

codingOn x posco

K-Digital Training 신재생에너지 활용 IoT 과정

Pandas

수업 내용

- Series
- DataFrame
- 공공 데이터
 - 서울시 공원 내 운동기구 설치 현황

Pandas?

- Python의 데이터 분석 및 조작을 위한 라이브러리
- 테이블 형식의 데이터를 다루는 데 최적화
- Series: 1차원 데이터 구조로, 배열과 비슷하며 인덱스가 포함됨
- DataFrame: 2차원 데이터 구조로, 행과 열로 구성된 테이블 형식

특징

- 데이터 처리 및 변환
 - 데이터 필터링, 선택, 정렬
 - 결측값 처리 (NaN 데이터 채우기, 제거 등)
- 유연한 데이터 입출력
 - CSV, Excel, SQL, JSON, HTML 등 다양한 파일 포맷 지원
 - 데이터를 읽고 저장하는 함수 제공
- 강력한 연산 및 분석 기능
 - 통계 계산 (평균, 분산, 중간값 등)
 - 그룹화(groupby) 및 집계

Series란?

- Series는 데이터가 순차적으로 나열된 1차원 배열의 형태
- 기본적으로 숫자형 인덱스(0, 1, 2, ...)가 제공되지만, 커스텀 인덱스를 설정할 수도 있음

List

인덱스	값
0	143
1	150
2	157
3	160

Series

인덱스	값
2018	143
2019	150
2020	157
2021	160

Series

- series 생성

```
import pandas as pd

data = [10, 20, 30, 40]
series = pd.Series(data, index=['a', 'b', 'c', 'd'])
print(series)
print(type(series))

a    10
b    20
c    30
d    40
dtype: int64
<class 'pandas.core.series.Series'>
```

커스텀 index로 생성

```
data = {'a': 10, 'b': 20, 'c': 30, 'd': 40}
series = pd.Series(data)
print(series)

a    10
b    20
c    30
d    40
dtype: int64
```

딕셔너리로 생성

Series 속성

```
list_data = ['2024-12-01', 3.14, 'ABC', 100, True]
sr = pd.Series(list_data, name="시리즈")
print(sr)
```

```
idx = sr.index
print(idx)
```

```
val = sr.values
print(val)
```

```
print(sr.shape)
```

```
0    2024-08-03
1         3.14
2         ABC
3         100
4         True
Name: 시리즈, dtype: object
RangeIndex(start=0, stop=5, step=1)
['2024-08-03' 3.14 'ABC' 100 True]
(5,)
```


실습 1

- 다음은 Kevin의 점수 데이터이다. 이를 Series로 생성하시요.
 - 다음 두 방법을 사용하여 각각 생성하시요.
 - 1. 점수를 list로 만들고 Series 생성
 - 2. 과목명과 점수를 dict로 만들고 Series 생성

	Kevin
국어	95
수학	80
영어	100
국사	85

```
국어    95
수학    80
영어   100
국사    85
Name: Kevin, dtype: int64
```

실습 2

- 다음과 같은 Series를 만드세요.

```
밀가루      4 cups  
우유        1 cup  
계란        2 large  
참치캔      1 can  
Name: Dinner, dtype: object
```

실습 3 (선택)

- KBO 데이터에서 승률 (7번째 열)을 Series로 만드세요.
 - index는 팀명으로 설정
 - 데이터는 실수형으로 변환하여 생성
- KBO 데이터에서 승수(4번째 열)을 Series로 만드세요.
 - index는 팀명으로 설정
 - 데이터는 ~~실수~~형으로 변환하여 생성
정수

Series 조작

- 값에 접근

```
# 튜플을 시리즈로 변환
tuple_data = ('민지', '여', False)
member = pd.Series(tuple_data, index=['이름', '성별', '결혼여부'])

print(member)
print("이름: ", member['이름']) # 이름 라벨을 가진 원소 선택
print("데이터\n", member[['성별', '결혼여부']])
```

```
이름      민지
성별      여
결혼여부  False
dtype: object
이름:  민지
데이터
  성별      여
결혼여부  False
dtype: object
```

Series 조작

- 값에 접근

```
data = [10, 20, 30, 40]
series = pd.Series(data, index=['a', 'b', 'c', 'd'])

print(series['a']) # 인덱스 'a'의 값
print(series[0])  # 첫 번째 데이터
print(series[series > 20]) # 값이 20보다 큰 데이터만 선택
series['b'] = 50 # 인덱스 'b'의 값을 변경
```

실습 4

- 실습 1의 점수 데이터에서 다음을 수행하시요.

- 수학 점수를 90점으로 수정

```
국어      95
수학      90
영어     100
국사      85
Name: Kevin, dtype: int64
```

- 국어, 영어 점수만 선택

```
국어      95
영어     100
Name: Kevin, dtype: int64
```

- 90점 이상인 과목만 선택

```
국어      95
수학      90
영어     100
Name: Kevin, dtype: int64
```

실습 5 (선택 문제)

- 실습 3에서 생성한 KBO 데이터 Series에서,
 - 승률이 0.6 이상인 팀의 데이터만 출력하시요.
 - 승률이 0.5 미만인 팀의 승률을 모두 0.5로 수정하시요.
 - 마지막 3팀의 승률을 0으로 수정하시요.

DataFrame이란?

- DataFrame은 Series들을 결합해 놓은 형태
- 즉, 같은 길이(원소의 개수가 동일한)의 1차원 배열 여러 개가 필요함

Series

영희

인덱스	값
2018	143
2019	150
2020	157
2021	160

+

Series

철수

인덱스	값
2018	165
2019	172
2020	175
2021	180

=

Data frame

인덱스	영희	철수
2018	143	165
2019	150	172
2020	157	175
2021	160	180

DataFrame

- DataFrame 생성

```
import pandas as pd

data = {
    'Name': ['홍길동', '임꺽정', '성춘향'],
    'Age': [25, 30, 35],
    'City': ['Seoul', 'Busan', 'Incheon']
}

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

	Name	Age	City
0	홍길동	25	Seoul
1	임꺽정	30	Busan
2	성춘향	35	Incheon

DataFrame

- DataFrame은 행과 열로 이루어진 2차원 배열
- **인덱스**와 **컬럼**을 기준으로 **표 형태**처럼 데이터를 저장(db 테이블과 유사)

Data frame

인덱스	영희	철수
2018	143	165
2019	150	172
2020	157	175
2021	160	180

↑ 인덱스 ↑ 컬럼 ↑



인덱스와 컬럼 2개를 기준으로
데이터가 형성됨!

DataFrame

Data frame

인덱스	영희	철수
2018	143	165
2019	150	172
2020	157	175
2021	160	180

인덱스 칼럼

```
index = ['2018', '2019', '2020', '2021']
```

```
Yeonghee = pd.Series([143, 150, 157, 160], index=index)
```

```
Cheolsu = pd.Series([165, 172, 175, 180], index=index)
```

```
growth = pd.DataFrame({
    '영희': Yeonghee,
    '철수': Cheolsu
})
```

```
growth
```

	영희	철수
2018	143	165
2019	150	172
2020	157	175
2021	160	180

실습 6

- 다음 두 사람의 점수 데이터를 각각 Series로 생성하시요.
- 생성한 Series를 가지고 DataFrame을 생성하시요.

	Kevin
국어	95
수학	80
영어	100
국사	85

```

국어    95
수학    80
영어   100
국사    85
Name: Kevin, dtype: int64

```

	Jane
국어	50
수학	100
영어	70
국사	75

```

국어    50
수학   100
영어    70
국사    75
Name: Jane, dtype: int64

```

```

      Kevin  Jane
국어      95    50
수학      80   100
영어     100    70
국사      85    75

```

실습 7

- 다음은 Jin의 점수이다.
- Kevin, Jane, Jin 3명의 점수로 DataFrame을 만드세요.

	Jin
국어	90
영어	95
국사	85
과학	100

	Kevin	Jane	Jin
과학	NaN	NaN	100.0
국사	85.0	75.0	85.0
국어	95.0	50.0	90.0
수학	80.0	100.0	NaN
영어	100.0	70.0	95.0

실습 8 (선택 문제)

- KBO 데이터에서 다음 열을 가지고 DataFrame을 생성하세요.
 - 승 - 4번째 열, 정수
 - 패 - 5번째 열, 정수
 - 무 - 6번째 열, 정수
 - 승률 - 7번째 열, 실수
 - 최근10경기 - 9번째 열, 문자열

DataFrame 조작

- 데이터 확인

