Python: Lab 5 - NumPy, Pandas

22130323: Vũ Anh Việt

```
from google.colab import drive
drive.mount('/content/gdrive')
%cd '/content/gdrive/MyDrive/TestPython/'
```

```
Mounted at /content/gdrive
/content/gdrive/MyDrive/TestPython
```

```
# code
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import os, glob
addFiles = glob.glob(os.path.join('https://drive.google.com/drive/u/1/my-drive',"*.csv"))
for file in addFiles:
  df = pd.read_csv(file)
```

Task 1 (Using **NumPy**)

Task 1.1. Generate a numpy array with 100 integer values from 1 to 20 without using any loop.

```
array1 = np.tile(np.arange(1,21),5)
print(array1)
```

```
[ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20  1  2  3  4
  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20  1  2  3  4  5  6  7  8
  9 10 11 12 13 14 15 16 17 18 19 20  1  2  3  4  5  6  7  8  9 10 11 12
 13 14 15 16 17 18 19 20  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
 17 18 19 20]
```

Task 1.2. Replace all elements of the array generated in Task 1.1. that are greater than a specified value (***v***) by ***v***

```
np.where(np.tile(np.arange(1,21),5)>10,10,array1)
```

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 10, 10, 10, 10, 10, 10, 10,
       10, 10, 10,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 10, 10, 10, 10,
       10, 10, 10, 10, 10, 10,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 10,
       10, 10, 10, 10, 10, 10, 10, 10,  1,  2,  3,  4,  5,  6,  7,  8,
        9, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10,  1,  2,  3,  4,  5,
        6,  7,  8,  9, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10])
```

Task 1.3. Find the most frequent value in the array obtained from Task 1.2

```
np.bincount(np.where(np.tile(np.arange(1,21),5)>10,10,np.tile(np.arange(1,21),5))).argmax()
```

```
10
```

Task 1.4. Replace all **odd numbers** generated in Task 1.1. by ***-1***

```
np.where(array1%2==1,-1,array1)
```

```
array([-1,  2, -1,  4, -1,  6, -1,  8, -1, 10, -1, 12, -1, 14, -1, 16, -1,
       18, -1, 20, -1,  2, -1,  4, -1,  6, -1,  8, -1, 10, -1, 12, -1, 14,
       -1, 16, -1, 18, -1, 20, -1,  2, -1,  4, -1,  6, -1,  8, -1, 10, -1,
       12, -1, 14, -1, 16, -1, 18, -1, 20, -1,  2, -1,  4, -1,  6, -1,  8,
       -1, 10, -1, 12, -1, 14, -1, 16, -1, 18, -1, 20, -1,  2, -1,  4, -1,
        6, -1,  8, -1, 10, -1, 12, -1, 14, -1, 16, -1, 18, -1, 20])
```

Task 2 (Using **Pandas**)

Task 2.1. Load **dataset1** and display 10 first rows

```
data_table = pd.read_csv('dataset1.csv')
data_table.head(10)
```

	location	continent	population	life_expectancy	hospital_beds_per_thousand	gdp_per_capita
0	Afghanistan	Asia	38928341.0	64.83	0.50	1803.987
1	Albania	Europe	2877800.0	78.57	2.89	11803.431
2	Algeria	Africa	43851043.0	76.88	1.90	13913.839
3	Andorra	Europe	77265.0	83.73	NaN	NaN
4	Angola	Africa	32866268.0	61.15	NaN	5819.495
5	Anguilla	North America	15002.0	81.88	NaN	NaN
6	Antigua and Barbuda	North America	97928.0	77.02	3.80	21490.943
7	Argentina	South America	45195777.0	76.67	5.00	18933.907
8	Armenia	Asia	2963234.0	75.09	4.20	8787.580
9	Aruba	North America	106766.0	76.29	NaN	35973.781

Task 2.2. List all continents in the dataset

```
data_table['continent'].unique()
```

```
array(['Asia', 'Europe', 'Africa', 'North America', 'South America',
       'Oceania'], dtype=object)
```

Task 2.3. Get 10 countries that have highest life_expectancy as a dataframe

