**Министерство науки и высшего образования Российской Федерации**
**Федеральное государственное бюджетное образовательное учреждение высшего образования**
**«Московский государственный технический университет имени Н.Э. Баумана**
**(национальный исследовательский университет)»**
**(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ ____ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ_____

КАФЕДРА _____КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ_____

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.04.01 **Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/05 Современные интеллектуальные программно-аппаратные комплексы.**

# Отчет

## по домашней работе №1

**Название работы:** Дескриптивный анализ данных

**Дисциплина:** Методы машинного обучения

Студенты гр. ИУ6-21М _____ Минайчев А.А.
(Подпись, дата) (Фамилия И.О.)

Преподаватель _____ (Фамилия
(Подпись, дата) И.О.)

2023 г.

## ДЗ1 Минайчев Антон

In [1]:
```
surname = "Минайчев"  # 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:  5
Задача № 1, шаг 11 - вариант:  2
задача № 2 - вариант:  2
```

# Задание 1

Для выполнения ДЗ используется набор данных за 2021 год с сайта
http://info.worldbank.org/governance/wgi/

In [2]:
```
import pandas as pd
```

## Пункт 1

Загрузим данные из EXCEL файла в dataframe, пропустив 14 первых строк, т.к. они не значимой информации и мешают правильному отображению df

In [3]:
```
#прочитаем excel файл при помощи функции read_excel библиотеки pandas
df = pd.read_excel('wgidataset.xlsx', skiprows=14, sheet_name="ControlofCorruption")
l_a = ['NumSrc','Lower','Upper','StdErr']
#создам список с годами, которые отображены в excel файле
years = ['1996', '1998', '2000', '2002', '2003', '2004', '2005', '2006', '2007', '2008', '200
         '2013', '2014', '2015', '2016', '2017', '2018', '2019', '2020', '2021']
# заполним список l_a наименованием столбцов, которые можно удалить и которые не понадобятся
for i in range (1, 23):
    l_a.append('NumSrc.'+str(i))
    l_a.append('Lower.'+str(i))
    l_a.append('Upper.'+str(i))
    l_a.append('StdErr.'+str(i))
df = df.drop(columns = l_a)
# в цикле изменим название столбцов, начиная с 3
for i in range(2,len(df.columns)):
    if i % 2 ==0:
        df = df.rename(columns = {df.columns[i]:years[0] + ' ' + 'Estimate'})
    else:
        df = df.rename(columns = {df.columns[i]:years[0] + ' ' + 'Rank'})
        years.pop(0)
df
```

Out[3]:

| | Country/Territory | Code | 1996 Estimate | 1996 Rank | 1998 Estimate | 1998 Rank | 2000 Estimate | 2000 Rank | 2002 Estimate | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Aruba | ABW | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **1** | Andorra | ADO | 1.318143 | 87.096771 | 1.334759 | 89.304810 | 1.313404 | 88.297874 | 1.310744 | 87.8 |
| **2** | Afghanistan | AFG | -1.291705 | 4.301075 | -1.176012 | 8.021390 | -1.271724 | 4.787234 | -1.251137 | 4.7 |
| **3** | Angola | AGO | -1.167702 | 9.677420 | -1.180451 | 7.486631 | -1.197514 | 8.510638 | -1.155493 | 7.9 |
| **4** | Anguilla | AIA | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **209** | Serbia | SRB | -1.140072 | 11.827957 | -1.195605 | 6.417112 | -1.156671 | 9.042553 | -0.895785 | 23.2 |
| **210** | South Africa | ZAF | 0.732927 | 76.344086 | 0.638809 | 72.727272 | 0.550270 | 70.212769 | 0.332902 | 67.1 |
| **211** | Congo, Dem. Rep. | ZAR | -1.647852 | 0.000000 | -1.416679 | 1.069519 | -1.459175 | 0.531915 | -1.449971 | 1.0 |
| **212** | Zambia | ZMB | -0.840641 | 24.731182 | -0.853156 | 26.737968 | -0.818261 | 26.595745 | -0.758519 | 29.6 |
| **213** | Zimbabwe | ZWE | -0.278847 | 47.849461 | -0.504802 | 37.433155 | -1.127275 | 10.638298 | -1.156760 | 7.4 |

214 rows × 48 columns

## Пункт 2

Сортируем данные по убыванию индекса

In [4]:
```
df = df.sort_index(ascending=False)
df
```

Out[4]:

| | Country/Territory | Code | 1996 Estimate | 1996 Rank | 1998 Estimate | 1998 Rank | 2000 Estimate | 2000 Rank | 2002 Estimate | |
|---|---|---|---|---|---|---|---|---|---|---|
| **213** | Zimbabwe | ZWE | -0.278847 | 47.849461 | -0.504802 | 37.433155 | -1.127275 | 10.638298 | -1.156760 | 7.4 |
| **212** | Zambia | ZMB | -0.840641 | 24.731182 | -0.853156 | 26.737968 | -0.818261 | 26.595745 | -0.758519 | 29.6 |
| **211** | Congo, Dem. Rep. | ZAR | -1.647852 | 0.000000 | -1.416679 | 1.069519 | -1.459175 | 0.531915 | -1.449971 | 1.0 |
| **210** | South Africa | ZAF | 0.732927 | 76.344086 | 0.638809 | 72.727272 | 0.550270 | 70.212769 | 0.332902 | 67.1 |
| **209** | Serbia | SRB | -1.140072 | 11.827957 | -1.195605 | 6.417112 | -1.156671 | 9.042553 | -0.895785 | 23.2 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **4** | Anguilla | AIA | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **3** | Angola | AGO | -1.167702 | 9.677420 | -1.180451 | 7.486631 | -1.197514 | 8.510638 | -1.155493 | 7.9 |
| **2** | Afghanistan | AFG | -1.291705 | 4.301075 | -1.176012 | 8.021390 | -1.271724 | 4.787234 | -1.251137 | 4.7 |
| **1** | Andorra | ADO | 1.318143 | 87.096771 | 1.334759 | 89.304810 | 1.313404 | 88.297874 | 1.310744 | 87.8 |
| **0** | Aruba | ABW | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |

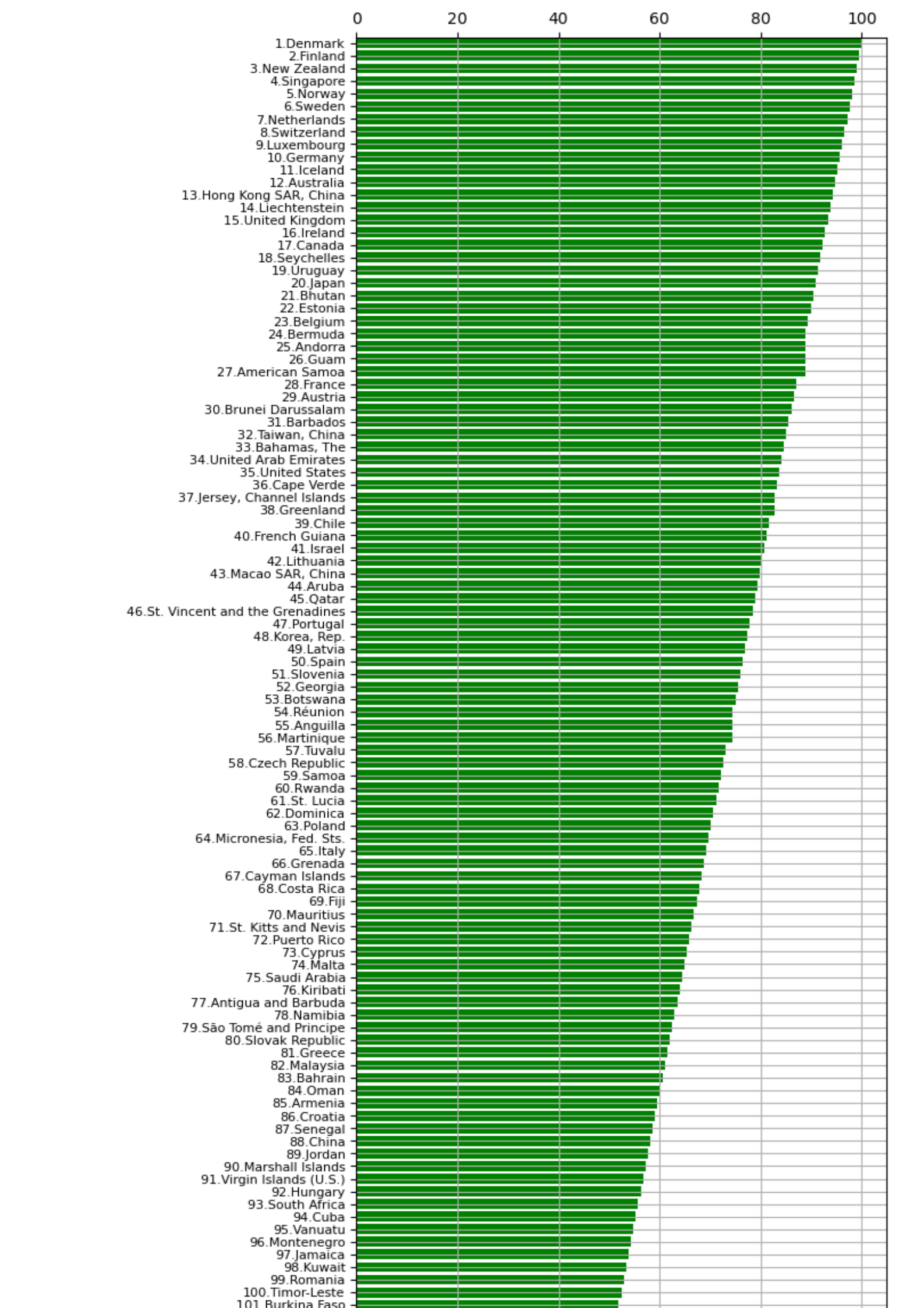
214 rows × 48 columns

In [5]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

## Пункт 3

Отобразим данные по индексу CPI за 2021 год в виде горизонтального столбчатого графика.

In [6]:
```
# удалим NAN значения для корректного отображения столбчатого графика и отсортируем по рангу
df_cpi = df[["Country/Territory","2021 Rank"]].dropna()
df_sorted = df_cpi.sort_values("2021 Rank", ascending=False)
countries = []
pos = 1
# цикл для добавления места страны в её наименование для отображения на графике
for i in df_sorted['Country/Territory']:
    countries.append(str(pos)+'.'+i)
    pos += 1
countries.reverse()
df_sorted.sort_values('2021 Rank',inplace=True)
# создадим столбчатый график согласно примеру в задании
plt.figure("Corruption Perceptions Index",figsize=[6, 30])
ax = plt.subplot(1,1,1)
#основные параметры графика
plt.barh(countries,color="green", width = df_sorted['2021 Rank'], height = 0.8)
#отображения значений оси x наверху графика
ax.xaxis.tick_top()
#изменение шрифта наименования стран
for label in (ax.get_yticklabels()):
    label.set_fontsize(8)
#включаем сетку x и y
plt.grid(True)
#убираем пустые поля в верхней и нижней частях графика
plt.margins(y=0)
```


0
20
40
60
80
100
1.Denmark
2.Finland
3.New Zealand
4.Singapore
5.Norway
6.Sweden
7.Netherlands
8.Switzerland
9.Luxembourg
10.Germany
11.Iceland
12.Australia
13.Hong Kong SAR, China
14.Liechtenstein
15.United Kingdom
16.Ireland
17.Canada
18.Seychelles
19.Uruguay
20.Japan
21.Bhutan
22.Estonia
23.Belgium
24.Bermuda
25.Andorra
26.Guam
27.American Samoa
28.France
29.Austria
30.Brunei Darussalam
31.Barbados
32.Taiwan, China
33.Bahamas, The
34.United Arab Emirates
35.United States
36.Cape Verde
37.Jersey, Channel Islands
38.Greenland
39.Chile
40.French Guiana
41.Israel
42.Lithuania
43.Macao SAR, China
44.Aruba
45.Qatar
46.St. Vincent and the Grenadines
47.Portugal
48.Korea, Rep.
49.Latvia
50.Spain
51.Slovenia
52.Georgia
53.Botswana
54.Réunion
55.Anguilla
56.Martinique
57.Tuvalu
58.Czech Republic
59.Samoa
60.Rwanda
61.St. Lucia
62.Dominica
63.Poland
64.Micronesia, Fed. Sts.
65.Italy
66.Grenada
67.Cayman Islands
68.Costa Rica
69.Fiji
70.Mauritius
71.St. Kitts and Nevis
72.Puerto Rico
73.Cyprus
74.Malta
75.Saudi Arabia
76.Kiribati
77.Antigua and Barbuda
78.Namibia
79.São Tomé and Principe
80.Slovak Republic
81.Greece
82.Malaysia
83.Bahrain
84.Oman
85.Armenia
86.Croatia
87.Senegal
88.China
89.Jordan
90.Marshall Islands
91.Virgin Islands (U.S.)
92.Hungary
93.South Africa
94.Cuba
95.Vanuatu
96.Montenegro
97.Jamaica
98.Kuwait
99.Romania
100.Timor-Leste
101.Burkina Faso

## Пункты 4, 5

Для последующего выполнения заданий создадим ещё один DF из другого EXCEL файла с соотношением регионов и стран. Используя этот DF добавим фильтр региона SSA (5 вариант задания), а также сбросим индексы, чтобы отсчёт начался с нуля.

