ДОМАШНЕЕ ЗАДАНИЕ 1. Дескриптивный анализ данных

Астахов С.В. 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.

Содержание

Вариант

In [1]:
```
surname = "Астахов"  # 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:  1
Задача № 1, шаг 11 - вариант:  1
задача № 2 - вариант:  1
```

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

1.1 Загрузите данные в DataFrame

In [2]:
```
# импорт библиотек
import matplotlib.pyplot as plt
%matplotlib inline
import pandas as pd
```

In [3]:
```
# Чтение в датафрейм с 14 строки
df = pd.read_excel("./wgidataset.xlsx", sheet_name = "ControlofCorruption
```

```
df.head() # вывод первых 5 записей
```

Out[3]:

	Unnamed: 0_level_0	Unnamed: 1_level_0						19
	Country/ Territory	Code	Estimate	StdErr	NumSrc	Rank	Lower	Up
0	Aruba	ABW	NaN	NaN	NaN	NaN	NaN	N
1	Andorra	ADO	1.318143	0.480889	1.0	87.096771	72.043015	96.774
2	Afghanistan	AFG	-1.291705	0.340507	2.0	4.301075	0.000000	27.419
3	Angola	AGO	-1.167702	0.262077	4.0	9.677420	0.537634	27.419
4	Anguilla	AIA	NaN	NaN	NaN	NaN	NaN	N

5 rows × 146 columns

In [4]:
```
# Избавление от мультииндекса и переименование колонок
df = df.rename(columns={"Unnamed: 0_level_0": "C/T", "Unnamed: 1_level_0"
new_cols = map(lambda x: x[1]+"."+str(x[0]), list(df.columns.values)) # c
df = df.droplevel(0, axis=1) # удаляем первую строку колоночного индекса
df.columns = new_cols
df = df.rename(columns={"Country/Territory.C/T": "Country/Territory", "Co
df.head()
```

Out[4]:

	Country/ Territory	Code	Estimate.1996	StdErr.1996	NumSrc.1996	Rank.1996	Lower.199
0	Aruba	ABW	NaN	NaN	NaN	NaN	NaN
1	Andorra	ADO	1.318143	0.480889	1.0	87.096771	72.04301
2	Afghanistan	AFG	-1.291705	0.340507	2.0	4.301075	0.00000
3	Angola	AGO	-1.167702	0.262077	4.0	9.677420	0.53763
4	Anguilla	AIA	NaN	NaN	NaN	NaN	NaN

5 rows × 146 columns

1.2 Отсортируйте данные по убыванию индекса DataFrame

In [5]:
```
df_sort_desc = df.iloc[::-1] #сортировка по убыванию индекса
df_sort_desc #вывод
```

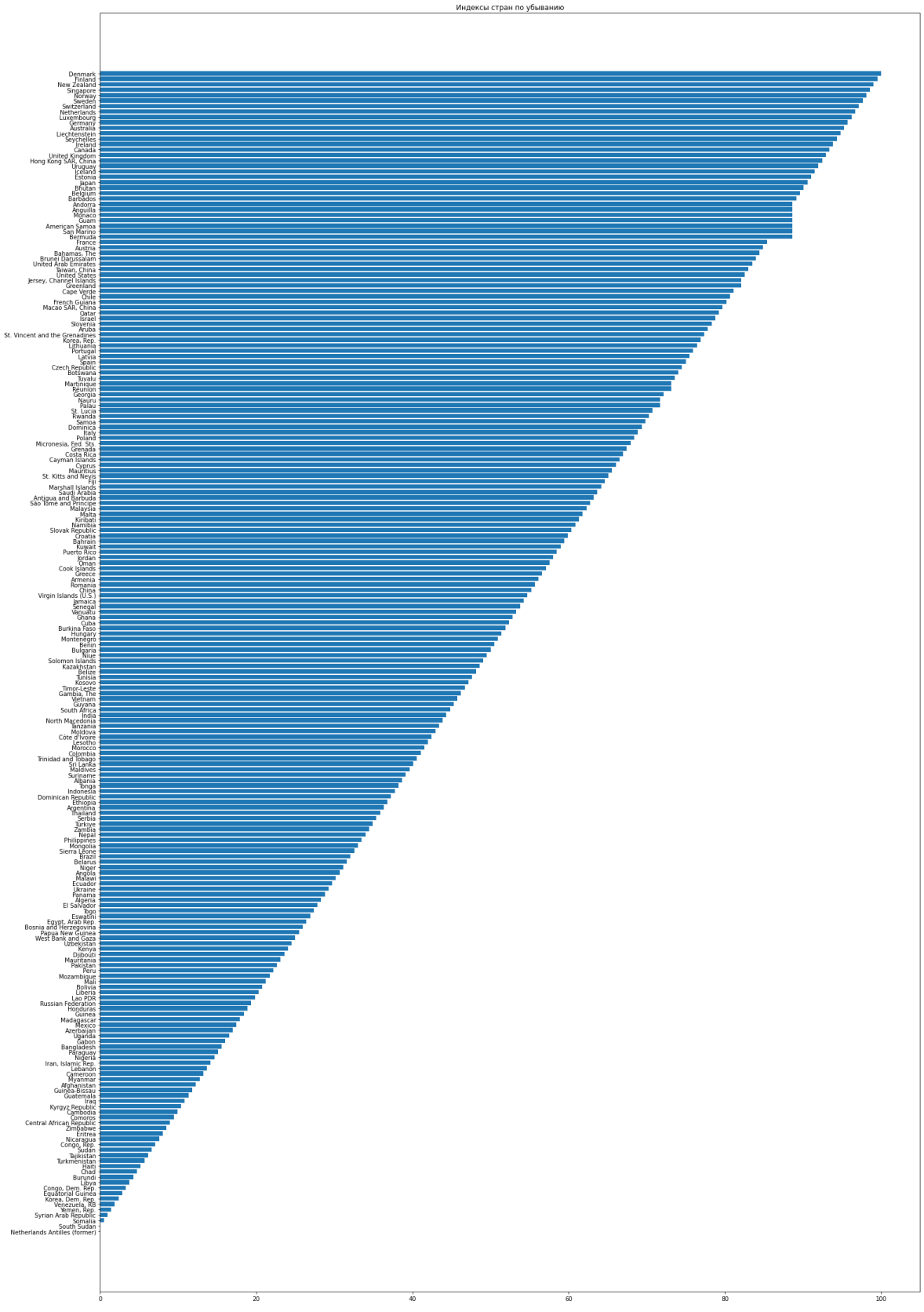
Out[5]:

	Country/ Territory	Code	Estimate.1996	StdErr.1996	NumSrc.1996	Rank.1996	Lower.19
213	Zimbabwe	ZWE	-0.278847	0.244907	5.0	47.849461	30.645
212	Zambia	ZMB	-0.840641	0.262077	4.0	24.731182	5.913
211	Congo, Dem. Rep.	ZAR	-1.647852	0.315914	3.0	0.000000	0.000
210	South Africa	ZAF	0.732927	0.210325	6.0	76.344086	66.129
209	Serbia	SRB	-1.140072	0.262077	4.0	11.827957	0.537
...	...	...	...	...	...	...	
4	Anguilla	AIA	NaN	NaN	NaN	NaN	N
3	Angola	AGO	-1.167702	0.262077	4.0	9.677420	0.537
2	Afghanistan	AFG	-1.291705	0.340507	2.0	4.301075	0.000
1	Andorra	ADO	1.318143	0.480889	1.0	87.096771	72.043
0	Aruba	ABW	NaN	NaN	NaN	NaN	N

214 rows × 146 columns

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

In [6]:
```
df_sort_desc_fillna = df_sort_desc.copy() # 0 вместо NaN
df_sort_desc_fillna["Rank.2022"] = df_sort_desc_fillna["Rank.2022"].filln
#сортировка данных по столбцу Rank за 2022-ый год
dfSort = df_sort_desc_fillna.sort_values("Rank.2022", ascending=True)
plt.figure(figsize=(25, 40)) #размер графика
plt.title('Индексы стран по убыванию')
plt.barh(dfSort["Country/Territory"], dfSort['Rank.2022']) #формирование
plt.show() #Отображение
```



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

In [7]:
```
# Чтение файла с регионами в датафрейм
df_reg = pd.read_excel("./regions.xlsx")
df_reg.head(10)
```

Out[7]:

	Country	Code	Region
0	Afghanistan	AFG	AP
1	Albania	ALB	ECA
2	Algeria	DZA	MENA
3	Angola	AGO	SSA
4	Argentina	ARG	AME
5	Armenia	ARM	ECA
6	Australia	AUS	AP
7	Austria	AUT	WE/EU
8	Azerbaijan	AZE	ECA
9	Bahamas	BHS	AME

In [8]:
```
# Join df с регионами и со странами
merged_df = pd.merge(df_reg, df, on='Code')
sort_for_reg = merged_df.groupby("Region")
sort_for_reg
#создание датафрейма по региону AP
df_region = sort_for_reg.get_group("AP")
df_region = df_region.drop('Country/Territory', axis=1, inplace=False) # ,
df_region = df_region.set_index("Country") # выставление столбца Country
```

1.5 Выведите данные DataFrame'a

In [9]:
```
df_region
```

Out[9]:

Country	Code	Region	Estimate.1996	StdErr.1996	NumSrc.1996	Rank.1996	Low
Afghanistan	AFG	AP	-1.291705	0.340507	2.0	4.301075	0.
Australia	AUS	AP	1.877356	0.210325	6.0	93.548386	90.
Bangladesh	BGD	AP	-0.969682	0.262077	4.0	17.741936	2.
Bhutan	BTN	AP	0.942838	0.340507	2.0	81.182793	66.
Cambodia	KHM	AP	-1.019842	0.275614	3.0	16.129032	2.
China	CHN	AP	-0.271190	0.188622	7.0	48.387096	32.
Fiji	FJI	AP	0.659303	0.340507	2.0	73.655914	59.
Hong Kong	HKG	AP	1.444894	0.204951	6.0	89.784943	81.
India	IND	AP	-0.381090	0.188622	7.0	43.010754	29.
Indonesia	IDN	AP	-0.864106	0.188622	7.0	22.043011	8.
Japan	JPN	AP	1.192312	0.188622	7.0	84.408600	80.
Korea, North	PRK	AP	-1.284347	0.315914	3.0	4.838710	0.
Korea, South	KOR	AP	0.382197	0.188622	7.0	65.591400	59.
Laos	LAO	AP	-0.722834	0.340507	2.0	28.494623	4.
Malaysia	MYS	AP	0.383065	0.188622	7.0	66.129036	59.
Maldives	MDV	AP	-0.322941	0.340507	2.0	46.774193	19.
Mongolia	MNG	AP	0.111758	0.315914	3.0	60.752689	41.
Myanmar	MMR	AP	-1.500767	0.262077	4.0	1.612903	0.
Nepal	NPL	AP	-0.639209	0.340507	2.0	31.720430	7.
New Zealand	NZL	AP	2.110246	0.210325	6.0	97.849464	93.
Pakistan	PAK	AP	-1.220030	0.262077	4.0	7.526882	0.
Papua New Guinea	PNG	AP	-0.433467	0.262077	4.0	40.322582	22.
Philippines	PHL	AP	-0.358872	0.188622	7.0	45.698925	30.
Singapore	SGP	AP	2.107434	0.188622	7.0	97.311829	93.
Solomon Islands	SLB	AP	0.340782	0.439480	1.0	65.053764	43.
Sri Lanka	LKA	AP	-0.056539	0.262077	4.0	54.301075	37.
Taiwan	TWN	AP	0.580821	0.188622	7.0	73.118279	62.
Thailand	THA	AP	-0.361192	0.188622	7.0	45.161289	30.
Vanuatu	VUT	AP	0.216309	0.439480	1.0	62.365593	36.
Vietnam	VNM	AP	-0.489799	0.212363	6.0	37.096775	24.

30 rows × 146 columns

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

In [10]:
```
WGI = df_region.filter(regex='Estimate|Country')#Фильтрация по Estimate|C
# WGI.columns = map(lambda x: x.split(".")[1], list(WGI.columns.values)) 
WGI.head()
```

Out[10]:

Country	Estimate.1996	Estimate.1998	Estimate.2000	Estimate.2002	Estimate.200
Afghanistan	-1.291705	-1.176012	-1.271724	-1.251137	-1.34418
Australia	1.877356	1.798130	1.862088	1.761436	1.89528
Bangladesh	-0.969682	-0.773011	-1.212083	-1.449087	-1.54172
Bhutan	0.942838	0.883641	0.574340	0.449922	1.08701
Cambodia	-1.019842	-0.988312	-0.967183	-0.990784	-0.98983

5 rows × 24 columns

In [11]:
```
WGI = WGI.T #транспонирование
WGI.head()
```

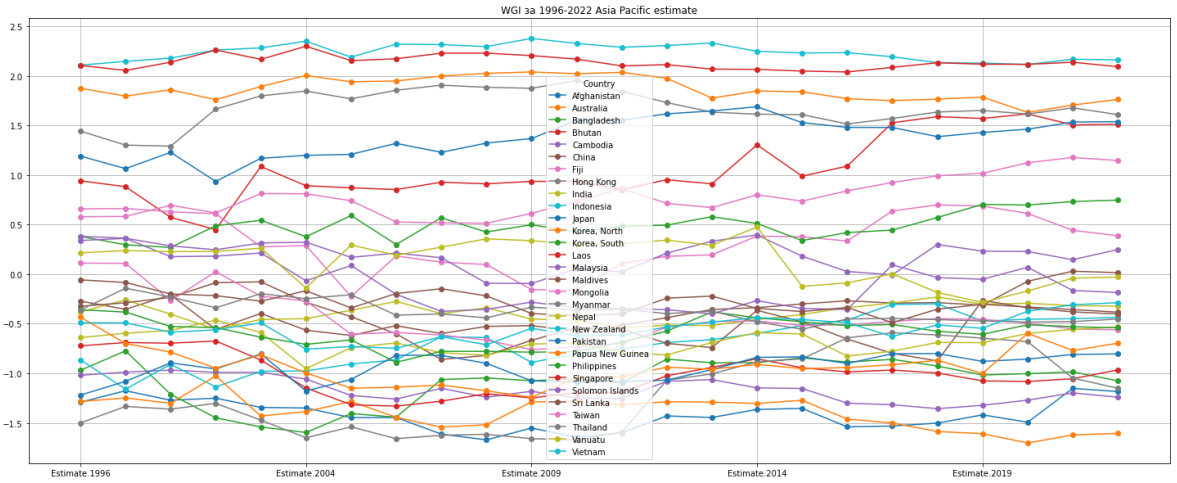
Out[11]:

Country	Afghanistan	Australia	Bangladesh	Bhutan	Cambodia	China	
Estimate.1996	-1.291705	1.877356	-0.969682	0.942838	-1.019842	-0.271190	0.6
Estimate.1998	-1.176012	1.798130	-0.773011	0.883641	-0.988312	-0.353955	0.6
Estimate.2000	-1.271724	1.862088	-1.212083	0.574340	-0.967183	-0.208549	0.6
Estimate.2002	-1.251137	1.761436	-1.449087	0.449922	-0.990784	-0.557898	0.6
Estimate.2003	-1.344180	1.895287	-1.541721	1.087011	-0.989836	-0.395265	0.2

5 rows × 30 columns

In [12]:
```
WGI.plot(grid=1,
         figsize=(25,10),
         title='WGI за 1996-2022 Asia Pacific estimate',
         marker='o',
         legend = True)
```

Out[12]:
```
<AxesSubplot:title={'center':'WGI за 1996-2022 Asia Pacific estimate'}>
```



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

In [13]:
```
WGI = WGI.T
#вывод наименьшего
min_W = WGI["Estimate.2022"].idxmin()
min_W
```

Out[13]: 'Korea, North'

In [14]:
```
#вывод наибольшего
max_W = WGI["Estimate.2022"].idxmax()
max_W
```

Out[14]: 'New Zealand'

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

In [15]:
```
#вывод среднего значения за каждый период
mean = WGI.mean()
mean.name = "mean"
mean
```

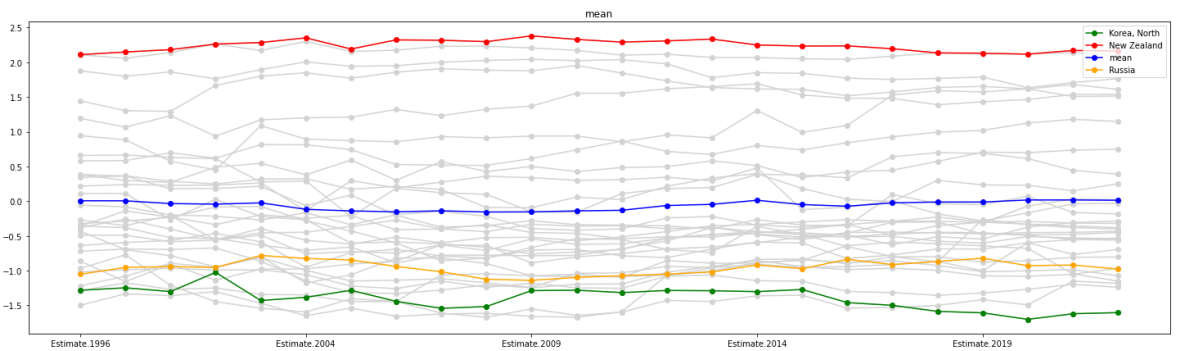
Out[15]:
```
Estimate.1996    0.005390
Estimate.1998    0.004409
Estimate.2000   -0.035114
Estimate.2002   -0.043584
Estimate.2003   -0.024761
Estimate.2004   -0.117873
Estimate.2005   -0.141014
Estimate.2006   -0.154418
Estimate.2007   -0.139037
Estimate.2008   -0.156720
Estimate.2009   -0.154840
Estimate.2010   -0.140798
Estimate.2011   -0.131607
Estimate.2012   -0.063054
Estimate.2013   -0.046824
Estimate.2014    0.012366
Estimate.2015   -0.049325
Estimate.2016   -0.073050
Estimate.2017   -0.025685
Estimate.2018   -0.012606
Estimate.2019   -0.013031
Estimate.2020    0.016584
Estimate.2021    0.016733
Estimate.2022    0.012814
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 [16]:
```
WGI = WGI.T
```

In [17]:
```
#выод графика индекса для всех стран региона АР
WGI.plot(color='lightgrey', grid=1, figsize=(25,7),
         title='WGA за 1996-2021 MENA estimate', marker='o', legend = Fal
WGI[min_W].plot(color='green', marker='o', legend = True)#вывод графика п
WGI[max_W].plot(color='red', marker='o', legend = True)#вывод графика по
mean.plot(title="mean", color='blue', marker='o', legend = True)#вывод гр
# #Вывод графика по России
df_russia = merged_df[merged_df["Country"].str.contains('Russ')].set_inde
df_russia.filter(regex='Estimate').T["Russia"].plot(color='orange',marker
```

Out[17]:
```
<AxesSubplot:title={'center':'mean'}>
```



1.11 Определите, как изменилось значение

показателя rank с 1996 по 2022 (rank)

In [18]:
```
df_rank_change = pd.concat([df_region.filter(regex='Rank'), df_russia.fil
df_rank_change.head()
```

Out[18]:

Country	Rank.1996	Rank.1998	Rank.2000	Rank.2002	Rank.2003	Rank.2004	Ran
Afghanistan	4.301075	8.021390	4.787234	4.761905	4.761905	6.403941	1.4
Australia	93.548386	92.513367	93.617020	92.063492	93.650795	96.551727	95.6
Bangladesh	17.741936	28.877005	6.914894	1.587302	0.529101	0.985222	2.9
Bhutan	81.182793	81.283424	71.276596	70.899467	82.010582	80.788177	79.0
Cambodia	16.129032	19.251337	18.085106	17.460318	14.285714	14.285714	10.2

5 rows × 24 columns

In [19]:
```
# вывод процена изменения относительно 1996-го года и 2022-го
# изменение в долях от единицы
procent_change = df_rank_change.filter(items=['Rank.1996','Rank.2022']).c
procent_change_readable = procent_change.map(lambda x: "{:.2f}%".format(x
procent_change_concat = pd.concat([df_rank_change[['Rank.1996','Rank.2022
procent_change_concat.columns = ['Rank.1996','Rank.2022', "Change.relativ
procent_change_concat
```

Out[19]:

Country	Rank.1996	Rank.2022	Change.relative	Change.percent
Afghanistan	4.301075	12.264151	1.851415	185.14%
Australia	93.548386	95.283020	0.018543	1.85%
Bangladesh	17.741936	15.566038	-0.122641	-12.26%
Bhutan	81.182793	90.094337	0.109771	10.98%
Cambodia	16.129032	9.905661	-0.385849	-38.58%
China	48.387096	55.188679	0.140566	14.06%
Fiji	73.655914	64.622643	-0.122641	-12.26%
Hong Kong	89.784943	92.452827	0.029714	2.97%
India	43.010754	44.339622	0.030896	3.09%
Indonesia	22.043011	37.735847	0.711919	71.19%
Japan	84.408600	90.566040	0.072948	7.29%
Korea, North	4.838710	2.358490	-0.512579	-51.26%
Korea, South	65.591400	76.886795	0.172208	17.22%
Laos	28.494623	19.811321	-0.304735	-30.47%
Malaysia	66.129036	62.264153	-0.058445	-5.84%
Maldives	46.774193	39.622643	-0.152895	-15.29%
Mongolia	60.752689	33.018867	-0.456504	-45.65%
Myanmar	1.612903	12.735849	6.896227	689.62%
Nepal	31.720430	33.962265	0.070675	7.07%
New Zealand	97.849464	99.056602	0.012337	1.23%
Pakistan	7.526882	22.641510	2.008086	200.81%
Papua New Guinea	40.322582	25.471699	-0.368302	-36.83%
Philippines	45.698925	33.490566	-0.267148	-26.71%
Singapore	97.311829	98.584908	0.013082	1.31%
Solomon Islands	65.053764	49.056602	-0.245907	-24.59%
Sri Lanka	54.301075	40.094341	-0.261629	-26.16%
Taiwan	73.118279	83.018867	0.135405	13.54%
Thailand	45.161289	35.849056	-0.206199	-20.62%
Vanuatu	62.365593	53.301888	-0.145332	-14.53%
Vietnam	37.096775	45.754719	0.233388	23.34%
Russia	15.053763	19.339622	0.284703	28.47%

1.12 Выведите таблицу для Вашего варианта (WGI - rank)

In [20]:
```
# вся таблица берется из датафрейма выше по найденным ранее ключам
#создание 0-го столбца таблицы
Rows = ['mean_2022', 'max_2022', 'min_2022', 'Russia_2022']
#создание 0-ой коллонки таблицы
Cols = ['Регион', 'Страна', 'WGI 1996', 'WGI 2022', 'Изменение']
Tabl_proc  = pd.DataFrame(index=Rows,columns=Cols)
```

In [21]:
```
#создание первого столбца таблицы
Tabl_proc.loc['mean_2022', 'Регион'] = "AP" # из вариантов
Tabl_proc.loc['max_2022', 'Регион'] = "AP" # из вариантов
Tabl_proc.loc['min_2022', 'Регион'] = "AP" # из вариантов
# Получаем название региона для России
Tabl_proc.loc['Russia_2022', 'Регион'] = list(merged_df[merged_df["Countr
```

In [22]:
```
#создание второго столбца таблицы
Tabl_proc.loc['mean_2022', 'Страна'] = "-"
Tabl_proc.loc['max_2022', 'Страна'] = max_W
Tabl_proc.loc['min_2022', 'Страна'] = min_W
Tabl_proc.loc['Russia_2022', 'Страна'] = "Russian Federation"
```

In [23]:
```
#создание третьего столбца таблицы
Tabl_proc.loc['mean_2022', 'WGI 1996'] = "{:.2f}".format(procent_change_c
Tabl_proc.loc['max_2022', 'WGI 1996'] = "{:.2f}".format(procent_change_co
Tabl_proc.loc['min_2022', 'WGI 1996'] = "{:.2f}".format(procent_change_co
Tabl_proc.loc['Russia_2022', 'WGI 1996'] = "{:.2f}".format(procent_change
```

In [24]:
```
#создание четвертого столбца таблицы
Tabl_proc.loc['mean_2022', 'WGI 2022'] = "{:.2f}".format(procent_change_c
Tabl_proc.loc['max_2022', 'WGI 2022'] = "{:.2f}".format(procent_change_co
Tabl_proc.loc['min_2022', 'WGI 2022'] = "{:.2f}".format(procent_change_co
Tabl_proc.loc['Russia_2022', 'WGI 2022'] = "{:.2f}".format(procent_change
```

In [25]:
```
#создание пятого столбца таблицы
Tabl_proc.loc['mean_2022', 'Изменение'] = "{:.2f}%".format(procent_change
Tabl_proc.loc['max_2022', 'Изменение'] = procent_change_concat["Change.pe
Tabl_proc.loc['min_2022', 'Изменение'] = procent_change_concat["Change.pe
Tabl_proc.loc['Russia_2022', 'Изменение'] = procent_change_concat["Change
Tabl_proc
```

Out[25]:

	Регион	Страна	WGI 1996	WGI 2022	Изменение
mean_2022	AP	-	49.06	48.20	29.62%
max_2022	AP	New Zealand	97.85	99.06	1.23%
min_2022	AP	Korea, North	4.84	2.36	-51.26%
Russia_2022	ECA	Russian Federation	15.05	19.34	28.47%

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

In [26]:
```
# получаем список регионов
regions = list(merged_df["Region"].drop_duplicates())
regions
```

Out[26]: ['AP', 'ECA', 'MENA', 'SSA', 'AME', 'WE/EU']

In [27]:
```
merged_df.head()
```

Out[27]:

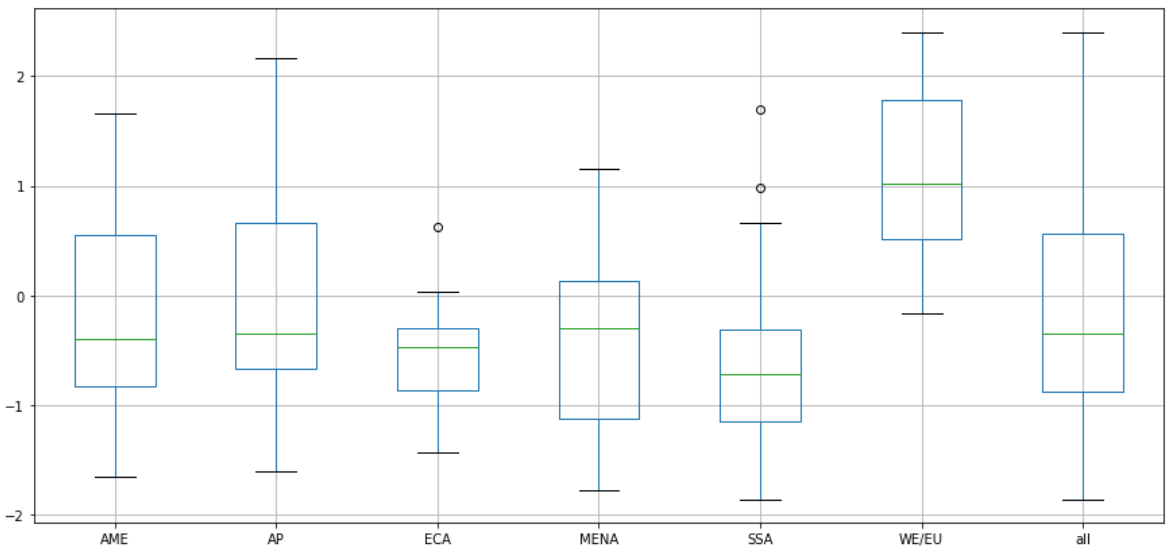
	Country	Code	Region	Country/ Territory	Estimate.1996	StdErr.1996	NumSrc.1996
0	Afghanistan	AFG	AP	Afghanistan	-1.291705	0.340507	2.0
1	Albania	ALB	ECA	Albania	-0.893903	0.315914	3.0
2	Algeria	DZA	MENA	Algeria	-0.566741	0.262077	4.0
3	Angola	AGO	SSA	Angola	-1.167702	0.262077	4.0
4	Argentina	ARG	AME	Argentina	-0.101317	0.210325	6.0

5 rows × 148 columns

In [28]:
```
df_box_all_regions = merged_df.filter(regex='Estimate.2022|Country') \
    .set_index('Country/Territory').rename(columns={'Estimate.2022': 'all
dfs = [df_box_all_regions]
# группировка данных по регионам
for region in regions:
    df_tmp = merged_df[merged_df.Region==region]
    df_tmp_box = df_tmp.filter(regex='Estimate.2022|Country').set_index('
        .rename(columns={'Estimate.2022': region}) \
        .drop('Country/Territory', axis=1, inplace=False)
    dfs.append(df_tmp_box)
df_box = pd.concat(dfs, sort = True)
```

In [29]:
```
plt.figure(figsize=(15,7))
df_box.boxplot()
```

Out[29]: <AxesSubplot:>



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Задача 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'a необходимо указать значения столбца "Date". Название столбцов должные соответствовать названию акций (имя файла без .csv), а их значения - значениям цены закрытия (столбец "Close" в файлах .csv)

In [30]:
```
#Подключение всех файлов
import glob
glob_file = glob.glob('./stock/*.csv')

#Объединение всех файлов
df_tmp = {}
for file in glob_file: #цикл по всем файлам
    end = file.find('.csv')
    data = pd.read_csv(file, index_col='Date') #считывание данных
    data_name = file.split("/")[-1].split(".")[0] #считывание названия
    df_tmp[data_name] = data['Close'] # цена закрытия
df2 = pd.concat(df_tmp, axis=1, sort = True)
df2.head()
```

Out[30]:

Date	AAPL	SPOT	TSLA	HPQ	NFLX	CSCO	
2022-01-01	174.779999	196.259995	312.239990	36.730000	427.140015	55.669998	37
2022-02-01	165.119995	156.190002	290.143341	34.360001	394.519989	55.770000	36
2022-03-01	174.610001	151.020004	359.200012	36.299999	374.589996	55.759998	35
2022-04-01	157.649994	101.650002	290.253326	36.630001	190.360001	48.980000	31
2022-05-01	148.839996	112.769997	252.753326	38.840000	197.440002	45.049999	23

5 rows × 25 columns

2.2 Рассчитайте корреляционную матрицу для всех акций

In [31]:
```
# корреляционная матрица
df2.corr()
```

Out[31]:

	AAPL	SPOT	TSLA	HPQ	NFLX	CSCO	UBER	
AAPL	1.000000	0.687415	0.248385	0.067074	0.701937	0.589552	0.661323	0.
SPOT	0.687415	1.000000	-0.092332	0.005774	0.920771	0.424007	0.933308	0.
TSLA	0.248385	-0.092332	1.000000	0.568231	-0.251616	0.253808	-0.221155	0.
HPQ	0.067074	0.005774	0.568231	1.000000	-0.203337	0.214262	-0.180970	0.
NFLX	0.701937	0.920771	-0.251616	-0.203337	1.000000	0.497727	0.937042	0.
CSCO	0.589552	0.424007	0.253808	0.214262	0.497727	1.000000	0.326346	0.
UBER	0.661323	0.933308	-0.221155	-0.180970	0.937042	0.326346	1.000000	0.
EBAY	0.115591	0.296858	0.434899	0.744560	0.138580	0.494938	0.085736	1.
ORCL	0.769309	0.763100	-0.310021	-0.260316	0.859397	0.463955	0.832075	-0.
MSFT	0.790691	0.949380	-0.117639	-0.034581	0.900263	0.391476	0.939538	0.
NVDA	0.633114	0.925270	-0.277600	-0.160502	0.910910	0.320159	0.969790	0.
TWLO	0.042914	0.059969	0.703872	0.728572	-0.102302	0.383777	-0.186828	0.
ABNB	0.617430	0.753797	0.353807	0.390153	0.646901	0.594365	0.680764	0.
MU	0.606787	0.902439	0.079944	0.308473	0.789551	0.472688	0.820809	0.
XIACY	0.408747	0.647331	0.184629	0.378627	0.505430	0.474311	0.495835	0.
META	0.705358	0.973401	-0.144519	-0.035611	0.897908	0.374998	0.954444	0.
DBX	0.740429	0.525305	0.037233	-0.177013	0.635239	0.496982	0.595928	-0.
GOOGL	0.806847	0.821587	0.326662	0.263251	0.717756	0.600025	0.737311	0.
TCOM	0.439363	0.640120	-0.586854	-0.443806	0.766681	0.257188	0.754442	-0.
GTLB	0.282373	0.540113	0.260908	0.094128	0.452625	0.068856	0.521399	0.
SHOP	0.465147	0.737909	0.025575	0.436406	0.852517	-0.144612	0.836565	0.
INTC	0.507251	0.645555	0.425236	0.591406	0.447049	0.420854	0.512572	0.
PINS	0.640294	0.842858	-0.253055	-0.285950	0.930638	0.384233	0.907751	-0.
AMZN	0.665715	0.875779	0.302321	0.235247	0.735466	0.404820	0.796897	0.
ADBE	0.833129	0.863827	0.071508	0.081518	0.821314	0.554172	0.834611	0.

25 rows × 25 columns

2.3 Отобразите корреляционную матрицу в виде диаграммы

In [32]:

```
import seaborn as sns # heatmap в sns намного удобнее, чем в matplotlib
plt.title("Корреляционная матрица")
sns.set(rc = {'figure.figsize':(10, 10)})
sns.heatmap(df2.corr(),
            annot=True,
            fmt=".1f",
            #square=True,
            linewidths=.5,
            cmap="viridis"
```

```
)#отображение корреляционной матрицы диаграммой
```

Out[32]: <AxesSubplot:title={'center':'Корреляционная матрица'}>

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)

In [33]:
```
# выберем строку корреляций для AAPL,удалив корреляцию с самой собой
df_aapl = df2.corr().drop(["AAPL"], axis=1)
df_aapl = df_aapl.loc[["AAPL"]].T
df_aapl.head()
```

Out[33]:

	AAPL
SPOT	0.687415
TSLA	0.248385
HPQ	0.067074
NFLX	0.701937
CSCO	0.589552

In [34]:
```
# максимальная корреляция
max_corr_key = df_aapl.idxmax()[0]
max_corr_key
```

Out[34]: 'ADBE'

In [35]:
```
df_aapl.loc[[max_corr_key]]
```

Out[35]:

	AAPL
ADBE	0.833129

In [36]:
```
# минимальная корреляция (с учетом знака)
min_corr_key = df_aapl.idxmin()[0]
min_corr_key
```

Out[36]: 'TWLO'

In [37]:
```
# отрицательных значений для AAPL нет в heatmap
df_aapl.loc[[min_corr_key]]
```

Out[37]:

	AAPL
TWLO	0.042914

In [38]:
```
# минимальная корреляция (без учета знака)
abs_min_corr_key = df_aapl.fillna(0).abs().idxmin()[0]
abs_min_corr_key
```

Out[38]: 'TWLO'

In [39]:
```
df_aapl.loc[[abs_min_corr_key]]
```

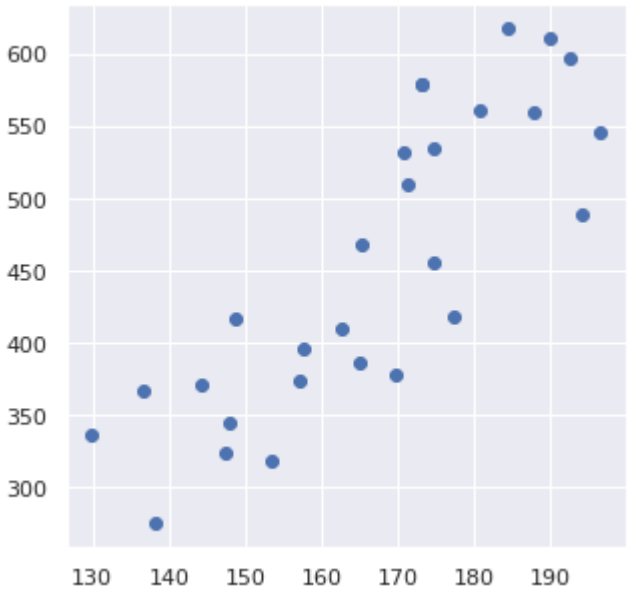
Out[39]:

	AAPL
TWLO	0.042914

2.5 Постройте диаграммы разброса (Ваша компания - Компания с min), (Ваша компания - Компания с max), (Ваша компания - Компания с none)

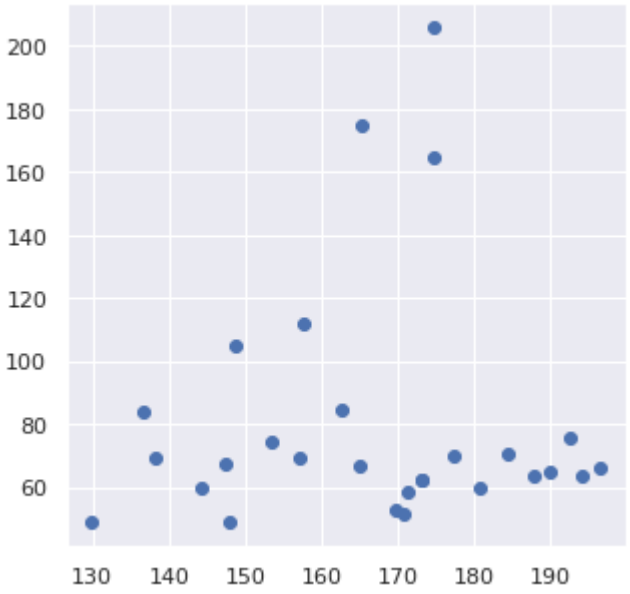
In [40]:
```
# диаграмма разброса с max corr
sns.set (rc = {'figure.figsize':(5, 5)})
plt.scatter (df2["AAPL"], df2[max_corr_key])
```

Out[40]: <matplotlib.collections.PathCollection at 0x7f29806f01c0>



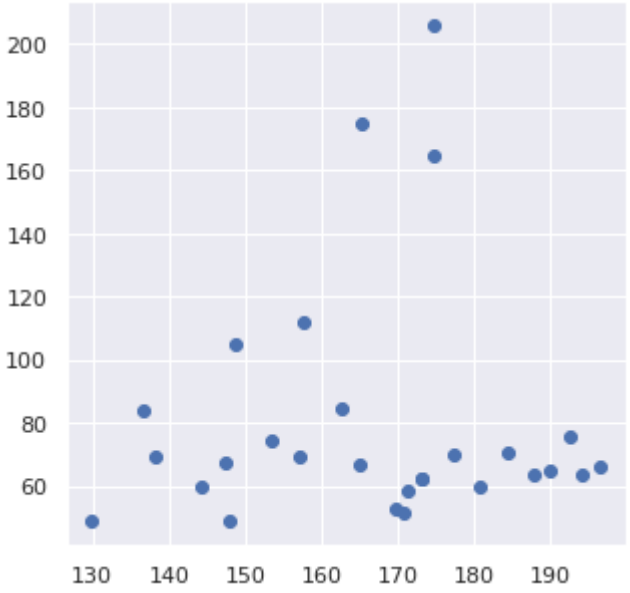
In [41]:
```
# диаграмма разброса с min corr (с учетом знака)
plt.scatter (df2["AAPL"], df2[min_corr_key])
```

Out[41]: <matplotlib.collections.PathCollection at 0x7f29806a4b80>



In [42]:
```
# диаграмма разброса с min corr (без учета знака)
plt.scatter (df2["AAPL"], df2[abs_min_corr_key])
```

Out[42]: <matplotlib.collections.PathCollection at 0x7f29805c9d30>



2.6 Рассчитайте среднюю цену акций для каждого месяца

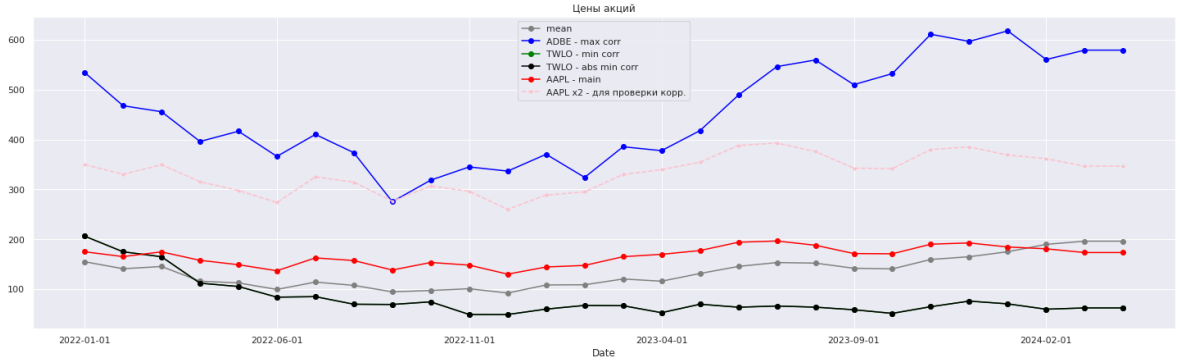
In [43]:
```
# средняя цена акций для каждого месяца
mean = df2.T.mean()
mean.name = "mean"
mean
```

Out[43]:
```
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 [44]:
```
mean.plot(grid=1, figsize=(25,7), color='grey', marker='o', legend=True,

df2.rename(columns={max_corr_key: max_corr_key + " - max corr"})[max_corr
    .plot(grid=1, figsize=(25,7), color='blue', marker='o', legend=True)

df2.rename(columns={abs_min_corr_key: min_corr_key + " - min corr"})[min_
    .plot(grid=1, figsize=(25,7), color='green', marker='o', legend=True)

df2.rename(columns={abs_min_corr_key: abs_min_corr_key + " - abs min corr
    .plot(grid=1, figsize=(25,7), color='black', marker='o', legend=True)

df2.rename(columns={"AAPL": "AAPL - main"})["AAPL - main"].plot(grid=1, f
                                                                color='re

# корреляция AAPL с ADBE плохо видна в изначальном виде, отмасштабируем з
aapl_x_2 = df2["AAPL"]*2
aapl_x_2.name = "AAPL x2 - для проверки корр."
aapl_x_2.plot(grid=1, figsize=(25,7), color='pink', style = "--", marker=
```

Out[44]:
```
<AxesSubplot:title={'center':'Цены акций'}, xlabel='Date'>
```


Цены акций
mean
ADBE - max corr
TWLO - min corr
TWLO - abs min corr
AAPL - main
AAPL x2 - для проверки корр.
600
500
400
300
200
100
2022-01-01
2022-06-01
2022-11-01
2023-04-01
2023-09-01
2024-02-01
Date

In [ ]: