```
In [ ]:
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Ex. 1: Vacation search results

Poland

```
In [ ]:
PL = pd.read_csv('vacationPL.csv', skiprows = 3, names = ['Date', 'PL'], index_col = 'Date', parse_dates = True)
PL
```

Out[ ]:

Date	PL
2004-01-01	42
2004-02-01	100
2004-03-01	66
2004-04-01	90
2004-05-01	42
...	...
2021-11-01	87
2021-12-01	69
2022-01-01	70
2022-02-01	86
2022-03-01	48

219 rows × 1 columns

UK

```
In [ ]:
UK = pd.read_csv('vacationUK.csv', skiprows = 3, names = ['Date', 'UK'], index_col = 'Date', parse_dates = True)
UK
```

Out[ ]:

Date	UK
2004-01-01	38
2004-02-01	92
2004-03-01	35
2004-04-01	33
2004-05-01	33
...	...
2021-11-01	24
2021-12-01	42
2022-01-01	31
2022-02-01	37
2022-03-01	22

219 rows × 1 columns

USA

```
In [ ]:
USA = pd.read_csv('vacationUSA.csv', skiprows = 3, names = ['Date', 'USA'], index_col = 'Date', parse_dates = True)
USA
```

Out[ ]:

Date	USA
2004-01-01	87
2004-02-01	76
2004-03-01	71
2004-04-01	66
2004-05-01	65
...	...
2021-11-01	48
2021-12-01	61
2022-01-01	52
2022-02-01	53
2022-03-01	50

219 rows × 1 columns

Merge

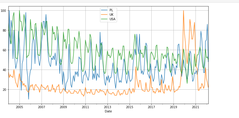
```
In [ ]:
result = pd.concat([PL, UK, USA], axis = 1)
result
```

Out[ ]:

Date	PL	UK	USA
2004-01-01	42	38	87
2004-02-01	100	92	76
2004-03-01	66	35	71
2004-04-01	90	33	66
2004-05-01	42	33	65
...	...	...	...
2021-11-01	87	24	48
2021-12-01	69	42	61
2022-01-01	70	31	52
2022-02-01	86	37	53
2022-03-01	48	22	50

219 rows × 3 columns

```
In [ ]:
result['PL'].plot(figsize = (12, 8)).autoscale(axis = 'x', tight = True)
result['UK'].plot()
result['USA'].plot()
plt.legend()
plt.grid()
```



Descriptive statistics

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

Out[ ]:

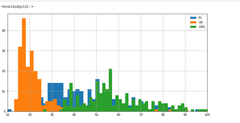
	PL	UK	USA
count	219.000000	219.000000	219.000000
mean	45.788087	34.543379	58.301370
std	17.214667	14.441330	14.930349
min	10.000000	13.000000	26.000000
25%	33.000000	17.000000	48.000000
50%	42.000000	33.000000	56.000000
75%	54.000000	44.500000	67.000000
max	100.000000	100.000000	100.000000

Histograms

```
In [ ]:
result['PL'].hist(bins = 50, legend = True, figsize = (12, 8)).autoscale(axis = 'x', tight = True)
result['UK'].hist(bins = 50, legend = True)
result['USA'].hist(bins = 50, legend = True)
```

Out[ ]:

```
<AxesSubplot:>
```

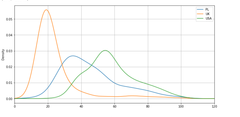


Kernel densities

```
In [ ]:
result['PL'].plot.kde(figsize = (12, 8)).autoscale(axis = 'x', tight = True)
result['UK'].plot.kde()
result['USA'].plot.kde()
plt.legend()
plt.grid()
plt.xlim(0, 120)
```

Out[ ]:

```
(0.0, 120.0)
```



Ex. 2: Average temperature dataset

Import

```
In [ ]:
T = pd.read_csv('temperature.csv', skiprows = 5, names = ['Date', 'Value', 'Anomaly'])
T
```

Out[ ]:

	Date	Value	Anomaly
0	193501	39.1	7.7
1	193502	33.3	-2.0
2	193503	47.3	2.7
3	193504	52.7	-3.8
4	193505	48.7	2.3
...	...	...	...
992	202109	73.5	3.3
993	202110	63.9	4.7
994	202111	46.2	0.9
995	202112	45.7	11.1
996	202201	29.0	-1.4

997 rows × 3 columns

Change -99 to NaN

```
In [ ]:
T.replace(-99.9, np.nan, inplace = True)
#check if it's -99
T.tail(110)
# Not sure about the anomaly of 2013-03, it's still for value -99, should I fix it?
```

Out[ ]:

	Date	Value	Anomaly
887	201212	41.2	6.6
888	201301	35.1	-4.7
889	201302	35.1	-0.2
890	201303	NaN	-143.8
891	201304	54.9	-1.6
...	...	...	...
992	202109	73.5	3.3
993	202110	63.9	4.7
994	202111	46.2	0.9
995	202112	45.7	11.1
996	202201	29.0	-1.4

110 rows × 3 columns

Interpolate NaN's

```
In [ ]:
T.interpolate(inplace = True)
#check if was NaN
T.tail(110)
```

Out[ ]:

	Date	Value	Anomaly
887	201212	41.2	6.6
888	201301	35.1	-4.7
889	201302	35.1	-0.2
890	201303	45.0	-143.8
891	201304	54.9	-1.6
...	...	...	...
992	202109	73.5	3.3
993	202110	63.9	4.7
994	202111	46.2	0.9
995	202112	45.7	11.1
996	202201	29.0	-1.4

110 rows × 3 columns

Change index to datetime format

```
In [ ]:
T['Date'] = pd.to_datetime(T['Date'], format = '%Ym')
T.set_index('Date', inplace = True)
T
```

Out[ ]:

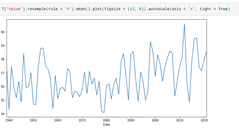
Date	Value	Anomaly
1935-01-01	39.1	7.7
1935-02-01	33.3	-2.0
1935-03-01	47.3	2.7
1935-04-01	52.7	-3.8
1935-05-01	48.7	2.3
...	...	...
2021-09-01	73.5	3.3
2021-10-01	63.9	4.7
2021-11-01	46.2	0.9
2021-12-01	45.7	11.1
2022-01-01	29.0	-1.4

997 rows × 2 columns

Average temperature for each year

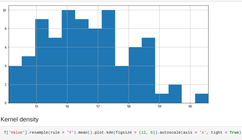
```
In [ ]:
# Annoying last value 2022-01-01 so the average is not just from 1 month - makes the graph look bad
T = T.drop('2022-01-01')['Value'].to_frame()
```

```
In [ ]:
T['Value'].resample(rule = 'Y').mean().plot(figsize = (12, 8)).autoscale(axis = 'x', tight = True)
```



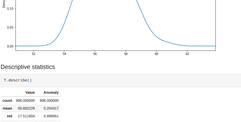
Histogram

```
In [ ]:
T['Value'].resample(rule = 'Y').mean().hist(bins = 15, figsize = (12, 8)).autoscale(axis = 'x', tight = True)
```



Kernel density

```
In [ ]:
T['Value'].resample(rule = 'Y').mean().plot.kde(figsize = (12, 8)).autoscale(axis = 'x', tight = True)
```



Descriptive statistics

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

Out[ ]:

	Value	Anomaly
count	996.000000	996.000000
mean	56.882209	0.204317
std	17.911854	5.299951
min	-14.400000	-143.800000
25%	40.675000	-1.900000
50%	58.400000	0.400000
75%	73.425000	2.700000
max	97.500000	19.900000