```
# 데이터 확인
print(growth.head())      # 상위 데이터 확인(기본5개)
print(growth.tail())      # 하위 데이터 확인(기본5개)
print(growth.shape)       # 데이터 크기 (행, 열)
print(growth.info())      # 데이터프레임 구조 및 타입 요약
print(growth.columns)     # 열 이름 확인
print(growth.values)      # 행 값 확인
print(growth.index)       # 인덱스 확인
print(growth.dtypes)      # 데이터 타입 확인
print(growth['철수'])      # 열 선택
print(growth[['철수']])   # 열 선택(열이름도 포함)
```

행 이름

내용

실습 9

- 실습 7의 DataFrame에서 다음 속성을 출력하여 결과를 확인하세요.

- shape
- columns
- values
- index
- dtypes

	Kevin	Jane	Jin
과학	NaN	NaN	100.0
국사	85.0	75.0	85.0
국어	95.0	50.0	90.0
수학	80.0	100.0	NaN
영어	100.0	70.0	95.0

실습 10

- 실습 7의 DataFrame에서
 - 1. Jin의 점수만 선택하시요.
 - 2. Kevin과 Jane의 점수를 선택하시요.
 - 3. 위 두 객체의 type은 각각 무엇인가?

```
과학    100.0  
국사     85.0  
국어     90.0  
수학     NaN  
영어     95.0  
Name: Jin, dtype: float64
```

	Kevin	Jane
과학	NaN	NaN
국사	85.0	75.0
국어	95.0	50.0
수학	80.0	100.0
영어	100.0	70.0

DataFrame 필터링

- **loc: 라벨(Label) 기반 접근**

- 행/열 이름(label)을 사용하여 데이터를 선택
- 문자형 인덱스나 라벨을 사용하는 경우
- 슬라이싱 사용 시 끝 값이 포함
- `df.loc[row_labels, column_labels]`

- **iloc: 정수(Integer) 기반 접근**

- 정수 위치(index)를 사용하여 데이터를 선택
- 순서 기반 접근에 적합
- 슬라이싱 사용 시 끝 값이 포함되지 않음 (Python 기본 슬라이싱과 동일)
- `df.iloc[row_indices, column_indices]`

DataFrame 필터링

- 필터링

```
data = {
    'Name': ['홍길동', '임꺽정', '성춘향'],
    'Age': [25, 30, 35],
    'City': ['Seoul', 'Busan', 'Incheon']
}

df = pd.DataFrame(data, index=['a', 'b', 'c'])
print(df.loc['b'])
print(df.loc['b', 'Age'])
print(df.loc['a':'c', 'Name':'Age'])
print(df.loc[df['Age'] >= 30])
print(df.loc[:, 'Name']) # 열 'Name'의 모든 행 선택
print(df.loc['a', :]) # 행 'a'의 모든 열 선택

print(df.iloc[1])
print(df.iloc[1, 1])
print(df.iloc[0:2, 0:2])
print(df.iloc[[0, 2], [1, 2]]) # 0번, 2번 행, 1번, 2번 열 선택
print(df.iloc[:, 1]) # 1번 열의 모든 행 선택
print(df.iloc[0, :]) # 0번 행의 모든 열 선택
```

실습 11

- 실습 7의 DataFrame에서
 - 학생 모두의 국어 점수를 출력하시요.
 - Jane의 수학 점수를 출력하시요.
 - Jin이 90점 이상인 과목들로 필터링 하시요.
 - 마지막 2과목으로 필터링 하시요.

	Kevin	Jane	Jin
과학	NaN	NaN	100.0
국사	85.0	75.0	85.0
국어	95.0	50.0	90.0
수학	80.0	100.0	NaN
영어	100.0	70.0	95.0

```
Kevin    95.0
Jane     50.0
Jin      90.0
Name: 국어, dtype: float64
```

```
100.0
```

	Kevin	Jane	Jin
과학	NaN	NaN	100.0
국어	95.0	50.0	90.0
영어	100.0	70.0	95.0

	Kevin	Jane	Jin
수학	80.0	100.0	NaN
영어	100.0	70.0	95.0

실습 12 (선택 문제)

- KBO 데이터에서,
 - LG, KT, 삼성의 승률을 출력하시요.
 - 팀별 승, 패 수를 출력하시요.
 - 경기 수가 18 경기 이상인 조건으로 필터링 하시요.
 - 승보다 패가 많은 팀으로 필터링 하시요.

DataFrame

- 추가, 수정

```
# 1. DataFrame 생성
data = {
    'Name': ['홍길동', '임꺽정', '성춘향'],
    'Age': [25, 30, 35],
    'City': ['서울', '부산', '인천']
}
df = pd.DataFrame(data)
print("기본 DataFrame:\n", df)

