프로그래밍특론

PANDAS를 이용한 데이터 분석 기초

이정미 교수

KMU 국민대학교 KOOKMIN UNIVERSITY

PANDAS를 이용한 데이터 분석 기초

이정미 교수

국민대학교
KOOKMIN UNIVERSITY

Pandas

정의

- 파이썬에서 사용할 수 있는 데이터 처리 및 분석 라이브러리

특징

- import 하여 사용
- 행과 열로 이루어진 데이터 객체를 다룰 수 있음
- 보다 안정적으로 대용량 데이터 처리 가능
- 시리즈(Series), 데이터프레임(DataFrame), 패널(Panel), 3종류의 데이터 구조 사용

pandas

import pandas

- pandas 라이브러리는 패키지 이름을 자주 호출 함으로, 별칭을 주어 짧게 줄여 쓰자.

import pandas as pd

- 위와 같이 라이브러리를 가져온 후,
  다음과 같이 pandas 라이브러리의 함수들을 간단히 불러쓰자

```
import pandas as pd
```

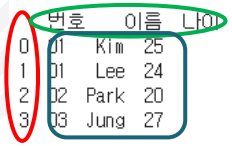
pandas의 활용

- Pandas는 크게 세 가지의 자료구조를 지원한다.
  - 1차원 자료구조인 Series
  - 2차원 자료구조인 DataFrame
  - 3차원 자료구조인 Panel

- Pandas의 데이터형
  - Objects : 문자 또는 문자열 형
  - Int64 : 정수형
  - Float64 : 실수형

02

DataFrame



정의 및 구성

- 행(row)과 열(column)을 가지는 자료구조 ***(#)***
- 인덱스(index), 열(column), 값(value)으로 구성

DataFrame 만들기

```
# pandas import
import pandas as pd

# 리스트 만들기 (리스트 안의 리스트)
data = [
        ['01', 'Kim', 25],
        ['01', 'Lee', 24],
        ['02', 'Park', 20],
        ['03', 'Jung', 27],
]

# 리스트를 DataFrame으로 변형하여 변수에 저장
df = pd.DataFrame(data, columns=['번호','이름','나이'])

print(df)
```

DataFrame 행과 열 추출

	번호	이름	나이
0	01	Kim	25
1	01	Lee	24
2	02	Park	20
3	03	Jung	27

- **열 추출: "열 이름" 이용**

```
# DataFrame변수이름['열 이름']
print(df['번호'])
```

```
0    01
1    01
2    02
3    03
Name: 번호, dtype: object
```

```
# 한 번에 두개 이상의 열 추출도 가능
print(df[['이름','나이']])
```

```
    이름  나이
0   Kim  25
1   Lee  24
2  Park  20
3  Jung  27
```

- **행 추출: loc 함수 이용**

```
# DataFram변수이름.loc[index번호]
print(df.loc[0])
```

```
번호     01
이름    Kim
나이     25
Name: 0, dtype: object
```

```
# 한 번에 두개 이상의 행 추출도 가능
print(df.loc[[0,2]])
```

```
   번호    이름  나이
0  01   Kim  25
2  02  Park  20
```

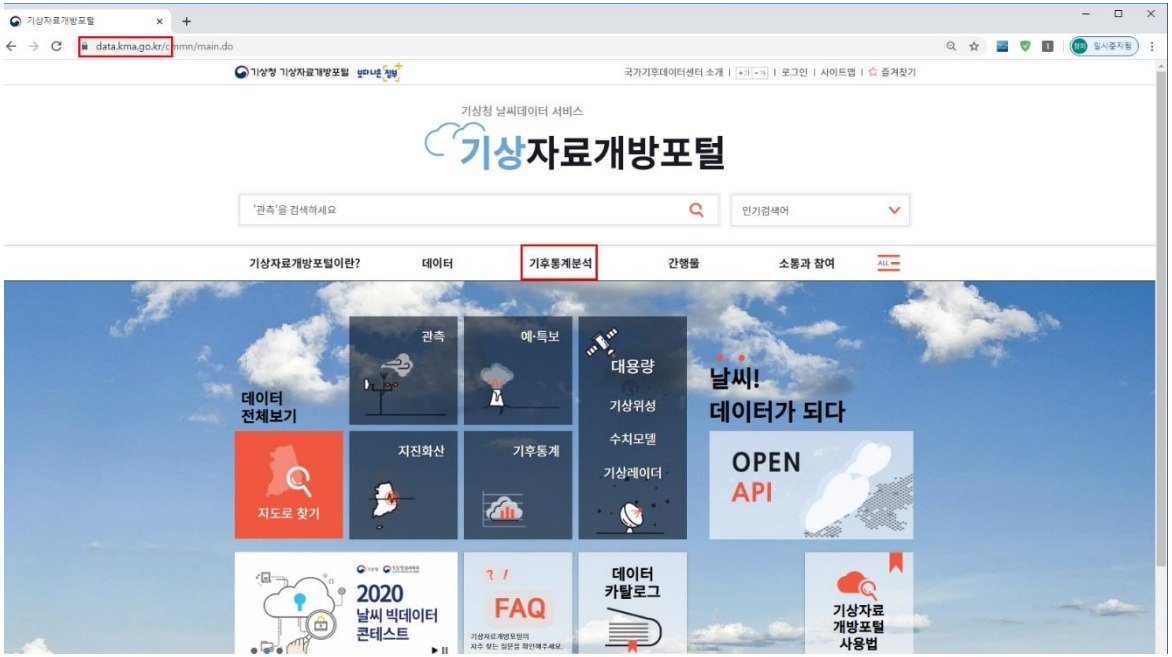
index location을 의미

목차

- 공공데이터 사용하기
- 데이터 저장하기 (csv,excel)

