02. 외부데이터 불러오기

1. Pandas 모듈을 활용한 외부데이터 가져오기

○ 외부 파일 및 웹에서 읽어오기

- Pandas는 다양한 형태의 외부 파일을 읽어와서 데이터프레임으로 변환하는 함수를 제공함.
- Pandas에서 제공하는 입출력 도구와 관련된 파일은 다음과 같음.

File Format	Reader	Writer
CSV	read_csv	to_csv
JSON	read_json	to_json
HTML	read_html	to_html
Local clipboard	read_clipboard	to_clipboard
MS Excel	read_excel	to_excel
HDF5 Format	read_hdf	to_hdf
SQL	read_sql	to_sql

1) CSV 파일

- 데이터 값을 쉼표로 구분하고 있다는 의미로 CSV라고 부르는 텍스트 파일을 말함.
- 쉼표로 열을 구분하고 줄바꿈으로 행을 구분함
- CSV 파일을 불러오는 명령어는 다음과 같음.

```
pandas.read_csv("파일경로")
```

- CSV 파일의 경로를 확장자까지 포함하여 함수의 인자로 전달하면, 데이터프레임으로 변환함.
- 파일의 경로 입력 시, 폴더 구분을 /를 이용해서 입력해야 하는 것을 주의해야 함.
- 함수의 header 옵션을 통해 데이터프레임의 열 이름으로 사용할 행을 지정하는데, 기본값은 0이고, 열 이름이 없는 경우 None으로 지정할 수 있음.

	A	B	C	D	E
1	**name**	**algol**	**basic**	**c++**	
2	**Lee**	A	C	B+	
3	**Kim**	A+	B	C	
4	**Park**	B	B+	C+	
5					
6					

```
In [1]: import pandas as pd

In [2]: file_path = 'C:/Users/LeeKJ/Desktop/df_sample.csv'
   ...: df1 = pd.read_csv(file_path)

In [3]: print(df1)
   name algol basic c++
0   Lee     A     C  B+
1   Kim    A+     B   C
2  Park     B    B+  C+
```

	A	B	C	D	E
1	Lee	A	C	B+	
2	Kim	A+	B	C	
3	Park	B	B+	C+	
4					
5					

```
In [1]: import pandas as pd

In [2]: file_path1 = 'C:/Users/LeeKJ/Desktop/df_sample1.csv'
   ...: df2 = pd.read_csv(file_path1, header=None)

In [3]: print(df2)
      0   1   2   3
0   Lee   A   C  B+
1   Kim  A+   B   C
2  Park   B  B+  C+
```

- 함수의 index_col 옵션을 통해 데이터프레임의 행 인덱스로 사용할 열의 번호 또는 이름을 지정할 수 있음. 기본값은 None임.

	A	B	C	D	E
1	**name**	**algol**	**basic**	**c++**	
2	**Lee**	A	C	B+	
3	**Kim**	A+	B	C	
4	**Park**	B	B+	C+	
5					
6					

```
In [1]: import pandas as pd

In [2]: file_path = 'C:/Users/LeeKJ/Desktop/df_sample.csv'
   ...: df3 = pd.read_csv(file_path, index_col='name')

In [3]: print(df3)
     algol basic c++
name
Lee      A     C  B+
Kim     A+     B   C
Park     B    B+  C+
```

- 함수의 names 옵션을 통해 데이터프레임의 열 이름을 문자열 리스트를 통해 지정할 수 있음. 기본값은 None임.

	A	B	C	D	E
1	Lee	A	C	B+	
2	Kim	A+	B	C	
3	Park	B	B+	C+	
4					
5					

```
In [1]: import pandas as pd

In [2]: file_path1 = 'C:/Users/LeeKJ/Desktop/df_sample1.csv'
   ...: df4 = pd.read_csv(file_path1, header=None, names=['Name', 'Algo', 'Bas', 'C'])

In [3]: print(df4)
   Name Algo Bas   C
0   Lee    A   C  B+
1   Kim   A+   B   C
2  Park    B  B+  C+
```

- 그 외 read_csv 함수의 옵션은 다음과 같음.

