DV0101EN-1-1-1-Introduction-to-Matplotlib-and-Line-Plots-py-v2.0

January 8, 2019

Introduction to Matplotlib and Line Plots

0.1 Introduction

The aim of these labs is to introduce you to data visualization with Python as concrete and as consistent as possible. Speaking of consistency, because there is no *best* data visualization library avaiblable for Python - up to creating these labs - we have to introduce different libraries and show their benefits when we are discussing new visualization concepts. Doing so, we hope to make students well-rounded with visualization libraries and concepts so that they are able to judge and decide on the best visualitzation technique and tool for a given problem *and* audience.

Please make sure that you have completed the prerequisites for this course, namely **Python for Data Science**, which is part of this specialization.

Note: The majority of the plots and visualizations will be generated using data stored in *pandas* dataframes. Therefore, in this lab, we provide a brief crash course on *pandas*. However, if you are interested in learning more about the *pandas* library, detailed description and explanation of how to use it and how to clean, munge, and process data stored in a *pandas* dataframe are provided in our course **Data Analysis with Python**, which is also part of this specialization.

0.2 Table of Contents

1 Exploring Datasets with *pandas*

pandas is an essential data analysis toolkit for Python. From their website: >*pandas* is a Python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, **real world** data analysis in Python.

The course heavily relies on *pandas* for data wrangling, analysis, and visualization. We encourage you to spend some time and familizare yourself with the *pandas* API Reference: http://pandas.pydata.org/pandas-docs/stable/api.html.

1.1 The Dataset: Immigration to Canada from 1980 to 2013

Dataset Source: International migration flows to and from selected countries - The 2015 revision.

The dataset contains annual data on the flows of international immigrants as recorded by the countries of destination. The data presents both inflows and outflows according to the place of birth, citizenship or place of previous / next residence both for foreigners and nationals. The current version presents data pertaining to 45 countries.

In this lab, we will focus on the Canadian immigration data.

For sake of simplicity, Canada's immigration data has been extracted and uploaded to one of IBM servers. You can fetch the data from here.

1.2 *pandas* Basics

The first thing we'll do is import two key data analysis modules: *pandas* and **Numpy**.

```
In [1]: import numpy as np  # useful for many scientific computing in Python
        import pandas as pd # primary data structure library
```

Let's download and import our primary Canadian Immigration dataset using *pandas* `read_excel()` method. Normally, before we can do that, we would need to download a module which *pandas* requires to read in excel files. This module is **xlrd**. For your convenience, we have pre-installed this module, so you would not have to worry about that. Otherwise, you would need to run the following line of code to install the **xlrd** module:

```
!conda install -c anaconda xlrd --yes
```

Now we are ready to read in our data.

```
In [2]: df_can = pd.read_excel('https://ibm.box.com/shared/static/lw190pt9zpy5bd1ptyg2aw15awomz9
                               sheet_name='Canada by Citizenship',
                               skiprows=range(20),
                               skipfooter=2)

        print ('Data read into a pandas dataframe!')
```

```
Data read into a pandas dataframe!
```

Let's view the top 5 rows of the dataset using the `head()` function.

```
In [3]: df_can.head()
        # tip: You can specify the number of rows you'd like to see as follows: df_can.head(10)
```

```
Out[3]:          Type    Coverage          OdName  AREA AreaName   REG  \
        0  Immigrants  Foreigners     Afghanistan   935     Asia  5501
        1  Immigrants  Foreigners         Albania   908   Europe   925
        2  Immigrants  Foreigners         Algeria   903   Africa   912
        3  Immigrants  Foreigners  American Samoa   909  Oceania   957
```

```
4  Immigrants  Foreigners         Andorra   908   Europe   925

           RegName  DEV             DevName  1980  ...   2004  2005  2006  \
0    Southern Asia  902  Developing regions    16  ...   2978  3436  3009
1  Southern Europe  901   Developed regions     1  ...   1450  1223   856
2  Northern Africa  902  Developing regions    80  ...   3616  3626  4807
3        Polynesia  902  Developing regions     0  ...      0     0     1
4  Southern Europe  901   Developed regions     0  ...      0     0     1

   2007  2008  2009  2010  2011  2012  2013
0  2652  2111  1746  1758  2203  2635  2004
1   702   560   716   561   539   620   603
2  3623  4005  5393  4752  4325  3774  4331
3     0     0     0     0     0     0     0
4     1     0     0     0     0     1     1

[5 rows x 43 columns]
```

We can also veiw the bottom 5 rows of the dataset using the `tail()` function.

```
In [4]: df_can.tail()
```

```
Out[4]:           Type    Coverage          OdName  AREA AreaName  REG  \
190  Immigrants  Foreigners        Viet Nam   935     Asia  920
191  Immigrants  Foreigners  Western Sahara   903   Africa  912
192  Immigrants  Foreigners           Yemen   935     Asia  922
193  Immigrants  Foreigners          Zambia   903   Africa  910
194  Immigrants  Foreigners        Zimbabwe   903   Africa  910

                RegName  DEV             DevName  1980  ...   2004  2005  \
190  South-Eastern Asia  902  Developing regions  1191  ...   1816  1852
191     Northern Africa  902  Developing regions     0  ...      0     0
192        Western Asia  902  Developing regions     1  ...    124   161
193      Eastern Africa  902  Developing regions    11  ...     56    91
194      Eastern Africa  902  Developing regions    72  ...   1450   615

     2006  2007  2008  2009  2010  2011  2012  2013
190  3153  2574  1784  2171  1942  1723  1731  2112
191     1     0     0     0     0     0     0     0
192   140   122   133   128   211   160   174   217
193    77    71    64    60   102    69    46    59
194   454   663   611   508   494   434   437   407

[5 rows x 43 columns]
```

When analyzing a dataset, it's always a good idea to start by getting basic information about your dataframe. We can do this by using the `info()` method.

```
In [5]: df_can.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 195 entries, 0 to 194
Data columns (total 43 columns):
Type        195 non-null object
Coverage    195 non-null object
OdName      195 non-null object
AREA        195 non-null int64
AreaName    195 non-null object
REG         195 non-null int64
RegName     195 non-null object
DEV         195 non-null int64
DevName     195 non-null object
1980        195 non-null int64
1981        195 non-null int64
1982        195 non-null int64
1983        195 non-null int64
1984        195 non-null int64
1985        195 non-null int64
1986        195 non-null int64
1987        195 non-null int64
1988        195 non-null int64
1989        195 non-null int64
1990        195 non-null int64
1991        195 non-null int64
1992        195 non-null int64
1993        195 non-null int64
1994        195 non-null int64
1995        195 non-null int64
1996        195 non-null int64
1997        195 non-null int64
1998        195 non-null int64
1999        195 non-null int64
2000        195 non-null int64
2001        195 non-null int64
2002        195 non-null int64
2003        195 non-null int64
2004        195 non-null int64
2005        195 non-null int64
2006        195 non-null int64
2007        195 non-null int64
2008        195 non-null int64
2009        195 non-null int64
2010        195 non-null int64
2011        195 non-null int64
2012        195 non-null int64
2013        195 non-null int64
dtypes: int64(37), object(6)
memory usage: 65.6+ KB
```

To get the list of column headers we can call upon the dataframe's `.columns` parameter.

```
In [6]: df_can.columns.values
```

```
Out[6]: array(['Type', 'Coverage', 'OdName', 'AREA', 'AreaName', 'REG', 'RegName',
               'DEV', 'DevName', 1980, 1981, 1982, 1983, 1984, 1985, 1986, 1987,
               1988, 1989, 1990, 1991, 1992, 1993, 1994, 1995, 1996, 1997, 1998,
               1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009,
               2010, 2011, 2012, 2013], dtype=object)
```

Similarly, to get the list of indicies we use the `.index` parameter.

```
In [7]: df_can.index.values
```

```
Out[7]: array([  0,   1,   2,   3,   4,   5,   6,   7,   8,   9,  10,  11,  12,
                13,  14,  15,  16,  17,  18,  19,  20,  21,  22,  23,  24,  25,
                26,  27,  28,  29,  30,  31,  32,  33,  34,  35,  36,  37,  38,
                39,  40,  41,  42,  43,  44,  45,  46,  47,  48,  49,  50,  51,
                52,  53,  54,  55,  56,  57,  58,  59,  60,  61,  62,  63,  64,
                65,  66,  67,  68,  69,  70,  71,  72,  73,  74,  75,  76,  77,
                78,  79,  80,  81,  82,  83,  84,  85,  86,  87,  88,  89,  90,
                91,  92,  93,  94,  95,  96,  97,  98,  99, 100, 101, 102, 103,
               104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116,
               117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129,
               130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142,
               143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155,
               156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168,
               169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181,
               182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194])
```

Note: The default type of index and columns is NOT list.

```
In [8]: print(type(df_can.columns))
        print(type(df_can.index))
```

```
<class 'pandas.core.indexes.base.Index'>
<class 'pandas.core.indexes.range.RangeIndex'>
```

To get the index and columns as lists, we can use the `tolist()` method.

```
In [9]: df_can.columns.tolist()
        df_can.index.tolist()

        print (type(df_can.columns.tolist()))
        print (type(df_can.index.tolist()))
```

```
<class 'list'>
<class 'list'>
```

To view the dimensions of the dataframe, we use the `.shape` parameter.

```
In [10]: # size of dataframe (rows, columns)
         df_can.shape

Out[10]: (195, 43)
```

Note: The main types stored in *pandas* objects are *float*, *int*, *bool*, *datetime64[ns]* and *datetime64[ns, tz] (in >= 0.17.0)*, *timedelta[ns]*, *category (in >= 0.15.0)*, and *object* (string). In addition these dtypes have item sizes, e.g. int64 and int32.

Let's clean the data set to remove a few unnecessary columns. We can use *pandas* `drop()` method as follows:

```
In [11]: # in pandas axis=0 represents rows (default) and axis=1 represents columns.
         df_can.drop(['AREA','REG','DEV','Type','Coverage'], axis=1, inplace=True)
         df_can.head(2)

Out[11]:         OdName AreaName          RegName             DevName  1980  1981  \
         0  Afghanistan     Asia    Southern Asia  Developing regions    16    39
         1      Albania   Europe  Southern Europe   Developed regions     1     0

            1982  1983  1984  1985  ...   2004  2005  2006  2007  2008  2009  2010  \
         0    39    47    71   340  ...   2978  3436  3009  2652  2111  1746  1758
         1     0     0     0     0  ...   1450  1223   856   702   560   716   561

            2011  2012  2013
         0  2203  2635  2004
         1   539   620   603

         [2 rows x 38 columns]
```

Let's rename the columns so that they make sense. We can use `rename()` method by passing in a dictionary of old and new names as follows:

```
In [12]: df_can.rename(columns={'OdName':'Country', 'AreaName':'Continent', 'RegName':'Region'},
         df_can.columns

Out[12]: Index([  'Country', 'Continent',    'Region',   'DevName',        1980,
                       1981,        1982,        1983,        1984,        1985,
                       1986,        1987,        1988,        1989,        1990,
                       1991,        1992,        1993,        1994,        1995,
                       1996,        1997,        1998,        1999,        2000,
                       2001,        2002,        2003,        2004,        2005,
                       2006,        2007,        2008,        2009,        2010,
                       2011,        2012,        2013],
               dtype='object')
```

We will also add a 'Total' column that sums up the total immigrants by country over the entire period 1980 - 2013, as follows:

```
In [13]: df_can['Total'] = df_can.sum(axis=1)
```

We can check to see how many null objects we have in the dataset as follows:

```
In [14]: df_can.isnull().sum()
```

```
Out[14]: Country      0
         Continent    0
         Region       0
         DevName      0
         1980         0
         1981         0
         1982         0
         1983         0
         1984         0
         1985         0
         1986         0
         1987         0
         1988         0
         1989         0
         1990         0
         1991         0
         1992         0
         1993         0
         1994         0
         1995         0
         1996         0
         1997         0
         1998         0
         1999         0
         2000         0
         2001         0
         2002         0
         2003         0
         2004         0
         2005         0
         2006         0
         2007         0
         2008         0
         2009         0
         2010         0
         2011         0
         2012         0
         2013         0
         Total        0
         dtype: int64
```

Finally, let's view a quick summary of each column in our dataframe using the `describe()` method.

```
In [15]: df_can.describe()
```

Out[15]:

	1980	1981	1982	1983	1984
count	195.000000	195.000000	195.000000	195.000000	195.000000
mean	508.394872	566.989744	534.723077	387.435897	376.497436
std	1949.588546	2152.643752	1866.997511	1204.333597	1198.246371
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	0.000000
50%	13.000000	10.000000	11.000000	12.000000	13.000000
75%	251.500000	295.500000	275.000000	173.000000	181.000000
max	22045.000000	24796.000000	20620.000000	10015.000000	10170.000000

	1985	1986	1987	1988	1989
count	195.000000	195.000000	195.000000	195.000000	195.000000
mean	358.861538	441.271795	691.133333	714.389744	843.241026
std	1079.309600	1225.576630	2109.205607	2443.606788	2555.048874
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.500000	0.500000	1.000000	1.000000
50%	17.000000	18.000000	26.000000	34.000000	44.000000
75%	197.000000	254.000000	434.000000	409.000000	508.500000
max	9564.000000	9470.000000	21337.000000	27359.000000	23795.000000

	...	2005	2006	2007	2008
count	...	195.000000	195.000000	195.000000	195.000000
mean	...	1320.292308	1266.958974	1191.820513	1246.394872
std	...	4425.957828	3926.717747	3443.542409	3694.573544
min	...	0.000000	0.000000	0.000000	0.000000
25%	...	28.500000	25.000000	31.000000	31.000000
50%	...	210.000000	218.000000	198.000000	205.000000
75%	...	832.000000	842.000000	899.000000	934.500000
max	...	42584.000000	33848.000000	28742.000000	30037.000000

