Just IT

B2W group

Apprenticeships | Training | Recruitment

Data Technician

Name:

Course Date:

Table of contents

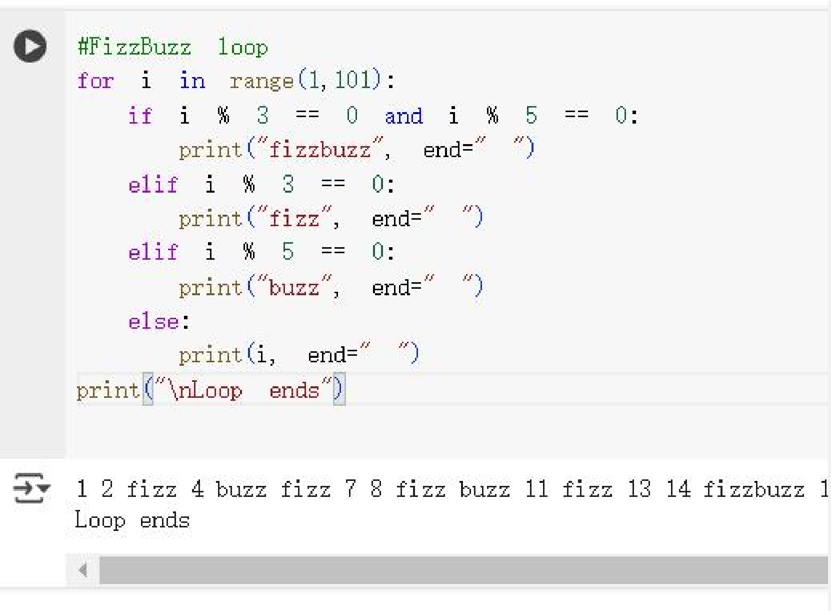
Day 2: Task 1

It is a common software development interview question to create the below with a certain programming language. Create the below using Python syntax, test it and past the completed syntax and output below.

FizzBuzz:

Go through the integers from 1 to 100.
If a number is divisible by 3, print "fizz."
If a number is divisible by 5, print "buzz."
If a number is both divisible by 3 and by 5, print "fizzbuzz."
Otherwise, print just the number.

Paste your completed work to the right



```
#FizzBuzz loop
for i in range(1,101):
    if i % 3 == 0 and i % 5 == 0:
        print("fizzbuzz", end=" ")
    elif i % 3 == 0:
        print("fizz", end=" ")
    elif i % 5 == 0:
        print("buzz", end=" ")
    else:
        print(i, end=" ")
print("\nLoop ends")
```

```
1 2 fizz 4 buzz fizz 7 8 fizz buzz 11 fizz 13 14 fizzbuzz 1
Loop ends
```

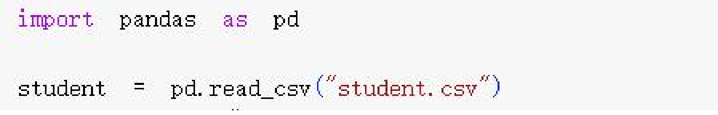
Day 3: Task 1

Download the ‘student.csv’, complete the below exercises as a group and paste your input and output. Although this is a group activity, everyone should have the below answered so it supports your portfolio:

Exercise 1: Loading and Exploring the Data

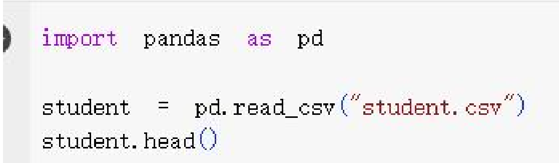
1. Question: "Write the code to read a CSV file into a Pandas DataFrame."
2. Question: "Write the code to display the first 5 rows of the DataFrame."
3. Question: "Write the code to get the information about the DataFrame."
4. Question: "Write the code to get summary statistics for the DataFrame."

1.



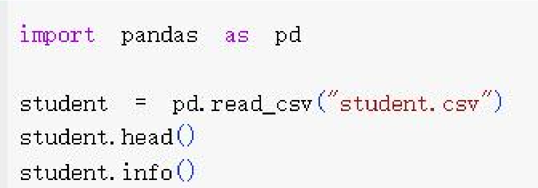
```
import pandas as pd

student = pd.read_csv("student.csv")
```

2.

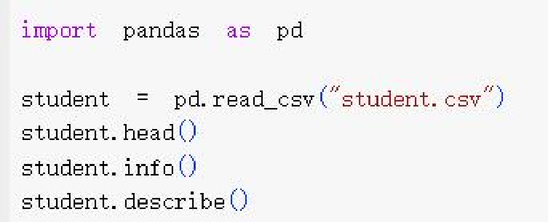


```
import pandas as pd

student = pd.read_csv("student.csv")
student.head()
```

3.



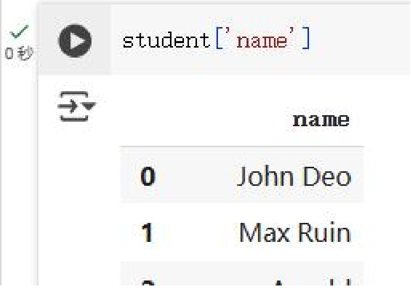
```
import pandas as pd

student = pd.read_csv("student.csv")
student.head()
student.info()
```

4. 

```
import pandas as pd

student = pd.read_csv("student.csv")
student.head()
student.info()
student.describe()
```

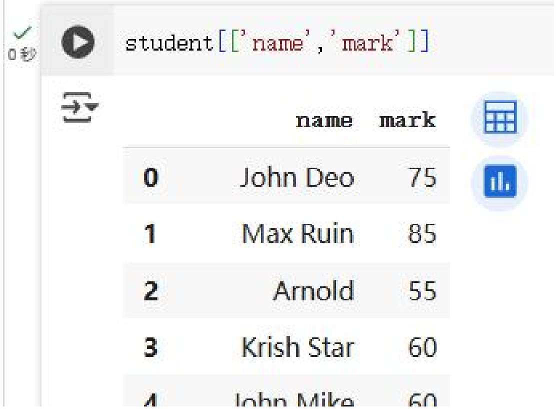
Exercise 2: Indexing and Slicing

1. Question: "Write the code to select the 'name' column."
2. Question: "Write the code to select the 'name' and 'mark' columns."
3. Question: "Write the code to select the first 3 rows."
4. Question: "Write the code to select all rows where the 'class' is 'Four'."

1. 

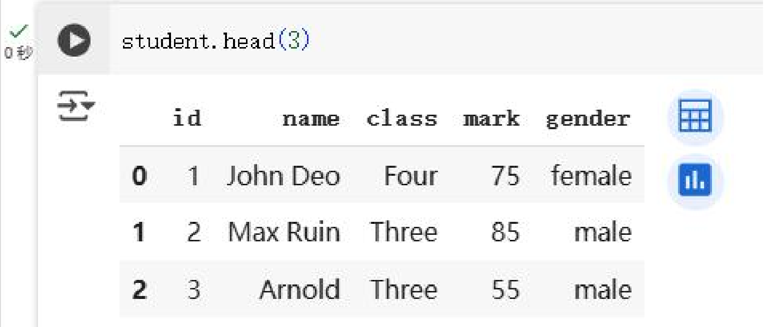
```
student['name']
```

```
      name
0  John Deo
1  Max Ruin
```

2. 

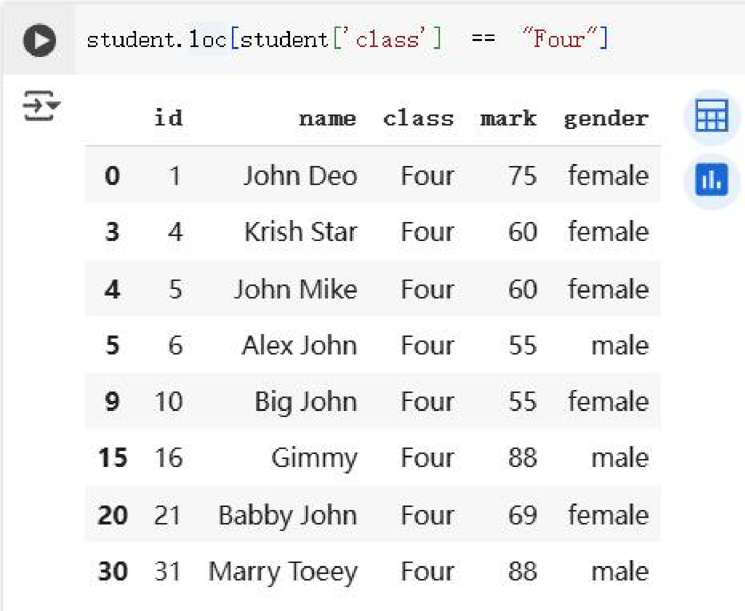
```
student[['name','mark']]
```

```
         name  mark
0    John Deo    75
1    Max Ruin    85
2      Arnold    55
3  Krish Star    60
4   John Mike    60
```

3. 

```
student.head(3)
```

```
   id      name  class  mark  gender
0   1  John Deo   Four    75  female
1   2  Max Ruin  Three    85    male
2   3    Arnold  Three    55    male
```



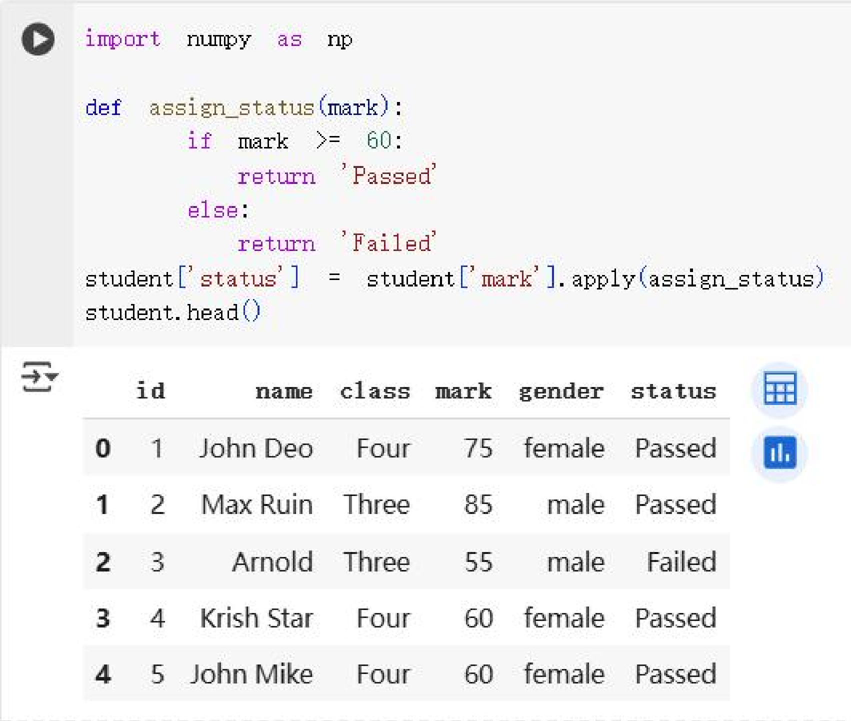
```python
student.loc[student['class'] == "Four"]
```

```
    id        name  class  mark  gender
 0   1    John Deo   Four    75  female
 3   4  Krish Star   Four    60  female
 4   5   John Mike   Four    60  female
 5   6   Alex John   Four    55    male
 9  10    Big John   Four    55  female
15  16       Gimmy   Four    88    male
20  21  Babby John   Four    69  female
30  31 Marry Toeey   Four    88    male
```

4.