# 2. 행 추가
new_row = {'Name': '이몽룡', 'Age': 40, 'City': '포항'}
df = pd.concat([df, pd.DataFrame([new_row])], ignore_index=True)
print("\n행 추가 후 DataFrame:\n", df)

# 3. 열 추가
df['직업'] = ['엔지니어', '의사', '디자이너', '개발자']
print("\n열 추가 후 DataFrame:\n", df)

# 4. 요소 수정 (특정 값 수정)
df.at[1, 'City'] = '천안' # 특정 위치 수정
df.loc[df['Name'] == '임꺽정', 'Age'] = 36 # 조건에 맞는 값 수정
print("\n요소 수정 후 DataFrame:\n", df)

# 5. 칼럼 이름 변경
df.rename(columns={'Name': '이름', 'Age': '나이'}, inplace=True)
print("\n칼럼 이름 변경 후 DataFrame:\n", df)
```

기타

sort_values : 정렬

drop : 칼럼 삭제

실습 13

- 실습 7의 DataFrame에 다음 Bob의 점수를 추가하시요.

```
과학      60
국사      90
국어      95
수학      85
영어     100
Name: Bob, dtype: int64
```

	Kevin	Jane	Jin	Bob
과학	NaN	NaN	100.0	60
국사	85.0	75.0	85.0	90
국어	95.0	50.0	90.0	95
수학	80.0	100.0	NaN	85
영어	100.0	70.0	95.0	100

- Kevin의 과학 점수를 65점으로 수정하시요.

	Kevin	Jane	Jin	Bob
과학	65.0	NaN	100.0	60
국사	85.0	75.0	85.0	90
국어	95.0	50.0	90.0	95
수학	80.0	100.0	NaN	85
영어	100.0	70.0	95.0	100

- 열 이름을 한글로 수정하시요.

	케빈	제인	진	밥
과학	65.0	NaN	100.0	60
국사	85.0	75.0	85.0	90
국어	95.0	50.0	90.0	95
수학	80.0	100.0	NaN	85
영어	100.0	70.0	95.0	100

결측값

- 데이터셋에서 값이 존재하지 않거나 누락된 경우를 의미
- NaN, None으로 표시
- isnull() : 각 값의 결측 여부
- dropna() : 결측값 제거
- fillna() : 결측값 채우기

```
import pandas as pd
data = {
    'Name': ['홍길동', '임꺽정', '성춘향'],
    'Age': [25, None, 35],
    'City': ['Seoul', 'Busan', None]
}
df = pd.DataFrame(data)
# 결측값 확인
print(df.isnull()) # 각 값의 결측 여부
print(df.isnull().sum()) # 열별 결측값 개수
print(df.info()) # 결측값 및 데이터 타입 요약 정보
# 결측값 있는 행 제거
df_dropped_rows = df.dropna()
# 결측값 있는 열 제거
df_dropped_columns = df.dropna(axis=1)
# 결측값을 0으로 채우기
df_filled = df.fillna(0)
```


실습 14

- 실습 7의 DataFrame에서
 - 학생별 결측값 개수를 출력하시요.
 - 과목별 결측값 개수를 출력하시요.
 - 결측 값이 있는 과목을 제거하시요.

```
케빈    0  
제인    1  
진      1  
밥      0  
dtype: int64
```

```
과학    1  
국사    0  
국어    0  
수학    1  
영어    0  
dtype: int64
```

	케빈	제인	진	밥
국사	85.0	75.0	85.0	90
국어	95.0	50.0	90.0	95
영어	100.0	70.0	95.0	100

메서드

- `isin()` : Series나 DataFrame에서 특정 값이 존재하는지 여부를 참/거짓으로 반환하는 메서드(필터링)
- `isin()`은 기본적으로 결측값을 무시

```
s = pd.Series(['홍길동', '임꺽정', '성춘향', '이몽룡'])
# '홍길동'과 '이몽룡'이 Series에 있는지 확인
result = s.isin(['홍길동', '이몽룡'])
print(result)

data = {
    'Name': ['홍길동', '임꺽정', '성춘향', '이몽룡'],
    'Age': [25, 30, 35, 40]
}
df = pd.DataFrame(data)
# '홍길동'과 '이몽룡'이 DataFrame에 있는지 확인
result = df.isin(['Alice', 'David'])
print(result)