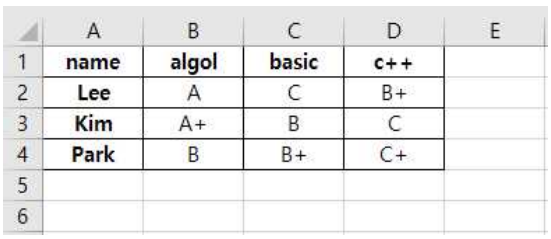
옵션	설명
path	파일의 위치, URL
sep(또는 delimiter)	텍스트 데이터를 필드별로 구분하는 문자
header	열 이름으로 사용될 행의 번호(기본값 : 0)
index_col	행 인덱스로 사용할 열의 번호 또는 열 이름
names	열 이름으로 사용할 문자열 리스트
skiprows	처음 몇 줄을 skip 할 거인지 설정(숫자입력) skip하려는 행의 번호를 담은 리스트도 설정 가능
skip_footer	마지막 몇 줄을 skip 할 것인지를 설정(숫자입력)
encoding	텍스트 인코딩 종류를 지정(예 : 'utf-8')

2) EXCEL 파일

- EXCEL 파일의 행과 열은 데이터프레임의 행과 열로 일대일 대응됨.
- EXCEL 파일을 불러오는 명령어는 다음과 같음.

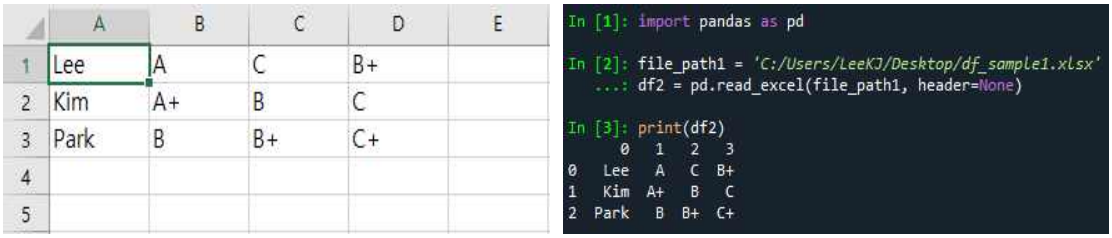
```
pandas.read_excel("파일경로")
```

- EXCEL 파일의 경로를 확장자까지 포함하여 함수의 인자로 전달함.
- read_excel 함수는 앞서 살펴본 read_csv 함수와 거의 비슷함. read_csv 함수의 옵션인 header, index_col 등 대부분의 옵션을 그대로 사용할 수 있음.
- 파일의 경로 입력 시, read_csv 함수와 마찬가지로 폴더 구분을 /를 이용해서 입력해야 하는 것을 주의해야 함.
- 함수의 header 옵션을 통해 데이터프레임의 열 이름으로 사용할 행을 지정하는데, 기본값은 0이고, 열 이름이 없는 경우 None으로 지정할 수 있음.



	A	B	C	D	E
1	name	algol	basic	c++	
2	Lee	A	C	B+	
3	Kim	A+	B	C	
4	Park	B	B+	C+	
5					
6					

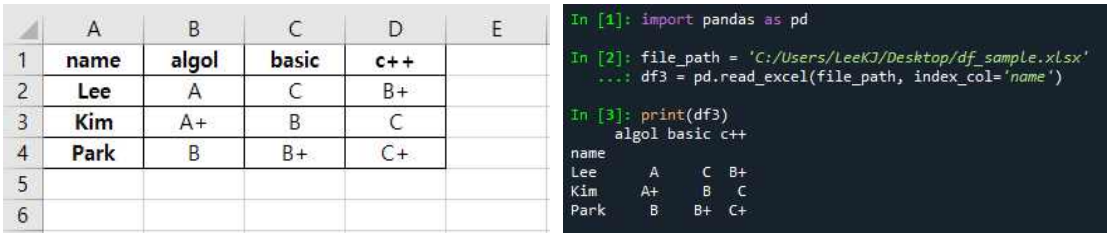
```
In [1]: import pandas as pd

In [2]: file_path = 'C:/Users/LeeKJ/Desktop/df_sample.xlsx'
   ...: df1 = pd.read_excel(file_path)

In [3]: print(df1)
   name algol basic c++
0   Lee     A     C  B+
1   Kim    A+     B   C
2  Park     B    B+  C+
```