```
dt1=data_table.sort_values(by='life_expectancy',ascending=False).head(10)
dt1['location'].unique()
pd.DataFrame(dt1)
```

	location	continent	population	life_expectancy	hospital_beds_per_thousand	gdp_per_capita
127	Monaco	Europe	39244.0	86.75	13.80	NaN
162	San Marino	Europe	33938.0	84.97	3.80	56861.470
87	Hong Kong	Asia	7496988.0	84.86	NaN	56054.920
99	Japan	Asia	126476458.0	84.63	13.05	39002.223
37	Cayman Islands	North America	65720.0	83.92	NaN	49903.029
183	Switzerland	Europe	8654618.0	83.78	4.53	57410.166
3	Andorra	Europe	77265.0	83.73	NaN	NaN
169	Singapore	Asia	5850343.0	83.62	2.40	85535.383
177	Spain	Europe	46754783.0	83.56	2.97	34272.360
97	Italy	Europe	60461828.0	83.51	3.18	35220.084

Task 2.4. Get 10 countries with the highest GDP per capita, among the countries with population greater than 100 million as a dataframe

```
dt2= data_table.sort_values(by='gdp_per_capita',ascending=False).head(10)
dt2[dt2['population']>100000.000]
dt2['location'].unique()
pd.DataFrame(dt2)
```

	location	continent	population	life_expectancy	hospital_beds_per_thousand	gdp_per_capita
155	Qatar	Asia	2881060.0	80.23	1.20	116935.600
115	Luxembourg	Europe	625976.0	82.25	4.51	94277.965
169	Singapore	Asia	5850343.0	83.62	2.40	85535.383
29	Brunei	Asia	437483.0	75.86	2.70	71809.251
94	Ireland	Europe	4937796.0	82.30	2.96	67335.293
...	...	...	...	...	...	...
185	Taiwan	Asia	23816775.0	80.46	NaN	NaN
194	Turks and Caicos Islands	North America	38718.0	80.22	NaN	NaN
200	United States Virgin Islands	North America	104423.0	80.58	NaN	NaN
203	Vatican	Europe	809.0	75.12	NaN	NaN
206	Western Sahara	Africa	597330.0	70.26	NaN	NaN

210 rows × 6 columns

Task 2.5. Report the the number countries in each continent as a dataframe

```
dt3=data_table.groupby('continent')['location'].nunique()
pd.DataFrame(dt3)
```

continent	location
Africa	55
Asia	47
Europe	51
North America	36
Oceania	8
South America	13

Task 2.6. Report the total population of each continent

```
dt4 = data_table.groupby('continent')['population'].sum()
pd.DataFrame(dt4)
```

continent	population
Africa	1.339424e+09
Asia	4.607388e+09
Europe	7.485062e+08
North America	5.912425e+08
Oceania	4.095832e+07
South America	4.304611e+08

Task 3. For a given set of csv files concerning **weather** information measured by 12 sensors

Task 3.1. Load all csv files into a single dataframe

```
csv_files = glob.glob('weather/*.csv')
combined_df = pd.DataFrame()
for csv_file in csv_files:
    df = pd.read_csv(csv_file)
    combined_df = pd.concat([combined_df, df])
print(combined_df)
```

```
           No  year  month  day  hour  PM2.5  PM10   SO2   NO2     CO    O3  \
0           1  2013      3    1     0    4.0   4.0   4.0   7.0  300.0  77.0
1           2  2013      3    1     1    8.0   8.0   4.0   7.0  300.0  77.0
2           3  2013      3    1     2    7.0   7.0   5.0  10.0  300.0  73.0
3           4  2013      3    1     3    6.0   6.0  11.0  11.0  300.0  72.0
4           5  2013      3    1     4    3.0   3.0  12.0  12.0  300.0  72.0
...       ...   ...    ...  ...   ...    ...   ...   ...   ...    ...   ...
35059   35060  2017      2   28    19   11.0  27.0   4.0  20.0  300.0  81.0
35060   35061  2017      2   28    20   15.0  43.0   6.0  55.0  500.0  45.0
35061   35062  2017      2   28    21   13.0  35.0   7.0  48.0  500.0  48.0
35062   35063  2017      2   28    22   12.0  31.0   5.0  47.0  500.0  50.0
35063   35064  2017      2   28    23    7.0  25.0   6.0  86.0  700.0  11.0