In [7]:
```
#считываем данные в df2
df2 = pd.read_excel('regions.xlsx')
#в новом df2_ssa оставляем только данные, которые относятся к региону SSA из df2
df2_ssa = df2.loc[df2['Region'] == "SSA"]
#создаем ещё один df2_ssa1, на основе df с основными данными, в котором будут только те стран
df2_ssa1 = df.loc[df['Code'].isin(df2_ssa.Code)].reset_index().drop(columns = ['index'])
df2_ssa1
```

Out[7]:

| | Country/Territory | Code | 1996 Estimate | 1996 Rank | 1998 Estimate | 1998 Rank | 2000 Estimate | 2000 Rank | 2002 Estimate | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Zimbabwe | ZWE | -0.278847 | 47.849461 | -0.504802 | 37.433155 | -1.127275 | 10.638298 | -1.156760 | 7.40 |
| **1** | Zambia | ZMB | -0.840641 | 24.731182 | -0.853156 | 26.737968 | -0.818261 | 26.595745 | -0.758519 | 29.62 |
| **2** | South Africa | ZAF | 0.732927 | 76.344086 | 0.638809 | 72.727272 | 0.550270 | 70.212769 | 0.332902 | 67.19 |
| **3** | Uganda | UGA | -0.723757 | 27.956989 | -0.989971 | 18.716578 | -1.028190 | 14.361702 | -1.025739 | 14.28 |
| **4** | Tanzania | TZA | -0.702762 | 29.569893 | -0.803638 | 27.807487 | -0.810984 | 27.127659 | -0.840932 | 24.86 |
| **5** | Togo | TGO | -0.842621 | 24.193548 | -0.831448 | 27.272728 | -0.751899 | 29.255320 | -0.758070 | 30.15 |
| **6** | Chad | TCD | -1.380424 | 3.225806 | -1.336609 | 1.604278 | -1.325014 | 2.659575 | -1.270306 | 3.70 |
| **7** | Seychelles | SYC | 0.545952 | 72.580643 | 0.570708 | 71.657753 | 0.592291 | 71.808510 | 0.694715 | 75.66 |
| **8** | Eswatini | SWZ | 0.037314 | 58.602150 | 0.054524 | 56.684490 | -0.045108 | 55.319149 | -0.060140 | 55.55 |
| **9** | São Tomé and Principe | STP | 0.401667 | 66.666664 | 0.401506 | 66.310158 | 0.346064 | 64.893616 | 0.237328 | 63.49 |
| **10** | Somalia | SOM | -1.273832 | 5.376344 | -1.169290 | 8.556149 | -1.231536 | 5.851064 | -1.032931 | 13.22 |
| **11** | Sierra Leone | SLE | -0.756474 | 26.344086 | -0.724984 | 30.481283 | -0.736802 | 30.319149 | -0.789459 | 27.51 |
| **12** | Senegal | SEN | -0.142715 | 52.150539 | -0.122647 | 52.406418 | -0.094586 | 53.723404 | 0.247914 | 64.55 |
| **13** | Sudan | SDN | -1.240006 | 6.989247 | -1.119795 | 12.834225 | -1.018885 | 14.893617 | -1.035823 | 12.16 |
| **14** | Rwanda | RWA | -0.745140 | 26.881720 | -0.729861 | 29.946524 | -0.733605 | 30.851065 | -0.716066 | 31.74 |
| **15** | Nigeria | NGA | -1.189009 | 8.602151 | -1.160613 | 10.160428 | -1.324634 | 3.191489 | -1.502068 | 0.52 |
| **16** | Niger | NER | -0.865590 | 20.967741 | -1.073254 | 14.973262 | -0.945642 | 19.148935 | -1.020487 | 15.34 |
| **17** | Namibia | NAM | 0.808773 | 76.881721 | 0.735703 | 75.935829 | 0.773133 | 76.595741 | 0.114023 | 59.25 |
| **18** | Malawi | MWI | -0.316063 | 47.311829 | -0.260301 | 49.197861 | -0.228705 | 48.936169 | -1.021618 | 14.81 |
| **19** | Mauritius | MUS | 0.034662 | 58.064518 | 0.221582 | 60.962566 | 0.198928 | 61.702129 | 0.328717 | 66.66 |
| **20** | Mauritania | MRT | -0.555694 | 34.408604 | -0.515584 | 36.898396 | -0.486817 | 39.893616 | -0.083034 | 52.91 |
| **21** | Mozambique | MOZ | -0.424003 | 40.860214 | -0.390382 | 43.315510 | -0.534364 | 37.234043 | -0.499133 | 38.62 |
| **22** | Mali | MLI | -0.782562 | 25.806452 | -0.899157 | 25.133690 | -0.878510 | 23.404255 | -0.724598 | 30.68 |
| **23** | Madagascar | MDG | -0.371094 | 44.086021 | -0.560736 | 35.828876 | -0.478630 | 40.957447 | 0.001223 | 56.61 |
| **24** | Lesotho | LSO | 0.086545 | 59.677418 | -0.110705 | 52.941177 | -0.165395 | 52.659573 | -0.173949 | 50.79 |
| **25** | Liberia | LBR | -1.500141 | 2.150538 | -1.536518 | 0.000000 | -1.098915 | 11.702127 | -1.034289 | 12.69 |
| **26** | Kenya | KEN | -1.158849 | 10.752688 | -1.165813 | 9.625669 | -1.144364 | 10.106383 | -1.003751 | 16.93 |
| **27** | Equatorial Guinea | GNQ | -1.264369 | 6.451613 | -1.187925 | 6.951872 | -1.212375 | 6.382979 | -1.222883 | 5.29 |
| **28** | Guinea-Bissau | GNB | -1.194655 | 8.064516 | -1.122083 | 12.299465 | -0.915325 | 21.276596 | -1.044214 | 11.64 |
| **29** | Gambia, The | GMB | -0.374195 | 43.548386 | -0.364896 | 44.919785 | -0.377791 | 45.212765 | -0.233313 | 48.67 |
| **30** | Guinea | GIN | -0.939942 | 18.279570 | -1.075010 | 14.438502 | -1.050161 | 13.297873 | -0.767054 | 28.57 |
| **31** | Ghana | GHA | -0.339950 | 46.236561 | -0.329743 | 47.058823 | -0.095986 | 53.191490 | -0.342790 | 43.91 |
| **32** | Gabon | GAB | -1.102377 | 13.978495 | -0.995288 | 17.647058 | -0.592372 | 35.638298 | -0.520379 | 37.56 |
| **33** | Ethiopia | ETH | -0.930546 | 18.817204 | -0.903204 | 24.598930 | -0.906757 | 22.340425 | -0.717909 | 31.21 |
| **34** | Eritrea | ERI | 0.505685 | 72.043015 | 0.516642 | 69.518715 | 0.496909 | 69.148933 | 0.125430 | 60.31 |
| **35** | Djibouti | DJI | -0.719981 | 29.032259 | -0.682330 | 32.085560 | -0.636604 | 34.042553 | -0.628374 | 34.39 |
| **36** | Cape Verde | CPV | 1.143337 | 83.333336 | 1.155066 | 83.957222 | 1.127157 | 84.042557 | 0.495939 | 71.95 |