Exercise 3: Data Manipulation

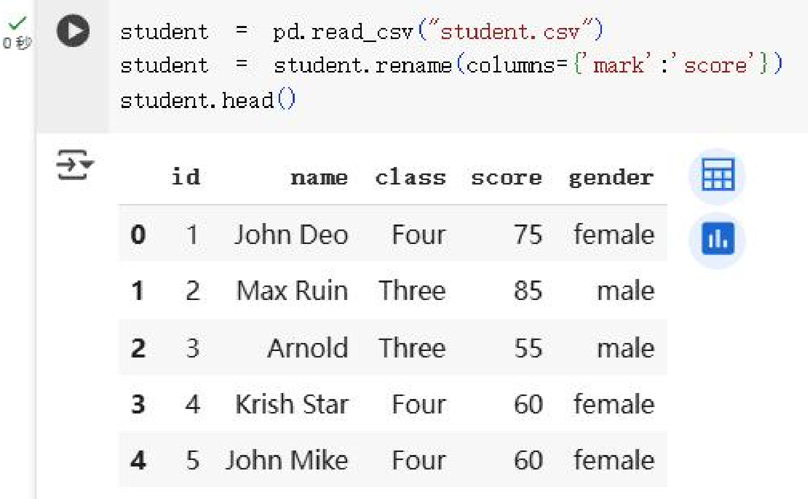
1. Question: "Write the code to add a new column 'passed' that indicates whether the student passed (mark >= 60)."
2. Question: "Write the code to rename the 'mark' column to 'score'."
3. Question: "Write the code to drop the 'passed' column."



```
import numpy as np

def assign_status(mark):
        if mark >= 60:
            return 'Passed'
        else:
            return 'Failed'
student['status'] = student['mark'].apply(assign_status)
student.head()
```

	id	name	class	mark	gender	status
0	1	John Deo	Four	75	female	Passed
1	2	Max Ruin	Three	85	male	Passed
2	3	Arnold	Three	55	male	Failed
3	4	Krish Star	Four	60	female	Passed
4	5	John Mike	Four	60	female	Passed

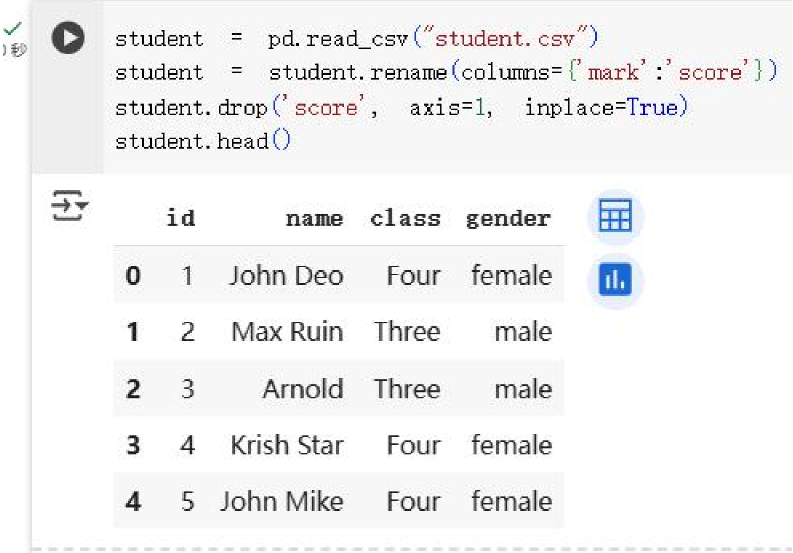
1.



```
student = pd.read_csv("student.csv")
student = student.rename(columns={'mark':'score'})
student.head()
```

	id	name	class	score	gender
0	1	John Deo	Four	75	female
1	2	Max Ruin	Three	85	male
2	3	Arnold	Three	55	male
3	4	Krish Star	Four	60	female
4	5	John Mike	Four	60	female

2.



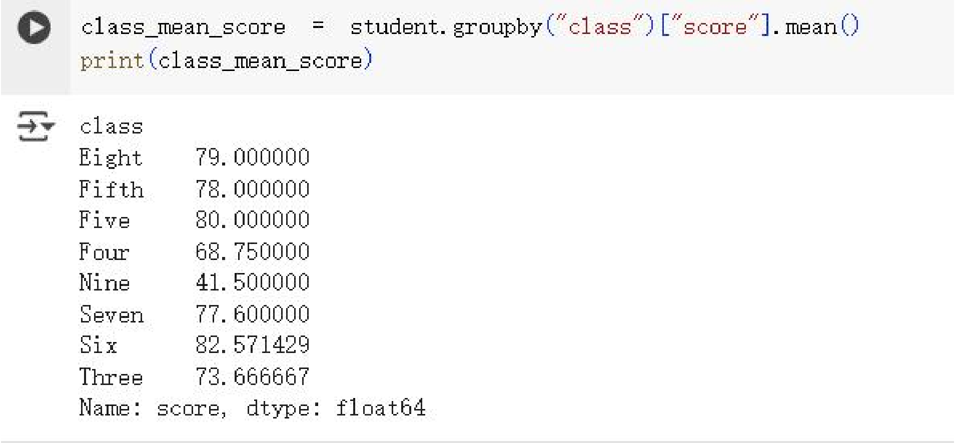
```
student = pd.read_csv("student.csv")
student = student.rename(columns={'mark':'score'})
student.drop('score', axis=1, inplace=True)
student.head()
```

	id	name	class	gender
0	1	John Deo	Four	female
1	2	Max Ruin	Three	male
2	3	Arnold	Three	male
3	4	Krish Star	Four	female
4	5	John Mike	Four	female

3.

Exercise 4: Aggregation and Grouping

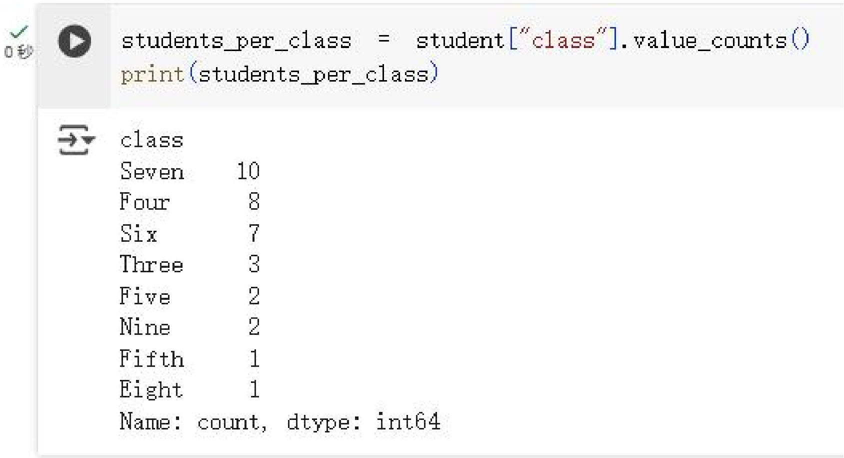
1. Question: "Write the code to group the DataFrame by the 'class' column and calculate the mean 'mark' for each group."
2. Question: "Write the code to count the number of students in each class."
3. Question: "Write the code to calculate the average mark for each gender."



```
class_mean_score = student.groupby("class")["score"].mean()
print(class_mean_score)
```

```
class
Eight     79.000000
Fifth     78.000000
Five      80.000000
Four      68.750000
Nine      41.500000
Seven     77.600000
Six       82.571429
Three     73.666667
Name: score, dtype: float64
```

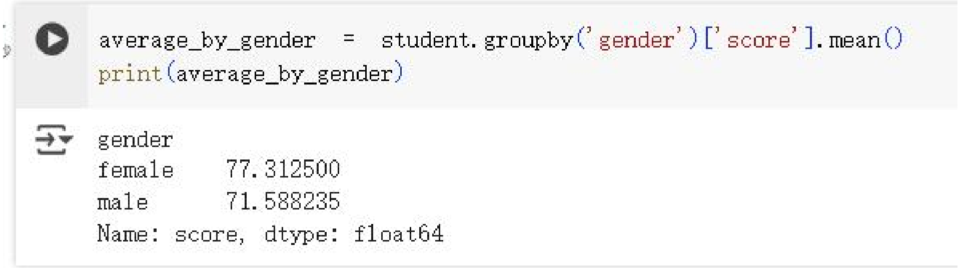
1.



```
students_per_class = student["class"].value_counts()
print(students_per_class)
```

```
class
Seven     10
Four       8
Six        7
Three      3
Five       2
Nine       2
Fifth      1
Eight      1
Name: count, dtype: int64
```

2.



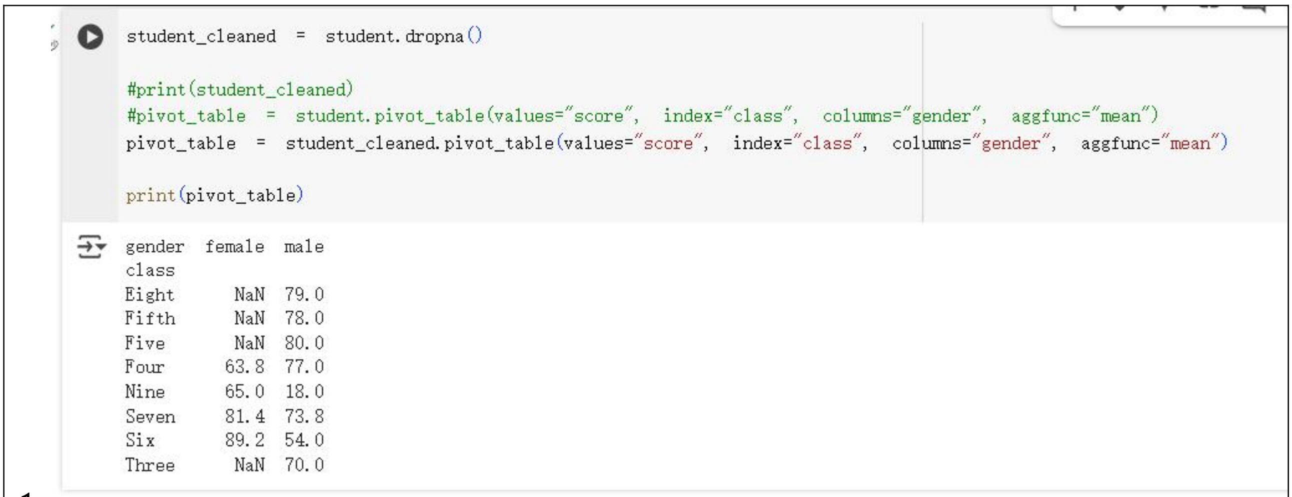
```
average_by_gender = student.groupby('gender')['score'].mean()
print(average_by_gender)
```

```
gender
female    77.312500
male      71.588235
Name: score, dtype: float64
```

3.