	2009	2010	2011	2012	2013
count	195.000000	195.000000	195.000000	195.000000	195.000000
mean	1275.733333	1420.287179	1262.533333	1313.958974	1320.702564
std	3829.630424	4462.946328	4030.084313	4247.555161	4237.951988
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	36.000000	40.500000	37.500000	42.500000	45.000000
50%	214.000000	211.000000	179.000000	233.000000	213.000000
75%	888.000000	932.000000	772.000000	783.000000	796.000000
max	29622.000000	38617.000000	36765.000000	34315.000000	34129.000000

	Total
count	195.000000
mean	32867.451282

```
std     91785.498686
min         1.000000
25%       952.000000
50%      5018.000000
75%     22239.500000
max    691904.000000

[8 rows x 35 columns]
```

1.3 *pandas* Intermediate: Indexing and Selection (slicing)

1.3.1 Select Column

There are two ways to filter on a column name:

Method 1: Quick and easy, but only works if the column name does NOT have spaces or special characters.

```
df.column_name
    (returns series)
```

Method 2: More robust, and can filter on multiple columns.

```
df['column']
    (returns series)

df[['column 1', 'column 2']]
    (returns dataframe)
```

Example: Let's try filtering on the list of countries ('Country').

```
In [16]: df_can.Country  # returns a series

Out[16]: 0                                      Afghanistan
         1                                          Albania
         2                                          Algeria
         3                                   American Samoa
         4                                          Andorra
         5                                           Angola
         6                              Antigua and Barbuda
         7                                        Argentina
         8                                          Armenia
         9                                        Australia
         10                                         Austria
         11                                      Azerbaijan
         12                                         Bahamas
         13                                         Bahrain
```

```
14                                          Bangladesh
15                                            Barbados
16                                             Belarus
17                                             Belgium
18                                              Belize
19                                               Benin
20                                              Bhutan
21                        Bolivia (Plurinational State of)
22                                  Bosnia and Herzegovina
23                                            Botswana
24                                              Brazil
25                                     Brunei Darussalam
26                                              Bulgaria
27                                          Burkina Faso
28                                               Burundi
29                                            Cabo Verde
                               ...
165                                           Suriname
166                                          Swaziland
167                                             Sweden
168                                        Switzerland
169                                 Syrian Arab Republic
170                                          Tajikistan
171                                            Thailand
172            The former Yugoslav Republic of Macedonia
173                                                Togo
174                                               Tonga
175                                 Trinidad and Tobago
176                                             Tunisia
177                                              Turkey
178                                        Turkmenistan
179                                              Tuvalu
180                                              Uganda
181                                             Ukraine
182                                 United Arab Emirates
183    United Kingdom of Great Britain and Northern I...
184                            United Republic of Tanzania
185                               United States of America
186                                             Uruguay
187                                          Uzbekistan
188                                             Vanuatu
189                   Venezuela (Bolivarian Republic of)
190                                             Viet Nam
191                                      Western Sahara
192                                               Yemen
193                                              Zambia
194                                            Zimbabwe
Name: Country, Length: 195, dtype: object
```

Let's try filtering on the list of countries ('OdName') and the data for years: 1980 - 1985.

```
In [17]: df_can[['Country', 1980, 1981, 1982, 1983, 1984, 1985]] # returns a dataframe
         # notice that 'Country' is string, and the years are integers.
         # for the sake of consistency, we will convert all column names to string later on.
```

Out[17]:

	Country	1980	1981	1982
0	Afghanistan	16	39	39
1	Albania	1	0	0
2	Algeria	80	67	71
3	American Samoa	0	1	0
4	Andorra	0	0	0
5	Angola	1	3	6
6	Antigua and Barbuda	0	0	0
7	Argentina	368	426	626
8	Armenia	0	0	0
9	Australia	702	639	484
10	Austria	234	238	201
11	Azerbaijan	0	0	0
12	Bahamas	26	23	38
13	Bahrain	0	2	1
14	Bangladesh	83	84	86
15	Barbados	372	376	299
16	Belarus	0	0	0
17	Belgium	511	540	519
18	Belize	16	27	13
19	Benin	2	5	4
20	Bhutan	0	0	0
21	Bolivia (Plurinational State of)	44	52	42
22	Bosnia and Herzegovina	0	0	0
23	Botswana	10	1	3
24	Brazil	211	220	192
25	Brunei Darussalam	79	6	8
26	Bulgaria	24	20	12
27	Burkina Faso	2	1	3
28	Burundi	0	0	0
29	Cabo Verde	1	1	2
..	...	...	...	...
165	Suriname	15	10	21
166	Swaziland	4	1	1
167	Sweden	281	308	222
168	Switzerland	806	811	634
169	Syrian Arab Republic	315	419	409
170	Tajikistan	0	0	0
171	Thailand	56	53	113
172	The former Yugoslav Republic of Macedonia	0	0	0
173	Togo	5	5	2
174	Tonga	2	4	7

175	Trinidad and Tobago	958	947	972
176	Tunisia	58	51	55
177	Turkey	481	874	706
178	Turkmenistan	0	0	0
179	Tuvalu	0	1	0
180	Uganda	13	16	17
181	Ukraine	0	0	0
182	United Arab Emirates	0	2	2
183	United Kingdom of Great Britain and Northern I...	22045	24796	20620
184	United Republic of Tanzania	635	832	621
185	United States of America	9378	10030	9074
186	Uruguay	128	132	146
187	Uzbekistan	0	0	0
188	Vanuatu	0	0	0
189	Venezuela (Bolivarian Republic of)	103	117	174
190	Viet Nam	1191	1829	2162
191	Western Sahara	0	0	0
192	Yemen	1	2	1
193	Zambia	11	17	11
194	Zimbabwe	72	114	102

	1983	1984	1985
0	47	71	340
1	0	0	0
2	69	63	44
3	0	0	0
4	0	0	0
5	6	4	3
6	0	42	52
7	241	237	196
8	0	0	0
9	317	317	319
10	117	127	165
11	0	0	0
12	12	21	28
13	1	1	3
14	81	98	92
15	244	265	285
16	0	0	0
17	297	183	181
18	21	37	26
19	3	4	3
20	0	1	0
21	49	38	44
22	0	0	0
23	3	7	4
24	139	145	130
25	2	2	4

```
26      33     11    24
27       2      3     2
28       0      1     2
29       0     11     1
..     ...    ...   ...
165     12      5    16
166      0     10     7
167    176    128   158
168    370    326   314
169    269    264   385
170      0      0     0
171     65     82    66
172      0      0     0
173      3      6     5
174      1      2     5
175    766    606   699
176     46     51    57
177    280    338   202
178      0      0     0
179      0      1     0
180     38     32    29
181      0      0     0
182      1      2     0
183  10015  10170  9564
184    474    473   460
185   7100   6661  6543
186    105     90    92
187      0      0     0
188      0      0     0
189    124    142   165
190   3404   7583  5907
191      0      0     0
192      6      0    18
193      7     16     9
194     44     32    29

[195 rows x 7 columns]
```

1.3.2 Select Row

There are main 3 ways to select rows:

```
df.loc[label]
    #filters by the labels of the index/column
df.iloc[index]
    #filters by the positions of the index/column
```

Before we proceed, notice that the defaul index of the dataset is a numeric range from 0 to 194. This makes it very difficult to do a query by a specific country. For example to search for data on

Japan, we need to know the corressponding index value.

This can be fixed very easily by setting the 'Country' column as the index using `set_index()` method.

```
In [18]: df_can.set_index('Country', inplace=True)
         # tip: The opposite of set is reset. So to reset the index, we can use df_can.reset_ind

In [19]: df_can.head(8)
```

Out[19]:

Country	Continent	Region	DevName	1980	1981	1982	1983	1984	1985
Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340
Albania	Europe	Southern Europe	Developed regions	1	0	0	0	0	0
Algeria	Africa	Northern Africa	Developing regions	80	67	71	69	63	44
American Samoa	Oceania	Polynesia	Developing regions	0	1	0	0	0	0
Andorra	Europe	Southern Europe	Developed regions	0	0	0	0	0	0
Angola	Africa	Middle Africa	Developing regions	1	3	6	6	4	3
Antigua and Barbuda	Latin America and the Caribbean	Caribbean	Developing regions	0	0	0	0	42	52
Argentina	Latin America and the Caribbean	South America	Developing regions	368	426	626	241	237	196

Country	1986	...	2005	2006	2007	2008	2009	2010	2011
Afghanistan	496	...	3436	3009	2652	2111	1746	1758	2203
Albania	1	...	1223	856	702	560	716	561	539
Algeria	69	...	3626	4807	3623	4005	5393	4752	4325
American Samoa	0	...	0	1	0	0	0	0	0
Andorra	2	...	0	1	1	0	0	0	0
Angola	5	...	295	184	106	76	62	61	39
Antigua and Barbuda	51	...	24	32	15	32	38	27	37
Argentina	213	...	1153	847	620	540	467	459	278

Country	2012	2013	Total
Afghanistan	2635	2004	58639
Albania	620	603	15699
Algeria	3774	4331	69439
American Samoa	0	0	6

```
Andorra                 1     1     15
Angola                 70    45   2113
Antigua and Barbuda    51    25    981
Argentina             263   282  19596

[8 rows x 38 columns]
```

```
In [20]: # optional: to remove the name of the index
         df_can.index.name = None
```

Example: Let's view the number of immigrants from Japan (row 87) for the following scenarios: 1. The full row data (all columns) 2. For year 2013 3. For years 1980 to 1985

```
In [21]: # 1. the full row data (all columns)
         print(df_can.loc['Japan'])

         # alternate methods
         #print(df_can.iloc[87])
         #print(df_can[df_can.index == 'Japan'].T.squeeze())
```

```
Continent                 Asia
Region            Eastern Asia
DevName      Developed regions
1980                       701
1981                       756
1982                       598
1983                       309
1984                       246
1985                       198
1986                       248
1987                       422
1988                       324
1989                       494
1990                       379
1991                       506
1992                       605
1993                       907
1994                       956
1995                       826
1996                       994
1997                       924
1998                       897
1999                      1083
2000                      1010
2001                      1092
2002                       806
2003                       817
2004                       973
2005                      1067
```

```
2006                    1212
2007                    1250
2008                    1284
2009                    1194
2010                    1168
2011                    1265
2012                    1214
2013                     982
Total                  27707
Name: Japan, dtype: object
```

```
In [22]: # 2. for year 2013
         #print(df_can.loc['Japan', 2013])
         print(df_can.loc['Kenya',2000])

         # alternate method
         #print(df_can.iloc[87, 36]) # year 2013 is the last column, with a positional index of
```

```
495
```

```
In [23]: # 3. for years 1980 to 1985
         print(df_can.loc['Japan', [1980, 1981, 1982, 1983, 1984, 1984]])
         #print(df_can.iloc[87, [3, 4, 5, 6, 7, 8]])
```

```
1980    701
1981    756
1982    598
1983    309
1984    246
1984    246
Name: Japan, dtype: object
```

Column names that are integers (such as the years) might introduce some confusion. For example, when we are referencing the year 2013, one might confuse that when the 2013th positional index.

To avoid this ambuigity, let's convert the column names into strings: '1980' to '2013'.

```
In [24]: df_can.columns = list(map(str, df_can.columns))
         # [print (type(x)) for x in df_can.columns.values] #<-- uncomment to check type of colu
```

Since we converted the years to string, let's declare a variable that will allow us to easily call upon the full range of years:

```
In [25]: # useful for plotting later on
         years = list(map(str, range(1980, 2014)))
         #print(years)
```

1.3.3 Filtering based on a criteria

To filter the dataframe based on a condition, we simply pass the condition as a boolean vector.

For example, Let's filter the dataframe to show the data on Asian countries (AreaName = Asia).

```
In [26]: # 1. create the condition boolean series
         condition = df_can['Continent'] == 'Asia'
         print (condition)
```

```
Afghanistan                                 True
Albania                                    False
Algeria                                    False
American Samoa                             False
Andorra                                    False
Angola                                     False
Antigua and Barbuda                        False
Argentina                                  False
Armenia                                     True
Australia                                  False
Austria                                    False
Azerbaijan                                  True
Bahamas                                    False
Bahrain                                     True
Bangladesh                                  True
Barbados                                   False
Belarus                                    False
Belgium                                    False
Belize                                     False
Benin                                      False
Bhutan                                      True
Bolivia (Plurinational State of)           False
Bosnia and Herzegovina                     False
Botswana                                   False
Brazil                                     False
Brunei Darussalam                           True
Bulgaria                                   False
Burkina Faso                               False
Burundi                                    False
Cabo Verde                                 False
                                           ...
Suriname                                   False
Swaziland                                  False
Sweden                                     False
Switzerland                                False
Syrian Arab Republic                        True
Tajikistan                                  True
Thailand                                    True
The former Yugoslav Republic of Macedonia  False
Togo                                       False
```

```
Tonga                                                 False
Trinidad and Tobago                                   False
Tunisia                                               False
Turkey                                                 True
Turkmenistan                                           True
Tuvalu                                                False
Uganda                                                False
Ukraine                                               False
United Arab Emirates                                   True
United Kingdom of Great Britain and Northern Ireland  False
United Republic of Tanzania                           False
United States of America                              False
Uruguay                                               False
Uzbekistan                                             True
Vanuatu                                               False
Venezuela (Bolivarian Republic of)                    False
Viet Nam                                               True
Western Sahara                                        False
Yemen                                                  True
Zambia                                                False
Zimbabwe                                              False
Name: Continent, Length: 195, dtype: bool
```

In [27]:
```
# 2. pass this condition into the dataFrame
df_can[condition]
```

Out[27]:

	Continent	Region	\
Afghanistan	Asia	Southern Asia	
Armenia	Asia	Western Asia	
Azerbaijan	Asia	Western Asia	
Bahrain	Asia	Western Asia	
Bangladesh	Asia	Southern Asia	
Bhutan	Asia	Southern Asia	
Brunei Darussalam	Asia	South-Eastern Asia	
Cambodia	Asia	South-Eastern Asia	
China	Asia	Eastern Asia	
China, Hong Kong Special Administrative Region	Asia	Eastern Asia	
China, Macao Special Administrative Region	Asia	Eastern Asia	
Cyprus	Asia	Western Asia	
Democratic People's Republic of Korea	Asia	Eastern Asia	
Georgia	Asia	Western Asia	
India	Asia	Southern Asia	
Indonesia	Asia	South-Eastern Asia	
Iran (Islamic Republic of)	Asia	Southern Asia	
Iraq	Asia	Western Asia	
Israel	Asia	Western Asia	
Japan	Asia	Eastern Asia	

Jordan	Asia	Western Asia	
Kazakhstan	Asia	Central Asia	
Kuwait	Asia	Western Asia	
Kyrgyzstan	Asia	Central Asia	
Lao People's Democratic Republic	Asia	South-Eastern Asia	
Lebanon	Asia	Western Asia	
Malaysia	Asia	South-Eastern Asia	
Maldives	Asia	Southern Asia	
Mongolia	Asia	Eastern Asia	
Myanmar	Asia	South-Eastern Asia	
Nepal	Asia	Southern Asia	
Oman	Asia	Western Asia	
Pakistan	Asia	Southern Asia	
Philippines	Asia	South-Eastern Asia	
Qatar	Asia	Western Asia	
Republic of Korea	Asia	Eastern Asia	
Saudi Arabia	Asia	Western Asia	
Singapore	Asia	South-Eastern Asia	
Sri Lanka	Asia	Southern Asia	
State of Palestine	Asia	Western Asia	
Syrian Arab Republic	Asia	Western Asia	
Tajikistan	Asia	Central Asia	
Thailand	Asia	South-Eastern Asia	
Turkey	Asia	Western Asia	
Turkmenistan	Asia	Central Asia	
United Arab Emirates	Asia	Western Asia	
Uzbekistan	Asia	Central Asia	
Viet Nam	Asia	South-Eastern Asia	
Yemen	Asia	Western Asia	

	DevName	1980	\
Afghanistan	Developing regions	16	
Armenia	Developing regions	0	
Azerbaijan	Developing regions	0	
Bahrain	Developing regions	0	
Bangladesh	Developing regions	83	
Bhutan	Developing regions	0	
Brunei Darussalam	Developing regions	79	
Cambodia	Developing regions	12	
China	Developing regions	5123	
China, Hong Kong Special Administrative Region	Developing regions	0	
China, Macao Special Administrative Region	Developing regions	0	
Cyprus	Developing regions	132	
Democratic People's Republic of Korea	Developing regions	1	
Georgia	Developing regions	0	
India	Developing regions	8880	
Indonesia	Developing regions	186	
Iran (Islamic Republic of)	Developing regions	1172	

Iraq	Developing regions	262
Israel	Developing regions	1403
Japan	Developed regions	701
Jordan	Developing regions	177
Kazakhstan	Developing regions	0
Kuwait	Developing regions	1
Kyrgyzstan	Developing regions	0
Lao People's Democratic Republic	Developing regions	11
Lebanon	Developing regions	1409
Malaysia	Developing regions	786
Maldives	Developing regions	0
Mongolia	Developing regions	0
Myanmar	Developing regions	80
Nepal	Developing regions	1
Oman	Developing regions	0
Pakistan	Developing regions	978
Philippines	Developing regions	6051
Qatar	Developing regions	0
Republic of Korea	Developing regions	1011
Saudi Arabia	Developing regions	0
Singapore	Developing regions	241
Sri Lanka	Developing regions	185
State of Palestine	Developing regions	0
Syrian Arab Republic	Developing regions	315
Tajikistan	Developing regions	0
Thailand	Developing regions	56
Turkey	Developing regions	481
Turkmenistan	Developing regions	0
United Arab Emirates	Developing regions	0
Uzbekistan	Developing regions	0
Viet Nam	Developing regions	1191
Yemen	Developing regions	1

	1981	1982	1983	1984	1985	\
Afghanistan	39	39	47	71	340	
Armenia	0	0	0	0	0	
Azerbaijan	0	0	0	0	0	
Bahrain	2	1	1	1	3	
Bangladesh	84	86	81	98	92	
Bhutan	0	0	0	1	0	
Brunei Darussalam	6	8	2	2	4	
Cambodia	19	26	33	10	7	
China	6682	3308	1863	1527	1816	
China, Hong Kong Special Administrative Region	0	0	0	0	0	
China, Macao Special Administrative Region	0	0	0	0	0	
Cyprus	128	84	46	46	43	
Democratic People's Republic of Korea	1	3	1	4	3	
Georgia	0	0	0	0	0	

India	8670	8147	7338	5704	4211
Indonesia	178	252	115	123	100
Iran (Islamic Republic of)	1429	1822	1592	1977	1648
Iraq	245	260	380	428	231
Israel	1711	1334	541	446	680
Japan	756	598	309	246	198
Jordan	160	155	113	102	179
Kazakhstan	0	0	0	0	0
Kuwait	0	8	2	1	4
Kyrgyzstan	0	0	0	0	0
Lao People's Democratic Republic	6	16	16	7	17
Lebanon	1119	1159	789	1253	1683
Malaysia	816	813	448	384	374
Maldives	0	0	1	0	0
Mongolia	0	0	0	0	0
Myanmar	62	46	31	41	23
Nepal	1	6	1	2	4
Oman	0	0	8	0	0
Pakistan	972	1201	900	668	514
Philippines	5921	5249	4562	3801	3150
Qatar	0	0	0	0	0
Republic of Korea	1456	1572	1081	847	962
Saudi Arabia	0	1	4	1	2
Singapore	301	337	169	128	139
Sri Lanka	371	290	197	1086	845
State of Palestine	0	0	0	0	0
Syrian Arab Republic	419	409	269	264	385
Tajikistan	0	0	0	0	0
Thailand	53	113	65	82	66
Turkey	874	706	280	338	202
Turkmenistan	0	0	0	0	0
United Arab Emirates	2	2	1	2	0
Uzbekistan	0	0	0	0	0
Viet Nam	1829	2162	3404	7583	5907
Yemen	2	1	6	0	18

	1986	...	2005	2006	\
Afghanistan	496	...	3436	3009	
Armenia	0	...	224	218	
Azerbaijan	0	...	359	236	
Bahrain	0	...	12	12	
Bangladesh	486	...	4171	4014	
Bhutan	0	...	5	10	
Brunei Darussalam	12	...	4	5	
Cambodia	8	...	370	529	
China	1960	...	42584	33518	
China, Hong Kong Special Administrative Region	0	...	729	712	
China, Macao Special Administrative Region	0	...	21	32	

Cyprus	48	...	7	9	
Democratic People's Republic of Korea	0	...	14	10	
Georgia	0	...	114	125	
India	7150	...	36210	33848	
Indonesia	127	...	632	613	
Iran (Islamic Republic of)	1794	...	5837	7480	
Iraq	265	...	2226	1788	
Israel	1212	...	2446	2625	
Japan	248	...	1067	1212	
Jordan	181	...	1940	1827	
Kazakhstan	0	...	506	408	
Kuwait	4	...	66	35	
Kyrgyzstan	0	...	173	161	
Lao People's Democratic Republic	21	...	42	74	
Lebanon	2576	...	3709	3802	
Malaysia	425	...	593	580	
Maldives	0	...	0	0	
Mongolia	0	...	59	64	
Myanmar	18	...	210	953	
Nepal	13	...	607	540	
Oman	0	...	14	18	
Pakistan	691	...	14314	13127	
Philippines	4166	...	18139	18400	
Qatar	1	...	11	2	
Republic of Korea	1208	...	5832	6215	
Saudi Arabia	5	...	198	252	
Singapore	205	...	392	298	
Sri Lanka	1838	...	4930	4714	
State of Palestine	0	...	453	627	
Syrian Arab Republic	493	...	1458	1145	
Tajikistan	0	...	85	46	
Thailand	78	...	575	500	
Turkey	257	...	2065	1638	
Turkmenistan	0	...	40	26	
United Arab Emirates	5	...	31	42	
Uzbekistan	0	...	330	262	
Viet Nam	2741	...	1852	3153	
Yemen	7	...	161	140	

	2007	2008	2009	2010	\
Afghanistan	2652	2111	1746	1758	
Armenia	198	205	267	252	
Azerbaijan	203	125	165	209	
Bahrain	22	9	35	28	
Bangladesh	2897	2939	2104	4721	
Bhutan	7	36	865	1464	
Brunei Darussalam	11	10	5	12	
Cambodia	460	354	203	200	

China	27642	30037	29622	30391
China, Hong Kong Special Administrative Region	674	897	657	623
China, Macao Special Administrative Region	16	12	21	21
Cyprus	4	7	6	18
Democratic People's Republic of Korea	7	19	11	45
Georgia	132	112	128	126
India	28742	28261	29456	34235
Indonesia	657	661	504	712
Iran (Islamic Republic of)	6974	6475	6580	7477
Iraq	2406	3543	5450	5941
Israel	2401	2562	2316	2755
Japan	1250	1284	1194	1168
Jordan	1421	1581	1235	1831
Kazakhstan	436	394	431	377
Kuwait	62	53	68	67
Kyrgyzstan	135	168	173	157
Lao People's Democratic Republic	53	32	39	54
Lebanon	3467	3566	3077	3432
Malaysia	600	658	640	802
Maldives	2	1	7	4
Mongolia	82	59	118	169
Myanmar	1887	975	1153	556
Nepal	511	581	561	1392
Oman	16	10	7	14
Pakistan	10124	8994	7217	6811
Philippines	19837	24887	28573	38617
Qatar	5	9	6	18
Republic of Korea	5920	7294	5874	5537
Saudi Arabia	188	249	246	330
Singapore	690	734	366	805
Sri Lanka	4123	4756	4547	4422
State of Palestine	441	481	400	654
Syrian Arab Republic	1056	919	917	1039
Tajikistan	44	15	50	52
Thailand	487	519	512	499
Turkey	1463	1122	1238	1492
Turkmenistan	37	13	20	30
United Arab Emirates	37	33	37	86
Uzbekistan	284	215	288	289
Viet Nam	2574	1784	2171	1942
Yemen	122	133	128	211

	2011	2012	2013	Total
Afghanistan	2203	2635	2004	58639
Armenia	236	258	207	3310
Azerbaijan	138	161	57	2649
Bahrain	21	39	32	475
Bangladesh	2694	2640	3789	65568

Bhutan	1879	1075	487	5876
Brunei Darussalam	6	3	6	600
Cambodia	196	233	288	6538
China	28502	33024	34129	659962
China, Hong Kong Special Administrative Region	591	728	774	9327
China, Macao Special Administrative Region	13	33	29	284
Cyprus	6	12	16	1126
Democratic People's Republic of Korea	97	66	17	388
Georgia	139	147	125	2068
India	27509	30933	33087	691904
Indonesia	390	395	387	13150
Iran (Islamic Republic of)	7479	7534	11291	175923
Iraq	6196	4041	4918	69789
Israel	1970	2134	1945	66508
Japan	1265	1214	982	27707
Jordan	1635	1206	1255	35406
Kazakhstan	381	462	348	8490
Kuwait	58	73	48	2025
Kyrgyzstan	159	278	123	2353
Lao People's Democratic Republic	22	25	15	1089
Lebanon	3072	1614	2172	115359
Malaysia	409	358	204	24417
Maldives	3	1	1	30
Mongolia	103	68	99	952
Myanmar	368	193	262	9245
Nepal	1129	1185	1308	10222
Oman	10	13	11	224
Pakistan	7468	11227	12603	241600
Philippines	36765	34315	29544	511391
Qatar	3	14	6	157
Republic of Korea	4588	5316	4509	142581
Saudi Arabia	278	286	267	3425
Singapore	219	146	141	14579
Sri Lanka	3309	3338	2394	148358
State of Palestine	555	533	462	6512
Syrian Arab Republic	1005	650	1009	31485
Tajikistan	47	34	39	503
Thailand	396	296	400	9174
Turkey	1257	1068	729	31781
Turkmenistan	20	20	14	310
United Arab Emirates	60	54	46	836
Uzbekistan	162	235	167	3368
Viet Nam	1723	1731	2112	97146
Yemen	160	174	217	2985

[49 rows x 38 columns]

In [28]: *# we can pass mutliple criteria in the same line.*

```
# let's filter for AreaNAme = Asia and RegName = Southern Asia
df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')]

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
# don't forget to enclose the two conditions in parentheses
```

Out[28]:

	Continent	Region	DevName	1980	1981	1982	1983	1984	1985	1986	...	2005	2006	2007	2008	2009	2010	2011	2012	2013	Total
Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340	496	...	3436	3009	2652	2111	1746	1758	2203	2635	2004	58639
Bangladesh	Asia	Southern Asia	Developing regions	83	84	86	81	98	92	486	...	4171	4014	2897	2939	2104	4721	2694	2640	3789	65568
Bhutan	Asia	Southern Asia	Developing regions	0	0	0	0	1	0	0	...	5	10	7	36	865	1464	1879	1075	487	5876
India	Asia	Southern Asia	Developing regions	8880	8670	8147	7338	5704	4211	7150	...	36210	33848	28742	28261	29456	34235	27509	30933	33087	691904
Iran (Islamic Republic of)	Asia	Southern Asia	Developing regions	1172	1429	1822	1592	1977	1648	1794	...	5837	7480	6974	6475	6580	7477	7479	7534	11291	175923
Maldives	Asia	Southern Asia	Developing regions	0	0	0	1	0	0	0	...	0	0	2	1	7	4	3	1	1	30
Nepal	Asia	Southern Asia	Developing regions	1	1	6	1	2	4	13	...	607	540	511	581	561	1392	1129	1185	1308	10222
Pakistan	Asia	Southern Asia	Developing regions	978	972	1201	900	668	514	691	...	14314	13127	10124	8994	7217	6811	7468	11227	12603	241600
Sri Lanka	Asia	Southern Asia	Developing regions	185	371	290	197	1086	845	1838	...	4930	4714	4123	4756	4547	4422	3309	3338	2004	58639

```
Sri Lanka                    2394  148358

[9 rows x 38 columns]
```

```
In [29]: df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')]
```

Out[29]:

	Continent	Region	DevName	1980	1981	1982	1983	1984	1985	1986	...	2005	2006	2007	2008	2009	2010	2011	2012	2013	Total
Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340	496	...	3436	3009	2652	2111	1746	1758	2203	2635	2004	58639
Bangladesh	Asia	Southern Asia	Developing regions	83	84	86	81	98	92	486	...	4171	4014	2897	2939	2104	4721	2694	2640	3789	65568
Bhutan	Asia	Southern Asia	Developing regions	0	0	0	0	1	0	0	...	5	10	7	36	865	1464	1879	1075	487	5876
India	Asia	Southern Asia	Developing regions	8880	8670	8147	7338	5704	4211	7150	...	36210	33848	28742	28261	29456	34235	27509	30933	33087	691904
Iran (Islamic Republic of)	Asia	Southern Asia	Developing regions	1172	1429	1822	1592	1977	1648	1794	...	5837	7480	6974	6475	6580	7477	7479	7534	11291	175923
Maldives	Asia	Southern Asia	Developing regions	0	0	0	1	0	0	0	...	0	0	2	1	7	4	3	1	1	30
Nepal	Asia	Southern Asia	Developing regions	1	1	6	1	2	4	13	...	607	540	511	581	561	1392	1129	1185	1308	10222
Pakistan	Asia	Southern Asia	Developing regions	978	972	1201	900	668	514	691	...	14314	13127	10124	8994	7217	6811	7468	11227	12603	241600
Sri Lanka	Asia	Southern Asia	Developing regions	185	371	290	197	1086	845	1838	...	4930	4714	4123	4756	4547	4422	3309	3338	2394	148358

```
Sri Lanka                    2394  148358

[9 rows x 38 columns]
```

Before we proceed: let's review the changes we have made to our dataframe.

```
In [30]: print ('data dimensions:', df_can.shape)
         print(df_can.columns)
         df_can.head(2)
```

```
data dimensions: (195, 38)
Index(['Continent', 'Region', 'DevName', '1980', '1981', '1982', '1983',
       '1984', '1985', '1986', '1987', '1988', '1989', '1990', '1991', '1992',
       '1993', '1994', '1995', '1996', '1997', '1998', '1999', '2000', '2001',
       '2002', '2003', '2004', '2005', '2006', '2007', '2008', '2009', '2010',
       '2011', '2012', '2013', 'Total'],
      dtype='object')
```

```
Out[30]:             Continent           Region             DevName  1980  1981  1982  \
         Afghanistan      Asia    Southern Asia  Developing regions    16    39    39
         Albania        Europe  Southern Europe   Developed regions     1     0     0

                      1983  1984  1985  1986  ...    2005  2006  2007  2008  2009  \
         Afghanistan    47    71   340   496  ...    3436  3009  2652  2111  1746
         Albania         0     0     0     1  ...    1223   856   702   560   716

                      2010  2011  2012  2013  Total
         Afghanistan  1758  2203  2635  2004  58639
         Albania       561   539   620   603  15699

         [2 rows x 38 columns]
```

---

2 Visualizing Data using Matplotlib

2.1 Matplotlib: Standard Python Visualization Library

The primary plotting library we will explore in the course is Matplotlib. As mentioned on their website: >Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shell, the jupyter notebook, web application servers, and four graphical user interface toolkits.

If you are aspiring to create impactful visualization with python, Matplotlib is an essential tool to have at your disposal.

2.1.1 Matplotlib.Pyplot

One of the core aspects of Matplotlib is `matplotlib.pyplot`. It is Matplotlib's scripting layer which we studied in details in the videos about Matplotlib. Recall that it is a collection of command style functions that make Matplotlib work like MATLAB. Each `pyplot` function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. In this lab, we will work with the scripting layer to learn how to generate line plots. In future labs, we will get to work with the Artist layer as well to experiment first hand how it differs from the scripting layer.

Let's start by importing `Matplotlib` and `Matplotlib.pyplot` as follows:

```
In [31]: # we are using the inline backend
         %matplotlib inline

         import matplotlib as mpl
         import matplotlib.pyplot as plt
```

*optional: check if Matplotlib is loaded.

```
In [32]: print ('Matplotlib version: ', mpl.__version__) # >= 2.0.0
```

```
Matplotlib version:  2.2.2
```

*optional: apply a style to Matplotlib.

```
In [33]: print(plt.style.available)
         mpl.style.use(['ggplot']) # optional: for ggplot-like style
```

```
['Solarize_Light2', '_classic_test', 'bmh', 'classic', 'dark_background', 'fast', 'fivethirtyeig
```

2.1.2 Plotting in *pandas*

Fortunately, pandas has a built-in implementation of Matplotlib that we can use. Plotting in *pandas* is as simple as appending a `.plot()` method to a series or dataframe.

Documentation: - Plotting with Series - Plotting with Dataframes

3 Line Pots (Series/Dataframe)

What is a line plot and why use it?

A line chart or line plot is a type of plot which displays information as a series of data points called 'markers' connected by straight line segments. It is a basic type of chart common in many fields. Use line plot when you have a continuous data set. These are best suited for trend-based visualizations of data over a period of time.

Let's start with a case study:

In 2010, Haiti suffered a catastrophic magnitude 7.0 earthquake. The quake caused widespread devastation and loss of life and aout three million people were affected by this natural disaster. As

part of Canada's humanitarian effort, the Government of Canada stepped up its effort in accepting refugees from Haiti. We can quickly visualize this effort using a `Line` plot:

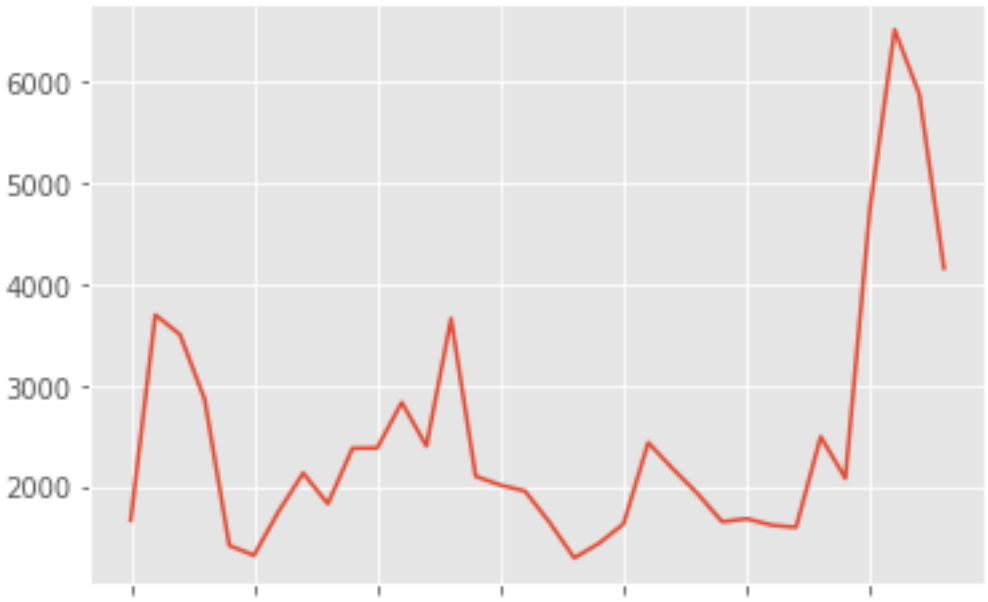
Question: Plot a line graph of immigration from Haiti using `df.plot()`.

First, we will extract the data series for Haiti.

```
In [34]: haiti = df_can.loc['Haiti', years] # passing in years 1980 - 2013 to exclude the 'total
         haiti.head()

Out[34]: 1980    1666
         1981    3692
         1982    3498
         1983    2860
         1984    1418
         Name: Haiti, dtype: object
```

Next, we will plot a line plot by appending `.plot()` to the `haiti` dataframe.

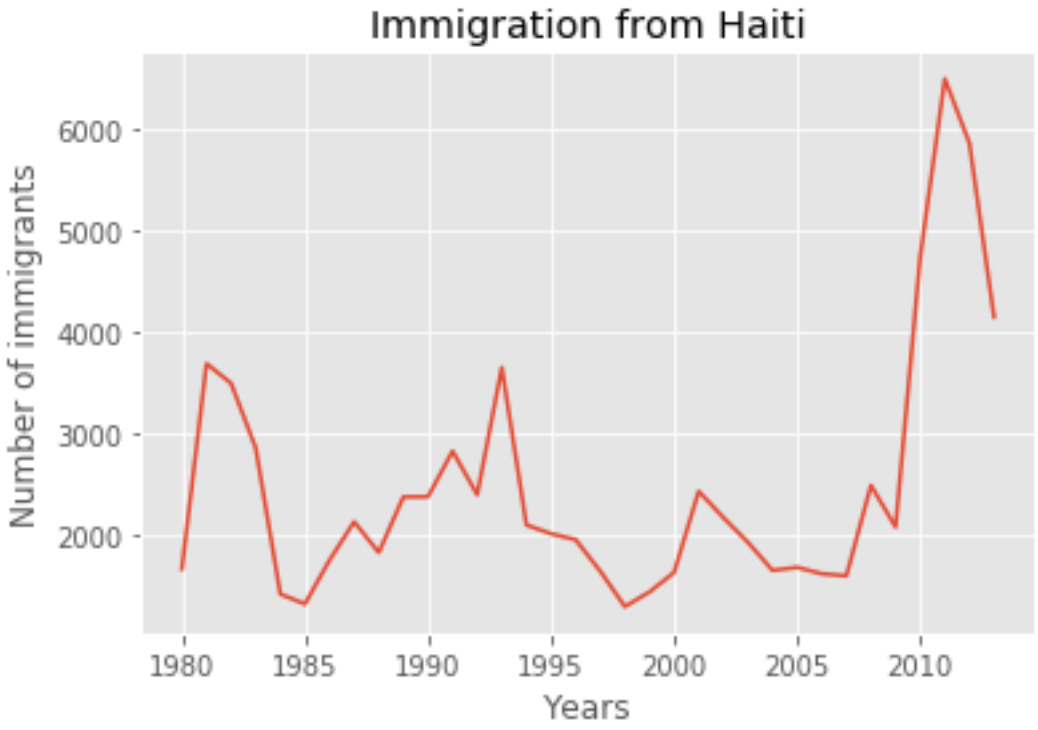
```
In [35]: haiti.plot()

Out[35]: <matplotlib.axes._subplots.AxesSubplot at 0x7f153bba0fd0>
```



pandas automatically populated the x-axis with the index values (years), and the y-axis with the column values (population). However, notice how the years were not displayed because they are of type *string*. Therefore, let's change the type of the index values to *integer* for plotting.

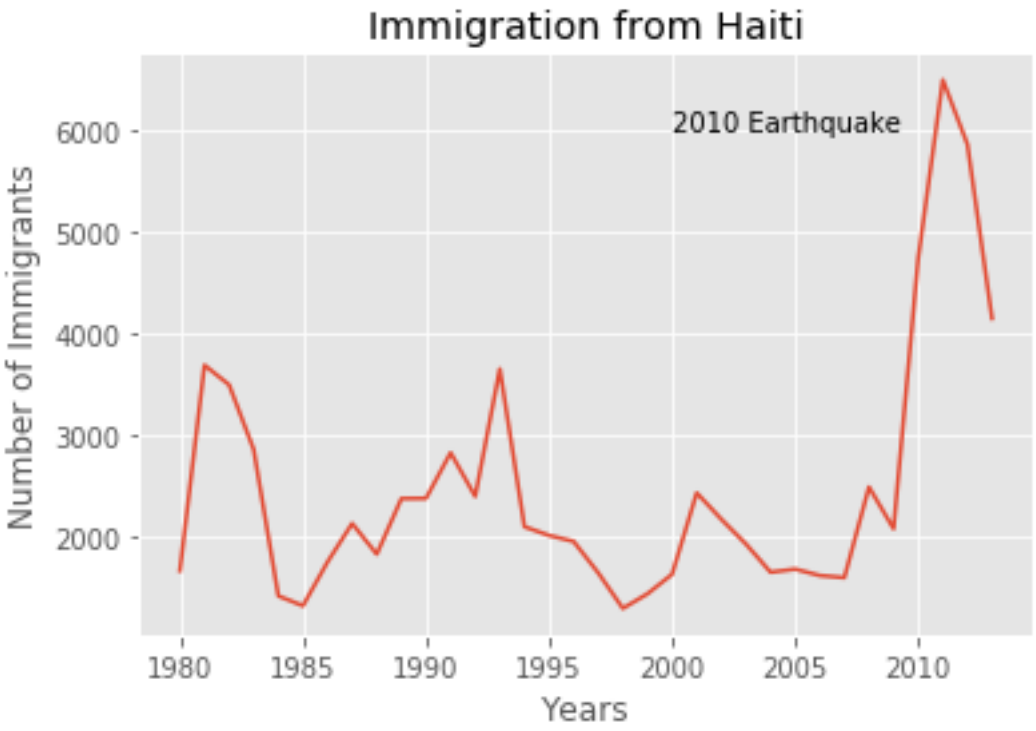
Also, let's label the x and y axis using `plt.title()`, `plt.ylabel()`, and `plt.xlabel()` as follows:

```
In [36]: haiti.index = haiti.index.map(int) # let's change the index values of Haiti to type int
         haiti.plot(kind='line')

         plt.title('Immigration from Haiti')
         plt.ylabel('Number of immigrants')
         plt.xlabel('Years')

         plt.show() # need this line to show the updates made to the figure
```



We can clearly notice how number of immigrants from Haiti spiked up from 2010 as Canada stepped up its efforts to accept refugees from Haiti. Let's annotate this spike in the plot by using the `plt.text()` method.

```
In [37]: haiti.plot(kind='line')

         plt.title('Immigration from Haiti')
         plt.ylabel('Number of Immigrants')
         plt.xlabel('Years')

         # annotate the 2010 Earthquake.
         # syntax: plt.text(x, y, label)
         plt.text(2000, 6000, '2010 Earthquake') # see note below

         plt.show()
```



With just a few lines of code, you were able to quickly identify and visualize the spike in immigration!

Quick note on x and y values in `plt.text(x, y, label)`:

```
 Since the x-axis (years) is type 'integer', we specified x as a year. The y axis (number of imm

    plt.text(2000, 6000, '2010 Earthquake') # years stored as type int

If the years were stored as type 'string', we would need to specify x as the index position of t

    plt.text(20, 6000, '2010 Earthquake') # years stored as type int

We will cover advanced annotation methods in later modules.
```

We can easily add more countries to line plot to make meaningful comparisons immigration from different countries.

Question: Let's compare the number of immigrants from India and China from 1980 to 2013.

Step 1: Get the data set for China and India, and display dataframe.

```
In [38]: ### type your answer here
         #china = df_can.loc['China',years]
         #india = df_can.loc['India',years]
         df_CHIN = df_can.loc[['China', 'India'], years]
         df_CHIN.head()
```

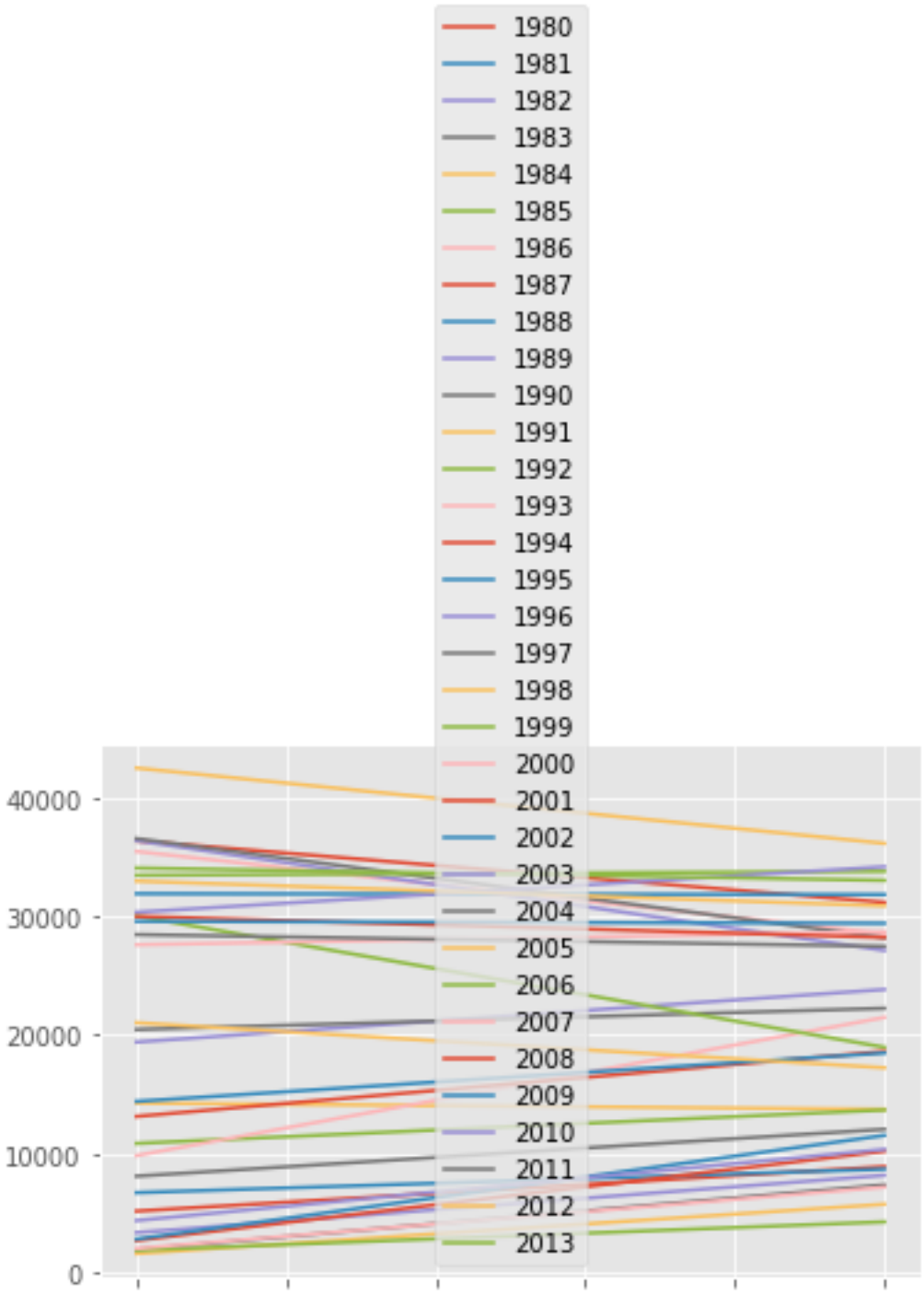
```
Out[38]:         1980  1981  1982  1983  1984  1985  1986   1987   1988   1989  ...    \
         China   5123  6682  3308  1863  1527  1816  1960   2643   2758   4323  ...
         India   8880  8670  8147  7338  5704  4211  7150  10189  11522  10343  ...

                 2004   2005   2006   2007   2008   2009   2010   2011   2012   2013
         China  36619  42584  33518  27642  30037  29622  30391  28502  33024  34129
         India  28235  36210  33848  28742  28261  29456  34235  27509  30933  33087

         [2 rows x 34 columns]
```

Double-click **here** for the solution.

Step 2: Plot graph. We will explicitly specify line plot by passing in `kind` parameter to `plot()`.

```
In [39]: ### type your answer here
         df_CHIN.plot(kind='line')
```

```
Out[39]: <matplotlib.axes._subplots.AxesSubplot at 0x7f153b7e74a8>
```



Double-click **here** for the solution.

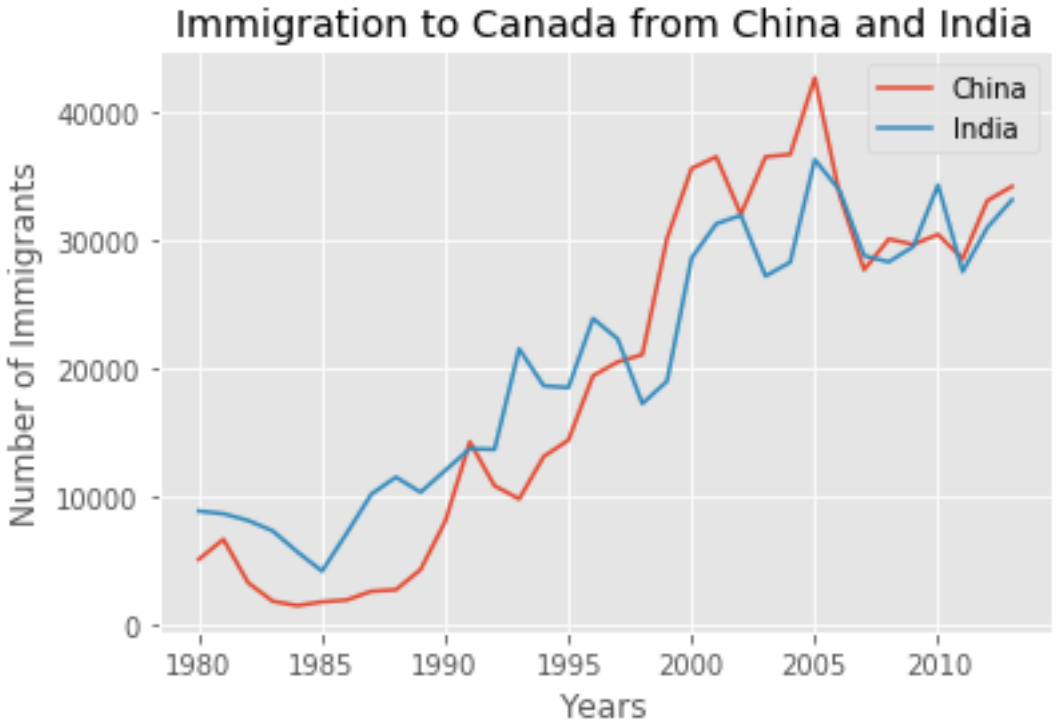
That doesn't look right...

Recall that *pandas* plots the indices on the x-axis and the columns as individual lines on the y-axis. Since `df_CI` is a dataframe with the `country` as the index and `years` as the columns, we must first transpose the dataframe using `transpose()` method to swap the row and columns.

```
In [40]: df_CHIN = df_CHIN[years].transpose()
         df_CHIN.head()

Out[40]:       China  India
         1980   5123   8880
         1981   6682   8670
         1982   3308   8147
         1983   1863   7338
         1984   1527   5704
```

pandas will auomatically graph the two countries on the same graph. Go ahead and plot the new transposed dataframe. Make sure to add a title to the plot and label the axes.

```
In [41]: ### type your answer here
         df_CHIN.index = df_CHIN.index.map(int)
         df_CHIN.plot(kind='line')

         plt.title('Immigration to Canada from China and India')
         plt.ylabel('Number of Immigrants')
         plt.xlabel('Years')
         plt.show()
```



Double-click **here** for the solution.

From the above plot, we can observe that the China and India have very similar immigration trends through the years.

Note: How come we didn't need to transpose Haiti's dataframe before plotting (like we did for df_CI)?

That's because `haiti` is a series as opposed to a dataframe, and has the years as its indices as shown below.

```
print(type(haiti))
print(haiti.head(5))
```

```
class 'pandas.core.series.Series' 1980 1666 1981 3692 1982 3498 1983 2860 1984 1418
Name: Haiti, dtype: int64
```

Line plot is a handy tool to display several dependent variables against one independent variable. However, it is recommended that no more than 5-10 lines on a single graph; any more than that and it becomes difficult to interpret.

Question: Compare the trend of top 5 countries that contributed the most to immigration to Canada.

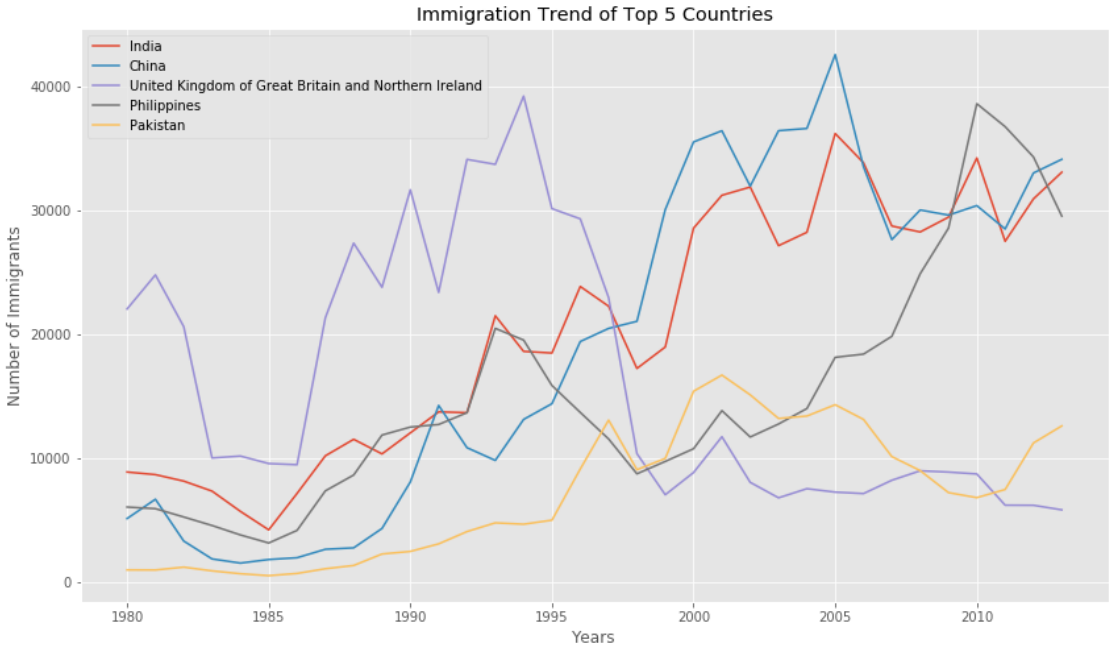
```
In [42]: ### type your answer here

         df_can.sort_values(by='Total', ascending=False, axis=0, inplace=True)
         df_top5 = df_can.head(5)
         df_top5 = df_top5[years].transpose()
         print(df_top5)
         print('\n')
         df_top5.index = df_top5.index.map(int)
         df_top5.plot(kind='line', figsize=(14, 8))
         plt.title('Immigration Trend of Top 5 Countries')
         plt.ylabel('Number of Immigrants')
         plt.xlabel('Years')
         plt.show()
```

```
      India  China  United Kingdom of Great Britain and Northern Ireland  \
1980   8880   5123                                                22045
1981   8670   6682                                                24796
1982   8147   3308                                                20620
1983   7338   1863                                                10015
1984   5704   1527                                                10170
1985   4211   1816                                                 9564
1986   7150   1960                                                 9470
1987  10189   2643                                                21337
1988  11522   2758                                                27359
1989  10343   4323                                                23795
1990  12041   8076                                                31668
1991  13734  14255                                                23380
1992  13673  10846                                                34123
1993  21496   9817                                                33720
1994  18620  13128                                                39231
1995  18489  14398                                                30145
1996  23859  19415                                                29322
```

1997	22268	20475	22965
1998	17241	21049	10367
1999	18974	30069	7045
2000	28572	35529	8840
2001	31223	36434	11728
2002	31889	31961	8046
2003	27155	36439	6797
2004	28235	36619	7533
2005	36210	42584	7258
2006	33848	33518	7140
2007	28742	27642	8216
2008	28261	30037	8979
2009	29456	29622	8876
2010	34235	30391	8724
2011	27509	28502	6204
2012	30933	33024	6195
2013	33087	34129	5827

	Philippines	Pakistan
1980	6051	978
1981	5921	972
1982	5249	1201
1983	4562	900
1984	3801	668
1985	3150	514
1986	4166	691
1987	7360	1072
1988	8639	1334
1989	11865	2261
1990	12509	2470
1991	12718	3079
1992	13670	4071
1993	20479	4777
1994	19532	4666
1995	15864	4994
1996	13692	9125
1997	11549	13073
1998	8735	9068
1999	9734	9979
2000	10763	15400
2001	13836	16708
2002	11707	15110
2003	12758	13205
2004	14004	13399
2005	18139	14314
2006	18400	13127
2007	19837	10124
2008	24887	8994

```
2009        28573       7217
2010        38617       6811
2011        36765       7468
2012        34315      11227
2013        29544      12603
```



Double-click **here** for the solution.

```
In [43]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [44]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [45]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [46]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [47]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [48]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [49]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [50]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [51]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia
```

```
condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
# don't forget to enclose the two conditions in parentheses
```

In [52]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
# don't forget to enclose the two conditions in parentheses
```

In [53]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
# don't forget to enclose the two conditions in parentheses
```

In [54]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
# don't forget to enclose the two conditions in parentheses
```

In [55]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
# don't forget to enclose the two conditions in parentheses
```

In [56]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
# don't forget to enclose the two conditions in parentheses
```

In [57]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia
```

```
condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
# don't forget to enclose the two conditions in parentheses
```

```
In [58]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [59]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [60]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [61]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [62]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia
         df_can['DevName']=='Developing regions'
         test1 = df_can['DevName']=='Developing regions'
         print(test1)
         df_can[test1]
         #condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
India                                                   True
China                                                   True
```

United Kingdom of Great Britain and Northern Ireland	False
Philippines	True
Pakistan	True
United States of America	False
Iran (Islamic Republic of)	True
Sri Lanka	True
Republic of Korea	True
Poland	False
Lebanon	True
France	False
Jamaica	True
Viet Nam	True
Romania	False
Haiti	True
Guyana	True
Portugal	False
Egypt	True
Morocco	True
Colombia	True
Iraq	True
Algeria	True
Israel	True
Bangladesh	True
Germany	False
Russian Federation	False
Mexico	True
Afghanistan	True
El Salvador	True
	...
Oman	True
Luxembourg	False
Cabo Verde	True
Comoros	True
Swaziland	True
Mozambique	True
Qatar	True
Tonga	True
Lesotho	True
Montenegro	False
Guinea-Bissau	True
Samoa	True
Papua New Guinea	True
Equatorial Guinea	True
Liechtenstein	False
Maldives	True
Monaco	False
Canada	False
Nauru	True

```
Andorra                                                 False
Kiribati                                                 True
Vanuatu                                                  True
Sao Tome and Principe                                    True
Tuvalu                                                   True
American Samoa                                           True
San Marino                                              False
New Caledonia                                            True
Marshall Islands                                         True
Western Sahara                                           True
Palau                                                    True
Name: DevName, Length: 195, dtype: bool
```

Out[62]:

	Continent	\
India	Asia	
China	Asia	
Philippines	Asia	
Pakistan	Asia	
Iran (Islamic Republic of)	Asia	
Sri Lanka	Asia	
Republic of Korea	Asia	
Lebanon	Asia	
Jamaica	Latin America and the Caribbean	
Viet Nam	Asia	
Haiti	Latin America and the Caribbean	
Guyana	Latin America and the Caribbean	
Egypt	Africa	
Morocco	Africa	
Colombia	Latin America and the Caribbean	
Iraq	Asia	
Algeria	Africa	
Israel	Asia	
Bangladesh	Asia	
Mexico	Latin America and the Caribbean	
Afghanistan	Asia	
El Salvador	Latin America and the Caribbean	
Trinidad and Tobago	Latin America and the Caribbean	
Somalia	Africa	
South Africa	Africa	
Nigeria	Africa	
Jordan	Asia	
Ethiopia	Africa	
Peru	Latin America and the Caribbean	
Turkey	Asia	
...	...	
Bahrain	Asia	
Botswana	Africa	

Democratic People's Republic of Korea	Asia
Namibia	Africa
Turkmenistan	Asia
Malawi	Africa
China, Macao Special Administrative Region	Asia
Oman	Asia
Cabo Verde	Africa
Comoros	Africa
Swaziland	Africa
Mozambique	Africa
Qatar	Asia
Tonga	Oceania
Lesotho	Africa
Guinea-Bissau	Africa
Samoa	Oceania
Papua New Guinea	Oceania
Equatorial Guinea	Africa
Maldives	Asia
Nauru	Oceania
Kiribati	Oceania
Vanuatu	Oceania
Sao Tome and Principe	Africa
Tuvalu	Oceania
American Samoa	Oceania
New Caledonia	Oceania
Marshall Islands	Oceania
Western Sahara	Africa
Palau	Oceania

	Region \
India	Southern Asia
China	Eastern Asia
Philippines	South-Eastern Asia
Pakistan	Southern Asia
Iran (Islamic Republic of)	Southern Asia
Sri Lanka	Southern Asia
Republic of Korea	Eastern Asia
Lebanon	Western Asia
Jamaica	Caribbean
Viet Nam	South-Eastern Asia
Haiti	Caribbean
Guyana	South America
Egypt	Northern Africa
Morocco	Northern Africa
Colombia	South America
Iraq	Western Asia
Algeria	Northern Africa
Israel	Western Asia

```
Bangladesh                                          Southern Asia
Mexico                                             Central America
Afghanistan                                         Southern Asia
El Salvador                                        Central America
Trinidad and Tobago                                      Caribbean
Somalia                                             Eastern Africa
South Africa                                       Southern Africa
Nigeria                                             Western Africa
Jordan                                              Western Asia
Ethiopia                                            Eastern Africa
Peru                                               South America
Turkey                                              Western Asia
...                                                          ...
Bahrain                                             Western Asia
Botswana                                           Southern Africa
Democratic People's Republic of Korea               Eastern Asia
Namibia                                            Southern Africa
Turkmenistan                                        Central Asia
Malawi                                              Eastern Africa
China, Macao Special Administrative Region          Eastern Asia
Oman                                                Western Asia
Cabo Verde                                          Western Africa
Comoros                                             Eastern Africa
Swaziland                                          Southern Africa
Mozambique                                          Eastern Africa
Qatar                                               Western Asia
Tonga                                                  Polynesia
Lesotho                                            Southern Africa
Guinea-Bissau                                       Western Africa
Samoa                                                  Polynesia
Papua New Guinea                                       Melanesia
Equatorial Guinea                                   Middle Africa
Maldives                                            Southern Asia
Nauru                                                 Micronesia
Kiribati                                              Micronesia
Vanuatu                                                Melanesia
Sao Tome and Principe                               Middle Africa
Tuvalu                                                 Polynesia
American Samoa                                         Polynesia
New Caledonia                                          Melanesia
Marshall Islands                                      Micronesia
Western Sahara                                     Northern Africa
Palau                                                  Micronesia

                                                        DevName  1980  1981  \
India                                  Developing regions  8880  8670
China                                  Developing regions  5123  6682
Philippines                            Developing regions  6051  5921
```

Pakistan	Developing regions	978	972
Iran (Islamic Republic of)	Developing regions	1172	1429
Sri Lanka	Developing regions	185	371
Republic of Korea	Developing regions	1011	1456
Lebanon	Developing regions	1409	1119
Jamaica	Developing regions	3198	2634
Viet Nam	Developing regions	1191	1829
Haiti	Developing regions	1666	3692
Guyana	Developing regions	2334	2943
Egypt	Developing regions	612	660
Morocco	Developing regions	325	471
Colombia	Developing regions	266	326
Iraq	Developing regions	262	245
Algeria	Developing regions	80	67
Israel	Developing regions	1403	1711
Bangladesh	Developing regions	83	84
Mexico	Developing regions	409	394
Afghanistan	Developing regions	16	39
El Salvador	Developing regions	110	295
Trinidad and Tobago	Developing regions	958	947
Somalia	Developing regions	7	10
South Africa	Developing regions	1026	1118
Nigeria	Developing regions	81	60
Jordan	Developing regions	177	160
Ethiopia	Developing regions	98	59
Peru	Developing regions	317	456
Turkey	Developing regions	481	874
...	...	...	...
Bahrain	Developing regions	0	2
Botswana	Developing regions	10	1
Democratic People's Republic of Korea	Developing regions	1	1
Namibia	Developing regions	0	5
Turkmenistan	Developing regions	0	0
Malawi	Developing regions	5	4
China, Macao Special Administrative Region	Developing regions	0	0
Oman	Developing regions	0	0
Cabo Verde	Developing regions	1	1
Comoros	Developing regions	0	2
Swaziland	Developing regions	4	1
Mozambique	Developing regions	0	0
Qatar	Developing regions	0	0
Tonga	Developing regions	2	4
Lesotho	Developing regions	1	1
Guinea-Bissau	Developing regions	0	0
Samoa	Developing regions	3	7
Papua New Guinea	Developing regions	0	2
Equatorial Guinea	Developing regions	0	0
Maldives	Developing regions	0	0

Nauru	Developing regions	1	0
Kiribati	Developing regions	0	0
Vanuatu	Developing regions	0	0
Sao Tome and Principe	Developing regions	0	0
Tuvalu	Developing regions	0	1
American Samoa	Developing regions	0	1
New Caledonia	Developing regions	0	0
Marshall Islands	Developing regions	0	0
Western Sahara	Developing regions	0	0
Palau	Developing regions	0	0

	1982	1983	1984	1985	1986	\
India	8147	7338	5704	4211	7150	
China	3308	1863	1527	1816	1960	
Philippines	5249	4562	3801	3150	4166	
Pakistan	1201	900	668	514	691	
Iran (Islamic Republic of)	1822	1592	1977	1648	1794	
Sri Lanka	290	197	1086	845	1838	
Republic of Korea	1572	1081	847	962	1208	
Lebanon	1159	789	1253	1683	2576	
Jamaica	2661	2455	2508	2938	4649	
Viet Nam	2162	3404	7583	5907	2741	
Haiti	3498	2860	1418	1321	1753	
Guyana	3575	2650	1932	2299	3942	
Egypt	755	455	447	348	514	
Morocco	447	335	248	328	388	
Colombia	360	244	235	214	257	
Iraq	260	380	428	231	265	
Algeria	71	69	63	44	69	
Israel	1334	541	446	680	1212	
Bangladesh	86	81	98	92	486	
Mexico	491	490	509	425	667	
Afghanistan	39	47	71	340	496	
El Salvador	882	2587	2666	2769	3106	
Trinidad and Tobago	972	766	606	699	955	
Somalia	7	12	14	10	37	
South Africa	781	379	271	310	718	
Nigeria	58	58	78	78	114	
Jordan	155	113	102	179	181	
Ethiopia	54	98	154	157	302	
Peru	401	241	306	328	628	
Turkey	706	280	338	202	257	
...	...	...	...	...	...	
Bahrain	1	1	1	3	0	
Botswana	3	3	7	4	2	
Democratic People's Republic of Korea	3	1	4	3	0	
Namibia	5	3	2	1	1	
Turkmenistan	0	0	0	0	0	

Malawi	6	3	2	0	4
China, Macao Special Administrative Region	0	0	0	0	0
Oman	0	8	0	0	0
Cabo Verde	2	0	11	1	9
Comoros	2	0	0	2	1
Swaziland	1	0	10	7	1
Mozambique	7	2	3	1	3
Qatar	0	0	0	0	1
Tonga	7	1	2	5	7
Lesotho	1	2	7	5	3
Guinea-Bissau	0	0	1	0	0
Samoa	4	1	3	0	3
Papua New Guinea	2	4	2	2	1
Equatorial Guinea	0	0	0	0	1
Maldives	0	1	0	0	0
Nauru	0	0	0	0	0
Kiribati	0	1	0	0	0
Vanuatu	0	0	0	0	0
Sao Tome and Principe	0	0	0	0	0
Tuvalu	0	0	1	0	0
American Samoa	0	0	0	0	0
New Caledonia	0	0	0	0	0
Marshall Islands	0	0	0	0	0
Western Sahara	0	0	0	0	0
Palau	0	0	0	0	0

	...	2005	2006	2007	\
India	...	36210	33848	28742	
China	...	42584	33518	27642	
Philippines	...	18139	18400	19837	
Pakistan	...	14314	13127	10124	
Iran (Islamic Republic of)	...	5837	7480	6974	
Sri Lanka	...	4930	4714	4123	
Republic of Korea	...	5832	6215	5920	
Lebanon	...	3709	3802	3467	
Jamaica	...	1945	1722	2141	
Viet Nam	...	1852	3153	2574	
Haiti	...	1682	1619	1598	
Guyana	...	1215	1286	1277	
Egypt	...	2496	2190	2356	
Morocco	...	2939	3322	4021	
Colombia	...	6424	6535	5357	
Iraq	...	2226	1788	2406	
Algeria	...	3626	4807	3623	
Israel	...	2446	2625	2401	
Bangladesh	...	4171	4014	2897	
Mexico	...	2837	2844	3239	
Afghanistan	...	3436	3009	2652	

El Salvador	...	436	430	929
Trinidad and Tobago	...	857	794	975
Somalia	...	1198	1061	1166
South Africa	...	988	1111	1200
Nigeria	...	2236	2594	2375
Jordan	...	1940	1827	1421
Ethiopia	...	1506	1801	1512
Peru	...	1653	1473	1490
Turkey	...	2065	1638	1463
...	...	...	...	...
Bahrain	...	12	12	22
Botswana	...	7	11	8
Democratic People's Republic of Korea	...	14	10	7
Namibia	...	6	19	13
Turkmenistan	...	40	26	37
Malawi	...	23	15	13
China, Macao Special Administrative Region	...	21	32	16
Oman	...	14	18	16
Cabo Verde	...	5	7	2
Comoros	...	5	7	11
Swaziland	...	7	7	5
Mozambique	...	5	9	5
Qatar	...	11	2	5
Tonga	...	2	0	1
Lesotho	...	4	0	4
Guinea-Bissau	...	7	12	7
Samoa	...	2	0	1
Papua New Guinea	...	1	0	0
Equatorial Guinea	...	3	4	5
Maldives	...	0	0	2
Nauru	...	1	1	0
Kiribati	...	1	0	0
Vanuatu	...	1	1	0
Sao Tome and Principe	...	1	1	0
Tuvalu	...	0	0	1
American Samoa	...	0	1	0
New Caledonia	...	0	0	0
Marshall Islands	...	0	0	2
Western Sahara	...	0	1	0
Palau	...	0	0	1

	2008	2009	2010	2011	2012	\
India	28261	29456	34235	27509	30933	
China	30037	29622	30391	28502	33024	
Philippines	24887	28573	38617	36765	34315	
Pakistan	8994	7217	6811	7468	11227	
Iran (Islamic Republic of)	6475	6580	7477	7479	7534	
Sri Lanka	4756	4547	4422	3309	3338	

Republic of Korea	7294	5874	5537	4588	5316
Lebanon	3566	3077	3432	3072	1614
Jamaica	2334	2456	2321	2059	2182
Viet Nam	1784	2171	1942	1723	1731
Haiti	2491	2080	4744	6503	5868
Guyana	1137	1180	953	804	676
Egypt	3347	3496	5982	4663	5555
Morocco	4226	5532	6242	4399	3878
Colombia	5452	4652	5218	4366	3741
Iraq	3543	5450	5941	6196	4041
Algeria	4005	5393	4752	4325	3774
Israel	2562	2316	2755	1970	2134
Bangladesh	2939	2104	4721	2694	2640
Mexico	2856	3092	3865	3947	4227
Afghanistan	2111	1746	1758	2203	2635
El Salvador	1115	845	787	691	641
Trinidad and Tobago	1002	1134	915	588	586
Somalia	1015	1214	1528	1535	1582
South Africa	1123	1188	1238	959	1243
Nigeria	2109	3156	3906	3103	3443
Jordan	1581	1235	1831	1635	1206
Ethiopia	1613	1289	1865	2163	1864
Peru	1094	1884	1283	886	787
Turkey	1122	1238	1492	1257	1068
...	...	...	...	...	...
Bahrain	9	35	28	21	39
Botswana	28	15	42	53	64
Democratic People's Republic of Korea	19	11	45	97	66
Namibia	26	14	16	23	24
Turkmenistan	13	20	30	20	20
Malawi	5	10	28	16	9
China, Macao Special Administrative Region	12	21	21	13	33
Oman	10	7	14	10	13
Cabo Verde	5	1	3	3	6
Comoros	3	19	22	14	18
Swaziland	6	10	3	13	17
Mozambique	5	12	7	3	4
Qatar	9	6	18	3	14
Tonga	0	5	5	2	3
Lesotho	1	8	7	1	0
Guinea-Bissau	2	0	2	2	4
Samoa	4	2	5	3	2
Papua New Guinea	9	1	3	0	3
Equatorial Guinea	1	5	4	4	4
Maldives	1	7	4	3	1
Nauru	0	1	0	0	0
Kiribati	1	1	0	0	2
Vanuatu	0	1	0	0	0

Sao Tome and Principe	0	1	2	0	0
Tuvalu	0	0	0	1	0
American Samoa	0	0	0	0	0
New Caledonia	1	0	0	0	0
Marshall Islands	0	0	0	0	0
Western Sahara	0	0	0	0	0
Palau	0	0	0	0	0

	2013	Total
India	33087	691904
China	34129	659962
Philippines	29544	511391
Pakistan	12603	241600
Iran (Islamic Republic of)	11291	175923
Sri Lanka	2394	148358
Republic of Korea	4509	142581
Lebanon	2172	115359
Jamaica	2479	106431
Viet Nam	2112	97146
Haiti	4152	84813
Guyana	656	75785
Egypt	4165	72745
Morocco	3261	72267
Colombia	3631	72088
Iraq	4918	69789
Algeria	4331	69439
Israel	1945	66508
Bangladesh	3789	65568
Mexico	3996	58712
Afghanistan	2004	58639
El Salvador	639	54878
Trinidad and Tobago	427	50016
Somalia	2028	42109
South Africa	1240	40568
Nigeria	4172	39732
Jordan	1255	35406
Ethiopia	1606	35136
Peru	682	32652
Turkey	729	31781
...	...	...
Bahrain	32	475
Botswana	76	396
Democratic People's Republic of Korea	17	388
Namibia	83	320
Turkmenistan	14	310
Malawi	10	294
China, Macao Special Administrative Region	29	284
Oman	11	224

```
Cabo Verde                                 2     201
Comoros                                   22     194
Swaziland                                 39     188
Mozambique                                12     160
Qatar                                      6     157
Tonga                                      5     142
Lesotho                                    6     107
Guinea-Bissau                              0      83
Samoa                                      0      78
Papua New Guinea                           3      64
Equatorial Guinea                         11      61
Maldives                                   1      30
Nauru                                      1      18
Kiribati                                   0      15
Vanuatu                                    0      11
Sao Tome and Principe                      1      10
Tuvalu                                     0       8
American Samoa                             0       6
New Caledonia                              2       5
Marshall Islands                           0       2
Western Sahara                             0       2
Palau                                      0       1

[147 rows x 38 columns]
```

```
In [63]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [64]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [65]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [66]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [67]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [68]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [69]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [70]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [71]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [72]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia
```

```
        condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

        # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
        # don't forget to enclose the two conditions in parentheses
```

In [73]:
```
        # we can pass mutliple criteria in the same line.
        # let's filter for AreaNAme = Asia and RegName = Southern Asia

        condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

        # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
        # don't forget to enclose the two conditions in parentheses
```

In [74]:
```
        # we can pass mutliple criteria in the same line.
        # let's filter for AreaNAme = Asia and RegName = Southern Asia

        condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

        # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
        # don't forget to enclose the two conditions in parentheses
```

In [75]:
```
        # we can pass mutliple criteria in the same line.
        # let's filter for AreaNAme = Asia and RegName = Southern Asia

        condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

        # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
        # don't forget to enclose the two conditions in parentheses
```

In [76]:
```
        # we can pass mutliple criteria in the same line.
        # let's filter for AreaNAme = Asia and RegName = Southern Asia

        condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

        # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
        # don't forget to enclose the two conditions in parentheses
```

In [77]:
```
        # we can pass mutliple criteria in the same line.
        # let's filter for AreaNAme = Asia and RegName = Southern Asia

        condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

        # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
        # don't forget to enclose the two conditions in parentheses
```

In [78]:
```
        # we can pass mutliple criteria in the same line.
        # let's filter for AreaNAme = Asia and RegName = Southern Asia
```

```
condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
# don't forget to enclose the two conditions in parentheses
```

```
In [79]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [80]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [81]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [82]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [83]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [84]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')
```

```
# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
# don't forget to enclose the two conditions in parentheses
```

```
In [85]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')]

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [86]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')]

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [87]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')]

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [88]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')]

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [89]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')]

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses
```

```
In [90]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')]
```

```
        # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
        # don't forget to enclose the two conditions in parentheses

In [91]: # we can pass mutliple criteria in the same line.
        # let's filter for AreaNAme = Asia and RegName = Southern Asia

        condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')]

        # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
        # don't forget to enclose the two conditions in parentheses

In [92]: # we can pass mutliple criteria in the same line.
        # let's filter for AreaNAme = Asia and RegName = Southern Asia

        condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')]

        # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
        # don't forget to enclose the two conditions in parentheses

In [93]: # we can pass mutliple criteria in the same line.
        # let's filter for AreaNAme = Asia and RegName = Southern Asia

        condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')]

        # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
        # don't forget to enclose the two conditions in parentheses

In [94]: # we can pass mutliple criteria in the same line.
        # let's filter for AreaNAme = Asia and RegName = Southern Asia

        condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')]

        # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
        # don't forget to enclose the two conditions in parentheses

In [95]: # we can pass mutliple criteria in the same line.
        # let's filter for AreaNAme = Asia and RegName = Southern Asia

        condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')]

        # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
        # don't forget to enclose the two conditions in parentheses

In [96]: # we can pass mutliple criteria in the same line.
        # let's filter for AreaNAme = Asia and RegName = Southern Asia

        condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')]

        # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
        # don't forget to enclose the two conditions in parentheses
```

```
In [97]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [98]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [99]: # we can pass mutliple criteria in the same line.
         # let's filter for AreaNAme = Asia and RegName = Southern Asia

         condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia')

         # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instead
         # don't forget to enclose the two conditions in parentheses

In [100]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses

In [101]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses

In [102]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses

In [103]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia
```

```
condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

```
In [104]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [105]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [106]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [107]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [108]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [109]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia
```

```
condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

```
In [110]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [111]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [112]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [113]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [114]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [115]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'
```

```
# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [116]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [117]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [118]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [119]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [120]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [121]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'
```

```
# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [122]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [123]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [124]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [125]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [126]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [127]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

```
In [128]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses

In [129]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses

In [130]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses

In [131]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses

In [132]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses

In [133]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses

In [134]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia
```

```
condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

```
In [135]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [136]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [137]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [138]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [139]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [140]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia
```

```
condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

```
In [141]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [142]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [143]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [144]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [145]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

          # note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
          # don't forget to enclose the two conditions in parentheses
```

```
In [146]: # we can pass mutliple criteria in the same line.
          # let's filter for AreaNAme = Asia and RegName = Southern Asia

          condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'
```

```
# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [147]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [148]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [149]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [150]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [151]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

In [152]:
```
# we can pass mutliple criteria in the same line.
# let's filter for AreaNAme = Asia and RegName = Southern Asia

condition2 = df_can[(df_can['Continent']=='Asia') & (df_can['Region']=='Southern Asia'

# note: When using 'and' and 'or' operators, pandas requires we use '&' and '|' instea
# don't forget to enclose the two conditions in parentheses
```

3.0.3 Other Plots

Congratulations! you have learned how to wrangle data with python and create a line plot with Matplotlib. There are many other plotting styles available other than the default Line plot, all of which can be accessed by passing `kind` keyword to `plot()`. The full list of available plots are as follows:

- `bar` for vertical bar plots
- `barh` for horizontal bar plots
- `hist` for histogram
- `box` for boxplot
- `kde` or `density` for density plots
- `area` for area plots
- `pie` for pie plots
- `scatter` for scatter plots
- `hexbin` for hexbin plot

3.0.4 Thank you for completing this lab!

This notebook was originally created by Jay Rajasekharan with contributions from Ehsan M. Kermani, and Slobodan Markovic.

This notebook was recently revised by Alex Aklson. I hope you found this lab session interesting. Feel free to contact me if you have any questions!

This notebook is part of a course on **Coursera** called *Data Visualization with Python*. If you accessed this notebook outside the course, you can take this course online by clicking here.


```
In [153]: df_can.head()
```

```
Out[153]:                                                  Continent  \
          India                                                 Asia
          China                                                 Asia
          United Kingdom of Great Britain and Northern Ir...  Europe
          Philippines                                           Asia
          Pakistan                                              Asia

                                                                    Region  \
          India                                              Southern Asia
          China                                               Eastern Asia
          United Kingdom of Great Britain and Northern Ir...  Northern Europe
          Philippines                                      South-Eastern Asia
          Pakistan                                           Southern Asia

                                                                  DevName   1980  \
          India                                          Developing regions   8880
          China                                          Developing regions   5123
          United Kingdom of Great Britain and Northern Ir...  Developed regions  22045
          Philippines                                    Developing regions   6051
```

```
Pakistan                                           Developing regions    978

                                                    1981   1982   1983  \
India                                               8670   8147   7338
China                                               6682   3308   1863
United Kingdom of Great Britain and Northern Ir...  24796  20620  10015
Philippines                                         5921   5249   4562
Pakistan                                            972    1201   900

                                                    1984  1985  1986   ...    \
India                                               5704  4211  7150   ...
China                                               1527  1816  1960   ...
United Kingdom of Great Britain and Northern Ir...  10170 9564  9470   ...
Philippines                                         3801  3150  4166   ...
Pakistan                                            668   514   691    ...

                                                    2005   2006   2007  \
India                                               36210  33848  28742
China                                               42584  33518  27642
United Kingdom of Great Britain and Northern Ir...  7258   7140   8216
Philippines                                         18139  18400  19837
Pakistan                                            14314  13127  10124

                                                    2008   2009   2010  \
India                                               28261  29456  34235
China                                               30037  29622  30391
United Kingdom of Great Britain and Northern Ir...  8979   8876   8724
Philippines                                         24887  28573  38617
Pakistan                                            8994   7217   6811

                                                    2011   2012   2013  \
India                                               27509  30933  33087
China                                               28502  33024  34129
United Kingdom of Great Britain and Northern Ir...  6204   6195   5827
Philippines                                         36765  34315  29544
Pakistan                                            7468   11227  12603

                                                    Total
India                                               691904
China                                               659962
United Kingdom of Great Britain and Northern Ir...  551500
Philippines                                         511391
Pakistan                                            241600

[5 rows x 38 columns]
```

In [154]:
```
test2 = df_can['DevName']=='Developed regions'
```

In [155]:
```
df_can[test2]
```

Out[155]:

	Continent
United Kingdom of Great Britain and Northern Ir...	Europe
United States of America	Northern America
Poland	Europe
France	Europe
Romania	Europe
Portugal	Europe
Germany	Europe
Russian Federation	Europe
Ukraine	Europe
Japan	Asia
Netherlands	Europe
Australia	Oceania
Bulgaria	Europe
Italy	Europe
Bosnia and Herzegovina	Europe
Ireland	Europe
Hungary	Europe
Albania	Europe
Switzerland	Europe
Republic of Moldova	Europe
Greece	Europe
Belgium	Europe
New Zealand	Oceania
Croatia	Europe
Belarus	Europe
Slovakia	Europe
Sweden	Europe
Spain	Europe
Austria	Europe
The former Yugoslav Republic of Macedonia	Europe
Czech Republic	Europe
Denmark	Europe
Finland	Europe
Lithuania	Europe
Norway	Europe
Latvia	Europe
Serbia	Europe
Malta	Europe
Estonia	Europe
Iceland	Europe
Slovenia	Europe
Luxembourg	Europe
Montenegro	Europe
Liechtenstein	Europe
Monaco	Europe
Canada	Northern America
Andorra	Europe

```
San Marino                                                  Europe

                                                                 Region  \
United Kingdom of Great Britain and Northern Ir...           Northern Europe
United States of America                                    Northern America
Poland                                                       Eastern Europe
France                                                       Western Europe
Romania                                                      Eastern Europe
Portugal                                                    Southern Europe
Germany                                                      Western Europe
Russian Federation                                           Eastern Europe
Ukraine                                                      Eastern Europe
Japan                                                          Eastern Asia
Netherlands                                                  Western Europe
Australia                                   Australia and New Zealand
Bulgaria                                                     Eastern Europe
Italy                                                       Southern Europe
Bosnia and Herzegovina                                      Southern Europe
Ireland                                                     Northern Europe
Hungary                                                      Eastern Europe
Albania                                                     Southern Europe
Switzerland                                                  Western Europe
Republic of Moldova                                          Eastern Europe
Greece                                                      Southern Europe
Belgium                                                      Western Europe
New Zealand                                 Australia and New Zealand
Croatia                                                     Southern Europe
Belarus                                                      Eastern Europe
Slovakia                                                     Eastern Europe
Sweden                                                      Northern Europe
Spain                                                       Southern Europe
Austria                                                      Western Europe
The former Yugoslav Republic of Macedonia                   Southern Europe
Czech Republic                                               Eastern Europe
Denmark                                                     Northern Europe
Finland                                                     Northern Europe
Lithuania                                                   Northern Europe
Norway                                                      Northern Europe
Latvia                                                      Northern Europe
Serbia                                                      Southern Europe
Malta                                                       Southern Europe
Estonia                                                     Northern Europe
Iceland                                                     Northern Europe
Slovenia                                                    Southern Europe
Luxembourg                                                   Western Europe
Montenegro                                                  Southern Europe
Liechtenstein                                                Western Europe
Monaco                                                       Western Europe
```

```
Canada                                                 Northern America
Andorra                                                 Southern Europe
San Marino                                              Southern Europe

                                                              DevName   1980  \
United Kingdom of Great Britain and Northern Ir...  Developed regions  22045
United States of America                            Developed regions   9378
Poland                                              Developed regions    863
France                                              Developed regions   1729
Romania                                             Developed regions    375
Portugal                                            Developed regions   4473
Germany                                             Developed regions   1626
Russian Federation                                  Developed regions      0
Ukraine                                             Developed regions      0
Japan                                               Developed regions    701
Netherlands                                         Developed regions   1889
Australia                                           Developed regions    702
Bulgaria                                            Developed regions     24
Italy                                               Developed regions   1820
Bosnia and Herzegovina                              Developed regions      0
Ireland                                             Developed regions    781
Hungary                                             Developed regions    205
Albania                                             Developed regions      1
Switzerland                                         Developed regions    806
Republic of Moldova                                 Developed regions      0
Greece                                              Developed regions   1065
Belgium                                             Developed regions    511
New Zealand                                         Developed regions    602
Croatia                                             Developed regions      0
Belarus                                             Developed regions      0
Slovakia                                            Developed regions      0
Sweden                                              Developed regions    281
Spain                                               Developed regions    211
Austria                                             Developed regions    234
The former Yugoslav Republic of Macedonia           Developed regions      0
Czech Republic                                      Developed regions      0
Denmark                                             Developed regions    272
Finland                                             Developed regions    208
Lithuania                                           Developed regions      1
Norway                                              Developed regions    116
Latvia                                              Developed regions      0
Serbia                                              Developed regions      0
Malta                                               Developed regions    191
Estonia                                             Developed regions      0
Iceland                                             Developed regions     17
Slovenia                                            Developed regions      0
Luxembourg                                          Developed regions     14
Montenegro                                          Developed regions      0
```

Liechtenstein	Developed regions	1
Monaco	Developed regions	0
Canada	Developed regions	0
Andorra	Developed regions	0
San Marino	Developed regions	1

	1981	1982	1983	\
United Kingdom of Great Britain and Northern Ir...	24796	20620	10015	
United States of America	10030	9074	7100	
Poland	2930	5881	4546	
France	2027	2219	1490	
Romania	438	583	543	
Portugal	3486	2432	1433	
Germany	1977	3062	2376	
Russian Federation	0	0	0	
Ukraine	0	0	0	
Japan	756	598	309	
Netherlands	1858	1852	716	
Australia	639	484	317	
Bulgaria	20	12	33	
Italy	2057	1480	820	
Bosnia and Herzegovina	0	0	0	
Ireland	895	707	298	
Hungary	310	397	337	
Albania	0	0	0	
Switzerland	811	634	370	
Republic of Moldova	0	0	0	
Greece	953	897	633	
Belgium	540	519	297	
New Zealand	480	364	140	
Croatia	0	0	0	
Belarus	0	0	0	
Slovakia	0	0	0	
Sweden	308	222	176	
Spain	299	260	133	
Austria	238	201	117	
The former Yugoslav Republic of Macedonia	0	0	0	
Czech Republic	0	0	0	
Denmark	293	299	106	
Finland	205	170	70	
Lithuania	1	0	0	
Norway	77	106	51	
Latvia	0	0	0	
Serbia	0	0	0	
Malta	242	153	64	
Estonia	0	0	0	
Iceland	33	10	9	
Slovenia	0	0	0	

Luxembourg	4	2	5	
Montenegro	0	0	0	
Liechtenstein	4	2	0	
Monaco	0	0	0	
Canada	0	0	0	
Andorra	0	0	0	
San Marino	0	0	0	

	1984	1985	1986	...	\
United Kingdom of Great Britain and Northern Ir...	10170	9564	9470	...	
United States of America	6661	6543	7074	...	
Poland	3588	2819	4808	...	
France	1169	1177	1298	...	
Romania	524	604	656	...	
Portugal	1398	1451	2617	...	
Germany	1610	1441	1233	...	
Russian Federation	0	0	0	...	
Ukraine	0	0	0	...	
Japan	246	198	248	...	
Netherlands	560	510	539	...	
Australia	317	319	356	...	
Bulgaria	11	24	33	...	
Italy	858	667	731	...	
Bosnia and Herzegovina	0	0	0	...	
Ireland	327	287	481	...	
Hungary	310	522	647	...	
Albania	0	0	1	...	
Switzerland	326	314	294	...	
Republic of Moldova	0	0	0	...	
Greece	580	584	547	...	
Belgium	183	181	197	...	
New Zealand	164	148	176	...	
Croatia	0	0	0	...	
Belarus	0	0	0	...	
Slovakia	0	0	0	...	
Sweden	128	158	187	...	
Spain	137	98	121	...	
Austria	127	165	196	...	
The former Yugoslav Republic of Macedonia	0	0	0	...	
Czech Republic	0	0	0	...	
Denmark	93	73	93	...	
Finland	83	69	68	...	
Lithuania	0	1	0	...	
Norway	31	54	56	...	
Latvia	0	0	1	...	
Serbia	0	0	0	...	
Malta	60	68	76	...	
Estonia	0	0	0	...	

Iceland	13	6	11	...
Slovenia	0	0	0	...
Luxembourg	1	3	6	...
Montenegro	0	0	0	...
Liechtenstein	0	3	0	...
Monaco	0	1	0	...
Canada	0	0	0	...
Andorra	0	0	2	...
San Marino	0	0	1	...

	2005	2006	2007	2008	\
United Kingdom of Great Britain and Northern Ir...	7258	7140	8216	8979	
United States of America	8394	9613	9463	10190	
Poland	1405	1263	1235	1267	
France	4429	4002	4290	4532	
Romania	5048	4468	3834	2837	
Portugal	338	424	405	665	
Germany	2226	2767	2449	3833	
Russian Federation	3972	3117	2983	2690	
Ukraine	2270	1973	2218	1937	
Japan	1067	1212	1250	1284	
Netherlands	813	837	615	818	
Australia	909	875	1033	1018	
Bulgaria	1738	1419	1172	994	
Italy	344	325	320	370	
Bosnia and Herzegovina	211	217	209	182	
Ireland	224	283	300	412	
Hungary	516	520	425	383	
Albania	1223	856	702	560	
Switzerland	270	258	308	357	
Republic of Moldova	655	803	1099	1127	
Greece	100	74	110	107	
Belgium	400	302	363	335	
New Zealand	370	318	350	480	
Croatia	128	102	102	131	
Belarus	558	439	568	524	
Slovakia	369	241	215	125	
Sweden	205	139	193	165	
Spain	126	124	137	169	
Austria	122	102	98	91	
The former Yugoslav Republic of Macedonia	285	256	211	191	
Czech Republic	193	146	137	104	
Denmark	62	101	97	108	
Finland	67	51	62	89	
Lithuania	167	113	108	109	
Norway	57	53	73	66	
Latvia	77	69	88	67	
Serbia	0	3	50	259	

Malta	28	24	7	21
Estonia	26	18	34	34
Iceland	10	2	15	13
Slovenia	23	11	29	18
Luxembourg	10	9	9	3
Montenegro	0	0	7	19
Liechtenstein	1	0	0	1
Monaco	1	1	1	0
Canada	0	0	0	0
Andorra	0	1	1	0
San Marino	0	0	0	0

	2009	2010	2011	2012	\
United Kingdom of Great Britain and Northern Ir...	8876	8724	6204	6195	
United States of America	8995	8142	7676	7891	
Poland	1013	795	720	779	
France	5051	4646	4080	6280	
Romania	2076	1922	1776	1588	
Portugal	623	629	528	560	
Germany	3887	2956	2053	1702	
Russian Federation	2931	2288	1963	2079	
Ukraine	2367	3159	2515	2265	
Japan	1194	1168	1265	1214	
Netherlands	786	759	586	504	
Australia	1018	933	851	982	
Bulgaria	784	556	365	451	
Italy	429	434	374	440	
Bosnia and Herzegovina	156	168	140	121	
Ireland	395	547	525	725	
Hungary	312	354	287	300	
Albania	716	561	539	620	
Switzerland	328	285	266	243	
Republic of Moldova	1535	1988	1367	1416	
Greece	119	101	102	146	
Belgium	358	363	332	402	
New Zealand	520	490	375	417	
Croatia	92	91	135	112	
Belarus	454	438	357	277	
Slovakia	128	135	134	136	
Sweden	167	159	134	140	
Spain	195	174	179	262	
Austria	109	124	84	102	
The former Yugoslav Republic of Macedonia	198	188	130	126	
Czech Republic	159	168	128	173	
Denmark	81	92	93	94	
Finland	63	63	72	62	
Lithuania	63	57	52	77	
Norway	75	46	49	53	

Latvia	85	70	103	202
Serbia	365	271	299	395
Malta	18	17	8	9
Estonia	26	28	15	26
Iceland	15	30	38	42
Slovenia	27	12	13	36
Luxembourg	5	8	11	6
Montenegro	10	14	19	26
Liechtenstein	1	0	2	1
Monaco	0	4	1	3
Canada	0	0	1	1
Andorra	0	0	0	1
San Marino	0	1	0	0

	2013	Total
United Kingdom of Great Britain and Northern Ir...	5827	551500
United States of America	8501	241122
Poland	852	139241
France	5623	109091
Romania	1512	93585
Portugal	630	74167
Germany	1217	63912
Russian Federation	2466	63344
Ukraine	2487	51410
Japan	982	27707
Netherlands	395	26884
Australia	1121	23829
Bulgaria	512	23310
Italy	545	22624
Bosnia and Herzegovina	119	21855
Ireland	1015	16964
Hungary	425	16822
Albania	603	15699
Switzerland	201	15678
Republic of Moldova	1231	15381
Greece	298	12410
Belgium	379	10762
New Zealand	415	10386
Croatia	125	8001
Belarus	247	7975
Slovakia	105	5963
Sweden	140	5866
Spain	331	5018
Austria	90	4985
The former Yugoslav Republic of Macedonia	112	4637
Czech Republic	174	4192
Denmark	81	3901
Finland	76	2781

```
Lithuania                                                 145    2414
Norway                                                     59    2327
Latvia                                                    213    2326
Serbia                                                    297    1939
Malta                                                       4    1546
Estonia                                                    45     935
Iceland                                                    72     570
Slovenia                                                   20     523
Luxembourg                                                  7     215
Montenegro                                                  9     105
Liechtenstein                                               0      40
Monaco                                                      1      29
Canada                                                      2      20
Andorra                                                     1      15
San Marino                                                  0       5

[48 rows x 38 columns]
```

In [156]:
```
test2.sort_values(by='Total', ascending=False, axis=0, inplace=True)
df_top10 = test2.head(5)
df_top7 = df_top7[years].transpose()
print(df_top7)
print('\n')
#df_top5.index = df_top5.index.map(int)
#df_top5.plot(kind='line', figsize=(14, 8))
#plt.title('Immigration Trend of Top 5 Countries')
#plt.ylabel('Number of Immigrants')
#plt.xlabel('Years')
#plt.show()
```

```
---------------------------------------------------------------------------

TypeError                                 Traceback (most recent call last)

<ipython-input-156-aad18e61aaf3> in <module>
----> 1 test2.sort_values(by='Total', ascending=False, axis=0, inplace=True)
      2 df_top10 = test2.head(5)
      3 df_top7 = df_top7[years].transpose()
      4 print(df_top7)
      5 print('\n')


TypeError: sort_values() got an unexpected keyword argument 'by'
```

In [ ]: