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197cm	90	85	100	95	85	Python
2번	정대만	북산고	184cm	40	35	50	55	25	Java
3번	송태섭	북산고	168cm	80	75	70	80	75	Javascript

- SW특기에 첫 글자를 대문자로 변환하여 저장한다.

```
def capitalize(lang):
    if pd.notnull(lang):  #NaN이 아닌 경우
        return lang.capitalize()  #첫 글자는 대문자로, 나머지는 소문자로 변환하는 내장함수
    return lang
df['SW특기'] = df['SW특기'].apply(capitalize)
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197cm	90	85	100	95	85	Python
2번	정대만	북산고	184cm	40	35	50	55	25	Java
6번	변덕규	능남고	202cm	80	100	95	85	80	C
7번	황태산	능남고	188cm	55	65	45	40	35	Python
8번	윤대협	능남고	190cm	100	85	90	95	95	C#

14. 그룹화

- **새로운 파일 '14.그룹화.ipynb'를 생성하고 'score.xlsx' 파일을 읽어 DataFrame(df)에 저장한다.**

```
import pandas as pd
df = pd.read_excel('score.xlsx', index_col='지원번호')
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)를 같을 '학교'별로 그룹한 후 '능남고' 학생들만 출력한다.**

```
df.groupby('학교').get_group('능남고')
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
6번	변덕규	능남고	202	80	100	95	85	80	C
7번	황태산	능남고	188	55	65	45	40	35	PYTHON
8번	윤대협	능남고	190	100	85	90	95	95	C#

- **DataFrame(df)를 같을 '학교'별로 그룹한 후 '북산고' 학생들만 출력한다.**

```
df.groupby('학교').get_group('능남고')
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
1번	채치수	북산고	197	90	85	100	95	85	Python
2번	정대만	북산고	184	40	35	50	55	25	Java
3번	송태섭	북산고	168	80	75	70	80	75	Javascript
4번	서태웅	북산고	187	40	60	70	75	80	NaN
5번	강백호	북산고	188	15	20	10	35	10	NaN

- **DataFrame(df)의 각 '학교'별 그룹의 학생 수를 출력한다.**

```
df.groupby('학교').size()
```

```
학교
능남고    3
북산고    5
dtype: int64
```

- **DataFrame(df)에 '학교'로 그룹화를 한 뒤에 '능남고'에 해당하는 학생의 수를 출력한다.**

```
df.groupby('학교').size()['능남고']
```

```
3
```

- **DataFrame(df)에 '학교'로 그룹화를 한 뒤에 '키'의 평균 데이터 값을 출력한다.**

```
df.groupby('학교')['키'].mean()
```

```
학교
능남고    193.333333
북산고    184.800000
Name: 키, dtype: float64
```

- **DataFrame(df)의 '학교'로 그룹화를 한 뒤에 '국어', '영어', '수학' 평균 데이터를 출력한다.**

```
df.groupby('학교')[['국어', '영어', '수학']].mean()
```

학교	국어	영어	수학
능남고	78.333333	83.333333	76.666667
북산고	53.000000	55.000000	60.000000

- **DataFrame(df)에 새로운 '학년' 칼럼을 추가하고 새로운 데이터를 입력한다.**

```
df['학년'] = [3, 3, 3, 1, 1, 3, 2, 2]
df
```

지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기	학년
1번	채치수	북산고	197	90	85	100	95	85	Python	3
2번	정대만	북산고	184	40	35	50	55	25	Java	3
3번	송태섭	북산고	168	80	75	70	80	75	Javascript	3
4번	서태웅	북산고	187	40	60	70	75	80	NaN	1
5번	강백호	북산고	188	15	20	10	35	10	NaN	1

- **DataFrame(df)의 '학교'별, '학년'별 '국어','영어','수학','과학','사회'의 평균 데이터를 출력한다.**

```
df.groupby(['학교', '학년'])[['국어', '영어', '수학', '과학', '사회']].mean()
```

		국어	영어	수학	과학	사회
학교	학년					
능남고	2	77.5	75.0	67.500000	67.500000	65.000000
	3	80.0	100.0	95.000000	85.000000	80.000000
북산고	1	27.5	40.0	40.000000	55.000000	45.000000
	3	70.0	65.0	73.333333	76.666667	61.666667

- **DataFrme(df)에서 '학년'별 '키'의 평균 데이터를 출력한다.**

```
df.groupby('학년').mean()
```

	키
학년	
1	187.50
2	189.00
3	187.75

- **DataFrame(df)에서 '학년'별 그룹한 후 '키'가 큰 순으로 '내림차순' 정렬한 결과를 출력한다.**

```
df.groupby('학년')[['국어', '영어', '수학', '과학', '사회','키']].mean().sort_values('키', ascending=False)
```

	국어	영어	수학	과학	사회	키
학년						
2	77.5	75.00	67.50	67.50	65.00	189.00
3	72.5	73.75	78.75	78.75	66.25	187.75
1	27.5	40.00	40.00	55.00	45.00	187.50

- **DataFrame(df)의 '학교', '학년'별 그룹한 후 합계를 출력한다.**

```
df.groupby(['학교', '학년']).sum()
```

		이름	키	국어	영어	수학	과학	사회	SW특기
학교	학년								
능남고	2	황태산윤대협	378	155	150	135	135	130	PYTHONC#
	3	변덕규	202	80	100	95	85	80	C
북산고	1	서태웅강백호	375	55	80	80	110	90	0
	3	채치수정대만송태섭	549	210	195	220	230	185	PythonJavaJavascript

- **DataFrame(df)에 '학교'로 그룹화를 한 뒤에 각 '학교'별 'SW특기' 데이터의 수를 출력한다.**

```
df.groupby('학교')[['이름','SW특기']].count()
```

	이름	SW특기
학교		
능남고	3	3
북산고	5	3

- **DataFrame(df)에 '학교'로 그룹화를 한 뒤에 '학년'별 학생 수를 출력한다.**

```
school = df.groupby('학교')
school['학년'].value_counts()
```

```
학교   학년
능남고  2    2
     3    1
북산고  3    3
     1    2
Name: 학년, dtype: int64
```

- **DataFrame(df)에서 '학교'로 그룹화를 한 뒤에 '북산고'에 대해서 '학년'별 학생 수를 출력한다.**

```
school['학년'].value_counts().loc['북산고']
```

```
학년
3    3
1    2
Name: 학년, dtype: int64
```

- **DataFrame(df)에서 '학교'로 그룹화를 한 뒤에 '능남고'에 대해서 '학년'별 학생 수를 출력한다.**

```
school['학년'].value_counts().loc['능남고']
```

```
학년
2    2
3    1
Name: 학년, dtype: int64
```

- **DataFrame(df)에서 '학교'로 그룹화를 한 뒤에 '북산고'에 대해서 '학년'별 학생 비율(%)을 출력한다.**

```
school['학년'].value_counts(normalize=True).loc['북산고']
```

```
학년
3    0.6
1    0.4
Name: 학년, dtype: float64
```

15. Pandas 퀴즈

다음은 대한민국 영화중에서 관객 수가 가장 많은 상위 8개의 데이터이다. 주어진 코드를 이용하여 퀴즈를 풀어보시오.

- **새로운 파일 '15.Pandas 퀴즈.ipynb'를 생성하여 아래 문제에 대한 답을 작성하시오.**

```
import pandas as pd
data = {
    '영화' : ['명량', '극한직업', '신과함께-죄와 벌', '국제시장', '괴물', '도둑들', '7번방의 선물', '암살'],
    '개봉 연도' : [2014, 2019, 2017, 2014, 2006, 2012, 2013, 2015],
    '관객 수' : [1761, 1626, 1441, 1426, 1301, 1298, 1281, 1270], # (단위 : 만 명)
    '평점' : [8.88, 9.20, 8.73, 9.16, 8.62, 7.64, 8.83, 9.10]
}
df = pd.DataFrame(data)
df
```

	영화	개봉 연도	관객 수	평점
0	명량	2014	1761	8.88
1	극한직업	2019	1626	9.20
2	신과함께-죄와 벌	2017	1441	8.73
3	국제시장	2014	1426	9.16
4	괴물	2006	1301	8.62
5	도둑들	2012	1298	7.64
6	7번방의 선물	2013	1281	8.83
7	암살	2015	1270	9.10

- **DataFrame(df) 데이터 중에서 '영화' 칼럼 정보만 출력하시오.**

```
df['영화']
```

```
0          명량
1        극한직업
2    신과함께-죄와 벌
3        국제시장
4          괴물
5         도둑들
6      7번방의 선물
7          암살
Name: 영화, dtype: object
```

- **DataFrame(df) 데이터 중에서 '영화', '평점' 칼럼 정보를 출력하시오.**

```
df[['영화', '평점']]
```

	영화	평점
0	명량	8.88
1	극한직업	9.20
2	신과함께-죄와 벌	8.73
3	국제시장	9.16
4	괴물	8.62
5	도둑들	7.64
6	7번방의 선물	8.83
7	암살	9.10

- **'2015'년 이후에 개봉한 데이터 중에서 '영화', '개봉연도' 정보를 출력하시오.**

```
df.loc[df['개봉 연도'] >= 2015, ['영화', '개봉 연도']]
```

	영화	개봉 연도
1	극한직업	2019
2	신과함께-죄와 벌	2017
7	암살	2015

- **주어진 계산식을 참고하여 '추천점수' 칼럼을 추가하시오. (추천점수 = (관객 수 * 평점) // 1000)**

```
df['추첨 점수'] = (df['관객 수'] * df['평점']) // 100
df
```

	영화	개봉 연도	관객 수	평점	추첨 점수
0	명량	2014	1761	8.88	156.0
1	극한직업	2019	1626	9.20	149.0
2	신과함께-죄와 벌	2017	1441	8.73	125.0
3	국제시장	2014	1426	9.16	130.0
4	괴물	2006	1301	8.62	112.0
5	도둑들	2012	1298	7.64	99.0
6	7번방의 선물	2013	1281	8.83	113.0
7	암살	2015	1270	9.10	115.0

- **DataFrme(df)의 전체 데이터를 '개봉연도' 기준 '내림차순'으로 출력하시오.**

```
df.sort_values('개봉 연도', ascending=False)
```

	영화	개봉 연도	관객 수	평점	추첨 점수
1	극한직업	2019	1626	9.20	149.0
2	신과함께-죄와 벌	2017	1441	8.73	125.0
7	암살	2015	1270	9.10	115.0
0	명량	2014	1761	8.88	156.0
3	국제시장	2014	1426	9.16	130.0
6	7번방의 선물	2013	1281	8.83	113.0
5	도둑들	2012	1298	7.64	99.0
4	괴물	2006	1301	8.62	112.0

• Matplotlib

맷플롯립(Matplotlib)은 다양한 형태의 그래프를 통해서 데이터 시각화를 할 수 있는 라이브러리이다.

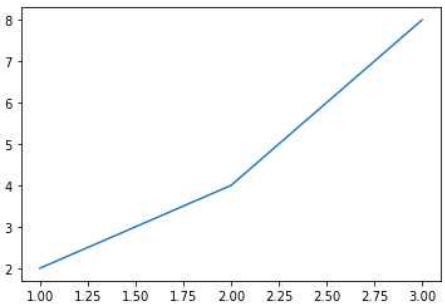
01. 그래프 기본

- **새로운 파일 '01.그래프 기본.ipynb'를 생성한다.**

```
import matplotlib.pyplot as plt
```

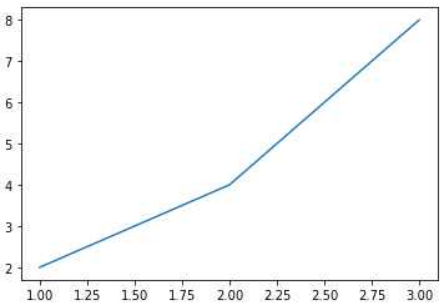
```
x = [1, 2, 3]
y = [2, 4, 8]
plt.plot(x, y)
```

```
[<matplotlib.lines.Line2D at 0x203f4238640>]
```

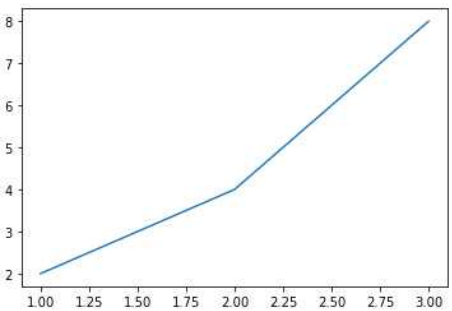



```
print(plt.plot(x, y))
```

```
[<matplotlib.lines.Line2D object at 0x00000203F49BF4F0>]
```

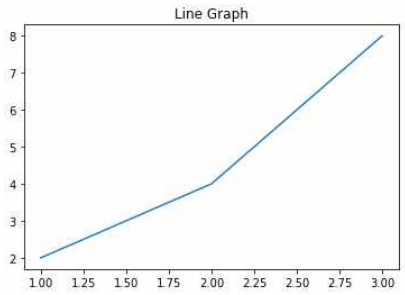



```
plt.plot(x, y)
plt.show()
```



- **직선 그래프에 제목(title)을 설정 한다.**

```
plt.plot(x, y)
plt.title('Line Graph')
```



- **그래프에서 한글 폰트를 사용하기 위한 설정을 한다.**

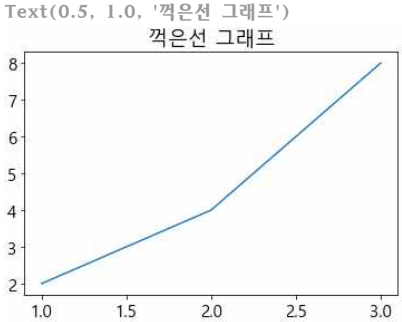
```
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'  #맑은 고딕
matplotlib.rcParams['font.size'] = 15  #글자 크기
matplotlib.rcParams['axes.unicode_minus'] = False  #한글 폰트 사용 시 마이너스 글자가 깨지는 현상을 해결
```

```
import matplotlib.font_manager as fm
fm.fontManager.ttflist  #사용 가능한 폰트 확인
[f.name for f in fm.fontManager.ttflist]
```

```
['DejaVu Sans Display',
 'cmmi10',
 'STIXSizeOneSym',
...]
```

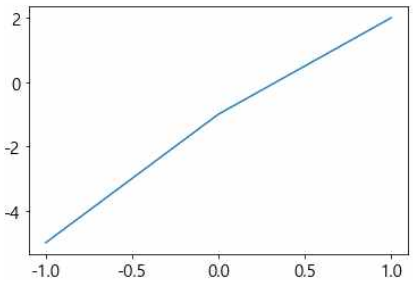
```
plt.plot(x, y)
plt.title('꺽은선 그래프')
```

```
Text(0.5, 1.0, '꺽은선 그래프')
```




```
plt.plot([-1, 0, 1], [-5, -1, 2])
```

```
[<matplotlib.lines.Line2D at 0x203f4bc06a0>]
```



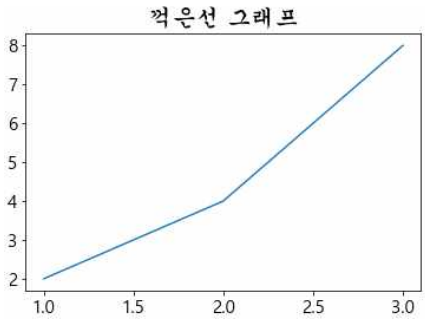
02. 축(Axis)

- 새로운 파일 '02.축.ipynb'를 생성한다.