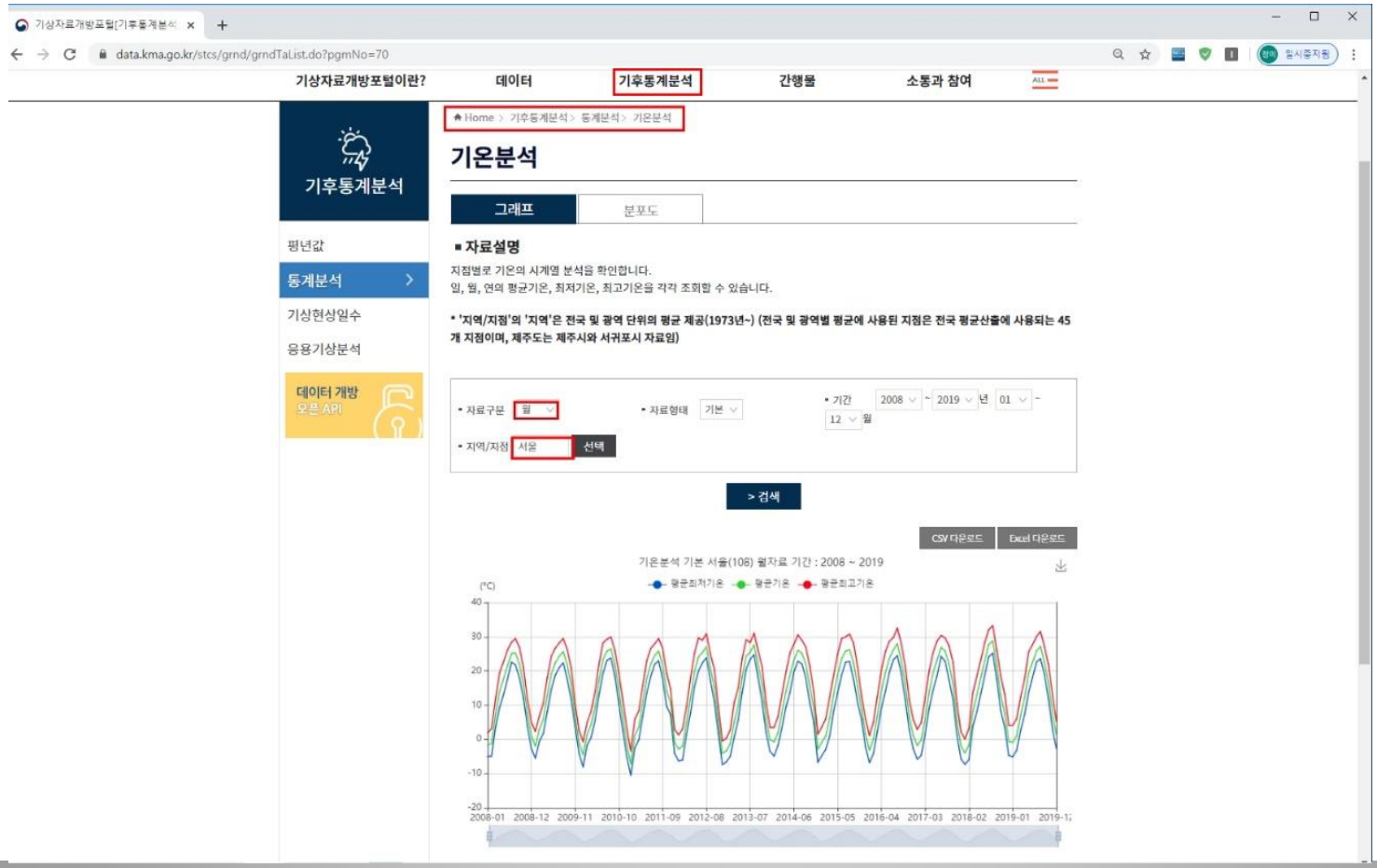
기상청 - 공공데이터 살펴보기

- 기상청은 기상자료개방포털 홈페이지를 통하여 기상관련 데이터를 무료료 제공합니다.
- http://data.kma.go.kr
- 상단메뉴에서 기후통계분석 메뉴 선택합니다.