	A	B	C	D	E
1	Lee	A	C	B+	
2	Kim	A+	B	C	
3	Park	B	B+	C+	
4					
5					

```
In [1]: import pandas as pd

In [2]: file_path1 = 'C:/Users/LeeKJ/Desktop/df_sample1.xlsx'
   ...: df2 = pd.read_excel(file_path1, header=None)

In [3]: print(df2)
      0   1   2   3
0   Lee   A   C  B+
1   Kim  A+   B   C
2  Park   B  B+  C+
```

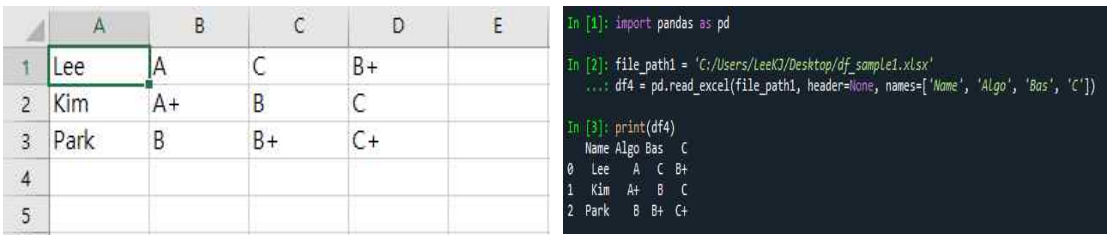
- 함수의 index_col 옵션을 통해 데이터프레임의 행 인덱스로 사용할 열의 번호 또는 이름을 지정할 수 있음. 기본값은 None임.



	A	B	C	D	E
1	**name**	**algol**	**basic**	**c++**	
2	**Lee**	A	C	B+	
3	**Kim**	A+	B	C	
4	**Park**	B	B+	C+	
5					
6					

```
In [1]: import pandas as pd

In [2]: file_path = 'C:/Users/LeeKJ/Desktop/df_sample.xlsx'
   ...: df3 = pd.read_excel(file_path, index_col='name')

In [3]: print(df3)
     algol basic c++
name
Lee      A     C  B+
Kim     A+     B   C
Park     B    B+  C+
```

- 함수의 names 옵션을 통해 데이터프레임의 열 이름을 문자열 리스트를 통해 지정할 수 있음. 기본값은 None임.



	A	B	C	D	E
1	Lee	A	C	B+	
2	Kim	A+	B	C	
3	Park	B	B+	C+	
4					
5					

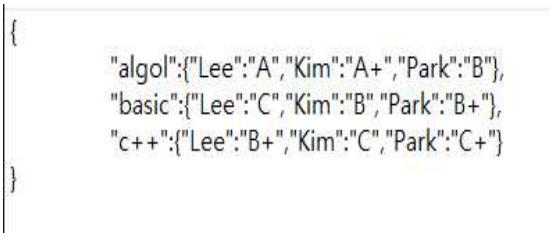
```
In [1]: import pandas as pd

In [2]: file_path1 = 'C:/Users/LeeKJ/Desktop/df_sample1.xlsx'
   ...: df4 = pd.read_excel(file_path1, header=None, names=['Name', 'Algo', 'Bas', 'C'])

In [3]: print(df4)
   Name Algo Bas   C
0   Lee    A   C  B+
1   Kim   A+   B   C
2  Park    B  B+  C+
```