# 결측값은 무시
s = pd.Series([1, 2, None])
print(s.isin([None, 2]))
```

메서드

- `value_counts()` : 데이터의 고유값과 해당 값의 빈도수를 계산

```
# value_count
import pandas as pd

# 예제 Series 생성
s = pd.Series(['apple', 'banana', 'apple', 'orange', 'banana', 'apple'])

# 고유값과 빈도수 계산
result = s.value_counts()
print(result)

df = pd.DataFrame({
    'Fruits': ['apple', 'banana', 'apple', 'orange', 'banana', 'apple'],
    'Quantity': [1, 2, 3, 1, 2, 3]
})

# 특정 열에 대해 value_counts 호출
result = df['Fruits'].value_counts()
print(result)

# 빈도를 비율(%)로 계산.
result = s.value_counts(normalize=True)
print(result)
```

`normalize=True`
빈도를 비율(%)로 계산

`sort=False`
결과를 정렬하지 않음

`ascending=True`
빈도수를 오름차순으로 정렬

`dropna=False`
결측값(NaN)도 빈도로 계산

메서드

- agg() : 원하는 통계지표 요약

```
#agg
import pandas as pd

s = pd.Series([1, 2, 3, 4, 5])
result = s.agg(['sum', 'mean', 'max'])
print(result)

df = pd.DataFrame({
    'A': [1, 2, 3],
    'B': [4, 5, 6]
})
result = df.agg(['sum', 'mean'])
print(result)

df = pd.DataFrame({
    'A': [1, 2, 3],
    'B': [4, 5, 6]
})
result = df.agg({'A': 'sum', 'B': 'mean'})
print(result)
```

연산

- Series 간의 산술 연산은 인덱스를 기준으로 수행되며 인덱스가 맞지 않으면 결측값이 반환 됨
- DataFrame은 행과 열 단위로 연산을 수행합니다. 기본적으로 동일한 열 이름(컬럼)과 인덱스를 기준으로 연산이 진행
- mean() : 평균
- std() : 표준편차
- var() : 분산
- describe() : 통계지표 요약

실습 15

- 실습 7의 DataFrame에서
 - 학생별 평균 점수를 구하시요.
 - 과목별 평균 점수를 구하시요.
 - 과목별 최대/최소/평균 점수를 구하시요.

```
케빈    85.00
제인    73.75
진      92.50
밥      86.00
dtype: float64
```

```
과학    75.000000
국사    83.750000
국어    82.500000
수학    88.333333
영어    91.250000
dtype: float64
```

	max	min	mean
과학	100.0	60.0	75.000000
국사	90.0	75.0	83.750000
국어	95.0	50.0	82.500000
수학	100.0	80.0	88.333333
영어	100.0	70.0	91.250000

그룹화

- `groupby(by=컬럼명).연산()`
 - `sum()`: 각 그룹의 합계를 계산
 - `mean()`: 각 그룹의 평균을 계산
 - `count()`: 각 그룹의 데이터 개수를 계산
 - `agg()`: 각 그룹에 대해 여러 함수를 동시에 적용
 - `size()`: 각 그룹의 크기를 반환
 - `first()`: 각 그룹에서 첫 번째 값을 반환
 - `last()`: 각 그룹에서 마지막 값을 반환

그룹화

- 예시코드

```
#그룹화
import pandas as pd

data = {
    'group': ['A', 'A', 'B', 'B', 'C'],
    'value': [10, 20, 30, 40, 50]
}
df = pd.DataFrame(data)

result = df.groupby('group')['value'].sum()
print(result)
result = df.groupby('group')['value'].agg(['sum', 'mean', 'max'])
print(result)

data = {
    'group': ['A', 'A', 'B', 'B', 'C'],
    'value1': [10, 20, 30, 40, 50],
    'value2': [5, 15, 25, 35, 45]
}
df = pd.DataFrame(data)