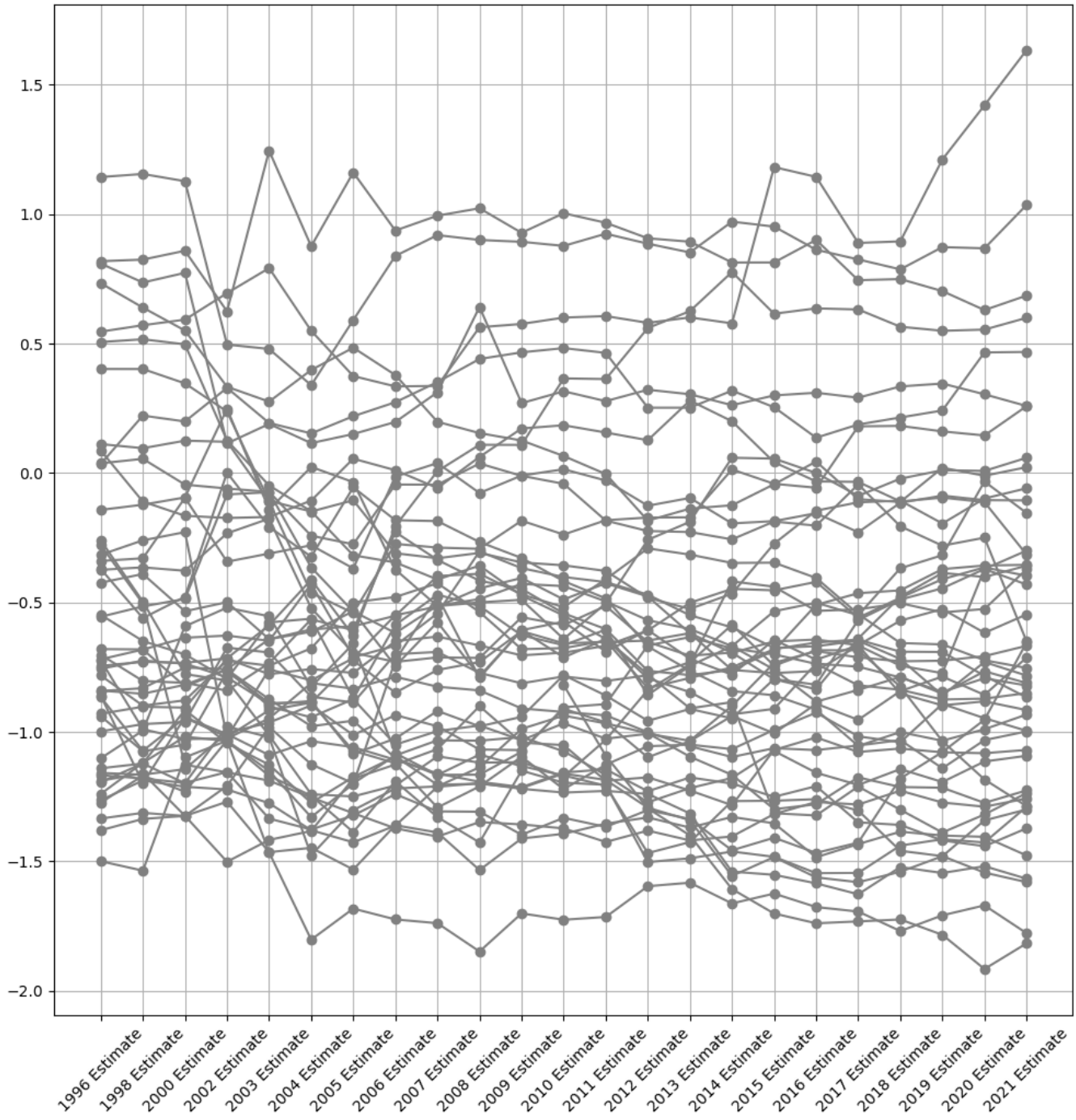
| | Country/Territory | Code | 1996 Estimate | 1996 Rank | 1998 Estimate | 1998 Rank | 2000 Estimate | 2000 Rank | 2002 Estimate | |
|---|---|---|---|---|---|---|---|---|---|---|
| **37** | Comoros | COM | -0.998565 | 16.666666 | -0.970211 | 20.320856 | -0.962092 | 18.617022 | -0.674768 | 32.27 |
| **38** | Congo, Rep. | COG | -0.860740 | 23.118280 | -1.198797 | 5.882353 | -0.943887 | 19.680851 | -1.009018 | 16.40 |
| **39** | Cameroon | CMR | -1.334985 | 3.763441 | -1.313289 | 3.208556 | -1.325292 | 2.127660 | -1.209112 | 5.82 |
| **40** | Côte d'Ivoire | CIV | -0.260567 | 49.462364 | -0.497032 | 37.967915 | -0.777199 | 28.723404 | -0.801531 | 26.45 |
| **41** | Central African Republic | CAF | -1.140931 | 11.290322 | -1.122684 | 11.764706 | -1.198896 | 7.978724 | -0.979823 | 18.51 |
| **42** | Botswana | BWA | 0.817961 | 77.419357 | 0.824314 | 78.074867 | 0.858369 | 78.191490 | 0.624538 | 74.60 |
| **43** | South Sudan | SSD | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **44** | Burkina Faso | BFA | 0.111519 | 60.215054 | 0.095590 | 57.754009 | 0.124697 | 60.106384 | 0.120320 | 59.78 |
| **45** | Benin | BEN | -0.548115 | 34.946236 | -0.649413 | 33.689838 | -0.700918 | 31.382978 | -0.795710 | 26.98 |
| **46** | Burundi | BDI | -0.680635 | 30.645161 | -0.681750 | 32.620319 | -0.824704 | 26.063829 | -0.759897 | 29.10 |
| **47** | Angola | AGO | -1.167702 | 9.677420 | -1.180451 | 7.486631 | -1.197514 | 8.510638 | -1.155493 | 7.93 |

## Пункт 6

Для построения графика индекса CPI за года с 1996 до 2021 необходимо транспонировать исходный DF, легенду убираем для читаемости.

In [8]:
```
plt.figure("SSA",figsize=[12, 12])
ax1 = plt.subplot()
#указываем колонки по которым будет строиться график
plt.plot(df2_ssa1[["1996 Estimate", "1998 Estimate","2000 Estimate","2002 Estimate",
                   "2003 Estimate","2004 Estimate","2005 Estimate","2006 Estimate",
                   "2007 Estimate","2008 Estimate","2009 Estimate", "2010 Estimate",
                   "2011 Estimate", "2012 Estimate", "2013 Estimate", "2014 Estimate",
                   '2015 Estimate', '2016 Estimate', '2017 Estimate', '2018 Estimate',
                   '2019 Estimate', '2020 Estimate', '2021 Estimate']].transpose(),'o-', colo
#указываем наименование каждого значения по оси x
ax1.set_xticklabels(["1996 Estimate", "1998 Estimate","2000 Estimate",
                     "2002 Estimate","2003 Estimate","2004 Estimate",
                     "2005 Estimate","2006 Estimate","2007 Estimate","2008 Estimate",
                     "2009 Estimate", "2010 Estimate", "2011 Estimate", "2012 Estimate",
                     "2013 Estimate", "2014 Estimate", '2015 Estimate', '2016 Estimate',
                     '2017 Estimate', '2018 Estimate', '2019 Estimate', '2020 Estimate', '202
#вкл. сетку
plt.grid(True)
```

```
C:\Users\User\AppData\Local\Temp\ipykernel_616\2250511806.py:11: UserWarning: FixedFormatter
should only be used together with FixedLocator
  ax1.set_xticklabels(["1996 Estimate", "1998 Estimate","2000 Estimate",
```




## Пункт 7

Найдем страны с наименьшим и наибольшим индексом коррупкции за 2021 год

In [9]:
```
#находим id строки с наименьшим и наибольшим значение 2021 estimate
min_cpi = df2_ssa1['2021 Estimate'].idxmin()
max_cpi = df2_ssa1['2021 Estimate'].idxmax()
#по данным id выводим таблицу с данными по двум странам
df2_ssa1.loc[[min_cpi, max_cpi],['Country/Territory', '2021 Estimate']]
```

Out[9]:

| | Country/Territory | 2021 Estimate |
|---|---|---|
| **43** | South Sudan | -1.817446 |
| **7** | Seychelles | 1.633352 |

Как видно - самая большая коррупция в Сомали, а наименьшая в рассматриваемом регионе - на Сейшелах.

## Пункт 8

Определим средние значения за каждый год в период с 1996 по 2021 и занесем их в колонку "Среднее значение"

In [10]:
```
#список необходимых нам колонок
list = ["1996 Estimate", "1998 Estimate","2000 Estimate","2002 Estimate","2003 Estimate",
        "2004 Estimate","2005 Estimate","2006 Estimate","2007 Estimate","2008 Estimate",
        "2009 Estimate", "2010 Estimate", "2011 Estimate", "2012 Estimate", "2013 Estimate",
        "2014 Estimate", '2015 Estimate', '2016 Estimate', '2017 Estimate', '2018 Estimate',
        '2019 Estimate', '2020 Estimate', '2021 Estimate']
#используя список, выбираем эти колонки из df2_ssa1 и транспонируем
df2_ssa_transposed = df2_ssa1[list].transpose()
#добавляем новый столбец со средним значением
df2_ssa_transposed['Среднее значение'] = df2_ssa_transposed.mean(axis=1)
df2_ssa_transposed
```

Out[10]:

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | |
|---|---|---|---|---|---|---|---|---|---|---|
| **1996 Estimate** | -0.278847 | -0.840641 | 0.732927 | -0.723757 | -0.702762 | -0.842621 | -1.380424 | 0.545952 | 0.037314 | 0.40 |
| **1998 Estimate** | -0.504802 | -0.853156 | 0.638809 | -0.989971 | -0.803638 | -0.831448 | -1.336609 | 0.570708 | 0.054524 | 0.40 |
| **2000 Estimate** | -1.127275 | -0.818261 | 0.550270 | -1.028190 | -0.810984 | -0.751899 | -1.325014 | 0.592291 | -0.045108 | 0.34 |
| **2002 Estimate** | -1.156760 | -0.758519 | 0.332902 | -1.025739 | -0.840932 | -0.758070 | -1.270306 | 0.694715 | -0.060140 | 0.23 |
| **2003 Estimate** | -1.188868 | -0.641859 | 0.275541 | -0.948509 | -0.734542 | -0.902239 | -1.465014 | 0.791488 | -0.073140 | -0.10 |
| **2004 Estimate** | -1.253563 | -0.604488 | 0.397753 | -0.811879 | -0.565570 | -0.943994 | -1.448346 | 0.549317 | -0.242530 | -0.46 |
| **2005 Estimate** | -1.314617 | -0.592440 | 0.483857 | -0.833649 | -0.607606 | -0.873771 | -1.531993 | 0.374255 | -0.275532 | -0.54 |
| **2006 Estimate** | -1.372949 | -0.551736 | 0.378271 | -0.787619 | -0.228577 | -1.049607 | -1.361009 | 0.334429 | -0.044869 | -0.27 |
| **2007 Estimate** | -1.404758 | -0.398109 | 0.197672 | -0.825755 | -0.335751 | -0.997273 | -1.388993 | 0.336277 | -0.045250 | -0.28 |
| **2008 Estimate** | -1.348838 | -0.389601 | 0.153829 | -0.840164 | -0.416348 | -0.975758 | -1.534189 | 0.562799 | 0.036358 | -0.29 |
| **2009 Estimate** | -1.357875 | -0.469498 | 0.125427 | -0.909787 | -0.447849 | -1.032310 | -1.412184 | 0.575691 | -0.011177 | -0.18 |
| **2010 Estimate** | -1.373287 | -0.518319 | 0.065393 | -0.923494 | -0.542279 | -0.967423 | -1.394239 | 0.600198 | -0.041858 | -0.23 |
| **2011 Estimate** | -1.425627 | -0.410753 | -0.004234 | -0.936402 | -0.603360 | -1.026531 | -1.355161 | 0.606024 | -0.184127 | -0.18 |
| **2012 Estimate** | -1.381803 | -0.291441 | -0.184171 | -1.001575 | -0.764620 | -1.007718 | -1.330997 | 0.579943 | -0.224430 | -0.17 |
| **2013 Estimate** | -1.419667 | -0.315543 | -0.137832 | -1.055946 | -0.780951 | -1.034692 | -1.370108 | 0.600916 | -0.228339 | -0.17 |
| **2014 Estimate** | -1.404367 | -0.347683 | -0.124875 | -1.097377 | -0.761074 | -0.899913 | -1.329005 | 0.577941 | -0.256666 | 0.01 |
| **2015 Estimate** | -1.317811 | -0.346599 | -0.043699 | -1.063391 | -0.686202 | -0.733910 | -1.353646 | 1.181637 | -0.187602 | -0.04 |
| **2016 Estimate** | -1.271258 | -0.405702 | 0.044075 | -1.070588 | -0.509641 | -0.688368 | -1.484633 | 1.143864 | -0.203669 | -0.05 |
| **2017 Estimate** | -1.281081 | -0.545036 | -0.101496 | -1.052901 | -0.464720 | -0.712416 | -1.433871 | 0.888518 | -0.044150 | 0.18 |
| **2018 Estimate** | -1.227581 | -0.657134 | -0.112126 | -1.032128 | -0.453522 | -0.727730 | -1.385962 | 0.894634 | -0.205999 | 0.18 |
| **2019 Estimate** | -1.273280 | -0.662320 | 0.020065 | -1.140858 | -0.371587 | -0.725558 | -1.400778 | 1.211095 | -0.281041 | 0.16 |
| **2020 Estimate** | -1.289440 | -0.723668 | -0.010780 | -1.032168 | -0.357577 | -0.707525 | -1.405847 | 1.420243 | -0.250072 | 0.14 |
| **2021 Estimate** | -1.257897 | -0.753424 | 0.022103 | -0.998557 | -0.355093 | -0.668003 | -1.477722 | 1.633352 | -0.650570 | 0.25 |

23 rows × 49 columns

## Пункт 9

Выделим столбец с данными CPI по России за года с 1996 по 2021

In [11]:
```
#выделяем данные из df с значением Russian Federation в поле Country/Territory, индексы сбрас
df_russia = df.loc[df['Country/Territory'] == "Russian Federation"].reset_index().drop(column
#из этих данных нас интересуют те же столбцы, что и в предыдущем пункте
column_russia = df_russia[list].transpose()
column_russia
```

Out[11]:

| | 0 |
|---|---|
| **1996 Estimate** | -1.053342 |
| **1998 Estimate** | -0.954374 |
| **2000 Estimate** | -0.943414 |
| **2002 Estimate** | -0.954848 |
| **2003 Estimate** | -0.783092 |
| **2004 Estimate** | -0.825626 |
| **2005 Estimate** | -0.847121 |
| **2006 Estimate** | -0.940848 |
| **2007 Estimate** | -1.017581 |
| **2008 Estimate** | -1.125229 |
| **2009 Estimate** | -1.141307 |
| **2010 Estimate** | -1.099215 |
| **2011 Estimate** | -1.074377 |
| **2012 Estimate** | -1.051572 |
| **2013 Estimate** | -1.020387 |
| **2014 Estimate** | -0.918525 |
| **2015 Estimate** | -0.952243 |
| **2016 Estimate** | -0.817593 |
| **2017 Estimate** | -0.893895 |
| **2018 Estimate** | -0.848471 |
| **2019 Estimate** | -0.802163 |
| **2020 Estimate** | -0.909426 |
| **2021 Estimate** | -0.900474 |