3) JSON 파일

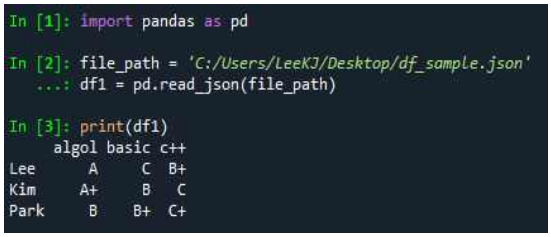
- JSON 파일은 데이터 공유를 목적으로 개발된 특수 파일 형식으로 파이썬의 딕셔너리와 비슷하게 키와 값의 구조를 갖고 있음.
- JSON 파일을 불러오는 명령어는 다음과 같음.

pandas.read_json("파일경로")

- 파일의 경로 입력 시, 폴더 구분을 /를 이용해서 입력해야 하는 것을 주의해야 함.
- JSON 파일의 키가 자동으로 열이름으로 지정됨.



```
{
        "algol":{"Lee":"A","Kim":"A+","Park":"B"},
        "basic":{"Lee":"C","Kim":"B","Park":"B+"},
        "c++":{"Lee":"B+","Kim":"C","Park":"C+"}
}
```



```
In [1]: import pandas as pd

In [2]: file_path = 'C:/Users/LeeKJ/Desktop/df_sample.json'
   ...: df1 = pd.read_json(file_path)

In [3]: print(df1)
      algol basic c++
Lee       A     C  B+
Kim      A+     B   C
Park      B    B+  C+
```

4) HTML 파일

- Pandas는 HTML 웹 페이지에 있는 <table> 태그에서 표 형식의 데이터를 모두 찾아서 데이터프레임으로 변환할 수 있음.
- 표 데이터들은 각각 별도의 데이터프레임으로 변환되기 때문에 여러 개의 데이터프레임을 원소로 갖는 리스트가 반환됨.
- HTML 파일을 불러오는 명령어는 다음과 같음.

pandas.read_html("웹주소")

- 파일의 경로 입력 시, 폴더 구분을 /를 이용해서 입력해야 하는 것을 주의해야 함.

file:///C:/Users/LeeKJ/Desktop/0500026

```
  c0 c1 c2 c3
0  0  1  4  7
1  1  2  5  8
2  2  3  6  9
        name year       developer opensource
0      NumPy 2006 Travis Oliphant       True
1 matplotlib 2003 John D. Hunter       True
2    pandas 2008 Wes Mckinneye       True
```

```
In [1]: import pandas as pd

In [2]: url ='C:/Users/LeeKJ/Desktop/df_sample.html'
   ...: tables = pd.read_html(url)

In [3]: print(tables)
[   Unnamed: 0  c0  c1  c2  c3
0           0   0   1   4   7
1           1   1   2   5   8
2           2   2   3   6   9,          name  year       developer  opensource
0       NumPy  2006  Travis Oliphant        True
1  matplotlib  2003   John D. Hunter        True
2      pandas  2008    Wes Mckinneye        True]
```

- HTML에 두 개의 표 형식의 데이터가 있으므로, 두 개의 표는 각각 별도의 데이터프레임으로 변환되어 데이터프레임을 원소로 갖는 리스트가 반환됨.
- 만약 두 개의 데이터프레임 중 하나만을 선택하려면 위치 인덱스를 사용하여 각각 출력할 수 있음.

```
In [4]: print(tables[0])
   Unnamed: 0  c0  c1  c2  c3
0           0   0   1   4   7
1           1   1   2   5   8
2           2   2   3   6   9
```

```
In [5]: print(tables[1])
        name  year        developer  opensource
0      NumPy  2006  Travis Oliphant        True
1  matplotlib  2003   John D. Hunter        True
2     pandas  2008    Wes Mckinneye        True
```

- set_index 함수를 사용하면 행 인덱스로 사용하고 싶은 변수를 지정할 수 있음.

```
In [6]: tables[1].set_index(['name'], inplace=True)

In [7]: print(tables[1])
            year        developer  opensource
name
NumPy       2006  Travis Oliphant        True
matplotlib  2003   John D. Hunter        True
pandas      2008    Wes Mckinneye        True
```

2. Pandas 모듈을 활용한 데이터프레임 내보내기

○ 외부파일로 데이터 저장하기

1) CSV 파일로 저장

- 데이터 값을 쉼표로 구분하고 있다는 의미로 CSV라고 부르는 텍스트 파일을 말함.
- 쉼표로 열을 구분하고 줄바꿈으로 행을 구분함
- 데이터프레임을 CSV 파일로 내보내는 명령어는 다음과 같음.