Exercise 5: Advanced Operations

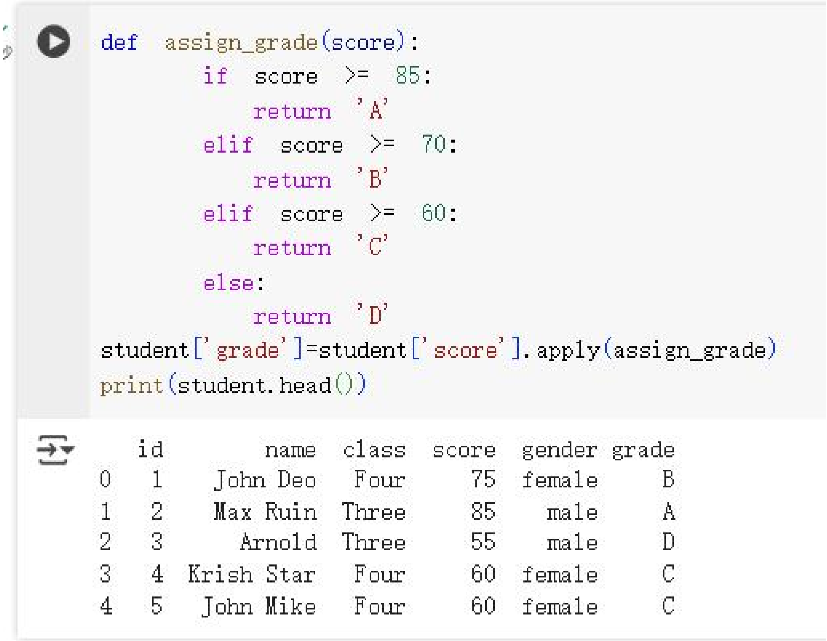
1. Question: "Write the code to create a pivot table with 'class' as rows, 'gender' as columns, and 'mark' as values."
2. Question: "Write the code to create a new column 'grade' where marks >= 85 are 'A', 70-84 are 'B', 60-69 are 'C', and below 60 are 'D'."
3. Question: "Write the code to sort the DataFrame by 'mark' in descending order."



```
student_cleaned = student.dropna()

#print(student_cleaned)
#pivot_table = student.pivot_table(values="score", index="class", columns="gender", aggfunc="mean")
pivot_table = student_cleaned.pivot_table(values="score", index="class", columns="gender", aggfunc="mean")

print(pivot_table)
```

```
gender  female  male
class
Eight      NaN  79.0
Fifth      NaN  78.0
Five       NaN  80.0
Four      63.8  77.0
Nine      65.0  18.0
Seven     81.4  73.8
Six       89.2  54.0
Three      NaN  70.0
```

1.

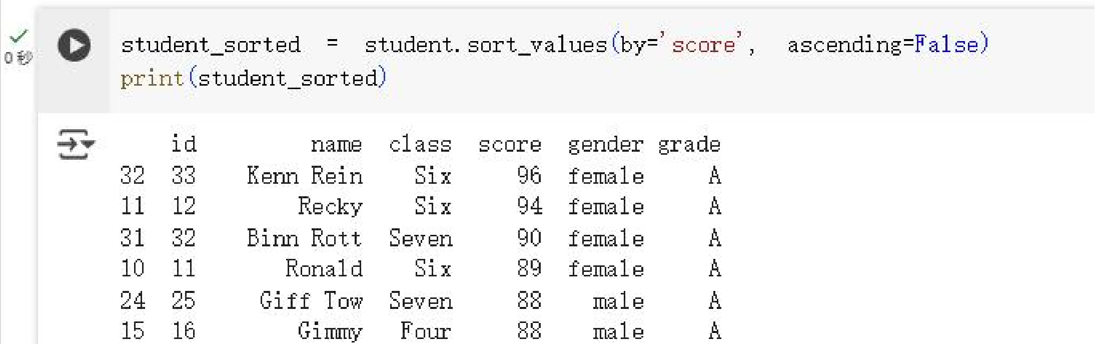


```
def assign_grade(score):
        if score >= 85:
            return 'A'
        elif score >= 70:
            return 'B'
        elif score >= 60:
            return 'C'
        else:
            return 'D'
student['grade']=student['score'].apply(assign_grade)
print(student.head())
```

```
   id        name  class  score  gender grade
0   1    John Deo   Four     75  female     B
1   2    Max Ruin  Three     85    male     A
2   3      Arnold  Three     55    male     D
3   4  Krish Star   Four     60  female     C
4   5   John Mike   Four     60  female     C
```

2.

3.



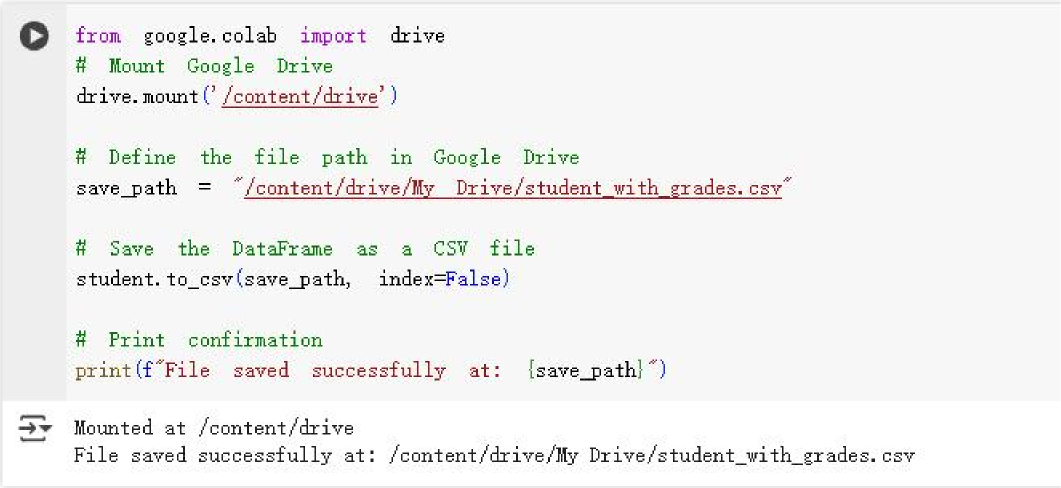
```
student_sorted  =  student.sort_values(by='score',  ascending=False)
print(student_sorted)
```

```
    id        name  class  score  gender grade
32  33   Kenn Rein    Six     96  female     A
11  12       Recky    Six     94  female     A
31  32   Binn Rott  Seven     90  female     A
10  11      Ronald    Six     89  female     A
24  25    Giff Tow  Seven     88    male     A
15  16       Gimmy   Four     88    male     A
```

Exercise 6: Exporting Data

1. Question: "Write the code to save the DataFrame with the new 'grade' column to a new CSV file."

1.



```
from  google.colab  import  drive
#  Mount  Google  Drive
drive.mount('/content/drive')

#  Define  the  file  path  in  Google  Drive
save_path  =  "/content/drive/My  Drive/student_with_grades.csv"

#  Save  the  DataFrame  as  a  CSV  file
student.to_csv(save_path,  index=False)

#  Print  confirmation
print(f"File  saved  successfully  at:  {save_path}")
```

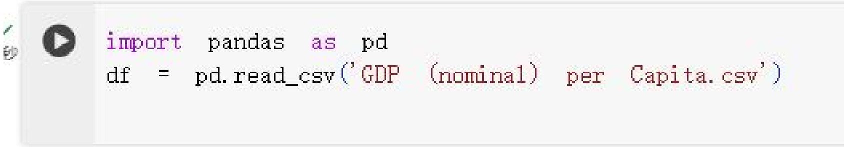
```
Mounted at /content/drive
File saved successfully at: /content/drive/My Drive/student_with_grades.csv
```

Exercise 7: If finished early try visualising the results

Day 4: Task 1

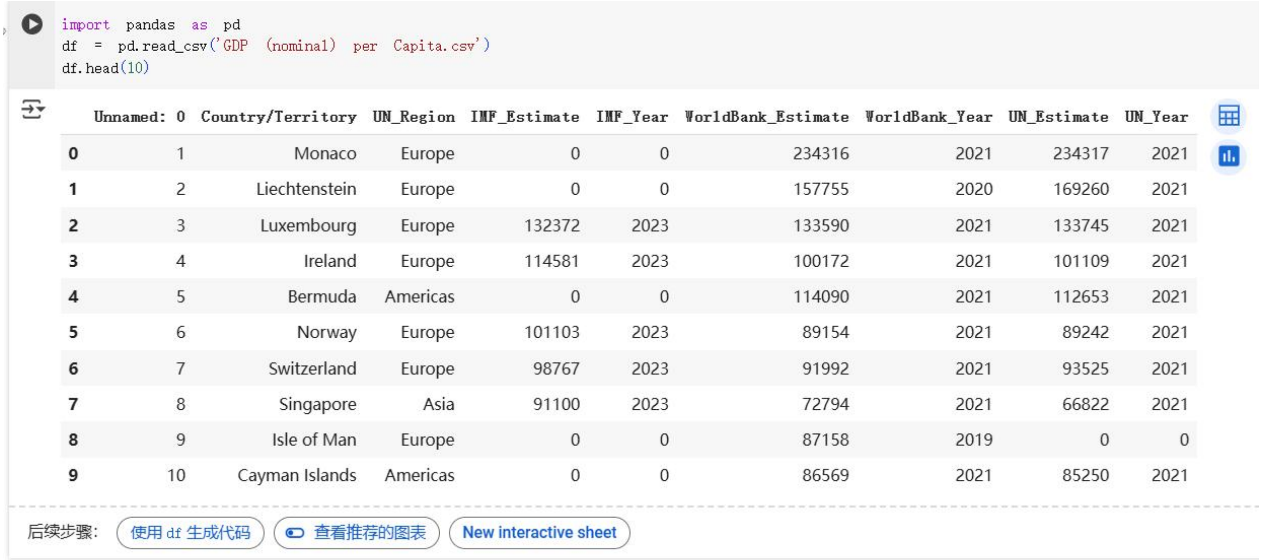
Using the ‘GDP (nominal) per Capita.csv’ which can be downloaded from the shared Folder, complete the below exercises and paste your input and output. Work individually, but we will work and support each other in the room.