Добавим выделенный столбец в исходный df2_ssa_transposed под названием "RUS"

In [12]:
```
df2_ssa_transposed['RUS'] = column_russia[0]
df2_ssa_transposed
```

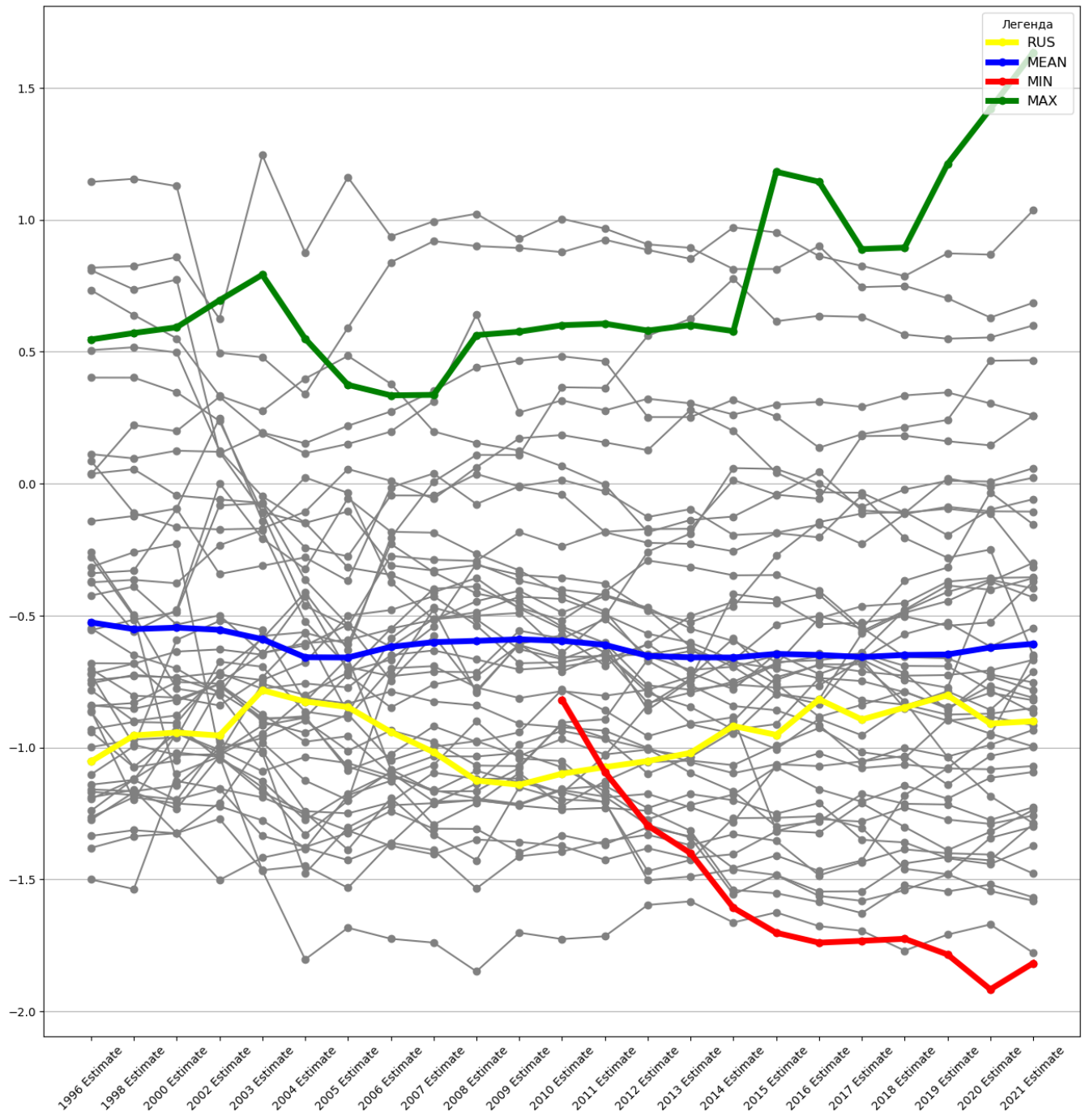
Out[12]:

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | |
|---|---|---|---|---|---|---|---|---|---|---|
| **1996 Estimate** | -0.278847 | -0.840641 | 0.732927 | -0.723757 | -0.702762 | -0.842621 | -1.380424 | 0.545952 | 0.037314 | 0.40 |
| **1998 Estimate** | -0.504802 | -0.853156 | 0.638809 | -0.989971 | -0.803638 | -0.831448 | -1.336609 | 0.570708 | 0.054524 | 0.40 |
| **2000 Estimate** | -1.127275 | -0.818261 | 0.550270 | -1.028190 | -0.810984 | -0.751899 | -1.325014 | 0.592291 | -0.045108 | 0.34 |
| **2002 Estimate** | -1.156760 | -0.758519 | 0.332902 | -1.025739 | -0.840932 | -0.758070 | -1.270306 | 0.694715 | -0.060140 | 0.23 |
| **2003 Estimate** | -1.188868 | -0.641859 | 0.275541 | -0.948509 | -0.734542 | -0.902239 | -1.465014 | 0.791488 | -0.073140 | -0.10 |
| **2004 Estimate** | -1.253563 | -0.604488 | 0.397753 | -0.811879 | -0.565570 | -0.943994 | -1.448346 | 0.549317 | -0.242530 | -0.46 |
| **2005 Estimate** | -1.314617 | -0.592440 | 0.483857 | -0.833649 | -0.607606 | -0.873771 | -1.531993 | 0.374255 | -0.275532 | -0.54 |
| **2006 Estimate** | -1.372949 | -0.551736 | 0.378271 | -0.787619 | -0.228577 | -1.049607 | -1.361009 | 0.334429 | -0.044869 | -0.27 |
| **2007 Estimate** | -1.404758 | -0.398109 | 0.197672 | -0.825755 | -0.335751 | -0.997273 | -1.388993 | 0.336277 | -0.045250 | -0.28 |
| **2008 Estimate** | -1.348838 | -0.389601 | 0.153829 | -0.840164 | -0.416348 | -0.975758 | -1.534189 | 0.562799 | 0.036358 | -0.29 |
| **2009 Estimate** | -1.357875 | -0.469498 | 0.125427 | -0.909787 | -0.447849 | -1.032310 | -1.412184 | 0.575691 | -0.011177 | -0.18 |
| **2010 Estimate** | -1.373287 | -0.518319 | 0.065393 | -0.923494 | -0.542279 | -0.967423 | -1.394239 | 0.600198 | -0.041858 | -0.23 |
| **2011 Estimate** | -1.425627 | -0.410753 | -0.004234 | -0.936402 | -0.603360 | -1.026531 | -1.355161 | 0.606024 | -0.184127 | -0.18 |
| **2012 Estimate** | -1.381803 | -0.291441 | -0.184171 | -1.001575 | -0.764620 | -1.007718 | -1.330997 | 0.579943 | -0.224430 | -0.17 |
| **2013 Estimate** | -1.419667 | -0.315543 | -0.137832 | -1.055946 | -0.780951 | -1.034692 | -1.370108 | 0.600916 | -0.228339 | -0.17 |
| **2014 Estimate** | -1.404367 | -0.347683 | -0.124875 | -1.097377 | -0.761074 | -0.899913 | -1.329005 | 0.577941 | -0.256666 | 0.01 |
| **2015 Estimate** | -1.317811 | -0.346599 | -0.043699 | -1.063391 | -0.686202 | -0.733910 | -1.353646 | 1.181637 | -0.187602 | -0.04 |
| **2016 Estimate** | -1.271258 | -0.405702 | 0.044075 | -1.070588 | -0.509641 | -0.688368 | -1.484633 | 1.143864 | -0.203669 | -0.05 |
| **2017 Estimate** | -1.281081 | -0.545036 | -0.101496 | -1.052901 | -0.464720 | -0.712416 | -1.433871 | 0.888518 | -0.044150 | 0.18 |
| **2018 Estimate** | -1.227581 | -0.657134 | -0.112126 | -1.032128 | -0.453522 | -0.727730 | -1.385962 | 0.894634 | -0.205999 | 0.18 |
| **2019 Estimate** | -1.273280 | -0.662320 | 0.020065 | -1.140858 | -0.371587 | -0.725558 | -1.400778 | 1.211095 | -0.281041 | 0.16 |
| **2020 Estimate** | -1.289440 | -0.723668 | -0.010780 | -1.032168 | -0.357577 | -0.707525 | -1.405847 | 1.420243 | -0.250072 | 0.14 |
| **2021 Estimate** | -1.257897 | -0.753424 | 0.022103 | -0.998557 | -0.355093 | -0.668003 | -1.477722 | 1.633352 | -0.650570 | 0.25 |