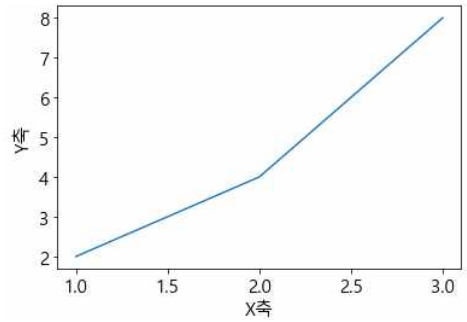
```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

```
x = [1, 2, 3]
y = [2, 4, 8]
```

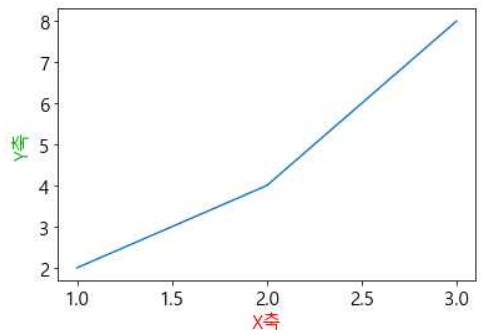
```
plt.plot(x, y)
plt.title('꺽은선 그래프', fontdict={'family':'HYGungSo-Bold', 'size':20})  #개별 폰트 설정
```



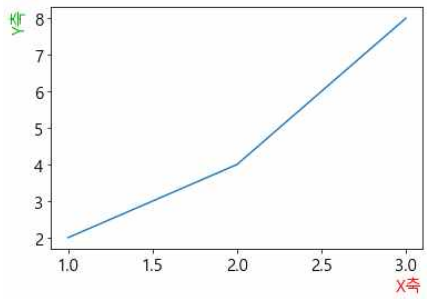

```
plt.plot(x, y)
plt.xlabel('X축')
plt.ylabel('Y축')
```



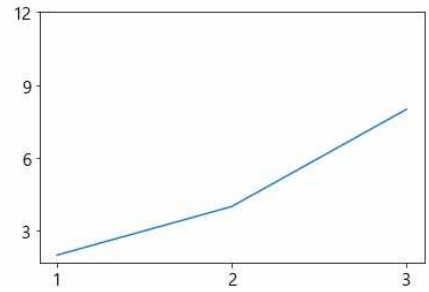

```
plt.plot(x, y)
plt.xlabel('X축', color='red')
plt.ylabel('Y축', color='#00AA00')
```



```
plt.plot(x, y)
plt.xlabel('X축', color='red', loc='right')  #left, center, right
plt.ylabel('Y축', color='#00AA00', loc='top')  #top, center, bottom
```




```
plt.plot(x, y)
plt.xticks([1, 2, 3])
plt.yticks([3, 6, 9, 12])
plt.show()
```



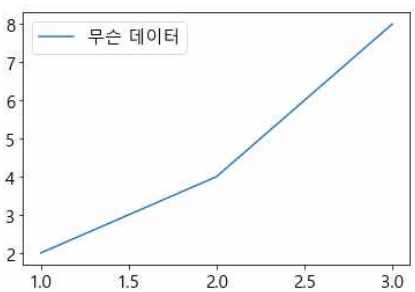
03. 범례(Legend)

- 새로운 파일 '03.범례.ipynb'를 생성한다.

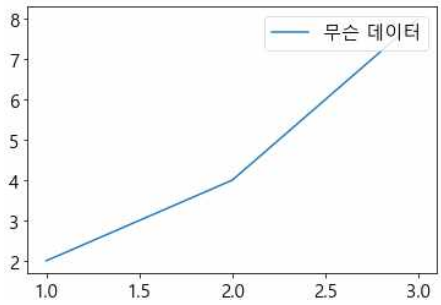
```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

```
x = [1, 2, 3]
y = [2, 4, 8]
```

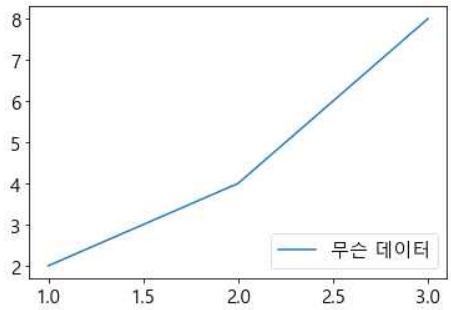
```
plt.plot(x, y, label='무슨 데이터')
plt.legend()
```



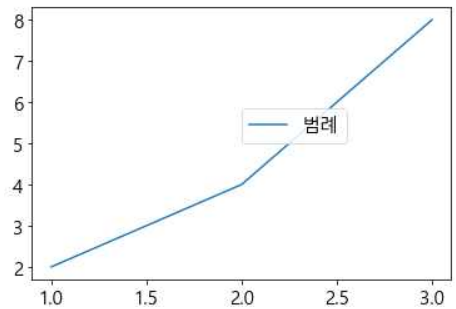
```
plt.plot(x, y, label='무슨 데이터')
plt.legend(loc='upper right')
```




```
plt.plot(x, y, label='무슨 데이터')
plt.legend(loc='lower right')
```




```
plt.plot(x, y, label='범례')
plt.legend(loc=(0.5, 0.5))  #x축, y축 (0~1 사이)
```



04. 스타일(Style)

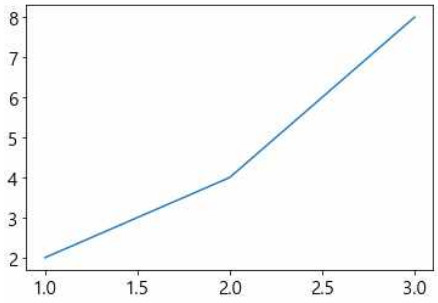
- **새로운 파일 '04.스타일.ipnb'를 생성한다.**

```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

```
x = [1, 2, 3]
y = [2, 4, 8]
```

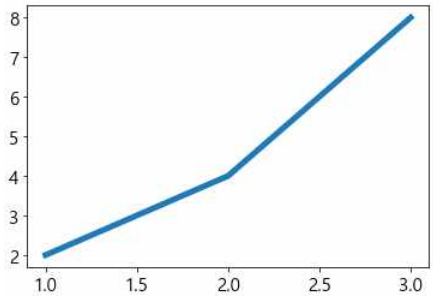
```
plt.plot(x, y)
```

```
[<matplotlib.lines.Line2D at 0x1aa29f20340>]
```




```
plt.plot(x, y, linewidth=5)
```

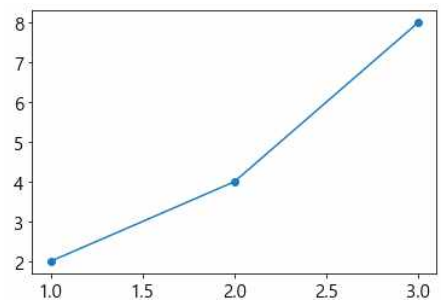
```
[<matplotlib.lines.Line2D at 0x1aa2a923460>
```



- **마커(Marker)**

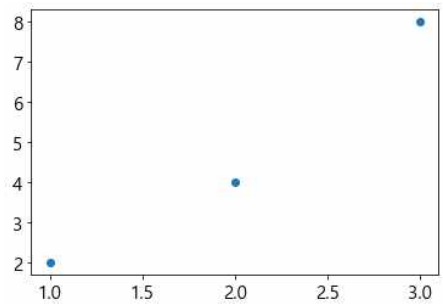
```
plt.plot(x, y, marker='o')
```

```
[<matplotlib.lines.Line2D at 0x1aa2a977c70>]
```



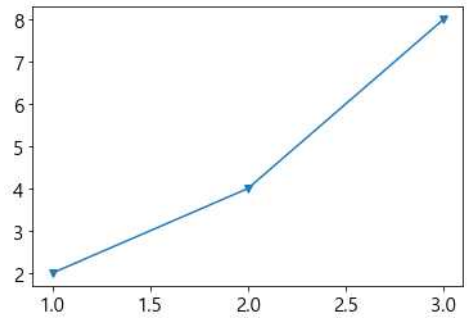

```
plt.plot(x, y, marker='o', linestyle='None')
```

```
[<matplotlib.lines.Line2D at 0x1aa2a9d6580>]
```



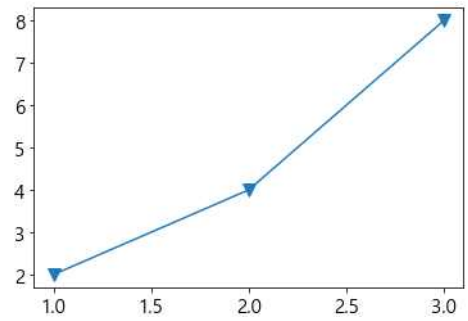
```
plt.plot(x, y, marker='v')
```

```
[<matplotlib.lines.Line2D at 0x1aa2aa26c40>]
```



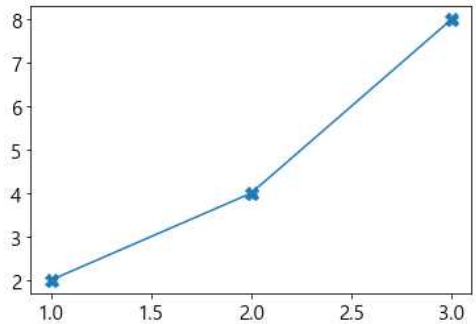

```
plt.plot(x, y, marker='v', markersize=10)
```

```
[<matplotlib.lines.Line2D at 0x1aa2aa84670>]
```



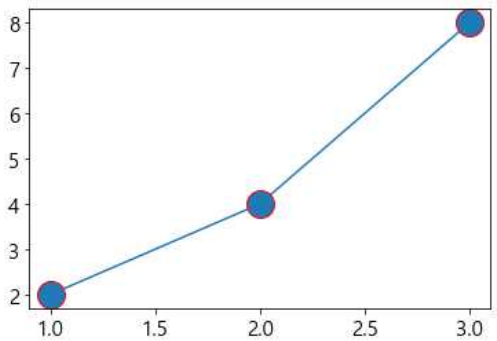

```
plt.plot(x, y, marker='X', markersize=10)
```

```
[<matplotlib.lines.Line2D at 0x1aa2aadac10>]
```



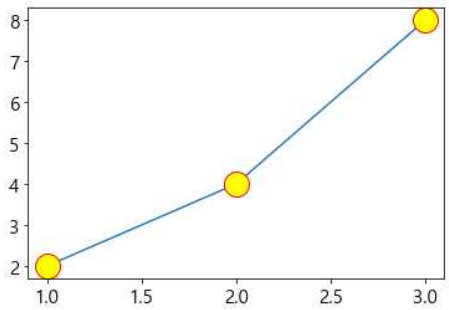

```
plt.plot(x, y, marker='o', markersize=20, markeredgecolor='red')
```

```
[<matplotlib.lines.Line2D at 0x1aa2abe8370>]
```



```
plt.plot(x, y, marker='o', markersize=20, markeredgecolor='red', markerfacecolor='yellow')
```

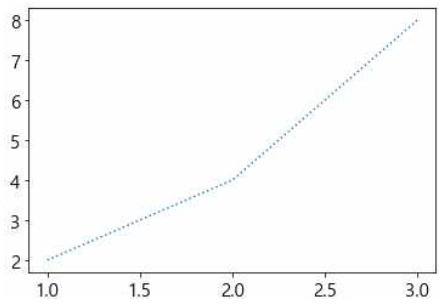
```
[<matplotlib.lines.Line2D at 0x1aa2a9aac40>]
```



- **선 스타일(Line Style)**

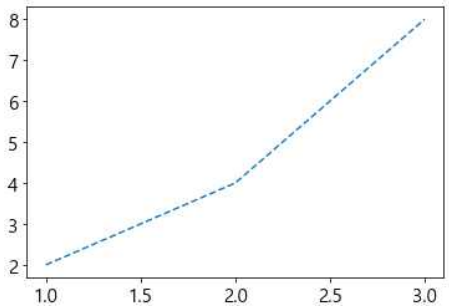
```
plt.plot(x, y, linestyle=':')
```

```
[<matplotlib.lines.Line2D at 0x1aa2ab625b0>]
```



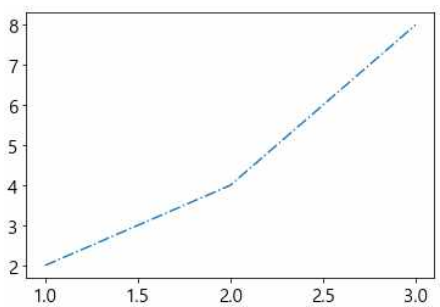

```
plt.plot(x, y, linestyle='--')
```

```
[<matplotlib.lines.Line2D at 0x1aa2acd17c0>]
```




```
plt.plot(x, y, linestyle='-.')
```

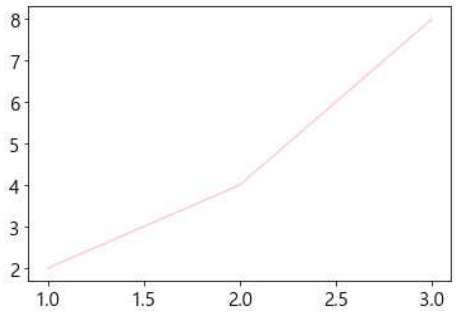
```
[<matplotlib.lines.Line2D at 0x1aa2ad22ee0>]
```



- **색깔(Color)**

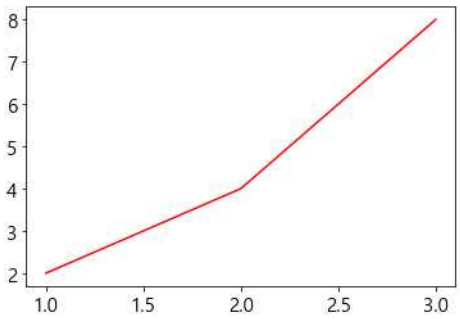
```
plt.plot(x, y, color='pink')
```

```
[<matplotlib.lines.Line2D at 0x1aa2ad81670>]
```



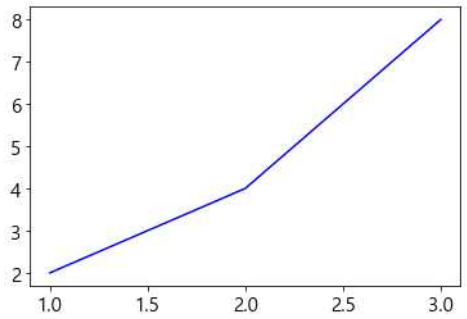

```
plt.plot(x, y, color='#FF0000')
```

```
[<matplotlib.lines.Line2D at 0x1aa2add5d00>]
```



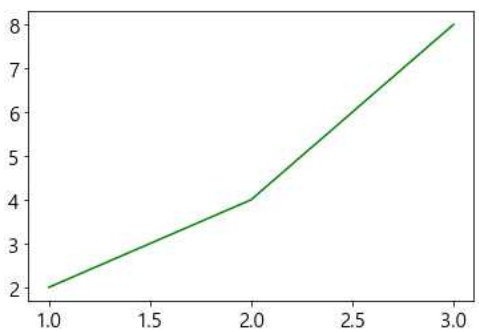

