In [ ]:
```
import numpy as np
import pandas as pd



df_can = pd.read_excel('https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNe

                       sheet_name='Canada by Citizenship',
                      skiprows=range(20),
                      skipfooter=2)

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

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

In [ ]:
```
import matplotlib.pyplot as plt
import matplotlib as mpl
```

In [ ]:
```
df_can
```

Out[ ]:

	Type	Coverage	OdName	AREA	AreaName	REG	RegName	DEV	DevName	1980	...	2004	2005	2006	2007	2008	2009
0	Immigrants	Foreigners	Afghanistan	935	Asia	5501	Southern Asia	902	Developing regions	16	...	2978	3436	3009	2652	2111	1746
1	Immigrants	Foreigners	Albania	908	Europe	925	Southern Europe	901	Developed regions	1	...	1450	1223	856	702	560	716
2	Immigrants	Foreigners	Algeria	903	Africa	912	Northern Africa	902	Developing regions	80	...	3616	3626	4807	3623	4005	5393
3	Immigrants	Foreigners	American Samoa	909	Oceania	957	Polynesia	902	Developing regions	0	...	0	0	1	0	0	0
4	Immigrants	Foreigners	Andorra	908	Europe	925	Southern Europe	901	Developed regions	0	...	0	0	1	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
190	Immigrants	Foreigners	Viet Nam	935	Asia	920	South-Eastern Asia	902	Developing regions	1191	...	1816	1852	3153	2574	1784	2171
191	Immigrants	Foreigners	Western Sahara	903	Africa	912	Northern Africa	902	Developing regions	0	...	0	0	1	0	0	0
192	Immigrants	Foreigners	Yemen	935	Asia	922	Western Asia	902	Developing regions	1	...	124	161	140	122	133	128
193	Immigrants	Foreigners	Zambia	903	Africa	910	Eastern Africa	902	Developing regions	11	...	56	91	77	71	64	60
194	Immigrants	Foreigners	Zimbabwe	903	Africa	910	Eastern Africa	902	Developing regions	72	...	1450	615	454	663	611	508

195 rows × 43 columns

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

Out[ ]:
```
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],
      dtype=int64)
```

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

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

In [ ]:
```
df_can.columns.to_list()

df_can.index.to_list()

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

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

In [ ]:
```
df_can.drop(['AREA','REG','DEV','Type','Coverage'], axis=1, inplace=True)
df_can.head(2)
```

Out[ ]:

	OdName	AreaName	RegName	DevName	1980	1981	1982	1983	1984	1985	...	2004	2005	2006	2007	2008	2009	2010	2011	
0	Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340	...	2978	3436	3009	2652	2111	1746	1758	2203	
1	Albania	Europe	Southern Europe	Developed regions	1	0	0	0	0	0	...	1450	1223	856	702	560	716	561	539	

2 rows × 38 columns

In [ ]:
```
df_can.rename(columns={'OdName':'Country', 'AreaName':'Continents', 'RegName':'Region', 'DevName' : 'Regions De
df_can.columns

df_can.head(5)
```

Out[ ]:

	Country	Continents	Region	Regions Development situation	1980	1981	1982	1983	1984	1985	...	2004	2005	2006	2007	2008	2009	2010	2011
0	Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340	...	2978	3436	3009	2652	2111	1746	1758	2203
1	Albania	Europe	Southern Europe	Developed regions	1	0	0	0	0	0	...	1450	1223	856	702	560	716	561	539
2	Algeria	Africa	Northern Africa	Developing regions	80	67	71	69	63	44	...	3616	3626	4807	3623	4005	5393	4752	4325
3	American Samoa	Oceania	Polynesia	Developing regions	0	1	0	0	0	0	...	0	0	1	0	0	0	0	0
4	Andorra	Europe	Southern Europe	Developed regions	0	0	0	0	0	0	...	0	0	1	1	0	0	0	0

5 rows × 38 columns

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

df_can.head(5)
```

```
C:\Users\aliki\AppData\Local\Temp\ipykernel_17264\2987482032.py:1: FutureWarning: Dropping of nuisance columns
in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError
.  Select only valid columns before calling the reduction.
  df_can['Total'] = df_can.sum(axis=1)
```

Out[ ]:

	Country	Continents	Region	Regions Development situation	1980	1981	1982	1983	1984	1985	...	2005	2006	2007	2008	2009	2010	2011	2012
0	Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340	...	3436	3009	2652	2111	1746	1758	2203	2635
1	Albania	Europe	Southern Europe	Developed regions	1	0	0	0	0	0	...	1223	856	702	560	716	561	539	620
2	Algeria	Africa	Northern Africa	Developing regions	80	67	71	69	63	44	...	3626	4807	3623	4005	5393	4752	4325	3774
3	American Samoa	Oceania	Polynesia	Developing regions	0	1	0	0	0	0	...	0	1	0	0	0	0	0	0
4	Andorra	Europe	Southern Europe	Developed regions	0	0	0	0	0	0	...	0	1	1	0	0	0	0	1

5 rows × 39 columns

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

Out[ ]:
```
Country                          0
Continents                       0
Region                           0
Regions Development situation    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
```

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

Out[ ]:

	1980	1981	1982	1983	1984	1985	1986	1987	1988	
count	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.0
mean	508.394872	566.989744	534.723077	387.435897	376.497436	358.861538	441.271795	691.133333	714.389744	843.2
std	1949.588546	2152.643752	1866.997511	1204.333597	1198.246371	1079.309600	1225.576630	2109.205607	2443.606788	2555.0
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
25%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.500000	0.500000	1.000000	1.0
50%	13.000000	10.000000	11.000000	12.000000	13.000000	17.000000	18.000000	26.000000	34.000000	44.0
75%	251.500000	295.500000	275.000000	173.000000	181.000000	197.000000	254.000000	434.000000	409.000000	508.5
max	22045.000000	24796.000000	20620.000000	10015.000000	10170.000000	9564.000000	9470.000000	21337.000000	27359.000000	23795.0

8 rows × 35 columns

In [ ]:
```
df_can.set_index("Country",inplace=True)
```

In [ ]:
```
df_can.columns = list(map(str, df_can.columns))

years = list(map(str, range(1980, 2014)))

years
```

Out[ ]:
```
['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']
```

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

Out[ ]:
```
Index(['Continents', 'Region', 'Regions Development situation', '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')
```

In [ ]:
```
df_can.head(3)
```

Out[ ]:

Country	Continents	Region	Regions Development situation	1980	1981	1982	1983	1984	1985	1986	...	2005	2006	2007	2008	2009	2010	2011	2
Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340	496	...	3436	3009	2652	2111	1746	1758	2203	2
Albania	Europe	Southern Europe	Developed regions	1	0	0	0	0	0	1	...	1223	856	702	560	716	561	539	
Algeria	Africa	Northern Africa	Developing regions	80	67	71	69	63	44	69	...	3626	4807	3623	4005	5393	4752	4325	3

3 rows × 38 columns

In [ ]:
```
df_can.index.name = None
```

In [ ]:
```
print(df_can.loc['Japan'])
```

```
Continents                                      Asia
Region                                  Eastern Asia
Regions Development situation     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 [ ]:
```
df_can.columns = list(map(str,df_can.columns))
```

In [ ]:
```
[print (type(x)) for x in df_can.columns.values]
```

```
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
```

Out[ ]:
```
[None,
 None,
 None,
 None,
 None,
 None,

 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None]
```

In [ ]:
```
years = list(map(str,range(1980,2014)))

years = [str(year) for year in range(1980, 2014)]
```

In [ ]:
```
condition_asia = df_can['Continents'] == 'Asia'

print(condition_asia)
```

```
Afghanistan        True
Albania           False
Algeria           False
American Samoa    False
Andorra           False
                  ...
Viet Nam           True
Western Sahara    False
Yemen              True
Zambia            False
Zimbabwe          False
Name: Continents, Length: 195, dtype: bool
```

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

Out[ ]:

	Continents	Region	Regions Development situation	1980	1981	1982	1983	1984	1985	1986	...	2005	2006	2007	2008	2009	2010
Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340	496	...	3436	3009	2652	2111	1746	1758
Bangladesh	Asia	Southern Asia	Developing regions	83	84	86	81	98	92	486	...	4171	4014	2897	2939	2104	4721
Bhutan	Asia	Southern Asia	Developing regions	0	0	0	0	1	0	0	...	5	10	7	36	865	1464
India	Asia	Southern Asia	Developing regions	8880	8670	8147	7338	5704	4211	7150	...	36210	33848	28742	28261	29456	34235
Iran (Islamic Republic of)	Asia	Southern Asia	Developing regions	1172	1429	1822	1592	1977	1648	1794	...	5837	7480	6974	6475	6580	7477
Maldives	Asia	Southern Asia	Developing regions	0	0	0	1	0	0	0	...	0	0	2	1	7	4
Nepal	Asia	Southern Asia	Developing regions	1	1	6	1	2	4	13	...	607	540	511	581	561	1392
Pakistan	Asia	Southern Asia	Developing regions	978	972	1201	900	668	514	691	...	14314	13127	10124	8994	7217	6811
Sri Lanka	Asia	Southern Asia	Developing regions	185	371	290	197	1086	845	1838	...	4930	4714	4123	4756	4547	4422

9 rows × 38 columns

In [ ]:
```
print('data dimensions:', df_can.shape)
print(df_can.columns)
df_can.head(3)
```

```
data dimensions: (195, 38)
Index(['Continents', 'Region', 'Regions Development situation', '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[ ]:

	Continents	Region	Regions Development situation	1980	1981	1982	1983	1984	1985	1986	...	2005	2006	2007	2008	2009	2010	2011
Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340	496	...	3436	3009	2652	2111	1746	1758	2203
Albania	Europe	Southern Europe	Developed regions	1	0	0	0	0	0	1	...	1223	856	702	560	716	561	539
Algeria	Africa	Northern Africa	Developing regions	80	67	71	69	63	44	69	...	3626	4807	3623	4005	5393	4752	4325

3 rows × 38 columns

In [ ]:
```
print(plt.style.available)

mpl.style.use(['ggplot'])
```

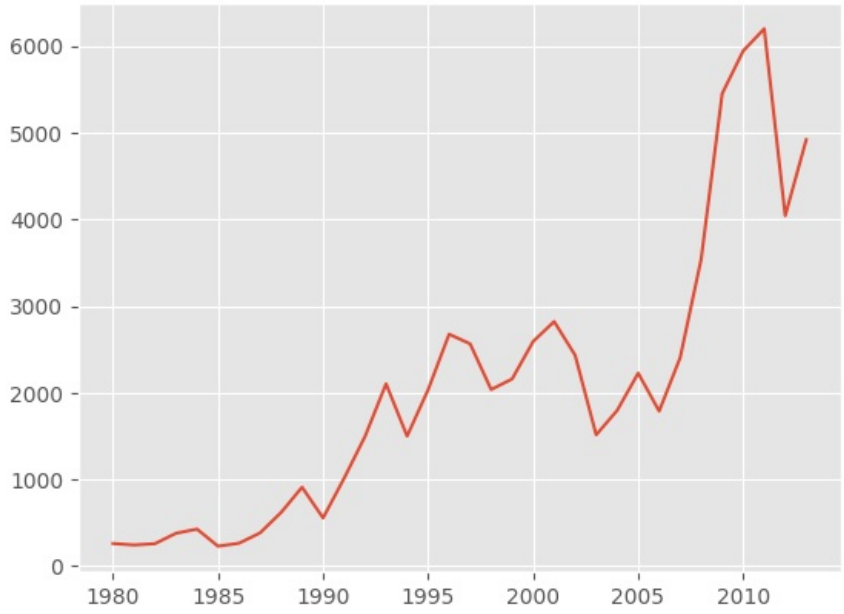
```
['Solarize_Light2', '_classic_test_patch', '_mpl-gallery', '_mpl-gallery-nogrid', 'bmh', 'classic', 'dark_backg
round', 'fast', 'fivethirtyeight', 'ggplot', 'grayscale', 'seaborn-v0_8', 'seaborn-v0_8-bright', 'seaborn-v0_8-
colorblind', 'seaborn-v0_8-dark', 'seaborn-v0_8-dark-palette', 'seaborn-v0_8-darkgrid', 'seaborn-v0_8-deep', 's
eaborn-v0_8-muted', 'seaborn-v0_8-notebook', 'seaborn-v0_8-paper', 'seaborn-v0_8-pastel', 'seaborn-v0_8-poster'
, 'seaborn-v0_8-talk', 'seaborn-v0_8-ticks', 'seaborn-v0_8-white', 'seaborn-v0_8-whitegrid', 'tableau-colorblin
d10']
```

In [ ]:
```
haiti = df_can.loc['Iraq', years]
print(haiti)
```

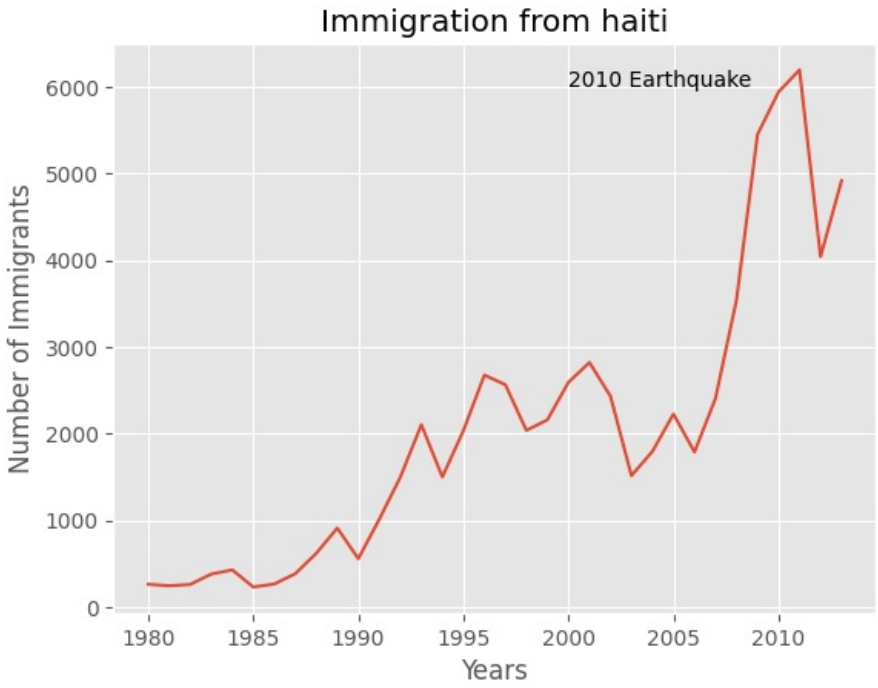
```
1980     262
1981     245
1982     260
1983     380
1984     428
1985     231
1986     265
1987     384
1988     619
1989     911
1990     557
1991    1013
1992    1498
1993    2103
1994    1500
1995    2034
1996    2675
1997    2564
1998    2037
1999    2159
2000    2591
2001    2821
2002    2432
2003    1515
2004    1796
2005    2226
2006    1788
2007    2406
2008    3543
2009    5450
2010    5941
2011    6196
2012    4041
2013    4918
Name: Iraq, dtype: object
```

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

Out[ ]:
```
<Axes: >
```



In [ ]:
```
haiti.index = haiti.index.map(str)
haiti.plot(kind = 'line')