DataFrame객체.to_csv("파일경로(이름)")

```
In [1]: import pandas as pd

In [2]: data = {'name' : [ 'Lee', 'Kim', 'Park'],
   ...:         'algol' : [ "A", "A+", "B"],
   ...:         'basic' : [ "C", "B", "B+"],
   ...:         'c++' : [ "B+", "C", "C+"],
   ...:         }
   ...:
   ...: df = pd.DataFrame(data)

In [3]: df.set_index('name', inplace=True)

In [4]: print(df)
     algol basic c++
name
Lee      A     C  B+
Kim     A+     B   C
Park     B    B+  C+

In [5]: df.to_csv("C:/Users/LeeKJ/Desktop/df_sample11.csv")
```

```
In [4]: print(df)
      algol basic c++
name
Lee       A     C  B+
Kim      A+     B   C
Park      B    B+  C+
```

	A	B	C	D	E
1	name	algol	basic	c++	
2	Lee	A	C	B+	
3	Kim	A+	B	C	
4	Park	B	B+	C+	
5					
6					

2) JSON 파일로 저장

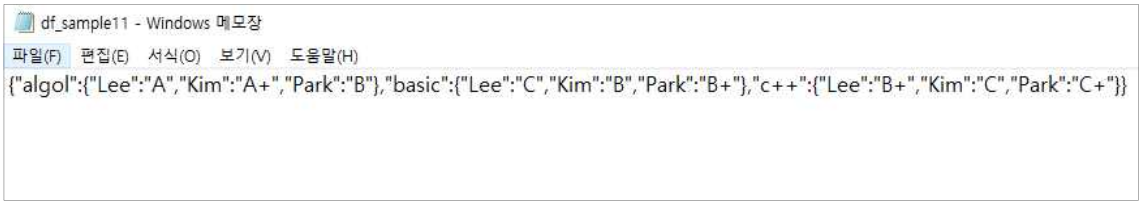
- JSON 파일은 데이터 공유를 목적으로 개발된 특수 파일 형식으로 파이썬의 딕셔너리와 비슷하게 키와 값의 구조를 갖고 있음.
- 데이터프레임을 JSON 파일로 내보내는 명령어는 다음과 같음.

DataFrame객체.to_json("파일경로(이름)")

- 데이터프레임의 열 이름이 키로 지정되고, 열 이름에 해당하는 값들은 행 인덱스와 값의 딕셔너리 형태로 지정됨.

```
In [1]: import pandas as pd

In [2]: data = {'name' : [ 'Lee', 'Kim', 'Park'],
   ...:         'algol' : [ "A", "A+", "B"],
   ...:         'basic' : [ "C", "B", "B+"],
   ...:         'c++' : [ "B+", "C", "C+"],
   ...:         }
   ...:
   ...: df = pd.DataFrame(data)

In [3]: df.set_index('name', inplace=True)

In [4]: print(df)
      algol basic c++
name
Lee       A     C  B+
Kim      A+     B   C
Park      B    B+  C+

In [5]: df.to_json("C:/Users/LeeKJ/Desktop/df_sample11.json")
```

```
In [4]: print(df)
      algol basic c++
name
Lee       A     C  B+
Kim      A+     B   C
Park      B    B+  C+
```



3) EXCEL 파일로 저장

- 데이터프레임은 EXCEL 파일과 아주 유사한 구조를 가짐.
- 데이터프레임의 행과 열은 EXCEL 파일의 행과 열로 일대일로 대응됨.
- 데이터프레임을 EXCEL 파일로 저장하는 명령어는 다음과 같음.

