

Домашнее задание №1. Дескриптивный анализ данных

Кадыров Т.И. ИУ6-22М

Цель работы

Приобрести опыт решения практических задач по анализу данных, таких как загрузка, трансформация, вычисление простых статистик и визуализация данных в виде графиков и диаграмм, посредством языка программирования Python.

Расчет варианта

```
In [ ]: surname = "Кадыров" # Ваша фамилия

alp = 'абвгдеёжзийклмнопрстуфхцчшщъыьэюя'
w = [1, 42, 21, 21, 34, 6, 44, 26, 18, 44, 38, 26, 14, 43, 4, 49, 45,
      7, 42, 29, 4, 9, 36, 34, 31, 29, 5, 30, 4, 19, 28, 25, 33]

d = dict(zip(alp, w))
variant = sum([d[el] for el in surname.lower()]) % 40 + 1

print("Задача № 1, шаг 5 - вариант: ", variant % 5 + 1)
print("Задача № 1, шаг 11 - вариант: ", variant % 2 + 1)
print("задача № 2 - вариант: ", variant % 4 + 1)
```

Задача № 1, шаг 5 - вариант: 4
Задача № 1, шаг 11 - вариант: 2
задача № 2 - вариант: 4

Задание 1. Анализ индикаторов качества государственного управления (The Worldwide Government Indicators, WGI)

1.1 Загрузите данные в DataFrame

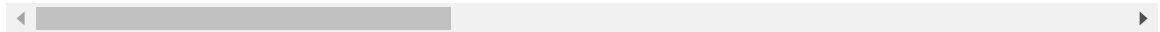
```
In [ ]: import pandas as pd

In [ ]: # Загружаем data frame, используем двойной заголовок, чтобы сохранить год
df = pd.read_excel('./data/wgidataset.xlsx', sheet_name='ControlofCorrupt')
df.head()
```

Out[]:

		Unnamed: 0_level_0	Unnamed: 1_level_0				
	Country/Territory	Code	Estimate	StdErr	NumSrc	Rank	Lower
0	Aruba	ABW	NaN	NaN	NaN	NaN	NaN
1	Andorra	ADO	1.318143	0.480889	1.0	87.096771	72.043015
2	Afghanistan	AFG	-1.291705	0.340507	2.0	4.301075	0.000000
3	Angola	AGO	-1.167702	0.262077	4.0	9.677420	0.537634
4	Anguilla	AIA	NaN	NaN	NaN	NaN	NaN

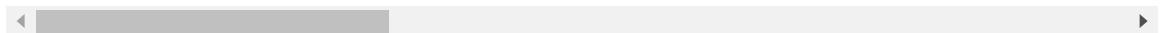
5 rows × 146 columns



```
In [ ]: # Объединяем год и название столбца и приводим все к одинарному заголовку
new_columns = [col[1] if col[1] in ['Country/Territory', 'Code'] else f"{
df.columns = new_columns
df.head()
```

	Country/Territory	Code	1996.Estimate	1996.StdErr	1996.NumSrc	1996.Rank	199
0	Aruba	ABW	NaN	NaN	NaN	NaN	
1	Andorra	ADO	1.318143	0.480889	1.0	87.096771	72
2	Afghanistan	AFG	-1.291705	0.340507	2.0	4.301075	0
3	Angola	AGO	-1.167702	0.262077	4.0	9.677420	0
4	Anguilla	AIA	NaN	NaN	NaN	NaN	

5 rows × 146 columns

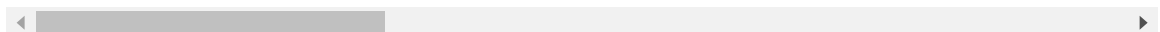


1.2 Отсортируйте данные по убыванию индекса DataFrame

```
In [ ]: df_desc = df.iloc[::-1]
df_desc.head()
```

	Country/Territory	Code	1996.Estimate	1996.StdErr	1996.NumSrc	1996.Rank	1
213	Zimbabwe	ZWE	-0.278847	0.244907	5.0	47.849461	
212	Zambia	ZMB	-0.840641	0.262077	4.0	24.731182	
211	Congo, Dem. Rep.	ZAR	-1.647852	0.315914	3.0	0.000000	
210	South Africa	ZAF	0.732927	0.210325	6.0	76.344086	
209	Serbia	SRB	-1.140072	0.262077	4.0	11.827957	

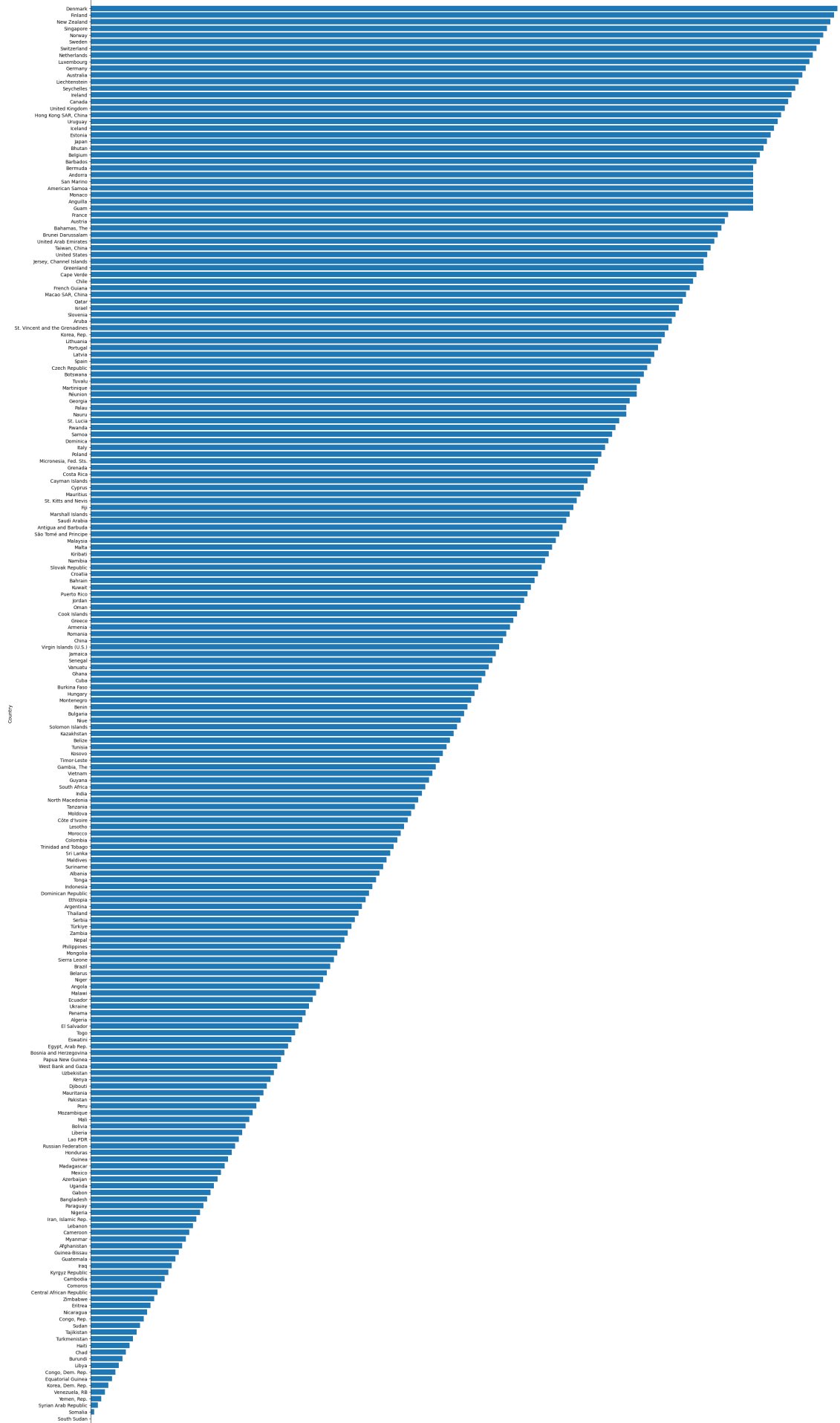
5 rows × 146 columns



1.3 Отобразите данные по индексу WGI за 2022 год в виде горизонтального столбчатого графика

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

```
In [ ]: # Сортируем по полю 2022.Rank
df_sort = df.sort_values(by='2022.Rank')
# Удаляем те строки, у которых за этот год нет данных
df_sort_dropna = df_sort.dropna(subset=['2022.Rank'])
plt.figure(figsize=(30,60))
plt.barh(df_sort_dropna['Country/Territory'], df_sort_dropna['2022.Rank'])
plt.xlabel('Rank')
plt.ylabel('Country')
plt.title('Страны по индексу WGI за 2022 год')
plt.show()
```



1.4 Сформируйте DataFrame из исходного для региона в соответствии с Вашим вариантом

Вариант 4. Middle East and North Africa (MENA)

```
In [ ]: # Исходный датасет
df_region = pd.read_excel('./data/regions.xlsx')
# Страны региона MENA
df_mena = df_region[df_region['Region'] == 'MENA']
# WGI индексы для стран MENA
df_wgi_mena = df.merge(df_mena, how='inner', left_on='Code', right_on='Co
# Удаляем столбец Country/Territory. Выставляем индекс по Country
df_wgi_mena.set_index('Country', inplace=True)
df_wgi_mena.drop(columns='Country/Territory', inplace=True)
```

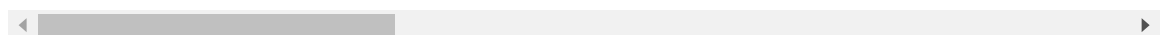
1.5 Выведите данные DataFrame'a

```
In [ ]: df_wgi_mena
```

Out[]: Code 1996.Estimate 1996.StdErr 1996.NumSrc 1996.Rank 1996.Lower 19

Country							
United Arab Emirates	ARE	-0.005579	0.312212	3.0	57.526882	35.483871	7
Bahrain	BHR	0.328689	0.312212	3.0	63.978493	50.537636	7
Algeria	DZA	-0.566741	0.262077	4.0	33.333332	16.666666	5
Egypt	EGY	-0.472254	0.244907	5.0	38.709679	19.892473	5
Iran	IRN	-0.480607	0.262077	4.0	37.634407	18.817204	5
Iraq	IRQ	-1.602183	0.262077	4.0	0.537634	0.000000	
Israel	ISR	1.354008	0.210325	6.0	88.172043	81.182793	9
Jordan	JOR	-0.035407	0.244907	5.0	55.376343	39.784946	6
Kuwait	KWT	0.478682	0.262077	4.0	70.430107	59.139786	8
Lebanon	LBN	-0.659695	0.262077	4.0	31.182796	13.978495	4
Libya	LBY	-0.871937	0.262077	4.0	20.430107	3.763441	3
Morocco	MAR	-0.106927	0.262077	4.0	53.225807	34.946236	6
Oman	OMN	0.414642	0.262077	4.0	67.741936	56.989246	7
Qatar	QAT	-0.045596	0.262077	4.0	54.838711	37.634407	6
Saudi Arabia	SAU	-0.163303	0.262077	4.0	51.075268	32.258064	6
Syria	SYR	-0.881176	0.262077	4.0	19.892473	3.763441	3
Tunisia	TUN	-0.533678	0.262077	4.0	35.483871	17.741936	5
Yemen	YEM	-0.743732	0.262077	4.0	27.419355	8.602151	4

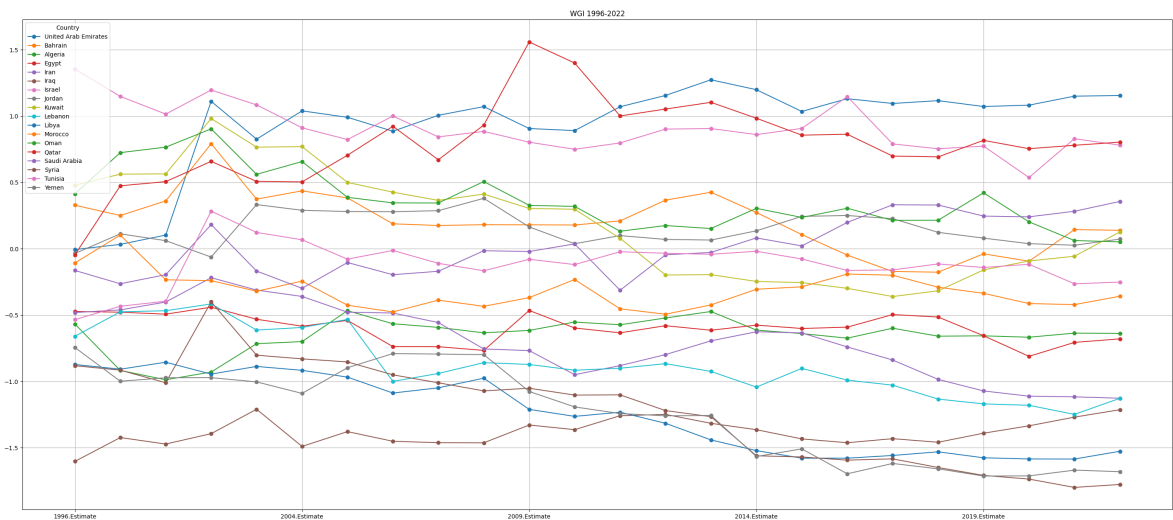
18 rows × 146 columns



1.6 Постройте графики индекса WGI за 1996-2022 для стран своего региона (estimate)

```
In [ ]: # Оставляем только столбцы Estimate и Country
df_plot_mena = df_wgi_mena.filter(regex='Estimate|Country')
# Транспонируем датафрейм и отрисовываем график на каждую страну
df_plot_mena.T.plot(figsize=(35,15), grid=True, marker='o', title='WGI 19
```