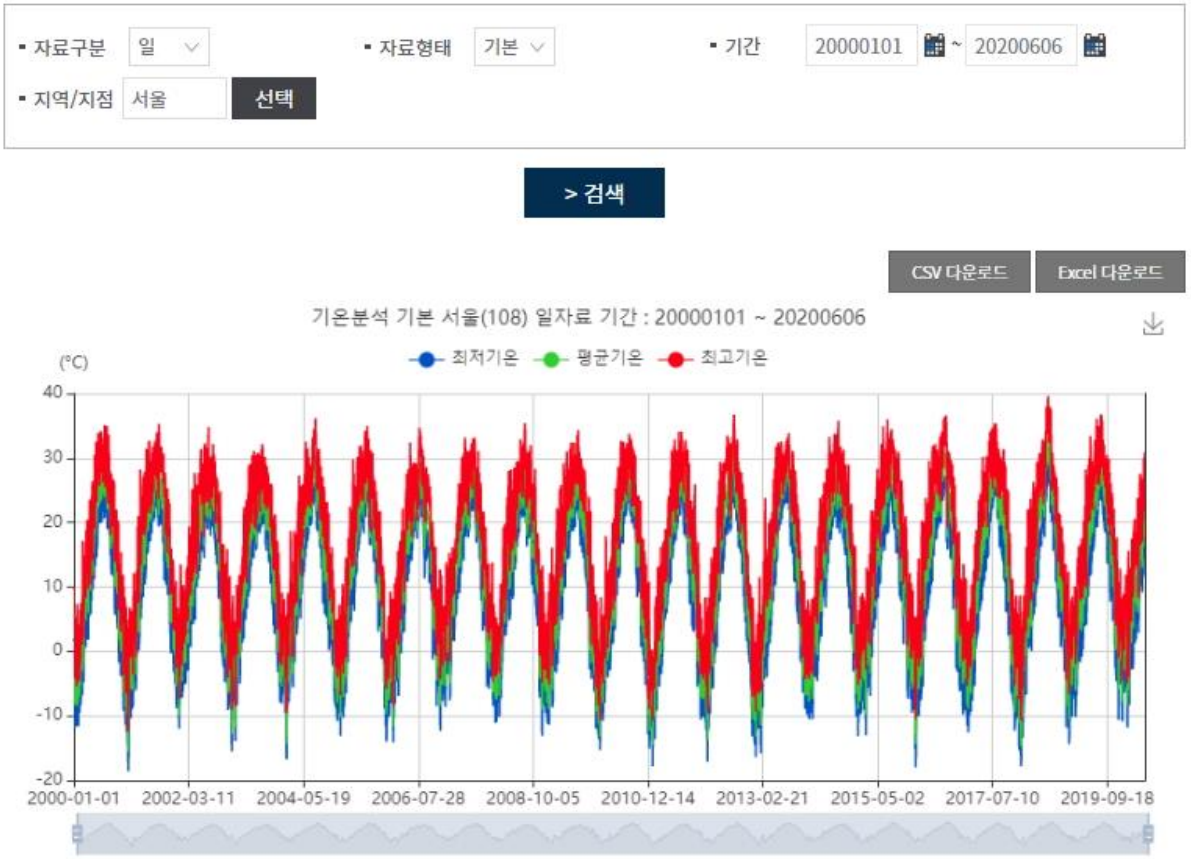
기상청 - 공공데이터 살펴보기

- 상단메뉴에서 기후통계분석 >통계분석>기온 분석 메뉴 선택합니다.
- 상세 분석 조건을 입력하고, 검색 버튼을 누릅니다



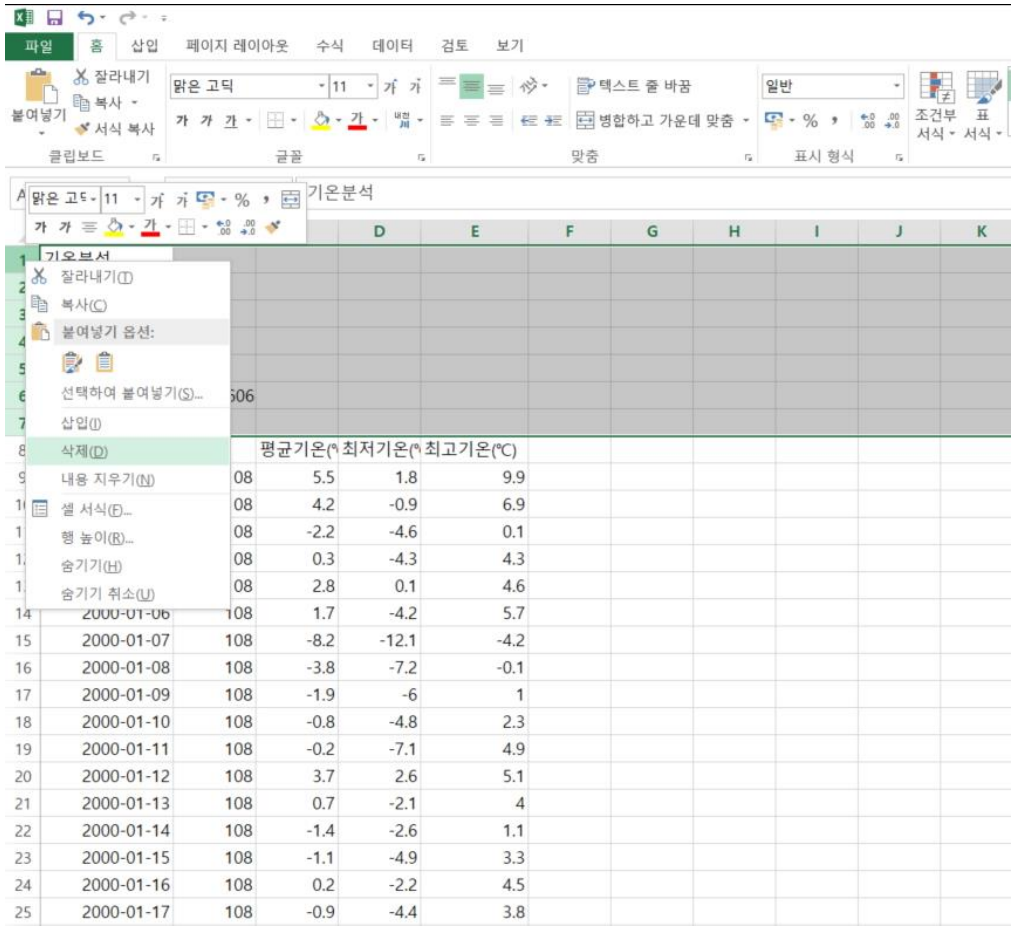
기상청 - 공공데이터 살펴보기

- [CSV다운로드] 버튼 클릭 또는 [Excel 다운로드] 버튼 클릭
- ‘ Comma - Separated Values ’의 약자
- csv 파일을 원하는 폴더에 저장(여기에서는 다운로드 폴더)



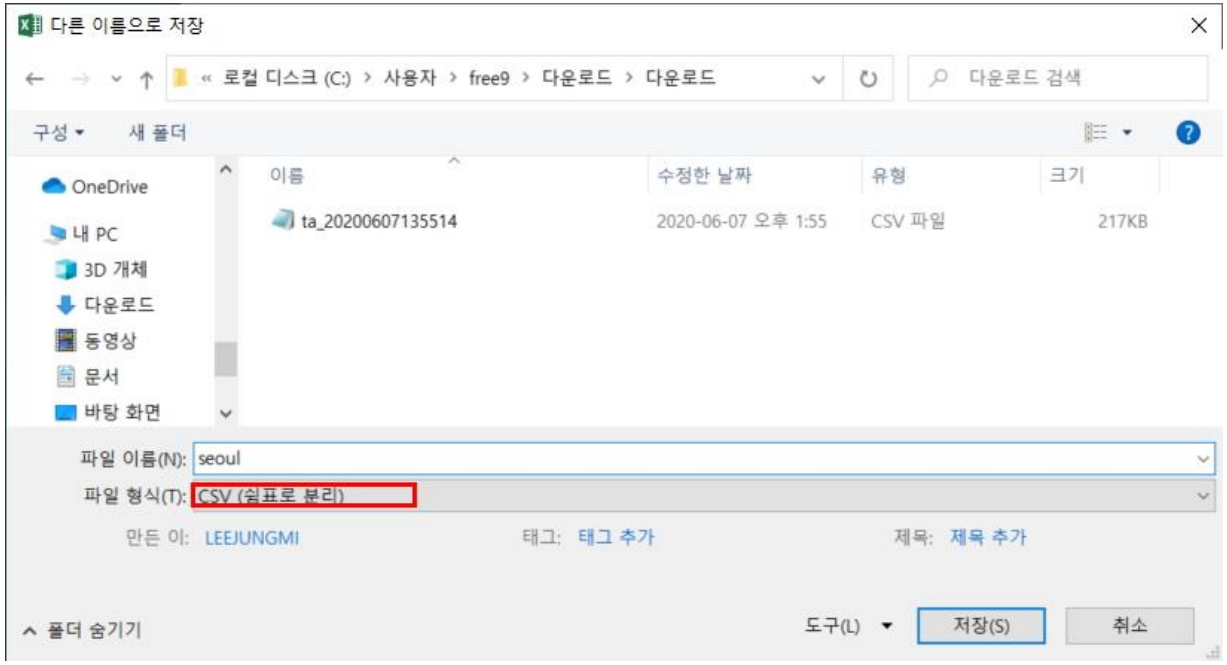
기상청 - 공공데이터 살펴보기

- [CSV다운로드] 버튼 클릭 또는 [Excel 다운로드] 버튼 클릭
- 파일을 원하는 폴더에 저장(여기에서는 다운로드 폴더)
- 불필요한 데이터 제거
- 1~7 행 삭제



