Just IT

B2W group

Apprenticeships | Training | Recruitment

Data Technician
Week 5
Introduction to Python & Pandas

Name: Al Amin

Course Date: From 03/03/2025 to 06/03/2025

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 implementation in Python
for num in range(1,101):
 if num%3==0 and num%5==0:
   print(num,'fizzbuzz')
 elif num%3==0:
   print(num,'fizz')
 elif num%5==0:
   print(num,'buzz')
 else:
   print(num)
# Print the number if it's not divisible by 3 or 5.
1
2
3 fizz
4
5 buzz
6 fizz
7
8
9 fizz
10 buzz
```

11

12 fizz

13

14

15 fizzbuzz

16

17

18 fizz

19

20 buzz

21 fizz

22

23

24 fizz

25 buzz

26

27 fizz

28

29

30 fizzbuzz

31

32

33 fizz

34

35 buzz

36 fizz

37

38

39 fizz

40 buzz

41

42 fizz

43

44

45 fizzbuzz

46

47

48 fizz

49

50 buzz

51 fizz

52

53

54 fizz

55 buzz

56

57 fizz

58

59

60 fizzbuzz

61

62

63 fizz

64

65 buzz

66 fizz

67

68

69 fizz

70 buzz

71

72 fizz

73

74

75 fizzbuzz

76

77

78 fizz

79

80 buzz

81 fizz

82

83

84 fizz

85 buzz

86

87 fizz

88

89

90 fizzbuzz

91

92

93 fizz

94

95 buzz

96 fizz

97

98

99 fizz

100 buzz

Day 3: Task 1

Using the 'student.csv' which can be downloaded here, 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.
```
#code to read a CSV file into a Pandas DataFrame
import pandas as pd
dataframe = pd.read_csv('student.csv')
dataframe
```

	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
3	4	Krish Star	Four	60	female
4	5	John Mike	Four	60	female
5	6	Alex John	Four	55	male
6	7	My John Rob	Fifth	78	male
7	8	Asruid	Five	85	male
8	9	Tes Qry	Six	78	NaN
9	10	Big John	Four	55	female
10	11	Ronald	Six	89	female
11	12	Recky	Six	94	female
12	13	Kty	Seven	88	female
13	14	Bigy	Seven	88	female
14	15	Tade Row	NaN	88	male
15	16	Gimmy	Four	88	male
16	17	Tumyu	Six	54	male
17	18	Honny	Five	75	male

18	19	Tinny	Nine	18	male
19	20	Jackly	Nine	65	female
20	21	Babby John	Four	69	female
21	22	Reggid	Seven	55	female
22	23	Herod	Eight	79	male
23	24	Tiddy Now	Seven	78	male
24	25	Giff Tow	Seven	88	male
25	26	Crelea	Seven	79	male
26	27	NaN	Three	81	NaN
27	28	Rojj Base	Seven	86	female
28	29	Tess Played	Seven	55	male
29	30	Reppy Red	Six	79	female
30	31	Marry Toeey	Four	88	male
31	32	Binn Rott	Seven	90	female
32	33	Kenn Rein	Six	96	female
33	34	Gain Toe	Seven	69	male
34	35	Rows Noump	Six	88	female

2. #code to display the first 5 rows of the DataFrame.
   dataframe.head()

	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
3	4	Krish Star	Four	60	female
4	5	John Mike	Four	60	female

3. #code to get the information about the DataFrame.

```
dataframe.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35 entries, 0 to 34
Data columns (total 5 columns):
 #   Column Non-Null Count Dtype
---  ------ -------------- -----
 0   id     35 non-null    int64
 1   name   34 non-null    object
 2   class  34 non-null    object
 3   mark   35 non-null    int64
 4   gender 33 non-null    object
dtypes: int64(2), object(3)
memory usage: 1.5+ KB
```

4.#code to get summary statistics for the DataFrame.

```
dataframe.describe()
```

	id	mark
count	35.000000	35.000000
mean	18.000000	74.657143
std	10.246951	16.401117
min	1.000000	18.000000
25%	9.500000	62.500000
50%	18.000000	79.000000
75%	26.500000	88.000000
max	35.000000	96.000000

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.  #code to select the 'name' column
dataframe['name']

2. #code to select the 'name' and 'mark' columns
dataframe[['name','mark']].head(5)
```

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

```
3. #code to display the first 3 rows
dataframe.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

```
4. #code to select all rows where the 'class' is 'Four'
dataframe[dataframe['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

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."

1. #code to add a new column 'passed' that indicates whether the student passed (mark >= 60).

```
dataframe['passed'] = dataframe['mark'] >= 60
dataframe
```

	id	name	class	mark	gender	passed
0	1	John Deo	Four	75	female	True
1	2	Max Ruin	Three	85	male	True
2	3	Arnold	Three	55	male	False
3	4	Krish Star	Four	60	female	True
4	5	John Mike	Four	60	female	True
5	6	Alex John	Four	55	male	False
6	7	My John Rob	Fifth	78	male	True
7	8	Asruid	Five	85	male	True
8	9	Tes Qry	Six	78	NaN	True
9	10	Big John	Four	55	female	False
10	11	Ronald	Six	89	female	True

2. #code to rename the 'mark' column to 'score'

```
dataframe.rename(columns={'mark':'score'}, inplace=True)
dataframe
```

	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. #code to drop the 'passed' column

```
dataframe.drop('passed', axis=1, inplace=True)
dataframe
```

	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

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."

1. #code to group the DataFrame by the 'class' column and calculate the mean 'mark' for each group

```
dataframe.groupby('class')['mark'].mean()
```

class	mark
Eight	79.000000
Fifth	78.000000
Five	80.000000
Four	68.750000
Nine	41.500000
Seven	77.600000
Six	82.571429
Three	73.666667

2. #code to count the number of students in each class

```
dataframe['class'].value_counts()
```

class	count
Seven	10
Four	8
Six	7
Three	3
Five	2
Nine	2
Fifth	1
Eight	1

3. #code to calculate the average mark for each gender

```
dataframe.groupby('gender')['mark'].mean()
```

gender	mark
female	77.312500
male	71.588235

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."

```
1. #code to create a pivot table with 'class' as rows, 'gender' as columns, and 'mark' as values
dataframe.pivot_table(index='class', columns='gender',values='mark')
2. #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'
dataframe['grade'] = pd.cut(dataframe['mark'], bins=[0,59,69,84,100], labels=['D','C','B','A'])
dataframe
```

	id	name	class	mark	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
5	6	Alex John	Four	55	male	D
6	7	My John Rob	Fifth	78	male	B
7	8	Asruid	Five	85	male	A
8	9	Tes Qry	Six	78	NaN	B
9	10	Big John	Four	55	female	D
10	11	Ronald	Six	89	female	A

```
3. #code to sort the DataFrame by 'mark' in descending order
dataframe.sort_values(by='mark', ascending=False)
```

	id	name	class	mark	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

14	15	Tade Row	NaN	88	male	A
13	14	Bigy	Seven	88	female	A
12	13	Kty	Seven	88	female	A
34	35	Rows Noump	Six	88	female	A
30	31	Marry Toeey	Four	88	male	A
27	28	Rojj Base	Seven	86	female	A
7	8	Asruid	Five	85	male	A
1	2	Max Ruin	Three	85	male	A
26	27	NaN	Three	81	NaN	B
22	23	Herod	Eight	79	male	B
29	30	Reppy Red	Six	79	female	B
25	26	Crelea	Seven	79	male	B
8	9	Tes Qry	Six	78	NaN	B
6	7	My John Rob	Fifth	78	male	B
23	24	Tiddy Now	Seven	78	male	B
0	1	John Deo	Four	75	female	B
17	18	Honny	Five	75	male	B
20	21	Babby John	Four	69	female	C
33	34	Gain Toe	Seven	69	male	C
19	20	Jackly	Nine	65	female	C
4	5	John Mike	Four	60	female	C
3	4	Krish Star	Four	60	female	C
21	22	Reggid	Seven	55	female	D
9	10	Big John	Four	55	female	D
28	29	Tess Played	Seven	55	male	D
5	6	Alex John	Four	55	male	D
2	3	Arnold	Three	55	male	D
16	17	Tumyu	Six	54	male	D
18	19	Tinny	Nine	18	male	D

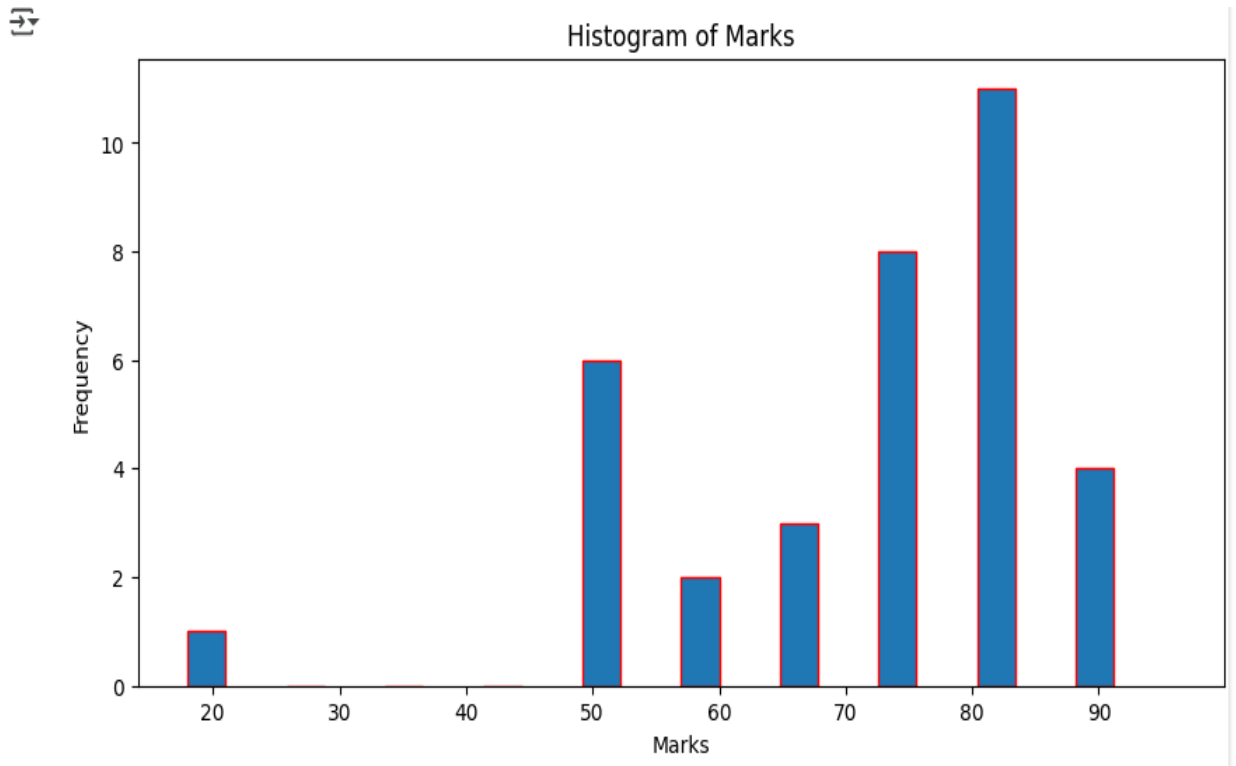
Exercise 6: Exporting Data

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

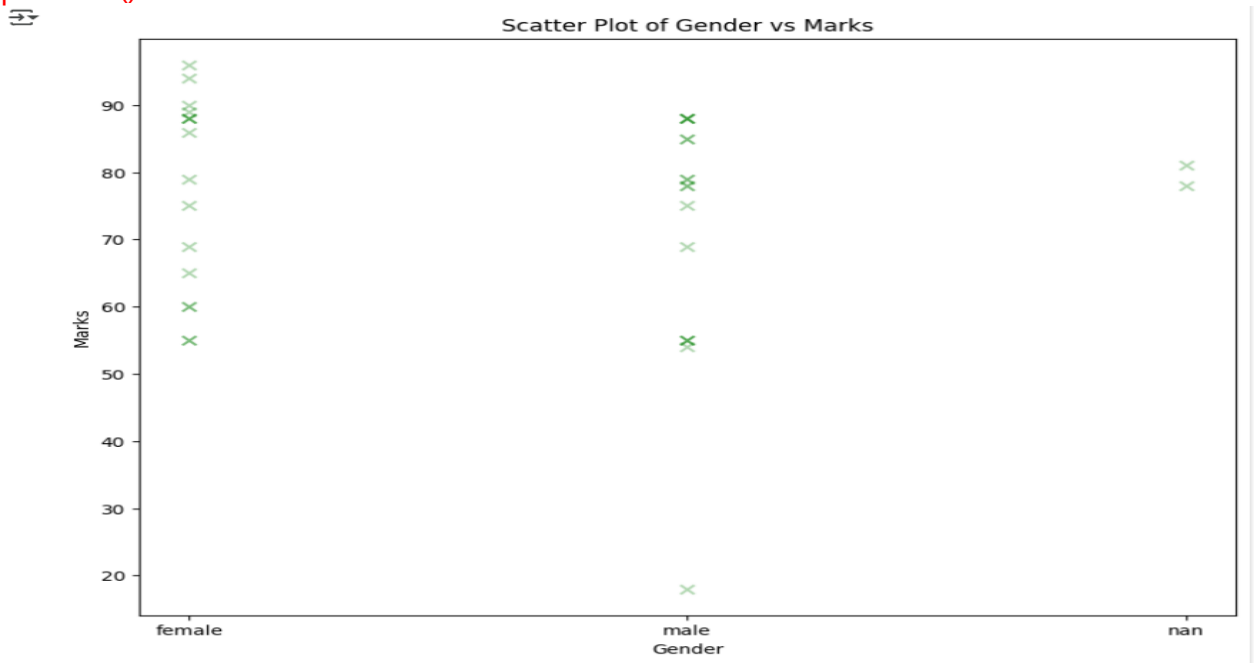
```
#code to save the DataFrame with the new 'grade' column to a new CSV file.
dataframe.to_csv('student_with_grade.csv', index=False)
```

Exercise 7: If finished early try visualising the results

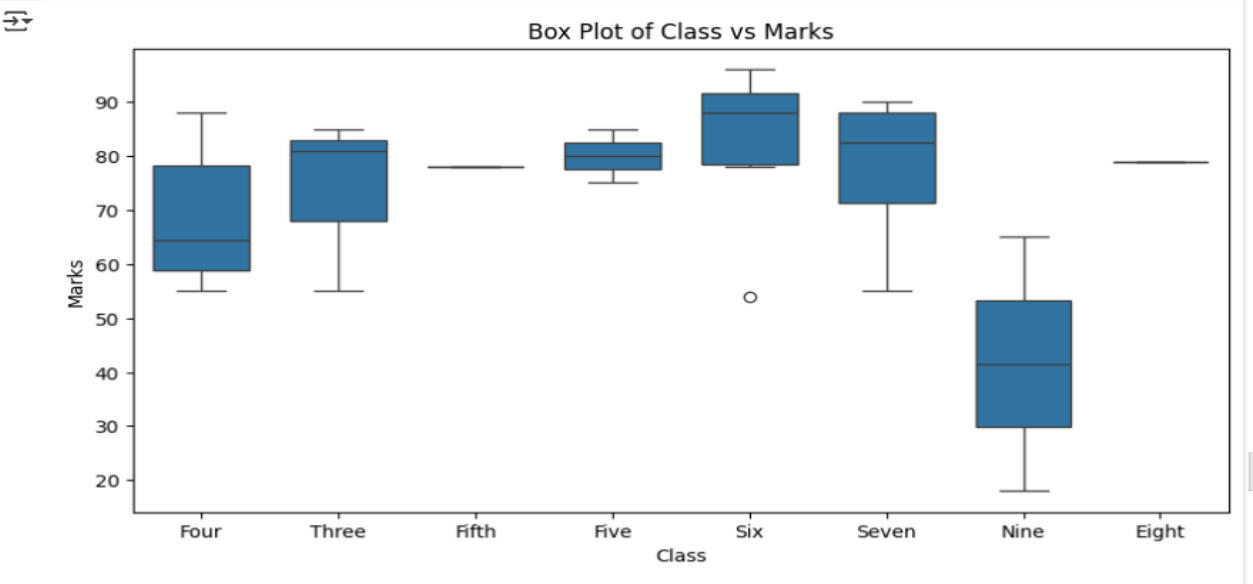
```
#Histogram plot
plt.figure(figsize=(10,5))
plt.hist(df['mark'], bins=10,  edgecolor='red',width=3)
plt.title('Histogram of Marks')
plt.xlabel('Marks')
plt.ylabel('Frequency')
plt.show()
```



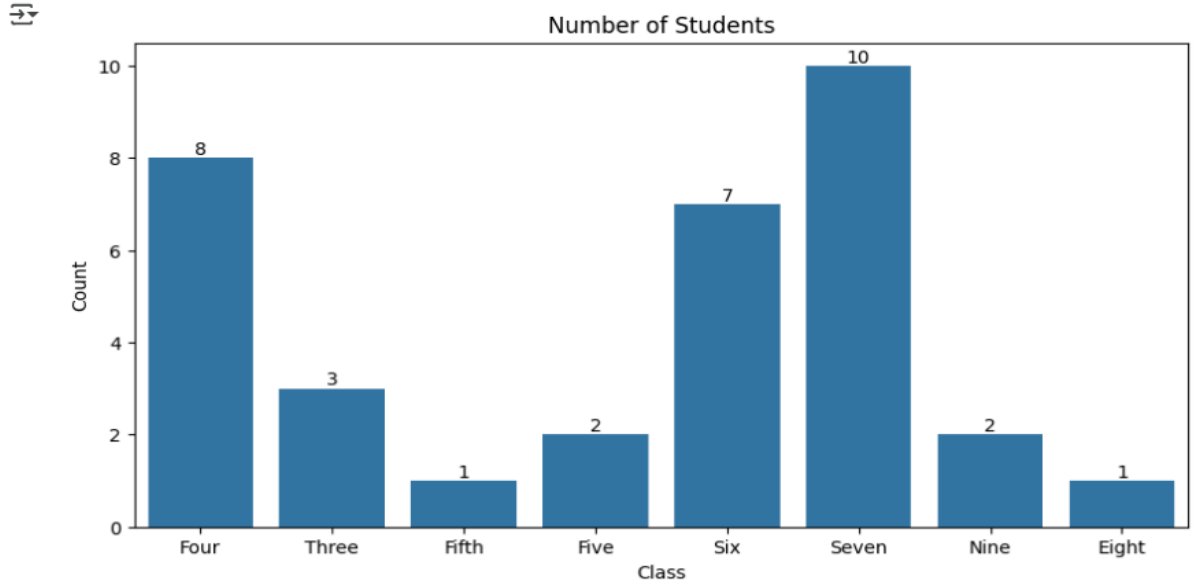
```
#Scatter plot
df['gender']=df['gender'].astype(str)
plt.figure(figsize=(10,8))
plt.scatter(df['gender'], df['mark'], alpha=0.3, color='green', marker='x', s=50)
plt.title('Scatter Plot of Gender vs Marks')
plt.xlabel('Gender')
plt.ylabel('Marks')
plt.show()
```



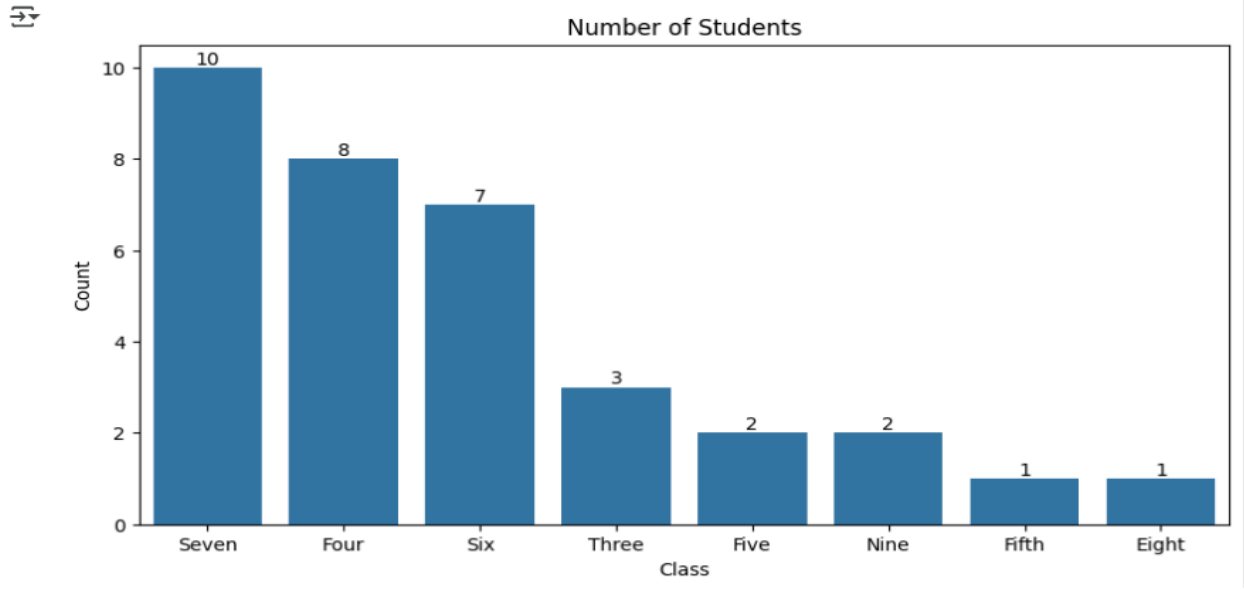

```
#Box Plot
plt.figure(figsize=(10,5))
sns.boxplot(x='class', y='mark', data=df, width=0.7)
plt.title('Box Plot of Class vs Marks')
plt.xlabel('Class')
plt.ylabel('Marks')
plt.show()
```



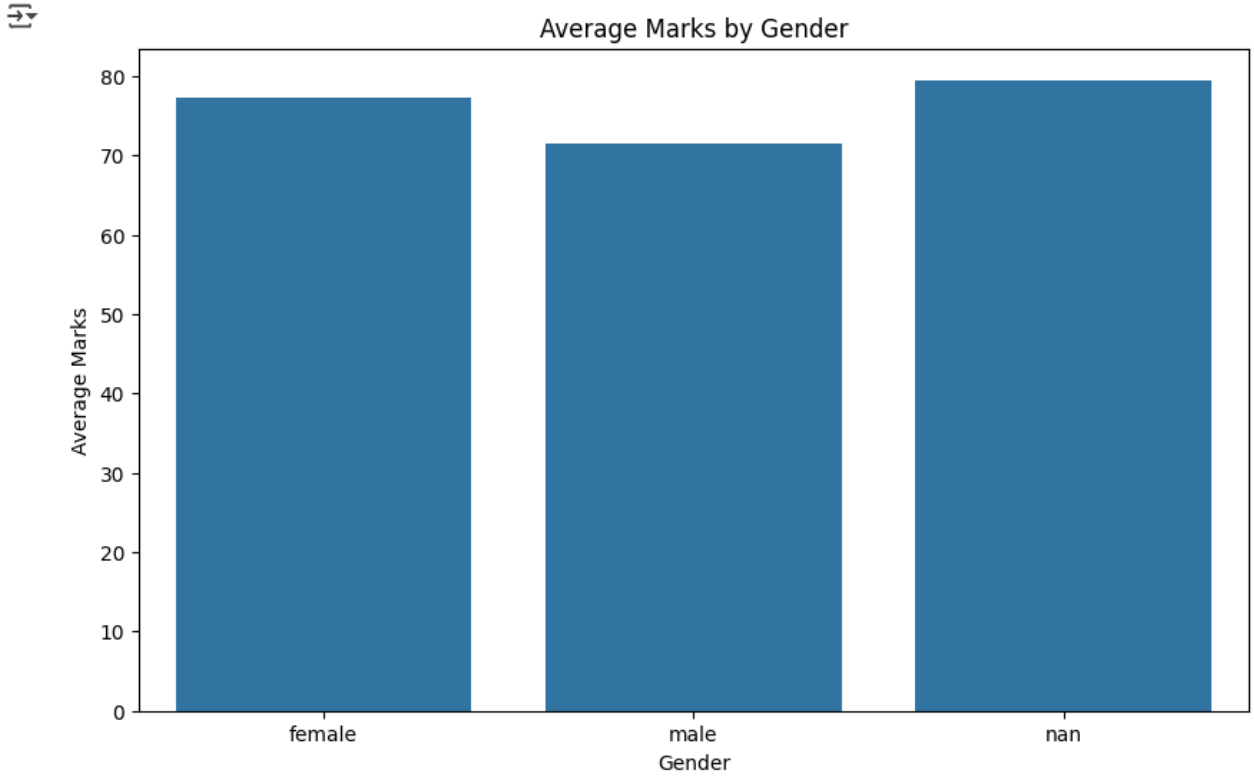
```
#Count Plot
plt.figure(figsize=(10,5))
std=sns.countplot(x='class',data=df)
std.bar_label(std.containers[0])
plt.title('Number of Students')
plt.xlabel('Class')
plt.ylabel('Count')
plt.show()
```




```
#To sort in descending order
plt.figure(figsize=(10,5))
std=sns.countplot(x='class',data=df, order=df['class'].value_counts().index)
std.bar_label(std.containers[0])
plt.title('Number of Students')
plt.xlabel('Class')
plt.ylabel('Count')
plt.show()
```



```
#Average Marks by Gender
plt.figure(figsize=(10,6))
avg_marks_gender=df.groupby('gender')['mark'].mean().reset_index()
sns.barplot(x='gender',y='mark',data=avg_marks_gender)
plt.title('Average Marks by Gender')
plt.xlabel('Gender')
plt.ylabel('Average Marks')
plt.show()
```



Day 4: Task 1

Using the 'GDP (nominal) per Capita.csv' which can be downloaded here, 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 Jupyter notebook
- Print the first 10 rows
- Print the last 5 rows
- Print 'Country/Territory' and 'UN_Region' columns

```
# Read and save the 'GDP (nominal) per Capita' data to a data frame called "df" in notebook.
df = pd.read_csv('GDP (nominal) per Capita.csv')
df.to_csv('GDP (nominal) per Capita.csv', index=False)

#Print the first 10 rows
df.head(10)
```

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

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
#Print the last rows
df.tail()
```

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

```
# Print 'Country/Territory' and 'UN_Region' columns
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 × 2 columns

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 here. 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.

```
# Countries below average by IMF_Estimate
gdp=df[df["IMF_Estimate"]<df["IMF_Estimate"].mean()]
gdp.iloc[:,1:4]
```

	UN_Region	IMF_Estimate	IMF_Year
1	Europe	0	0
2	Europe	0	0
5	Americas	0	0
9	Europe	0	0
10	Americas	0	0
...	...	...	...
219	Africa	496	2023
220	Africa	467	2023
221	Africa	415	2023
222	Asia	611	2020
223	Africa	249	2023

159 rows × 3 columns

```
#Country has highest UN_Estimate
gdp=df[df["UN_Estimate"]==df["UN_Estimate"].max()]
gdp['Country/Territory']
```

	Country/Territory
1	Monaco

```
#Country has highest Worlbank Estimate
gdp=df[df["WorldBank_Estimate"]==df["WorldBank_Estimate"].max()]
gdp['Country/Territory']
```

	Country/Territory
1	Monaco

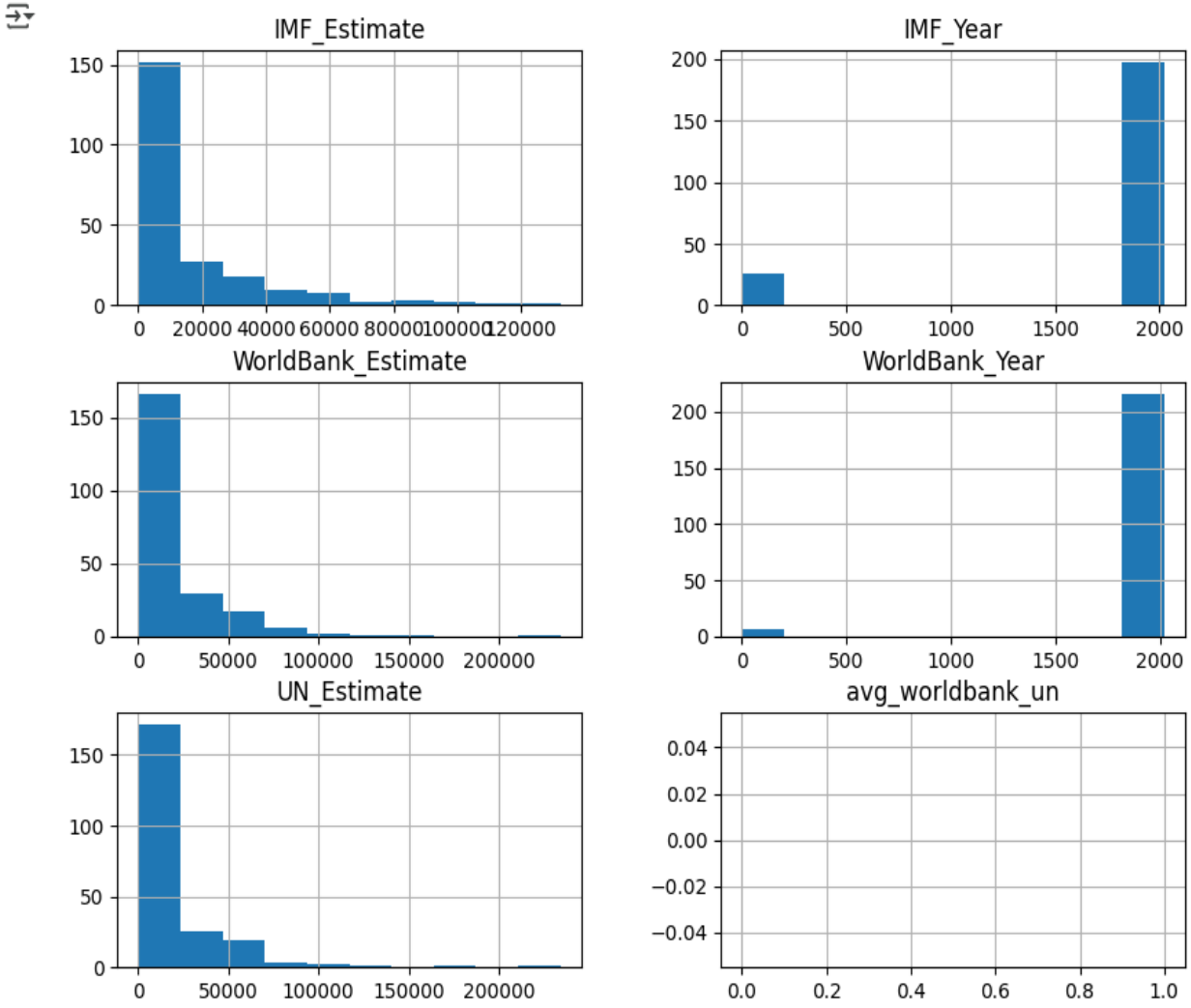
```
#Country has highest IMF Estimate
gdp=df[df["IMF_Estimate"]==df["IMF_Estimate"].max()]
gdp['Country/Territory']
```

	Country/Territory
3	Luxembourg

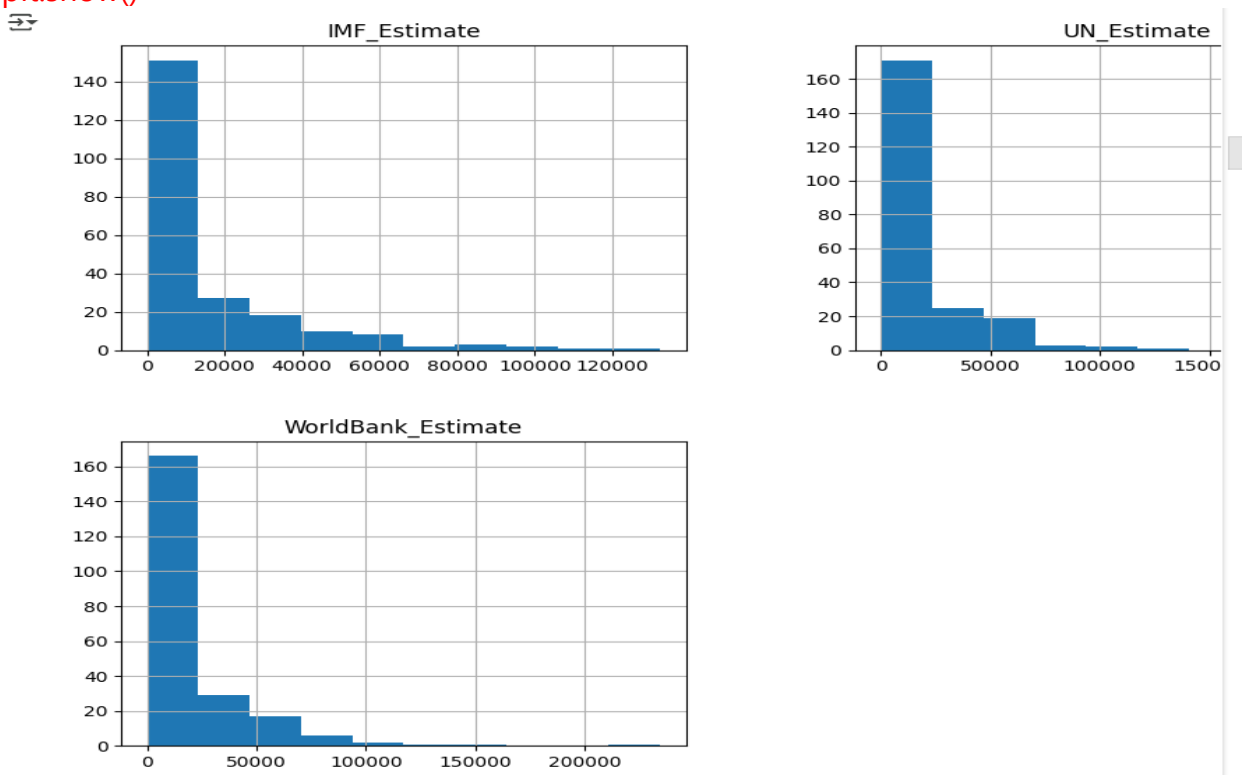
```
# Calculate the average of 'Worldbank_Estimate' and 'UN_Estimate' columns
avg_worldbank_UN = df[['WorldBank_Estimate', 'UN_Estimate']].mean()
avg_worldbank_UN
```

WorldBank_Estimate	18927.417040
UN_Estimate	17767.304933

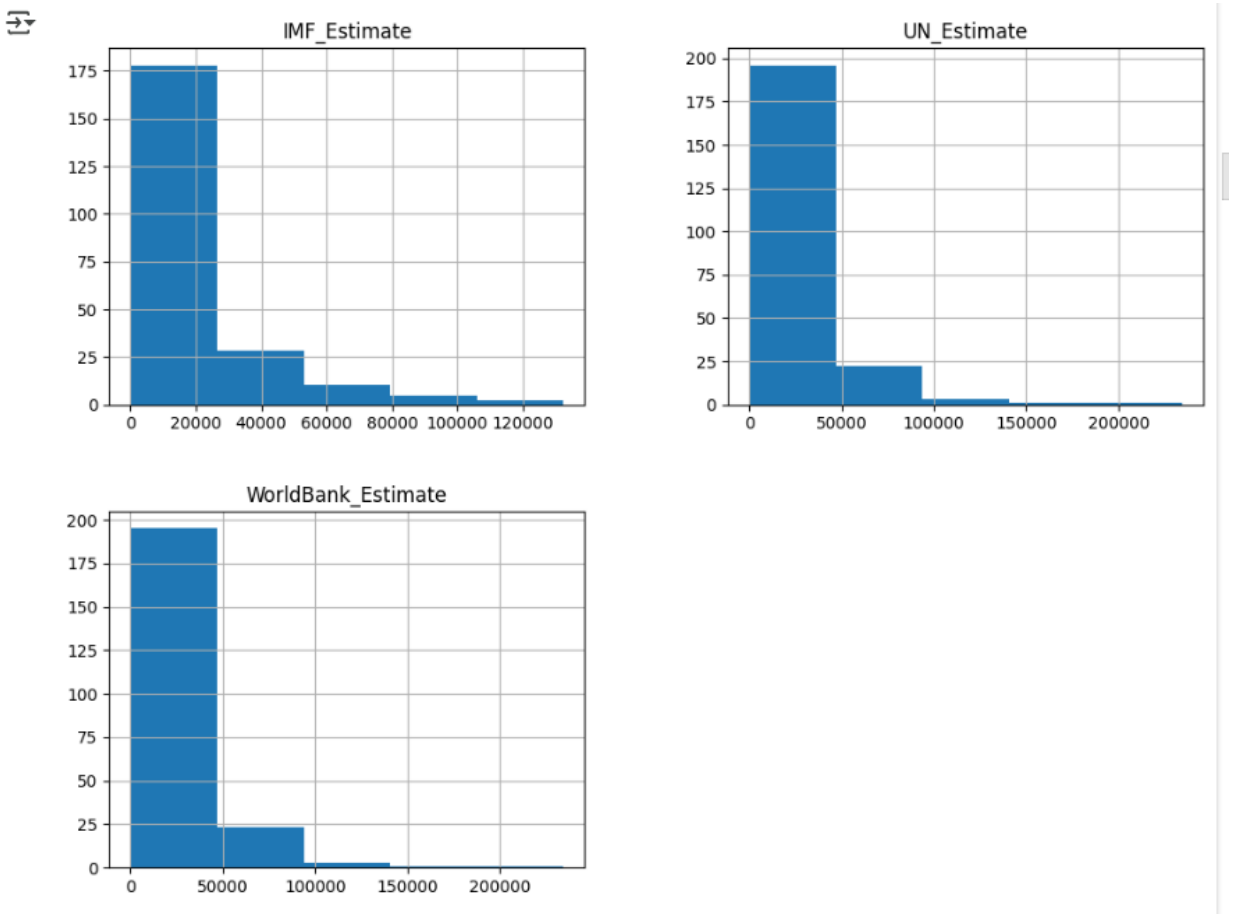
```
#Histogram
df.hist(figsize=(10,8))
plt.show()
```



```
df[["IMF_Estimate", "UN_Estimate", "WorldBank_Estimate"]].hist(figsize=(12,9))
plt.show()
```



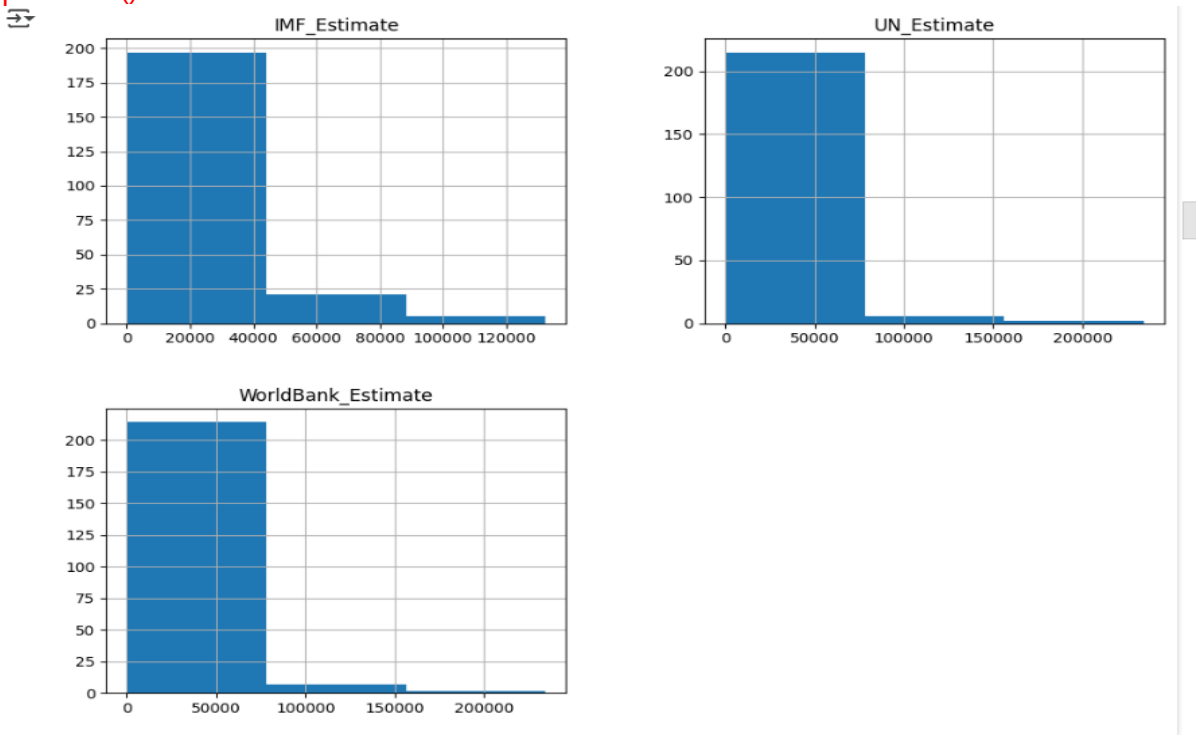

```
df[["IMF_Estimate", "UN_Estimate", "WorldBank_Estimate"]].hist(bins=5, figsize=(12,9))
plt.show()
```



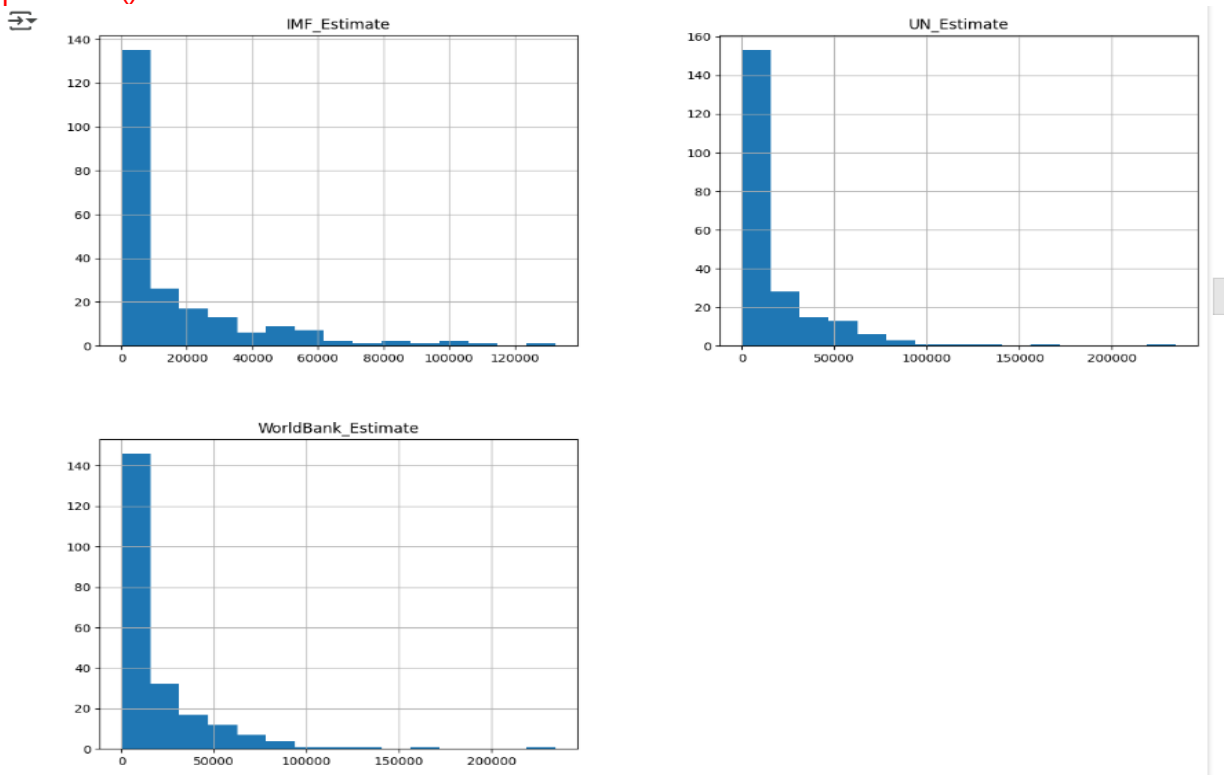
```
df["WorldBank_Estimate"].agg(["min","max"])
```

	WorldBank_Estimate
min	0
max	234316

```
df[["IMF_Estimate", "UN_Estimate", "WorldBank_Estimate"]].hist(bins=3, figsize=(12,9))
plt.show()
```



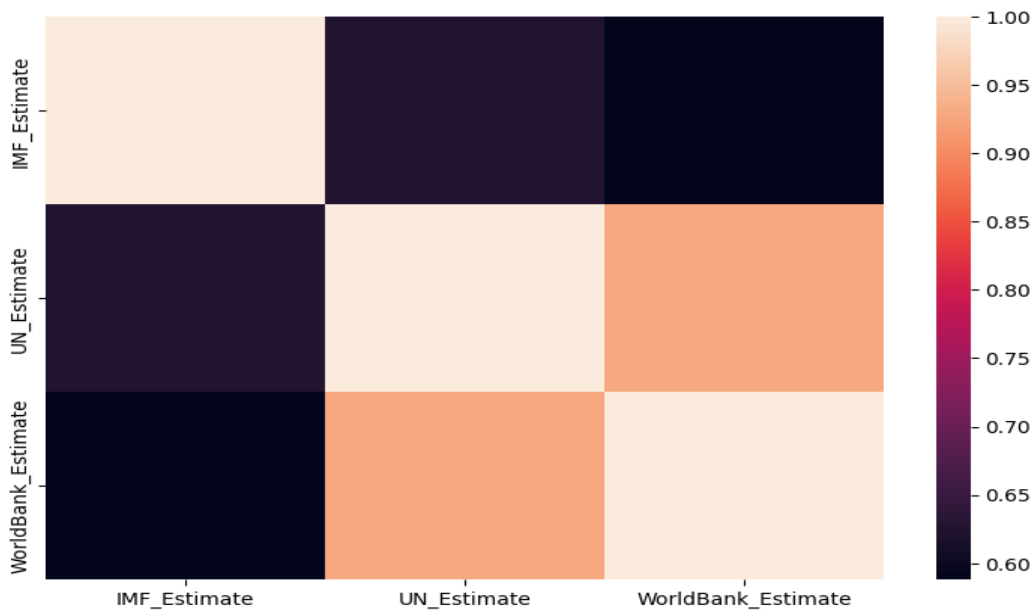

```
df[["IMF_Estimate", "UN_Estimate", "WorldBank_Estimate"]].hist(bins=15, figsize=(15,12))
plt.show()
```



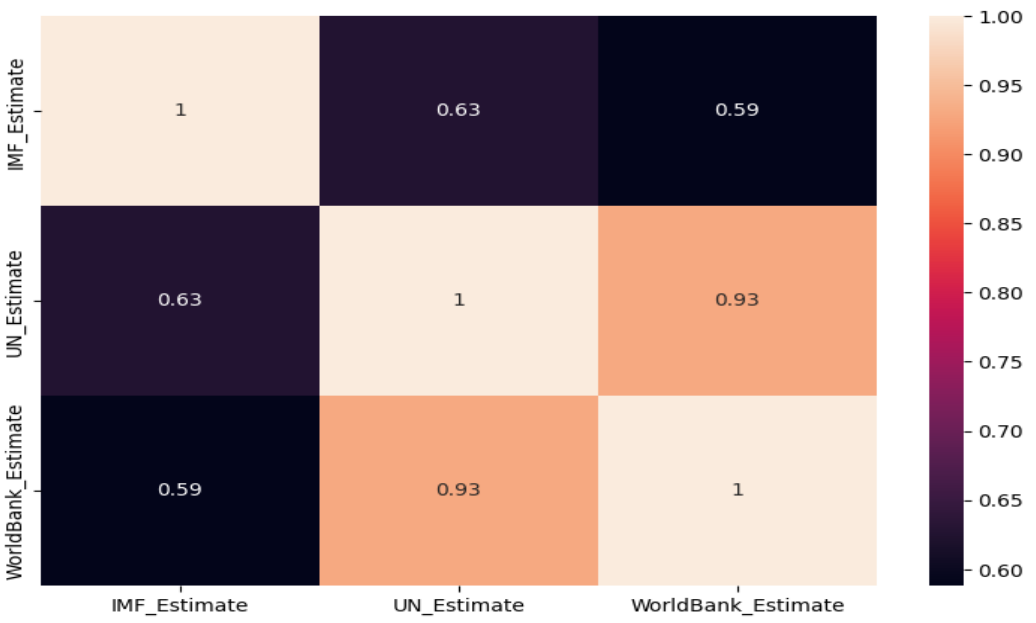
```
df[["IMF_Estimate", "UN_Estimate", "WorldBank_Estimate"]].corr()
```

	IMF_Estimate	UN_Estimate	WorldBank_Estimate
IMF_Estimate	1.000000	0.626513	0.587988
UN_Estimate	0.626513	1.000000	0.930331
WorldBank_Estimate	0.587988	0.930331	1.000000

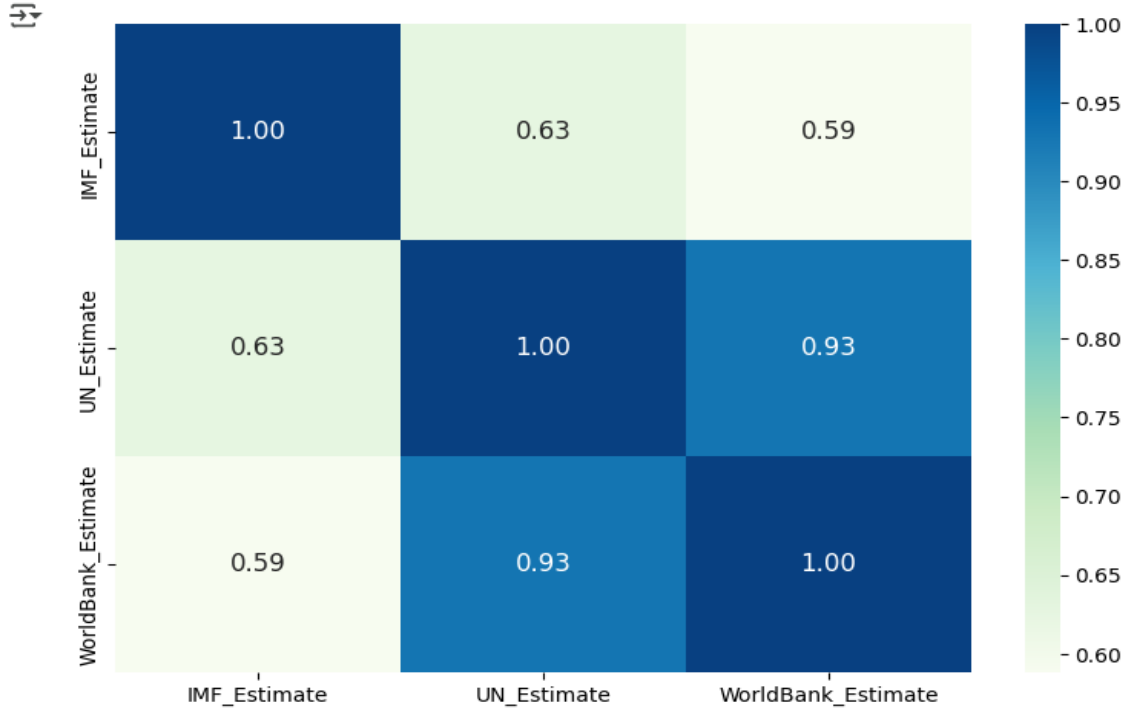
```
corr = df[["IMF_Estimate", "UN_Estimate", "WorldBank_Estimate"]].corr()
plt.figure(figsize=(9,6))
sns.heatmap(corr)
plt.show()
```



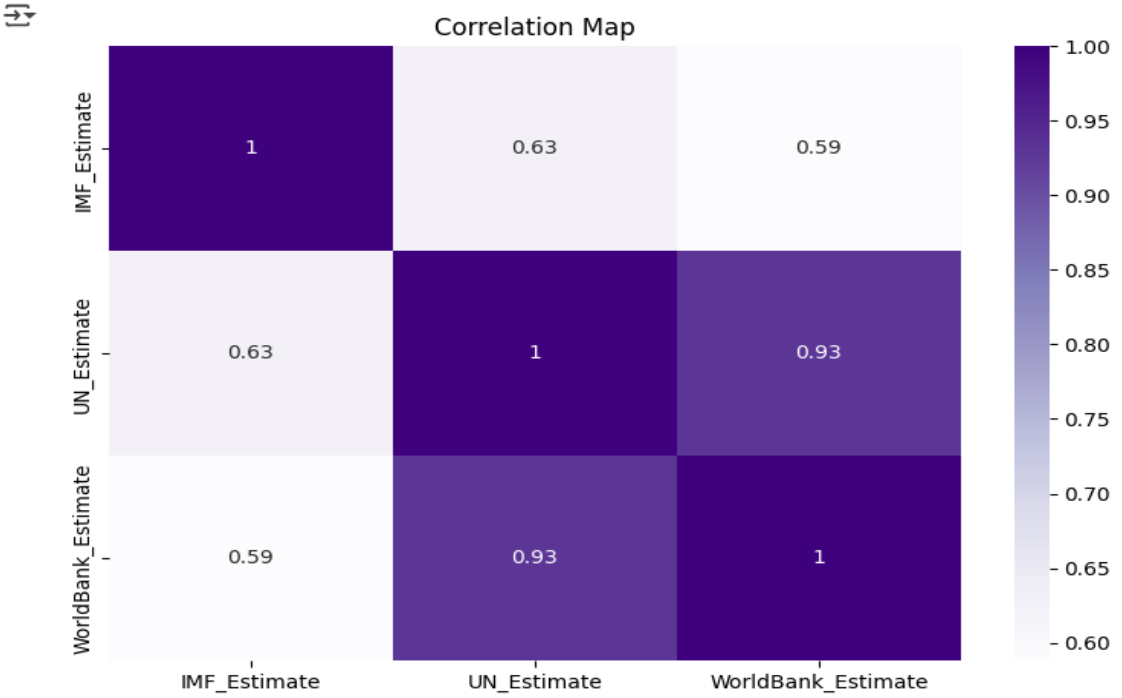

```
corr = df[["IMF_Estimate", "UN_Estimate", "WorldBank_Estimate"]].corr()
plt.figure(figsize=(9,6))
sns.heatmap(corr, annot=True)
plt.show()
```



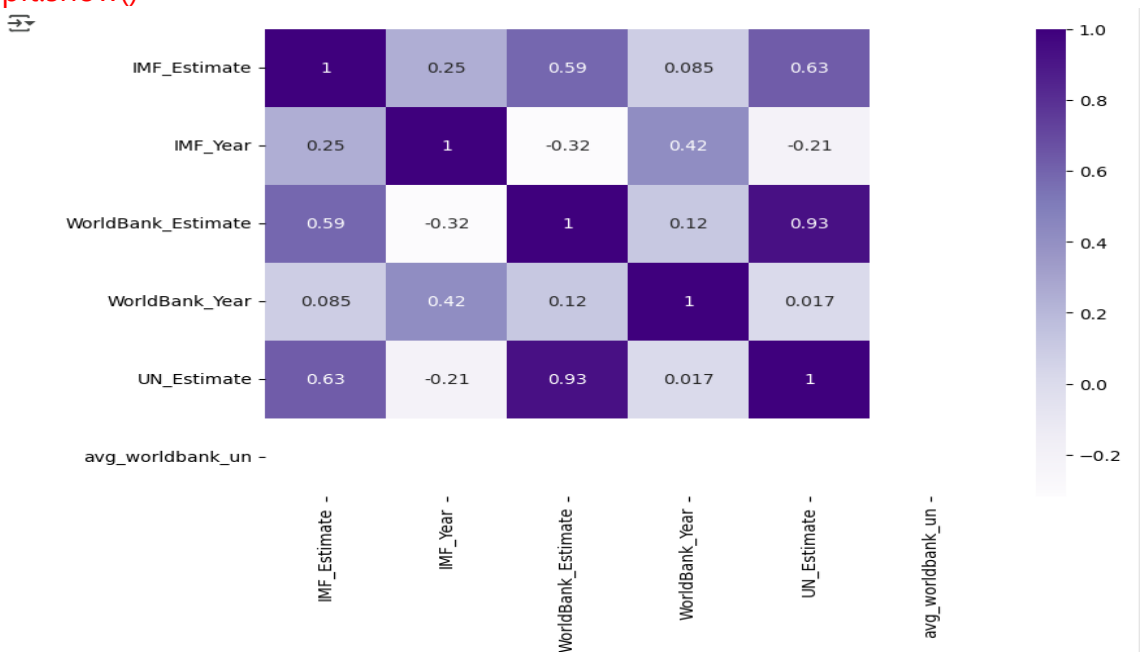
```
corr = df[["IMF_Estimate", "UN_Estimate", "WorldBank_Estimate"]].corr()
plt.figure(figsize=(9,6))
sns.heatmap(corr, annot=True, fmt=".2f", cmap = 'GnBu', annot_kws={"size": 12})
plt.show()
```




```
corr = df[["IMF_Estimate", "UN_Estimate", "WorldBank_Estimate"]].corr()
plt.figure(figsize=(9,6))
sns.heatmap(corr, annot=True, cmap = 'Purples')
plt.title("Correlation Map")
plt.show()
```



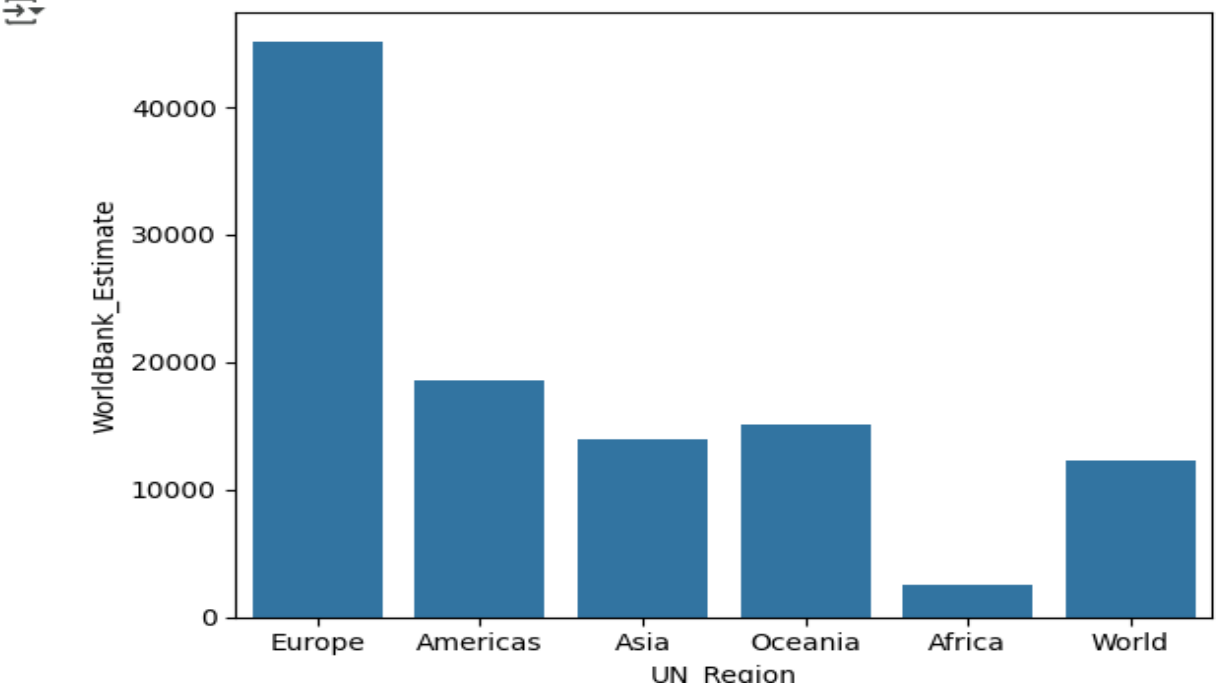
```
corr = df.select_dtypes(include=[int, float]).corr()
plt.figure(figsize=(9,6))
sns.heatmap(corr, annot=True, cmap = 'Purples')
plt.show()
```



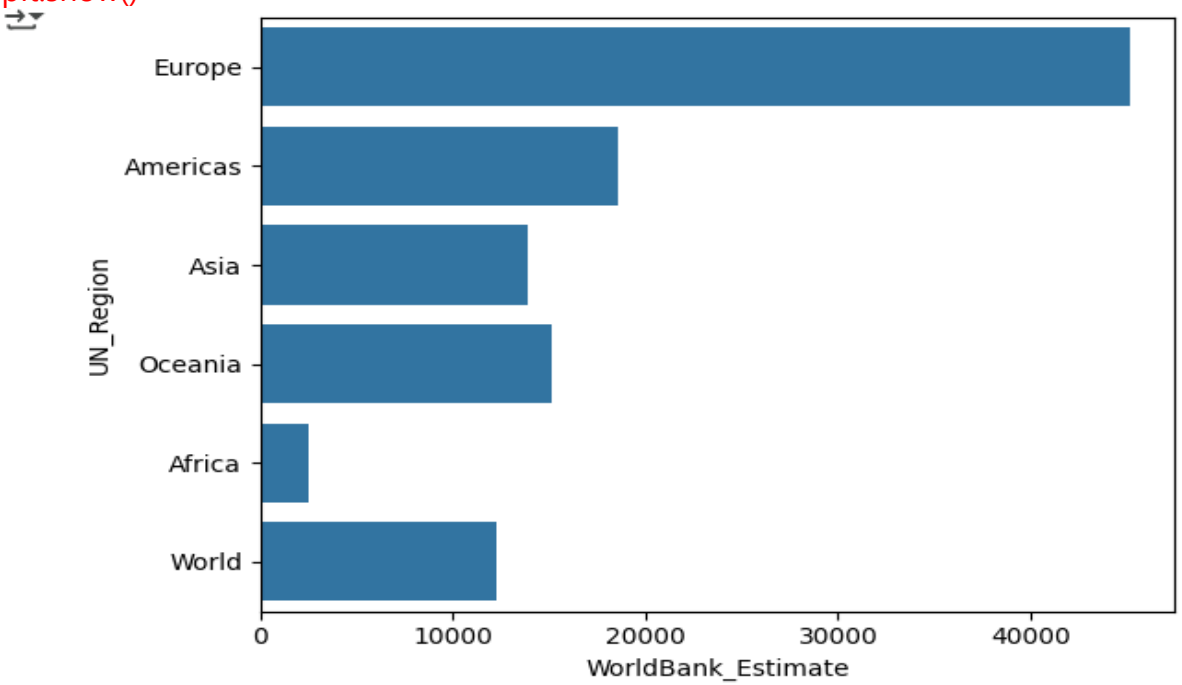

```
df.head()
```

	Country/Territory	UN_Region	IMF_Estimate	IMF_Year	WorldBank_Estimate	WorldBank_Year	UN_Estimate	UN_
1	Monaco	Europe	0	0	234316	2021	234317	
2	Liechtenstein	Europe	0	0	157755	2020	169260	
3	Luxembourg	Europe	132372	2023	133590	2021	133745	
4	Ireland	Europe	114581	2023	100172	2021	101109	
5	Bermuda	Americas	0	0	114090	2021	112653	

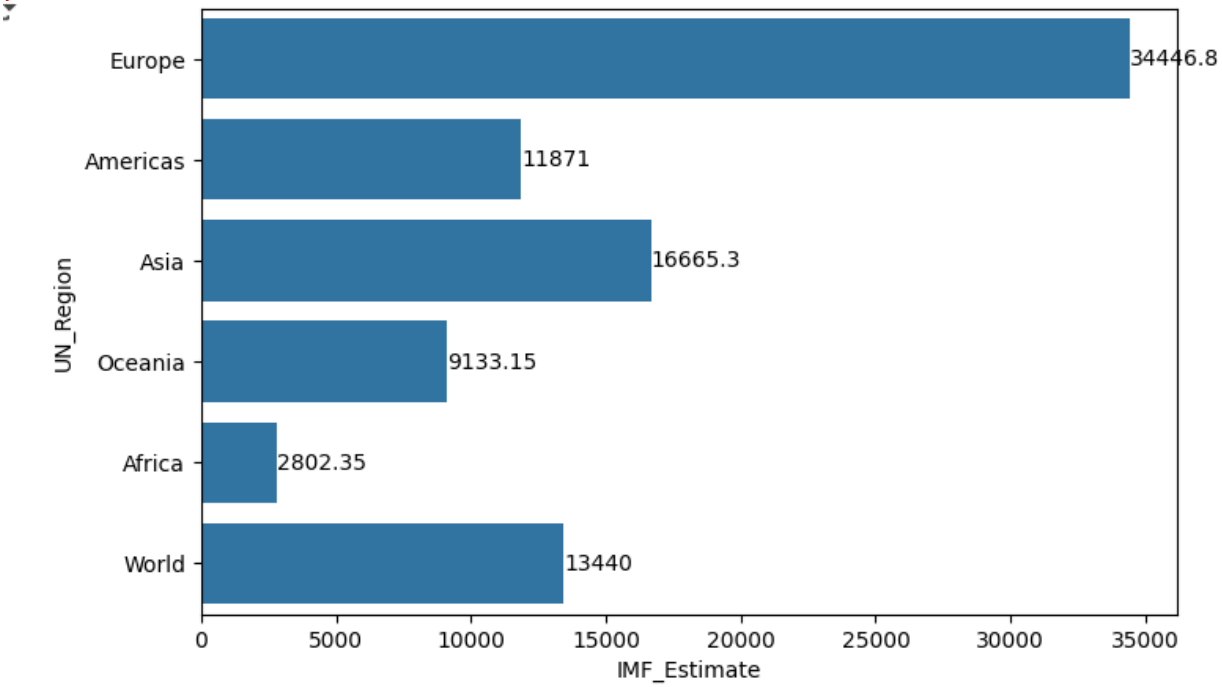
```
sns.barplot(x="UN_Region", y="WorldBank_Estimate", data=df, errorbar=None)
plt.show()
```



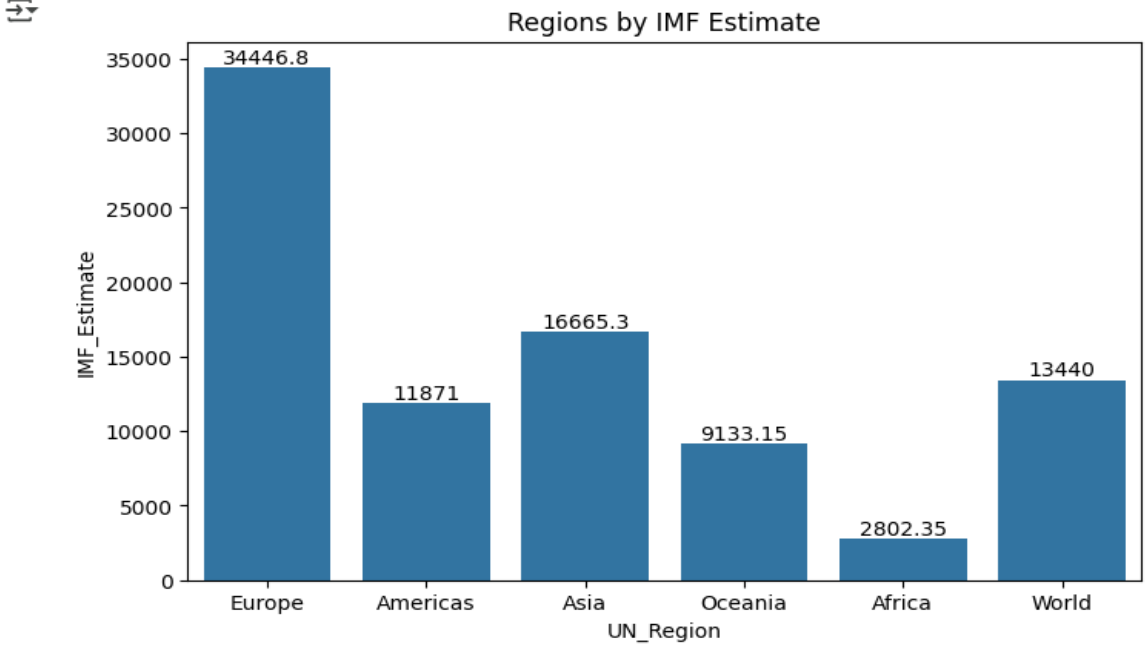
```
sns.barplot(x="WorldBank_Estimate", y="UN_Region", data=df, errorbar=None)
plt.show()
```



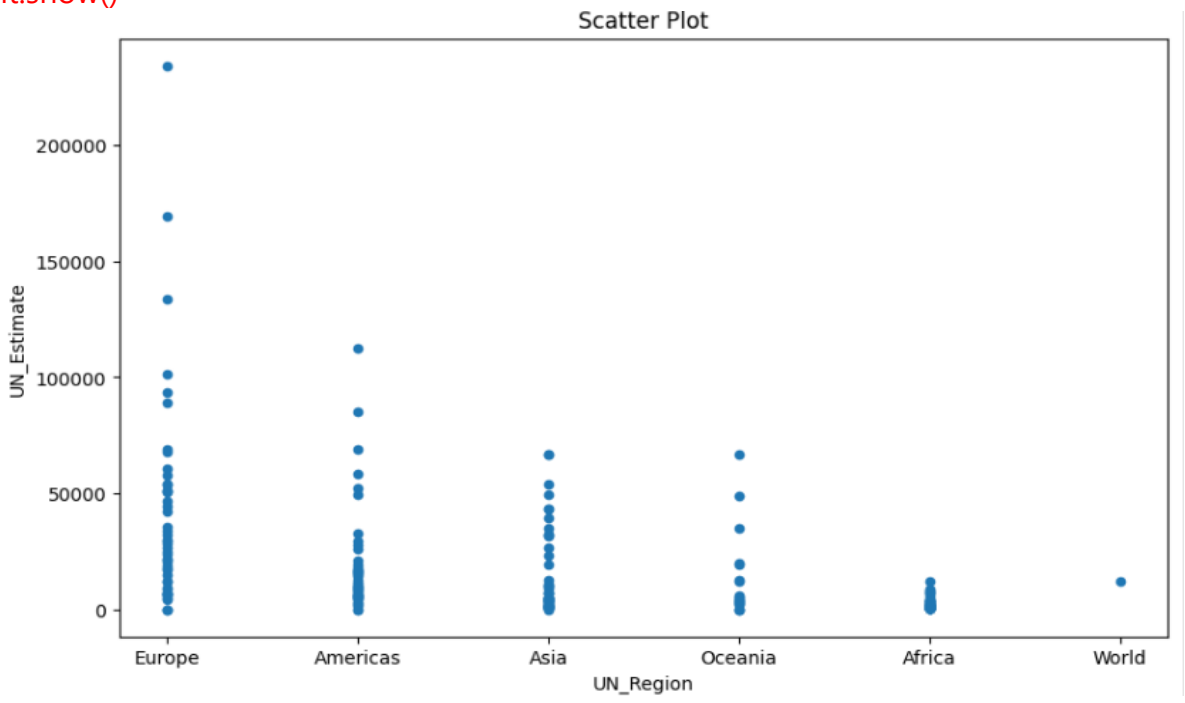

```
fig = plt.figure(figsize = (8,5))
ax = sns.barplot(x = "IMF_Estimate",  y = "UN_Region",
data = df, errorbar = None)
ax.bar_label(ax.containers[0])
plt.show()
```



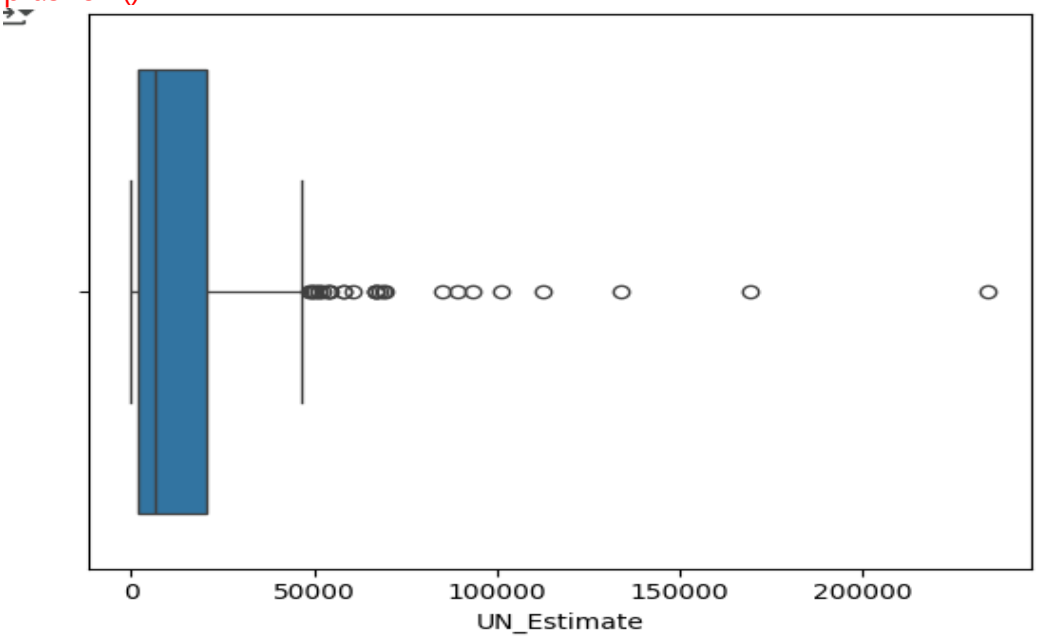
```
fig = plt.figure(figsize = (8,5))
ax = sns.barplot(x = "UN_Region",  y = "IMF_Estimate",
                 data = df, errorbar = None)
ax.bar_label(ax.containers[0])
ax.set_title("Regions by IMF Estimate")
plt.show()
```




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



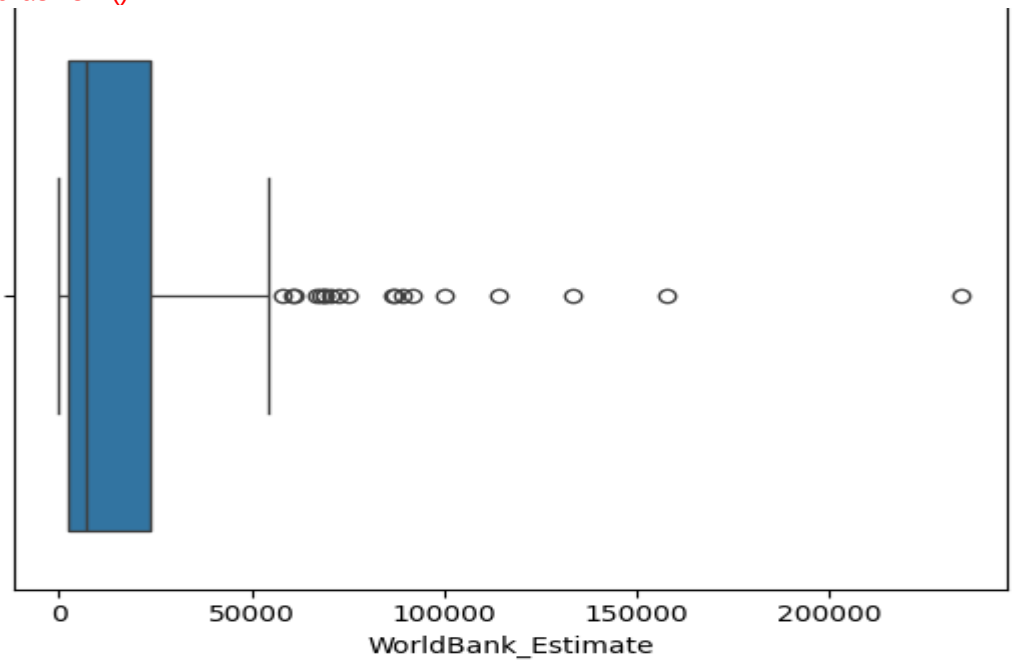
```
#Boxplot
sns.boxplot(x=df["UN_Estimate"])
plt.show()
```



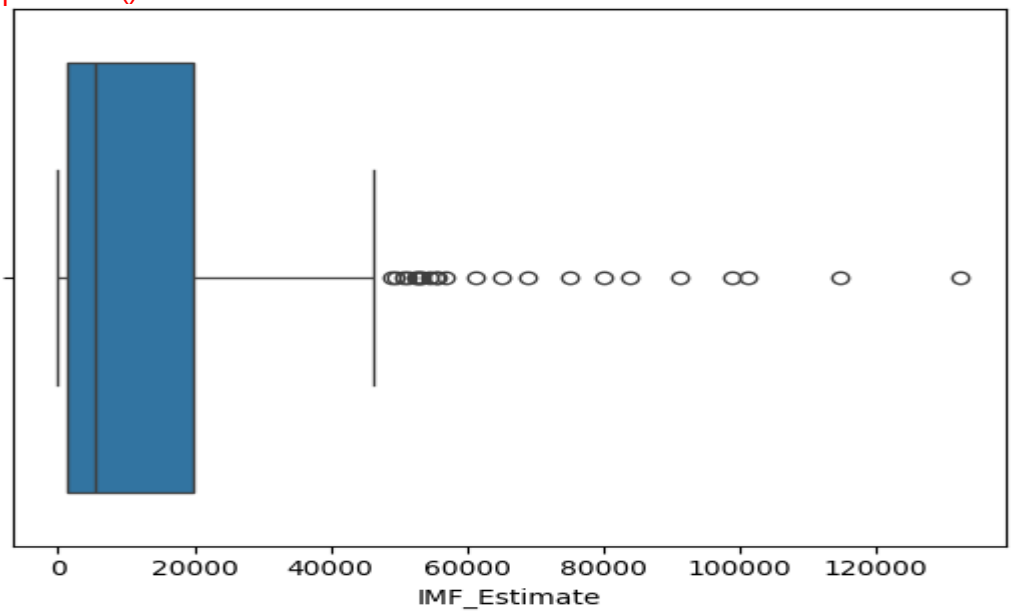

```
df[df["UN_Estimate"]>50000].head()
```

	Country/Territory	UN_Region	IMF_Estimate	IMF_Year	WorldBank_Estimate	WorldBank_Year	UN_Estimate	UN_
1	Monaco	Europe	0	0	234316	2021	234317	
2	Liechtenstein	Europe	0	0	157755	2020	169260	
3	Luxembourg	Europe	132372	2023	133590	2021	133745	
4	Ireland	Europe	114581	2023	100172	2021	101109	
5	Bermuda	Americas	0	0	114090	2021	112653	

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



```
sns.boxplot(x=df["IMF_Estimate"])
plt.show()
```




```
df[df["UN_Estimate"]>100000]
```

	Country/Territory	UN_Region	IMF_Estimate	IMF_Year	WorldBank_Estimate	WorldBank_Year	UN_Estimate	UN_
1	Monaco	Europe	0	0	234316	2021	234317	
2	Liechtenstein	Europe	0	0	157755	2020	169260	
3	Luxembourg	Europe	132372	2023	133590	2021	133745	
4	Ireland	Europe	114581	2023	100172	2021	101109	
5	Bermuda	Americas	0	0	114090	2021	112653	

```
#Create another dataframe called data excluding 5 countries with highest UN estimate
data = df[-(df["UN_Estimate"]>100000)]
data.head()
```

	Country/Territory	UN_Region	IMF_Estimate	IMF_Year	WorldBank_Estimate	WorldBank_Year	UN_Estimate	UN
6	Norway	Europe	101103	2023	89154	2021	89242	
7	Switzerland	Europe	98767	2023	91992	2021	93525	
8	Singapore	Asia	91100	2023	72794	2021	66822	
9	Isle of Man	Europe	0	0	87158	2019	0	
10	Cayman Islands	Americas	0	0	86569	2021	85250	

Removing outliers

```
lower_q = df["UN_Estimate"].quantile(0.25)
lower_q
```

```
2039.0
```

```
[100] higher_q = df["UN_Estimate"].quantile(0.75)
higher_q
```

```
20740.0
```

```
[101] iqr = higher_q - lower_q
iqr
```

```
18701.0
```

```
[102] upper_boundary = higher_q + 1.5 * iqr
upper_boundary
```

```
48791.5
```

```
[103] lower_boundary = lower_q - 1.5 * iqr
lower_boundary
```

```
-26012.5
```

```
df_filtered = df[(df["UN_Estimate"] < upper_boundary) & (df["UN_Estimate"] > lower_boundary)]
df_filtered.head()
```

	Country/Territory	UN_Region	IMF_Estimate	IMF_Year	WorldBank_Estimate	WorldBank_Year	UN_Estimate	UN
9	Isle of Man	Europe	0	0	87158	2019	0	
14	Channel Islands	Europe	0	0	75153	2007	0	
15	Faroe Islands	Europe	0	0	69010	2021	0	
29	Macau	Asia	50571	2023	43874	2021	43555	
30	United Arab Emirates	Asia	49451	2023	44316	2021	43295	

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.