```
plt.plot(x, y, color='b')
```

```
[<matplotlib.lines.Line2D at 0x1aa2ae35490>]
```




```
plt.plot(x, y, color='g')
```

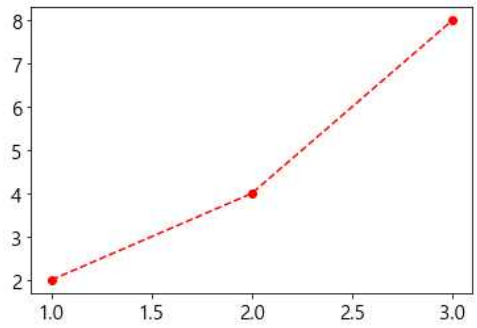
```
[<matplotlib.lines.Line2D at 0x1aa2ae85b80>]
```



- **포맷(Format)**

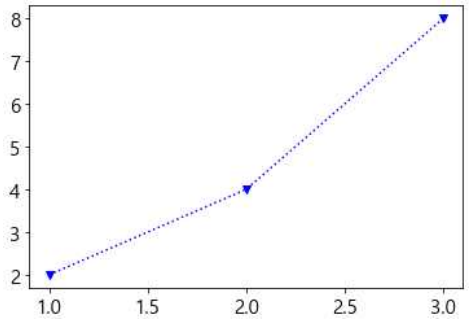
```
plt.plot(x, y, 'ro--')  #color, marker, linestyle
```

```
[<matplotlib.lines.Line2D at 0x1aa2beb5490>]
```



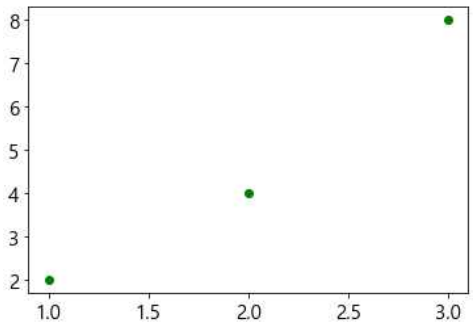

```
plt.plot(x, y, 'bv:')  #파랑, 역삼각형, 점선
```

```
[<matplotlib.lines.Line2D at 0x1aa2bf08a00>]
```



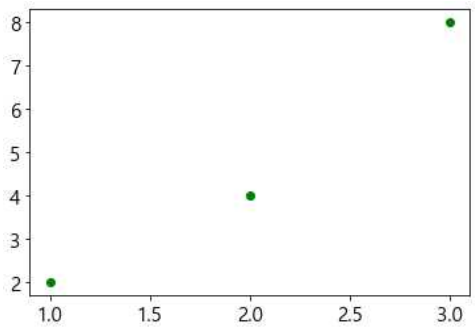

```
plt.plot(x, y, 'go')  #초록색 원 선 없음
```

```
[<matplotlib.lines.Line2D at 0x1aa2bf67370>]
```



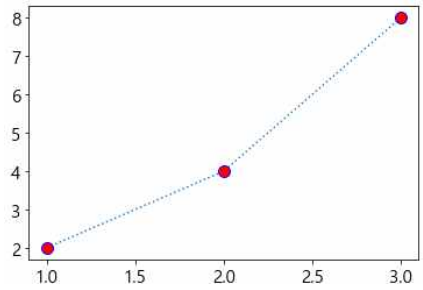

```
plt.plot(x, y, color='g', marker='o', linestyle='None')
```

```
[<matplotlib.lines.Line2D at 0x1aa2aca0370>]
```



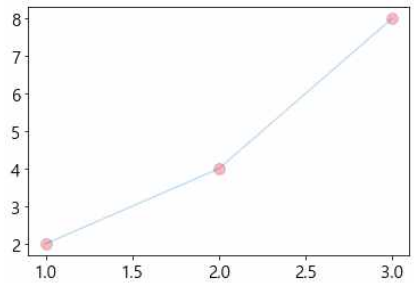
- **축약어**

```
plt.plot(x, y, marker='o', mfc='red', ms=10, mec='blue', ls=':')
```



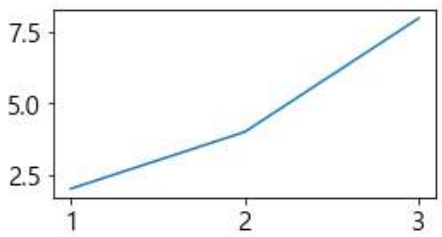
- **투명도**

```
plt.plot(x, y, marker='o', mfc='red', ms=10, alpha=0.3)  #alpha 투명도(0~1)
```

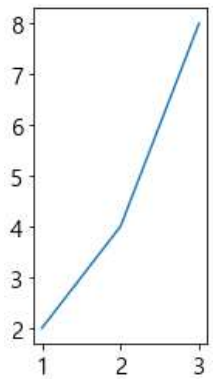


- **그래프 크기**

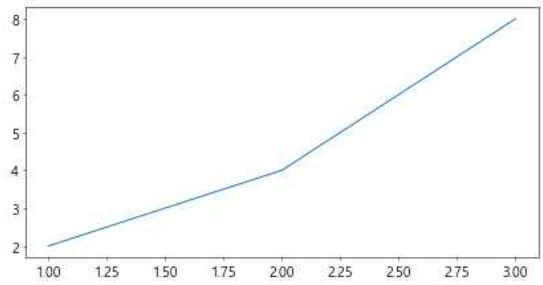
```
plt.figure(figsize=(10, 5))  #넓이, 높이
plt.plot(x, y)
```




```
plt.figure(figsize=(2, 4))
plt.plot(x, y)
```

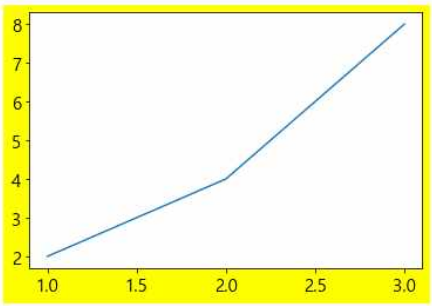


```
plt.figure(figsize=(10, 5), dpi=50)  #dot for inch, 확대
plt.plot(x, y)
```

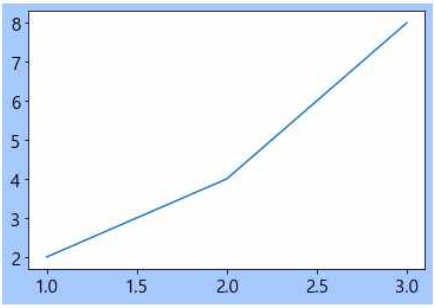


- **배경색(Background)**

```
plt.figure(facecolor='yellow')
plt.plot(x, y)
```




```
plt.figure(facecolor='#A1C3FF')
plt.plot(x, y)
```



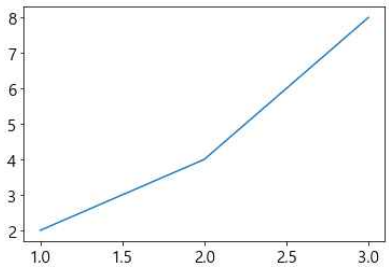
05. 파일저장

- **새로운 파일 '05.파일저장.ipynb'를 생성한다.**

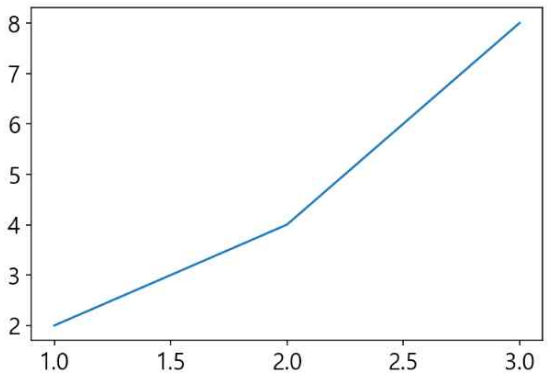
```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

```
x = [1, 2, 3]
y = [2, 4, 8]
```

```
plt.plot(x, y)  #화면에 보여 지는 사이즈 100dpi
plt.savefig('graph.png', dpi=200)  #파일에 저장되는 사이즈 200dpi
```




```
plt.figure(dpi=150)
plt.plot(x, y)
plt.savefig('graph_200.png', dpi=100)
```



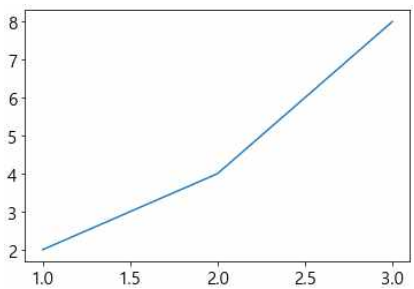
06. 텍스트(Text)

- 새로운 파일 '06.텍스트.ipynb'를 생성한다.

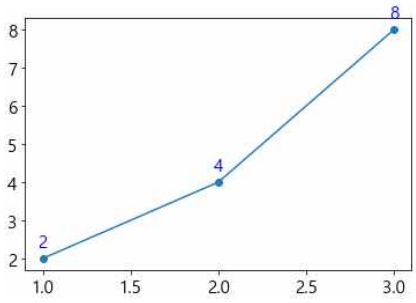
```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

```
x = [1, 2, 3]
y = [2, 4, 8]
```

```
plt.plot(x, y)
```



```
plt.plot(x, y, marker='o')
for idx, txt in enumerate(y):
    plt.text(x[idx], y[idx] + 0.3, txt, ha='center', color='blue')  #txt는 y값
```

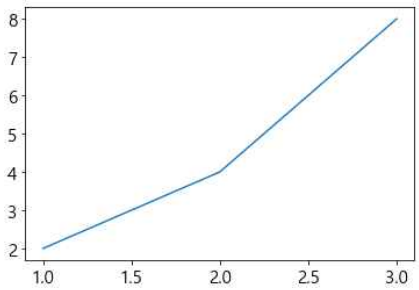


07. 여러 데이터

- 새로운 파일 '07.여러 데이터.ipynb'를 생성한다.

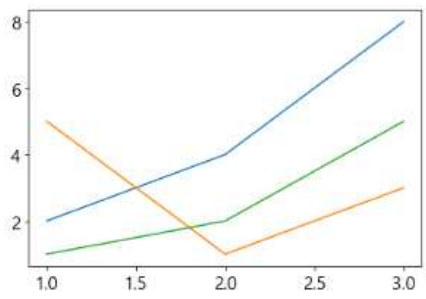
```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
x = [1, 2, 3]
y = [2, 4, 8]
```

```
plt.plot(x, y)
```

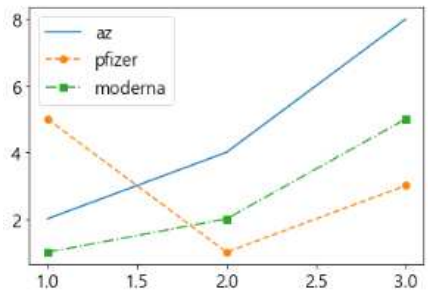


- COVID-19 백신 종류별 접종 인구

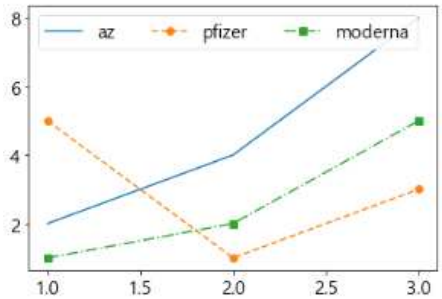
```
days = [1, 2, 3]  #1일, 2일, 3일
az = [2, 4, 8]  #[단위 : 만명] 1일부터 3일까지 아스트라제네카 접종인구
pfizer = [5, 1, 3]  #화이자
moderna = [1, 2, 5]  #모더나
plt.plot(days, az)
plt.plot(days, pfizer)
plt.plot(days, moderna)
```



```
plt.plot(days, az, label='az')
plt.plot(days, pfizer, label='pfizer', marker='o', linestyle='--')
plt.plot(days, moderna, label='moderna', marker='s', ls='-.')
plt.legend()
```




```
plt.plot(days, az, label='az')
plt.plot(days, pfizer, label='pfizer', marker='o', linestyle='--')
plt.plot(days, moderna, label='moderna', marker='s', ls='-.')
plt.legend(ncol=3)
```

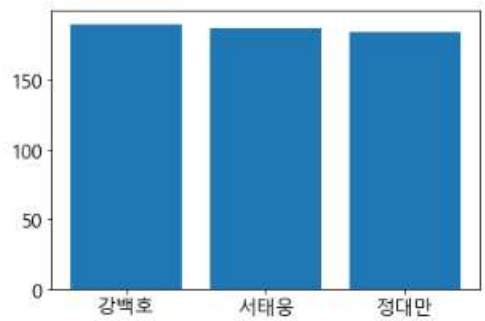


08. 막대 그래프(기본)

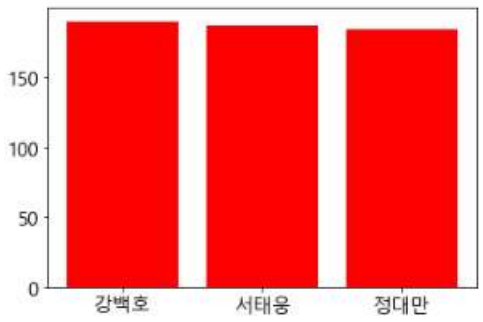
- 새로운 파일 '08.막대 그래프(기본).ipynb'를 생성한다.

```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

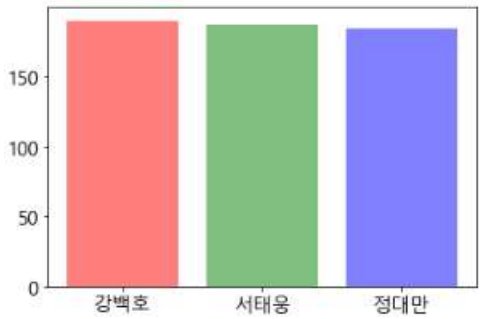
```
labels = ['강백호', '서태웅', '정대만']  #이름
values =[190, 187, 184]  #키
plt.bar(labels, values)
```



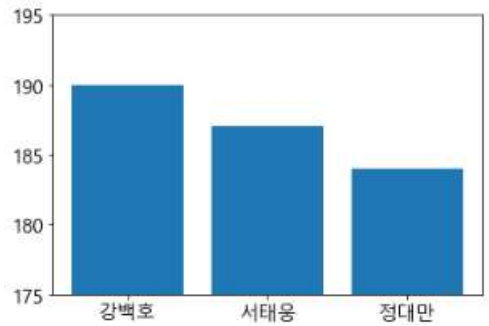
```
plt.bar(labels, values, color='r')
```



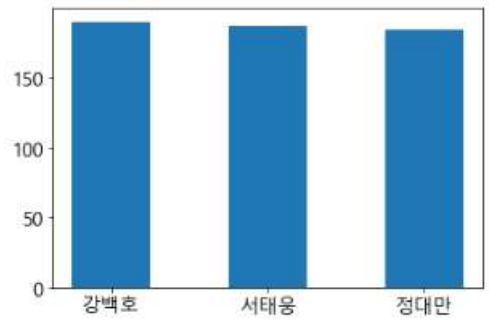

```
colors = ['r', 'g', 'b']
plt.bar(labels, values, color=colors, alpha=0.5)
```



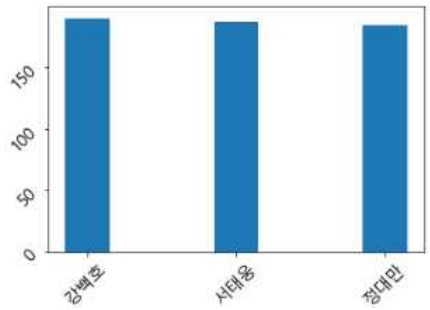

```
plt.bar(labels, values)
plt.ylim(175, 195)
```



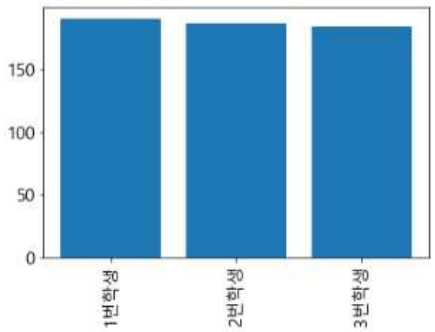

```
plt.bar(labels, values, width=0.5)
```



```
plt.bar(labels, values, width=0.3)
plt.xticks(rotation=45)  #x 축의 이름 데이터 각도를 45도로 설정
plt.yticks(rotation=45)  #y 축의 키 데이터 각도를 45도로 설정
```




```
labels = ['강백호', '서태웅', '정대만']
values =[190, 187, 184]
ticks =['1번학생', '2번학생', '3번학생']
plt.bar(labels, values)
plt.xticks(labels, ticks, rotation=90)
```

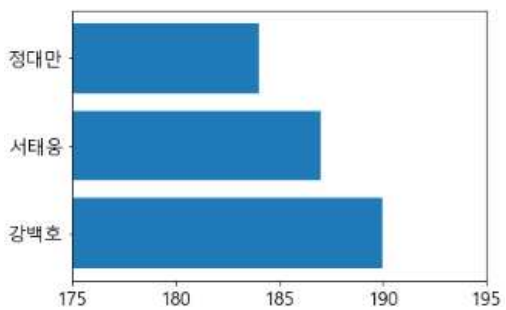


09. 막대 그래프(심화)

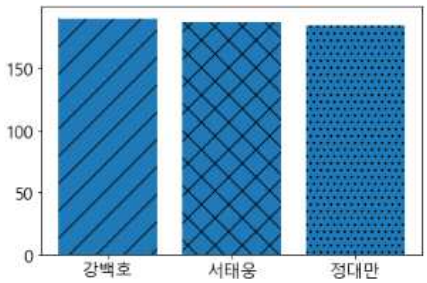
- 새로운 파일 '09.막대그래프(심화).ipynb'를 생성한다.

```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

```
labels = ['강백호', '서태웅', '정대만']
values =[190, 187, 184]
plt.barh(labels, values)
plt.xlim(175, 195)
```



```
bar = plt.bar(labels, values)
bar[0].set_hatch('/')
bar[1].set_hatch('x')
bar[2].set_hatch('..')
```




```
bar = plt.bar(labels, values)
plt.ylim(175, 195)
for idx, rect in enumerate(bar):
    plt.text(idx, rect.get_height() + 0.5, values[idx], ha='center', color='blue')
```

10. DataFrame 활용

- **새로운 파일 '10.DataFrame 활용.ipynb'를 생성한다.**

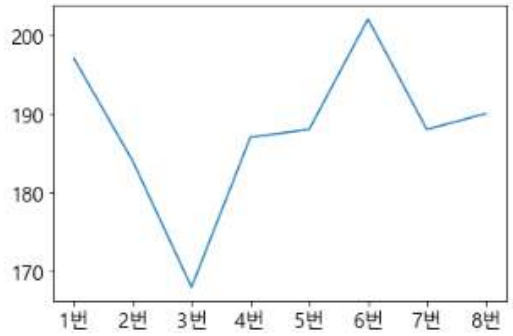
```
import pandas as pd
```

```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

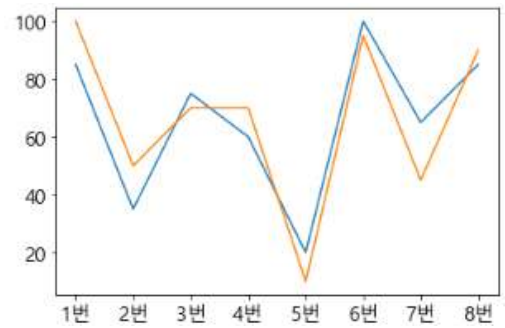
```
df = pd.read_excel('../Pandas/score.xlsx')
df
```

	지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
0	1번	채치수	북산고	197	90	85	100	95	85	Python
1	2번	정대만	북산고	184	40	35	50	55	25	Java
2	3번	송태섭	북산고	168	80	75	70	80	75	Javascript
3	4번	서태웅	북산고	187	40	60	70	75	80	NaN
4	5번	강백호	북산고	188	15	20	10	35	10	NaN
5	6번	변덕규	능남고	202	80	100	95	85	80	C
6	7번	황태산	능남고	188	55	65	45	40	35	PYTHON
7	8번	윤대협	능남고	190	100	85	90	95	95	C#

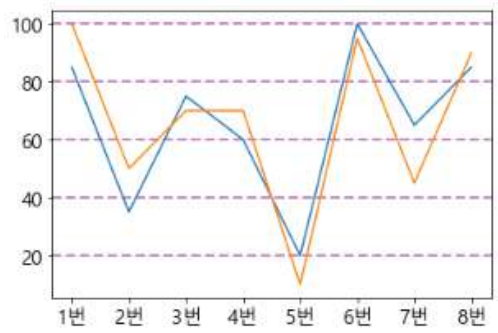
```
plt.plot(df['지원번호'], df['키'])
```




```
plt.plot(df['지원번호'], df['영어'])
plt.plot(df['지원번호'], df['수학'])
```




```
plt.plot(df['지원번호'], df['영어'])
plt.plot(df['지원번호'], df['수학'])
plt.grid(axis='y', color='purple', alpha=0.5, linestyle='--', linewidth=2)
```



11. 누적 막대그래프

- 새로운 파일 '11.누적 막대그래프.ipynb'를 생성한다.

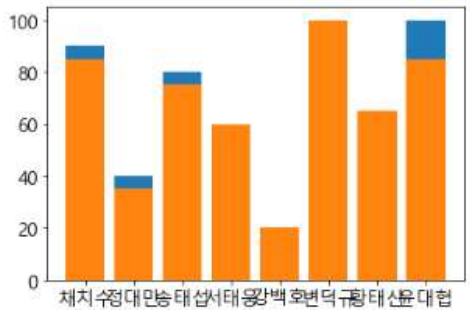
```
import pandas as pd
```

```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

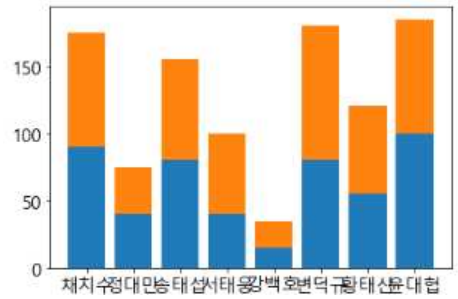
```
df = pd.read_excel('../Pandas/score.xlsx')
df
```

	지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
0	1번	채치수	북산고	197	90	85	100	95	85	Python
1	2번	정대만	북산고	184	40	35	50	55	25	Java
2	3번	송태섭	북산고	168	80	75	70	80	75	Javascript
3	4번	서태웅	북산고	187	40	60	70	75	80	NaN
4	5번	강백호	북산고	188	15	20	10	35	10	NaN
5	6번	변덕규	능남고	202	80	100	95	85	80	C
6	7번	황태산	능남고	188	55	65	45	40	35	PYTHON
7	8번	윤대협	능남고	190	100	85	90	95	95	C#

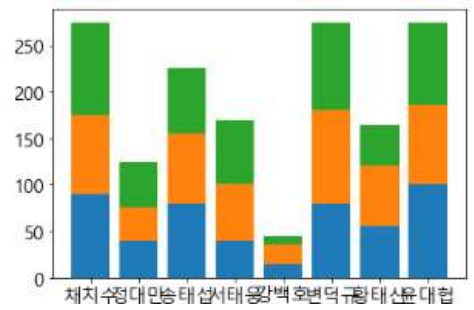
```
plt.bar(df['이름'], df['국어'])
plt.bar(df['이름'], df['영어'])
```



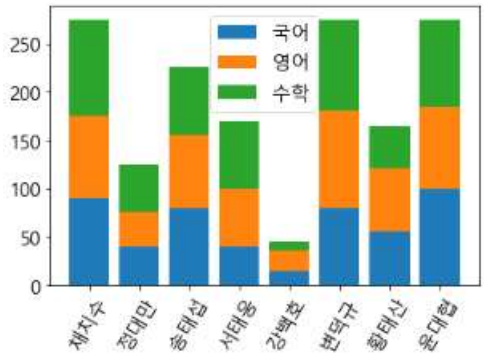

```
plt.bar(df['이름'], df['국어'])
plt.bar(df['이름'], df['영어'], bottom=df['국어'])  #아래에 국어 위에 영어
```




```
plt.bar(df['이름'], df['국어'])
plt.bar(df['이름'], df['영어'], bottom=df['국어'])
plt.bar(df['이름'], df['수학'], bottom=df['국어'] + df['영어'])
```



```
plt.bar(df['이름'], df['국어'], label='국어')
plt.bar(df['이름'], df['영어'], bottom=df['국어'], label='영어')
plt.bar(df['이름'], df['수학'], bottom=df['국어'] + df['영어'], label='수학')
plt.xticks(rotation=60)
plt.legend()
```



12. 다중 막대그래프

- 새로운 파일 '12.다중 막대그래프.ipynb'를 생성한다.

```
import pandas as pd
```

```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

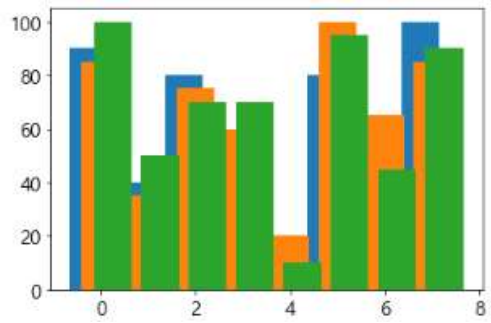
```
df = pd.read_excel('../Pandas/score.xlsx')
df
```

	지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
0	1번	채지수	북산고	197	90	85	100	95	85	Python
1	2번	정대만	북산고	184	40	35	50	55	25	Java
2	3번	송태섭	북산고	168	80	75	70	80	75	Javascript
3	4번	서태웅	북산고	187	40	60	70	75	80	NaN
4	5번	강백호	북산고	188	15	20	10	35	10	NaN
5	6번	변덕규	능남고	202	80	100	95	85	80	C
6	7번	황태산	능남고	188	55	65	45	40	35	PYTHON
7	8번	윤대협	능남고	190	100	85	90	95	95	C#

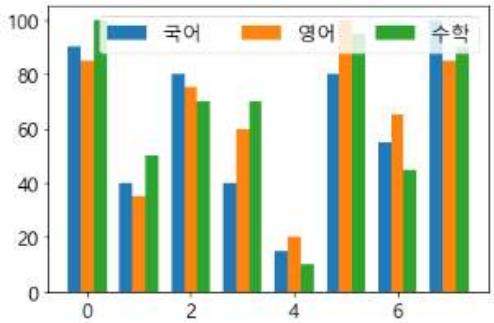
```
import numpy as np
index = np.arange(df.shape[0])
index
```

```
array([0, 1, 2, 3, 4, 5, 6, 7])
```

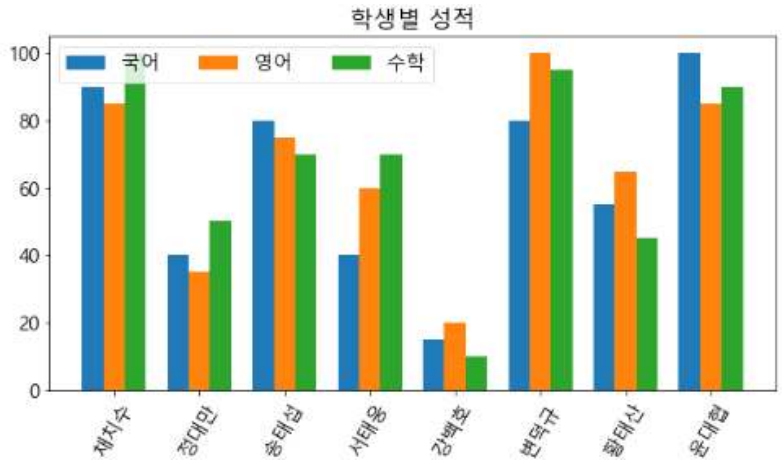
```
w = 0.25
plt.bar(index - w, df['국어'])
plt.bar(index, df['영어'])
plt.bar(index + w, df['수학'])
plt.show()
```




```
w = 0.25
plt.bar(index - w, df['국어'], width=w, label='국어')
plt.bar(index, df['영어'], width=w, label='영어')
plt.bar(index + w, df['수학'], width=w, label='수학')
plt.legend(ncol=3)
plt.show()
```




```
plt.figure(figsize=(10, 5))
plt.title('학생별 성적')
w = 0.25
plt.bar(index - w, df['국어'], width=w, label='국어')
plt.bar(index, df['영어'], width=w , label='영어')
plt.bar(index + w, df['수학'], width=w, label='수학')
plt.legend(ncol=3)
plt.xticks(index, df['이름'], rotation=60)
plt.show()
```

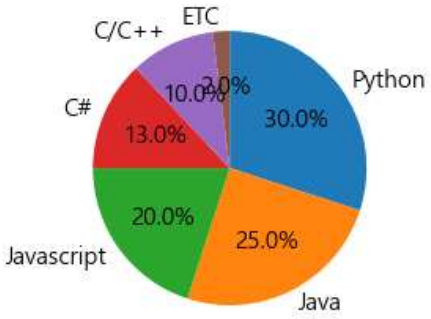


13. 원 그래프(기본)

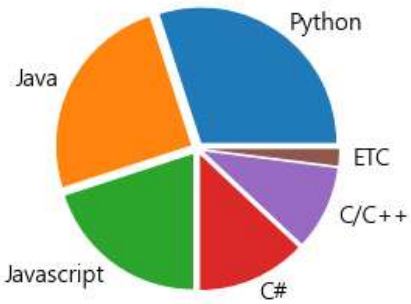
- 새로운 파일 '13.원 그래프(기본).ipynb'를 생성한다.

```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

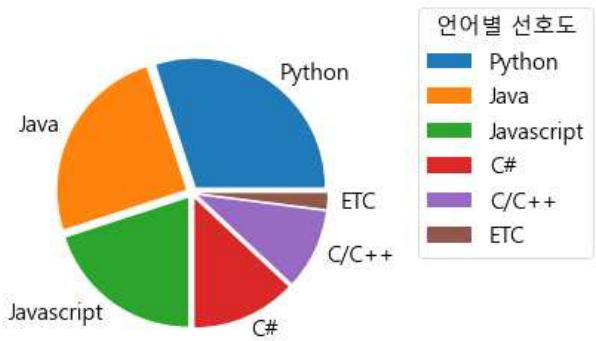
```
values = [30, 25, 20, 13, 10, 2]
labels =['Python', 'Java', 'Javascript', 'C#', 'C/C++', 'ETC']
plt.pie(values, labels = labels, autopct='%.1f%%', startangle=90, counterclock=False)
plt.show()
```




```
values = [30, 25, 20, 13, 10, 2]
labels =['Python', 'Java', 'Javascript', 'C#', 'C/C++', 'ETC']
explode = [0.05] * 6
plt.pie(values, labels=labels, explode=explode)
plt.show()
```




```
plt.pie(values, labels=labels, explode=explode)
plt.legend(loc=(1.2, 0.3), title='언어별 선호도')
plt.show()
```

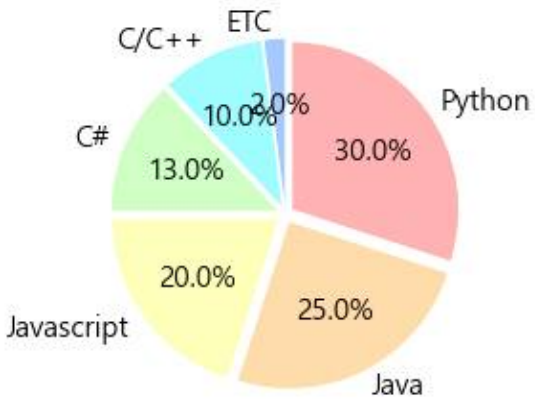


14. 원 그래프(심화)

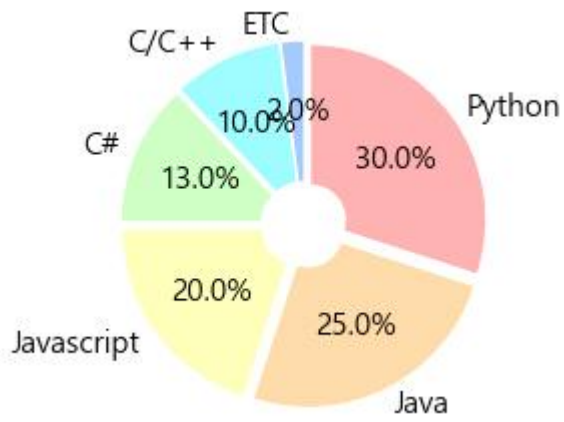
- 새로운 파일 '14.원 그래프(심화).ipynb'를 생성한다.