DataFrame객체.to_excel("파일경로(이름)")

- 파일의 경로 입력 시, 폴더 구분을 /를 이용해서 입력해야 하는 것을 주의해야 함.
- to_excel 함수를 사용하려면 openpyxl 라이브러리를 사전에 설치해야 하는데, 아나콘다를 사용하는 경우 openpyxl 라이브러리가 기본 제공되므로 따로 설치하지 않아도 됨.

```
In [1]: import pandas as pd

In [2]: data = {'name' : [ 'Lee', 'Kim', 'Park'],
   ...:         'algol' : [ "A", "A+", "B"],
   ...:         'basic' : [ "C", "B", "B+"],
   ...:         'c++' : [ "B+", "C", "C+"],
   ...:         }
   ...:
   ...: df = pd.DataFrame(data)

In [3]: df.set_index('name', inplace=True)

In [4]: print(df)
      algol basic c++
name
Lee       A     C  B+
Kim      A+     B   C
Park      B    B+  C+

In [5]: df.to_excel("C:/Users/LeeKJ/Desktop/df_sample11.xlsx")
```

```
In [4]: print(df)
      algol basic c++
name
Lee       A     C  B+
Kim      A+     B   C
Park      B    B+  C+
```

	A	B	C	D	E
1	name	algol	basic	c++	
2	Lee	A	C	B+	
3	Kim	A+	B	C	
4	Park	B	B+	C+	
5					
6					

4) 여러 개의 데이터프레임을 하나의 EXCEL 파일로 저장

- Pandas의 ExcelWriter 함수를 통해 EXCEL 워크북 객체를 생성함. 그런 후, to_excel 함수를 적용할 때 삽입하려는 워크북 객체를 인자로 전달함.
- to_excel 함수의 옵션인 sheet_name을 사용하여 EXCEL 파일의 시트 이름을 입력할 수 있음. 이 때, 시트 이름이 모두 같으면 가장 최근 명령어의 데이터프레임만 남게되고, 시트 이름이 모두 다르면 서로 다른 시트에 여러 데이터프레임을 구분하여 저장할 수 있음.
- 내보낼 데이터프레임에 대한 시트 이름을 모두 완료하였다면 최종적으로 writer.save()명령어를 통해 최종적으로 EXCEL 파일을 저장함.

```
객체이름 = pandas.ExcelWriter("파일경로(이름)")
내보낼데이터프레임.to_excel(객체이름,sheet_name="시트이름")
writer.save()
```

```
In [1]: import pandas as pd

In [2]: data1 = {'name' : [ 'Jerry', 'Riah', 'Paul'],
   ...:          'algol' : [ "A", "A+", "B"],
   ...:          'basic' : [ "C", "B", "B+"],
   ...:          'c++' : [ "B+", "C", "C+"],
   ...:         }

In [3]: df1 = pd.DataFrame(data1)
   ...: df1.set_index('name', inplace=True)

In [4]: print(df1)
      algol basic c++
name
Jerry     A     C  B+
Riah     A+     B   C
Paul      B    B+  C+

In [5]: data2 = {'c0':[1,2,3], 'c1':[4,5,6],
   ...:          'c2':[7,8,9], 'c3':[10,11,12],
   ...:          'c4':[13,14,15]}

In [6]: df2 = pd.DataFrame(data2)
   ...: df2.set_index('c0', inplace=True)

In [7]: print(df2)
    c1  c2  c3  c4
c0
1    4   7  10  13
2    5   8  11  14
3    6   9  12  15

In [8]: writer = pd.ExcelWriter("C:/Users/LeeKJ/Desktop/df_excel11.xlsx")
   ...: df1.to_excel(writer, sheet_name="data1")
   ...: df2.to_excel(writer, sheet_name="data2")
   ...: writer.save()
```

```
In [4]: print(df1)
      algol basic c++
name
Jerry     A     C  B+
Riah     A+     B   C
Paul      B    B+  C+
```

```
In [7]: print(df2)
    c1  c2  c3  c4
c0
1    4   7  10  13
2    5   8  11  14
3    6   9  12  15
```

	A	B	C	D	E
1	name	algol	basic	c++	
2	Jerry	A	C	B+	
3	Riah	A+	B	C	
4	Paul	B	B+	C+	
5					
6					

data1 data2

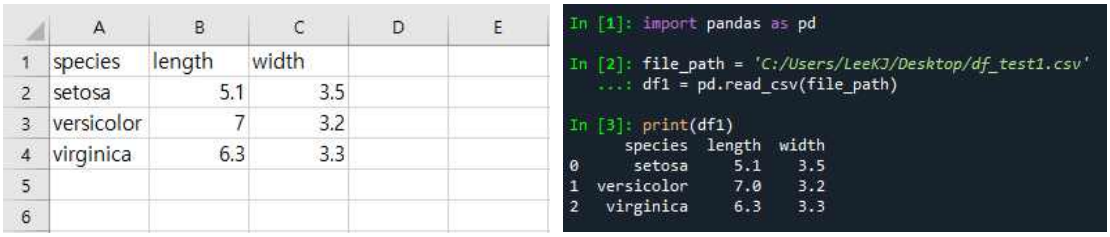
	A	B	C	D	E	F
1	c0	c1	c2	c3	c4	
2	1	4	7	10	13	
3	2	5	8	11	14	
4	3	6	9	12	15	
5						
6						

data1 data2

3. 외부데이터 가져오기 및 내보내기 실습

1) 주어진 자료 데이터프레임으로 가져오기

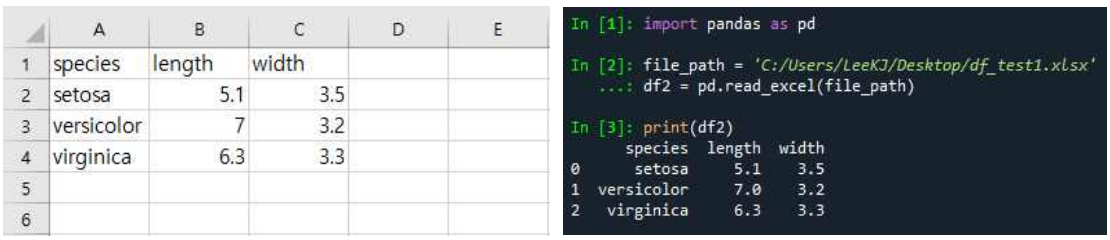
※ 다음의 CSV 자료 (df_test1.csv)를 데이터프레임으로 가져오세요.



	A	B	C	D	E
1	species	length	width		
2	setosa	5.1	3.5		
3	versicolor	7	3.2		
4	virginica	6.3	3.3		
5					
6					

```
In [1]: import pandas as pd

In [2]: file_path = 'C:/Users/LeeKJ/Desktop/df_test1.csv'
   ...: df1 = pd.read_csv(file_path)

In [3]: print(df1)
      species  length  width
0      setosa     5.1    3.5
1  versicolor     7.0    3.2
2   virginica     6.3    3.3
```

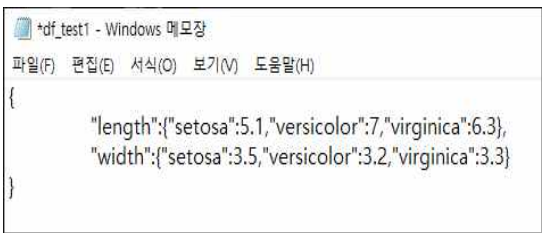
※ 다음의 EXCEL 자료 (df_test1.xlsx)를 데이터프레임으로 가져오세요.



	A	B	C	D	E
1	species	length	width		
2	setosa	5.1	3.5		
3	versicolor	7	3.2		
4	virginica	6.3	3.3		
5					
6					

```
In [1]: import pandas as pd

In [2]: file_path = 'C:/Users/LeeKJ/Desktop/df_test1.xlsx'
   ...: df2 = pd.read_excel(file_path)

In [3]: print(df2)
      species  length  width
0      setosa     5.1    3.5
1  versicolor     7.0    3.2
2   virginica     6.3    3.3
```

※ 다음의 JSON 자료 (df_test1.json)를 데이터프레임으로 가져오세요.



*df_test1 - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
{
        "length":{"setosa":5.1,"versicolor":7,"virginica":6.3},
        "width":{"setosa":3.5,"versicolor":3.2,"virginica":3.3}
}
```

```
In [1]: import pandas as pd

In [2]: file_path = 'C:/Users/LeeKJ/Desktop/df_test1.json'
   ...: df3 = pd.read_json(file_path)

In [3]: print(df3)
            length  width
setosa         5.1    3.5
versicolor     7.0    3.2
virginica      6.3    3.3
```

2) 주어진 자료 데이터프레임 생성 후 내보내기

※ 다음의 표를 데이터프레임으로 변환 후, Origin을 행 인덱스로 지정 후, df_test라는 이름으로 CSV, EXCEL 및 JSON 파일로 내보내시오.

Origin	Mpg	Weight
USA	18.0	3504
EU	15.0	3693
USA	18.0	3436

```
In [1]: import pandas as pd

In [2]: exam_data = {'Origin' : ['USA', 'EU', 'KOR'],
   ...:              'Mpg' : [18.0, 15.0, 18.0],
   ...:              'Weight' : [3504, 3693, 3436]}

In [3]: df = pd.DataFrame(exam_data)
   ...: df.set_index('Origin',inplace=True)

In [4]: print(df)
         Mpg  Weight
Origin
USA     18.0    3504
EU      15.0    3693
KOR     18.0    3436
```

```
In [5]: df.to_csv("C:/Users/LeeKJ/Desktop/df_test.csv")
   ...: df.to_excel("C:/Users/LeeKJ/Desktop/df_test.xlsx")
   ...: df.to_json("C:/Users/LeeKJ/Desktop/df_test.json")
```

	A	B	C	D
1	Origin	Mpg	Weight	
2	USA	18	3504	
3	EU	15	3693	
4	KOR	18	3436	
5				

	A	B	C	D
1	**Origin**	**Mpg**	**Weight**	
2	**USA**	18	3504	
3	**EU**	15	3693	
4	**KOR**	18	3436	
5				

df_test - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

{"Mpg":{"USA":18.0,"EU":15.0,"KOR":18.0},"Weight":{"USA":3504,"EU":3693,"KOR":3436}}

※ 다음의 표를 각각 데이터프레임으로 변환한 후, 시트이름을 sh1, sh2로 하여 df_test1이라는 이름의 EXCEL 파일로 내보내세요.

species	length	mass
Adelie	181	3750
Chinstrap	192	3500
Gentoo	211	4500

Embarked	age	pclass
S	22	3
C	38	1
Q	35	2

```
In [1]: import pandas as pd

In [2]: exam_data1 = {'species' : ['Adelie', 'Chinstrap', 'Gentoo'],
   ...:               'length' : [181, 192, 211],
   ...:               'mass' : [3750, 3500, 4500]}
   ...: df1 = pd.DataFrame(exam_data1)

In [3]: print(df1)
     species  length  mass
0     Adelie     181  3750
1  Chinstrap     192  3500
2     Gentoo     211  4500

In [4]: exam_data2 = {'Embarked' : ['S', 'C', 'Q'],
   ...:               'age' : [22, 38, 35],
   ...:               'pclass' : [3, 1, 2]}
   ...: df2 = pd.DataFrame(exam_data2)

In [5]: print(df2)
  Embarked  age  pclass
0        S   22       3
1        C   38       1
2        Q   35       2
```

```
In [6]: writer = pd.ExcelWriter("C:/Users/LeeKJ/Desktop/df_test11.xlsx")
   ...: df1.to_excel(writer, sheet_name="sh1")
   ...: df2.to_excel(writer, sheet_name="sh2")
   ...: writer.save()
```

	A	B	C	D	E
1		species	length	mass	
2	0	Adelie	181	3750	
3	1	Chinstrap	192	3500	
4	2	Gentoo	211	4500	
5					

sh1 sh2

	A	B	C	D	E
1		Embarked	age	pclass	
2	0	S	22	3	
3	1	C	38	1	
4	2	Q	35	2	
5					

sh1 sh2