- Read and save the ‘GDP (nominal) per Capita’ data to a data frame called “df” in Jyputer notebook
- Print the first 10 rows
- Print the last 5 rows
- Print ‘Country/Territory’ and ‘UN_Region’ columns



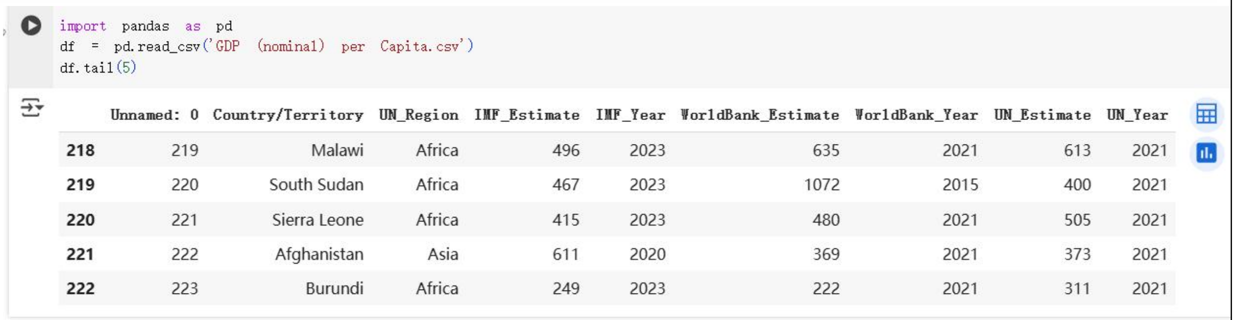
```python
import pandas as pd
df = pd.read_csv('GDP (nominal) per Capita.csv')
```

1.
2.



```python
import pandas as pd
df = pd.read_csv('GDP (nominal) per Capita.csv')
df.head(10)
```

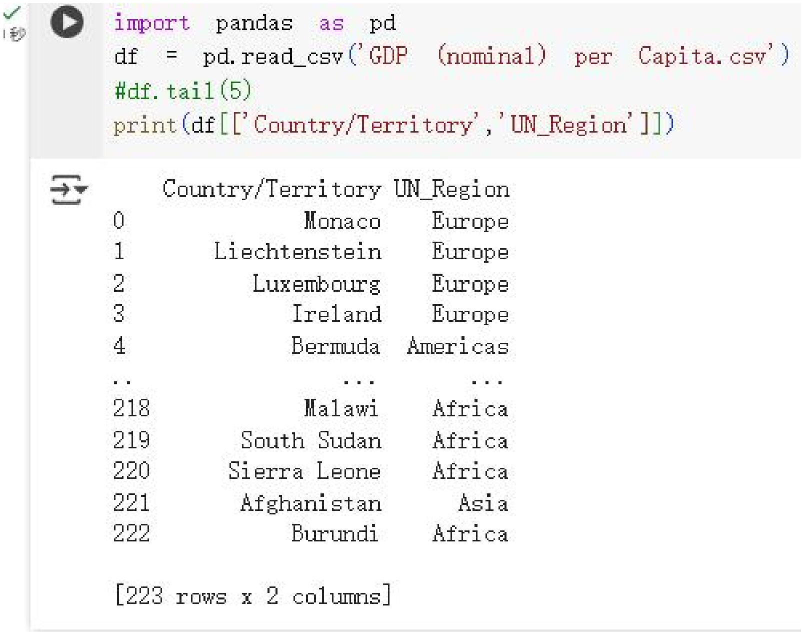
	Unnamed: 0	Country/Territory	UN_Region	IMF_Estimate	IMF_Year	WorldBank_Estimate	WorldBank_Year	UN_Estimate	UN_Year
0	1	Monaco	Europe	0	0	234316	2021	234317	2021
1	2	Liechtenstein	Europe	0	0	157755	2020	169260	2021
2	3	Luxembourg	Europe	132372	2023	133590	2021	133745	2021
3	4	Ireland	Europe	114581	2023	100172	2021	101109	2021
4	5	Bermuda	Americas	0	0	114090	2021	112653	2021
5	6	Norway	Europe	101103	2023	89154	2021	89242	2021
6	7	Switzerland	Europe	98767	2023	91992	2021	93525	2021
7	8	Singapore	Asia	91100	2023	72794	2021	66822	2021
8	9	Isle of Man	Europe	0	0	87158	2019	0	0
9	10	Cayman Islands	Americas	0	0	86569	2021	85250	2021



```python
import pandas as pd
df = pd.read_csv('GDP (nominal) per Capita.csv')
df.tail(5)
```

	Unnamed: 0	Country/Territory	UN_Region	IMF_Estimate	IMF_Year	WorldBank_Estimate	WorldBank_Year	UN_Estimate	UN_Year
218	219	Malawi	Africa	496	2023	635	2021	613	2021
219	220	South Sudan	Africa	467	2023	1072	2015	400	2021
220	221	Sierra Leone	Africa	415	2023	480	2021	505	2021
221	222	Afghanistan	Asia	611	2020	369	2021	373	2021
222	223	Burundi	Africa	249	2023	222	2021	311	2021

3.



```python
import pandas as pd
df = pd.read_csv('GDP (nominal) per Capita.csv')
#df.tail(5)
print(df[['Country/Territory','UN_Region']])
```

```
    Country/Territory UN_Region
0              Monaco    Europe
1       Liechtenstein    Europe
2          Luxembourg    Europe
3             Ireland    Europe
4             Bermuda  Americas
..                ...       ...
218            Malawi    Africa
219       South Sudan    Africa
220      Sierra Leone    Africa
221       Afghanistan      Asia
222           Burundi    Africa

[223 rows x 2 columns]
```

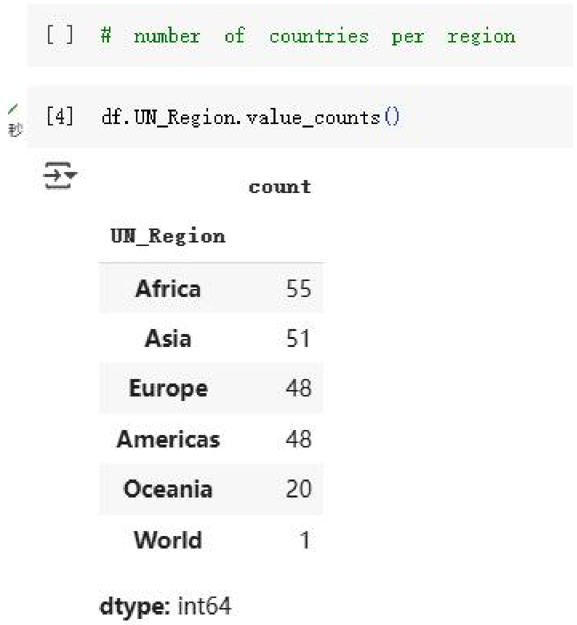
4.

Day 4: Task 2

Back with 'GDP (nominal) per Capita'. As a group, import and work your way through the Day_4_Python_Activity.ipynb notebook which can be found on the shared Folder. There are questions to answer, but also opportunities to have fun with the data – paste your input and output below.

Once complete, and again as a group, work with some more data and have some fun – there is no set agenda for this section, other than to embed the skills developed this week. Paste your input and output below and upon return we'll discuss progress made.

Additional data found here.



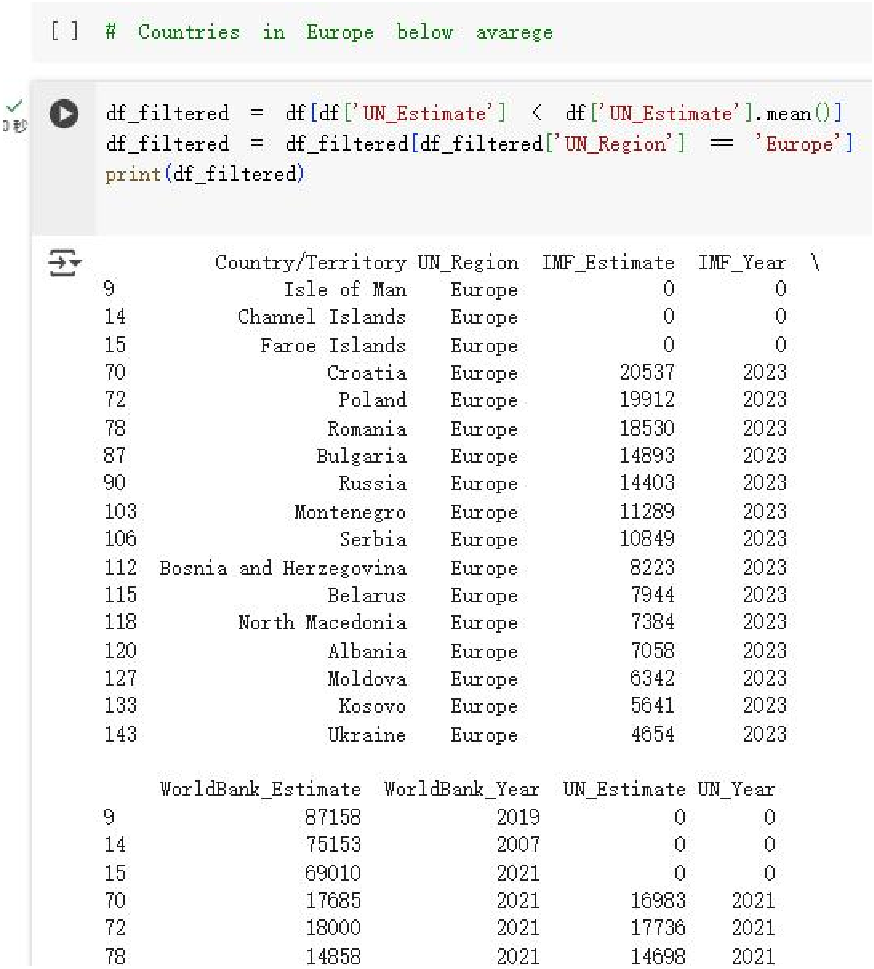
```
[ ]  # number of countries per region
```

```
[4]  df.UN_Region.value_counts()
```

UN_Region	count
Africa	55
Asia	51
Europe	48
Americas	48
Oceania	20
World	1

dtype: int64

1.

