pandas는 pd라는 이름으로 임포트하는 것

```
import pandas as pd
```

read_csv 함수를 사용하여 csv를 pandas의 DataFrame이라는 형식으로 읽어들일 수 있음

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

	학생번호	학년	악력	윗몸일으키기	점수	순위
0	1	1	40.2	34	15	4
1	2	1	34.2	14	7	10
2	3	1	28.8	27	11	7
3	4	2	39.0	27	14	5
4	5	2	50.9	32	17	2
5	6	2	36.5	20	9	9
6	7	3	36.6	31	13	6
7	8	3	49.2	37	18	1
8	9	3	26.0	28	10	8
9	10	3	47.4	32	16	3

추출

DataFrame이 10행으로 비교적 소규모이지만, 수백행・수천행의 DataFrame을 다루는 경우 DataFrame의 구조를 알기 위해서는 DataFrame의 head 메서드가 자주 사용
head 메서드는 DataFrame의 앞 부분의 행을 추출하는 메서드 여기에서는 3행을 추출

```
df.head(3)
```

	학생번호	학년	악력	윗몸일으키기	점수	순위
0	1	1	40.2	34	15	4
1	2	1	34.2	14	7	10
2	3	1	28.8	27	11	7

아무것도 지정하지 않으면 5행이 추출

```
df.head()
```

	학생번호	학년	악력	윗몸일으키기	점수	순위
0	1	1	40.2	34	15	4
1	2	1	34.2	14	7	10
2	3	1	28.8	27	11	7
3	4	2	39.0	27	14	5
4	5	2	50.9	32	17	2

마찬가지로 tail 메서드는 DataFrame의 뒷부분을 추출

```
df.tail()
```

	학생번호	학년	악력	윗몸일으키기	점수	순위
5	6	2	36.5	20	9	9
6	7	3	36.6	31	13	6
7	8	3	49.2	27	18	1

슬라이스로 추출할 수도 있음

```
df[2:5]
```

	학생번호	학년	악력	윗몸일으키기	점수	순위
2	3	1	28.8	27	11	7
3	4	2	39.0	27	14	5
4	5	2	50.9	32	17	2

열은 열 이름으로 추출할 수 있음

```
df['악력']
```

```
0    40.2
1    34.2
2    28.8
3    39.0
4    50.9
5    36.5
6    36.6
7    49.2
8    26.0
9    47.4
Name: 악력, dtype: float64
```

복수열을 동시에 지정할 수도 있음

```
df[['학년', '점수']]
```

	학년	점수
0	1	15
1	1	7
2	1	11
3	2	14
4	2	17
5	2	9
6	3	13
7	3	18
8	3	10
9	3	16

행과 열을 동시에 지정하여 추출하고 싶을 때에는 loc 메서드가 편리 예를 들어 첫 번째 행의 학년은 다음과 같이 추출할 수 있음

```
df.loc[1, '학년']
```

```
1
```

슬라이스에 의한 지정도 가능

```
df.loc[1:3, '악력']
```

```
1    34.2
2    28.8
3    39.0
Name: 악력, dtype: float64
```

조건에 일치하는 행을 추출하고 싶은 경우는 다음과 같이 작성 여기에서는 악력이 30보다 작은 행을 추출

```
df[df['악력'] < 30]
```

	학생번호	학년	악력	윗몸일으키기	점수	순위
2	3	1	28.8	27	11	7
8	9	3	26.0	28	10	8

데이터의 개요

데이터의 크기는 shape로 알 수 있음

```
df.shape
```

```
(10, 6)
```

열 이름은 columns임

```
df.columns
```

```
Index(['학생번호', '학년', '악력', '윗몸일으키기', '점수', '순위'], dtype='object')
```

```
df.index
```

```
RangeIndex(start=0, stop=10, step=1)
```

```
df.describe()
```

	학생번호	학년	악력	윗몸일으키기	점수	순위
count	10.00000	10.000000	10.000000	10.000000	10.000000	10.00000
mean	5.50000	2.100000	38.880000	28.200000	13.000000	5.50000
std	3.02765	0.875595	8.337306	6.828047	3.651484	3.02765
min	1.00000	1.000000	26.000000	14.000000	7.000000	1.00000
25%	3.25000	1.250000	34.775000	27.000000	10.250000	3.25000
50%	5.50000	2.000000	37.800000	29.500000	13.500000	5.50000
75%	7.75000	3.000000	45.600000	32.000000	15.750000	7.75000
max	10.00000	3.000000	50.900000	37.000000	18.000000	10.00000

데이터의 추가

df에 데이터를 추가 우선은 열의 추가

```
df['팔 굽혀 펴기'] = [15, 12, 10, 4, 20,
                  24, 9, 11, 11, 17]
df
```

	학생번호	학년	악력	윗몸일으키기	점수	순위	팔 굽혀 펴기
0	1	1	40.2	34	15	4	15
1	2	1	34.2	14	7	10	12

다음은 행을 추가

```
new_row = pd.Series([11, 3, 42.4, 30, 15, 11, 12], index=df.columns, name=10)
new_row
```

```
학생번호        11.0
학년           3.0
악력          42.4
윗몸일으키기      30.0
점수          15.0
순위          11.0
팔 굽혀 펴기     12.0
Name: 10, dtype: float64
```

```
df.append(new_row)
```

	학생번호	학년	악력	윗몸일으키기	점수	순위	팔 굽혀 펴기
0	1.0	1.0	40.2	34.0	15.0	4.0	15.0
1	2.0	1.0	34.2	14.0	7.0	10.0	12.0
2	3.0	1.0	28.8	27.0	11.0	7.0	10.0
3	4.0	2.0	39.0	27.0	14.0	5.0	4.0
4	5.0	2.0	50.9	32.0	17.0	2.0	20.0
5	6.0	2.0	36.5	20.0	9.0	9.0	24.0
6	7.0	3.0	36.6	31.0	13.0	6.0	9.0
7	8.0	3.0	49.2	37.0	18.0	1.0	11.0
8	9.0	3.0	26.0	28.0	10.0	8.0	11.0
9	10.0	3.0	47.4	32.0	16.0	3.0	17.0
10	11.0	3.0	42.4	30.0	15.0	11.0	12.0