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

plt.text(20,6000 , '2010 Earthquake')

plt.show()
```



In [ ]:
```
df_China_India = df_can.loc[['China','India'],years]
df_China_India.head(5)
```

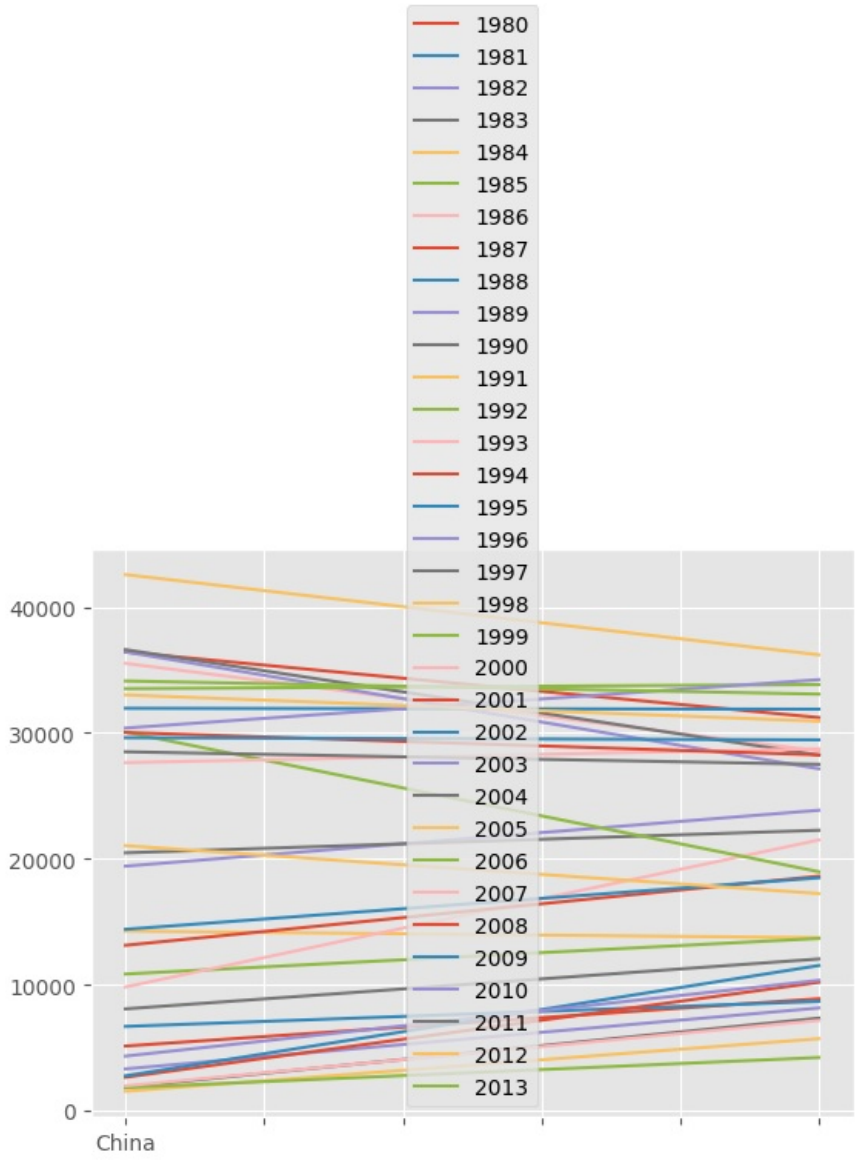
Out[ ]:

	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	...	2004	2005	2006	2007	2008	2009	2010	2011	2012
China	5123	6682	3308	1863	1527	1816	1960	2643	2758	4323	...	36619	42584	33518	27642	30037	29622	30391	28502	33024
India	8880	8670	8147	7338	5704	4211	7150	10189	11522	10343	...	28235	36210	33848	28742	28261	29456	34235	27509	30933

2 rows × 34 columns

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

Out[ ]:
```
<Axes: >
```



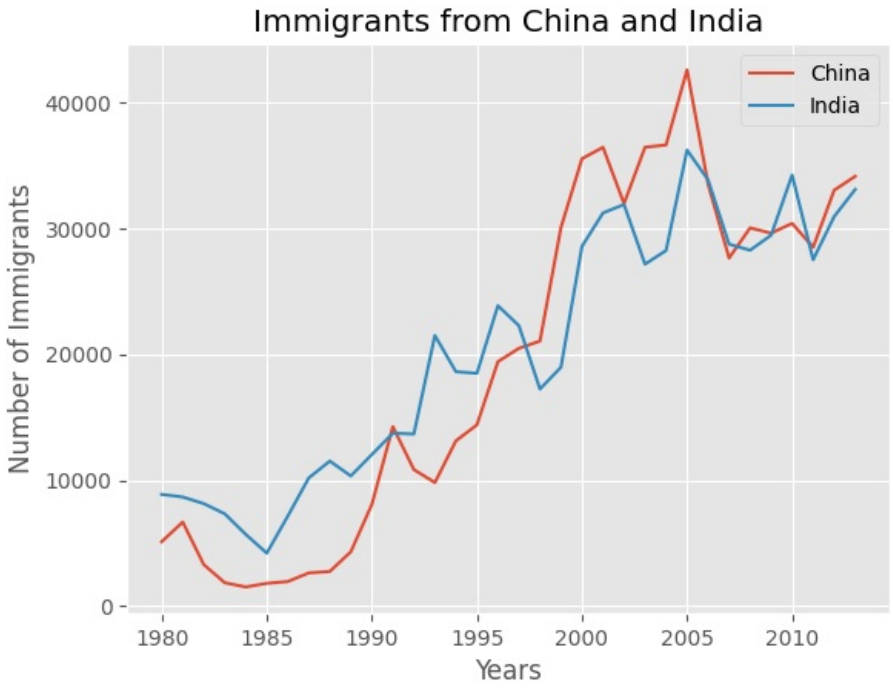
In [ ]:
```
df_China_India = df_China_India.transpose()
df_China_India.head()
```

Out[ ]:

	China	India
1980	5123	8880
1981	6682	8670
1982	3308	8147
1983	1863	7338
1984	1527	5704

In [ ]:
```
df_China_India.plot(kind="line")

plt.title("Immigrants from China and India")
plt.ylabel("Number of Immigrants")
plt.xlabel("Years")
```

Out[ ]:
```
Text(0.5, 0, 'Years')
```



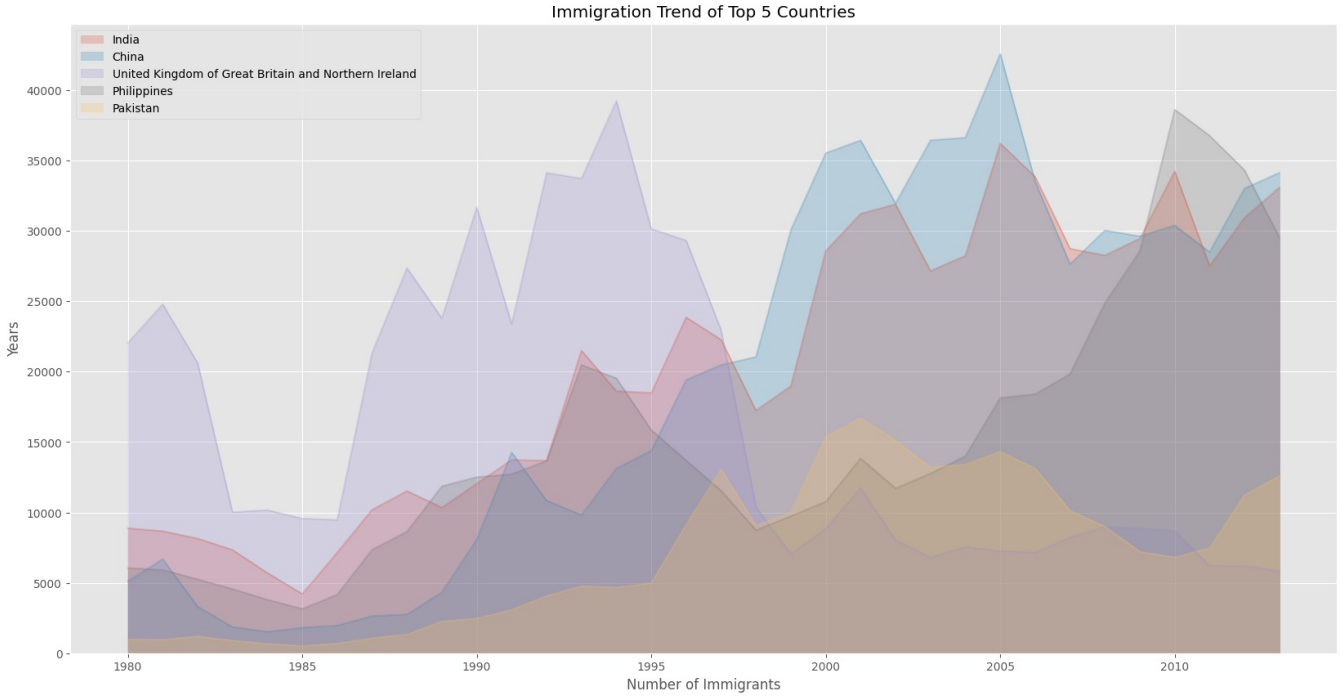
In [ ]:
```
df_can.sort_values(["Total"], ascending=False, axis=0, inplace=True)

df_top5 = df_can.head()

df_top5 = df_top5[years].transpose()

df_top5.head()
```

Out[ ]:

	India	China	United Kingdom of Great Britain and Northern Ireland	Philippines	Pakistan
1980	8880	5123	22045	6051	978
1981	8670	6682	24796	5921	972
1982	8147	3308	20620	5249	1201
1983	7338	1863	10015	4562	900
1984	5704	1527	10170	3801	668

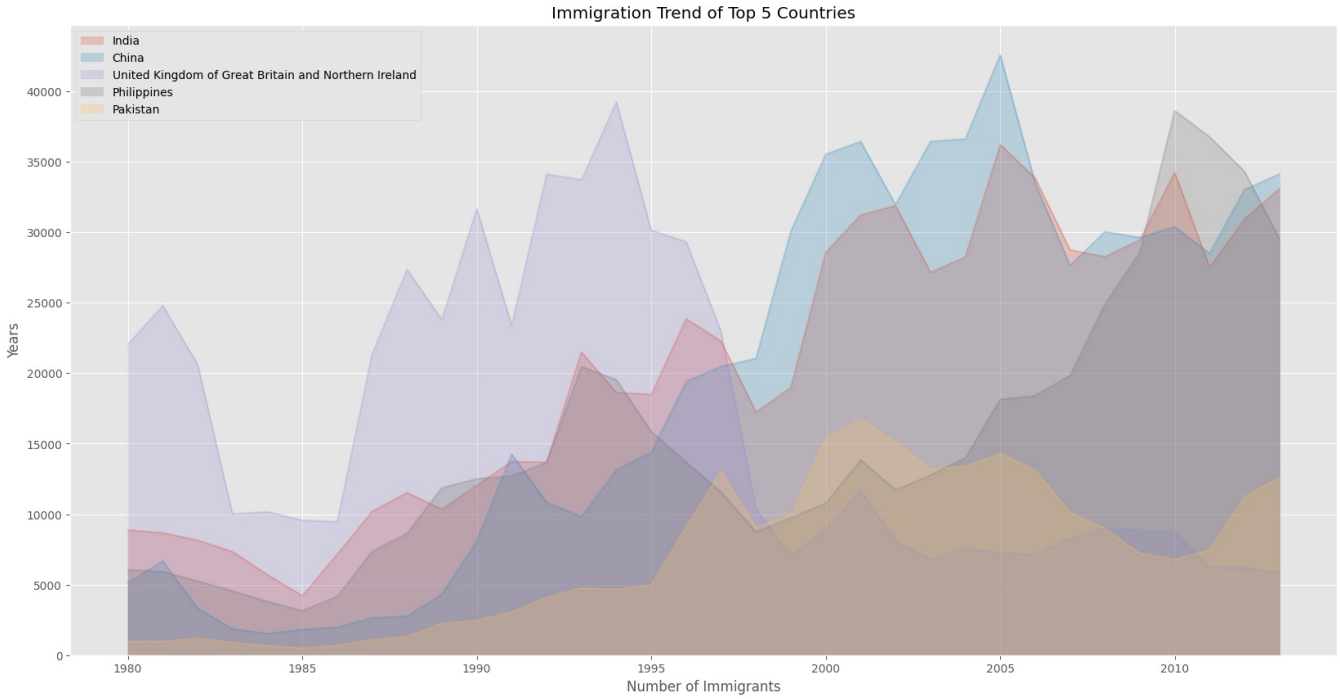
In [ ]:
```
df_top5.index =df_top5.index.map(int)
df_top5.plot(kind="area",
             alpha =0.25,
             stacked=False,
             figsize=(20,10)
             )

plt.title("Immigration Trend of Top 5 Countries")
plt.xlabel("Number of Immigrants")
plt.ylabel("Years")
```

Out[ ]:
```
Text(0, 0.5, 'Years')
```



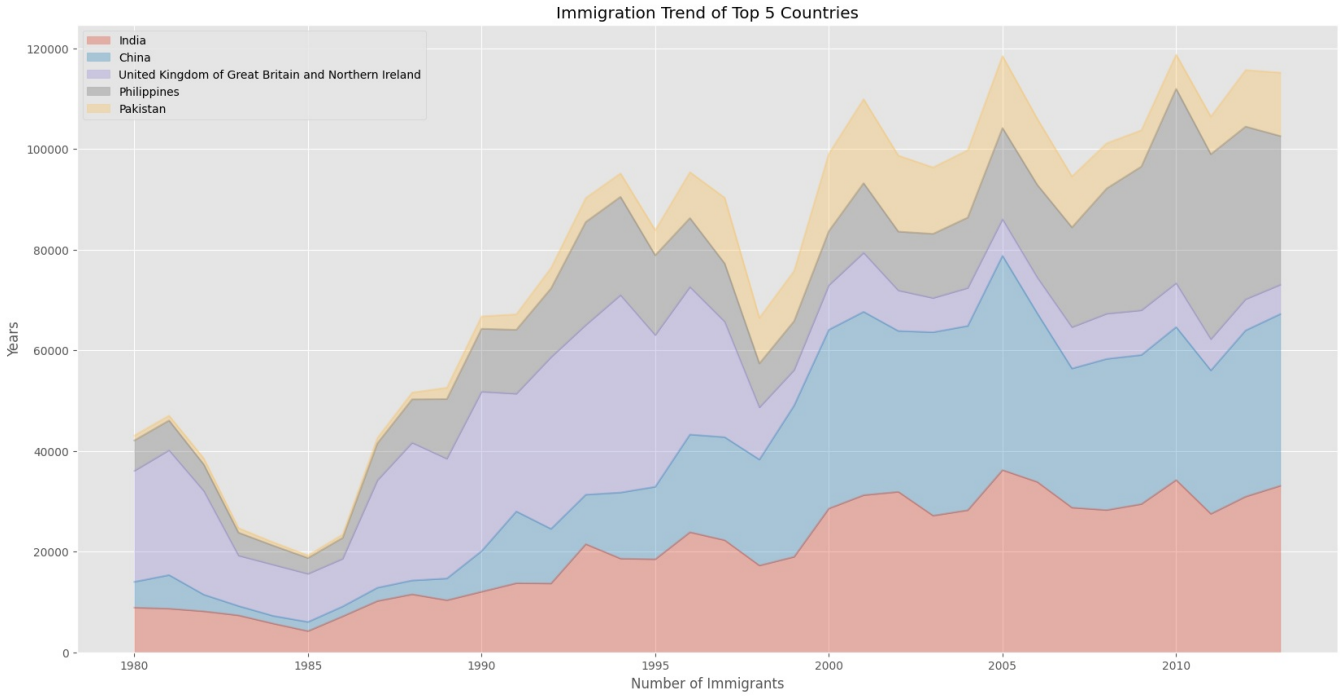
In [ ]:
```
df_top5.index =df_top5.index.map(int)
df_top5.plot(kind="area",
             alpha =0.25,
             stacked=False,
             figsize=(20,10)
             )

plt.title("Immigration Trend of Top 5 Countries")
plt.xlabel("Number of Immigrants")
plt.ylabel("Years")
```

Out[ ]:
```
Text(0, 0.5, 'Years')
```



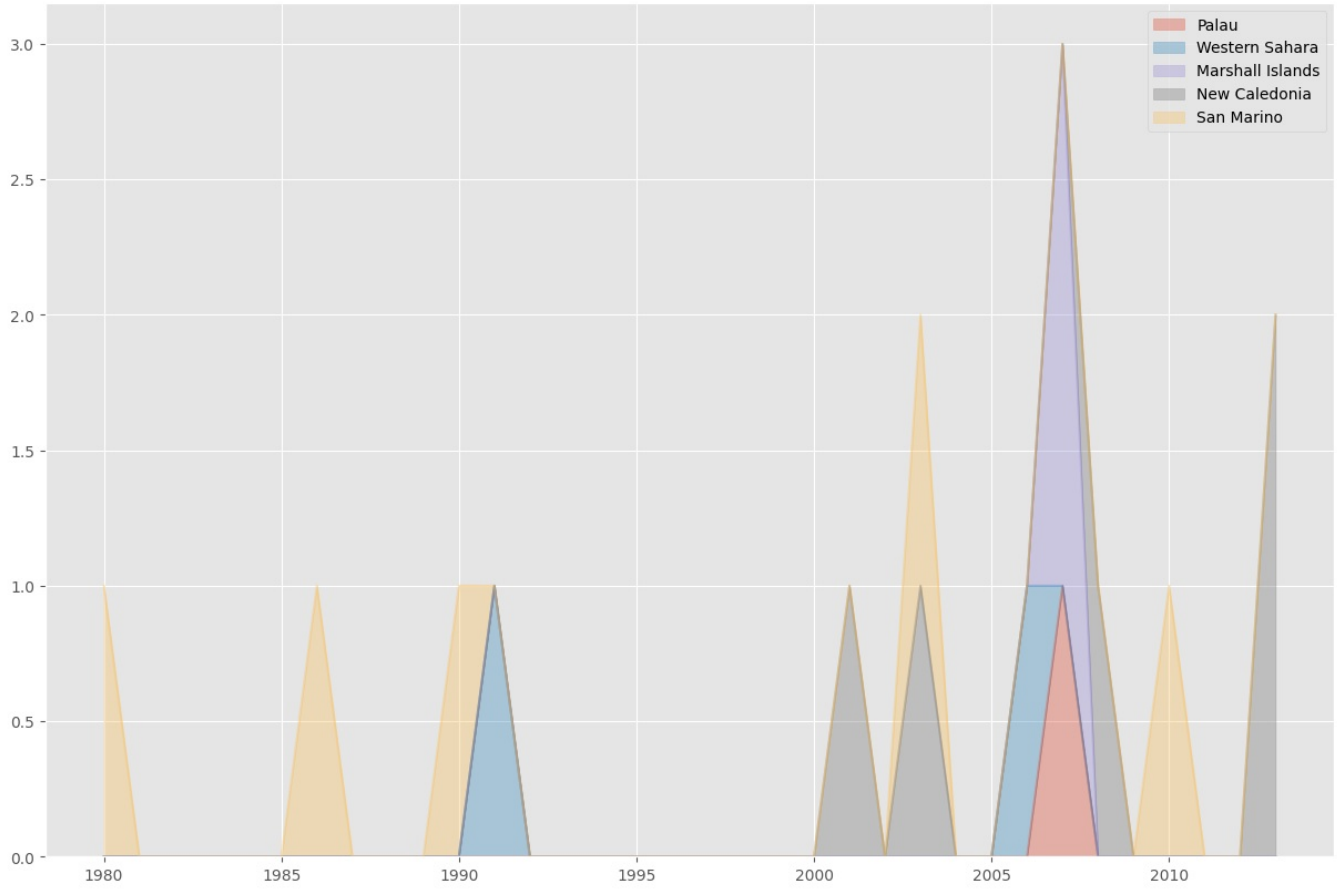
In [ ]:
```
ax = df_top5.plot(kind="area",alpha = 0.35, figsize=(20,10))

ax.set_title("Immigration Trend of Top 5 Countries")
ax.set_xlabel("Number of Immigrants")
ax.set_ylabel("Years")
```

Out[ ]:
```
Text(0, 0.5, 'Years')
```



In [ ]:
```
df_least_Immigrations = df_can.sort_values(["Total"], ascending=True, axis=0)
df_least_Immigrations = df_least_Immigrations[years]
df_least_Immigrations = df_least_Immigrations.iloc[0:5,:]
df_least_Immigrations = df_least_Immigrations.transpose()
df_least_Immigrations.head()
df_least_Immigrations.plot(kind='area', alpha = 0.35, figsize=(15,10))
```

Out[ ]:
```
<Axes: >
```



In [ ]:
```
df_can['2013'].head()
```

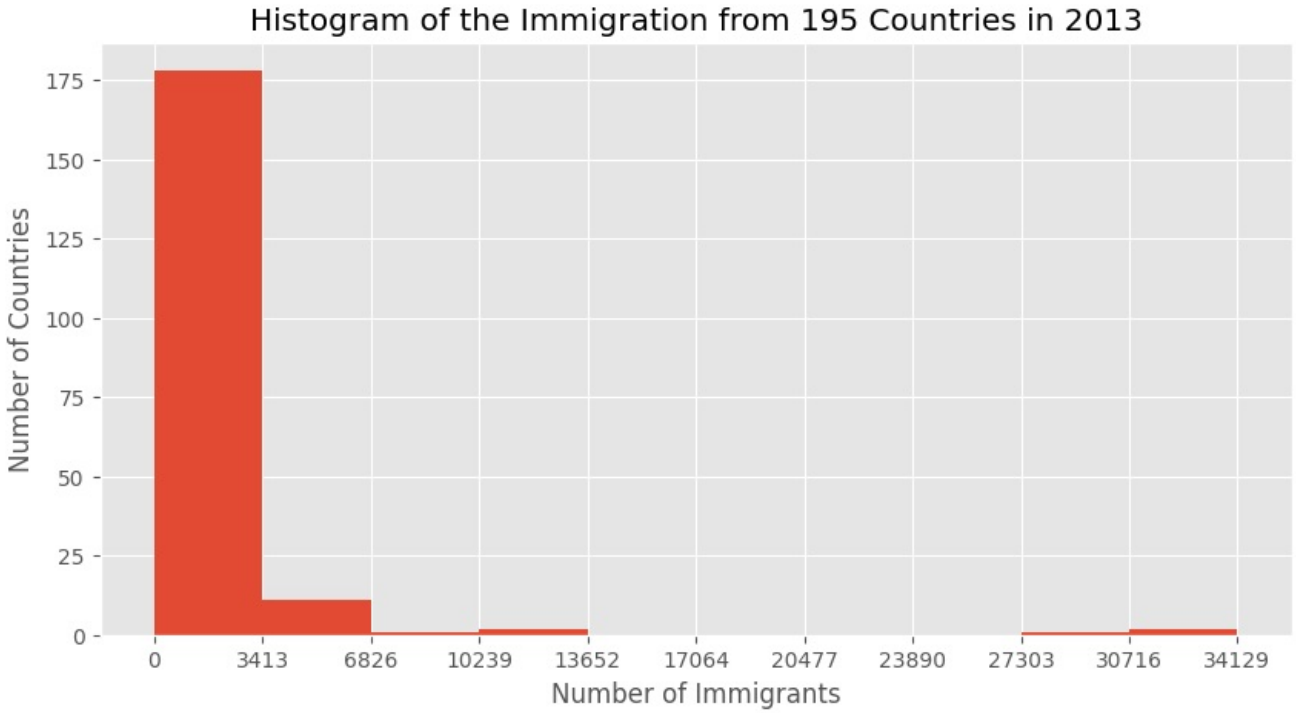
Out[ ]:
```
India                                                33087
China                                                34129
United Kingdom of Great Britain and Northern Ireland     5827
Philippines                                          29544
Pakistan                                             12603
Name: 2013, dtype: int64
```

In [ ]:
```
count, bin_edges = np.histogram(df_can['2013'])

print(count)
```

```
print(bin_edges)
```

```
[178  11   1   2   0   0   0   0   1   2]
[    0.   3412.9  6825.8 10238.7 13651.6 17064.5 20477.4 23890.3 27303.2
 30716.1 34129. ]
```

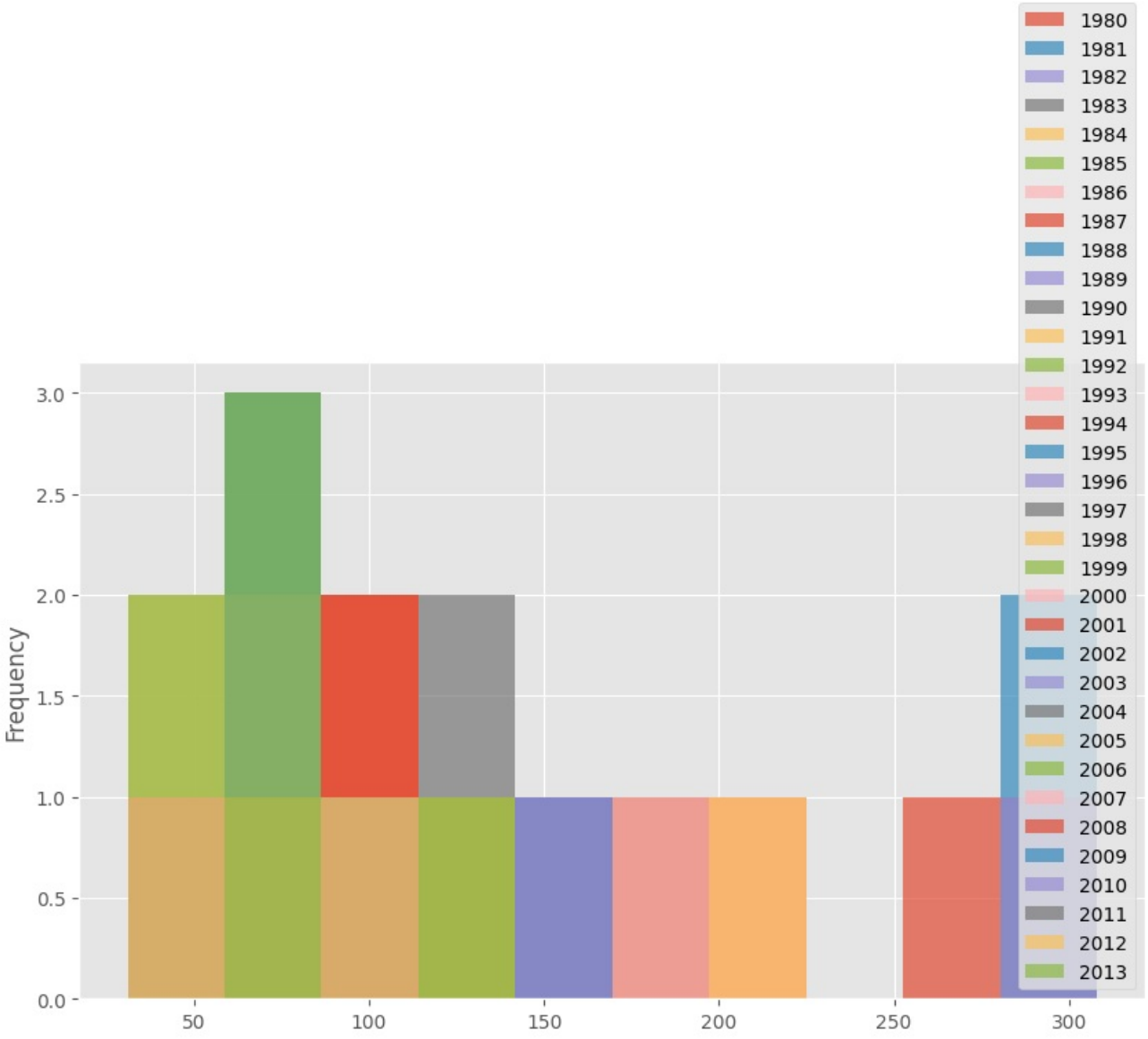
In [ ]:
```
count, bin_edges = np.histogram(df_can['2013'])

df_can['2013'].plot(kind='hist', figsize=(10,5), xticks=bin_edges)

plt.title('Histogram of the Immigration from 195 Countries in 2013')
plt.ylabel('Number of Countries')
plt.xlabel('Number of Immigrants')

plt.show()
```



In [ ]:
```
Countries_to_plot = ['Denmark', 'Norway', 'Finland', 'Sweden']
```

In [ ]:
```
df_can.loc[Countries_to_plot, years].plot(kind='hist',
                                          figsize= (10, 6),
                                          stacked=False,
                                          alpha=0.7,
                                          )
```

Out[ ]:
```
<Axes: ylabel='Frequency'>
```



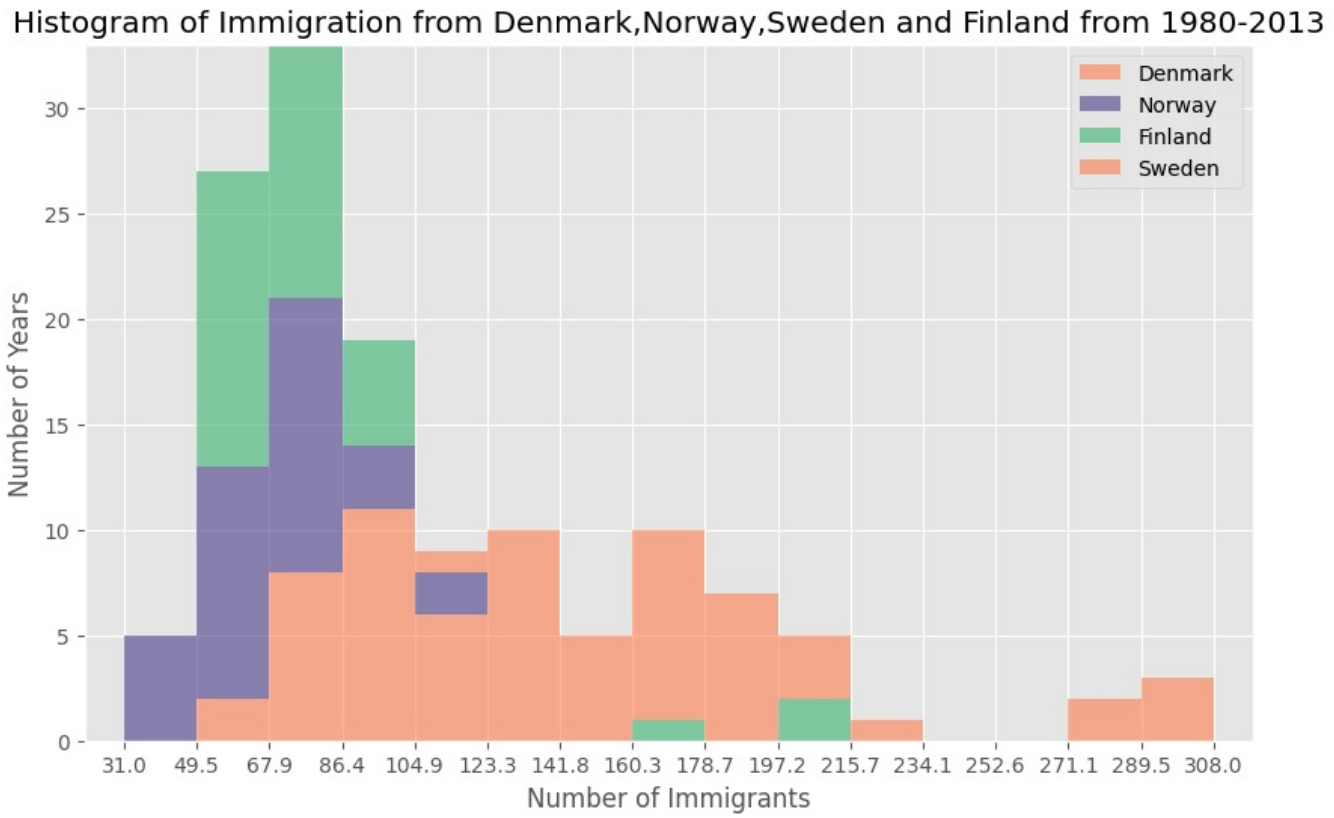
In [ ]:
```
df_t = df_can.loc[Countries_to_plot, years].transpose()
df_t.head()
```

Out[ ]:

	Denmark	Norway	Finland	Sweden
1980	272	116	208	281
1981	293	77	205	308
1982	299	106	170	222
1983	106	51	70	176
1984	93	31	83	128

In [ ]:

```
count, bin_edges = np.histogram(df_t, 15)
xmin = bin_edges[0] - 10        #  first bin value is 31.0, adding buffer of 10 for aesthetic purposes
xmax = bin_edges[-1] + 10       #  last bin value is 308.0, adding buffer of 10 for aesthetic purposes



## stacking the histogram for be able to see it more clear
df_t.plot(kind='hist',
          figsize=(10, 6),
          bins=15,
          alpha=0.6,
          xticks=bin_edges,
          color=['coral', 'darkslateblue', 'mediumseagreen'],
          stacked=True,
          xlim=(xmin, xmax)
          )


plt.title("Histogram of Immigration from Denmark,Norway,Sweden and Finland from 1980-2013")
plt.ylabel("Number of Years")
plt.xlabel("Number of Immigrants")

plt.show()
```

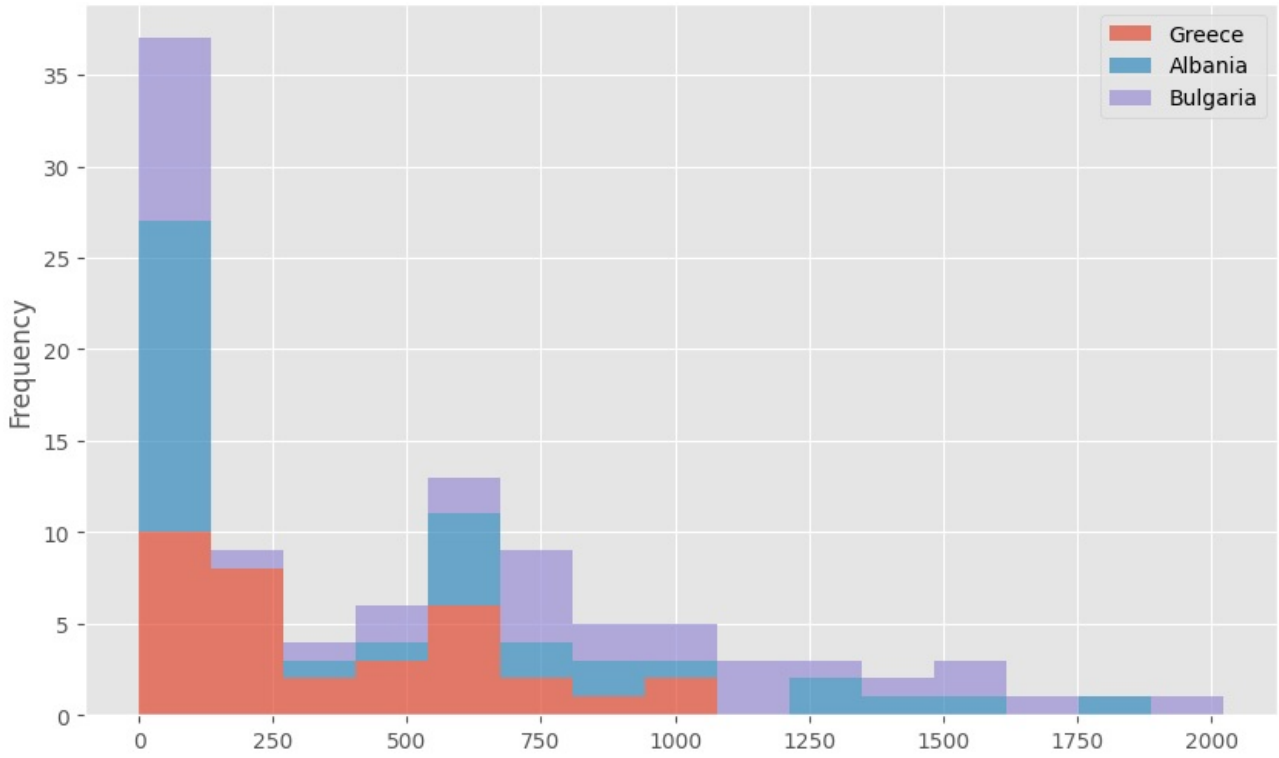


In [ ]:

```
df_Greece_Albania_Bulgaria = df_can.loc[['Greece', 'Albania', 'Bulgaria'], years]
df_Greece_Albania_Bulgaria= df_Greece_Albania_Bulgaria.transpose()
df_Greece_Albania_Bulgaria.plot(kind='hist', figsize=(10, 6), bins = 15, alpha = 0.7, stacked=True)
```

Out[ ]:

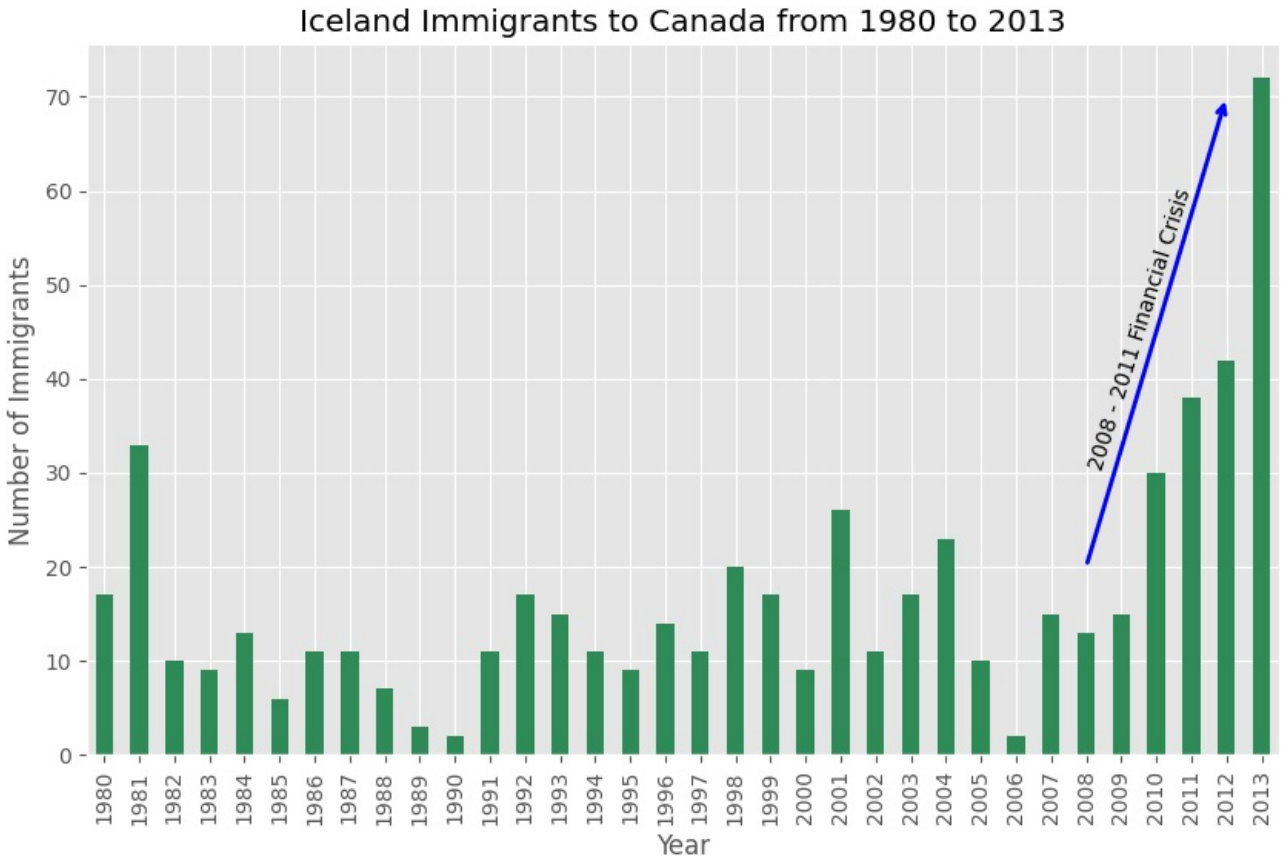
```
<Axes: ylabel='Frequency'>
```



In [ ]:
```
df_iceland = df_can.loc["Iceland", years]
df_iceland.head(5)
```

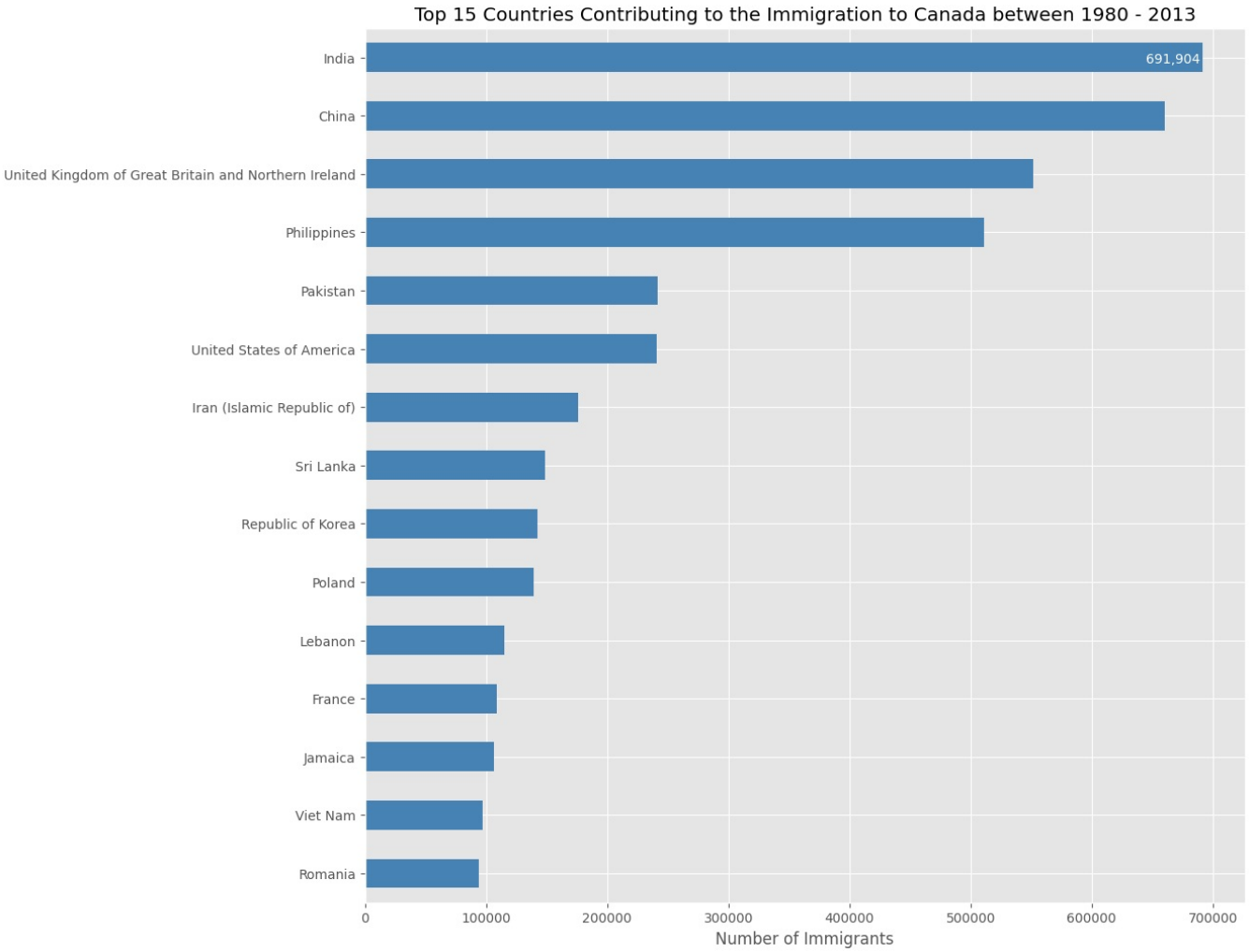
Out[ ]:
```
1980    17
1981    33
1982    10
1983     9
1984    13
Name: Iceland, dtype: object
```

In [ ]:
```
df_iceland.plot(kind='bar', figsize=(10, 6),
                color=["seagreen"])

plt.xlabel('Year')
plt.ylabel('Number of Immigrants')
plt.title("Iceland Immigrants to Canada from 1980 to 2013")


plt.annotate('',
             xy=(32,70),
             xytext=(28,20),
             xycoords='data',
             arrowprops=dict(arrowstyle='->', connectionstyle='arc3', color='blue', lw=2)
             )


plt.annotate('2008 - 2011 Financial Crisis',
             xy=(28,30),
             rotation=72.5,
             va = 'bottom',
             ha='left',
             )

plt.show()
```



In [ ]:

```
df_can.sort_values(['Total'], ascending= True, axis=0, inplace=True)


df_can_top15 = df_can['Total'].tail(15)
df_can_top15

df_can_top15.plot(kind='barh', figsize=(12,12), color='steelblue')
plt.xlabel('Number of Immigrants')
plt.title('Top 15 Countries Contributing to the Immigration to Canada between 1980 - 2013')

for index, value in enumerate(df_can_top15):
    label = format(int(value), ',')

plt.annotate(label, xy=(value - 47000, index - 0.10), color='white')

plt.show()
```



In [ ]:
```
df_Continents = df_can.groupby('Continents', axis=0).sum()

print(type(df_can.groupby('Continents', axis=0)))

df_Continents.head()
```

```
<class 'pandas.core.groupby.generic.DataFrameGroupBy'>
C:\Users\aliki\AppData\Local\Temp\ipykernel_17264\2730698030.py:1: FutureWarning: The default value of numeric_
only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will default to False. Either spe
cify numeric_only or select only columns which should be valid for the function.
  df_Continents = df_can.groupby('Continents', axis=0).sum()
```

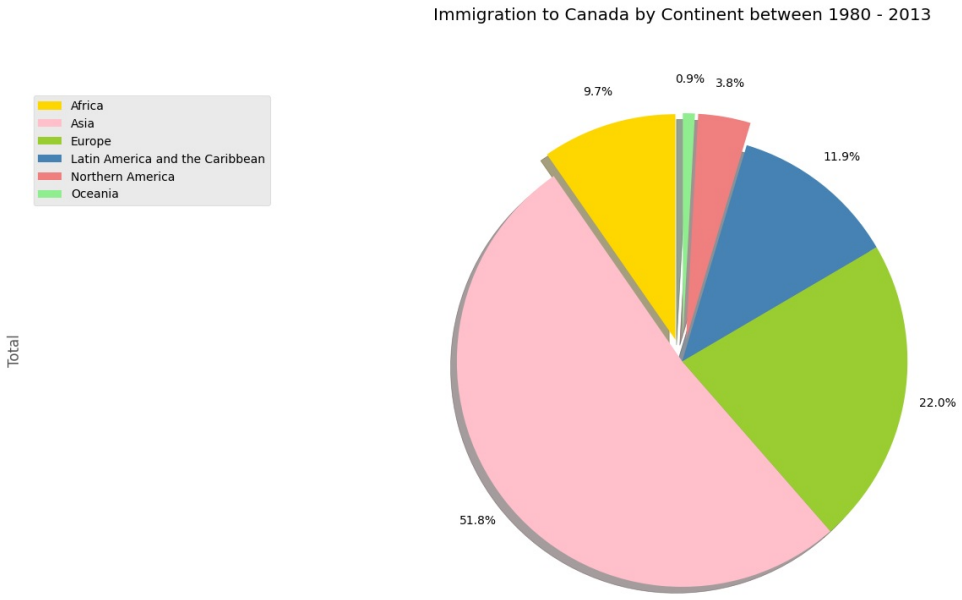
Out[ ]:

Continents	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	...	2005	2006	2007	2008	2009	2010
Africa	3951	4363	3819	2671	2639	2650	3782	7494	7552	9894	...	27523	29188	28284	29890	34534	40892
Asia	31025	34314	30214	24696	27274	23850	28739	43203	47454	60256	...	159253	149054	133459	139894	141434	163845
Europe	39760	44802	42720	24638	22287	20844	24370	46698	54726	60893	...	35955	33053	33495	34692	35078	33425
Latin America and the Caribbean	13081	15215	16769	15427	13678	15171	21179	28471	21924	25060	...	24747	24676	26011	26547	26867	28818
Northern America	9378	10030	9074	7100	6661	6543	7074	7705	6469	6790	...	8394	9613	9463	10190	8995	8142

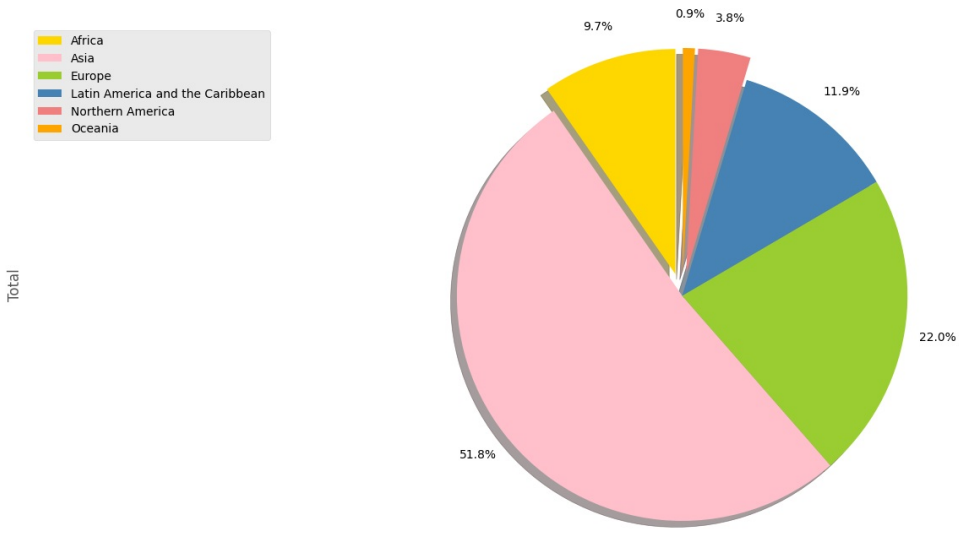
5 rows × 35 columns

In [ ]:
```
color_list=['gold','pink','yellowgreen','steelblue','lightcoral','lightgreen']

explode_list = [0.1,0, 0, 0 , 0.1, 0.1]

df_Continents['Total'].plot(kind='pie',
                            figsize=(20,8),
                            shadow=True,
                            startangle=90,
                            autopct= '%1.1f%%',
                            labels=None,
                            pctdistance=1.15,
                            colors=color_list,
                            explode = explode_list
                            )
```

```
plt.legend(labels=df_Continents.index, loc= 'upper left')

plt.title('Immigration to Canada by Continent between 1980 - 2013',y=1.12)
plt.axis('equal')

plt.show()
```



In [ ]:
```
color_list=['gold','pink','yellowgreen','steelblue','lightcoral','orange']

explode_list = [0.1,0, 0, 0 , 0.1, 0.1]

df_Continents['Total'].plot(kind='pie',
                            figsize=(20,8),
                            shadow=True,
                            startangle=90,
                            autopct= '%1.1f%%',
                            labels=None,
                            pctdistance=1.15,
                            colors=color_list,
                            explode = explode_list
                            )

plt.legend(labels=df_Continents.index, loc= 'upper left')

plt.title('Immigration to Canada by Continent between 1980 - 2013',y=1.12)
plt.axis('equal')

plt.show()
```



In [ ]:
```
df_Continents
```

Out[ ]:

Continents	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	...	2005	2006	2007	2008	2009	2010
Africa	3951	4363	3819	2671	2639	2650	3782	7494	7552	9894	...	27523	29188	28284	29890	34534	40892
Asia	31025	34314	30214	24696	27274	23850	28739	43203	47454	60256	...	159253	149054	133459	139894	141434	163845
Europe	39760	44802	42720	24638	22287	20844	24370	46698	54726	60893	...	35955	33053	33495	34692	35078	33425
Latin America and the Caribbean	13081	15215	16769	15427	13678	15171	21179	28471	21924	25060	...	24747	24676	26011	26547	26867	28818
Northern America	9378	10030	9074	7100	6661	6543	7074	7705	6469	6790	...	8394	9613	9463	10190	8995	8142
Oceania	1942	1839	1675	1018	878	920	904	1200	1181	1539	...	1585	1473	1693	1834	1860	1834

6 rows × 35 columns

In [ ]:
```
df_Japan = df_can.loc[['Japan'], years].transpose()

df_Japan.head()
```

Out[ ]:

	Japan
1980	701
1981	756
1982	598
1983	309
1984	246

In [ ]:
```
df_Japan.describe()
```

Out[ ]:

	Japan
count	34.000000
mean	814.911765
std	337.219771
min	198.000000
25%	529.000000
50%	902.000000
75%	1079.000000
max	1284.000000

In [ ]:
```
df_China_India = df_can.loc[['China','India'], years].transpose()

df_China_India.head()
```

Out[ ]:

	China	India
1980	5123	8880
1981	6682	8670
1982	3308	8147
1983	1863	7338
1984	1527	5704

In [ ]:
```
df_China_India.describe()
```

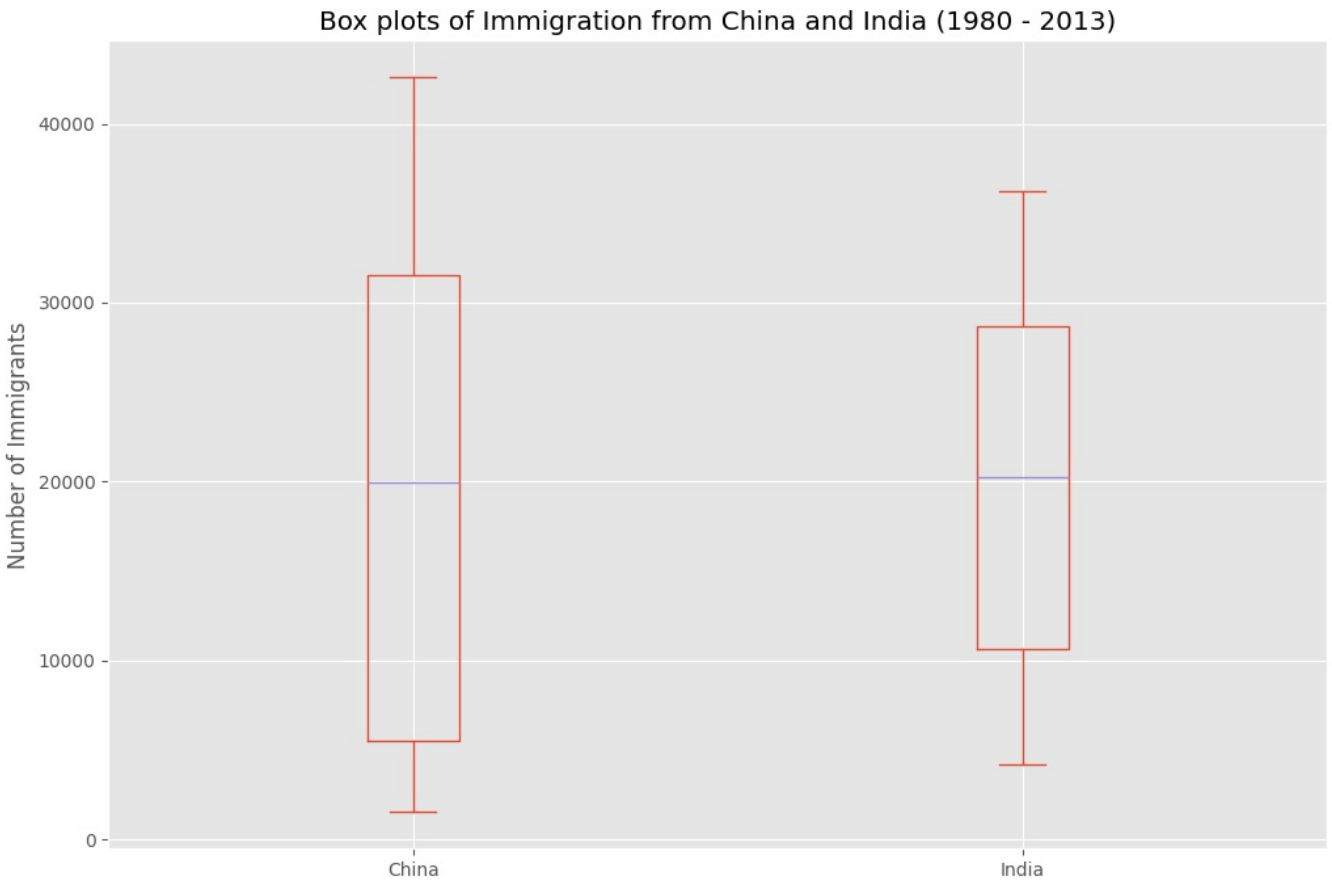
Out[ ]:

	China	India
count	34.000000	34.000000
mean	19410.647059	20350.117647
std	13568.230790	10007.342579
min	1527.000000	4211.000000
25%	5512.750000	10637.750000
50%	19945.000000	20235.000000
75%	31568.500000	28699.500000
max	42584.000000	36210.000000

In [ ]:
```
df_China_India.plot(kind='box', figsize=(12,8))
```

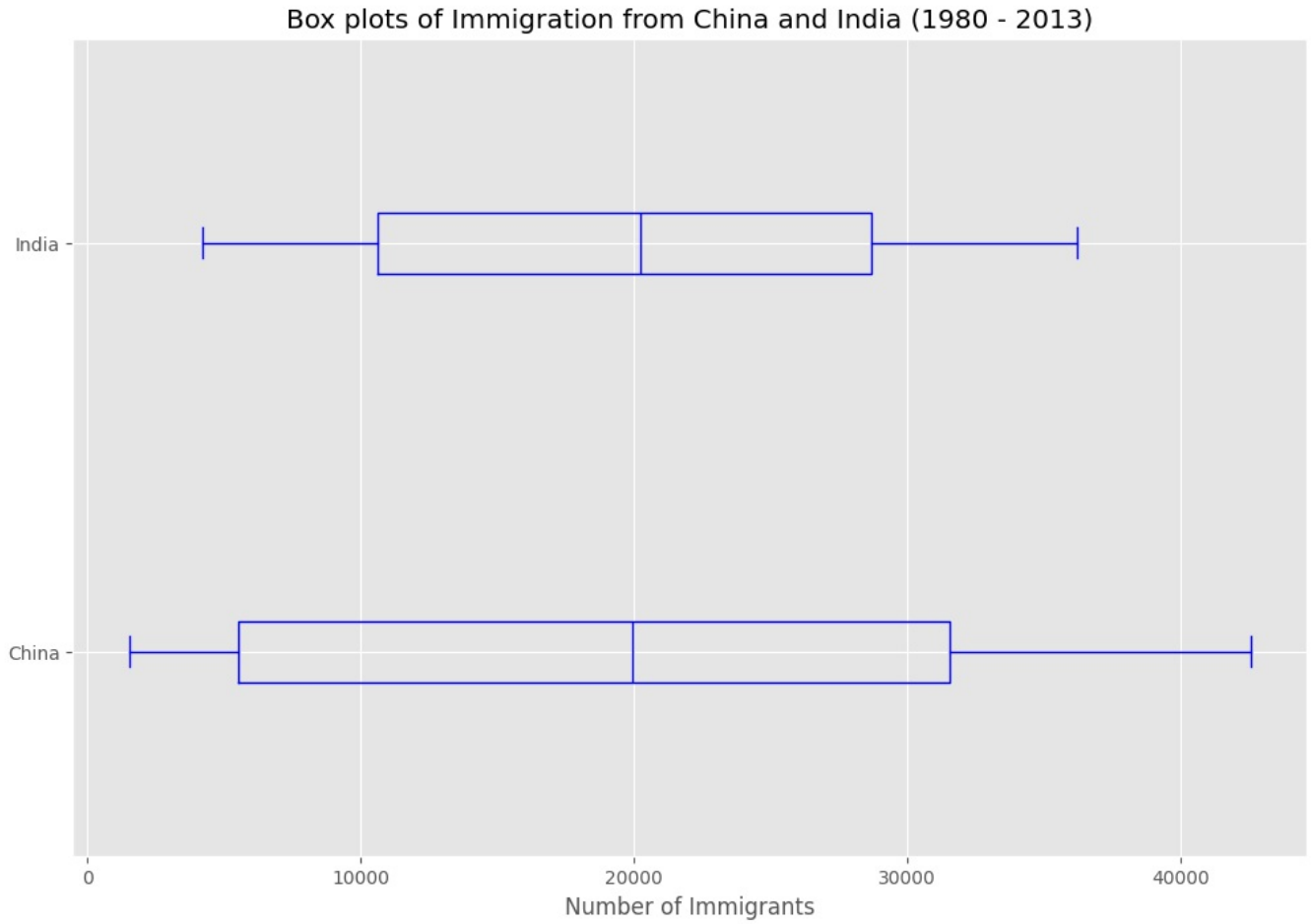
```
plt.title('Box plots of Immigration from China and India (1980 - 2013)')
plt.ylabel('Number of Immigrants')
```

Out[ ]:
```
Text(0, 0.5, 'Number of Immigrants')
```



In [ ]:
```
df_China_India.plot(kind='box', figsize=(12,8), color='blue', vert=False)
plt.title('Box plots of Immigration from China and India (1980 - 2013)')
plt.xlabel('Number of Immigrants')
```

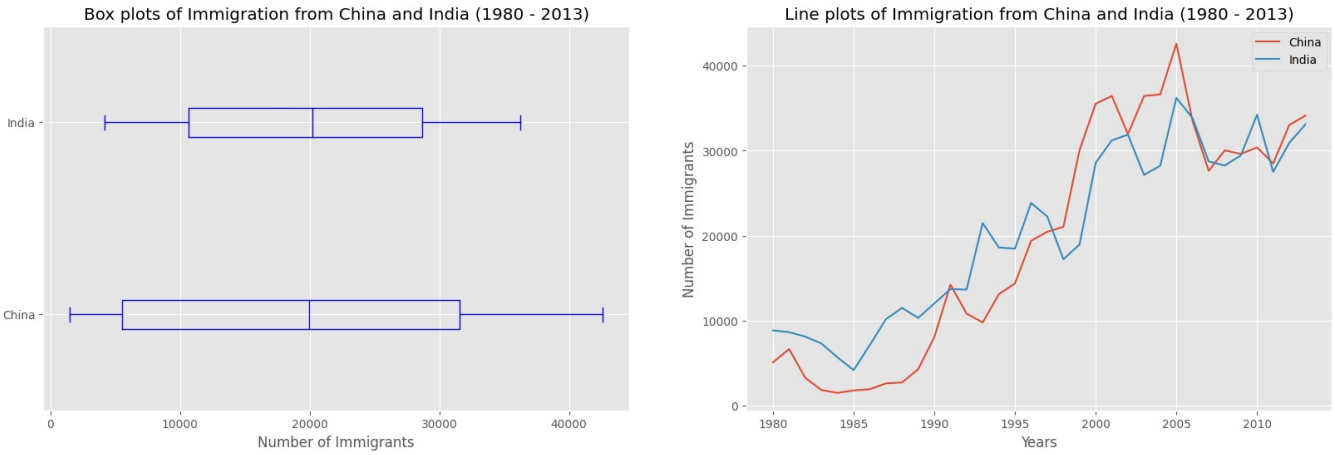
Out[ ]:
```
Text(0.5, 0, 'Number of Immigrants')
```



In [ ]:
```
fig = plt.figure()
```

In [ ]:
```
fig = plt.figure()

ax0 = fig.add_subplot(1,2,1)
ax1 = fig.add_subplot(1,2,2)

df_China_India.plot(kind='box', color='blue', vert=False, figsize=(20,6), ax=ax0)
ax0.set_title('Box plots of Immigration from China and India (1980 - 2013)')
ax0.set_xlabel('Number of Immigrants')

df_China_India.plot(kind='line', figsize=(20,6), ax=ax1)
ax1.set_title('Line plots of Immigration from China and India (1980 - 2013)')
ax1.set_xlabel('Years')
ax1.set_ylabel('Number of Immigrants')
```

Out[ ]:
```
Text(0, 0.5, 'Number of Immigrants')
```



In [ ]:
```
df_can.sort_values(by='Total', inplace=True, ascending=False, axis=0)

df_can_top15= df_can[years].head(15)

df_can_top15
```

Out[ ]:

	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	...	2004	2005	2006	2007	2008	2009	2010	2
India	8880	8670	8147	7338	5704	4211	7150	10189	11522	10343	...	28235	36210	33848	28742	28261	29456	34235	27
China	5123	6682	3308	1863	1527	1816	1960	2643	2758	4323	...	36619	42584	33518	27642	30037	29622	30391	28
United Kingdom of Great Britain and Northern Ireland	22045	24796	20620	10015	10170	9564	9470	21337	27359	23795	...	7533	7258	7140	8216	8979	8876	8724	6
Philippines	6051	5921	5249	4562	3801	3150	4166	7360	8639	11865	...	14004	18139	18400	19837	24887	28573	38617	36
Pakistan	978	972	1201	900	668	514	691	1072	1334	2261	...	13399	14314	13127	10124	8994	7217	6811	7
United States of America	9378	10030	9074	7100	6661	6543	7074	7705	6469	6790	...	6990	8394	9613	9463	10190	8995	8142	7
Iran (Islamic Republic of)	1172	1429	1822	1592	1977	1648	1794	2989	3273	3781	...	6348	5837	7480	6974	6475	6580	7477	7
Sri Lanka	185	371	290	197	1086	845	1838	4447	2779	2758	...	4495	4930	4714	4123	4756	4547	4422	3
Republic of Korea	1011	1456	1572	1081	847	962	1208	2338	2805	2979	...	5352	5832	6215	5920	7294	5874	5537	4
Poland	863	2930	5881	4546	3588	2819	4808	6931	9211	16025	...	1533	1405	1263	1235	1267	1013	795	
Lebanon	1409	1119	1159	789	1253	1683	2576	3803	3970	7157	...	3293	3709	3802	3467	3566	3077	3432	3
France	1729	2027	2219	1490	1169	1177	1298	1658	2038	2332	...	4391	4429	4002	4290	4532	5051	4646	4
Jamaica	3198	2634	2661	2455	2508	2938	4649	5415	3924	3946	...	2237	1945	1722	2141	2334	2456	2321	2
Viet Nam	1191	1829	2162	3404	7583	5907	2741	1406	1411	3004	...	1816	1852	3153	2574	1784	2171	1942	1
Romania	375	438	583	543	524	604	656	1202	1106	1582	...	5755	5048	4468	3834	2837	2076	1922	1

15 rows × 34 columns

In [ ]:
```
df_80s = df_can_top15.iloc[:, 0:10].sum(axis=1)
df_90s = df_can_top15.iloc[:, 10:20].sum(axis=1)
df_00s = df_can_top15.iloc[:, 20:].sum(axis=1)

new_df_withdecades =pd.DataFrame({'1980s': df_80s, '1990s': df_90s, '2000s': df_00s})

new_df_withdecades.head()
```

Out[ ]:

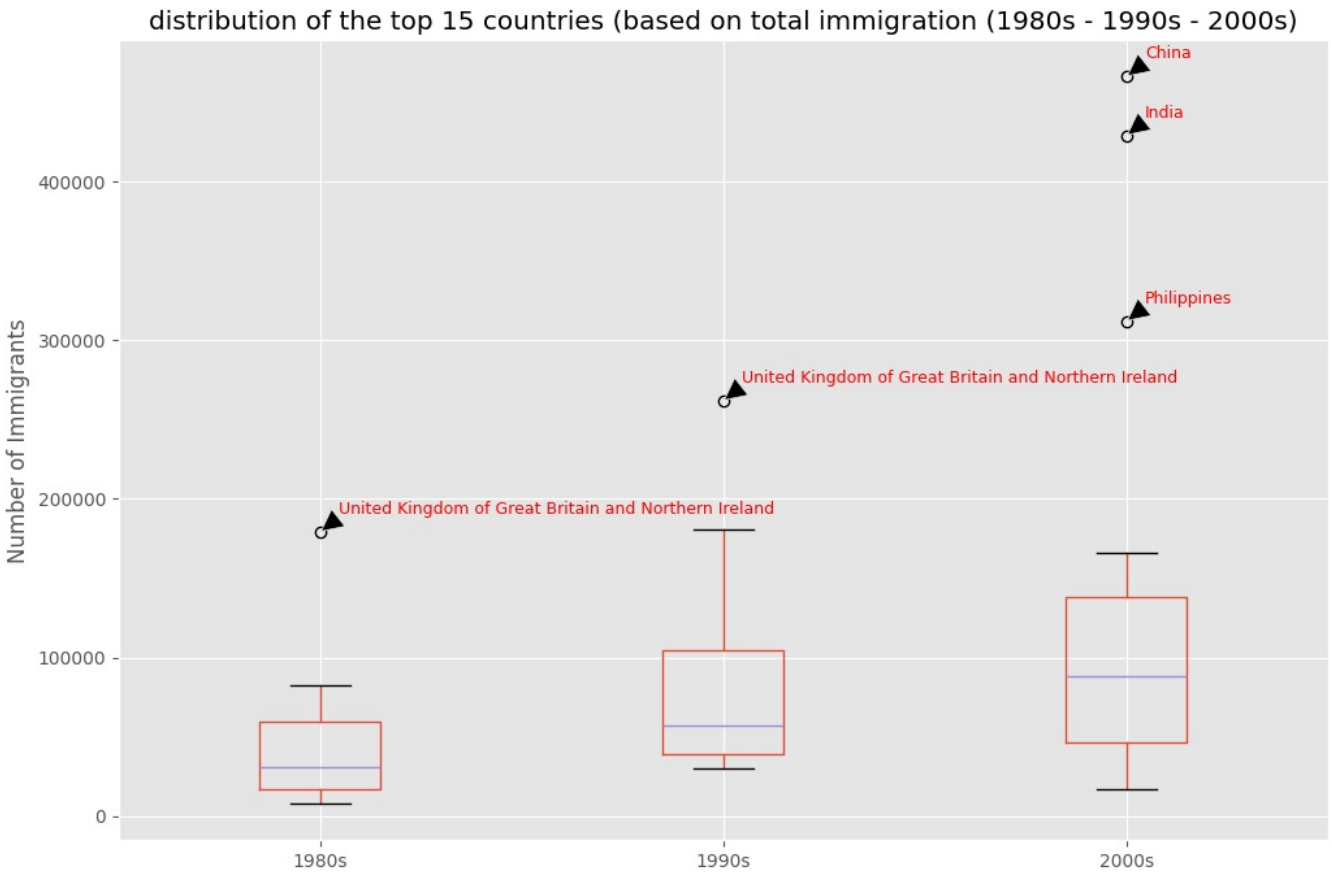
	1980s	1990s	2000s
India	82154	180395	429355
China	32003	161528	466431
United Kingdom of Great Britain and Northern Ireland	179171	261966	110363
Philippines	60764	138482	312145
Pakistan	10591	65302	165707

In [ ]:
```
new_df_withdecades.describe()
```

Out[ ]:

	1980s	1990s	2000s
count	15.000000	15.000000	15.000000
mean	44418.333333	85594.666667	138333.266667
std	44190.676455	68237.560246	145288.871956
min	7613.000000	30028.000000	16775.000000
25%	16698.000000	39259.000000	46754.500000
50%	30638.000000	56915.000000	88133.000000
75%	59183.000000	104451.500000	138035.000000
max	179171.000000	261966.000000	466431.000000

In [ ]:
```
plt.figure(figsize=(12, 8))
boxplot = new_df_withdecades.boxplot()

positions = range(len(new_df_withdecades.columns))

# Identifying and annotating outliers in the df
for pos, column in zip(positions, ['1980s', '1990s', '2000s']):
    Q1 = new_df_withdecades[column].quantile(0.25)
    Q3 = new_df_withdecades[column].quantile(0.75)
    IQR = Q3 - Q1  # Interquartile range

    # setting borders for outliers
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    # Get the outliers
    outliers = new_df_withdecades[(new_df_withdecades[column] < lower_bound) |
                                  (new_df_withdecades[column] > upper_bound)]

    # Annotate each outlier
    for idx in outliers.index:
        plt.annotate(
            idx,
            xy=(pos + 1, outliers.loc[idx, column]),  # 'pos + 1' to match boxplot positions
            xytext=(10, 10),
            textcoords='offset points',
            arrowprops=dict(facecolor='black', shrink=0.05),
            fontsize=9,
            color='red'
        )

plt.title('distribution of the top 15 countries (based on total immigration (1980s - 1990s - 2000s)')
plt.ylabel('Number of Immigrants')
plt.show()
```



In [ ]:
```
new_df_withdecades[new_df_withdecades['2000s'] > 209611]
```

Out[ ]:

	1980s	1990s	2000s
India	82154	180395	429355
China	32003	161528	466431
Philippines	60764	138482	312145

In [ ]:
```
new_df_withdecades[new_df_withdecades['1990s'] > 200000]
```

Out[ ]:

	1980s	1990s	2000s
United Kingdom of Great Britain and Northern Ireland	179171	261966	110363

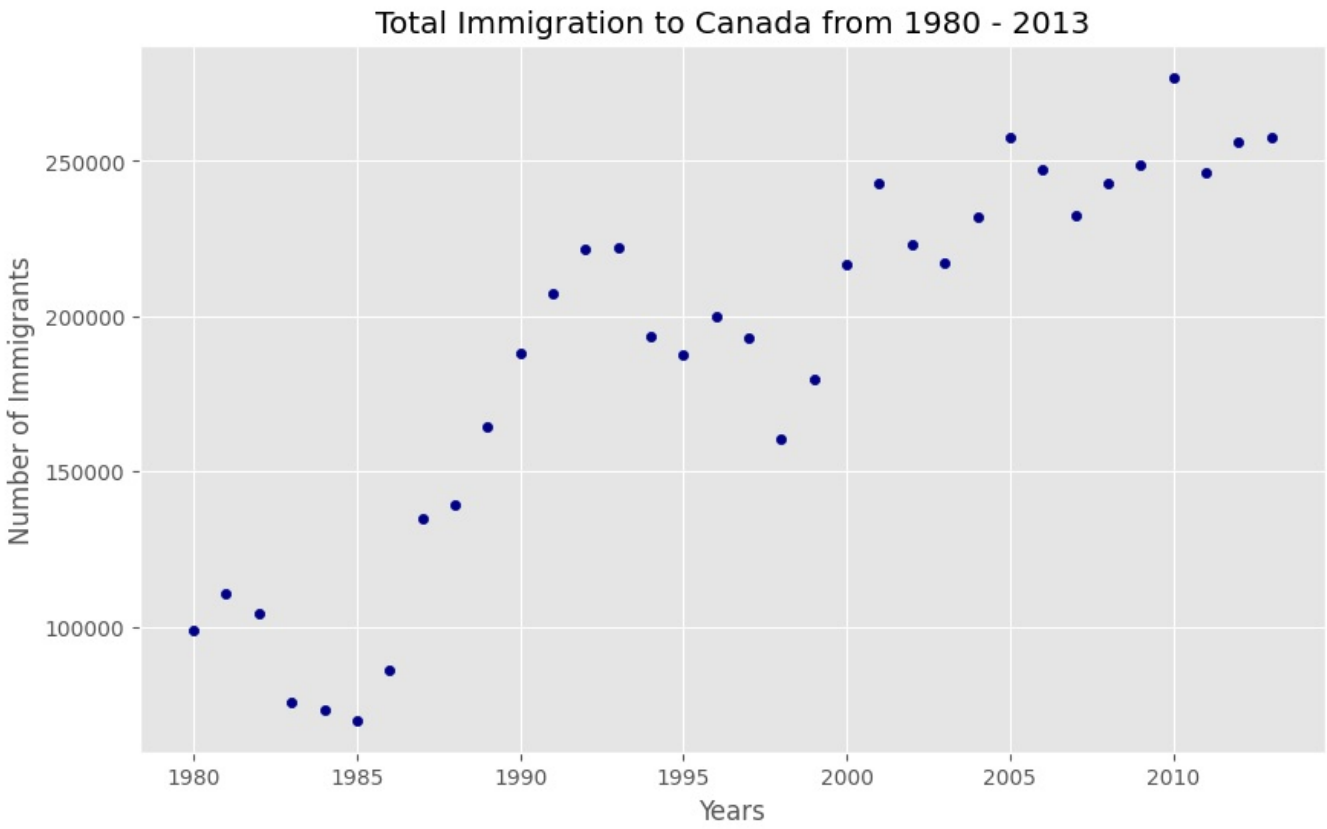
In [ ]:
```
df_tot = pd.DataFrame(df_can[years].sum(axis=0))

df_tot.index = map(int, df_tot.index)

df_tot.reset_index(inplace=True)

df_tot.columns = ['year','total']

df_tot.head()
```

Out[ ]:

	year	total
0	1980	99137
1	1981	110563
2	1982	104271
3	1983	75550
4	1984	73417

In [ ]:
```
df_tot.plot(kind='scatter', x='year', y='total', figsize=(10, 6), color='darkblue')
plt.title('Total Immigration to Canada from 1980 - 2013')
plt.ylabel('Number of Immigrants')
plt.xlabel('Years')
```

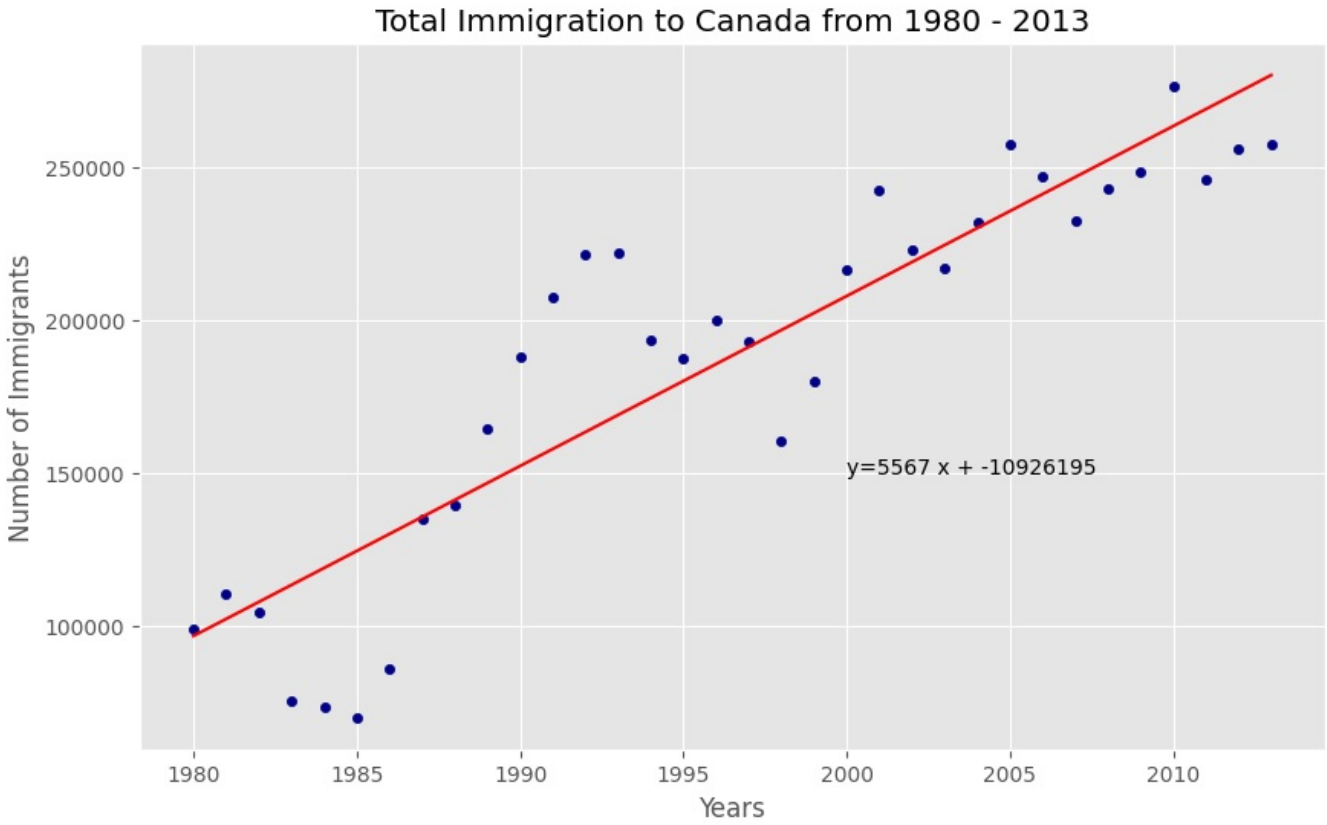
Out[ ]:
```
Text(0.5, 0, 'Years')
```



In [ ]:
```
x = df_tot['year']
y = df_tot['total']
fit = np.polyfit(x , y, deg=1)

fit
```

Out[ ]:
```
array([ 5.56709228e+03, -1.09261952e+07])
```

In [ ]:
```
df_tot.plot(kind='scatter', x='year', y='total', figsize=(10, 6), color='darkblue')

plt.title('Total Immigration to Canada from 1980 - 2013')
plt.ylabel('Number of Immigrants')
plt.xlabel('Years')

plt.plot(x, fit[0] * x + fit[1], color = 'red')
plt.annotate('y={0:.0f} x + {1:.0f}'.format(fit[0], fit[1]), xy=(2000, 150000))

plt.show()

'No. Immigrants = {0:.0f} * Year + {1:.0f}'.format(fit[0], fit[1])
```



Out[ ]:
```
'No. Immigrants = 5567 * Year + -10926195'
```

In [ ]:
```
df_can_trans = df_can[years].transpose()

df_can_trans.index = map(int, df_can_trans.index)

df_can_trans.index.name = 'Year'

df_can_trans.reset_index(inplace=True)

df_can_trans.head()
```

Out[ ]:

	Year	India	China	United Kingdom of Great Britain and Northern Ireland	Philippines	Pakistan	United States of America	Iran (Islamic Republic of)	Sri Lanka	Republic of Korea	...	Kiribati	Vanuatu	Sao Tome and Principe	Tuvalu	American Samoa
0	1980	8880	5123	22045	6051	978	9378	1172	185	1011	...	0	0	0	0	0
1	1981	8670	6682	24796	5921	972	10030	1429	371	1456	...	0	0	0	1	1
2	1982	8147	3308	20620	5249	1201	9074	1822	290	1572	...	0	0	0	0	0
3	1983	7338	1863	10015	4562	900	7100	1592	197	1081	...	1	0	0	0	0
4	1984	5704	1527	10170	3801	668	6661	1977	1086	847	...	0	0	0	1	0

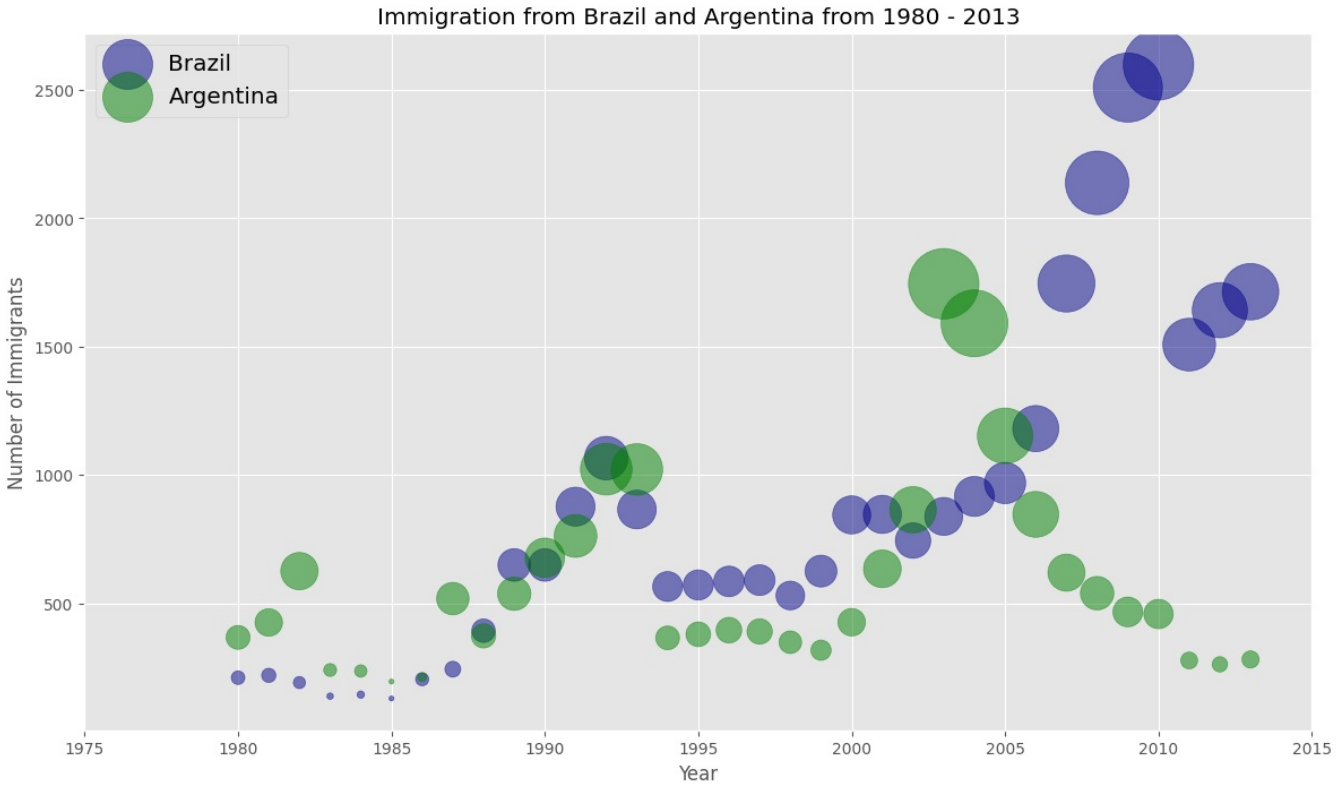
5 rows × 196 columns

In [ ]:
```
norm_brazil = (df_can_trans['Brazil'] - df_can_trans['Brazil'].min()) / (df_can_trans['Brazil'].max() - df_can_

norm_argentina = (df_can_trans['Argentina'] - df_can_trans['Argentina'].min()) / (df_can_trans['Argentina'].max
```

In [ ]:
```
## BRAZIL
ax0 = df_can_trans.plot(kind='scatter',
                        x='Year',
                        y='Brazil',
                        figsize=(14, 8),
                        alpha = 0.5,
                        color = 'darkblue',
                        s = norm_brazil * 2000 + 10,
                        xlim=(1975,2015)
                        )


##ARGENTINA
ax1 = df_can_trans.plot(kind='scatter',
                        x='Year',
                        y='Argentina',
                        alpha = 0.5,
```

```
                      figsize=(14, 8),
                      color ='Green',
                      s = norm_argentina * 2000 + 10,
                      ax= ax0
                      )

ax0.set_title('Immigration from Brazil and Argentina from 1980 - 2013')
ax0.set_ylabel('Number of Immigrants')
ax0.legend(['Brazil', 'Argentina'], loc = 'upper left', fontsize='x-large')
```

Out[ ]:
```
<matplotlib.legend.Legend at 0x22fc73a5850>
```



In [ ]:
```
df_DSN = df_can.loc[['Denmark','Sweden','Norway'], :]

df_DSN
```

Out[ ]:

	Continents	Region	Regions Development situation	1980	1981	1982	1983	1984	1985	1986	...	2005	2006	2007	2008	2009	2010	2011	201
Denmark	Europe	Northern Europe	Developed regions	272	293	299	106	93	73	93	...	62	101	97	108	81	92	93	9
Sweden	Europe	Northern Europe	Developed regions	281	308	222	176	128	158	187	...	205	139	193	165	167	159	134	14
Norway	Europe	Northern Europe	Developed regions	116	77	106	51	31	54	56	...	57	53	73	66	75	46	49	5

3 rows × 38 columns

In [ ]:
```
# compute the proportion of each category with respect to the total
total_values = sum(df_DSN['Total'])
category_proportions = [(float(value) / total_values) for value in df_DSN['Total']]

# print out proportions
for i, proportion in enumerate(category_proportions):
    print (df_DSN.index.values[i] + ': ' + str(proportion))
```

```
Denmark: 0.32255663965602777
Sweden: 0.48503390110798744
Norway: 0.1924094592359848
```

In [ ]:
```
width = 40
height = 10

total_num_tiles_waffle = width * height

print('Total number of tiles is', total_num_tiles_waffle)
```

```
Total number of tiles is 400
```

In [ ]:
```
tiles_per_cat =[round(proportion * total_num_tiles_waffle) for proportion in category_proportions]

for i, tiles in enumerate(tiles_per_cat):
```

```
    print(df_DSN.index.values[i] + ': ' + str(tiles))
```

```
Denmark: 129
Sweden: 194
Norway: 77
```

In [ ]:

```
waffle_chart = np.zeros((height, width))

category_index = 0
title_index = 0

for col in range(width):
    for row in range(height):
        title_index += 1

        if title_index > sum(tiles_per_cat[0 : category_index]):
            category_index +=1
            print(title_index)


        waffle_chart[row,col] = category_index

print('Waffle chart Created!')
```

```
1
130
324
Waffle chart Created!
```

In [ ]:

```
waffle_chart
```

Out[ ]:

```
array([[1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 2., 2., 2.,
        2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2.,
        2., 3., 3., 3., 3., 3., 3., 3.],
       [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 2., 2., 2.,
        2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2.,
        2., 3., 3., 3., 3., 3., 3., 3.],
       [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 2., 2., 2.,
        2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2.,
        2., 3., 3., 3., 3., 3., 3., 3.],
       [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 2., 2., 2.,
        2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2.,
        3., 3., 3., 3., 3., 3., 3., 3.],
       [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 2., 2., 2.,
        2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2.,
        3., 3., 3., 3., 3., 3., 3., 3.],
       [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 2., 2., 2.,
        2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2.,
        3., 3., 3., 3., 3., 3., 3., 3.],
       [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 2., 2., 2.,
        2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2.,
        3., 3., 3., 3., 3., 3., 3., 3.],
       [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 2., 2., 2.,
        2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2.,
        3., 3., 3., 3., 3., 3., 3., 3.],
       [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 2., 2., 2.,
        2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2.,
        3., 3., 3., 3., 3., 3., 3., 3.],
       [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 2., 2., 2., 2.,
        2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2.,
        3., 3., 3., 3., 3., 3., 3., 3.]])
```

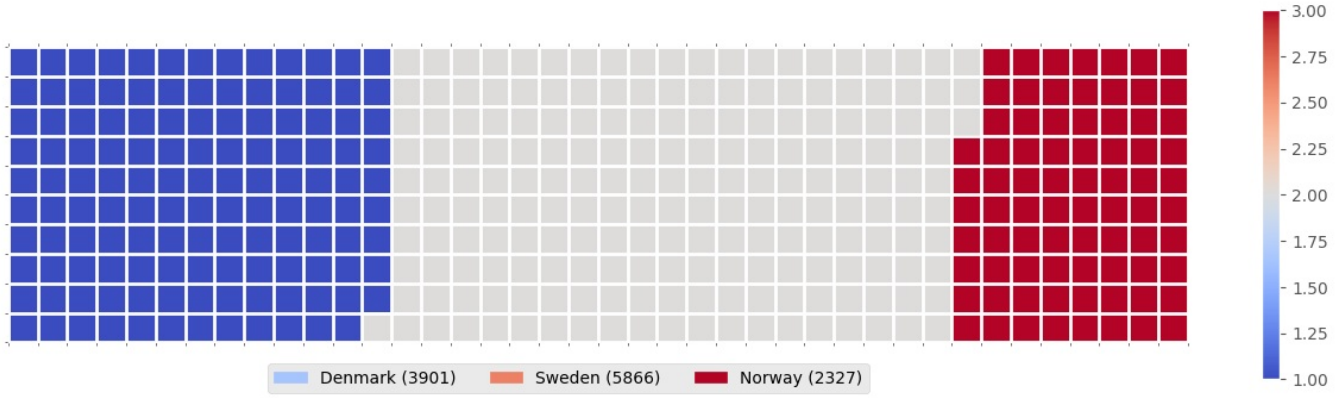
In [ ]:

```
import matplotlib.patches as mpatches

fig = plt.figure()

colormap = plt.cm.coolwarm
plt.matshow(waffle_chart, cmap = colormap)
plt.colorbar()

ax = plt.gca()

ax.set_xticks(np.arange(-.5, (width), 1), minor=True)
ax.set_yticks(np.arange(-.5, (height), 1), minor=True)

ax.grid(which='minor', color='w', linestyle='-', linewidth=2)

plt.xticks([])
plt.yticks([])

values_cumsum = np.cumsum(df_DSN['Total'])
total_values = values_cumsum[len(values_cumsum)- 1]


legend_handles = []
for i, category in enumerate(df_DSN.index.values):
    label_str = category +' (' + str(df_DSN['Total'][i]) + ')'
    color_val = colormap(float(values_cumsum[i])/total_values)
    legend_handles.append(mpatches.Patch(color=color_val, label=label_str))
```

```
plt.legend(handles=legend_handles,
           loc='lower center',
           ncol=len(df_DSN.index.values),
           bbox_to_anchor=(0., -0.2, 0.95, .1)
          )
```

Out[ ]:

```
<matplotlib.legend.Legend at 0x22fcb280510>

<Figure size 640x480 with 0 Axes>
```



In [ ]:

```
import urllib.request

url = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DV0101EN-Sk
urllib.request.urlretrieve(url, 'world_countries.json')

print('GeoJSON file downloaded!')
```

```
GeoJSON file downloaded!
```

```
'wget' is not recognized as an internal or external command,
operable program or batch file.
```

In [ ]:

```
import plotly.express as px
import pandas as pd

# Assuming df_can is already defined and contains the necessary data

# Create the 'Total' column if not already created
df_can['Total'] = df_can.sum(axis=1)

# Create the choropleth map using Plotly Express
fig = px.choropleth(
    df_can,
    locations='index',  # Column in your DataFrame containing country names or ISO-3 country codes
    locationmode='country names',  # Set the mode to match country names
    color='Total',  # Data to color the map by
    hover_name='index',  # Column to use for hover information
    color_continuous_scale='YlOrRd',  # Color scale
    labels={'Total': 'Total Immigration'},  # Labels for the color bar
    title='Immigration to Canada by Country'  # Title of the map
)

# Update layout for better appearance
fig.update_layout(
    geo=dict(
        showframe=False,
        showcoastlines=False,
        projection_type='equirectangular'
    ),
    margin={"r":0,"t":50,"l":0,"b":0}  # Adjust margins
)

# Show the map
fig.show()
```

```
C:\Users\aliki\AppData\Local\Temp\ipykernel_17264\1951021561.py:7: FutureWarning:

Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future vers
ion this will raise TypeError.  Select only valid columns before calling the reduction.
```