       TEMP    PRES  DEWP  RAIN   wd  WSPM       station
0      -0.7  1023.0 -18.8   0.0  NNW   4.4  Aotizhongxin
1      -1.1  1023.2 -18.2   0.0    N   4.7  Aotizhongxin
2      -1.1  1023.5 -18.2   0.0  NNW   5.6  Aotizhongxin
3      -1.4  1024.5 -19.4   0.0   NW   3.1  Aotizhongxin
4      -2.0  1025.2 -19.5   0.0    N   2.0  Aotizhongxin
...     ...     ...   ...   ...  ...   ...           ...
35059  12.6  1011.9 -14.3   0.0    N   2.0        Wanliu
35060   9.4  1012.3 -11.9   0.0  WSW   1.0        Wanliu
35061   8.7  1012.8 -13.7   0.0    N   1.1        Wanliu
35062   7.8  1012.9 -12.6   0.0  NNE   1.0        Wanliu
35063   7.0  1012.6 -11.2   0.0   NE   1.1        Wanliu

[420768 rows x 18 columns]
```

Task 3.2. Make a statistic related to the merged dataset (see the format below)

No.	station	#Records	From date	To date
1	Aotizhongxin	35,064	2013-01-03T00:00:00.000000000	2017-02-28T23:00:00.000000000
1	Changping	35,064	2013-01-03T00:00:00.000000000	2017-02-28T23:00:00.000000000
1	Dingling	35,064	2013-01-03T00:00:00.000000000	2017-02-28T23:00:00.000000000
1	Dongsi	35,064	2013-01-03T00:00:00.000000000	2017-02-28T23:00:00.000000000
1	Guanyuan	35,064	2013-01-03T00:00:00.000000000	2017-02-28T23:00:00.000000000
1	Gucheng	35,064	2013-01-03T00:00:00.000000000	2017-02-28T23:00:00.000000000
1	Huairou	35,064	2013-01-03T00:00:00.000000000	2017-02-28T23:00:00.000000000
1	Nongzhanguan	35,064	2013-01-03T00:00:00.000000000	2017-02-28T23:00:00.000000000
1	Shunyi	35,064	2013-01-03T00:00:00.000000000	2017-02-28T23:00:00.000000000
1	Tiantan	35,064	2013-01-03T00:00:00.000000000	2017-02-28T23:00:00.000000000
1	Wanliu	35,064	2013-01-03T00:00:00.000000000	2017-02-28T23:00:00.000000000
1	Wanshouxigong	35,064	2013-01-03T00:00:00.000000000	2017-02-28T23:00:00.000000000

```
statistics = combined_df.groupby('station').agg(
    Records=('station', 'size'),
    From_date=('year', lambda x: f"{x.min()}-01-01T00:00:00.000000000"),
    To_date=('year', lambda x: f"{x.max()}-12-31T23:00:00.000000000")
).reset_index()
statistics.insert(0, 'No.', range(1, 1 + len(statistics)))
print(statistics)
```

```
    No.        station  Records                    From_date  \
0     1   Aotizhongxin    35064  2013-01-01T00:00:00.000000000
1     2      Changping    35064  2013-01-01T00:00:00.000000000
2     3       Dingling    35064  2013-01-01T00:00:00.000000000
3     4         Dongsi    35064  2013-01-01T00:00:00.000000000
4     5       Guanyuan    35064  2013-01-01T00:00:00.000000000
5     6        Gucheng    35064  2013-01-01T00:00:00.000000000
6     7        Huairou    35064  2013-01-01T00:00:00.000000000
7     8   Nongzhanguan    35064  2013-01-01T00:00:00.000000000
8     9         Shunyi    35064  2013-01-01T00:00:00.000000000
9    10        Tiantan    35064  2013-01-01T00:00:00.000000000
10   11         Wanliu    35064  2013-01-01T00:00:00.000000000
11   12  Wanshouxigong    35064  2013-01-01T00:00:00.000000000

                         To_date
0   2017-12-31T23:00:00.000000000
1   2017-12-31T23:00:00.000000000
2   2017-12-31T23:00:00.000000000
3   2017-12-31T23:00:00.000000000
4   2017-12-31T23:00:00.000000000
5   2017-12-31T23:00:00.000000000
6   2017-12-31T23:00:00.000000000
7   2017-12-31T23:00:00.000000000
8   2017-12-31T23:00:00.000000000
9   2017-12-31T23:00:00.000000000
10  2017-12-31T23:00:00.000000000
11  2017-12-31T23:00:00.000000000
```

Task 3.3. Display the percentage of missing values for each attribute

```
missing_values = missing_values.round(2)
for col, pct in missing_values.items():
    print(f"{col}: {pct}% missing")
```