```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

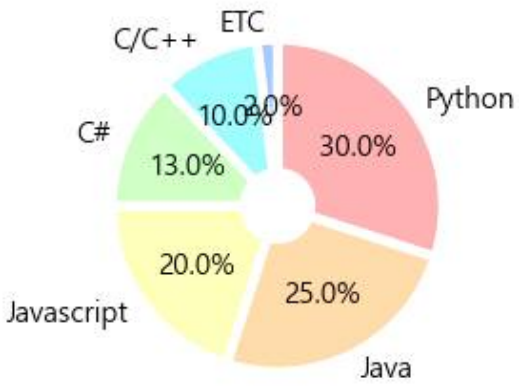
```
values = [30, 25, 20, 13, 10, 2]
labels =['Python', 'Java', 'Javascript', 'C#', 'C/C++', 'ETC']
colors=['#FFADAD','#FFD6A5','#FDFFB6','#CAFFBF','#9BF6FF','#A0C4FF']  #colors=['b','g','r','c','m','y']
explode = [0.05] * 6
plt.pie(values, labels = labels, autopct='%.1f%%', startangle=90, counterclock=False, colors=colors, explode=explode)
plt.show()
```



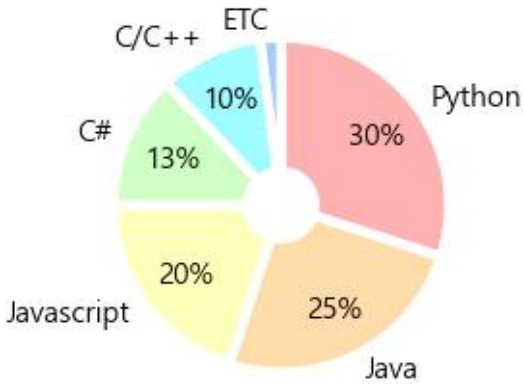

```
wedgeprops={'width':0.8}
plt.pie(values, labels = labels, autopct='%.1f%%', startangle=90, counterclock=False, colors=colors, explode=explode, wedgeprops=wedgeprops)
plt.show()
```



```
wedgeprops={'width':0.8, 'edgecolor':'w', 'linewidth':5}
plt.pie(values, labels=labels, autopct='%.1f%%', startangle=90, counterclock=False,
colors=colors, wedgeprops=wedgeprops)
plt.show()
```




```
def custom_autopct(pct):
    return '{:.0f}%'.format(pct) if pct >=10 else ''  #return ('%.1f%%' % pct) if pct >= 10 else ''
plt.pie(values, labels=labels, autopct=custom_autopct, startangle=90, counterclock=False,
colors=colors, wedgeprops=wedgeprops, pctdistance=0.7)
plt.show()
```



- **데이터 프레임 활용**

```
import pandas as pd
df = pd.read_excel('../Pandas/score.xlsx')
df
```

	지원번호	이름	학교	키	국어	영어	수학	과학	사회	sw특기
0	1번	채치수	북산고	197	90	85	100	95	85	Python
1	2번	정대만	북산고	184	40	35	50	55	25	Java
2	3번	송태섭	북산고	168	80	75	70	80	75	Javascript
3	4번	서태웅	북산고	187	40	60	70	75	80	NaN
4	5번	강백호	북산고	188	15	20	10	35	10	NaN
5	6번	변덕규	능남고	202	80	100	95	85	80	C
6	7번	황태산	능남고	188	55	65	45	40	35	PYTHON
7	8번	윤대협	능남고	190	100	85	90	95	95	C#

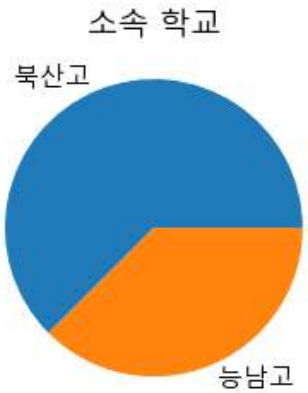
```
grp = df.groupby('학교')
grp
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x000001905BB77B50>
```

```
grp.size()['북산고']
```

```
5
```

```
values = [grp.size()['북산고'], grp.size()['능남고']]  #[5, 3]
labels = ['북산고', '능남고']
plt.pie(values, labels=labels)
plt.title('소속 학교')
plt.show()
```



15. 산점도 그래프

- **새로운 파일 '15.산점도 그래프.ipynb'를 생성한다.**

```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

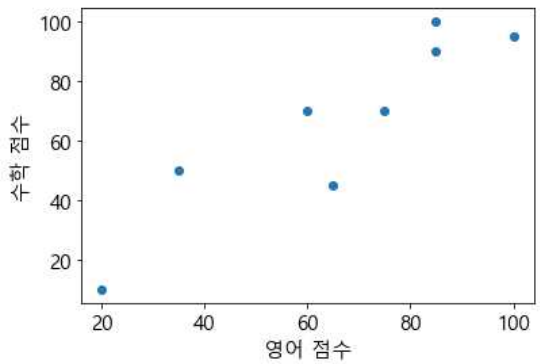
```
import pandas as pd
df = pd.read_excel('../Pandas/score.xlsx')
df
```

	지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기
0	1번	채치수	북산고	197	90	85	100	95	85	Python
1	2번	정대만	북산고	184	40	35	50	55	25	Java
2	3번	송태섭	북산고	168	80	75	70	80	75	Javascript
3	4번	서태웅	북산고	187	40	60	70	75	80	NaN
4	5번	강백호	북산고	188	15	20	10	35	10	NaN
5	6번	변덕규	능남고	202	80	100	95	85	80	C
6	7번	황태산	능남고	188	55	65	45	40	35	PYTHON
7	8번	윤대협	능남고	190	100	85	90	95	95	C#

```
df['학년'] = [3, 3, 2, 1, 1, 3, 2, 2]
df
```

	지원번호	이름	학교	키	국어	영어	수학	과학	사회	SW특기	학년
0	1번	채치수	북산고	197	90	85	100	95	85	Python	3
1	2번	정대만	북산고	184	40	35	50	55	25	Java	3
2	3번	송태섭	북산고	168	80	75	70	80	75	Javascript	2
3	4번	서태웅	북산고	187	40	60	70	75	80	NaN	1
4	5번	강백호	북산고	188	15	20	10	35	10	NaN	1
5	6번	변덕규	능남고	202	80	100	95	85	80	C	3
6	7번	황태산	능남고	188	55	65	45	40	35	PYTHON	2
7	8번	윤대협	능남고	190	100	85	90	95	95	C#	2

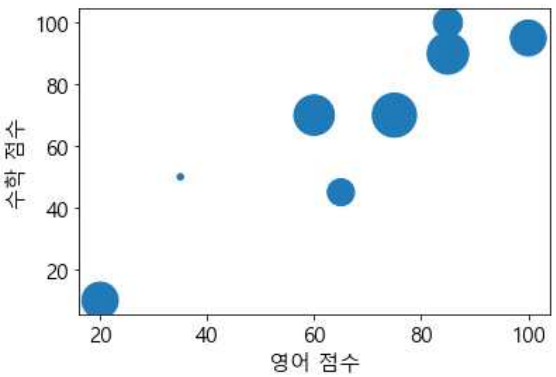
```
plt.scatter(df['영어'], df['수학'])
plt.xlabel('영어 점수')
plt.ylabel('수학 점수')
```



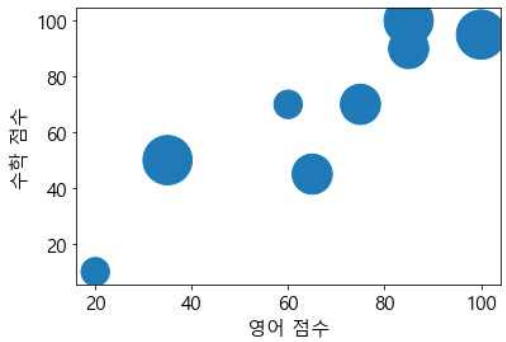

```
import numpy as np
sizes = np.random.rand(8) * 1000
sizes
```

```
array([427.98519264, 20.2128775 , 984.40437778, 825.01884958,
       665.10058677, 651.54107635, 370.6308619 , 866.26282854])
```

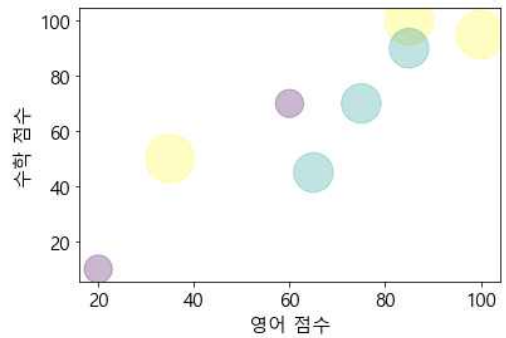
```
plt.scatter(df['영어'], df['수학'], s=sizes)
plt.xlabel('영어 점수')
plt.ylabel('수학 점수')
```



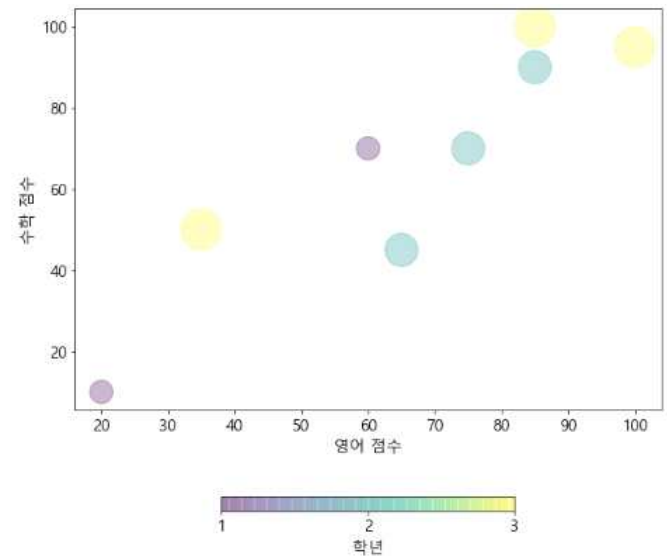
```
sizes = df['학년'] * 500  #1학년=500, 2학년=1000, 3학년=1500
plt.scatter(df['영어'], df['수학'], s=sizes)
plt.xlabel('영어 점수')
plt.ylabel('수학 점수')
```




```
plt.scatter(df['영어'], df['수학'], s=sizes, c=df['학년'], cmap='viridis', alpha=0.3)
plt.xlabel('영어 점수')
plt.ylabel('수학 점수')
```




```
plt.figure(figsize=(10, 10))
plt.scatter(df['영어'], df['수학'], s=sizes, c=df['학년'], cmap='viridis', alpha=0.3)
plt.xlabel('영어 점수')
plt.ylabel('수학 점수')
plt.colorbar(ticks=[1, 2, 3], label='학년', shrink=0.5, orientation='horizontal')
```

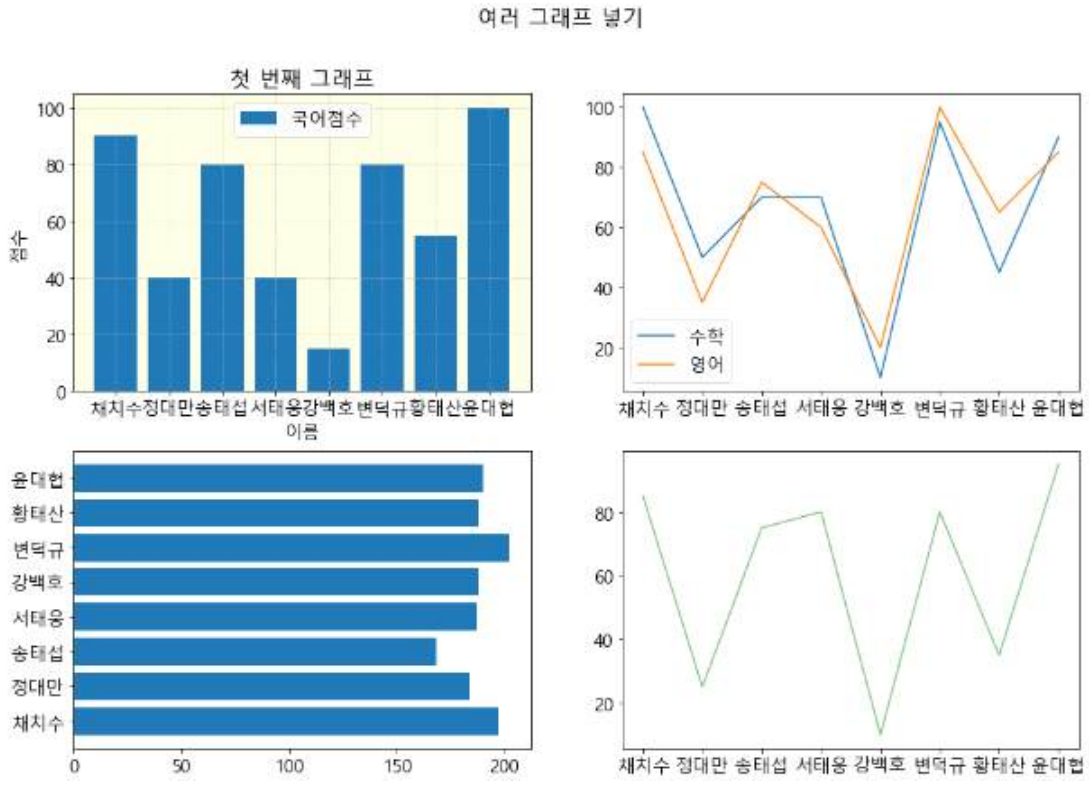


16. 여러 그래프

- 새로운 파일 '16.여러 그래프.ipynb'를 생성한다.

```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
import pandas as pd
df = pd.read_excel('../Pandas/score.xlsx')
df
```

```
fig, axs = plt.subplots(2, 2, figsize=(15, 10))  # 2x2 에 해당하는 plot들을 생성
fig.suptitle('여러 그래프 넣기')
#첫 번째 그래프
axs[0, 0].bar(df['이름'], df['국어'], label='국어점수')  #데이터 설정
axs[0, 0].set_title('첫 번째 그래프')  #제목
axs[0, 0].legend()  #범례
axs[0, 0].set(xlabel='이름', ylabel='점수')  #x, y 축 label
axs[0, 0].set_facecolor('lightyellow')  #전면 색
axs[0, 0].grid(linestyle='--', linewidth=0.5)

#두 번째 그래프
axs[0, 1].plot(df['이름'], df['수학'], label='수학')
axs[0, 1].plot(df['이름'], df['영어'], label='영어')
axs[0, 1].legend()

#세 번째 그래프
axs[1, 0].barh(df['이름'], df['키'])

#네 번째 그래프
axs[1, 1].plot(df['이름'], df['사회'], color='green', alpha=0.5)
```



17. 퀴즈

- 새로운 파일 '17.퀴즈.ipynb'를 생성한다.

