1. 문제 정의

서울시의 구별 CCTV 현황 분석

단순히 어디에 CCTV가 많이 설치됐는지 ~ 구별 인구 대비 비율을 확인

pandas, Matplotlib 기초 학습

1.1 파이썬에서 텍스트 파일과 엑셀 파일 읽기 - pandas

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

cctv = pd.read_csv("/content/drive/MyDrive/스터디/데이터 주무르기/data/01_seoul_cctv.
display(cctv)
```

	※ 2021.7.31. 현황을 기준으로 각 연도별 설치된 CCTV 수량. 교체(저화질교체, 성능개선)는 최초 설치연도가 아닌 교체년도 수량에 포함	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Uni
0	구분	총계	2012년 이전	2012년	2013년	2014년	2015년	2016년	
1	계	77,032	7,667	2,200	3,491	4,439	6,582	8,129	
2	종로구	1,772	813	0	0	210	150	1	
3	중 구	2,333	16	114	87	77	236	240	
4	용산구	2,383	34	71	234	125	221	298	
5	성동구	3,602	448	125	212	105	339	310	
6	광진구	2,588	35	57	100	187	98	52	
7	동대문구	2,497	1,090	146	60	29	111	233	
8	중랑구	3,296	302	24	253	88	141	161	
9	성북구	3,958	83	78	170	230	323	594	
10	강북구	2,462	0	0	24	65	105	243	
11	도봉구	1,629	39	22	96	181	79	159	
12	노원구	2,415	0	97	193	77	516	331	
13	은평구	3,791	14	3	44	332	329	555	
14	서대문구	2,940	730	253	155	108	137	223	
15	마포구	2,421	58	93	77	61	162	454	

	※ 2021.7.31. 현황을 기준으로 각 연도별 설치된 CCTV 수량. 교체(저화질교체, 성능개선)는 최초 설치연도가 아닌 교체년도 수량에 포함	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Uni
16	양천구	3,312	1,413	167	181	143	180	350	
17	강서구	2,560	55	129	106	213	188	168	
18	구로구	4,075	852	216	349	187	268	326	
19	금천구	2,374	0	0	178	80	361	133	
20	영등포구	3,778	572	136	238	123	209	248	
21	동작구	2,297	41	24	25	503	128	253	
22	관악구	4,942	428	205	291	513	529	621	
23	서초구	3,704	116	75	94	71	563	516	
24	강남구	6,502	124	77	75	597	840	1310	
25	송파구	2,854	72	61	86	85	215	148	
26	강동구	2,547	332	27	163	49	154	202	

header 입력 안하니까 오류가 난다. header 입력을 통해 1번째 행부터 가져오기

In [2]:
```
cctv = pd.read_csv("/content/drive/MyDrive/스터디/데이터 주무르기/data/01_seoul_cctv.c
                   header=1, thousands = ",")
display(cctv)
```

	구분	총계	2012년 이전	2012년	2013년	2014년	2015년	2016년	2017년	2018년	2019년	2020년	2021년
0	계	77032	7667	2200	3491	4439	6582	8129	9947	9876	11961	11132	1608
1	종로구	1772	813	0	0	210	150	1	261	85	9	200	43
2	중 구	2333	16	114	87	77	236	240	372	386	155	361	289
3	용산구	2383	34	71	234	125	221	298	351	125	307	617	0
4	성동구	3602	448	125	212	105	339	310	874	390	262	461	76
5	광진구	2588	35	57	100	187	98	52	675	465	712	175	32
6	동대문구	2497	1090	146	60	29	111	233	136	197	209	223	63
7	중랑구	3296	302	24	253	88	141	161	162	173	1049	939	4

	구분	총계	2012년 이전	2012년	2013년	2014년	2015년	2016년	2017년	2018년	2019년	2020년	2021년
8	성북구	3958	83	78	170	230	323	594	460	867	714	251	188
9	강북구	2462	0	0	24	65	105	243	6	392	1000	588	39
10	도봉구	1629	39	22	96	181	79	159	134	222	198	168	331
11	노원구	2415	0	97	193	77	516	331	175	216	320	386	104
12	은평구	3791	14	3	44	332	329	555	403	635	1057	288	131
13	서대문구	2940	730	253	155	108	137	223	356	237	343	397	1
14	마포구	2421	58	93	77	61	162	454	359	343	494	300	20
15	양천구	3312	1413	167	181	143	180	350	139	140	274	325	0
16	강서구	2560	55	129	106	213	188	168	506	259	457	356	123
17	구로구	4075	852	216	349	187	268	326	540	488	434	415	0
18	금천구	2374	0	0	178	80	361	133	196	540	369	508	9
19	영등포구	3778	572	136	238	123	209	248	311	658	65	1213	5
20	동작구	2297	41	24	25	503	128	253	271	300	322	419	11
21	관악구	4942	428	205	291	513	529	621	687	663	640	331	34
22	서초구	3704	116	75	94	71	563	516	1060	428	358	420	3
23	강남구	6502	124	77	75	597	840	1310	999	748	789	942	1
24	송파구	2854	72	61	86	85	215	148	241	542	1068	235	101
25	강동구	2547	332	27	163	49	154	202	273	377	356	614	0

In [3]:
```
# 데이터 프레임 컬럼명 불러오기

cctv.columns
```

Out[3]:
```
Index(['구분', '총계', '2012년 이전', '2012년', '2013년', '2014년', '2015년', '2016년',
       '2017년', '2018년', '2019년', '2020년', '2021년'],
      dtype='object')
```

In [4]:
```
# 데이터 프레임의 컬럼명 변경
cctv.rename(columns = {cctv.columns[0] : '구별'},inplace = True)
```

```
cctv
```

Out[4]:

	구별	총계	2012년 이전	2012년	2013년	2014년	2015년	2016년	2017년	2018년	2019년	2020년	2021년
0	계	77032	7667	2200	3491	4439	6582	8129	9947	9876	11961	11132	1608
1	종로구	1772	813	0	0	210	150	1	261	85	9	200	43
2	중 구	2333	16	114	87	77	236	240	372	386	155	361	289
3	용산구	2383	34	71	234	125	221	298	351	125	307	617	0
4	성동구	3602	448	125	212	105	339	310	874	390	262	461	76
5	광진구	2588	35	57	100	187	98	52	675	465	712	175	32
6	동대문구	2497	1090	146	60	29	111	233	136	197	209	223	63
7	중랑구	3296	302	24	253	88	141	161	162	173	1049	939	4
8	성북구	3958	83	78	170	230	323	594	460	867	714	251	188
9	강북구	2462	0	0	24	65	105	243	6	392	1000	588	39
10	도봉구	1629	39	22	96	181	79	159	134	222	198	168	331
11	노원구	2415	0	97	193	77	516	331	175	216	320	386	104
12	은평구	3791	14	3	44	332	329	555	403	635	1057	288	131
13	서대문구	2940	730	253	155	108	137	223	356	237	343	397	1
14	마포구	2421	58	93	77	61	162	454	359	343	494	300	20
15	양천구	3312	1413	167	181	143	180	350	139	140	274	325	0
16	강서구	2560	55	129	106	213	188	168	506	259	457	356	123
17	구로구	4075	852	216	349	187	268	326	540	488	434	415	0
18	금천구	2374	0	0	178	80	361	133	196	540	369	508	9
19	영등포구	3778	572	136	238	123	209	248	311	658	65	1213	5
20	동작구	2297	41	24	25	503	128	253	271	300	322	419	11
21	관악구	4942	428	205	291	513	529	621	687	663	640	331	34

	구별	총계	2012년 이전	2012 년	2013 년	2014 년	2015 년	2016 년	2017 년	2018 년	2019 년	2020 년	2021 년
22	서초구	3704	116	75	94	71	563	516	1060	428	358	420	3
23	강남구	6502	124	77	75	597	840	1310	999	748	789	942	1
24	송파구	2854	72	61	86	85	215	148	241	542	1068	235	101
25	강동구	2547	332	27	163	49	154	202	273	377	356	614	0

In [5]:
```
# 첫 행 제거
cctv = cctv.drop(0,axis = 0)
cctv
```

Out[5]:

	구별	총계	2012년 이전	2012 년	2013 년	2014 년	2015 년	2016 년	2017 년	2018 년	2019 년	2020 년	2021 년
1	종로구	1772	813	0	0	210	150	1	261	85	9	200	43
2	중 구	2333	16	114	87	77	236	240	372	386	155	361	289
3	용산구	2383	34	71	234	125	221	298	351	125	307	617	0
4	성동구	3602	448	125	212	105	339	310	874	390	262	461	76
5	광진구	2588	35	57	100	187	98	52	675	465	712	175	32
6	동대문구	2497	1090	146	60	29	111	233	136	197	209	223	63
7	중랑구	3296	302	24	253	88	141	161	162	173	1049	939	4
8	성북구	3958	83	78	170	230	323	594	460	867	714	251	188
9	강북구	2462	0	0	24	65	105	243	6	392	1000	588	39
10	도봉구	1629	39	22	96	181	79	159	134	222	198	168	331
11	노원구	2415	0	97	193	77	516	331	175	216	320	386	104
12	은평구	3791	14	3	44	332	329	555	403	635	1057	288	131
13	서대문구	2940	730	253	155	108	137	223	356	237	343	397	1
14	마포구	2421	58	93	77	61	162	454	359	343	494	300	20
15	양천구	3312	1413	167	181	143	180	350	139	140	274	325	0
16	강서구	2560	55	129	106	213	188	168	506	259	457	356	123
17	구로구	4075	852	216	349	187	268	326	540	488	434	415	0
18	금천구	2374	0	0	178	80	361	133	196	540	369	508	9
19	영등포구	3778	572	136	238	123	209	248	311	658	65	1213	5
20	동작구	2297	41	24	25	503	128	253	271	300	322	419	11
21	관악구	4942	428	205	291	513	529	621	687	663	640	331	34
22	서초구	3704	116	75	94	71	563	516	1060	428	358	420	3

	구별	총계	2012년 이전	2012년	2013년	2014년	2015년	2016년	2017년	2018년	2019년	2020년	2021년
23	강남구	6502	124	77	75	597	840	1310	999	748	789	942	1
24	송파구	2854	72	61	86	85	215	148	241	542	1068	235	101
25	강동구	2547	332	27	163	49	154	202	273	377	356	614	0

In [6]:
```
## 판다스 버젼 오류
# !pip install pandas==1.2.0
```

In [7]:
```
# 엑셀 파일 불러오기
import pandas as pd

pop = pd.read_excel("/content/drive/MyDrive/스터디/데이터 주무르기/data/01_seoul_peop
pop
```

Out[7]:

	기간	자치구	세대	인구	인구.1	인구.2	인구.3	인구.4	인구.5	인구.6	인구.7	인
0	기간	자치구	세대	합계	합계	합계	한국인	한국인	한국인	등록외국인	등록외국인	등
1	기간	자치구	세대	계	남자	여자	계	남자	여자	계	남자	
2	2021	합계	4426007	9736027	4721977	5014050	9509458	4618040	4891418	226569	103937	12
3	2021	종로구	73494	153789	74186	79603	144683	70183	74500	9106	4003	
4	2021	중구	63519	131787	64083	67704	122499	59630	62869	9288	4453	
5	2021	용산구	111036	237285	115085	122200	222953	107210	115743	14332	7875	
6	2021	성동구	134233	292672	142259	150413	285990	139380	146610	6682	2879	
7	2021	광진구	167949	352627	169317	183310	339996	164058	175938	12631	5259	
8	2021	동대문구	168219	352006	172162	179844	337400	166646	170754	14606	5516	

	기간	자치구	세대	인구	인구.1	인구.2	인구.3	인구.4	인구.5	인구.6	인구.7	인
9	2021	중랑구	185712	391885	192903	198982	387350	191112	196238	4535	1791	
10	2021	성북구	195017	440142	211150	228992	430528	207578	222950	9614	3572	
11	2021	강북구	144536	302563	146984	155579	299182	145705	153477	3381	1279	
12	2021	도봉구	138656	319373	155221	164152	317366	154487	162879	2007	734	
13	2021	노원구	217378	514946	247813	267133	510956	246053	264903	3990	1760	
14	2021	은평구	214711	477173	228024	249149	473307	226458	246849	3866	1566	
15	2021	서대문구	143647	315659	149507	166152	304819	145694	159125	10840	3813	
16	2021	마포구	179861	378686	177233	201453	368905	173504	195401	9781	3729	
17	2021	양천구	181404	450487	220537	229950	447302	219214	228088	3185	1323	
18	2021	강서구	271175	579768	278720	301048	574315	276292	298023	5453	2428	
19	2021	구로구	180929	421163	208873	212290	396754	195537	201217	24409	13336	1
20	2021	금천구	117595	244891	124110	120781	230811	116527	114284	14080	7583	
21	2021	영등포구	187020	400908	198740	202168	376837	185926	190911	24071	12814	1
22	2021	동작구	184812	394364	190241	204123	385483	186426	199057	8881	3815	
23	2021	관악구	276597	499449	250058	249391	485699	243991	241708	13750	6067	

	기간	자치구	세대	인구	인구.1	인구.2	인구.3	인구.4	인구.5	인구.6	인구.7	(
24	2021	서초구	170244	416167	198914	217253	412279	197034	215245	3888	1880	
25	2021	강남구	234233	537800	257280	280520	533042	255047	277995	4758	2233	
26	2021	송파구	282417	663965	319895	344070	658338	317346	340992	5627	2549	
27	2021	강동구	201613	466472	228682	237790	462664	227002	235662	3808	1680	

In [8]:
```
## 불러오기 열, 행 조절
pop = pd.read_excel("/content/drive/MyDrive/스터디/데이터 주무르기/data/01_seoul_peop
                    usecols = 'B,D,G,J,N') ## usecols**
pop
```

Out[8]:

	자치구	계	계.1	계.2	65세이상고령자
0	합계	9736027	9509458	226569	1605416
1	종로구	153789	144683	9106	27818
2	중구	131787	122499	9288	24392
3	용산구	237285	222953	14332	39070
4	성동구	292672	285990	6682	46380
5	광진구	352627	339996	12631	51723
6	동대문구	352006	337400	14606	62211
7	중랑구	391885	387350	4535	71682
8	성북구	440142	430528	9614	74709
9	강북구	302563	299182	3381	64333
10	도봉구	319373	317366	2007	64160
11	노원구	514946	510956	3990	88345
12	은평구	477173	473307	3866	87241
13	서대문구	315659	304819	10840	54268
14	마포구	378686	368905	9781	54582
15	양천구	450487	447302	3185	68627
16	강서구	579768	574315	5453	92558
17	구로구	421163	396754	24409	72611
18	금천구	244891	230811	14080	41041
19	영등포구	400908	376837	24071	62516

	자치구	계	계.1	계.2	65세이상고령자
20	동작구	394364	385483	8881	66613
21	관악구	499449	485699	13750	79871
22	서초구	416167	412279	3888	60678
23	강남구	537800	533042	4758	78226
24	송파구	663965	658338	5627	97691
25	강동구	466472	462664	3808	74070

- usecols 활용하여 필요한 열만 사용

In [9]:
```
# 데이터 프레임의 컬럼명 변경
pop.rename(columns = {pop.columns[0] : '구별',
                      pop.columns[1] : '인구수',
                      pop.columns[2] : '한국인',
                      pop.columns[3] : '외국인',
                      pop.columns[4] : '고령자'},inplace = True)
pop
```

Out[9]:

	구별	인구수	한국인	외국인	고령자
0	합계	9736027	9509458	226569	1605416
1	종로구	153789	144683	9106	27818
2	중구	131787	122499	9288	24392
3	용산구	237285	222953	14332	39070
4	성동구	292672	285990	6682	46380
5	광진구	352627	339996	12631	51723
6	동대문구	352006	337400	14606	62211
7	중랑구	391885	387350	4535	71682
8	성북구	440142	430528	9614	74709
9	강북구	302563	299182	3381	64333
10	도봉구	319373	317366	2007	64160
11	노원구	514946	510956	3990	88345
12	은평구	477173	473307	3866	87241
13	서대문구	315659	304819	10840	54268
14	마포구	378686	368905	9781	54582
15	양천구	450487	447302	3185	68627
16	강서구	579768	574315	5453	92558
17	구로구	421163	396754	24409	72611
18	금천구	244891	230811	14080	41041
19	영등포구	400908	376837	24071	62516
20	동작구	394364	385483	8881	66613
21	관악구	499449	485699	13750	79871

	구별	인구수	한국인	외국인	고령자
22	서초구	416167	412279	3888	60678
23	강남구	537800	533042	4758	78226
24	송파구	663965	658338	5627	97691
25	강동구	466472	462664	3808	74070

In [10]:
```
# 필요없는 열 제거
pop = pop.drop(0,axis = 0)
pop
```

Out[10]:

	구별	인구수	한국인	외국인	고령자
1	종로구	153789	144683	9106	27818
2	중구	131787	122499	9288	24392
3	용산구	237285	222953	14332	39070
4	성동구	292672	285990	6682	46380
5	광진구	352627	339996	12631	51723
6	동대문구	352006	337400	14606	62211
7	중랑구	391885	387350	4535	71682
8	성북구	440142	430528	9614	74709
9	강북구	302563	299182	3381	64333
10	도봉구	319373	317366	2007	64160
11	노원구	514946	510956	3990	88345
12	은평구	477173	473307	3866	87241
13	서대문구	315659	304819	10840	54268
14	마포구	378686	368905	9781	54582
15	양천구	450487	447302	3185	68627
16	강서구	579768	574315	5453	92558
17	구로구	421163	396754	24409	72611
18	금천구	244891	230811	14080	41041
19	영등포구	400908	376837	24071	62516
20	동작구	394364	385483	8881	66613
21	관악구	499449	485699	13750	79871
22	서초구	416167	412279	3888	60678
23	강남구	537800	533042	4758	78226
24	송파구	663965	658338	5627	97691
25	강동구	466472	462664	3808	74070

1.2 판다스 기초 익히기

In [11]:
```
import numpy as np

# 날짜 데이터 생성
dates = pd.date_range('20130101', periods=6)
dates
```

Out[11]:
```
DatetimeIndex(['2013-01-01', '2013-01-02', '2013-01-03', '2013-01-04',
               '2013-01-05', '2013-01-06'],
              dtype='datetime64[ns]', freq='D')
```

In [12]:
```
# 데이터 프레임 생성
df = pd.DataFrame(np.random.randn(6,4), index=dates,
                  columns=['A','B','C','D'])
df
```

Out[12]:

	A	B	C	D
2013-01-01	0.462532	-0.300111	-1.047239	0.053844
2013-01-02	0.042378	2.375705	-1.332680	0.484368
2013-01-03	0.068205	-0.264785	1.723141	0.898625
2013-01-04	-0.881218	0.645013	0.896024	-0.268400
2013-01-05	-1.379214	-0.143299	1.628601	-0.841581
2013-01-06	-0.057692	-1.230375	0.525049	0.416577

In [13]:
```
# 인덱스 확인
df.index
```

Out[13]:
```
DatetimeIndex(['2013-01-01', '2013-01-02', '2013-01-03', '2013-01-04',
               '2013-01-05', '2013-01-06'],
              dtype='datetime64[ns]', freq='D')
```

In [14]:
```
# 열 이름 확인
df.columns
```

Out[14]:
```
Index(['A', 'B', 'C', 'D'], dtype='object')
```

In [15]:
```
# 데이터 값 확인
df.values
```

Out[15]:
```
array([[ 0.46253173, -0.30011091, -1.04723945,  0.05384386],
       [ 0.04237764,  2.37570546, -1.33267963,  0.48436846],
       [ 0.06820546, -0.26478536,  1.72314123,  0.89862534],
       [-0.88121819,  0.64501343,  0.89602415, -0.26839989],
       [-1.3792139 , -0.14329902,  1.62860068, -0.84158142],
       [-0.05769163, -1.23037528,  0.52504862,  0.41657692]])
```

In [16]:
```
# 데이터 정보 확인
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 6 entries, 2013-01-01 to 2013-01-06
Freq: D
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   A       6 non-null      float64
 1   B       6 non-null      float64
 2   C       6 non-null      float64
 3   D       6 non-null      float64
dtypes: float64(4)
memory usage: 240.0 bytes
```

In [17]:
```
# 데이터 통계량 확인
df.describe()
```

Out[17]:

	A	B	C	D
count	6.000000	6.000000	6.000000	6.000000
mean	-0.290835	0.180358	0.398816	0.123906
std	0.692095	1.229584	1.312898	0.617537
min	-1.379214	-1.230375	-1.332680	-0.841581
25%	-0.675337	-0.291280	-0.654167	-0.187839
50%	-0.007657	-0.204042	0.710536	0.235210
75%	0.061749	0.447935	1.445457	0.467421
max	0.462532	2.375705	1.723141	0.898625

In [18]:
```
# 데이터 정렬
df.sort_values(by='B', ascending=False)
```

Out[18]:

	A	B	C	D
2013-01-02	0.042378	2.375705	-1.332680	0.484368
2013-01-04	-0.881218	0.645013	0.896024	-0.268400
2013-01-05	-1.379214	-0.143299	1.628601	-0.841581
2013-01-03	0.068205	-0.264785	1.723141	0.898625
2013-01-01	0.462532	-0.300111	-1.047239	0.053844
2013-01-06	-0.057692	-1.230375	0.525049	0.416577

In [19]:
```
# 데이터 슬라이싱
display(df.loc['20130102':'20130104',['A','B']])

display(df.iloc[3:5,0:2])
```

	A	B
2013-01-02	0.042378	2.375705
2013-01-03	0.068205	-0.264785
2013-01-04	-0.881218	0.645013

	A	B
2013-01-04	-0.881218	0.645013
2013-01-05	-1.379214	-0.143299

In [20]:
```
# 데이터 프레임 조건문
display(df)

display(df[df.A > 0])
```

```
display(df[df > 0])
```

	A	B	C	D
2013-01-01	0.462532	-0.300111	-1.047239	0.053844
2013-01-02	0.042378	2.375705	-1.332680	0.484368
2013-01-03	0.068205	-0.264785	1.723141	0.898625
2013-01-04	-0.881218	0.645013	0.896024	-0.268400
2013-01-05	-1.379214	-0.143299	1.628601	-0.841581
2013-01-06	-0.057692	-1.230375	0.525049	0.416577

	A	B	C	D
2013-01-01	0.462532	-0.300111	-1.047239	0.053844
2013-01-02	0.042378	2.375705	-1.332680	0.484368
2013-01-03	0.068205	-0.264785	1.723141	0.898625

	A	B	C	D
2013-01-01	0.462532	NaN	NaN	0.053844
2013-01-02	0.042378	2.375705	NaN	0.484368
2013-01-03	0.068205	NaN	1.723141	0.898625
2013-01-04	NaN	0.645013	0.896024	NaN
2013-01-05	NaN	NaN	1.628601	NaN
2013-01-06	NaN	NaN	0.525049	0.416577

In [21]:
```
# 데이터 복사
df2 = df.copy()
```

In [22]:
```
# 열 추가
df2['E'] = ['one', 'one','two','three','four','three']
df2
```

Out[22]:

	A	B	C	D	E
2013-01-01	0.462532	-0.300111	-1.047239	0.053844	one
2013-01-02	0.042378	2.375705	-1.332680	0.484368	one
2013-01-03	0.068205	-0.264785	1.723141	0.898625	two
2013-01-04	-0.881218	0.645013	0.896024	-0.268400	three
2013-01-05	-1.379214	-0.143299	1.628601	-0.841581	four
2013-01-06	-0.057692	-1.230375	0.525049	0.416577	three

In [23]:
```
# 데이터 포함학 있는 행 추출
df2[df2['E'].isin(['two','four'])]
```

Out[23]:

	A	B	C	D	E

	A	B	C	D	E
2013-01-03	0.068205	-0.264785	1.723141	0.898625	two
2013-01-05	-1.379214	-0.143299	1.628601	-0.841581	four

In [24]:
```
# 열 합계 출력
display(df)

display(df.apply(np.cumsum))
```

	A	B	C	D
2013-01-01	0.462532	-0.300111	-1.047239	0.053844
2013-01-02	0.042378	2.375705	-1.332680	0.484368
2013-01-03	0.068205	-0.264785	1.723141	0.898625
2013-01-04	-0.881218	0.645013	0.896024	-0.268400
2013-01-05	-1.379214	-0.143299	1.628601	-0.841581
2013-01-06	-0.057692	-1.230375	0.525049	0.416577

	A	B	C	D
2013-01-01	0.462532	-0.300111	-1.047239	0.053844
2013-01-02	0.504909	2.075595	-2.379919	0.538212
2013-01-03	0.573115	1.810809	-0.656778	1.436838
2013-01-04	-0.308103	2.455823	0.239246	1.168438
2013-01-05	-1.687317	2.312524	1.867847	0.326856
2013-01-06	-1.745009	1.082148	2.392896	0.743433

In [25]:
```
# 열 최대 - 최소 차 출력
df.apply(lambda x: x.max() - x.min())
```

Out[25]:
```
A    1.841746
B    3.606081
C    3.055821
D    1.740207
dtype: float64
```

1.3 pandas 이용해서 CCTV 데이터 ㅍ파악하기

In [26]:
```
cctv.head()
```

Out[26]:

	구별	총계	2012년 이전	2012 년	2013 년	2014 년	2015 년	2016 년	2017 년	2018 년	2019 년	2020 년	2021 년
1	종로구	1772	813	0	0	210	150	1	261	85	9	200	43
2	중 구	2333	16	114	87	77	236	240	372	386	155	361	289
3	용산구	2383	34	71	234	125	221	298	351	125	307	617	0

	구별	총계	2012년 이전	2012년	2013년	2014년	2015년	2016년	2017년	2018년	2019년	2020년	2021년
4	성동구	3602	448	125	212	105	339	310	874	390	262	461	76
5	광진구	2588	35	57	100	187	98	52	675	465	712	175	32

In [27]:
```
# 오름차순으로 정렬
cctv.sort_values(by = "총계",ascending = True)
```

Out[27]:

	구별	총계	2012년 이전	2012년	2013년	2014년	2015년	2016년	2017년	2018년	2019년	2020년	2021년
10	도봉구	1629	39	22	96	181	79	159	134	222	198	168	331
1	종로구	1772	813	0	0	210	150	1	261	85	9	200	43
20	동작구	2297	41	24	25	503	128	253	271	300	322	419	11
2	중 구	2333	16	114	87	77	236	240	372	386	155	361	289
18	금천구	2374	0	0	178	80	361	133	196	540	369	508	9
3	용산구	2383	34	71	234	125	221	298	351	125	307	617	0
11	노원구	2415	0	97	193	77	516	331	175	216	320	386	104
14	마포구	2421	58	93	77	61	162	454	359	343	494	300	20
9	강북구	2462	0	0	24	65	105	243	6	392	1000	588	39
6	동대문구	2497	1090	146	60	29	111	233	136	197	209	223	63
25	강동구	2547	332	27	163	49	154	202	273	377	356	614	0
16	강서구	2560	55	129	106	213	188	168	506	259	457	356	123
5	광진구	2588	35	57	100	187	98	52	675	465	712	175	32
24	송파구	2854	72	61	86	85	215	148	241	542	1068	235	101
13	서대문구	2940	730	253	155	108	137	223	356	237	343	397	1
7	중랑구	3296	302	24	253	88	141	161	162	173	1049	939	4
15	양천구	3312	1413	167	181	143	180	350	139	140	274	325	0
4	성동구	3602	448	125	212	105	339	310	874	390	262	461	76
22	서초구	3704	116	75	94	71	563	516	1060	428	358	420	3
19	영등포구	3778	572	136	238	123	209	248	311	658	65	1213	5
12	은평구	3791	14	3	44	332	329	555	403	635	1057	288	131
8	성북구	3958	83	78	170	230	323	594	460	867	714	251	188
17	구로구	4075	852	216	349	187	268	326	540	488	434	415	0
21	관악구	4942	428	205	291	513	529	621	687	663	640	331	34
23	강남구	6502	124	77	75	597	840	1310	999	748	789	942	1

In [28]:
```
# 내림차순으로 정렬
cctv.sort_values(by = "총계",ascending = False).head(5)
```

Out[28]:

	구별	총계	2012년 이전	2012년	2013년	2014년	2015년	2016년	2017년	2018년	2019년	2020년	2021년
23	강남구	6502	124	77	75	597	840	1310	999	748	789	942	1
21	관악구	4942	428	205	291	513	529	621	687	663	640	331	34
17	구로구	4075	852	216	349	187	268	326	540	488	434	415	0
8	성북구	3958	83	78	170	230	323	594	460	867	714	251	188
12	은평구	3791	14	3	44	332	329	555	403	635	1057	288	131

In [29]:
```
cctv.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 25 entries, 1 to 25
Data columns (total 13 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   구별        25 non-null     object
 1   총계        25 non-null     int64 
 2   2012년 이전  25 non-null     int64 
 3   2012년     25 non-null     int64 
 4   2013년     25 non-null     int64 
 5   2014년     25 non-null     int64 
 6   2015년     25 non-null     int64 
 7   2016년     25 non-null     int64 
 8   2017년     25 non-null     int64 
 9   2018년     25 non-null     int64 
 10  2019년     25 non-null     int64 
 11  2020년     25 non-null     int64 
 12  2021년     25 non-null     int64 
dtypes: int64(12), object(1)
memory usage: 2.7+ KB
```

In [30]:
```
# 데이터 타입 변경
cctv.iloc[:,1:] = cctv.iloc[:,1:].astype("float")
```

In [31]:
```
cctv["최근증가율"] = (cctv["2021년"]+cctv["2020년"]+cctv["2019년"]+
                  cctv["2018년"]) / (cctv["2012년 이전"] + cctv["2012년"] + cctv["2013
                                     cctv["2014년"] +cctv["2015년"] +cctv["2016년"] +cc
cctv.sort_values(by = "최근증가율",ascending = False).head(5)
```

Out[31]:

	구별	총계	2012년 이전	2012년	2013년	2014년	2015년	2016년	2017년	2018년	2019년	2020년	2021년	최근증가율
9	강북구	2462.0	0.0	0.0	24.0	65.0	105.0	243.0	6.0	392.0	1000.0	588.0	39.0	4.557562
24	송파구	2854.0	72.0	61.0	86.0	85.0	215.0	148.0	241.0	542.0	1068.0	235.0	101.0	2.143172

	구별	총계	2012년 이전	2012년	2013년	2014년	2015년	2016년	2017년	2018년	2019년	2020년	2021년	최근증가율
7	중랑구	3296.0	302.0	24.0	253.0	88.0	141.0	161.0	162.0	173.0	1049.0	939.0	4.0	1.914235
18	금천구	2374.0	0.0	0.0	178.0	80.0	361.0	133.0	196.0	540.0	369.0	508.0	9.0	1.504219
10	도봉구	1629.0	39.0	22.0	96.0	181.0	79.0	159.0	134.0	222.0	198.0	168.0	331.0	1.294366

최근 4년간 CCTV가 그 이전 대비 많이 증가한 구는 다음과 같이 5개이다.

- 강북구
- 송파구
- 중랑구
- 금천구
- 도봉구

1.4 인구수 데이터 확인

In [32]:
```
pop.head()
```

Out[32]:

	구별	인구수	한국인	외국인	고령자
1	종로구	153789	144683	9106	27818
2	중구	131787	122499	9288	24392
3	용산구	237285	222953	14332	39070
4	성동구	292672	285990	6682	46380
5	광진구	352627	339996	12631	51723

In [33]:
```
# 구별 컬럼의 유니크 값 조사
pop["구별"].unique()
```

Out[33]:
```
array(['종로구', '중구', '용산구', '성동구', '광진구', '동대문구', '중랑구', '성북구',
'강북구',
       '도봉구', '노원구', '은평구', '서대문구', '마포구', '양천구', '강서구', '구로
구', '금천구',
       '영등포구', '동작구', '관악구', '서초구', '강남구', '송파구', '강동구'], dtype=
object)
```

In [34]:
```
# 외국인 비율과 고령자 비율 열 추가
pop['외국인비율'] = pop['외국인'] / pop['인구수']
pop['고령자비율'] = pop['고령자'] / pop['인구수']
pop.head()
```

Out[34]:

	구별	인구수	한국인	외국인	고령자	외국인비율	고령자비율
1	종로구	153789	144683	9106	27818	0.059211	0.180884
2	중구	131787	122499	9288	24392	0.070477	0.185087

	구별	인구수	한국인	외국인	고령자	외국인비율	고령자비율
3	용산구	237285	222953	14332	39070	0.060400	0.164654
4	성동구	292672	285990	6682	46380	0.022831	0.158471
5	광진구	352627	339996	12631	51723	0.035820	0.146679

In [35]:
```
# 인구수 기준 내림 차순
pop.sort_values(by='인구수', ascending=False).head(5)
```

Out[35]:

	구별	인구수	한국인	외국인	고령자	외국인비율	고령자비율
24	송파구	663965	658338	5627	97691	0.008475	0.147133
16	강서구	579768	574315	5453	92558	0.009405	0.159647
23	강남구	537800	533042	4758	78226	0.008847	0.145456
11	노원구	514946	510956	3990	88345	0.007748	0.171562
21	관악구	499449	485699	13750	79871	0.027530	0.159918

In [36]:
```
# 외국인 인구수 기준 내림 차순
pop.sort_values(by='외국인', ascending=False).head(5)
```

Out[36]:

	구별	인구수	한국인	외국인	고령자	외국인비율	고령자비율
17	구로구	421163	396754	24409	72611	0.057956	0.172406
19	영등포구	400908	376837	24071	62516	0.060041	0.155936
6	동대문구	352006	337400	14606	62211	0.041494	0.176733
3	용산구	237285	222953	14332	39070	0.060400	0.164654
18	금천구	244891	230811	14080	41041	0.057495	0.167589

In [37]:
```
# 외국인 비율 기준 내림 차순
pop.sort_values(by='외국인비율', ascending=False).head(5)
```

Out[37]:

	구별	인구수	한국인	외국인	고령자	외국인비율	고령자비율
2	중구	131787	122499	9288	24392	0.070477	0.185087
3	용산구	237285	222953	14332	39070	0.060400	0.164654
19	영등포구	400908	376837	24071	62516	0.060041	0.155936
1	종로구	153789	144683	9106	27818	0.059211	0.180884
17	구로구	421163	396754	24409	72611	0.057956	0.172406

In [38]:
```
# 고령자 인구수 기준 내림 차순
pop.sort_values(by='고령자', ascending=False).head(5)
```

Out[38]:

	구별	인구수	한국인	외국인	고령자	외국인비율	고령자비율
24	송파구	663965	658338	5627	97691	0.008475	0.147133
16	강서구	579768	574315	5453	92558	0.009405	0.159647

	구별	인구수	한국인	외국인	고령자	외국인비율	고령자비율
11	노원구	514946	510956	3990	88345	0.007748	0.171562
12	은평구	477173	473307	3866	87241	0.008102	0.182829
21	관악구	499449	485699	13750	79871	0.027530	0.159918

In [39]:
```
# 고령자 인구수 비율 기준 내림 차순
pop.sort_values(by='고령자비율', ascending=False).head(5)
```

Out[39]:

	구별	인구수	한국인	외국인	고령자	외국인비율	고령자비율
9	강북구	302563	299182	3381	64333	0.011175	0.212627
10	도봉구	319373	317366	2007	64160	0.006284	0.200894
2	중구	131787	122499	9288	24392	0.070477	0.185087
7	중랑구	391885	387350	4535	71682	0.011572	0.182916
12	은평구	477173	473307	3866	87241	0.008102	0.182829

1.5 데이터 합치기

- concat
- merge

예시

In [40]:
```
# 예시
df1 = pd.DataFrame({'A': ['A0', 'A1', 'A2', 'A3'],
                    'B': ['B0', 'B1', 'B2', 'B3'],
                    'C': ['C0', 'C1', 'C2', 'C3'],
                    'D': ['D0', 'D1', 'D2', 'D3']},
                   index=[0, 1, 2, 3])

df2 = pd.DataFrame({'A': ['A4', 'A5', 'A6', 'A7'],
                    'B': ['B4', 'B5', 'B6', 'B7'],
                    'C': ['C4', 'C5', 'C6', 'C7'],
                    'D': ['D4', 'D5', 'D6', 'D7']},
                   index=[4, 5, 6, 7])

df3 = pd.DataFrame({'A': ['A8', 'A9', 'A10', 'A11'],
                    'B': ['B8', 'B9', 'B10', 'B11'],
                    'C': ['C8', 'C9', 'C10', 'C11'],
                    'D': ['D8', 'D9', 'D10', 'D11']},
                   index=[8, 9, 10, 11])
```

In [41]:
```
result = pd.concat([df1, df2, df3], keys=['x', 'y', 'z'])
result
```

Out[41]:

		A	B	C	D
x	**0**	A0	B0	C0	D0
	1	A1	B1	C1	D1
	2	A2	B2	C2	D2
	3	A3	B3	C3	D3

		A	B	C	D
y	4	A4	B4	C4	D4
	5	A5	B5	C5	D5
	6	A6	B6	C6	D6
	7	A7	B7	C7	D7
z	8	A8	B8	C8	D8
	9	A9	B9	C9	D9
	10	A10	B10	C10	D10
	11	A11	B11	C11	D11

In [42]:
```
# 레벨 형성된거 확인할 수 있다.
result.index
```

Out[42]:
```
MultiIndex([('x',  0),
            ('x',  1),
            ('x',  2),
            ('x',  3),
            ('y',  4),
            ('y',  5),
            ('y',  6),
            ('y',  7),
            ('z',  8),
            ('z',  9),
            ('z', 10),
            ('z', 11)],
           )
```

In [43]:
```
result.index.get_level_values(0)
```

Out[43]:
```
Index(['x', 'x', 'x', 'x', 'y', 'y', 'y', 'y', 'z', 'z', 'z', 'z'], dtype='object')
```

In [44]:
```
result.index.get_level_values(1)
```

Out[44]:
```
Int64Index([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11], dtype='int64')
```

In [45]:
```
df4 = pd.DataFrame({'B': ['B2', 'B3', 'B6', 'B7'],
                    'D': ['D2', 'D3', 'D6', 'D7'],
                    'F': ['F2', 'F3', 'F6', 'F7']},
                   index=[2, 3, 6, 7])

result = pd.concat([df1, df4], axis=1)
```

In [46]:
```
df1
```

Out[46]:

	A	B	C	D
0	A0	B0	C0	D0
1	A1	B1	C1	D1
2	A2	B2	C2	D2
3	A3	B3	C3	D3

In [47]:
```
df4
```

Out[47]:

	B	D	F
2	B2	D2	F2
3	B3	D3	F3
6	B6	D6	F6
7	B7	D7	F7

In [48]:
```
result
```

Out[48]:

	A	B	C	D	B	D	F
0	A0	B0	C0	D0	NaN	NaN	NaN
1	A1	B1	C1	D1	NaN	NaN	NaN
2	A2	B2	C2	D2	B2	D2	F2
3	A3	B3	C3	D3	B3	D3	F3
6	NaN	NaN	NaN	NaN	B6	D6	F6
7	NaN	NaN	NaN	NaN	B7	D7	F7

In [49]:
```
# inner 조인
result = pd.concat([df1, df4], axis=1, join='inner')
result
```

Out[49]:

	A	B	C	D	B	D	F
2	A2	B2	C2	D2	B2	D2	F2
3	A3	B3	C3	D3	B3	D3	F3

In [50]:
```
## merge
left = pd.DataFrame({'key': ['K0', 'K4', 'K2', 'K3'],
                     'A': ['A0', 'A1', 'A2', 'A3'],
                     'B': ['B0', 'B1', 'B2', 'B3']})

right = pd.DataFrame({'key': ['K0', 'K1', 'K2', 'K3'],
                      'C': ['C0', 'C1', 'C2', 'C3'],
                      'D': ['D0', 'D1', 'D2', 'D3']})
```

In [51]:
```
left
```

Out[51]:

	key	A	B
0	K0	A0	B0
1	K4	A1	B1
2	K2	A2	B2
3	K3	A3	B3

In [52]:
```
right
```

Out[52]:

	key	C	D

	key	C	D
0	K0	C0	D0
1	K1	C1	D1
2	K2	C2	D2
3	K3	C3	D3

In [53]:
```
pd.merge(left, right, on='key')
```

Out[53]:

	key	A	B	C	D
0	K0	A0	B0	C0	D0
1	K2	A2	B2	C2	D2
2	K3	A3	B3	C3	D3

In [54]:
```
pd.merge(left, right, how='left', on='key')
```

Out[54]:

	key	A	B	C	D
0	K0	A0	B0	C0	D0
1	K4	A1	B1	NaN	NaN
2	K2	A2	B2	C2	D2
3	K3	A3	B3	C3	D3

In [55]:
```
# outer 조인
pd.merge(left, right, how='outer', on='key')
```

Out[55]:

	key	A	B	C	D
0	K0	A0	B0	C0	D0
1	K4	A1	B1	NaN	NaN
2	K2	A2	B2	C2	D2
3	K3	A3	B3	C3	D3
4	K1	NaN	NaN	C1	D1

cctv데이터와 인구 데이터 합치기

In [56]:
```
merge_result = pd.merge(cctv, pop, on='구별')
merge_result.head()
```

Out[56]:

	구별	총계	2012년 이전	2012년	2013년	2014년	2015년	2016년	2017년	2018년	2019년	2020년	2021년	최근증가율
0	종로구	1772.0	813.0	0.0	0.0	210.0	150.0	1.0	261.0	85.0	9.0	200.0	43.0	0.234843

	구별	총계	2012년 이전	2012년	2013년	2014년	2015년	2016년	2017년	2018년	2019년	2020년	2021년	최근증가율
1	용산구	2383.0	34.0	71.0	234.0	125.0	221.0	298.0	351.0	125.0	307.0	617.0	0.0	0.786357
2	성동구	3602.0	448.0	125.0	212.0	105.0	339.0	310.0	874.0	390.0	262.0	461.0	76.0	0.492748
3	광진구	2588.0	35.0	57.0	100.0	187.0	98.0	52.0	675.0	465.0	712.0	175.0	32.0	1.149502
4	동대문구	2497.0	1090.0	146.0	60.0	29.0	111.0	233.0	136.0	197.0	209.0	223.0	63.0	0.383380

In [57]:
```
merge_result.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 24 entries, 0 to 23
Data columns (total 20 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   구별        24 non-null     object
 1   총계        24 non-null     float64
 2   2012년 이전  24 non-null     float64
 3   2012년     24 non-null     float64
 4   2013년     24 non-null     float64
 5   2014년     24 non-null     float64
 6   2015년     24 non-null     float64
 7   2016년     24 non-null     float64
 8   2017년     24 non-null     float64
 9   2018년     24 non-null     float64
 10  2019년     24 non-null     float64
 11  2020년     24 non-null     float64
 12  2021년     24 non-null     float64
 13  최근증가율     24 non-null     float64
 14  인구수       24 non-null     int64
 15  한국인       24 non-null     int64
 16  외국인       24 non-null     int64
 17  고령자       24 non-null     int64
 18  외국인비율     24 non-null     float64
 19  고령자비율     24 non-null     float64
dtypes: float64(15), int64(4), object(1)
memory usage: 3.9+ KB
```

In [58]:
```
drop_col = merge_result.columns[2:13]
drop_col
```

Out[58]:
```
Index(['2012년 이전', '2012년', '2013년', '2014년', '2015년', '2016년', '2017년',
       '2018년', '2019년', '2020년', '2021년'],
      dtype='object')
```

In [59]:
```
merge_result = merge_result.drop(drop_col,axis = 1)
merge_result
```

Out[59]:

	구별	총계	최근증가율	인구수	한국인	외국인	고령자	외국인비율	고령자비율
0	종로구	1772.0	0.234843	153789	144683	9106	27818	0.059211	0.180884
1	용산구	2383.0	0.786357	237285	222953	14332	39070	0.060400	0.164654

	구별	총계	최근증가율	인구수	한국인	외국인	고령자	외국인비율	고령자비율
2	성동구	3602.0	0.492748	292672	285990	6682	46380	0.022831	0.158471
3	광진구	2588.0	1.149502	352627	339996	12631	51723	0.035820	0.146679
4	동대문구	2497.0	0.383380	352006	337400	14606	62211	0.041494	0.176733
5	중랑구	3296.0	1.914235	391885	387350	4535	71682	0.011572	0.182916
6	성북구	3958.0	1.042312	440142	430528	9614	74709	0.021843	0.169738
7	강북구	2462.0	4.557562	302563	299182	3381	64333	0.011175	0.212627
8	도봉구	1629.0	1.294366	319373	317366	2007	64160	0.006284	0.200894
9	노원구	2415.0	0.738661	514946	510956	3990	88345	0.007748	0.171562
10	은평구	3791.0	1.256548	477173	473307	3866	87241	0.008102	0.182829
11	서대문구	2940.0	0.498471	315659	304819	10840	54268	0.034341	0.171920
12	마포구	2421.0	0.915348	378686	368905	9781	54582	0.025829	0.144135
13	양천구	3312.0	0.287213	450487	447302	3185	68627	0.007070	0.152340
14	강서구	2560.0	0.875458	579768	574315	5453	92558	0.009405	0.159647
15	구로구	4075.0	0.488313	421163	396754	24409	72611	0.057956	0.172406
16	금천구	2374.0	1.504219	244891	230811	14080	41041	0.057495	0.167589
17	영등포구	3778.0	1.056614	400908	376837	24071	62516	0.060041	0.155936
18	동작구	2297.0	0.844980	394364	385483	8881	66613	0.022520	0.168912
19	관악구	4942.0	0.509469	499449	485699	13750	79871	0.027530	0.159918
20	서초구	3704.0	0.484569	416167	412279	3888	60678	0.009342	0.145802
21	강남구	6502.0	0.616609	537800	533042	4758	78226	0.008847	0.145456
22	송파구	2854.0	2.143172	663965	658338	5627	97691	0.008475	0.147133
23	강동구	2547.0	1.122500	466472	462664	3808	74070	0.008163	0.158788

In [60]:
```
# 인덱스 재설정
merge_result.set_index('구별', inplace=True)
merge_result.head()
```

Out[60]:

구별	총계	최근증가율	인구수	한국인	외국인	고령자	외국인비율	고령자비율
종로구	1772.0	0.234843	153789	144683	9106	27818	0.059211	0.180884
용산구	2383.0	0.786357	237285	222953	14332	39070	0.060400	0.164654
성동구	3602.0	0.492748	292672	285990	6682	46380	0.022831	0.158471
광진구	2588.0	1.149502	352627	339996	12631	51723	0.035820	0.146679
동대문구	2497.0	0.383380	352006	337400	14606	62211	0.041494	0.176733

In [61]:
```
np.corrcoef(merge_result['고령자비율'],merge_result['총계'])
# 고령자 비율은 CCTV 설치 개수와 음의 상관관계를 갖는다.
```

Out[61]:
```
array([[ 1.        , -0.37376628],
```

```
       [-0.37376628,  1.        ]])
```

In [62]:
```
np.corrcoef(merge_result['외국인비율'],merge_result['총계'])
# 외국인 비율은 CCTV 설치 개수와 음의 상관관계를 갖는다.
```

Out[62]:
```
array([[ 1.        , -0.13359732],
       [-0.13359732,  1.        ]])
```

In [63]:
```
np.corrcoef(merge_result['인구수'],merge_result['총계'])
# 인구수는 CCTV 설치 개수와 양의 상관관계를 갖는다.
```

Out[63]:
```
array([[1.        , 0.44295137],
       [0.44295137, 1.        ]])
```

1.6 그래프 그리기

In [64]:
```
# #나눔 글꼴 설치
# !sudo apt-get install -y fonts-nanum
# !sudo fc-cache -fv
# !rm ~/.cache/matplotlib -rf

# #설치후 상단에서 런타임 - 런타임 다시 시작


import matplotlib.pyplot as plt
%matplotlib inline
# 플롯 스타일 설정
plt.style.use('fivethirtyeight')

# matplotlib을 사용하는 모든 plot에 나눔 글꼴 적용
plt.rc('font', family='NanumBarunGothic')
```

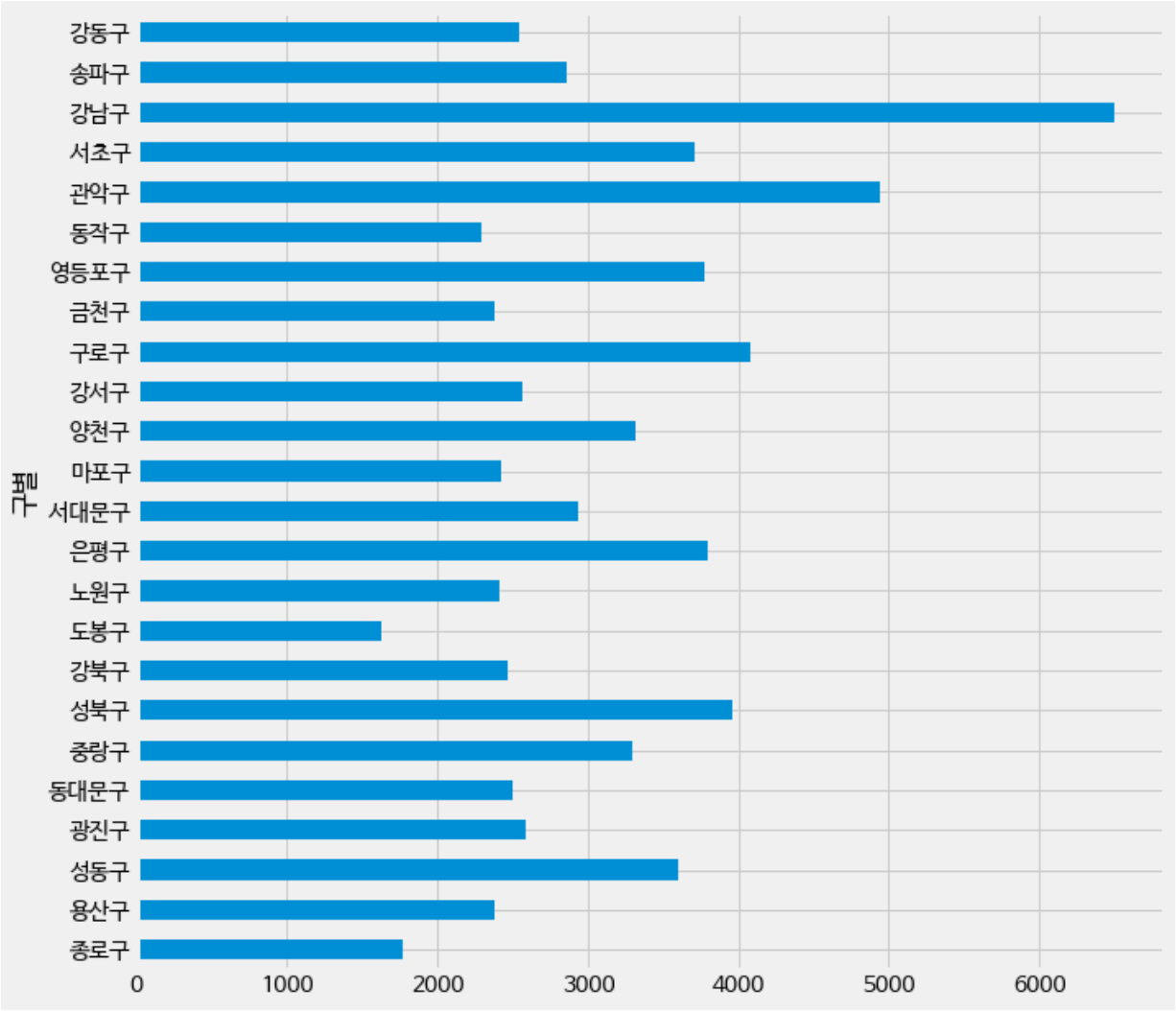
In [64]:
```
```

In [65]:
```
merge_result.head()
```

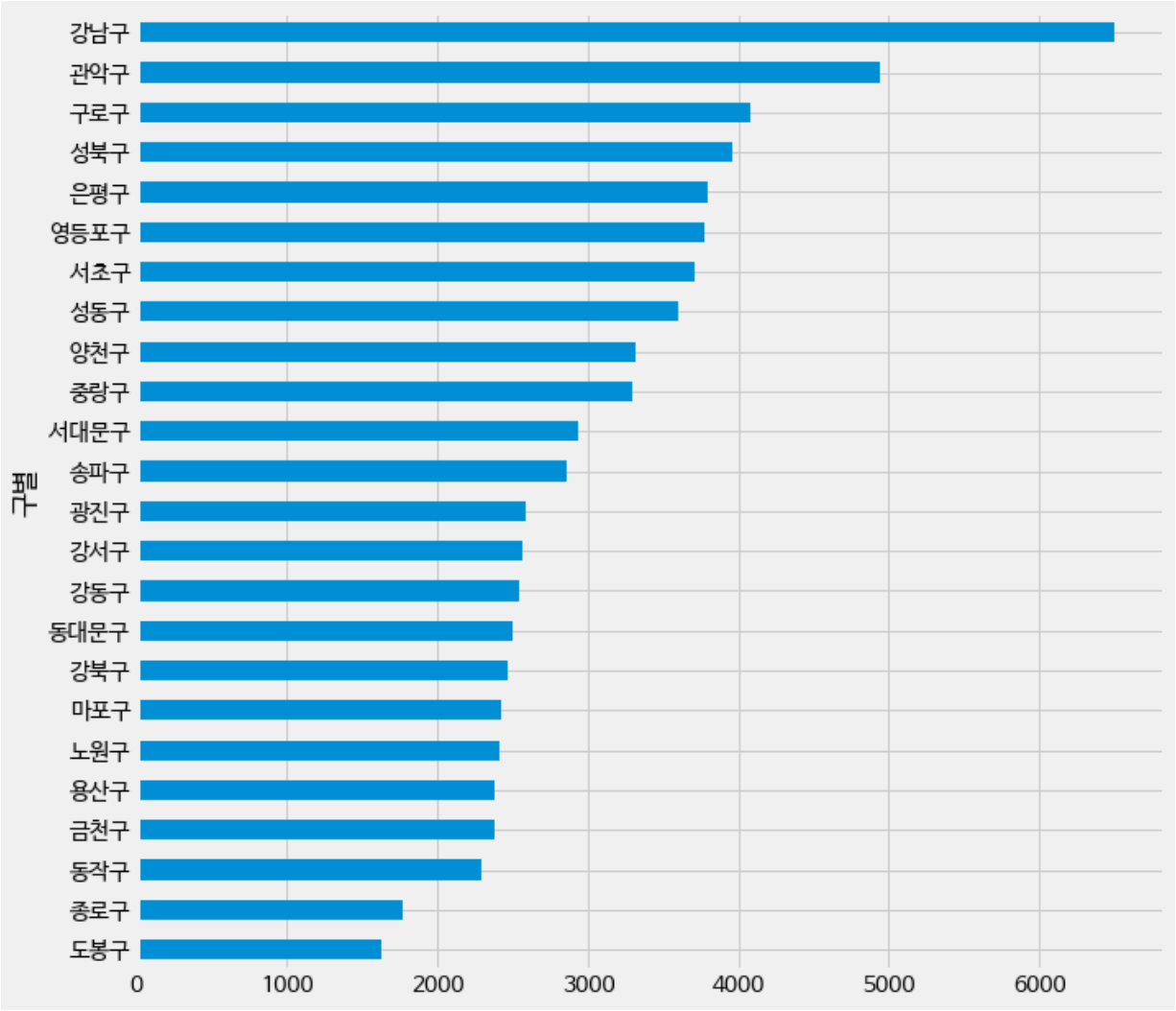
Out[65]:

구별	총계	최근증가율	인구수	한국인	외국인	고령자	외국인비율	고령자비율
종로구	1772.0	0.234843	153789	144683	9106	27818	0.059211	0.180884
용산구	2383.0	0.786357	237285	222953	14332	39070	0.060400	0.164654
성동구	3602.0	0.492748	292672	285990	6682	46380	0.022831	0.158471
광진구	2588.0	1.149502	352627	339996	12631	51723	0.035820	0.146679
동대문구	2497.0	0.383380	352006	337400	14606	62211	0.041494	0.176733

In [66]:
```
# 바형 차트로 그리기
plt.figure()
merge_result['총계'].plot(kind='barh', grid=True, figsize=(10,10))
plt.show()
```

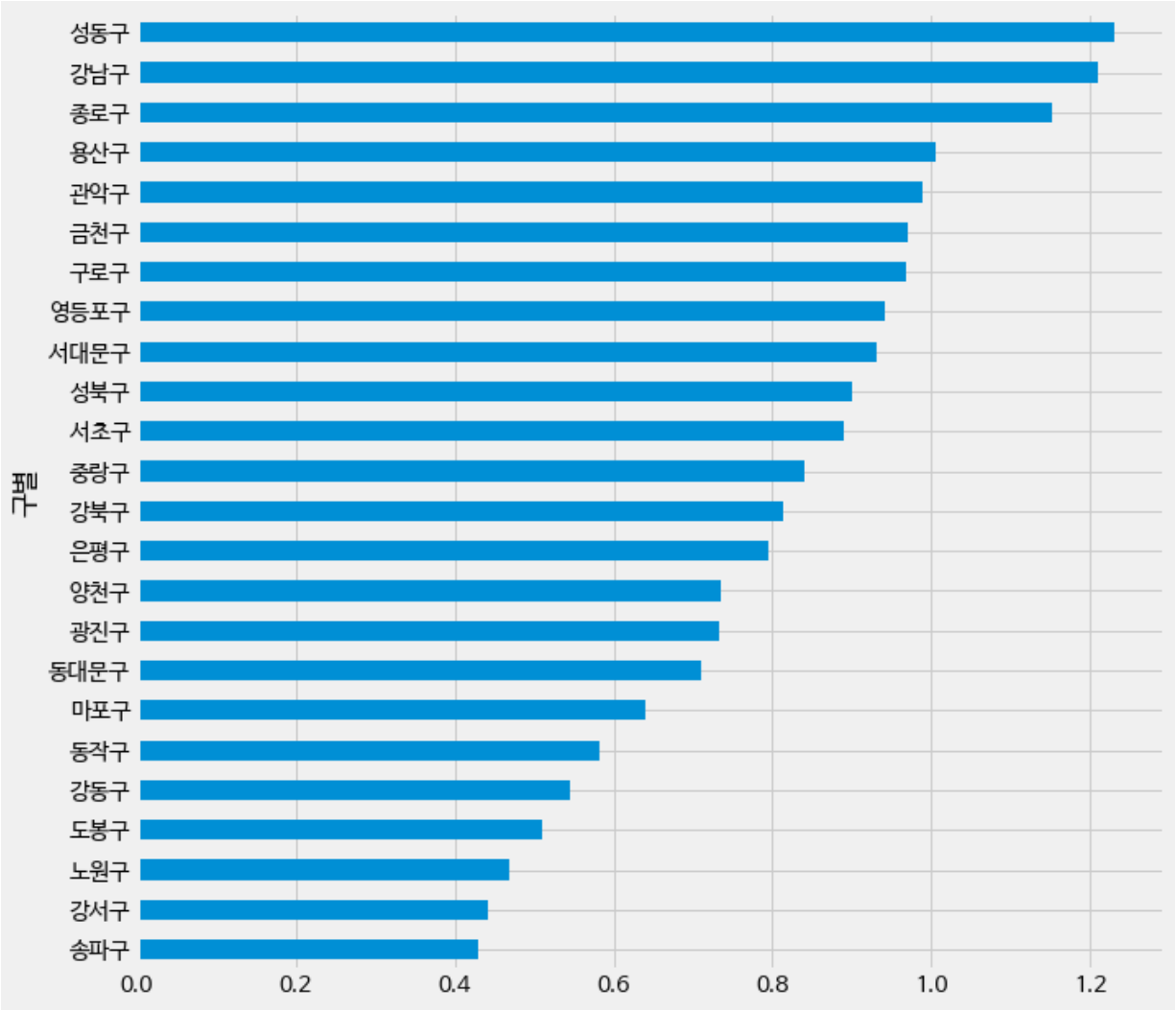


In [67]:
```
# 데이터 정렬하여 그리기
merge_result['총계'].sort_values().plot(kind='barh',
                                        grid=True, figsize=(10,10))
plt.show()
```



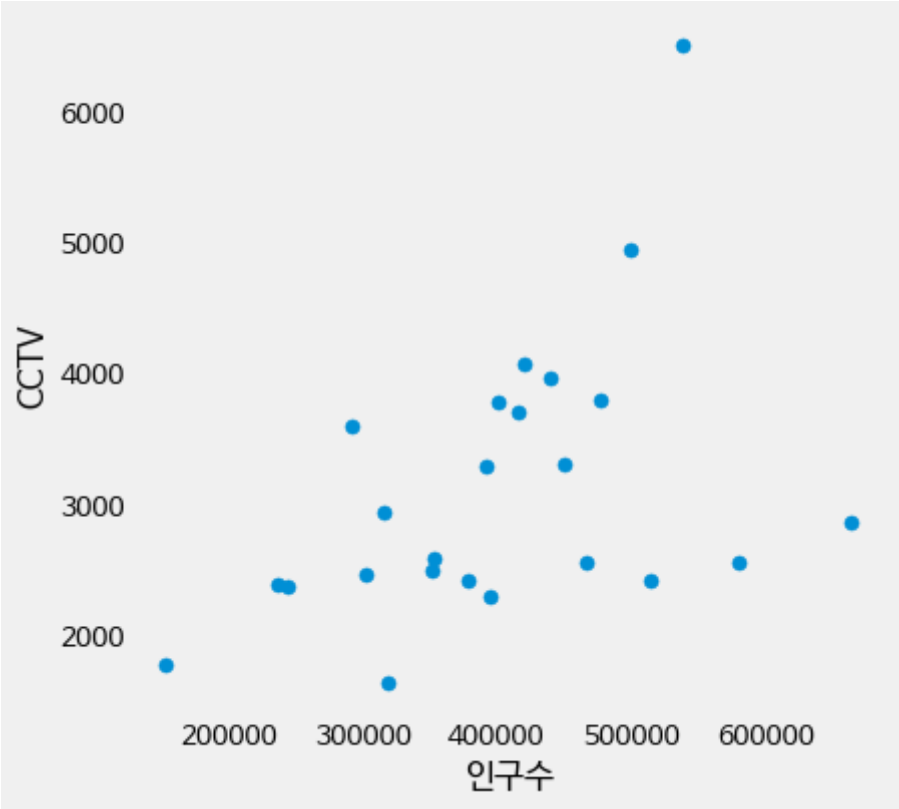
In [68]:
```
merge_result['CCTV비율'] = merge_result['총계'] / merge_result['인구수'] * 100

merge_result['CCTV비율'].sort_values().plot(kind='barh',
                                            grid=True, figsize=(10,10))
plt.show()
```



In [69]:

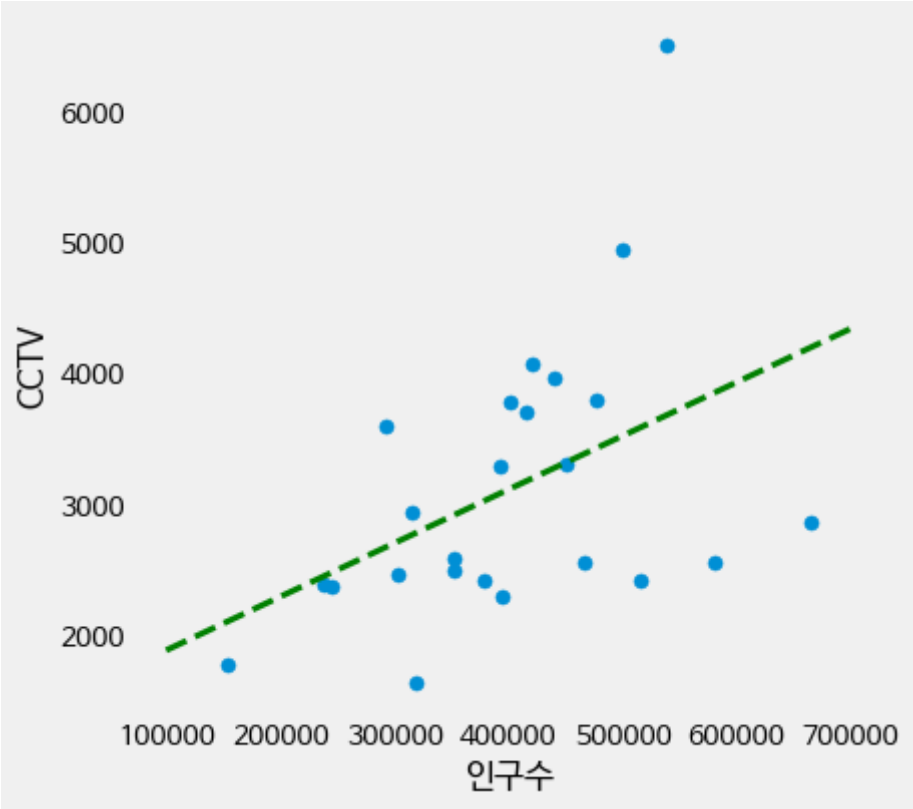
```
# scatter plot 그리기
plt.figure(figsize=(6,6))
plt.scatter(merge_result['인구수'], merge_result['총계'], s=50)
plt.xlabel('인구수')
plt.ylabel('CCTV')
plt.grid()
plt.show()
```



In [70]:
```
# 추세선 그리기 위한 값 설정
fp1 = np.polyfit(merge_result['인구수'], merge_result['총계'], 1)
f1 = np.poly1d(fp1)
fx = np.linspace(100000, 700000, 100)
```

In [71]:
```
# 추세선 추가

plt.figure(figsize=(6,6))
plt.scatter(merge_result['인구수'], merge_result['총계'], s=50)
plt.plot(fx, f1(fx), ls='dashed', lw=3, color='g') # 추가
plt.xlabel('인구수')
plt.ylabel('CCTV')
plt.grid()
plt.show()
```



1.7 최종 마무리

In [72]:
```
fp1 = np.polyfit(merge_result['인구수'], merge_result['총계'], 1)

f1 = np.poly1d(fp1)
fx = np.linspace(100000, 700000, 100)

# 오차 열 새로 생성
merge_result['오차'] = np.abs(merge_result['총계'] - f1(merge_result['인구수']))

df_sort = merge_result.sort_values(by='오차', ascending=False)
df_sort.head()
```

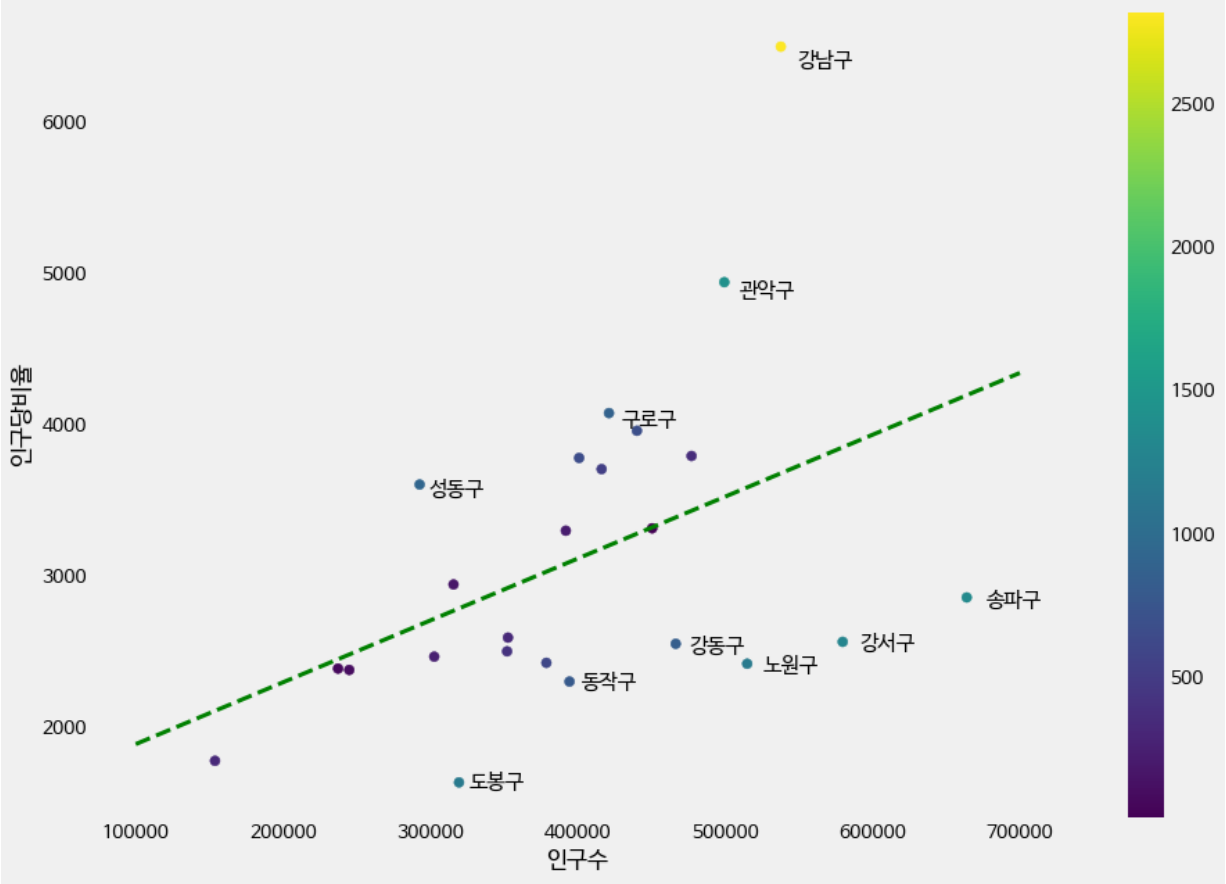
Out[72]:

구별	총계	최근증가율	인구수	한국인	외국인	고령자	외국인비율	고령자비율	CCTV비율	오차
강남구	6502.0	0.616609	537800	533042	4758	78226	0.008847	0.145456	1.209000	2825.609537
관악구	4942.0	0.509469	499449	485699	13750	79871	0.027530	0.159918	0.989490	1422.758478
송파구	2854.0	2.143172	663965	658338	5627	97691	0.008475	0.147133	0.429842	1339.370363
강서구	2560.0	0.875458	579768	574315	5453	92558	0.009405	0.159647	0.441556	1288.360601
노원구	2415.0	0.738661	514946	510956	3990	88345	0.007748	0.171562	0.468981	1167.742790

In [73]:
```
plt.figure(figsize=(14,10))
plt.scatter(merge_result['인구수'], merge_result['총계'],
            c=merge_result['오차'], s=50)
plt.plot(fx, f1(fx), ls='dashed', lw=3, color='g')

for n in range(10):
    plt.text(df_sort['인구수'][n]*1.02, df_sort['총계'][n]*0.98,
             df_sort.index[n], fontsize=15)

plt.xlabel('인구수')
plt.ylabel('인구당비율')
plt.colorbar()
plt.grid()
plt.show()
```



- 상단에 위치한 강남구, 관악구 등은 인구수에 비해 많은 cctv가 위치해 있는 지역이다.
- 하단에 위치한 송파구, 강서구 등의 지역은 인구수 대비 cctv가 너무 적게 설치 되어 있는 지역이다.