```
No: 0.0% missing
year: 0.0% missing
month: 0.0% missing
day: 0.0% missing
hour: 0.0% missing
PM2.5: 2.08% missing
PM10: 1.53% missing
SO2: 2.14% missing
NO2: 2.88% missing
CO: 4.92% missing
O3: 3.16% missing
TEMP: 0.09% missing
PRES: 0.09% missing
DEWP: 0.1% missing
RAIN: 0.09% missing
```

```
wd: 0.43% missing
WSPM: 0.08% missing
station: 0.0% missing
```

Task 3.4. Remove rows having missing values (i.e., NaN, empty, or not a number, ...) in the dataset obtained in the Task 3.1

```
cleaned_df = combined_df.dropna()
print(f"Original DataFrame shape: {combined_df.shape}")
print(f"Cleaned DataFrame shape: {cleaned_df.shape}")
```

```
Original DataFrame shape: (420768, 18)
Cleaned DataFrame shape: (382168, 18)
```

Task 3.5. Report the min, max values of the following attributes for each station ("PM2.5","PM10","SO2","NO2","CO","O3","TEMP","PRES","DEWP","RAIN")

```
attributes = ["PM2.5", "PM10", "SO2", "NO2", "CO", "O3", "TEMP", "PRES", "DEWP", "RAIN"]
min_values = combined_df.groupby('station')[attributes].min().reset_index()
max_values = combined_df.groupby('station')[attributes].max().reset_index()
report = pd.merge(min_values, max_values, on='station', suffixes=('_min', '_max'))
for station in report['station']:
    station_data = report[report['station'] == station]
    print(f"Station: {station}")
    for attribute in attributes:
        min_val = station_data[f"{attribute}_min"].values[0]
        max_val = station_data[f"{attribute}_max"].values[0]
        print(f"  {attribute}: Min = {min_val}, Max = {max_val}")
    print("\n")
```

```
Station: Aotizhongxin
  PM2.5: Min = 3.0, Max = 898.0
  PM10: Min = 2.0, Max = 984.0
  SO2: Min = 0.2856, Max = 341.0
  NO2: Min = 2.0, Max = 290.0
  CO: Min = 100.0, Max = 10000.0
  O3: Min = 0.2142, Max = 423.0
  TEMP: Min = -16.8, Max = 40.5
  PRES: Min = 985.9, Max = 1042.0
  DEWP: Min = -35.3, Max = 28.5
  RAIN: Min = 0.0, Max = 72.5


Station: Changping
  PM2.5: Min = 2.0, Max = 882.0
  PM10: Min = 2.0, Max = 999.0
  SO2: Min = 0.2856, Max = 310.0
  NO2: Min = 1.8477, Max = 226.0
  CO: Min = 100.0, Max = 10000.0
  O3: Min = 0.2142, Max = 429.0
  TEMP: Min = -16.6, Max = 41.4
  PRES: Min = 982.4, Max = 1036.5
  DEWP: Min = -35.1, Max = 27.2
  RAIN: Min = 0.0, Max = 52.1


Station: Dingling
  PM2.5: Min = 3.0, Max = 881.0
  PM10: Min = 2.0, Max = 905.0
  SO2: Min = 0.2856, Max = 156.0
  NO2: Min = 1.0265, Max = 205.0
  CO: Min = 100.0, Max = 10000.0
  O3: Min = 0.2142, Max = 500.0
  TEMP: Min = -16.6, Max = 41.4
  PRES: Min = 982.4, Max = 1036.5
  DEWP: Min = -35.1, Max = 27.2
  RAIN: Min = 0.0, Max = 52.1


Station: Dongsi
  PM2.5: Min = 3.0, Max = 737.0
  PM10: Min = 2.0, Max = 955.0
  SO2: Min = 0.2856, Max = 300.0
```

```
  NO2: Min = 2.0, Max = 258.0
  CO: Min = 100.0, Max = 10000.0
  O3: Min = 0.6426, Max = 1071.0
  TEMP: Min = -16.8, Max = 41.1
  PRES: Min = 987.1, Max = 1042.0
  DEWP: Min = -35.3, Max = 28.8
  RAIN: Min = 0.0, Max = 46.4


Station: Guanyuan
  PM2.5: Min = 2.0, Max = 680.0
  PM10: Min = 2.0, Max = 999.0
  SO2: Min = 1.0, Max = 293.0
  NO2: Min = 2.0, Max = 270.0
  CO: Min = 100.0, Max = 10000.0
```