```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

Lab 2 : Web scraping and API requests

In this lab exercise you will practice scraping data from a website, as well as doing some priliminary analysis on them.

Deadline: Friday, Feb 25 11:59

Part 1: Scraping Data From Wikipedia

We have completed a similar task during lecture. You have to scrap a specific page of Wikipedia and answer some questions regarding the data you have collected. You have to get the data about different countries and their respective populations from the following page: https://en.wikipedia.org/wiki/List_of_countries_by_past_and_future_population

This page contains multiple tables for past and future population of countries. For the first part of this lab do the following:

1. Fetch the data from wikipedia with "requests" library
2. Parse html data with BeautifulSoup library
3. Use BeautifulSoup to extract specific tables
4. Combine the tables and convert the data into a dictionary
5. Make a pandas dataframe from the dictionary
6. Answer some questions and do some basic visualization!

1.1 Get the data from wikipedia (5 pts)

Use "requests" library.

```
# Your code here
import requests
url = "https://en.wikipedia.org/wiki/List_of_countries_by_past_and_projected_future_population
response =  requests.get(url)
```

1.2 Parse html data with BeautifulSoup (10 pts)

Parse the data using BeautifulSoup. Remember that BeautifulSoup has many useful attributes such as prettify(), find(attribute), and find_all(attribute). Check the documentation for more info: https://www.crummy.com/software/BeautifulSoup/bs4/doc/

1.2.a Find the first title object and extract and print the string stored in it (5 pts)

```
# Your code here
from bs4 import BeautifulSoup as bsoup

wiki_content = response.text

soup = bsoup(wiki_content,'html5lib')

soup.find_all('title')[0].text
```

```
'List of countries by past and projected future population - Wikipedia'
```

1.2.b Find all the paragrpahs, store them in a list, and print the first 10 (5 pts)

```
# Your code here
paragraph_list = []
for i in soup.find_all('p'):
  paragraph_list.append(i.text)

paragraph_list[:10]
```

```
['All the figures shown here have been sourced from the International Data Base (IDB) Div
 'Population estimates, as long as they are based on recent censuses, can be more easily
 'However, no projected population figures can be considered exact. As the IDB states, "f
 'To make things complicated, not all countries carry out censuses regularly, especially
 "On the other hand, some other countries, like the small Asian state of Bhutan, have onl
 'Besides, the IDB usually takes some time before including new data, as happened in the
 'The largest absolute potential discrepancies are naturally related to the most populous
 'The national 1 July, mid-year population estimates (usually based on past national cens
 'The table columns can be sorted by clicking on their respective heading.\n',
 'The retrospective figures use the present-day names and world political division: for e
```

1.3 Extract the tables (10 pts)

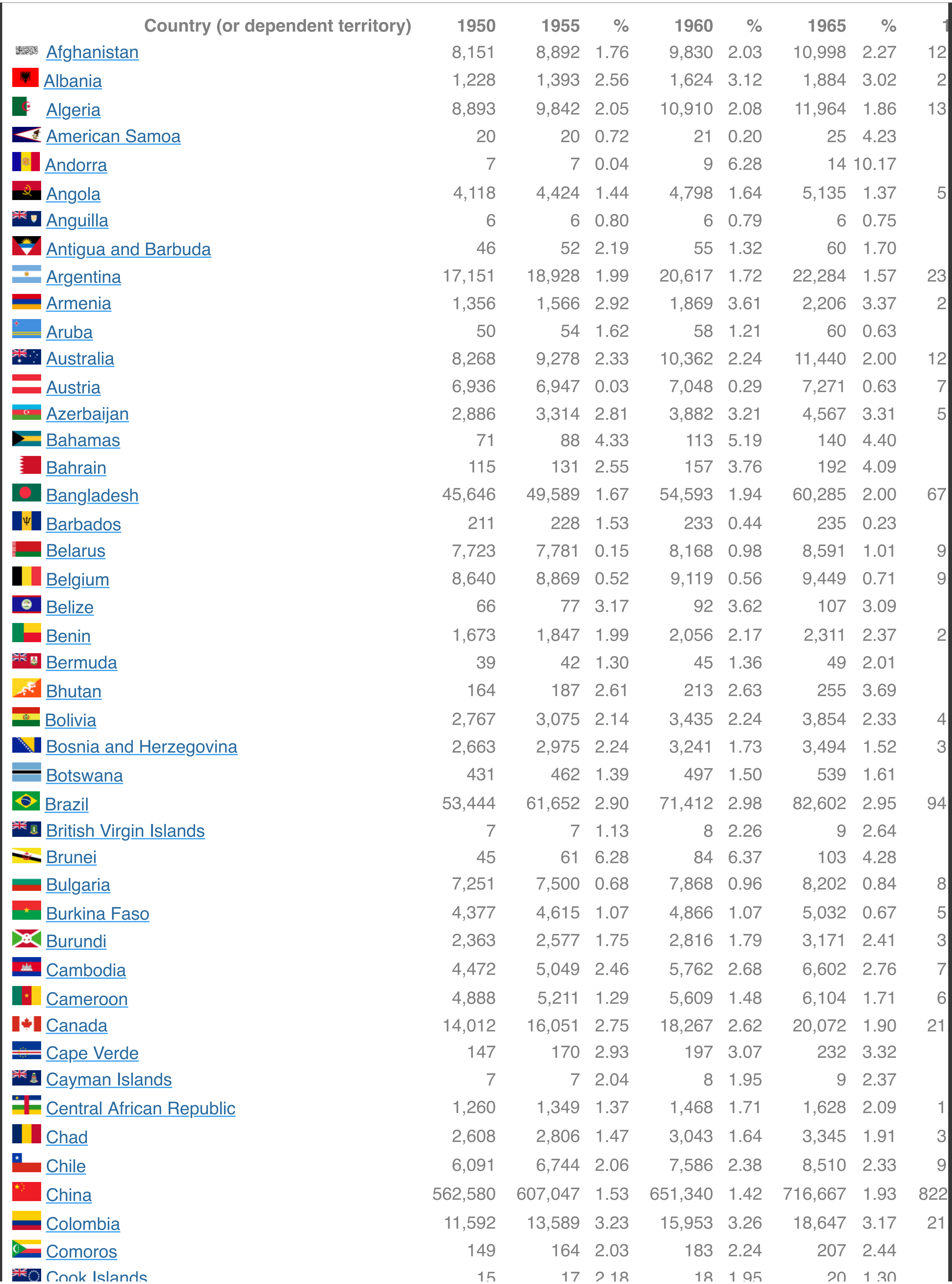
We only care about the tables that contain historical population data. Extract all of them.

```
# Your code here
# You need to  find all objects that include the css class “wikitable” within the soup object.

tables  = soup.find_all("table",{"class":"sortable wikitable"})
```

```
# check the tables you extracted

from IPython.core.display import display, HTML
display(HTML(tables[0].prettify()))
```



Country (or dependent territory)	1950	1955	%	1960	%	1965	%	1
Afghanistan	8,151	8,892	1.76	9,830	2.03	10,998	2.27	12
Albania	1,228	1,393	2.56	1,624	3.12	1,884	3.02	2
Algeria	8,893	9,842	2.05	10,910	2.08	11,964	1.86	13
American Samoa	20	20	0.72	21	0.20	25	4.23	
Andorra	7	7	0.04	9	6.28	14	10.17	
Angola	4,118	4,424	1.44	4,798	1.64	5,135	1.37	5
Anguilla	6	6	0.80	6	0.79	6	0.75	
Antigua and Barbuda	46	52	2.19	55	1.32	60	1.70	
Argentina	17,151	18,928	1.99	20,617	1.72	22,284	1.57	23
Armenia	1,356	1,566	2.92	1,869	3.61	2,206	3.37	2
Aruba	50	54	1.62	58	1.21	60	0.63	
Australia	8,268	9,278	2.33	10,362	2.24	11,440	2.00	12
Austria	6,936	6,947	0.03	7,048	0.29	7,271	0.63	7
Azerbaijan	2,886	3,314	2.81	3,882	3.21	4,567	3.31	5
Bahamas	71	88	4.33	113	5.19	140	4.40	
Bahrain	115	131	2.55	157	3.76	192	4.09	
Bangladesh	45,646	49,589	1.67	54,593	1.94	60,285	2.00	67
Barbados	211	228	1.53	233	0.44	235	0.23	
Belarus	7,723	7,781	0.15	8,168	0.98	8,591	1.01	9
Belgium	8,640	8,869	0.52	9,119	0.56	9,449	0.71	9
Belize	66	77	3.17	92	3.62	107	3.09	
Benin	1,673	1,847	1.99	2,056	2.17	2,311	2.37	2
Bermuda	39	42	1.30	45	1.36	49	2.01	
Bhutan	164	187	2.61	213	2.63	255	3.69	
Bolivia	2,767	3,075	2.14	3,435	2.24	3,854	2.33	4
Bosnia and Herzegovina	2,663	2,975	2.24	3,241	1.73	3,494	1.52	3
Botswana	431	462	1.39	497	1.50	539	1.61	
Brazil	53,444	61,652	2.90	71,412	2.98	82,602	2.95	94
British Virgin Islands	7	7	1.13	8	2.26	9	2.64	
Brunei	45	61	6.28	84	6.37	103	4.28	
Bulgaria	7,251	7,500	0.68	7,868	0.96	8,202	0.84	8
Burkina Faso	4,377	4,615	1.07	4,866	1.07	5,032	0.67	5
Burundi	2,363	2,577	1.75	2,816	1.79	3,171	2.41	3
Cambodia	4,472	5,049	2.46	5,762	2.68	6,602	2.76	7
Cameroon	4,888	5,211	1.29	5,609	1.48	6,104	1.71	6
Canada	14,012	16,051	2.75	18,267	2.62	20,072	1.90	21
Cape Verde	147	170	2.93	197	3.07	232	3.32	
Cayman Islands	7	7	2.04	8	1.95	9	2.37	
Central African Republic	1,260	1,349	1.37	1,468	1.71	1,628	2.09	1
Chad	2,608	2,806	1.47	3,043	1.64	3,345	1.91	3
Chile	6,091	6,744	2.06	7,586	2.38	8,510	2.33	9
China	562,580	607,047	1.53	651,340	1.42	716,667	1.93	822
Colombia	11,592	13,589	3.23	15,953	3.26	18,647	3.17	21
Comoros	149	164	2.03	183	2.24	207	2.44	
Cook Islands	15	17	2.18	18	1.95	20	1.30	

Cook Islands	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	
Costa Rica	867	1,032	3.54	1,249	3.88	1,488	3.57	1
Croatia	3,838	3,956	0.61	4,037	0.40	4,134	0.48	4
Cuba	5,785	6,382	1.98	7,028	1.95	7,810	2.13	8
Curaçao	102	112	2.04	124	2.04	134	1.59	
Cyprus	495	533	1.52	579	1.68	601	0.72	
Czech Republic	8,926	9,366	0.97	9,660	0.62	9,777	0.24	9
Democratic Republic of the Congo	13,569	14,953	1.96	16,611	2.13	18,856	2.57	21
Denmark	4,272	4,440	0.77	4,582	0.63	4,759	0.76	4
Djibouti	80	91	2.67	112	4.26	143	5.00	
Dominica	52	57	1.93	60	1.13	64	1.29	
Dominican Republic	2,353	2,738	3.07	3,232	3.38	3,806	3.33	4
Ecuador	3,370	3,843	2.66	4,416	2.82	5,118	2.99	5
Egypt	21,198	23,856	2.39	26,847	2.39	30,266	2.43	33
El Salvador	1,940	2,222	2.75	2,582	3.05	3,018	3.17	3
Equatorial Guinea	212	226	1.35	244	1.54	253	0.72	
Eritrea	1,403	1,499	1.34	1,615	1.50	1,747	1.58	2
Estonia	1,096	1,155	1.05	1,211	0.96	1,288	1.24	1
Eswatini	278	312	2.34	352	2.48	400	2.57	
Ethiopia	20,175	21,991	1.74	24,169	1.91	26,741	2.04	29
Faroe Islands	32	33	0.69	35	1.26	38	1.29	
Federated States of Micronesia	31	36	3.12	42	3.12	49	3.12	
Fiji	288	333	2.94	394	3.44	464	3.33	
Finland	4,009	4,235	1.10	4,430	0.90	4,564	0.60	4
France	42,540	44,243	0.79	46,612	1.05	49,834	1.35	51
French Polynesia	63	72	2.97	82	2.44	95	3.15	
Gabon	416	429	0.62	447	0.79	475	1.24	
Gambia	272	307	2.46	352	2.81	412	3.20	
Georgia	3,516	3,828	1.71	4,147	1.62	4,465	1.49	4
Germany	68,375	70,196	0.53	72,481	0.64	75,639	0.86	77
Ghana	5,298	6,049	2.69	6,959	2.84	8,010	2.85	8
Gibraltar	23	24	0.58	25	0.49	26	0.88	
Greece	7,567	7,966	1.03	8,328	0.89	8,551	0.53	8
Greenland	23	27	3.59	33	4.09	40	3.83	
Grenada	76	85	2.22	91	1.27	94	0.69	
Guam	60	69	2.78	67	-0.53	75	2.07	
Guatemala	2,969	3,488	3.27	4,100	3.29	4,746	2.97	5
Guernsey	46	47	0.36	47	0.33	51	1.31	
Guinea	2,586	2,787	1.51	3,027	1.66	3,314	1.83	3
Guinea-Bissau	574	592	0.64	617	0.82	604	-0.42	
Guyana	428	492	2.79	572	3.06	641	2.31	
Haiti	3,098	3,365	1.67	3,697	1.90	4,094	2.06	4
Honduras	1,432	1,663	3.03	1,952	3.26	2,330	3.60	2
Hong Kong	2,238	2,491	2.17	3,076	4.31	3,598	3.19	3
Hungary	9,339	9,826	1.02	9,984	0.32	10,153	0.34	10
Iceland	143	159	2.03	176	2.16	193	1.80	

India	369,881	404,268	1.79	445,394	1.96	494,964	2.13	553
Indonesia	82,979	90,255	1.70	100,146	2.10	110,754	2.03	122
Iran	16,358	18,739	2.76	21,600	2.88	25,040	3.00	28
Iraq	5,164	5,904	2.71	6,823	2.94	7,971	3.16	9
Ireland	2,964	2,917	-0.32	2,833	-0.58	2,877	0.31	2
Isle of Man	55	52	-1.40	48	-1.34	49	0.52	
Israel	1,287	1,771	6.60	2,139	3.84	2,573	3.77	2
Italy	47,106	48,634	0.64	50,198	0.64	51,988	0.70	53
Ivory Coast	2,861	3,165	2.04	3,577	2.48	4,357	4.03	5
Jamaica	1,385	1,489	1.46	1,632	1.85	1,778	1.72	1
Japan	83,806	89,816	1.39	94,092	0.93	98,883	1.00	104
Jersey	57	60	1.04	63	1.02	67	1.04	
Jordan	562	689	4.17	853	4.36	1,069	4.62	1
Kazakhstan	6,694	7,977	3.57	9,983	4.59	11,903	3.58	13
Kenya	6,122	7,034	2.82	8,157	3.01	9,550	3.20	11
Kiribati	34	37	1.99	41	1.99	45	1.90	
Kosovo	762	854	2.32	947	2.10	1,079	2.63	1
Kuwait	145	187	5.21	293	9.38	477	10.26	
Kyrgyzstan	1,739	1,902	1.80	2,172	2.69	2,574	3.45	2
Laos	1,886	2,078	1.95	2,310	2.14	2,566	2.12	2
Latvia	1,937	2,003	0.67	2,116	1.10	2,254	1.28	2
Lebanon	1,365	1,561	2.73	1,787	2.73	2,058	2.87	2
Lesotho	727	787	1.60	859	1.78	953	2.10	1
Liberia	824	929	2.41	1,055	2.59	1,209	2.76	1
Libya	962	1,123	3.15	1,338	3.57	1,624	3.95	2
Liechtenstein	14	15	1.72	17	2.25	19	2.65	
Lithuania	2,554	2,615	0.47	2,765	1.13	2,960	1.37	3
Luxembourg	296	305	0.61	314	0.60	332	1.09	
Macau	206	193	-1.27	187	-0.70	224	3.77	
Madagascar	4,621	5,003	1.60	5,482	1.85	6,071	2.06	6
Malawi	2,817	3,089	1.86	3,451	2.24	3,915	2.55	4
Malaysia	6,434	7,312	2.59	8,429	2.88	9,648	2.74	10
Maldives	80	81	0.21	93	2.86	98	1.17	
Mali	3,688	4,072	2.00	4,495	2.00	4,978	2.06	5
Malta	312	315	0.15	329	0.88	320	-0.58	
Marshall Islands	11	13	3.32	16	3.32	18	3.32	
Mauritania	1,006	1,054	0.93	1,118	1.19	1,196	1.35	1
Mauritius	482	572	3.50	664	3.02	756	2.64	
Mexico	28,486	32,930	2.94	38,579	3.22	45,143	3.19	52
Moldova	2,337	2,623	2.34	2,999	2.72	3,334	2.14	3
Monaco	19	19	0.17	21	2.57	23	1.59	
Mongolia	779	845	1.63	955	2.49	1,091	2.69	1
Montenegro	396	432	1.76	462	1.35	492	1.25	
Montserrat	14	13	-1.08	13	-0.99	12	-0.62	
Morocco	9,344	10,782	2.90	12,424	2.88	14,067	2.51	15
Mozambique	6,251	6,782	1.64	7,473	1.96	8,302	2.13	9
M	[illegible]	21,050	1.55	[illegible]	1.64	[illegible]	1.77	27

Myanmar	19,488	21,050	1.55	22,840	1.64	24,938	1.77	27
Namibia	464	522	2.39	591	2.51	671	2.58	
Nauru	4	4	1.38	5	4.01	6	4.44	
Nepal	8,990	9,480	1.07	10,035	1.15	10,863	1.60	11
Netherlands	10,121	10,759	1.23	11,494	1.33	12,302	1.37	13
New Caledonia	56	65	3.36	79	3.94	91	2.77	
New Zealand	1,909	2,137	2.28	2,372	2.11	2,641	2.17	2
Nicaragua	1,098	1,278	3.07	1,493	3.17	1,751	3.24	2
Niger	3,272	3,560	1.70	3,913	1.91	4,344	2.11	4
Nigeria	31,797	35,955	2.49	41,551	2.94	48,068	2.96	55
North Korea	9,472	8,864	-1.32	10,448	3.34	11,965	2.75	14
North Macedonia	1,225	1,341	1.82	1,367	0.39	1,470	1.47	1
Northern Mariana Islands	7	8	3.52	9	3.47	11	3.38	
Norway	3,266	3,428	0.97	3,582	0.88	3,724	0.78	3
Oman	489	540	2.02	602	2.17	682	2.55	
Pakistan	40,383	45,536	2.43	51,719	2.58	59,047	2.69	67
Palau	8	9	2.72	10	2.72	11	2.72	
Palestine	1,018	1,055	0.72	1,115	1.11	1,212	1.68	1