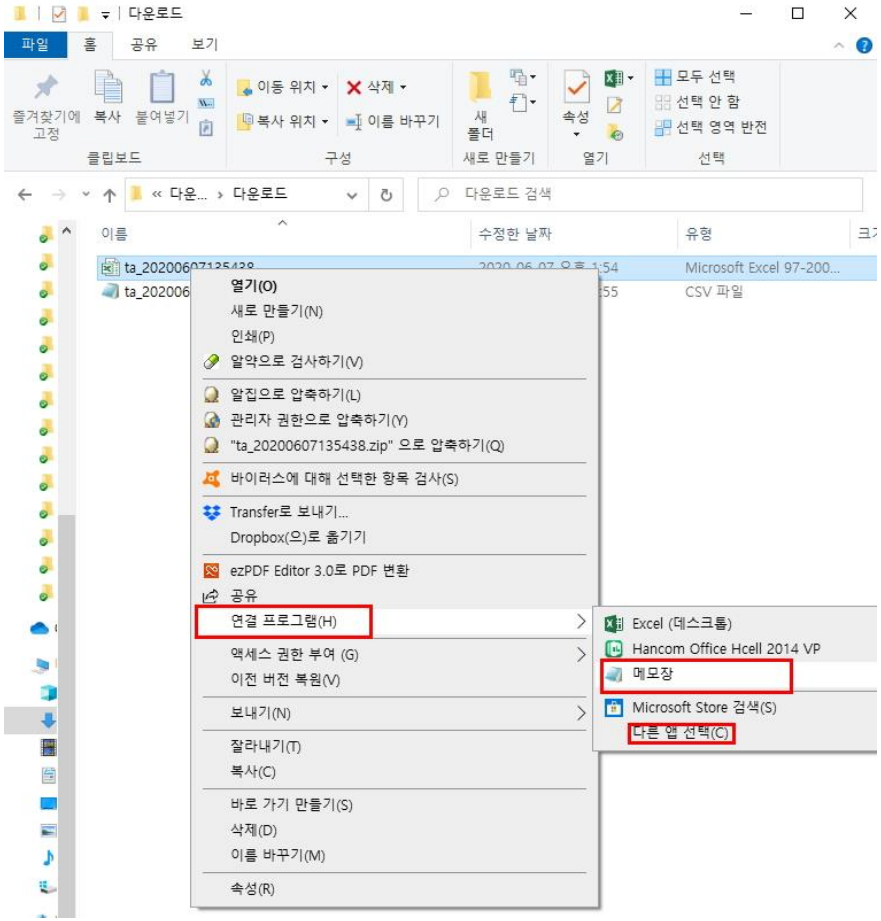
csv 파일 저장하기

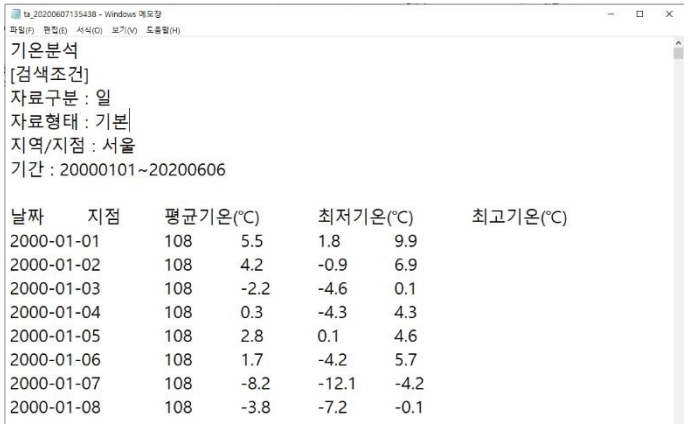
- csv 파일을 원하는 폴더에 저장(여기에서는 다운로드 폴더)
- 파일>다른이름으로 저장하기>파일형식
- CSV(쉼표로 분리) 선택
- 파일명 seoul 저장하기



csv 파일 열어보기

- [CSV다운로드] 후 csv파일 메모장으로 열어보기
- 메모장 프로그램이 보이지 않을시 다른 앱 선택합니다.





04

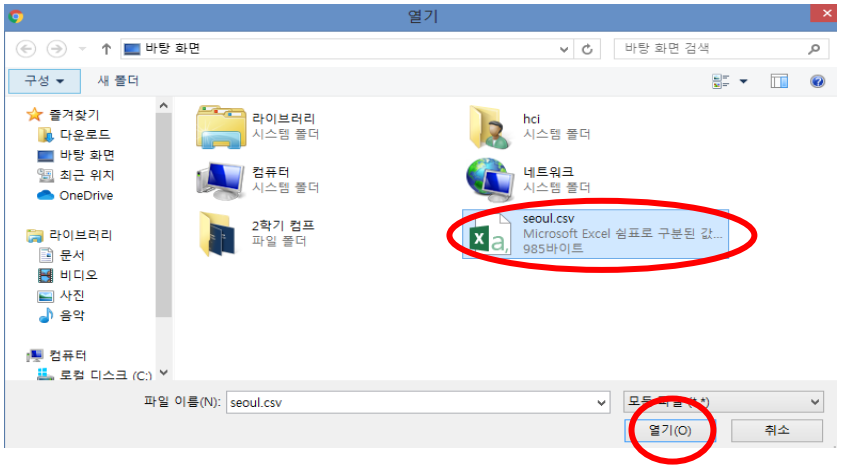
Pandas csv 파일 처리: csv 파일 업로드

- Colab 새파일을 열고, 셀에 다음 코드 작성 후, 실행버튼 클릭
- 실행이 되면서 파일 업로드를 위한 "파일 선택 " 버튼 클릭

```
from google.colab import files
uploaded = files.upload()
```

파일 선택 선택된 파일 없음 Cancel upload

- 파일 선택 창이 열리면, 미리 준비해 놓은 csv 파일을 열기



업로드 완료

```
from google.colab import files
uploaded = files.upload()
```

파일 선택 seoul.csv

- **seoul.csv**(application/vnd.ms-excel) - 985 bytes, last modified: 2020. 8. 25. - 100% done

Saving seoul.csv to seoul.csv

05

Pandas csv 파일 처리: csv 파일 읽기

```
# pandas, matplotlib import하기
import pandas as pd
import matplotlib.pyplot as plt
```

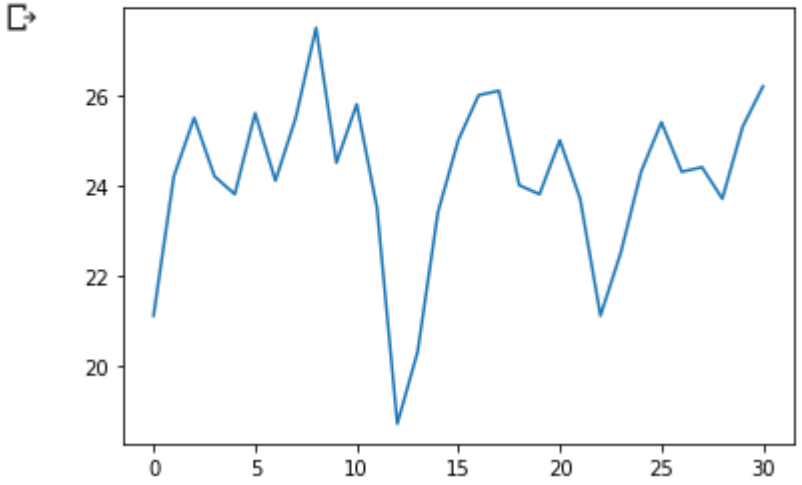
```
# csv 파일 읽어오기, 한글 데이터가 있으므로 encoding="cp949" 추가
df = pd.read_csv("seoul.csv", encoding="cp949")

print(df)
```

```
           날짜   지점  평균기온  최저기온  최고기온
0   2020-07-01  108  21.1  18.1  24.3
1   2020-07-02  108  24.2  20.4  29.8
2   2020-07-03  108  25.5  21.6  30.6
3   2020-07-04  108  24.2  20.4  29.5
4   2020-07-05  108  23.8  19.9  27.6
5   2020-07-06  108  25.6  21.8  30.9
6   2020-07-07  108  24.1  22.1  28.1
7   2020-07-08  108  25.5  21.3  30.4
8   2020-07-09  108  27.5  21.8  32.9
9   2020-07-10  108  24.5  22.7  27.7
10  2020-07-11  108  25.8  22.2  30.2
11  2020-07-12  108  23.5  21.7  26.0
12  2020-07-13  108  18.7  17.7  21.7
13  2020-07-14  108  20.3  17.2  24.3
14  2020-07-15  108  23.4  18.9  28.7
15  2020-07-16  108  25.0  20.4  29.7
16  2020-07-17  108  26.0  21.5  31.6
17  2020-07-18  108  26.1  22.8  31.0
18  2020-07-19  108  24.0  20.6  26.6
19  2020-07-20  108  23.8  22.4  26.6
20  2020-07-21  108  25.0  21.4  30.5
21  2020-07-22  108  23.7  22.6  25.7
22  2020-07-23  108  21.1  18.5  22.6
23  2020-07-24  108  22.5  18.7  26.8
24  2020-07-25  108  24.3  22.3  29.0
25  2020-07-26  108  25.4  21.6  29.6
26  2020-07-27  108  24.3  22.9  25.5
27  2020-07-28  108  24.4  22.7  26.4
28  2020-07-29  108  23.7  22.6  24.9
29  2020-07-30  108  25.3  23.4  28.4
30  2020-07-31  108  26.2  22.8  29.6
```

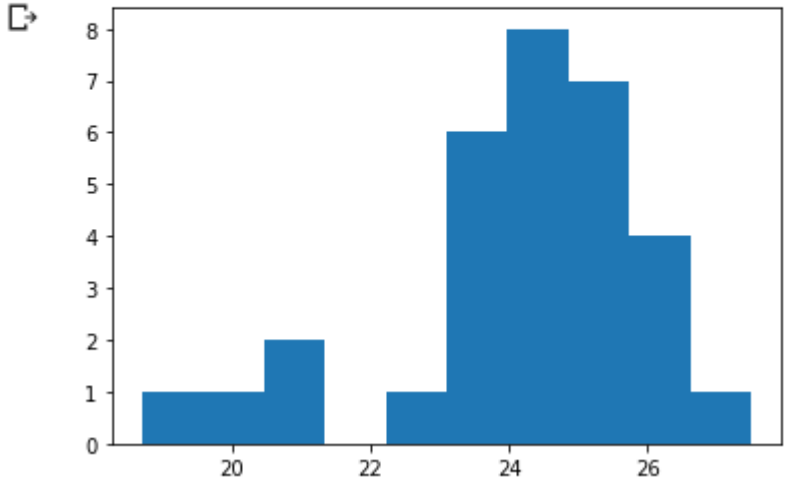
그래프 그리기: matplotlib 이용

- **matplotlib.pyplot과 열 추출 방법 이용**

```
# plot 그래프 그리기, 열추출방법 df["열이름"]
plt.plot(df["평균기온"])
plt.show()
```




```
# 히스토그램 그리기
plt.hist(df["평균기온"], bins=10)
plt.show()
```



통계함수

- **특정 열에 대한 통계값을 구할 때, 열 추출하고, 그 뒤에 통계함수를 메서드 형식으로 써줌**
  (예) df.["평균기온"].sum( ), df["평균기온"].mean( )

```
# 최저기온의 합계 구하기

low_sum = df["최저기온"].sum()
print(low_sum)
```

```
655.0
```

```
# 최저기온의 평균 구하기

low_mean = df["최저기온"].mean()
print(low_mean)
```

```
21.129032258064512
```

```
# 최저기온의 표준편차 구하기

low_sd = df["최저기온"].std()
print(low_sd)
```

```
1.7094235964962183
```

: standard deviation)는 자료의 산포도를 나타내는 수치로, 분산의 양의 제곱근으로 정의된다. #'표준편차가 작다' 평균에 몰려 있을 때,

```
# 최저기온의 중간값 구하기

low_med = df["최저기온"].median()
print(low_med)
```

```
21.6
```

중위수는 어떤 주어진 값들을 크기의 순서대로 정렬했을 때 가장 중앙에 위치하는 값

```
# 최저기온의 최소값 구하기

low_mn = df["최저기온"].min()
print(low_mn)
```

```
17.2
```

```
# 최저기온의 최대값 구하기

low_mx = df["최저기온"].max()
print(low_mx)
```

```
23.4
```

08

정렬함수

- 특정 열의 값에 따른 정렬, sort 함수 사용.
- 오름차순이 기본, 내림차순의 경우 ascending=False 추가

(예) df.sort_values(by="평균기온"), df.sort_values(by="최저기온", ascending=False)

```
# 평균기온 오름차순 정렬
print(df.sort_values(by="평균기온"))
```

	날짜	지점	평균기온	최저기온	최고기온
12	2020-07-13	108	18.7	17.7	21.7
13	2020-07-14	108	20.3	17.2	24.3
0	2020-07-01	108	21.1	18.1	24.3
22	2020-07-23	108	21.1	18.5	22.6
23	2020-07-24	108	22.5	18.7	26.8
14	2020-07-15	108	23.4	18.9	28.7
11	2020-07-12	108	23.5	21.7	26.0
21	2020-07-22	108	23.7	22.6	25.7
28	2020-07-29	108	23.7	22.6	24.9
19	2020-07-20	108	23.8	22.4	26.6
4	2020-07-05	108	23.8	19.9	27.6
18	2020-07-19	108	24.0	20.6	26.6
6	2020-07-07	108	24.1	22.1	28.1
3	2020-07-04	108	24.2	20.4	29.5
1	2020-07-02	108	24.2	20.4	29.8
24	2020-07-25	108	24.3	22.3	29.0
26	2020-07-27	108	24.3	22.9	25.5
27	2020-07-28	108	24.4	22.7	26.4
9	2020-07-10	108	24.5	22.7	27.7
20	2020-07-21	108	25.0	21.4	30.5

```
# 평균기온 내림차순 정렬
print(df.sort_values(by="평균기온", ascending=False))
```

	날짜	지점	평균기온	최저기온	최고기온
8	2020-07-09	108	27.5	21.8	32.9
30	2020-07-31	108	26.2	22.8	29.6
17	2020-07-18	108	26.1	22.8	31.0
16	2020-07-17	108	26.0	21.5	31.6
10	2020-07-11	108	25.8	22.2	30.2
5	2020-07-06	108	25.6	21.8	30.9
2	2020-07-03	108	25.5	21.6	30.6
7	2020-07-08	108	25.5	21.3	30.4
25	2020-07-26	108	25.4	21.6	29.6
29	2020-07-30	108	25.3	23.4	28.4
20	2020-07-21	108	25.0	21.4	30.5
15	2020-07-16	108	25.0	20.4	29.7
9	2020-07-10	108	24.5	22.7	27.7
27	2020-07-28	108	24.4	22.7	26.4
24	2020-07-25	108	24.3	22.3	29.0
26	2020-07-27	108	24.3	22.9	25.5
1	2020-07-02	108	24.2	20.4	29.8
3	2020-07-04	108	24.2	20.4	29.5
6	2020-07-07	108	24.1	22.1	28.1
19	2020-07-10	108	24.0	20.6	26.6

감사합니다.