result = df.groupby('group').agg({
    'value1': 'sum',
    'value2': 'mean'
})
print(result)
```


실습 16

- 실습 7의 DataFrame에서
- 아래와 같이 '분류' 열을 추가하시요.

	케빈	제인	진	밥	분류
과학	65.0	NaN	100.0	60	이과
국사	85.0	75.0	85.0	90	문과
국어	95.0	50.0	90.0	95	문과
수학	80.0	100.0	NaN	85	이과
영어	100.0	70.0	95.0	100	공통

- 각 학생의 '분류' 별 평균 점수를 출력하시요.
- 각 학생의 '분류' 별 최대/최소 점수를 출력하시요.

	케빈	제인	진	밥
분류				
공통	100.0	70.0	95.0	100.0
문과	90.0	62.5	87.5	92.5
이과	72.5	100.0	100.0	72.5

	케빈		제인		진		밥	
	max	min	max	min	max	min	max	min
분류								
공통	100.0	100.0	70.0	70.0	95.0	95.0	100	100
문과	95.0	85.0	75.0	50.0	90.0	85.0	95	90
이과	80.0	65.0	100.0	100.0	100.0	100.0	85	60

파일불러오기

파일형식	입력함수	출력함수
CSV	read_csv()	to_csv()
Excel	read_excel()	to_excel()
JSON	read_json()	to_json()
SQL	read_sql()	to_sql()
HTML	read_html()	to_html()

데이터 프레임(DataFrame)

pd.to_csv(“파일경로”) : .csv 파일 쓰기

CSV(영어: comma-separated values)는 몇 가지 필드를 쉼표(,)로 구분한 텍스트 데이터 및 텍스트 파일이다. 확장자는 .csv이며 MIME 형식은 text/csv이다.

```
# csv 파일로 내보내기
import pandas as pd

index=['2018', '2019', '2020', '2021']

data = {
    '영희': [143, 150, 157, 160],
    '철수': [165, 172, 175, 180]
}

growth = pd.DataFrame(data, columns=['영희', '철수'], index=index)
growth.to_csv("./source/growth.csv")
```

Pandas

`pd.read_csv("파일경로")` : .csv 파일 읽기

`pd.read_csv("파일경로", index_col = 0)` -> `index_col=0`은 기본 인덱스 삭제

```
# csv 파일 읽어오기
import pandas as pd

growth = pd.read_csv("./source/growth.csv", index_col=0)
print(growth)
growth.shape
```

	영희	철수
2018	143	165
2019	150	172
2020	157	175
2021	160	180

(4, 2)

실습 17

- 실습 7의 DataFrame을 score.csv 파일로 저장하시요.
- 저장한 파일을 읽어 DataFrame을 만드세요.

실습 18. Pandas Exercise 1

1. <https://www.kaggle.com/learn/pandas>
2. 위 링크에서 Creating, Reading and Writing의 exercise 풀어보기
3. 모든 문제를 해결 후 아래 캡처 화면 댓글로 올리기!

Lessons

1

Creating, Reading and Writing

You can't work with data if you can't read it. Get started here.

Tutorial

Exercise



실습 19. Pandas Exercise 2

1. <https://www.kaggle.com/learn/pandas>
2. 위 링크에서 Indexing, Selecting & Assigning의 exercise 풀어보기
3. 모든 문제를 해결 후 캡처 화면 댓글로 올리기!

실습 20. 공공데이터 활용

- 서울 열린데이터 광장 홈페이지 접속하여 서울시 공원 내 운동기구 설치 현황 파일을 다운로드 하세요(csv파일)

데이터셋

Home > 공공데이터 > 공공데이터

Q

상세 검색

통합 검색

☐ 결과 내 재검색

공공데이터

문화/관광

활용사례 등록

URL 복사

목록 이동

서울시 공원 내 운동기구 설치 현황

서울시 각 공원녹지사업소에서 관리하고 있는 공원 내 운동기구 설치 현황입니다.(구분, 위치(구역명), 운동기구 기종명, 수량, 관리번호 등)

파일내려받기

* 파일에 이상이 있는 경우 '오류신고'를 통해 운영자에게 알려주세요. [오류신고](#)

NO	항목	파일명	용량 (MB)	수정일	내려받기
1	데이터	서울시 공원 내 운동기구 설치 현황_201231.csv	0.17	2023.12.22.	↓

실습 20. 공공데이터 활용

아래 내용을 분석하여 출력하세요

- 공원별 총 운동기구 설치 수
- 운동기구 종류별 설치 개수
- 관리기관별 총 운동기구 설치 수
- 특정 공원 데이터 필터링(예:남산공원(회현))
- 특정 운동기구 종류 데이터 필터링(예:스텝사이클)
- 운동기구 수량 기준 내림차순 정렬
- (선택) 각각 파일 저장

실습 21 (선택문제)

- 2025년 팀별 승률 데이터를 수집하여 DataFrame을 만드세요.
- 생성한 DataFrame을 to_csv() 메소드를 이용하여 kbo.csv 파일로 저장하세요.

복.습.철.저

수고하셨습니다