Panama	893	1,011	2.52	1,148	2.57	1,326	2.93	1
Papua New Guinea	1,413	1,546	1.81	1,719	2.15	1,941	2.46	2
Paraguay	1,476	1,684	2.67	1,910	2.55	2,171	2.59	2
Peru	7,633	8,672	2.59	9,932	2.75	11,468	2.92	13
Philippines	21,132	24,336	2.86	28,026	2.86	32,391	2.94	37
Poland	24,825	27,221	1.86	29,590	1.68	31,263	1.11	32
Portugal	8,443	8,693	0.58	9,037	0.78	9,129	0.20	9
Puerto Rico	2,219	2,251	0.29	2,359	0.94	2,597	1.95	2
Qatar	26	36	6.94	46	5.13	71	9.21	
Republic of the Congo	827	904	1.81	1,003	2.09	1,124	2.32	1
Romania	16,312	17,326	1.21	18,404	1.22	19,028	0.67	20
Russia	101,937	111,126	1.74	119,632	1.49	126,542	1.13	130
Rwanda	2,440	2,699	2.04	3,032	2.36	3,265	1.49	3
Saint Barthélemy	3	3	0.62	3	0.63	3	1.12	
Saint Helena, Ascension and Tristan da Cunha	6	6	0.05	6	0.21	6	0.24	
Saint Kitts and Nevis	45	50	2.31	52	0.56	49	-0.87	
Saint Lucia	80	86	1.66	88	0.43	95	1.42	
Saint Martin	3	4	4.07	5	4.08	5	2.50	
Saint Pierre and Miquelon	5	5	0.40	5	0.98	6	0.63	
Saint Vincent and the Grenadines	67	76	2.58	82	1.43	86	0.98	
Samoa	82	94	2.77	111	3.23	128	2.93	
San Marino	13	14	1.59	16	2.19	18	2.49	
São Tomé and Príncipe	60	61	0.17	64	1.03	69	1.62	
Saudi Arabia	3,860	4,244	1.91	4,719	2.15	5,328	2.46	6
Senegal	2,654	2,927	1.98	3,270	2.24	3,744	2.75	4
Serbia	5,957	6,314	1.17	6,659	1.07	6,959	0.88	7
Seychelles	33	36	1.74	42	3.04	48	2.78	
Sierra Leone	2,088	2,233	1.36	2,397	1.43	2,582	1.50	2

Singapore	1,023	1,306	5.02	1,647	4.75	1,887	2.76	2
Sint Maarten	3	3	1.73	3	1.72	5	8.64	
Slovakia	3,464	3,727	1.48	3,995	1.40	4,370	1.81	4
Slovenia	1,468	1,518	0.67	1,558	0.53	1,621	0.79	1
Solomon Islands	107	115	1.43	127	1.99	144	2.50	
Somalia	2,438	2,674	1.86	2,956	2.03	3,283	2.12	3
South Africa	13,596	15,369	2.48	17,417	2.53	19,899	2.70	22
South Korea	20,846	21,552	0.67	24,785	2.83	28,706	2.98	32
South Sudan	2,707	2,757	0.37	2,809	0.37	2,862	0.37	2
Spain	28,063	29,319	0.88	30,642	0.89	32,085	0.92	33
Sri Lanka	7,534	8,694	2.91	9,914	2.66	11,261	2.58	12
Sudan	6,468	7,391	2.71	8,447	2.71	9,653	2.71	11
Suriname	209	241	2.93	285	3.41	337	3.42	
Sweden	7,015	7,263	0.70	7,481	0.59	7,734	0.67	8
Switzerland	4,695	4,981	1.19	5,363	1.49	5,944	2.08	6
Syria	3,496	3,938	2.41	4,531	2.84	5,323	3.27	6
Taiwan	7,982	9,486	3.51	11,210	3.39	12,978	2.97	14
Tajikistan	1,531	1,781	3.08	2,081	3.16	2,511	3.83	2
Tanzania	7,935	8,971	2.48	10,260	2.72	11,871	2.96	13
Thailand	20,042	23,452	3.19	27,513	3.25	32,062	3.11	37
Timor-Leste	436	473	1.63	509	1.51	554	1.69	
Togo	1,172	1,299	2.07	1,456	2.32	1,648	2.51	1
Tonga	46	55	3.55	64	3.19	75	3.08	
Trinidad and Tobago	633	721	2.66	842	3.14	940	2.23	
Tunisia	3,518	3,847	1.80	4,150	1.53	4,566	1.93	5
Turkey	21,122	24,145	2.71	28,218	3.17	31,951	2.52	35
Turkmenistan	1,205	1,348	2.28	1,585	3.29	1,883	3.50	2
Turks and Caicos Islands	6	6	0.49	6	2.10	6	0.15	
Tuvalu	5	5	1.19	6	1.19	6	1.14	
Uganda	5,522	6,318	2.73	7,262	2.83	8,390	2.93	9
Ukraine	36,775	39,369	1.37	42,645	1.61	45,235	1.19	47
United Arab Emirates	72	83	2.95	104	4.49	144	6.89	
United Kingdom	50,128	50,947	0.32	52,373	0.55	54,351	0.74	55
United States	151,869	165,070	1.68	179,980	1.74	193,527	1.46	203
United States Virgin Islands	27	28	0.52	33	3.32	44	6.00	
Uruguay	2,195	2,354	1.41	2,531	1.47	2,694	1.25	2
Uzbekistan	6,293	7,233	2.82	8,532	3.36	10,206	3.65	11
Vanuatu	53	59	2.43	67	2.43	75	2.34	
Venezuela	5,010	6,171	4.26	7,557	4.14	9,068	3.71	10
Vietnam	25,349	27,739	1.82	31,657	2.68	37,259	3.31	42
Wallis and Futuna	7	8	1.26	8	1.26	9	1.02	
Western Sahara	10	16	11.87	28	11.87	50	12.33	
Yemen	4,778	5,266	1.97	5,872	2.20	6,511	2.09	7
Zambia	2,554	2,870	2.36	3,255	2.55	3,695	2.57	4
Zimbabwe	2,854	3,410	3.62	4,011	3.31	4,686	3.16	5
World	**2,557,629**	**2,782,099**	**1.70**	**3,043,002**	**1.81**	**3,350,426**	**1.94**	**3,712**

1.4 Convert the tables into a dictionary (30 pts)

Looking at the tables, we only care about the population number throughout the history. You want to associate each country with a series of population values to make a proper time series table you can use to analyze the population in a given coutnry.

First, you need to clean the tables cells from any footnote, links, commas or any garbage values. Once your data is cleaned, make a dictionary and combine each country with its corresponding year/population values across all three tables. An entry in your final dictionary should look like this:

'Albania': {'1950': 1228, '1955': 1393, '1960': 1624, '1965': 1884, '1970': 2157, '1975': 2402, '1980': 2672, '1985': 2957, '1990': 3245, '1995': 3159, '2000': 3159, '2005': 3025, '2010': 2987, '2015': 3030, '2020': 3075, '2025': 3105, '2030': 3103, '2035': 3063, '2040': 2994, '2045': 2913, '2050': 2825},

One way to do it is:

1. First extract the header
2. From your header only store values that are numeric (you can use isnumeric() function, recall that we only care about year values and we don't want to store columns represented by %
3. Once you have all the relevant column names (column that correspond to a year value), you can go over every row of the table
    - Create a dictionary key with the country name
    - Collect and add values corresponding to one of your column names to the dictionary

```
import pandas as pd
import numpy as np

df = pd.read_html(wiki_content,header=0)[:-1]

for j in df:
  for i in j.columns:
    if i.find('%')==0:
      a = j.pop(i)
      del a
all_dict_list = []
for i in df:

  table_dict_list = i.to_dict(orient='records')
  all_dict_list.append(table_dict_list)
```

```
# adding all the countries data in the final dictionary
country_dict = {}
for i in all_dict_list:

  for j in i:

    country_column = 'Country (or dependent territory)'
    country_name = str(j['Country (or dependent territory)'])

    if country_name in country_dict.keys():

      for k in j:

        if not(k==country_column):
          country_dict[country_name].update({k:j.get(k)})

    else:
      country_dict[country_name] = {}

      for k in j:

        if not(k==country_column):
          country_dict[country_name].update({k:j.get(k)})
```

1.5 Create a dataframe from your dictionary (10 pts)

Now that all tables are stored in a dictionary, we can convert the dictionary into a pandas dataframe.

1. Remove the "World" row
2. Replace 'NaN' values with 0
3. Display the first 8 rows

```
# Your code here

# making a data frame from the country dictionary

population_dataframe_final = pd.DataFrame.from_dict(country_dict,orient='index')
```

```
# Your code here

# removing world column and replacing NaN values with 0
population_dataframe_final = population_dataframe_final.drop('World',axis=0,errors='ignore')
population_dataframe_final = population_dataframe_final.fillna(0)

population_dataframe_final[0:8]
```

	1950	1955	1960	1965	1970	1975	1980	1985	1990	1995	2000	2
Afghanistan	8151	8892	9830	10998	12431	14133	15045	13120	13569	19446	22462	26
Albania	1228	1393	1624	1884	2157	2402	2672	2957	3245	3159	3159	3
Algeria	8893	9842	10910	11964	13932	16141	18807	22009	25191	28322	30639	32
American Samoa	20	20	21	25	28	30	33	39	48	54	58	
Andorra	7	7	9	14	20	27	34	45	53	64	66	
Angola	4118	4424	4798	5135	5606	6051	7206	8390	9486	11000	12683	14
Anguilla	6	6	6	6	7	7	7	7	9	10	12	

Part 2. Exploring the data

Now let's look at the data at hand.

2.1 Plotting population (10 pts)

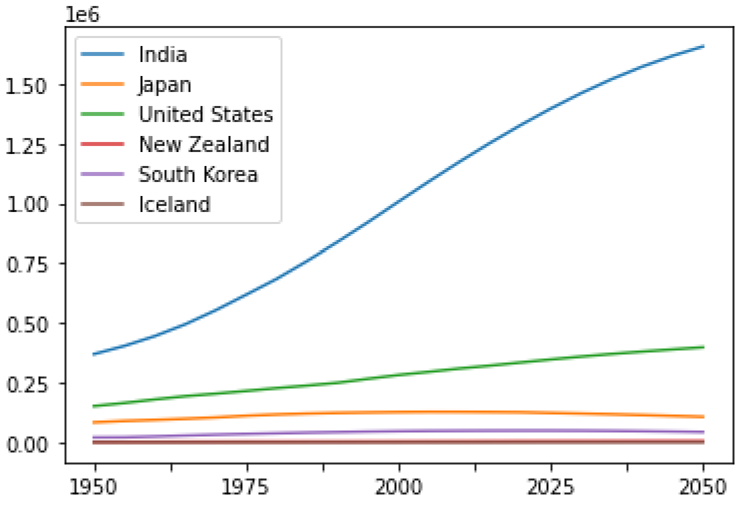
Pick 6 countries of your choice and plot their population growth.

```
# Your code here
countries_of_choice = ['India','Japan','United States','New Zealand','South Korea','Iceland']
country_index = []

for j in range(len(countries_of_choice)):
  for i in range(population_dataframe_final.index.size):
    if (population_dataframe_final.index[i]==countries_of_choice[j]):
      country_index.append(i)

population_dataframe_final.iloc[country_index].T.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f30a92d0c90>
```



2.2 Find 6 most populous countries (15 pts)

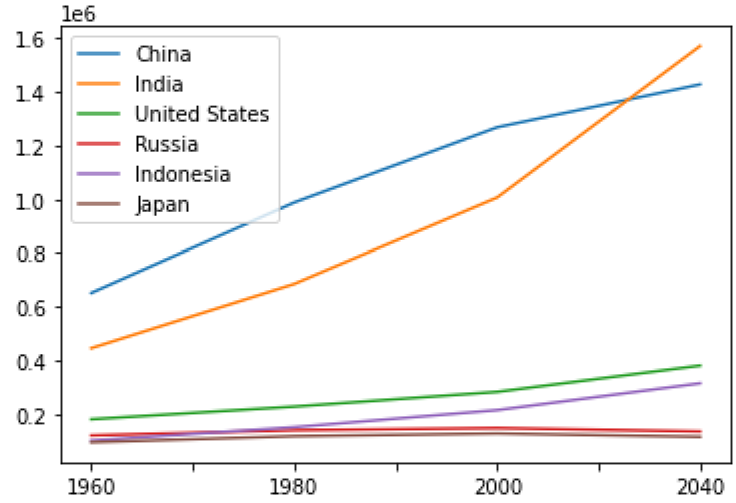
Find 6 most popoulous coutntries in 1960. Then find their population in 1980, 2000, 2020, and 2040. plot their population changes. Are there countries that consistently remain the most populous throught the years?

```
question_2_2 = population_dataframe_final.sort_values('1960',ascending=False)[0:6][['1960','19
question_2_2
```

	1960	1980	2000	2040
China	651340	987822	1268302	1428383
India	445394	684888	1006301	1571716
United States	179980	227225	282163	380220
Russia	119632	139039	147054	134496
Indonesia	100146	150322	214091	314085
Japan	94092	116808	126776	114449

```
question_2_2.T.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f30a7ef8e90>
```



China has remained the most populous throughout the years, however India has been projected to be more populous than China in 2040.

2.3 Declining population (10 pts)

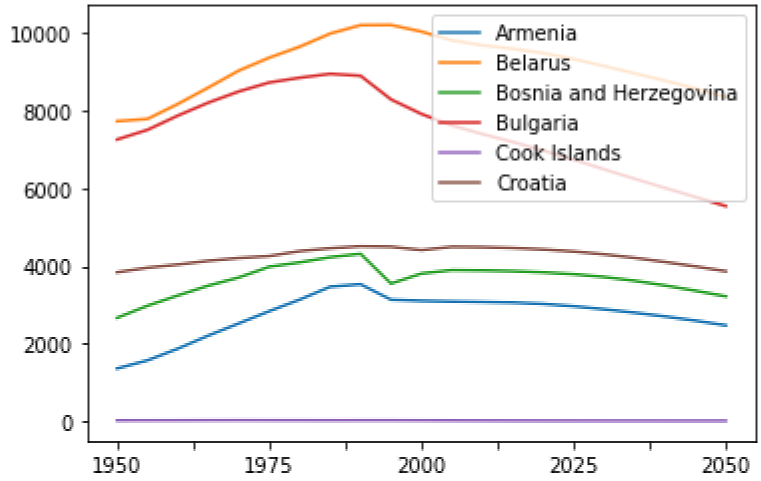
Check the population estimates between the years of 2020 and 2050 and find 6 countries that are experiencing decline in their population. Plot their population changes from 1960 - 2050.

```
# Your code here
question_2_3 = (population_dataframe_final.loc[population_dataframe_final['2020']>population_da
question_2_3_country_list =  question_2_3.index.to_list()
```

```
question_2_3_index_list = []
for i in question_2_3_country_list:
  question_2_3_index_list.append(population_dataframe_final.index.get_loc(i))
```

```
population_dataframe_final.iloc[question_2_3_index_list].T.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f30a7f4a0d0>
```


Armenia
Belarus
Bosnia and Herzegovina
Bulgaria
Cook Islands
Croatia