Out[]: <Axes: title={'center': 'WGI 1996-2022'}>



1.7 Найдите страны с наибольшим и наименьшим значением WGI Вашего варианта региона за 2022 год

```
In [ ]: # Страна с минимальным значением
minWgi = df_wgi_mena['2022.Estimate'].idxmin()
minWgi
```

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

```
In [ ]: # Страна с максимальным значением
maxWgi = df_wgi_mena['2022.Estimate'].idxmax()
maxWgi
```

```
Out[ ]: 'United Arab Emirates'
```

1.8 Определите средние значения региона за каждый год в период с 1996 по 2022

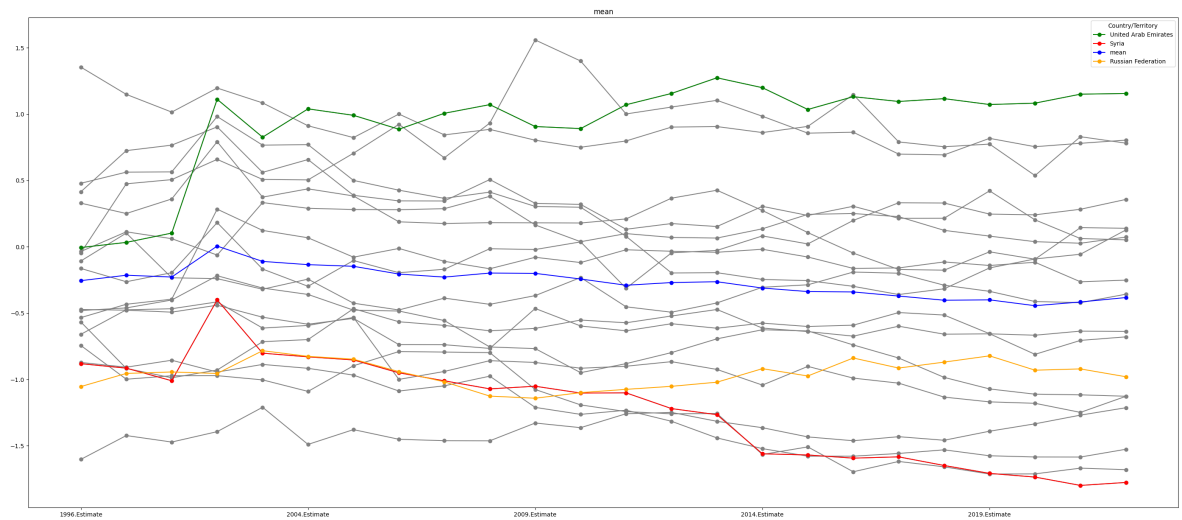
```
In [ ]: # Оставляем только столбцы Estimate
df_wgi_mena_estimate = df_wgi_mena.filter(regex='Estimate')
# Считаем среднее по региону за каждый год
mean = df_wgi_mena_estimate.mean()
mean.name="mean"
mean
```

```
Out[ ]: 1996.Estimate    -0.255155
        1998.Estimate    -0.214392
        2000.Estimate    -0.227920
        2002.Estimate     0.005365
        2003.Estimate    -0.110138
        2004.Estimate    -0.134619
        2005.Estimate    -0.146972
        2006.Estimate    -0.205434
        2007.Estimate    -0.228394
        2008.Estimate    -0.197921
        2009.Estimate    -0.200338
        2010.Estimate    -0.242790
        2011.Estimate    -0.289805
        2012.Estimate    -0.269579
        2013.Estimate    -0.263348
        2014.Estimate    -0.310855
        2015.Estimate    -0.337412
        2016.Estimate    -0.340242
        2017.Estimate    -0.371198
        2018.Estimate    -0.402726
        2019.Estimate    -0.399918
        2020.Estimate    -0.443916
        2021.Estimate    -0.416095
        2022.Estimate    -0.382464
        Name: mean, dtype: float64
```

1.9 Постройте графики индекса WGI за 1996-2022 для стран своего региона и выделите страны с наибольшим и наименьшим значением WGI за 2022 год, а также отобразите среднее значение по региону и РФ.

```
In [ ]: # Графики по всем странам
ax = df_plot_mena.T.plot(figsize=(35,15), grid=True, marker='o', title='W
# Графики по странам с наибольшим и наименьшим значениями за 2022 год
df_plot_mena.loc[maxWgi].plot(color='green', marker = 'o', legend=True, a
df_plot_mena.loc[minWgi].plot(color='red', marker = 'o', legend=True, ax
# График по среднему значению
mean.plot(title='mean', marker = 'o', color='blue', legend=True, ax = ax)
# Данные по России
df_wgi_russia = df[df['Code'] == 'RUS']
df_wgi_russia.set_index('Country/Territory', inplace=True)
df_wgi_russia = df_wgi_russia.filter(regex='Estimate')
# График по России
df_wgi_russia.T.plot(color='orange', marker = 'o', legend=True, ax = ax)
```

```
Out[ ]: <Axes: title={'center': 'mean'}>
```

1.11 Определите, как изменилось значение показателя rank с 1996 по 2022

Вариант 2. Americas

```
In [ ]: # Страны региона AME
df_ame = df_region[df_region['Region'] == 'AME']
# WGI индексы для стран MENA
df_wgi_ame = df.merge(df_ame, how='inner', left_on='Code', right_on='Code')
# Создаем индекс, оставляем только поля Rank за 1996 и 2022
df_wgi_ame.set_index('Country', inplace=True)
df_wgi_ame = df_wgi_ame.filter(regex='1996.Rank|2022.Rank')
# Данные Rank по России
df_wgi_russia = df[df['Code'] == 'RUS']
df_wgi_russia.set_index('Country/Territory', inplace=True)
df_wgi_russia.rename_axis('Country', inplace=True)
df_wgi_russia = df_wgi_russia.filter(regex='1996.Rank|2022.Rank')
# Объединяем два датафрейма
df_wgi_ame_rus = pd.concat([df_wgi_ame, df_wgi_russia])
# Вывод промежуточного результата
df_wgi_ame_rus
```

Out[]:

	1996.Rank	2022.Rank
Country		
Argentina	53.763439	36.320755
Bahamas	83.870964	84.433960
Bolivia	25.268818	20.754717
Brazil	56.989246	32.075470
Barbados	90.860214	89.150940
Canada	96.236557	93.396225
Chile	90.322578	80.660378
Colombia	36.559139	41.037735
Costa Rica	75.268814	66.981133
Cuba	63.440861	52.358490
Dominica	80.107529	69.339622
Dominican Republic	41.397850	37.264153
Ecuador	30.107527	29.716982
Grenada	80.107529	67.452827
Guatemala	23.655914	11.320755
Guyana	52.688171	45.283020
Honduras	14.516129	18.867924
Haiti	9.139785	5.188679
Jamaica	61.827957	54.245281
Saint Lucia	NaN	70.754715
Mexico	36.021507	17.452829
Nicaragua	33.870968	7.547170
Panama	50.537636	28.773584
Peru	41.935482	22.169811
Paraguay	10.215054	15.094339
El Salvador	21.505377	27.830189
Suriname	61.290321	39.150944
Trinidad and Tobago	80.645164	40.566036
Uruguay	82.258064	91.981133
United States of America	91.397850	82.547173
Saint Vincent and the Grenadines	NaN	77.358490
Venezuela	22.580645	1.886792
Russian Federation	15.053763	19.339622

```
In [ ]: # Вычисляем изменение 2022 года относительно 1996
df_changes = df_wgi_ame_rus.pct_change(fill_method=None, axis=1)
# Умножаем на 100 для получения процентов и добавляем к исходным данным
```

```
df_wgi_changes = df_wgi_ame_rus.copy()  
df_wgi_changes['Change procent'] = (df_changes['2022.Rank'] * 100).round()  
df_wgi_changes
```

```
/tmp/ipykernel_36574/3028844007.py:2: FutureWarning: The 'fill_method' and  
'limit' keywords in DataFrame.pct_change are deprecated and will be remove  
d in a future version. Call ffill before calling pct_change instead.  
df_changes = df_wgi_ame_rus.pct_change(fill_method=None, axis=1)
```

Out[]:

	1996.Rank	2022.Rank	Change procent
Country			
Argentina	53.763439	36.320755	-32.44
Bahamas	83.870964	84.433960	0.67
Bolivia	25.268818	20.754717	-17.86
Brazil	56.989246	32.075470	-43.72
Barbados	90.860214	89.150940	-1.88
Canada	96.236557	93.396225	-2.95
Chile	90.322578	80.660378	-10.70
Colombia	36.559139	41.037735	12.25
Costa Rica	75.268814	66.981133	-11.01
Cuba	63.440861	52.358490	-17.47
Dominica	80.107529	69.339622	-13.44
Dominican Republic	41.397850	37.264153	-9.99
Ecuador	30.107527	29.716982	-1.30
Grenada	80.107529	67.452827	-15.80
Guatemala	23.655914	11.320755	-52.14
Guyana	52.688171	45.283020	-14.05
Honduras	14.516129	18.867924	29.98
Haiti	9.139785	5.188679	-43.23
Jamaica	61.827957	54.245281	-12.26
Saint Lucia	NaN	70.754715	NaN
Mexico	36.021507	17.452829	-51.55
Nicaragua	33.870968	7.547170	-77.72
Panama	50.537636	28.773584	-43.07
Peru	41.935482	22.169811	-47.13
Paraguay	10.215054	15.094339	47.77
El Salvador	21.505377	27.830189	29.41
Suriname	61.290321	39.150944	-36.12
Trinidad and Tobago	80.645164	40.566036	-49.70
Uruguay	82.258064	91.981133	11.82
United States of America	91.397850	82.547173	-9.68
Saint Vincent and the Grenadines	NaN	77.358490	NaN
Venezuela	22.580645	1.886792	-91.64
Russian Federation	15.053763	19.339622	28.47

1.12 Выведите таблицу для Вашего варианта (WGI - rank)

```

In [ ]: # Создаем новый датафрейм для заполнения
rows = ['mean_2022', 'max_2022', 'min_2022', 'Russia_2022'] # Список строк
cols = ['Регион', 'Страна', 'WGI 1996', 'WGI 2022', 'Изменение'] # Список столбцов
table = pd.DataFrame(index=rows, columns=cols)
# Заполняем первый столбец (Регион)
table.loc['mean_2022', 'Регион'] = "AME"
table.loc['max_2022', 'Регион'] = "AME"
table.loc['min_2022', 'Регион'] = "AME"
table.loc['Russia_2022', 'Регион'] = "ECA"
# Заполняем второй столбец (Страна)
minAmeIdx = df_wgi_ame['2022.Rank'].idxmin()
maxAmeIdx = df_wgi_ame['2022.Rank'].idxmax()
rusIdx = 'Russian Federation'
table.loc['mean_2022', 'Страна'] = "-"
table.loc['max_2022', 'Страна'] = maxAmeIdx
table.loc['min_2022', 'Страна'] = minAmeIdx
table.loc['Russia_2022', 'Страна'] = rusIdx
# Заполняем третий столбец (WGI 1996)
table.loc['mean_2022', 'WGI 1996'] = df_wgi_ame['1996.Rank'].mean()
table.loc['max_2022', 'WGI 1996'] = df_wgi_ame.loc[maxAmeIdx, '1996.Rank']
table.loc['min_2022', 'WGI 1996'] = df_wgi_ame.loc[minAmeIdx, '1996.Rank']
table.loc['Russia_2022', 'WGI 1996'] = df_wgi_ame_rus.loc[rusIdx, '1996.Rank']
# Заполняем четвертый столбец (WGI 2022)
table.loc['mean_2022', 'WGI 2022'] = df_wgi_ame['2022.Rank'].mean()
table.loc['max_2022', 'WGI 2022'] = df_wgi_ame.loc[maxAmeIdx, '2022.Rank']
table.loc['min_2022', 'WGI 2022'] = df_wgi_ame.loc[minAmeIdx, '2022.Rank']
table.loc['Russia_2022', 'WGI 2022'] = df_wgi_ame_rus.loc[rusIdx, '2022.Rank']
# Заполняем пятый столбец (Изменение)
# Для расчета среднего изменения по региону из таблицы изменений удаляем
df_wgi_changes_ame = df_wgi_changes.drop(rusIdx)
table.loc['mean_2022', 'Изменение'] = df_wgi_changes_ame['Change procent'].mean()
table.loc['max_2022', 'Изменение'] = df_wgi_changes_ame.loc[maxAmeIdx, 'Change procent']
table.loc['min_2022', 'Изменение'] = df_wgi_changes_ame.loc[minAmeIdx, 'Change procent']
table.loc['Russia_2022', 'Изменение'] = df_wgi_changes.loc[rusIdx, 'Change procent']
table

```

```

Out[ ]:

```

	Регион	Страна	WGI 1996	WGI 2022	Изменение
mean_2022	AME	-	53.27957	45.59257	-19.16
max_2022	AME	Canada	96.236557	93.396225	-2.95
min_2022	AME	Venezuela	22.580645	1.886792	-91.64
Russia_2022	ECA	Russian Federation	15.053763	19.339622	28.47

1.13 Отобразите диаграмму размаха индекса WGI за 2022 для всех стран и для каждого региона в отдельности (на одном графике)

```

In [ ]: # Создаем датафрейм со всеми странами и объединяем с датафреймом с названиями регионов
df_merged = df.merge(df_region, how='inner', left_on='Code', right_on='Country')
# Оставляем только индекс Country, код региона и WGI за 2022 год
df_merged = df_merged.filter(regex='^Country$|2022.Estimate|Region')
df_merged.set_index('Country', inplace=True)
# Данные по каждому региону. Переименовываем столбец Estimate для отображения
df_boxplot_ame = df_merged[df_merged['Region'] == 'AME'].rename(columns={'2022.Estimate': 'WGI_2022'})
df_boxplot_eca = df_merged[df_merged['Region'] == 'ECA'].rename(columns={'2022.Estimate': 'WGI_2022'})

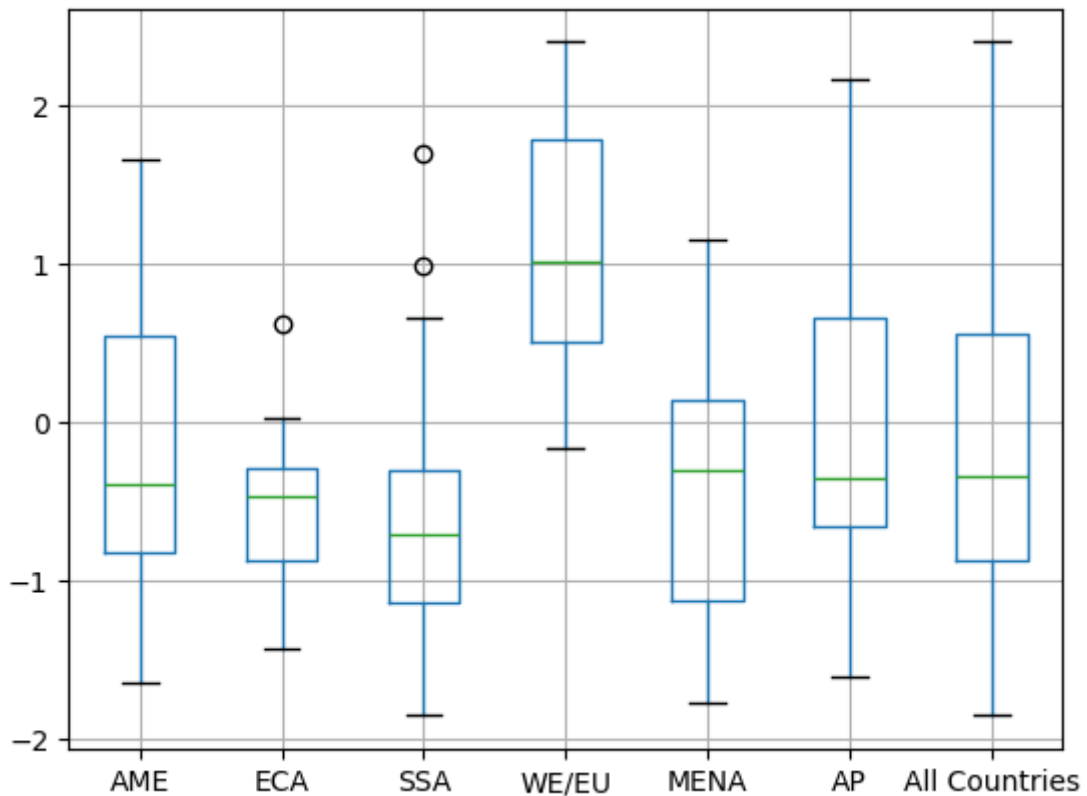
```

```

df_boxplot_ssa = df_merged[df_merged['Region'] == 'SSA'].rename(columns={
df_boxplot_eu = df_merged[df_merged['Region'] == 'WE/EU'].rename(columns={
df_boxplot_mena = df_merged[df_merged['Region'] == 'MENA'].rename(columns={
df_boxplot_ap = df_merged[df_merged['Region'] == 'AP'].rename(columns={'2
# Данные по всем странам
df_boxplot_all = df_merged.rename(columns={'2022.Estimate':'All Countries
# Объединяем в общий датафрейм
df_boxplot = pd.concat([df_boxplot_ame, df_boxplot_eca, df_boxplot_ssa, d
# Строим график
df_boxplot.boxplot()

```

Out[]: <Axes: >



Задание 2. Анализ рынка акций

2.1 Загрузите данные в один dataframe из всех файлов в папке /data/stock. Все файлы имеют одинаковую структуру, в том числе наименование столбцов. В качестве значений индекса dataframe'а необходимо указать значения столбца "Date". Название столбцов должны соответствовать названию акций (имя файла без .csv), а их значения - значениям цены закрытия (столбец "Close" в файлах .csv)

```

In [ ]: import glob
import pandas as pd

```

```

In [ ]: # Получение списка csv
files = glob.glob('./data/stock/*.csv')
# Загрузка данных по каждой компании в один датафрейм
df = pd.DataFrame()

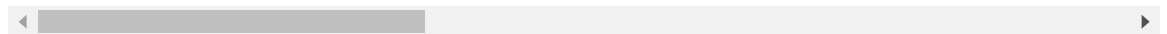
```

```
for file in files:
    data = pd.read_csv(file, index_col='Date', usecols=['Date', 'Close'])
    compName = file.split('/')[-1].split('.')[0]
    df[compName] = data['Close']
df.head()
```

```
Out[ ]:
```

	TWLO	CSCO	DBX	AMZN	AAPL	SPOT	NV
Date							
2022-01-01	206.119995	55.669998	24.750000	149.573502	174.779999	196.259995	244.860
2022-02-01	174.800003	55.770000	22.690001	153.563004	165.119995	156.190002	243.850
2022-03-01	164.809998	55.759998	23.250000	162.997498	174.610001	151.020004	272.850
2022-04-01	111.820000	48.980000	21.750000	124.281502	157.649994	101.650002	185.470
2022-05-01	105.169998	45.049999	20.840000	120.209503	148.839996	112.769997	186.720

5 rows × 25 columns



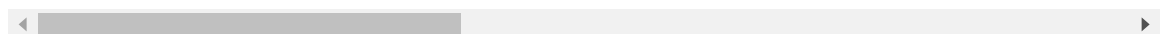
2.2 Рассчитайте корреляционную матрицу для всех акций

```
In [ ]: corr = df.corr()
corr
```

Out[]:

	TWLO	CSCO	DBX	AMZN	AAPL	SPOT	NVDA	
TWLO	1.000000	0.383777	-0.113102	0.314869	0.042914	0.059969	-0.244797	-0.7
CSCO	0.383777	1.000000	0.496982	0.404820	0.589552	0.424007	0.320159	0.4
DBX	-0.113102	0.496982	1.000000	0.478171	0.740429	0.525305	0.519374	0.6
AMZN	0.314869	0.404820	0.478171	1.000000	0.665715	0.875779	0.765294	0.7
AAPL	0.042914	0.589552	0.740429	0.665715	1.000000	0.687415	0.633114	0.7
SPOT	0.059969	0.424007	0.525305	0.875779	0.687415	1.000000	0.925270	0.9
NVDA	-0.244797	0.320159	0.519374	0.765294	0.633114	0.925270	1.000000	0.9
NFLX	-0.102302	0.497727	0.635239	0.735466	0.701937	0.920771	0.910910	1.0
SHOP	0.657843	-0.144612	0.424923	0.824934	0.465147	0.737909	0.713391	0.8
ABNB	0.429915	0.594365	0.332740	0.830690	0.617430	0.753797	0.649664	0.6
PINS	-0.141953	0.384233	0.710191	0.666996	0.640294	0.842858	0.815629	0.9
HPQ	0.728572	0.214262	-0.177013	0.235247	0.067074	0.005774	-0.160502	-0.2
XIACY	0.447846	0.474311	0.382992	0.654564	0.408747	0.647331	0.445645	0.5
MU	0.315313	0.472688	0.440043	0.906932	0.606787	0.902439	0.796707	0.7
META	-0.072886	0.374998	0.552874	0.830910	0.705358	0.973401	0.961389	0.8
MSFT	-0.094023	0.391476	0.648164	0.838702	0.790691	0.949380	0.935386	0.9
TCOM	-0.562073	0.257188	0.423136	0.309545	0.439363	0.640120	0.787859	0.7
ORCL	-0.393536	0.463955	0.667833	0.534556	0.769309	0.763100	0.875089	0.8
INTC	0.585988	0.420854	0.390625	0.816519	0.507251	0.645555	0.458281	0.4
GOOGL	0.315410	0.600025	0.669228	0.912332	0.806847	0.821587	0.715287	0.7
TSLA	0.703872	0.253808	0.037233	0.302321	0.248385	-0.092332	-0.277600	-0.2
UBER	-0.186828	0.326346	0.595928	0.796897	0.661323	0.933308	0.969790	0.9
EBAY	0.753732	0.494938	-0.157363	0.434078	0.115591	0.296858	0.087027	0.7
ADBE	0.067604	0.554172	0.816359	0.819614	0.833129	0.863827	0.802739	0.8
GTLB	0.310273	0.068856	0.402517	0.690644	0.282373	0.540113	0.404702	0.4

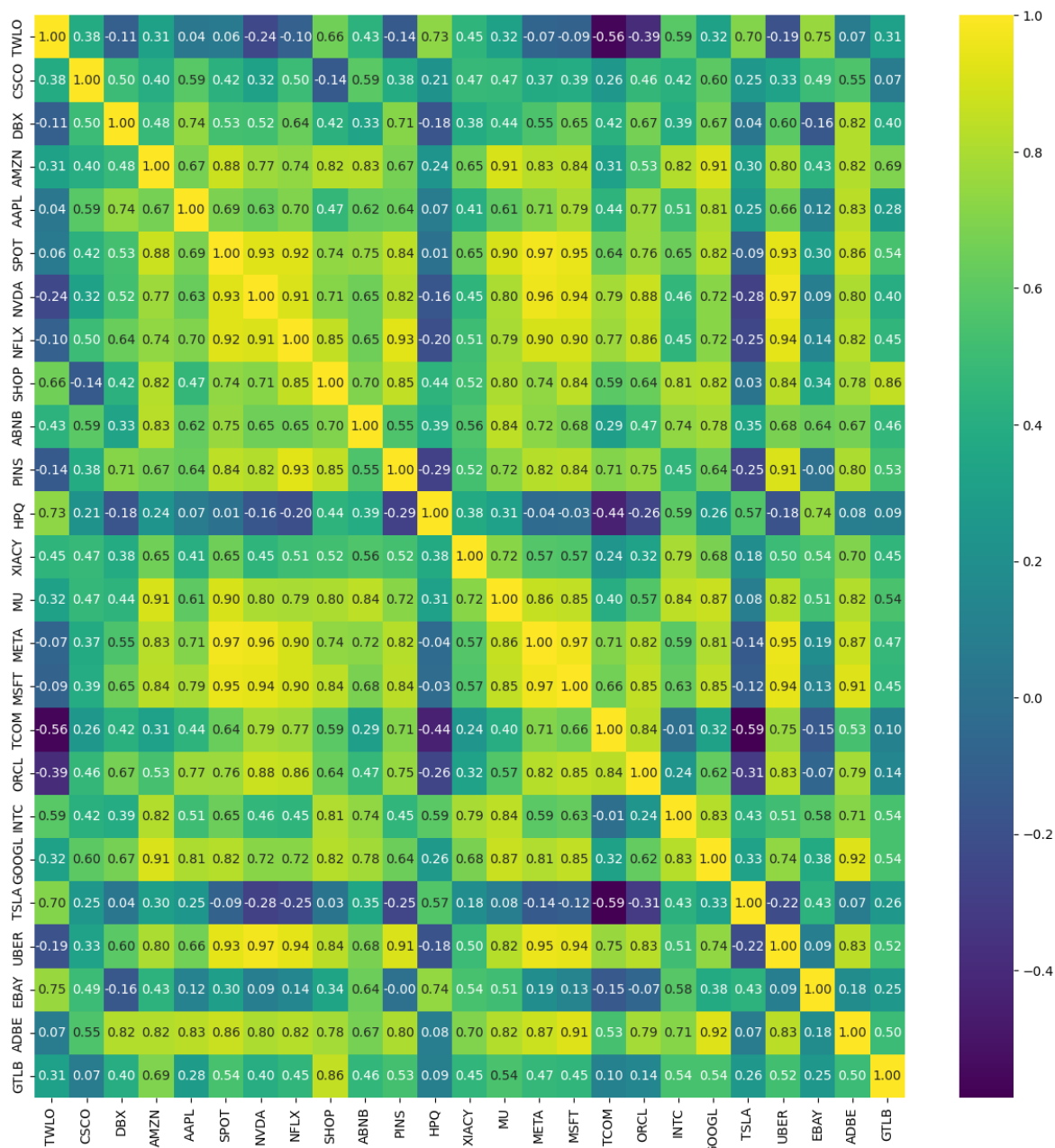
25 rows × 25 columns



2.3 Отобразите корреляционную матрицу в виде диаграммы

```
In [ ]: import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [ ]: plt.figure(figsize=(15,15))
sns.heatmap(corr, annot=True, cmap='viridis', fmt=".2f")
plt.show()
```

2.4 В соответствии с Вашим вариантом определите:

- акцию с максимальной положительной корреляцией (max)
- акцию с максимальной отрицательной корреляцией (min)
- акцию с минимальной корреляцией (которая больше всего соответствует отсутствию какой-либо корреляции (none))

Вариант 4. Uber (UBER)

```
In [ ]: # Корреляция акций UBER с другими акциями
uber_corr = corr['UBER']
uber_corr = uber_corr.drop('UBER')
uber_corr
```

```
Out[ ]: TWLO      -0.186828
        CSCO      0.326346
        DBX       0.595928
        AMZN      0.796897
        AAPL      0.661323
        SPOT      0.933308
        NVDA      0.969790
        NFLX      0.937042
        SHOP      0.836565
        ABNB      0.680764
        PINS      0.907751
        HPQ       -0.180970
        XIACY     0.495835
        MU        0.820809
        META      0.954444
        MSFT      0.939538
        TCOM      0.754442
        ORCL      0.832075
        INTC      0.512572
        GOOGL     0.737311
        TSLA      -0.221155
        EBAY      0.085736
        ADBE      0.834611
        GTLB      0.521399
        Name: UBER, dtype: float64
```

```
In [ ]: # Акция с максимальной положительная корреляция
max_corr = uber_corr.idxmax()
max_corr
```

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

```
In [ ]: # Акция с максимальной отрицательная корреляция
min_corr = uber_corr.idxmin()
min_corr
```

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

```
In [ ]: # Акция с минимальной корреляцией
none_corr = uber_corr.abs().idxmin()
none_corr
```

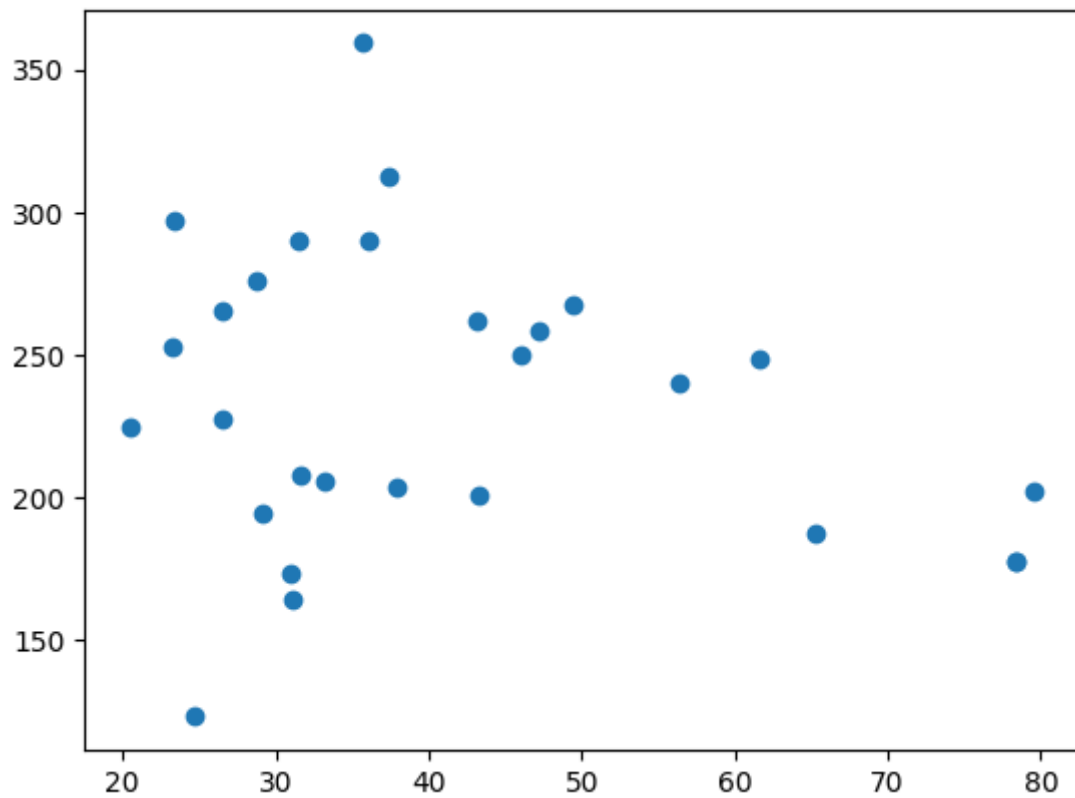
```
Out[ ]: 'EBAY'
```

2.5 Постройте диаграммы разброса

- Ваша компания - Компания с min
- Ваша компания - Компания с max
- Ваша компания - Компания с none

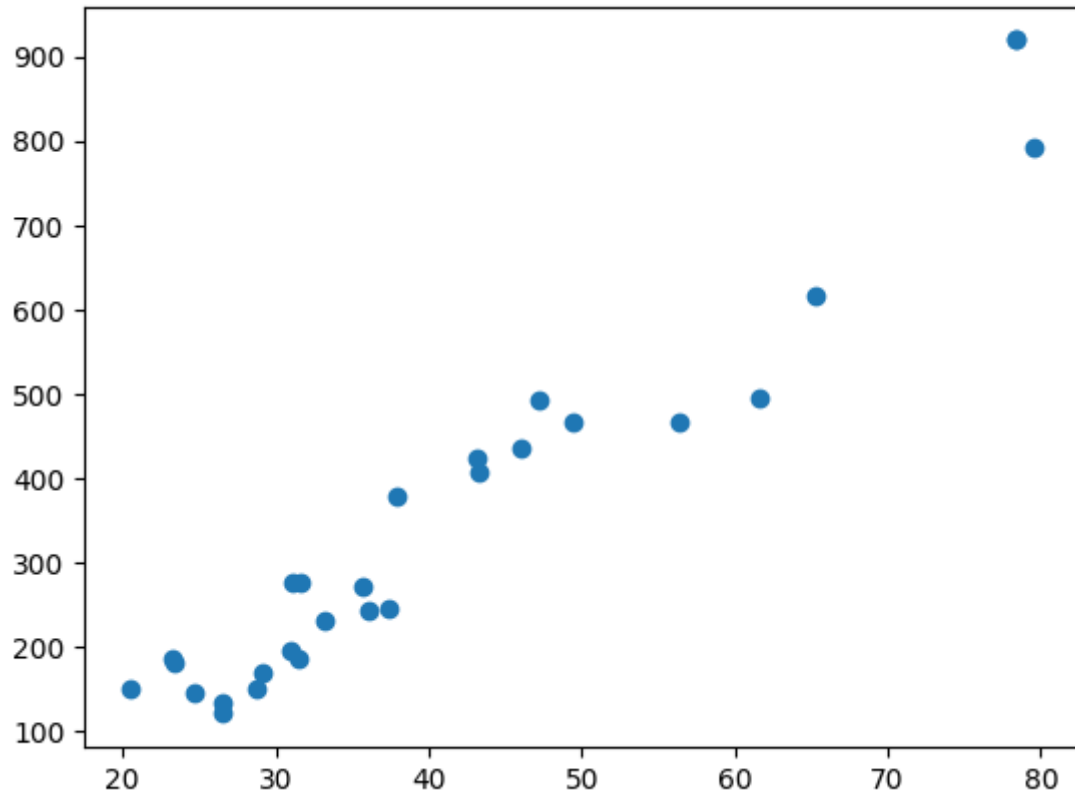
```
In [ ]: # Диаграмма разброса UBER - min corr
plt.scatter(df['UBER'], df[min_corr])
```

```
Out[ ]: <matplotlib.collections.PathCollection at 0x7bfe68a11db0>
```



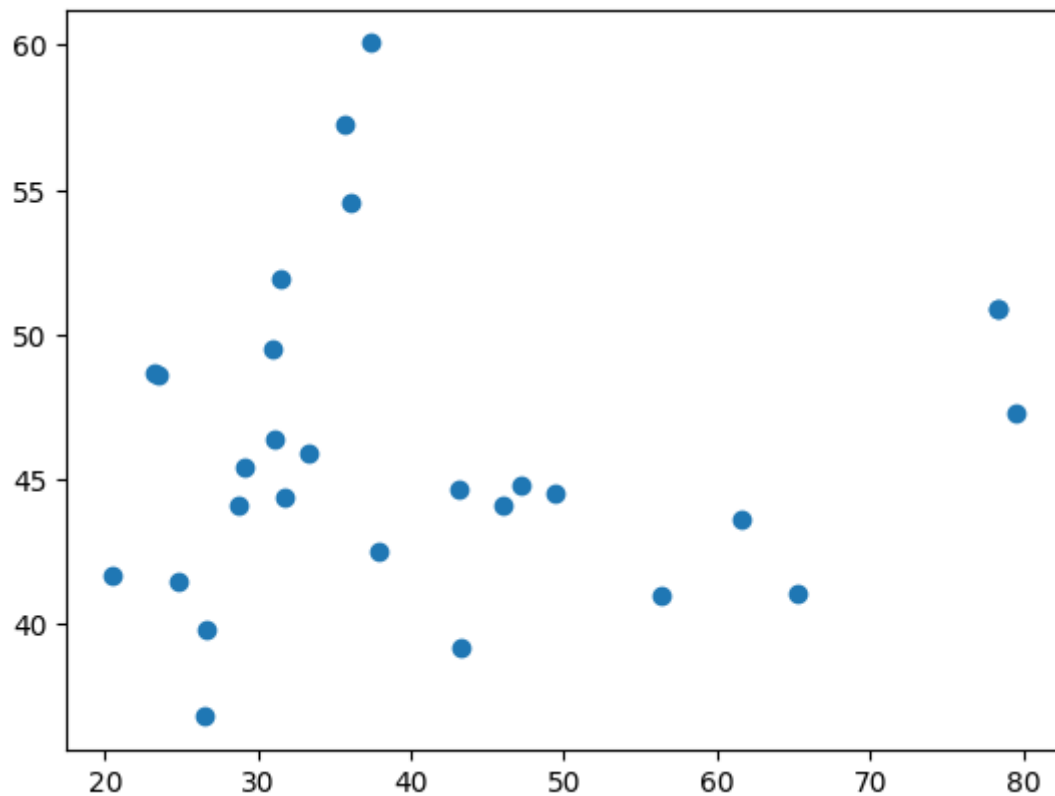
```
In [ ]: # Диаграмма разброса UBER - max corr
plt.scatter(df['UBER'], df[max_corr])
```

```
Out[ ]: <matplotlib.collections.PathCollection at 0x7bfe688a9510>
```



```
In [ ]: # Диаграмма разброса UBER - none corr
plt.scatter(df['UBER'], df[none_corr])
```

```
Out[ ]: <matplotlib.collections.PathCollection at 0x7bfe689261a0>
```



2.6 Рассчитайте среднюю цену акций для каждого месяца (исходные данные взяты с интервалом в месяц)

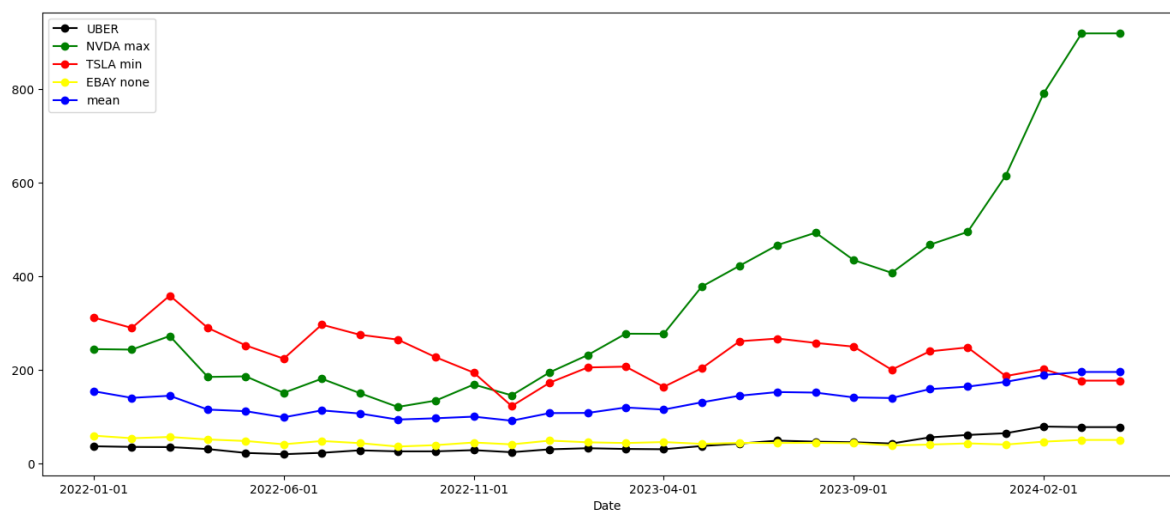
```
In [ ]: # Транспонируем исходный датафрейм, чтобы каждый месяц был столбцом, по к
mean = df.T.mean()
mean.name='mean'
mean
```

```
Out[ ]: Date
2022-01-01    154.857167
2022-02-01    140.774723
2022-03-01    145.272287
2022-04-01    115.763514
2022-05-01    112.316034
2022-06-01     99.256929
2022-07-01    114.014999
2022-08-01    107.380833
2022-09-01     94.437083
2022-10-01     97.227501
2022-11-01    100.671666
2022-12-01     92.028958
2023-01-01    108.279540
2023-02-01    108.613126
2023-03-01    120.210832
2023-04-01    115.778799
2023-05-01    131.258401
2023-06-01    145.426799
2023-07-01    153.207200
2023-08-01    152.016000
2023-09-01    141.760400
2023-10-01    140.454598
2023-11-01    159.367601
2023-12-01    164.859599
2024-01-01    174.886801
2024-02-01    189.609962
2024-03-01    196.083201
2024-03-12    196.083201
Name: mean, dtype: float64
```

2.7 Постройте графики для акций из пункта 4 и средней из пункта 6

```
In [ ]: df['UBER'].plot(figsize=(17,7), marker='o', color='black', legend=True)
df[max_corr].plot(label=max_corr + ' max', marker='o', color='green', leg
df[min_corr].plot(label=min_corr + ' min', marker='o', color='red', leger
df[none_corr].plot(label=none_corr + ' none', marker='o', color='yellow',
mean.plot(marker='o', color='blue', legend=True)
```

```
Out[ ]: <Axes: xlabel='Date'>
```



Вывод

В данном домашнее задании были отработаны с использованием языка Python такие навыки и умения, как:

- Решение практических задач по анализу данных
- Загрузка, трансформация, вычисление простых статистик
- Визуализация данных в виде графиков и диаграмм