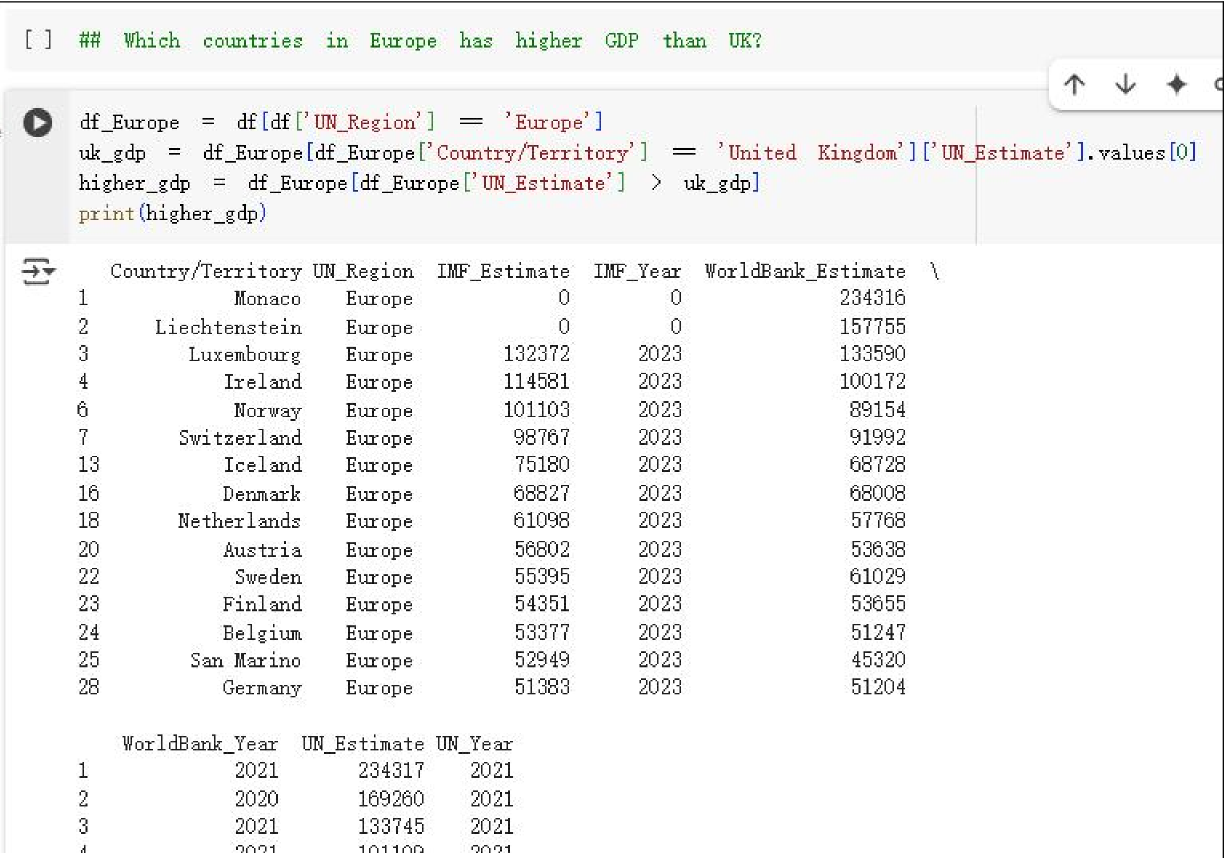


```
[ ]  # Countries in Europe below avarege
```

```
df_filtered = df[df['UN_Estimate'] < df['UN_Estimate'].mean()]
df_filtered = df_filtered[df_filtered['UN_Region'] == 'Europe']
print(df_filtered)
```

```
           Country/Territory UN_Region  IMF_Estimate  IMF_Year  \
9                Isle of Man    Europe             0         0
14           Channel Islands    Europe             0         0
15             Faroe Islands    Europe             0         0
70                   Croatia    Europe         20537      2023
72                    Poland    Europe         19912      2023
78                   Romania    Europe         18530      2023
87                  Bulgaria    Europe         14893      2023
90                    Russia    Europe         14403      2023
103               Montenegro    Europe         11289      2023
106                   Serbia    Europe         10849      2023
112   Bosnia and Herzegovina    Europe          8223      2023
115                  Belarus    Europe          7944      2023
118          North Macedonia    Europe          7384      2023
120                  Albania    Europe          7058      2023
127                  Moldova    Europe          6342      2023
133                   Kosovo    Europe          5641      2023
143                  Ukraine    Europe          4654      2023

     WorldBank_Estimate  WorldBank_Year  UN_Estimate  UN_Year
9                 87158            2019            0        0
14                75153            2007            0        0
15                69010            2021            0        0
70                17685            2021        16983     2021
72                18000            2021        17736     2021
78                14858            2021        14698     2021
```

2.

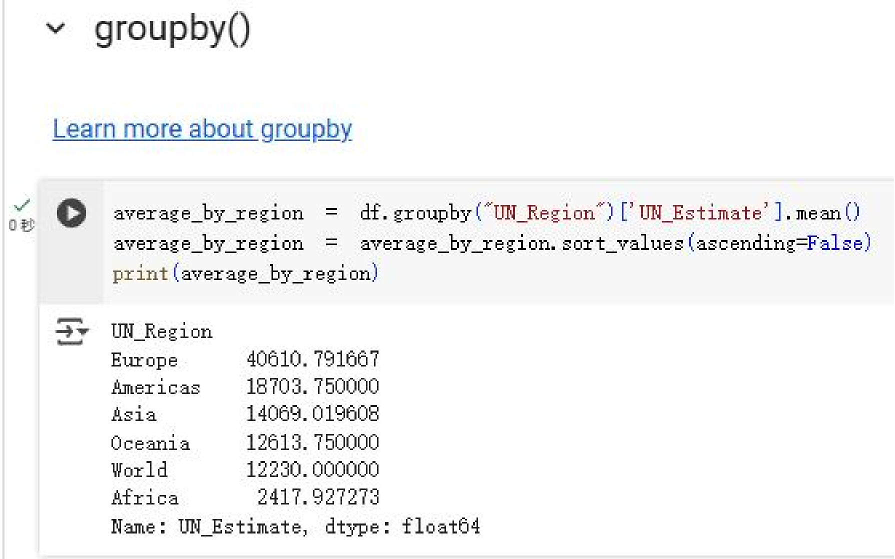


```
[ ] ## Which countries in Europe has higher GDP than UK?
```

```
df_Europe = df[df['UN_Region'] == 'Europe']
uk_gdp = df_Europe[df_Europe['Country/Territory'] == 'United Kingdom']['UN_Estimate'].values[0]
higher_gdp = df_Europe[df_Europe['UN_Estimate'] > uk_gdp]
print(higher_gdp)
```

```
   Country/Territory UN_Region  IMF_Estimate  IMF_Year  WorldBank_Estimate  \
1             Monaco    Europe             0         0              234316
2      Liechtenstein    Europe             0         0              157755
3         Luxembourg    Europe        132372      2023              133590
4            Ireland    Europe        114581      2023              100172
6             Norway    Europe        101103      2023               89154
7        Switzerland    Europe         98767      2023               91992
13           Iceland    Europe         75180      2023               68728
16           Denmark    Europe         68827      2023               68008
18       Netherlands    Europe         61098      2023               57768
20           Austria    Europe         56802      2023               53638
22            Sweden    Europe         55395      2023               61029
23           Finland    Europe         54351      2023               53655
24           Belgium    Europe         53377      2023               51247
25        San Marino    Europe         52949      2023               45320
28           Germany    Europe         51383      2023               51204

    WorldBank_Year  UN_Estimate  UN_Year
1             2021       234317     2021
2             2020       169260     2021
3             2021       133745     2021
```

3.



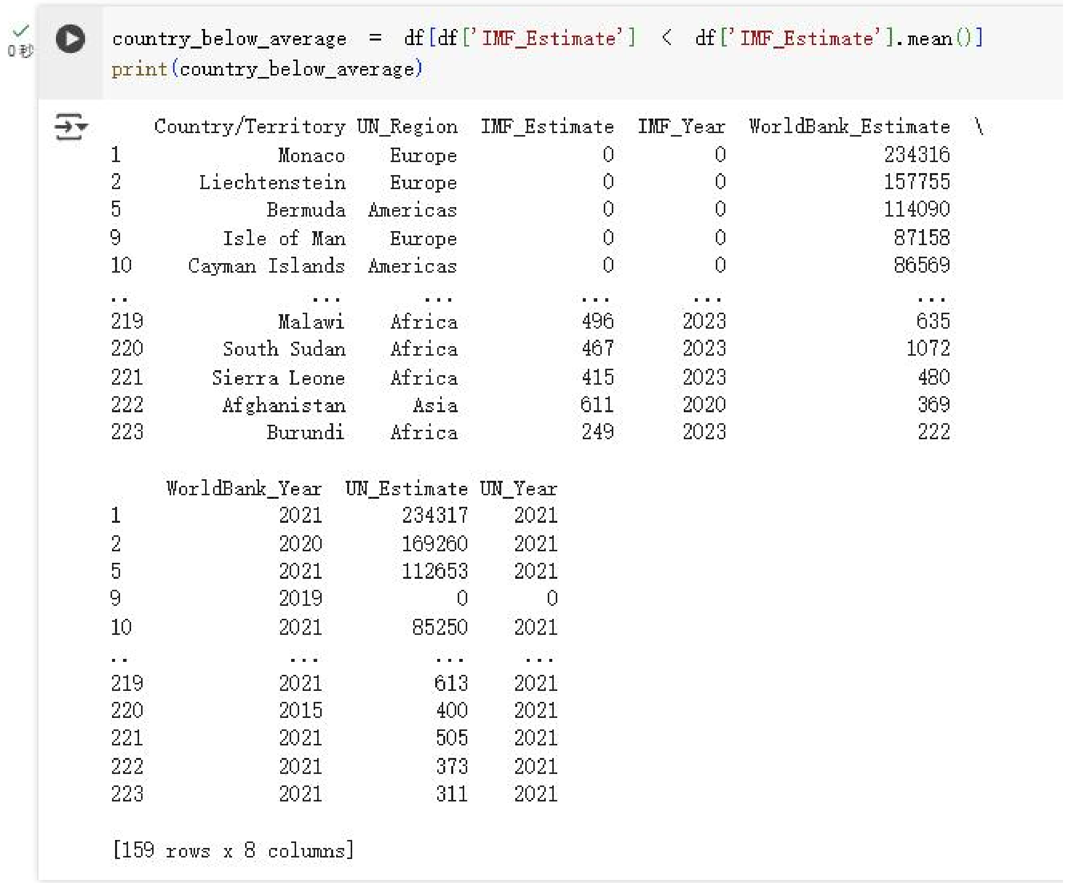
groupby()

Learn more about groupby

```
average_by_region = df.groupby("UN_Region")['UN_Estimate'].mean()
average_by_region = average_by_region.sort_values(ascending=False)
print(average_by_region)
```

```
UN_Region
Europe      40610.791667
Americas    18703.750000
Asia        14069.019608
Oceania     12613.750000
World       12230.000000
Africa       2417.927273
Name: UN_Estimate, dtype: float64
```

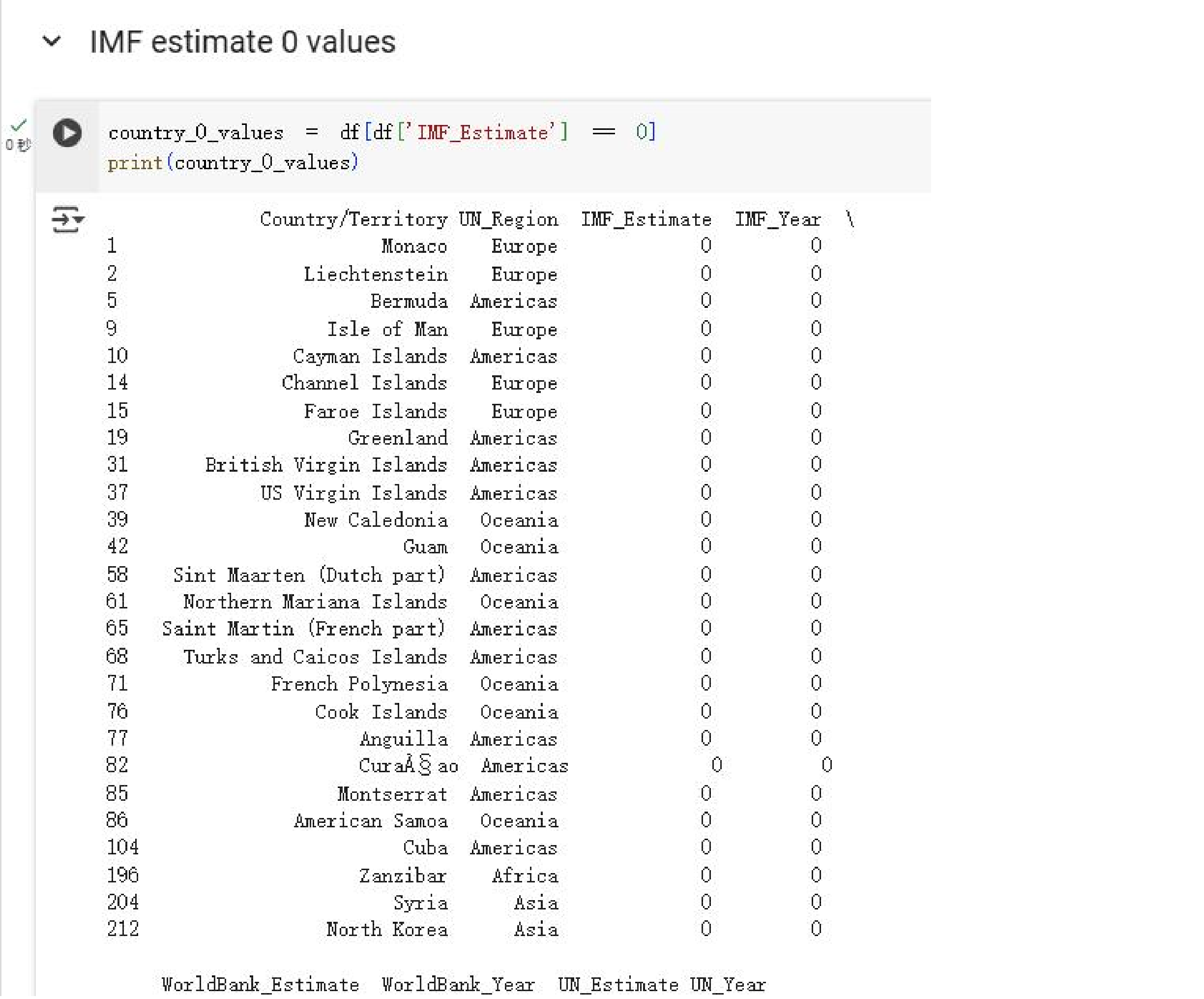
4.



```
country_below_average = df[df['IMF_Estimate'] < df['IMF_Estimate'].mean()]
print(country_below_average)
```

```
    Country/Territory UN_Region  IMF_Estimate  IMF_Year  WorldBank_Estimate  \
1              Monaco    Europe             0         0              234316
2       Liechtenstein    Europe             0         0              157755
5             Bermuda  Americas             0         0              114090
9         Isle of Man    Europe             0         0               87158
10     Cayman Islands  Americas             0         0               86569
..                ...       ...           ...       ...                 ...
219            Malawi    Africa           496      2023                 635
220       South Sudan    Africa           467      2023                1072
221      Sierra Leone    Africa           415      2023                 480
222       Afghanistan      Asia           611      2020                 369
223           Burundi    Africa           249      2023                 222

     WorldBank_Year  UN_Estimate  UN_Year
1              2021       234317     2021
2              2020       169260     2021
5              2021       112653     2021
9              2019            0        0
10             2021        85250     2021
..              ...          ...      ...
219            2021          613     2021
220            2015          400     2021
221            2021          505     2021
222            2021          373     2021
223            2021          311     2021

[159 rows x 8 columns]
```

5.

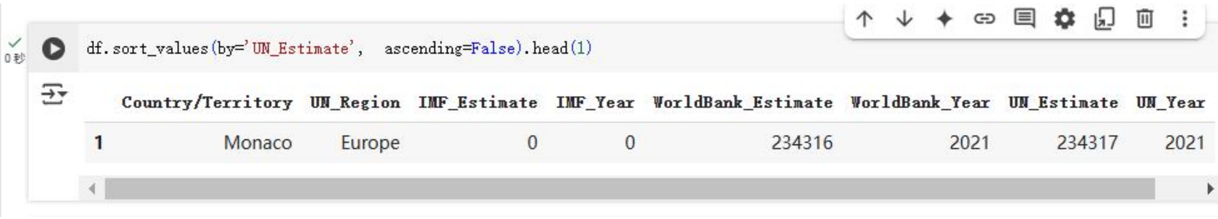


IMF estimate 0 values

```
country_0_values = df[df['IMF_Estimate'] == 0]
print(country_0_values)
```

```
              Country/Territory UN_Region  IMF_Estimate  IMF_Year  \
1                        Monaco    Europe             0         0
2                 Liechtenstein    Europe             0         0
5                       Bermuda  Americas             0         0
9                   Isle of Man    Europe             0         0
10               Cayman Islands  Americas             0         0
14              Channel Islands    Europe             0         0
15                Faroe Islands    Europe             0         0
19                    Greenland  Americas             0         0
31       British Virgin Islands  Americas             0         0
37            US Virgin Islands  Americas             0         0
39                New Caledonia   Oceania             0         0
42                         Guam   Oceania             0         0
58    Sint Maarten (Dutch part)  Americas             0         0
61     Northern Mariana Islands   Oceania             0         0
65   Saint Martin (French part)  Americas             0         0
68     Turks and Caicos Islands  Americas             0         0
71             French Polynesia   Oceania             0         0
76                 Cook Islands   Oceania             0         0
77                     Anguilla  Americas             0         0
82                     CuraÃ§ao  Americas             0         0
85                   Montserrat  Americas             0         0
86               American Samoa   Oceania             0         0
104                        Cuba  Americas             0         0
196                    Zanzibar    Africa             0         0
204                       Syria      Asia             0         0
212                 North Korea      Asia             0         0

     WorldBank_Estimate  WorldBank_Year  UN_Estimate UN_Year
```

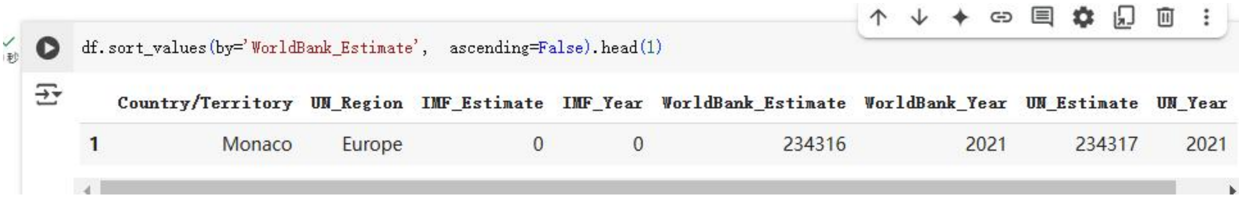
6.



```
df.sort_values(by='UN_Estimate', ascending=False).head(1)
```

	Country/Territory	UN_Region	IMF_Estimate	IMF_Year	WorldBank_Estimate	WorldBank_Year	UN_Estimate	UN_Year
1	Monaco	Europe	0	0	234316	2021	234317	2021

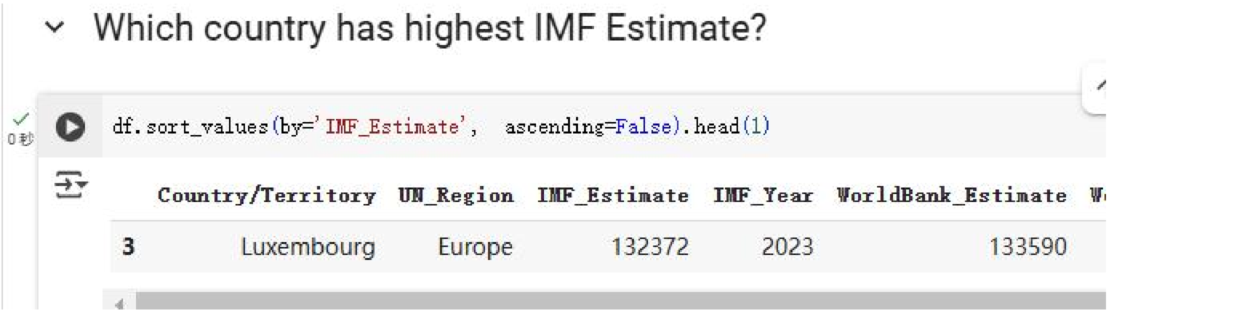
7.



```
df.sort_values(by='WorldBank_Estimate', ascending=False).head(1)
```

	Country/Territory	UN_Region	IMF_Estimate	IMF_Year	WorldBank_Estimate	WorldBank_Year	UN_Estimate	UN_Year
1	Monaco	Europe	0	0	234316	2021	234317	2021

8.

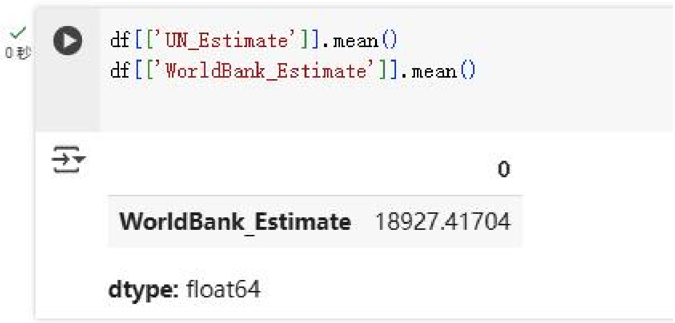


Which country has highest IMF Estimate?

```
df.sort_values(by='IMF_Estimate', ascending=False).head(1)
```

	Country/Territory	UN_Region	IMF_Estimate	IMF_Year	WorldBank_Estimate	W
3	Luxembourg	Europe	132372	2023	133590	

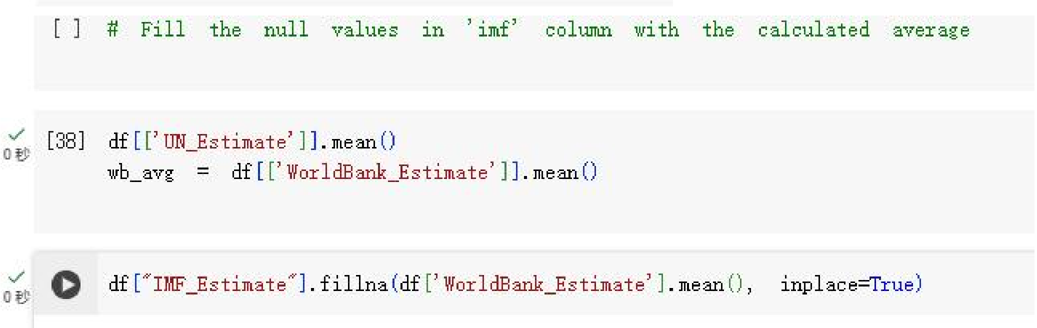
9.



```
df[['UN_Estimate']].mean()
df[['WorldBank_Estimate']].mean()
```

```
                          0
WorldBank_Estimate  18927.41704

dtype: float64
```

10.

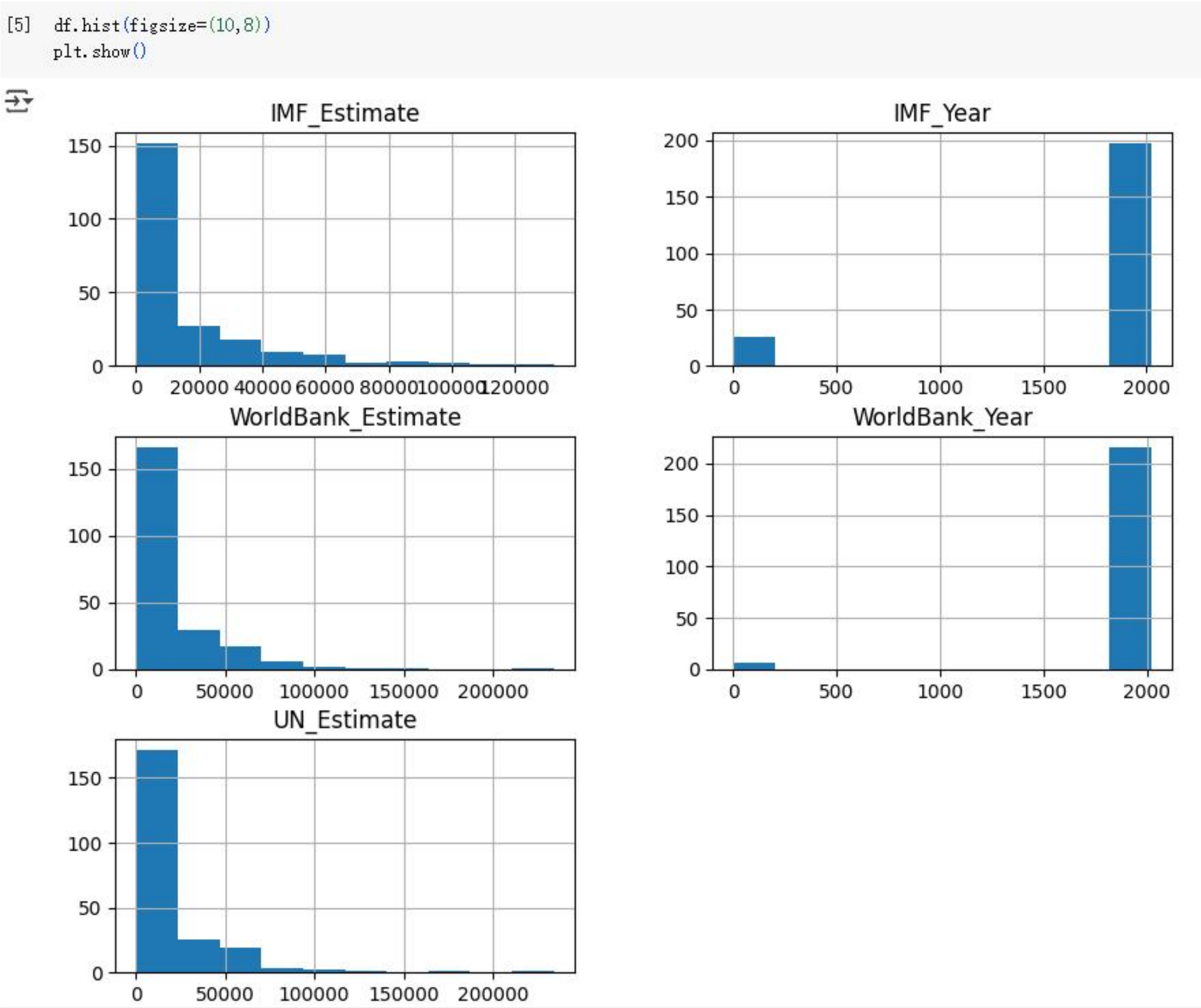


```
[ ]  # Fill the null values in 'imf' column with the calculated average
```

```
[38]  df[['UN_Estimate']].mean()
      wb_avg = df[['WorldBank_Estimate']].mean()
```

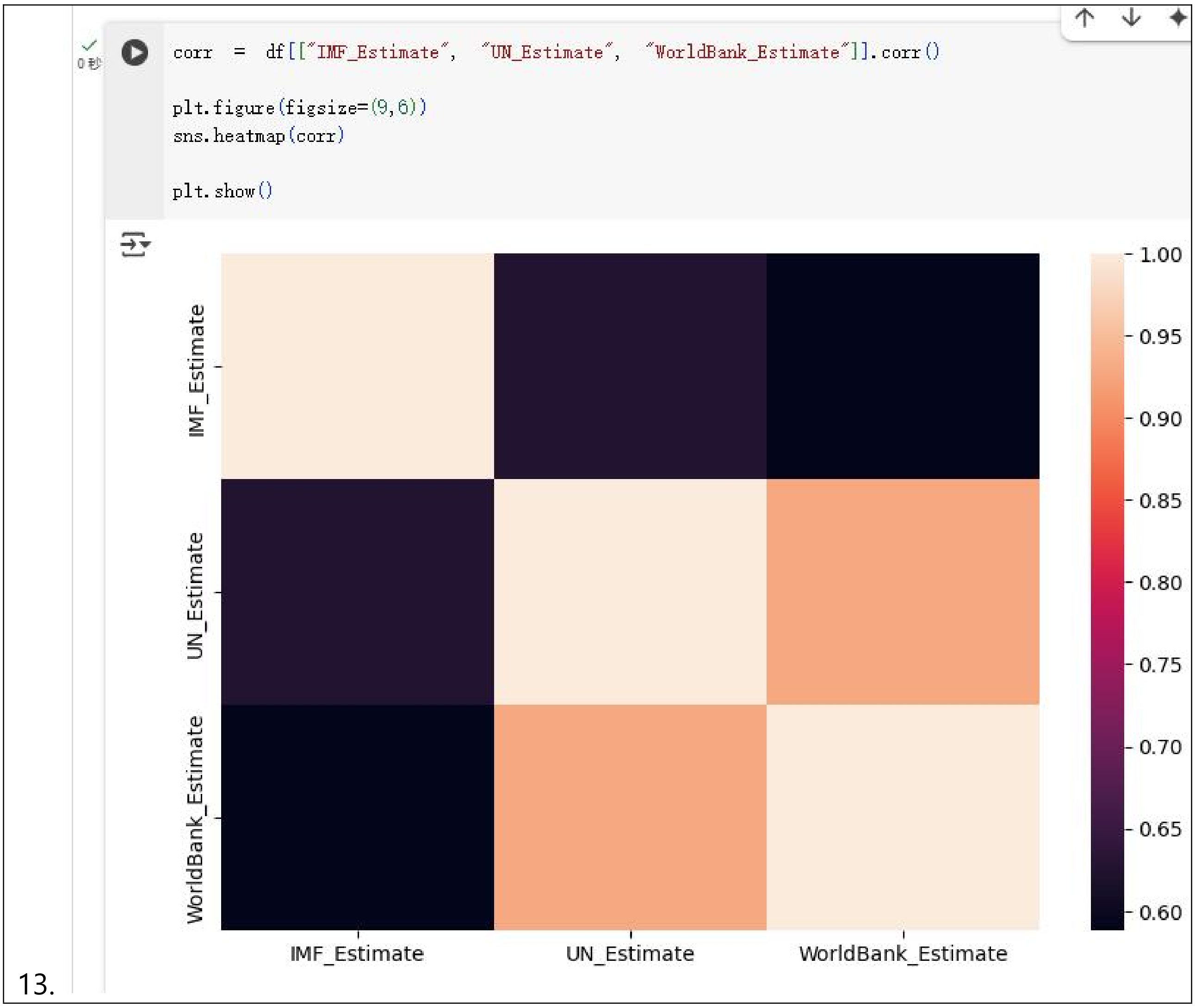
```
df["IMF_Estimate"].fillna(df['WorldBank_Estimate'].mean(), inplace=True)
```

11.



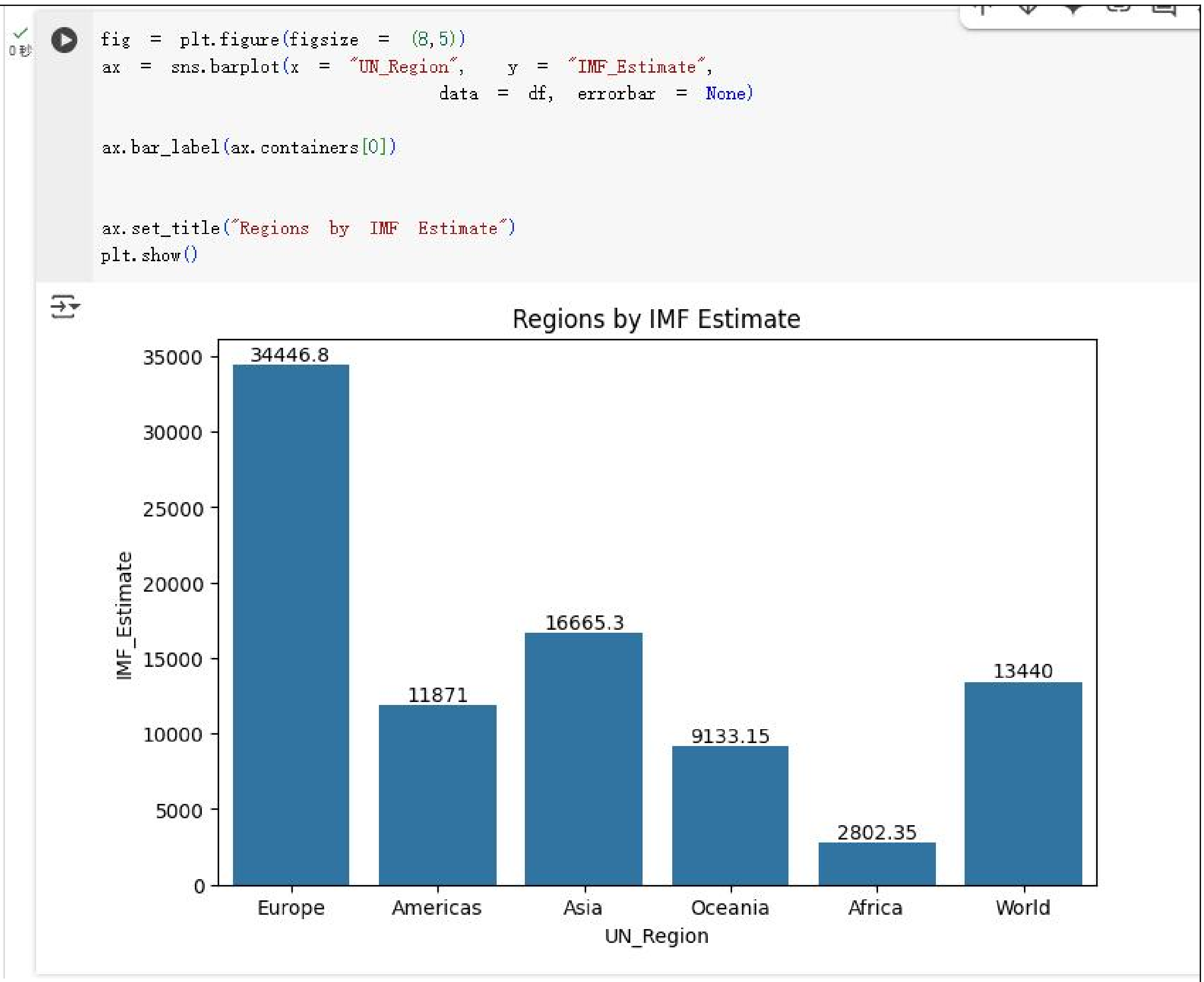
```
[5]  df.hist(figsize=(10,8))
     plt.show()
```

12.



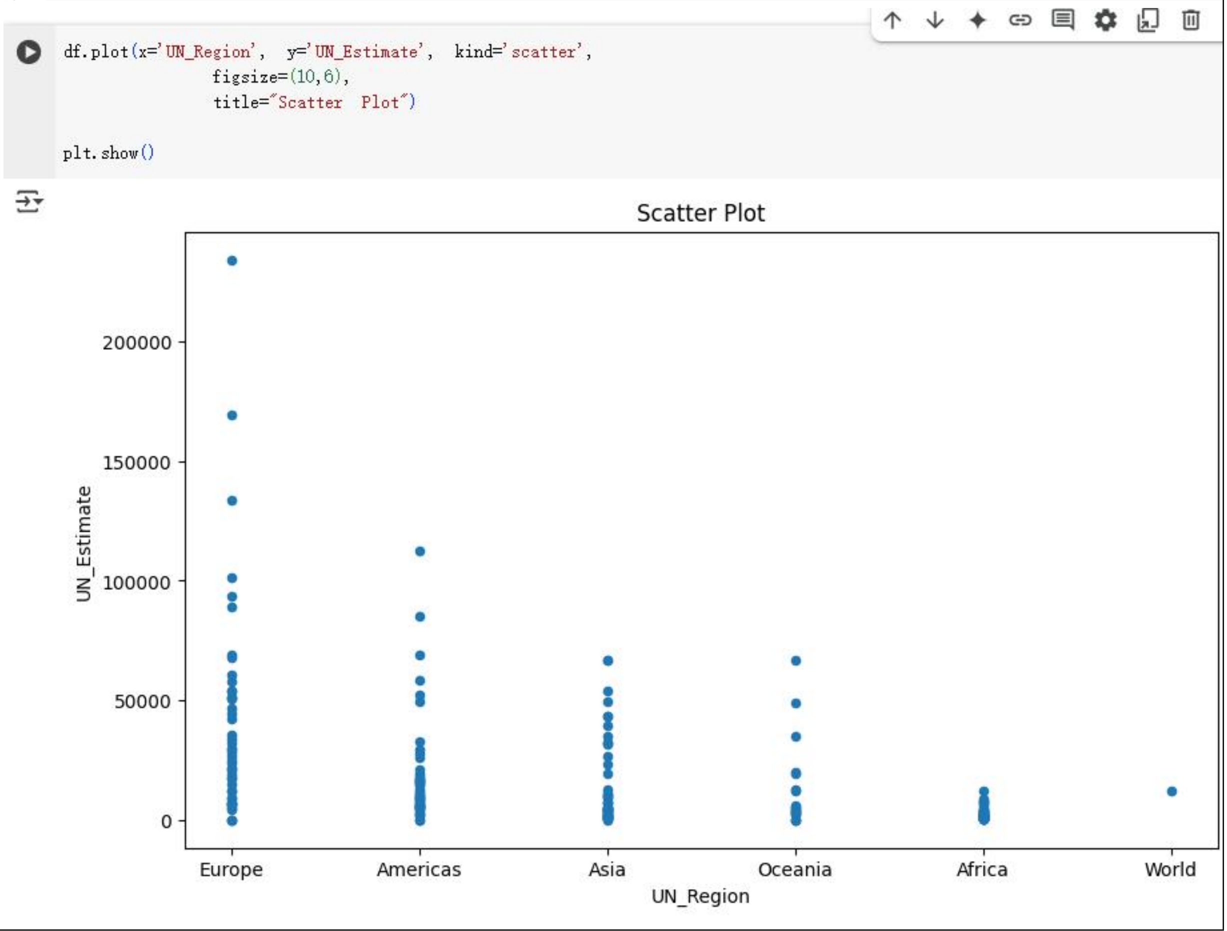
```
corr = df[["IMF_Estimate", "UN_Estimate", "WorldBank_Estimate"]].corr()

plt.figure(figsize=(9,6))
sns.heatmap(corr)

plt.show()
```

13.



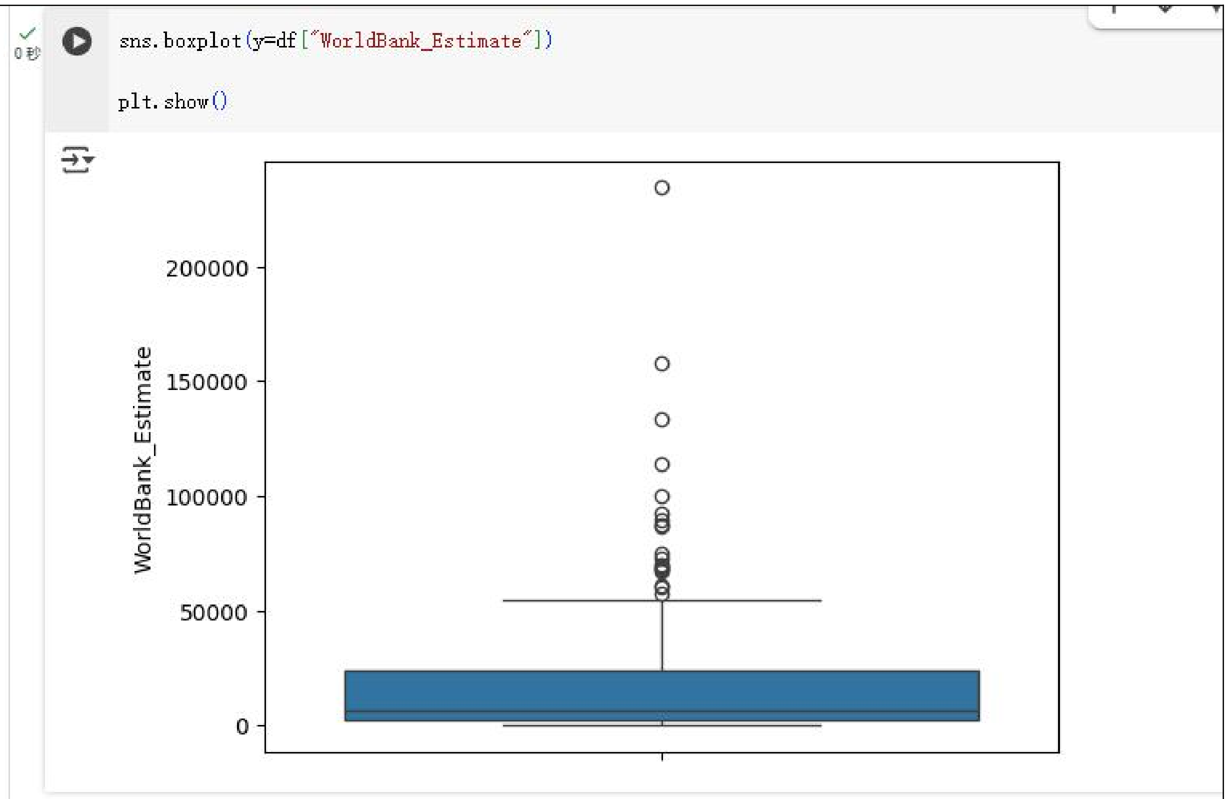
```
fig = plt.figure(figsize = (8,5))
ax = sns.barplot(x = "UN_Region",   y = "IMF_Estimate",
                            data = df,  errorbar = None)

ax.bar_label(ax.containers[0])


ax.set_title("Regions by IMF Estimate")
plt.show()
```

14.



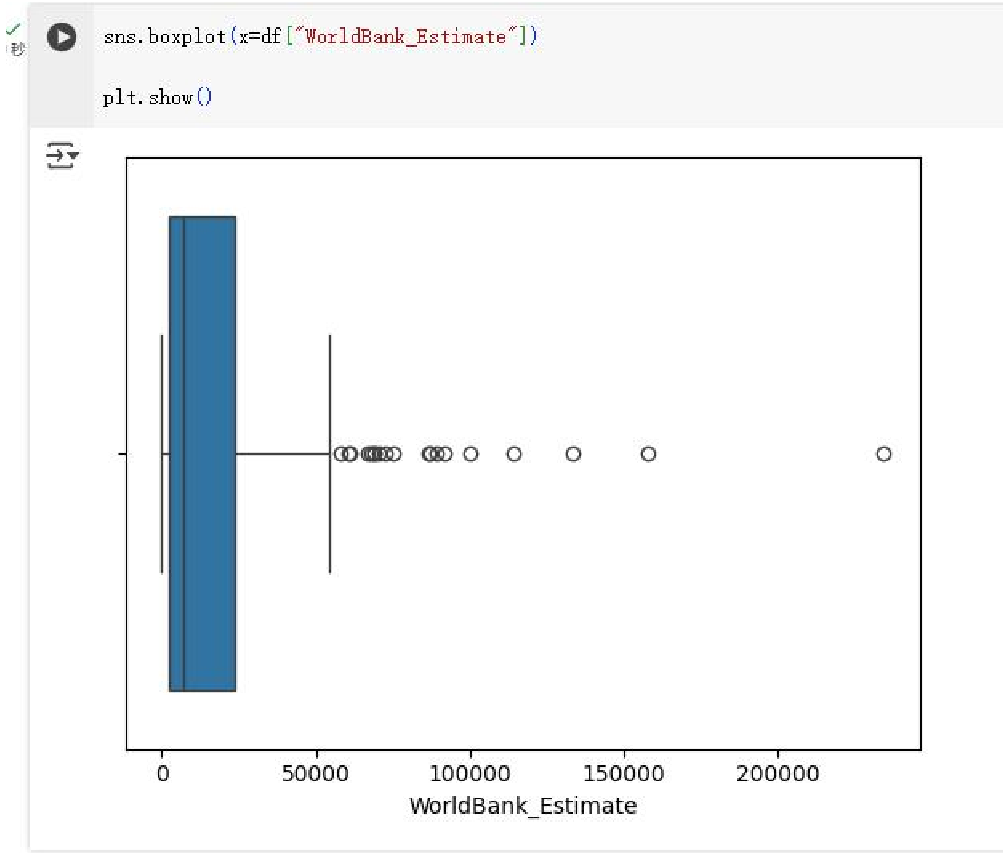
```
df.plot(x='UN_Region',  y='UN_Estimate',  kind='scatter',
            figsize=(10,6),
            title="Scatter Plot")

plt.show()
```

15.



```
sns.boxplot(y=df["WorldBank_Estimate"])

plt.show()
```

16.



```
sns.boxplot(x=df["WorldBank_Estimate"])

plt.show()
```

17.

Course Notes

It is recommended to take notes from the course, use the space below to do so, or use the revision guide shared with the class:

We have included a range of additional links to further resources and information that you may find useful, these can be found within your revision guide.

END OF WORKBOOK

Please check through your work thoroughly before submitting and update the table of contents if required.

Please send your completed work booklet to your trainer.