```
# Matplotlib 퀴즈
다음은 대한민국 영화중에서 관객 수가 가장 많은 상위 8개의 데이터입니다.
주어진 코드를 이용하여 퀴즈를 풀어보시오.
```

> **Matplotlib 퀴즈**
>
> 다음은 대한민국 영화중에서 관객 수가 가장 많은 상위 8개의 데이터입니다.
> 주어진 코드를 이용하여 퀴즈를 풀어보시오.

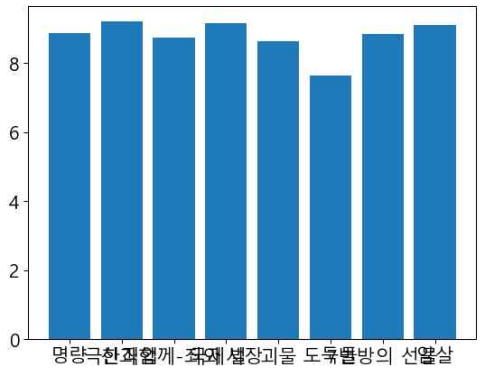
```
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False

data = {
    '영화' : ['명량', '극한직업', '신과함께-죄와 벌', '국제시장', '괴물', '도둑들', '7번방의 선물', '암살'],
    '개봉 연도' : [2014, 2019, 2017, 2014, 2006, 2012, 2013, 2015],
    '관객 수' : [1761, 1626, 1441, 1426, 1301, 1298, 1281, 1270],  #(단위 : 만명)
    '평점' : [8.88, 9.20, 8.73, 9.16, 8.62, 7.64, 8.83, 9.10]
}

df = pd.DataFrame(data)
df
```

	영화	개봉 연도	관객 수	평점
0	명량	2014	1761	8.88
1	극한직업	2019	1626	9.20
2	신과함께-죄와 벌	2017	1441	8.73
3	국제시장	2014	1426	9.16
4	괴물	2006	1301	8.62
5	도둑들	2012	1298	7.64
6	7번방의 선물	2013	1281	8.83
7	암살	2015	1270	9.10

1) 영화 데이터를 활용하여 x 축은 영화, y 축은 평점인 막대그래프를 만드시오.

> 1) 영화 데이터를 활용하여 x 축은 영화, y 축은 평점인 막대 그래프를 만드시오.

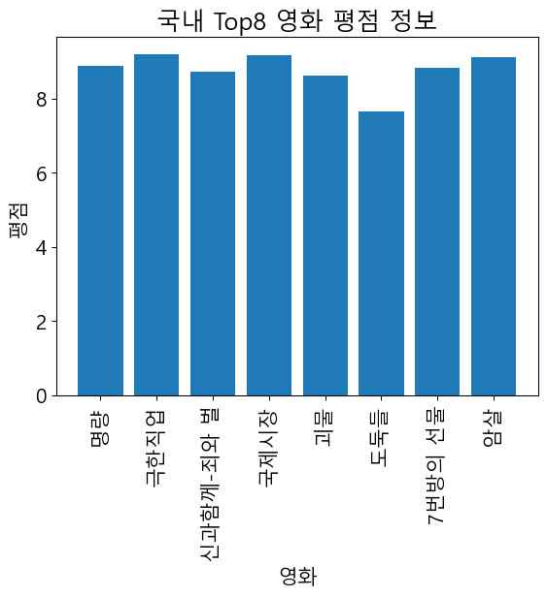
```
plt.bar(df['영화'], df['평점'])
```



2) 앞에서 만든 막대그래프에 제시된 세부 사항을 적용하시오.
- 제목: 국내 Top8 영화 평점 정보
- x축 label : 영화(90도 회전)
- y축 label : 평점

> 2) 앞에서 만든 막대 그래프에 제시된 세부 사항을 적용하시오.
>
> - 제목: 국내 Top8 영화 평점 정보
> - x축 label : 영화(90도 회전)
> - y축 label : 평점

```
plt.bar(df['영화'], df['평점'])
plt.title('국내 Top8 영화 평점 정보')
plt.xlabel('영화')
plt.xticks(rotation=90)
plt.ylabel('평점')
```



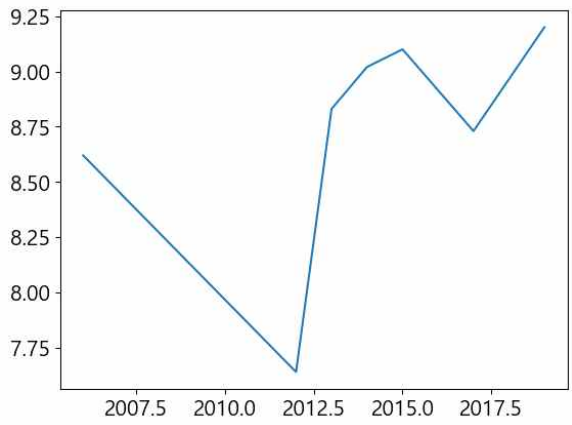
3) 개봉 연도별 평점 변화 추이를 꺽은선 그래프로 그리시오.
연도별 평균 데이터를 구하는 코드는 다음과 같습니다.

> 3) 개봉 연도별 평점 변화 추이를 꺽은선 그래프로 그리시오.
>
> 연도별 평균 데이터를 구하는 코드는 다음과 같습니다.

```
df_group = df.groupby('개봉 연도')[['관객 수','평점']].mean()
df_group
```

개봉 연도	관객 수	평점
2006	1301.0	8.62
2012	1298.0	7.64
2013	1281.0	8.83
2014	1593.5	9.02
2015	1270.0	9.10
2017	1441.0	8.73
2019	1626.0	9.20

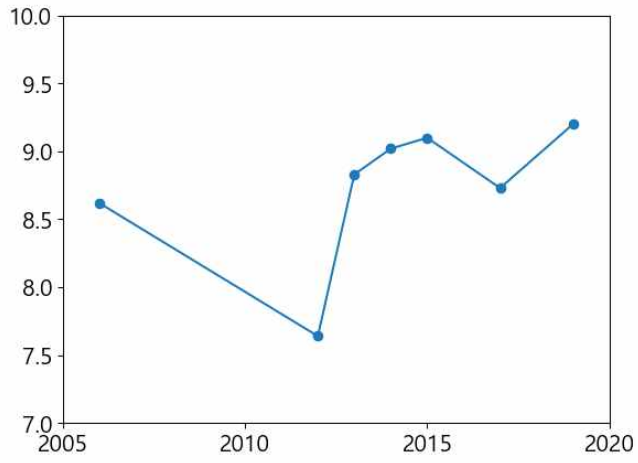
```
plt.plot(df_group.index, df_group['평점'])
```



4)앞에서 만든 그래프에 제시된 세부 사항을 적용하시오.
- marker : 'o'
- x축 눈금 : 5년 단위(2005, 2010, 2015, 2020)
- y축 범위 : 최소 7, 최대 10

> 4)앞에서 만든 그래프에 제시된 세부 사항을 적용하시오.
>
> - marker : 'o'
> - x축 눈금 : 5년 단위(2005, 2010, 2015, 2020)
> - y축 범위 : 최소 7, 최대 10

```
plt.plot(df_group.index, df_group['평점'], marker='o')
plt.xticks([2005, 2010, 2015, 2020])
plt.ylim(7, 10)
```



5) 평점이 9점 이상인 영화의 비율을 확인할 수 있는 원 그래프를 제시된 세부 사항을 적용하여 그리시오.

- label : 9점 이상/ 9점 미만
- 퍼센트 : 소수점 첫째자리까지 표시
- 범례 : 그래프 우측에 표시

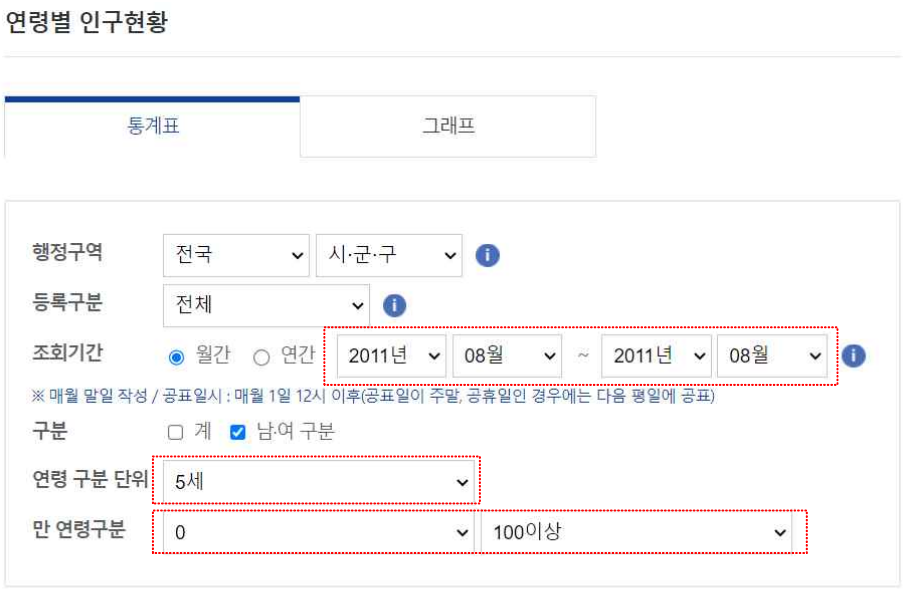
> **5) 평점이 9점이상인 영화의 비율을 확인할 수 있는 원 그래프를 제시된 세부 사항을 적용하여 그리시오.**
>
> - label : 9점 이상/ 9점 미만
> - 퍼센트 : 소수점 첫째자리까지 표시
> - 범례 : 그래프 우측에 표시

```
filt = df['평점'] >= 9.0
df[filt]
df[~filt]
values = [len(df[filt]), len(df[~filt])]
labels = ['9점 이상', '9점 미만']
plt.pie(values, labels=labels, autopct='%.1f%%')
plt.legend(loc=(1, 0.3))
plt.show()
```

18. 인구 피라미드

2011년8월 남여, 나이별 전국 인구수를 시각화하고 2021년8월 남여, 나이별 전국 인구수를 시각화하여 인구수를 비교한다.

- **새로운 파일 '18.인구 피라미드.ipynb'를 생성한다.**
- **[구글]-[연령별 인구현황]을 검색하여 행정안전부에서 제공되는 아래 조건의 데이터(엑셀)를 이용한다.**
- **아래 조건의 검색 결과를 엑셀 파일로 저장한다.**



- **남자 데이터를 처리한다. (행정기관별 0세~100세까지 전국 시도의 남자 데이터만 저장한다.)**

```
import pandas as pd
df_m=pd.read_excel('2011_인구현황_월간.xlsx', skiprows=3, index_col='행정기관', usecols='B, E:Y')
df_m.head(1)
```

행정기관	0~4세	5~9세	10~14세	15~19세	20~24세	25~29세	30~34세	35~39세	40~44세
전국	1,195,951	1,233,465	1,663,534	1,890,498	1,667,705	1,844,651	2,048,038	2,178,901	2,335,456

- **DataFrame(df_m) '전국'(인덱스 0) 모든 칼럼의 데이터에 콤마를 제거한 후 다시 저장한다.**

```
df_m.iloc[0]=df_m.iloc[0].str.replace(',', '').astype(int)  #1,195,951 -> 1195951 (정수형)
df_m
```

행정기관	0~4세	5~9세	10~14세	15~19세	20~24세	25~29세	30~34세	35~39세	40~44세	
전국	1195951	1233465	1663534	1890498	1667705	1844651	2048038	2178901	2335456	2

- **여자 데이터를 처리한다. (행정기관별 0세~100세까지 전국 시도의 여자 데이터만 저장한다.)**

```
df_w=pd.read_excel('2011_인구현황_월간.xlsx', skiprows=3, index_col='행정기관', usecols='B, AB:AV')
df_w.head(1)
```

행정기관	0~4세.1	5~9세.1	10~14세.1	15~19세.1	20~24세.1	25~29세.1	30~34세.1	35~39세.1	40~44세.1
전국	1,125,896	1,141,102	1,524,143	1,668,412	1,506,676	1,737,973	1,963,094	2,092,175	2,257,824

- **DataFrame(df_w) '전국'(인덱스 0) 모든 칼럼의 데이터에 콤마를 제거한 후 다시 저장한다.**

```
df_w.iloc[0]=df_w.iloc[0].str.replace(',', '').astype(int)
df_w
```

행정기관	0~4세.1	5~9세.1	10~14세.1	15~19세.1	20~24세.1	25~29세.1	30~34세.1	35~39세.1	40~44세.1
전국	1125896	1141102	1524143	1668412	1506676	1737973	1963094	2092175	2257824

- **DataFrame(df_m)의 모든 칼럼 이름들을 출력한다.**

```
df_m.columns
```

```
Index(['0~4세', '5~9세', '10~14세', '15~19세', '20~24세', '25~29세', '30~34세',
       '35~39세', '40~44세', '45~49세', '50~54세', '55~59세', '60~64세', '65~69세',
       '70~74세', '75~79세', '80~84세', '85~89세', '90~94세', '95~99세', '100세 이상'], dtype='object')
```

- **DataFrame(df_w)의 모든 칼럼 이름들을 출력한다.**

```
df_w.columns
```

```
Index(['0~4세.1', '5~9세.1', '10~14세.1', '15~19세.1', '20~24세.1', '25~29세.1',
       '30~34세.1', '35~39세.1', '40~44세.1', '45~49세.1', '50~54세.1', '55~59세.1',
       '60~64세.1', '65~69세.1', '70~74세.1', '75~79세.1', '80~84세.1', '85~89세.1',
       '90~94세.1', '95~99세.1', '100세 이상.1'], dtype='object')
```

- **DataFrame(df_w)의 칼럼 이름들을 DataFrame(df_m)의 칼럼 이름들로 같게 변경한다.**

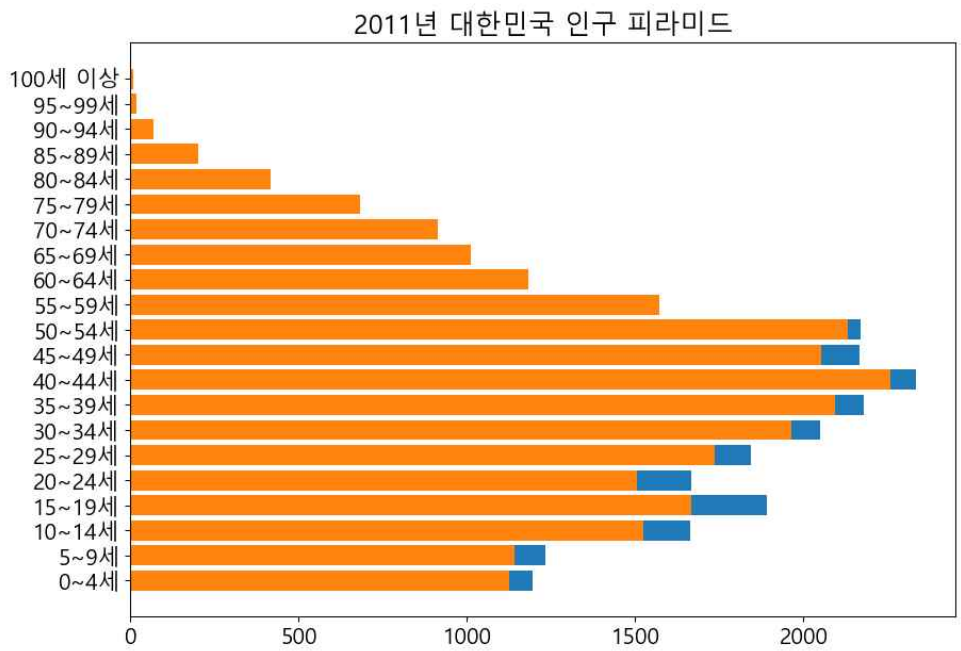
```
df_m.columns = df_w.columns
df_w.columns
```

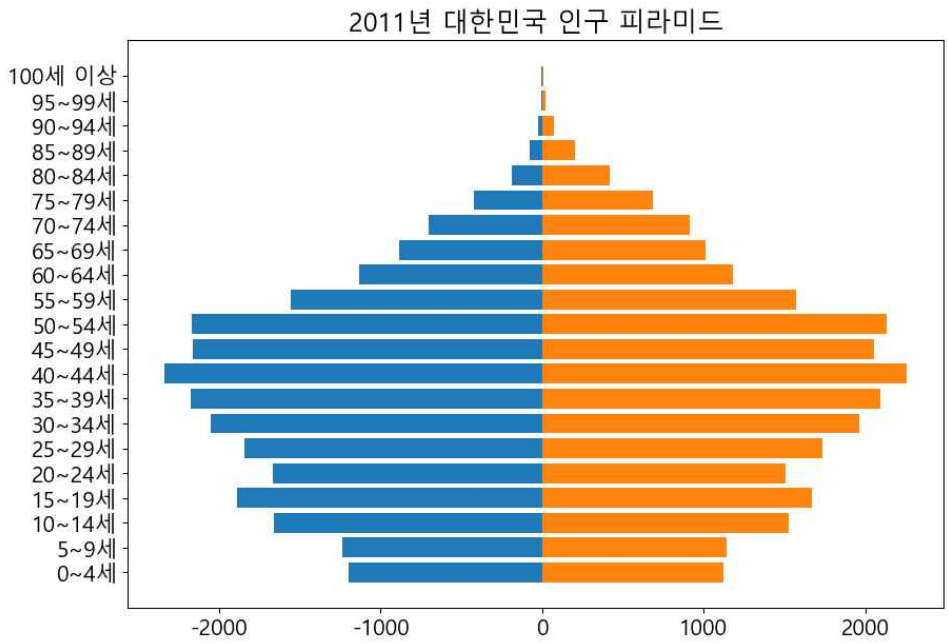
```
Index(['0~4세', '5~9세', '10~14세', '15~19세', '20~24세', '25~29세', '30~34세',
       '35~39세', '40~44세', '45~49세', '50~54세', '55~59세', '60~64세', '65~69세',
       '70~74세', '75~79세', '80~84세', '85~89세', '90~94세', '95~99세', '100세 이상'], dtype='object')
```

- **맷플롯립(matplotlib) 라이브러리를 import하고 한글 폰트와 폰트 사이즈를 지정한 후 인구수를 시각화 한다.**

```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

```
plt.figure(figsize=(10,7))
plt.barh(df_m.columns, -df_m.iloc[0] // 1000)  #단위: 천명
plt.barh(df_w.columns,  df_w.iloc[0] // 1000)
plt.title('2011년 대한민국 인구 피라미드')
plt.savefig('2011_인구피라미드.png', dpi=100)
plt.show()
```





연령별 인구현황

통계표 그래프

행정구역 전국 시·군·구

등록구분 전체

조회기간 월간 연간 2021년 08월 ~ 2021년 08월

※ 매월 말일 작성 / 공표일시 : 매월 1일 12시 이후(공표일이 주말, 공휴일인 경우에는 다음 평일에 공표)

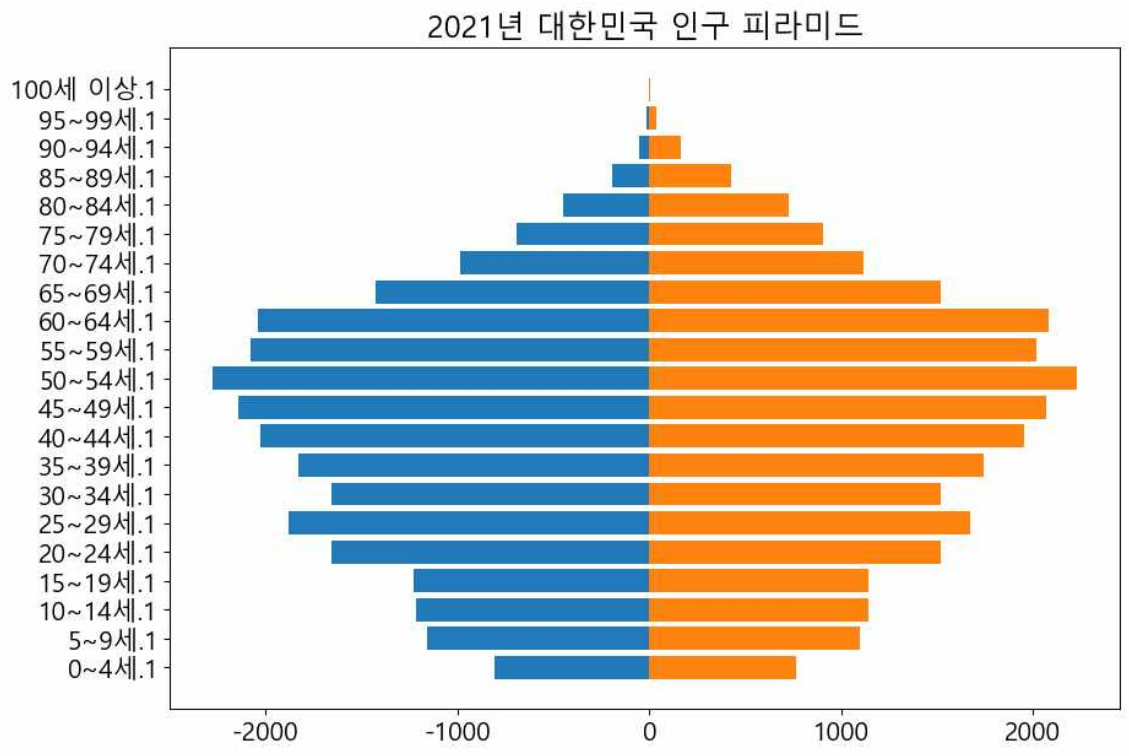
구분 계 남·여 구분

연령 구분 단위 5세

만 연령구분 0 100이상

```
df_m=pd.read_excel('2021_인구현황_월간.xlsx', skiprows=3, index_col='행정기관', usecols='B, E:Y')
df_m.iloc[0]=df_m.iloc[0].str.replace(',', '').astype(int)
df_w=pd.read_excel('2021_인구현황_월간.xlsx', skiprows=3, index_col='행정기관', usecols='B, AB:AV')
df_w.iloc[0]=df_w.iloc[0].str.replace(',', '').astype(int)
df_m.columns = df_w.columns  #칼럼 명을 통일
```

```
plt.figure(figsize=(10,7))
plt.barh(df_m.columns, -df_m.iloc[0] // 1000)  #단위: 천명
plt.title('2021년 대한민국 인구 피라미드')
plt.barh(df_w.columns, df_w.iloc[0] // 1000)
plt.savefig('2021_인구피라미드.png', dpi=100)
plt.show()
```



19. 출생아 수 및 합계출산율

- 새로운 파일 '19.출생아 수 및 합계출산율.ipynb'를 생성한다.
- [구글]-[출생아 수 합계출산율]을 검색하여 e-나라지표에서 제공되는 데이터(엑셀)를 이용한다.

- nrows=2 헤더를 제외한 2행을 추출하고 index_col=0 0칼럼을 인덱스로 지정한 후 DataFrame(df)에 저장한다.

```
import pandas as pd
df=pd.read_excel('stat_2101.xlsx', skiprows=2, nrows=2, index_col=0)
df
```

	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021
출생아 수	484.600	436.500	435.400	438.400	406.200	357.800	326.800	302.700	272.300	260.600
합계 출산율	1.297	1.187	1.205	1.239	1.172	1.052	0.977	0.918	0.837	0.808

- DataFrame(df)의 index의 값들을 출력하면 공백에 이상한 문자가 저장되어있다.

```
df.index
```

```
Index(['출생아 수', '합계 출산율'], dtype='object')
```

```
df.index.values
```

```
array(['출생아\xa0수', '합계\xa0출산율'], dtype=object)
```

- DataFrame(df)의 인덱스 이름 공백에 이상한 문자가 들어가 있으므로 이름을 새로 변경하여 저장한다.

```
df.rename(index={'출생아\xa0수':'출생아 수', '합계\xa0출산율':'합계 출산율'}, inplace=True)
df
```

	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021
출생아 수	484.600	436.500	435.400	438.400	406.200	357.800	326.800	302.700	272.300	260.600
합계 출산율	1.297	1.187	1.205	1.239	1.172	1.052	0.977	0.918	0.837	0.808

```
df.index.value
```

```
array(['출생아 수', '합계 출산율'], dtype=object)
```

- DataFrame(df)에서 '출생아 수'를 출력하려면 아래 두가지(loc, iloc) 방법을 이용할 수 있다.

```
df.loc[['출생아 수']]
```

	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021
출생아 수	484.6	436.5	435.4	438.4	406.2	357.8	326.8	302.7	272.3	260.6

```
df.iloc[[0]]
```

	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021
출생아 수	484.6	436.5	435.4	438.4	406.2	357.8	326.8	302.7	272.3	260.6

- **DataFrame(df)에서 'T'를 이용하여 행과 열을 바꾼 후 DataFrme(df)에 새로 저장한다.**

```
df = df.T
df
```

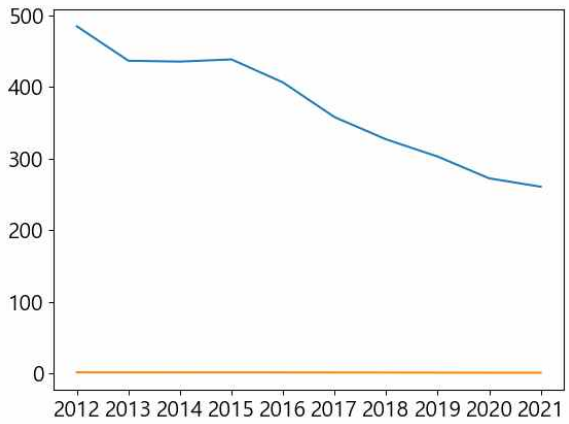
	출생아 수	합계 출산율
2012	484.6	1.297
2013	436.5	1.187
2014	435.4	1.205
2015	438.4	1.239
2016	406.2	1.172

- **데이터 시각화를 위하여 맷플롯립(matplotlib) 라이브러리를 import하고 한글 폰트와 폰트사이즈를 설정한다.**

```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
```

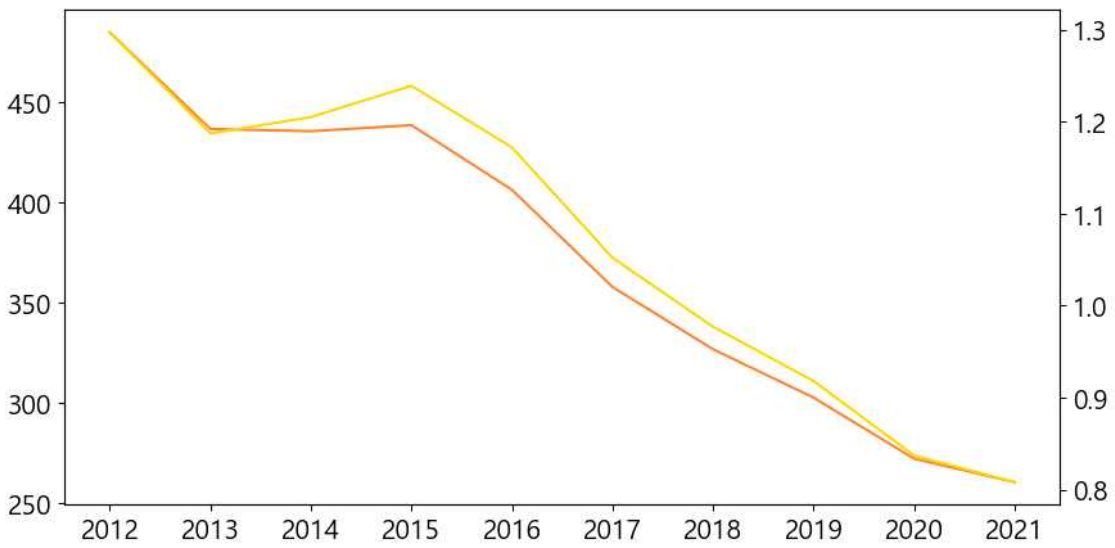
- **X축을 인덱스, Y축은 '출생아 수', '합계 출산율'로 Line 그래프로 시각화 작업을 한다.**

```
plt.plot(df.index, df['출생아 수'])
plt.plot(df.index, df['합계 출산율'])
```



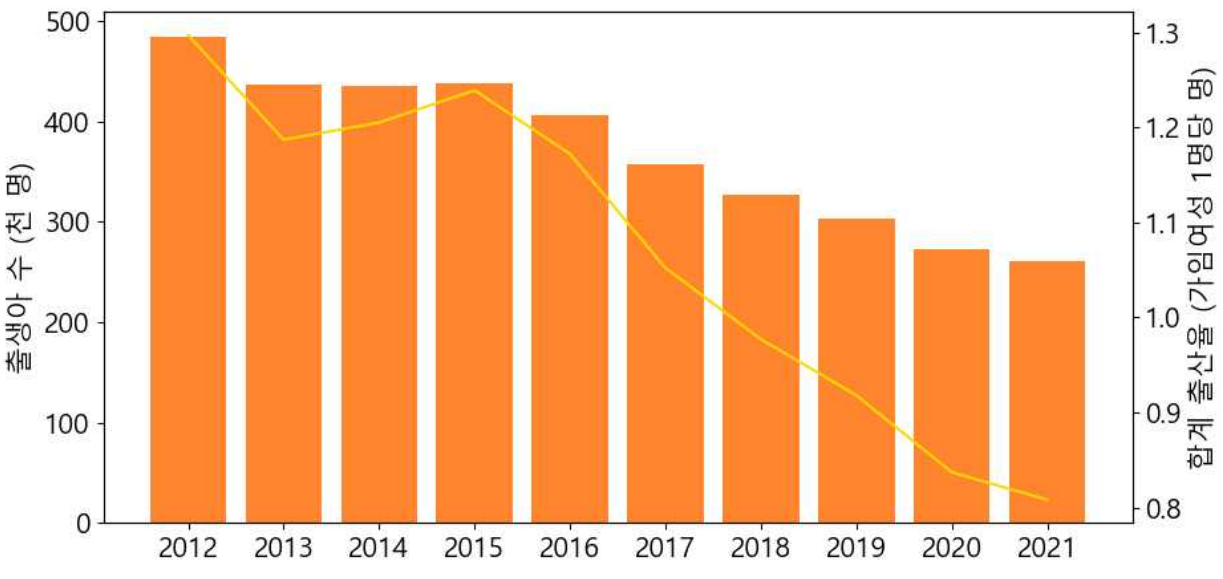
- '합계 출산율'의 값이 너무 작으므로 X축은 공유하고 '합계 출산율'의 Y축은 오른쪽으로 지정한다.

```
fig, ax1 = plt.subplots(figsize=(10, 5))
ax1.plot(df.index, df['출생아 수'], color='#ff812d')
ax2 = ax1.twinx()
ax2.plot(df.index, df['합계 출산율'], color='#ffd100')
```

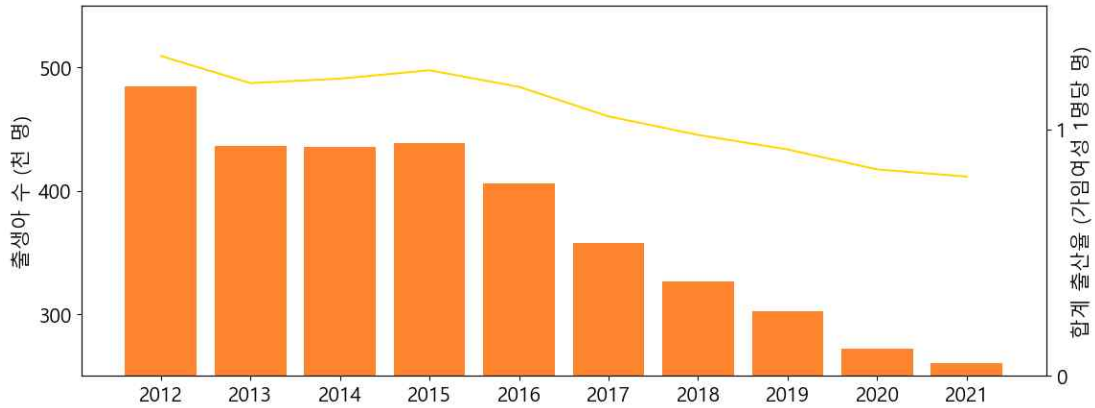


- '출생아 수'를 막대그래프로 변경하고 Y축의 레이블을 아래와 같이 변경한다.

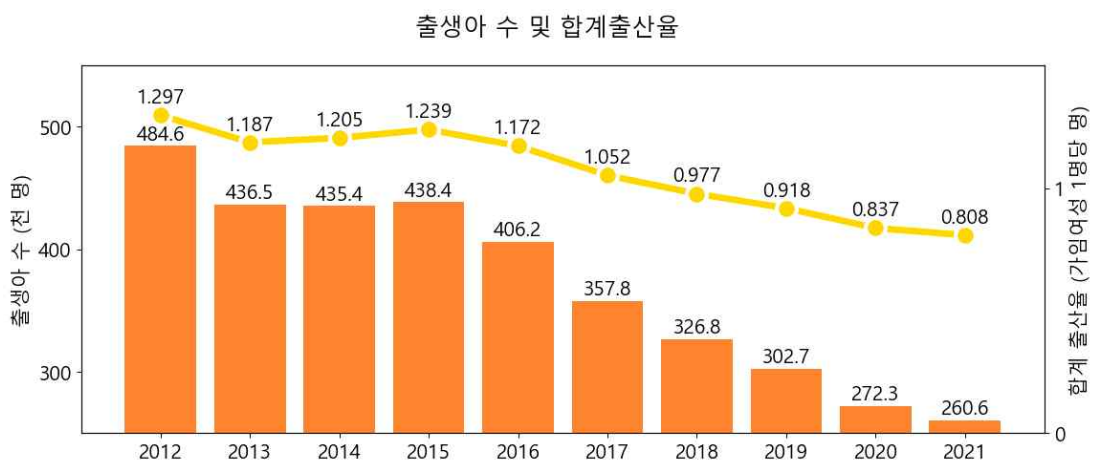
```
fig, ax1 = plt.subplots(figsize=(10, 5))
ax1.bar(df.index, df['출생아 수'], color='#ff812d')
ax1.set_ylabel('출생아 수 (천 명)')
ax2 = ax1.twinx()
ax2.plot(df.index, df['합계 출산율'], color='#ffd100')
ax2.set_ylabel('합계 출산율 (가임여성 1명당 명)')
```



```
fig, ax1 = plt.subplots(figsize=(13, 5))
ax1.bar(df.index, df['출생아 수'], color='#ff812d')
ax1.set_ylabel('출생아 수 (천 명)')
ax1.set_ylim(250, 550)
ax1.set_yticks([300, 400, 500])
ax2 = ax1.twinx()
ax2.plot(df.index, df['합계 출산율'], color='#ffd100')
ax2.set_ylabel('합계 출산율 (가임여성 1명당 명)')
ax2.set_ylim(0, 1.5)
ax2.set_yticks([0, 1])
```




```
fig, ax1 = plt.subplots(figsize=(13, 5))
fig.suptitle('출생아 수 및 합계출산율')
ax1.bar(df.index, df['출생아 수'], color='#ff812d')
ax1.set_ylabel('출생아 수 (천 명)')
ax1.set_ylim(250, 550)
ax1.set_yticks([300, 400, 500])
for idx, val in enumerate(df['출생아 수']):
    ax1.text(idx,val+5, val, ha='center')
ax2 = ax1.twinx()
ax2.plot(df.index, df['합계 출산율'], color='#ffd100', marker='o', ms=15, lw=5, mec='w', mew=3)
ax2.set_ylabel('합계 출산율 (가임여성 1명당 명)')
ax2.set_ylim(0, 1.5)
ax2.set_yticks([0, 1])
for idx, val in enumerate(df['합계 출산율']):
    ax2.text(idx, val+0.05, val, ha='center')
```



• Flask

Flask 는 소규모의 어플리케이션을 빠르게 만들 수 있고, 배포 환경에 따라 대규모 어플리케이션의 기능 확장의 역할을 하기 쉬운 장점이, Django는 대규모의 어플리케이션을 빠르게 만들 수 있으며, 기본으로 제공 해 주는 기능이 많은 장점이 있다.

- **가상머신을 생성한다.**

```
C:\data\python\flask>python -m venv myenv
```

- **가상머신을 실행한다.**

```
C:\data\python\flask>.\myenv\Scripts\activate
```

- **플라스크(Flask) 패키지를 설치한다.**

```
(myenv) C:\data\python\flask>pip install flask
```

- **홈페이지를 작성한다.**

[flask] app.py

```
from flask import Flask, render_template
app = Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html', title='홈페이지')

if __name__ == '__main__':
    app.run(port=5001, debug=True)
```

[flask]-[templates] base.html

```
<html lang="en">
<head>
    <meta name="viewport" content="width=<device-width>, initial-scale=1.0">
    <link rel="stylesheet" href="/static/css/style.css"/>
    <script src="http://code.jquery.com/jquery-1.9.1.js"></script>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css" rel="stylesheet">
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/js/bootstrap.bundle.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/handlebars.js/3.0.1/handlebars.js"></script>
    <title>{{title}}</title>
</head>
<body>
    <div class="container">
        {%include 'header.html'%}
        {%block main_area%}
        {%endblock%}
        {%include 'footer.html'%}
    </div>
</body>
</html>
```

[flask]-[templates] index.html

```
{%extends 'base.html'%}
{%block main_area%}
    <div class="row my-5">
        <div class="col">
            <h1 class="text-center">홈페이지입니다...</h1>
        </div>
    </div>
{%endblock%}
```

- 스타일 파일을 작성한다.

[flast]-[static]-[css] style.css

```
@font-face {
    font-family: 'GmarketSansMedium';
    src: url('https://cdn.jsdelivr.net/gh/projectnoonnu/noonfonts_2001@1.1/GmarketSansMedium.woff') format('woff');
    font-weight: normal;
    font-style: normal;
}
* {
    font-family: 'GmarketSansMedium';
}
```

- 헤더 페이지를 작성한다.

[flask]-[templates] menu.html

```
<div>
    <a href="/score/page1" class="me-3">학교검색</a>
    <a href="/score/page2" class="me-3">이름검색</a>
    <a href="/score/page3" class="me-3">이름검색(JSON)</a>
    <a href="/score/graph" class="me-3">그래프</a>
</div>
```

[flask]-[templates] header.html

```
<div class="row mt-5">
    <div class="col">
        <h1 class="text-center mb-5">Flask</h1>
        {%include 'menu.html'%}
        <hr>
    </div>
</div>
```

- 푸더 페이지를 작성한다.

[flask]-[templates] footer.html

```
<div class="row">
    <div class="col">
        <hr>
        <h4 class="text-center">Copyright 2023 홍길동 All rights reserved.</h4>
    </div>
</div>
```

- 웹서버를 실행하고 localhost:5001로 접속한다.

```
(myenv) C:\data\python\flask>python app.py
```

- 학교별 학생 성적목록과 학생정보를 출력하는 페이지를 작성한다.

[flask] scoreRoute.py

```
from flask import Blueprint, render_template, request
import pandas as pd
score = Blueprint('score', __name__)

data = {
    '이름' : ['채치수', '정대만', '송태섭', '서태웅', '강백호', '변덕규', '황태산', '윤대협'],
    '학교' : ['북산고', '북산고', '북산고', '북산고', '북산고', '능남고', '능남고', '능남고'],
    '키' : [197, 184, 168, 187, 188, 202, 188, 190],
    '국어' : [90, 40, 80, 40, 15, 80, 55, 100],
    '영어' : [85, 35, 75, 60, 20, 100, 65, 85],
    '수학' : [100, 50, 70, 70, 10, 95, 45, 90],
    '과학' : [95, 55, 80, 75, 35, 85, 40, 95],
    '사회' : [85, 25, 75, 80, 10, 80, 35, 95],
    'SW특기' : ['Python', 'Java', 'Javascript', '', '', 'C', 'PYTHON', 'C#']
}

@score.route('/page1')
def score_page1():
    if request.args.get('school') == None:
        school = '북산고'
    else:
        school =request.args.get('school')
    df = pd.DataFrame(data)
    filter = df['학교']==school
    df1 = df[filter]
    df1 = df1.loc[:, ['학교','이름','국어','영어','수학','과학','사회']]
    df2 = df[filter]
    df2 = df2[['이름','학교','키','SW특기']]
    return render_template('page1.html', table1=df1.to_html(classes='table', table_id='tbl1'),
                                         table2=df2.to_html(classes='table', table_id='tbl2'))
```

- scoreRoute.py 파일을 app.py에 등록한다.

[flask] app.py

```
from flask import Blueprint, render_template, request
from scoreRoute import score

app = Flask(__name__)
app.register_blueprint(score, url_prefix='/score')

@app.route('/')
def index():
    return render_template('index.html', title='홈페이지')

if __name__ == '__main__':
    app.run(port=5001, debug=True)
```

[flask]-[templates] page1.html

```
{%extends 'base.html'%}
{%block main_area%}
    <div>
        <div class="mb-3">
            <a href="/score/page1?school=북산고" class="btn btn-primary">북산고</a>
            <a href="/score/page1?school=능남고" class="btn btn-primary">능남고</a>
        </div>
        {{table1|safe}}
        {{table2|safe}}
    </div>
    <script>
        $("#tbl1 thead tr th:first").text("No")
        $("#tbl2 thead tr th:first").text("No")
        $("th, td").addClass("text-center")
    </script>
{%endblock%}
```

- 이름검색 후 학생정보를 출력하는 페이지를 작성한다.

[flask] scoreRoute.py

```
from flask import Blueprint, render_template, request
import pandas as pd

score = Blueprint('score', __name__)

data = {
    ...
}

@score.route('/page1')
def score_page1():
    ...

@score.route('/page2')
def score_page2():
    return render_template('page2.html')

@score.route('/table', methods=['POST'])
def score_table():
    if  request.args.get('query') == None:
        query=''
    else:
        query=request.args.get('query')

    df = pd.DataFrame(data)
    filter = df['이름'].str.contains(query)
    df = df[filter]
    return df.to_html(classes='table')
```

[flask]-[templates] page2.html

```
{%extends 'base.html'%}
{%block main_area%}
    <div>
        <form name="frm" class="col-6 mb-3">
            <div class="input-group">
                <input name="query" class="form-control" placeholder="이름">
                <button class="btn btn-primary">검색</button>
            </div>
        </form>
        <div id="tbl"></div>
    </div>
    <script>
        getList();
        $(frm).on("submit", function(e){
            e.preventDefault();
            getList();
        });
        function getList(){
            $.ajax({
                type:"post",
                url:"/score/table",
                data:{query:$(frm.query).val()},
                success:function(data){
                    $("#tbl").html(data);
                    $("#tbl thead tr th:first").text("No")
                    $("th, td").addClass("text-center")
                }
            });
        }
    </script>
{%endblock%}
```

- 이름검색 결과를 JSON으로 만들어 출력하는 페이지를 작성한다.

[flask] scoreRoute.py

```
...
@score.route('/page3')
def score_page3():
    return render_template('page3.html')

@score.route('/table.json')
def score_json():
    query=request.args['query']
    df = pd.DataFrame(data)
    filter = df['이름'].str.contains(query)
    df = df[filter]
    json = df.to_json(orient='records')
    return  json
```

[flak]-[templates] page3.html

```
{%extends 'base.html' %}
{%block main_area%}
    <div>
        <form name="frm" class="col-6 mb-3">
            <div class="input-group">
                <input name="query" class="form-control">
                <button class="btn btn-primary">검색</button>
            </div>
        </form>
        <div id="tbl"></div>
    </div>
    {% raw %}
    <script type="x-handlebars-template" id="temp">
        <table class="table">
            {{#each .}}
            <tr>
                <td>{{이름}}</td>
                <td>{{학교}}</td>
                <td>{{키}}</td>
                <td>{{SW특기}}</td>
                <td>{{국어}}</td>
                <td>{{영어}}</td>
                <td>{{수학}}</td>
                <td>{{과학}}</td>
                <td>{{사회}}</td>
            </tr>
            {{/each}}
        </tr>
    </script>
    {% endraw %}
    <script>
        getList();
        $(frm).on("submit", function(e){
            e.preventDefault();
            getList();
        });
        function getList(){
            $.ajax({
                type:"get",
                url:"/score/table.json",
                data:{query:$(frm.query).val()},
                dataType:"json",
                success:function(data){
                    const temp=Handlebars.compile($("#temp").html());
                    $("#tbl").html(temp(data));
                }
            });
        }
    </script>
{%endblock%}
```

- 과목별 학생들의 성적 그래프를 출력하는 페이지를 작성한다.

[flask] scoreRoute.py

```
from io import BytesIO
from flask import Blueprint, render_template, request,  send_file
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'  #맑은 고딕
matplotlib.rcParams['font.size'] = 15  #글자 크기
matplotlib.rcParams['axes.unicode_minus'] = False  #한글 폰트 사용 시 마이너스 글자가 깨지는 현상을 해결
plt.switch_backend('agg')
...
@score.route('/graph')
def score_graph():
    return render_template('graph.html')

@score.route('/graph/<subject>')
def score_chart1(subject):
    df = pd.DataFrame(data)
    plt.figure(figsize=(10, 5))
    plt.title(subject + ' 성적')
    plt.bar(df['이름'], df[subject])
    for idx, val in enumerate(df[subject]):
        plt.text(idx, val+1, val, ha='center')

    img = BytesIO()
    plt.savefig(img, format='png', dpi=200)
    img.seek(0)
    return send_file(img, mimetype='image/png')
```

[flak]-[templates] graph.html

```
{%extends 'base.html'%}
{%block main_area%}
    <div>
        <div>
            <button id="btn_kor" class="btn btn-primary">국어</button>
            <button id="btn_eng" class="btn btn-primary">영어</button>
            <button id="btn_mat" class="btn btn-primary">수학</button>
        </div>
        <div>
            <img src="/score/graph/국어" id="img_graph" alt="Chart" width="100%">
        </div>
    </div>
    <script>
        $("#btn_kor, #btn_eng, #btn_mat").on("click", function(){
            $("#img_graph").attr("src", "/score/graph/" + title)
        })
    </script>
{%endblock%}
```