23 rows × 50 columns

Построим график индекса CPI за 1996-2021 для стран SSA и выделим страны с наибольшим и наименьшим значением CPI за 2021 год, а также отобразим среднее значение по региону и РФ.

In [13]:
```
plt.figure("CPI",figsize=[16, 16])
plt.plot(df2_ssa_transposed,"o-", color = 'grey')
plt.plot(df2_ssa_transposed['RUS'],'o-', label="RUS", color = 'yellow', linewidth=5)
plt.plot(df2_ssa_transposed['Среднее значение'],'o-', label="MEAN", color = 'blue', linewidth
plt.plot(df2_ssa_transposed[min_cpi],'o-', label="MIN", color = 'red', linewidth=5)
plt.plot(df2_ssa_transposed[max_cpi],'o-', label="MAX", color = 'green', linewidth=5)
plt.legend(loc=1, title="Легенда", fontsize=12)
plt.xticks(rotation=45)
plt.grid(True, axis="y")
```




Как видно на графике - показатели России - ниже чем стран Африки, а на Сейшелах в 2016 - данных нет.

## Пункт 11

Определим, как изменилось значение показателя в процентах с 1996 по 2021 для максимального, минимального, среднего значения и России. Использовать будем для этого df - так как он включает в себя все страны всех регионов.

In [14]:
```
#найдём минимальное значение на 2021 год и минимальное значение этой же страны на ближайший г
min_2021 = df.loc[df["2021 Estimate"].idxmin(), "2021 Estimate"]
min_1996 = df.loc[df["2021 Estimate"].idxmin(), "2010 Estimate"]
#посчитаем изменение значений в %
change_min = (min_2021 - min_1996) / abs(min_1996) * 100
#найдём максимальное значение на 2021 год и максимальное значение этой же страны на 1996 год
max_2021 = df.loc[df["2021 Estimate"].idxmax(), "2021 Estimate"]
max_1996 = df.loc[df["2021 Estimate"].idxmax(), "1996 Estimate"]
#посчитаем изменение значений в %
change_max = (max_2021 - max_1996) / abs(max_1996) * 100
#найдём среднее значение на 2021 год и среднее значение 1996 год
mean_2021 = df["2021 Estimate"].mean()
mean_1996 = df["1996 Estimate"].mean()
#посчитаем изменение значений в %
change_mean = (mean_2021 - mean_1996) / abs(mean_1996) * 100
#найдём значение России на 2021 год и значение России на 1996 год
rus_1996 = df.loc[df.index[df['Country/Territory']=='Russian Federation'][0],"1996 Estimate"]
rus_2021 = df.loc[df.index[df['Country/Territory']=='Russian Federation'][0],"2021 Estimate"]
#посчитаем изменение значений в %
change_rus = (rus_2021 - rus_1996) / abs(rus_1996) * 100
print("Изменение значения WGI score в России составило: " + str(change_rus) + '%.')
print("Изменение минимального значения WGI score составило: " + str(change_min) + '%.')
print("Изменение максимального значения WGI score составило: " + str(change_max) + '%.')
print("Изменение среднего значения WGI score составило: " + str(change_mean) + '%.')
```

```
Изменение значения WGI score в России составило: 14.512662303490126%.
Изменение минимального значения WGI score составило: -121.78705519027898%.
Изменение максимального значения WGI score составило: 6.0443108848329015%.
Изменение среднего значения WGI score составило: 42.2565999663696%.
```

In [15]:
```
#для более удобного пользования, создадим df3 в котором при помощи left join(merge) добавим в
df3 = pd.read_excel('regions.xlsx')
df = df.merge(df3, how='left')
df
```

Out[15]:

| | Country/Territory | Code | 1996 Estimate | 1996 Rank | 1998 Estimate | 1998 Rank | 2000 Estimate | 2000 Rank | 2002 Estimate | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Zimbabwe | ZWE | -0.278847 | 47.849461 | -0.504802 | 37.433155 | -1.127275 | 10.638298 | -1.156760 | 7.4 |
| **1** | Zambia | ZMB | -0.840641 | 24.731182 | -0.853156 | 26.737968 | -0.818261 | 26.595745 | -0.758519 | 29.6 |
| **2** | Congo, Dem. Rep. | ZAR | -1.647852 | 0.000000 | -1.416679 | 1.069519 | -1.459175 | 0.531915 | -1.449971 | 1.0 |
| **3** | South Africa | ZAF | 0.732927 | 76.344086 | 0.638809 | 72.727272 | 0.550270 | 70.212769 | 0.332902 | 67.1 |
| **4** | Serbia | SRB | -1.140072 | 11.827957 | -1.195605 | 6.417112 | -1.156671 | 9.042553 | -0.895785 | 23.2 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **209** | Anguilla | AIA | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **210** | Angola | AGO | -1.167702 | 9.677420 | -1.180451 | 7.486631 | -1.197514 | 8.510638 | -1.155493 | 7.9 |
| **211** | Afghanistan | AFG | -1.291705 | 4.301075 | -1.176012 | 8.021390 | -1.271724 | 4.787234 | -1.251137 | 4.7 |
| **212** | Andorra | ADO | 1.318143 | 87.096771 | 1.334759 | 89.304810 | 1.313404 | 88.297874 | 1.310744 | 87.8 |
| **213** | Aruba | ABW | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |

214 rows × 50 columns

## Пункт 12

Выведу таблицу из предыдущего пункта

In [16]:
```
data = {'Регион' : ['-',df.loc[df["2021 Estimate"].idxmax(), "Region"],
                     df.loc[df["2021 Estimate"].idxmin(), "Region"],
                     df.loc[df['Country/Territory'] == "Russian Federation", "Region"].values[
                    ],
        "Страна" : ['-',df.loc[df["2021 Estimate"].idxmax(), "Country/Territory"],
                     df.loc[df["2021 Estimate"].idxmin(), "Country/Territory"],
                     df.loc[df['Country/Territory'] == "Russian Federation", "Country/Territor
                    ],
        'CPI 1996' : [mean_1996,max_1996,min_1996,rus_1996],
        "CPI 2021" : [mean_2021,max_2021,min_2021,rus_2021],
        "Изменение, %" : [change_mean, change_max, change_min, change_rus]}
df_table = pd.DataFrame(data, index=['mean_2021', 'max_2021', 'min_2021', 'Russia_2021'])
df_table
```

Out[16]:

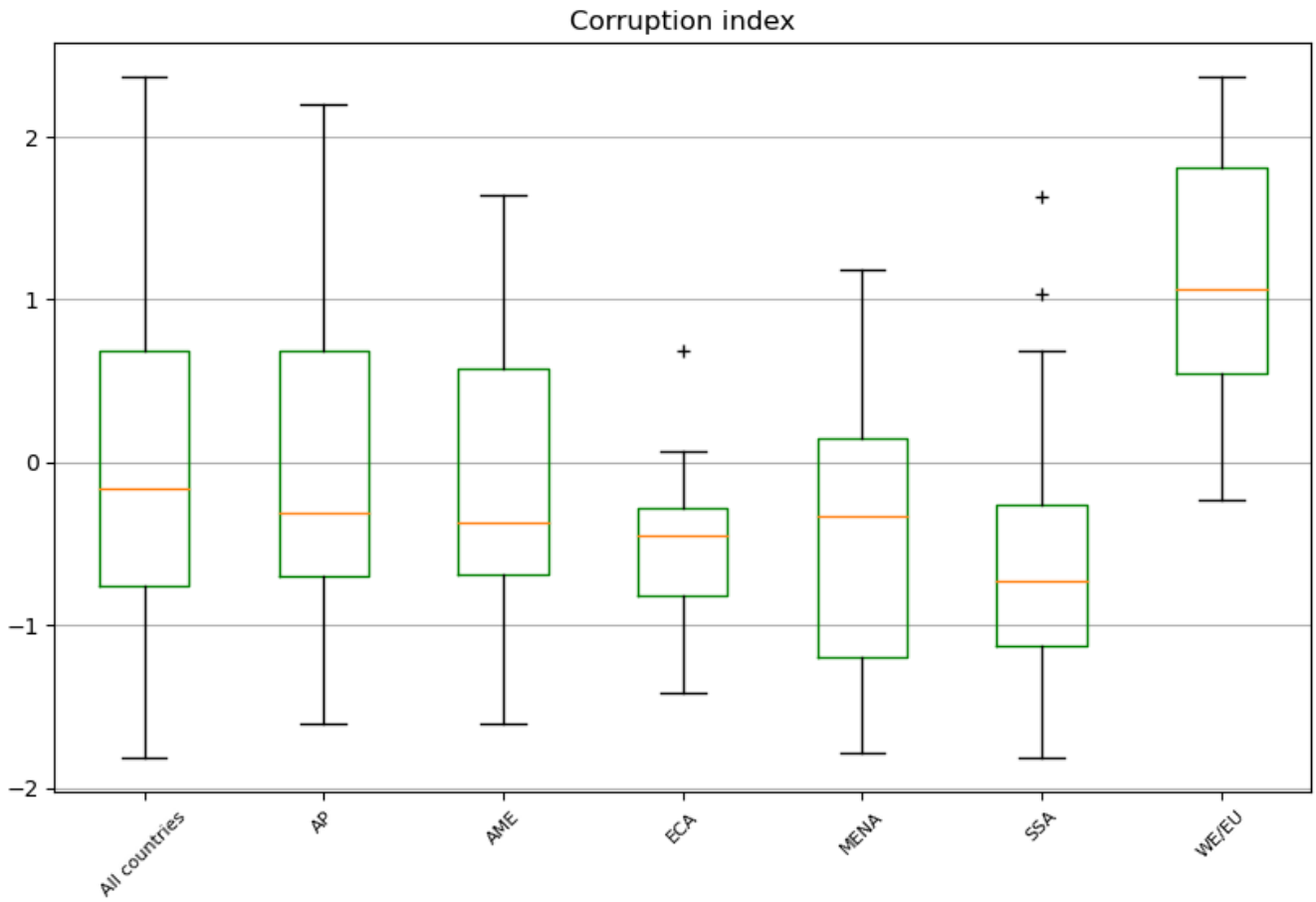
| | Регион | Страна | CPI 1996 | CPI 2021 | Изменение, % |
|---|---|---|---|---|---|
| **mean_2021** | - | - | -1.558845e-09 | -9.001300e-10 | 42.256600 |
| **max_2021** | WE/EU | Denmark | 2.231308e+00 | 2.366175e+00 | 6.044311 |
| **min_2021** | SSA | South Sudan | -8.194555e-01 | -1.817446e+00 | -121.787055 |
| **Russia_2021** | ECA | Russian Federation | -1.053342e+00 | -9.004737e-01 | 14.512662 |

## Пункт 13

Отображу диаграмму размаха индекса коррупции для всех стран и каждого региона в отдельности на одном графике

In [17]:
```
fig, ax1 = plt.subplots(figsize=(10, 6))
names = ['All countries', 'AP', 'AME', 'ECA', 'MENA', 'SSA', 'WE/EU']
data = [df['2021 Estimate'].dropna(),df.loc[df['Region'] == "AP"]['2021 Estimate'],
       df.loc[df['Region'] == "AME"]['2021 Estimate'], df.loc[df['Region'] == "ECA"]['2021 Es
       df.loc[df['Region'] == "MENA"]['2021 Estimate'], df.loc[df['Region'] == "SSA"]['2021 E
       df.loc[df['Region'] == "WE/EU"]['2021 Estimate']]
bp = ax1.boxplot(data, notch=False, sym='+', vert=True, whis=1.5)
plt.setp(bp['boxes'], color='green')
ax1.set_xticklabels(names, rotation=45, fontsize=8)
ax1.set_title('Corruption index')
plt.grid(True, axis="y")
plt.show
```

Out[17]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```




# Задание 2

## Пункт 1

Загрузим данные в один dataframe из всех файлов в папке /data/stock. Будем использовать только колонки Date и Close, т.к. только они нас интересуют, а после добавления соответсвующих колонок по названию удалим колонку Close. Также удалим две строки "2015-07-06" и "2016-03-02", можем считать их выбросами, т.к. на данные даты имеется стоимость только для компании BIDU и отсортируем в хронологическом расположении дат.

In [18]:
```
import glob
import os

#создадим переменную с наименованием всех .csv файлов в директории
path = r'C:\Users\User\Documents\Assignments-master\data\A1_Descriptive_Analysis\stock'
csv_files = glob.glob(path + "/*.csv")

l = []
#прочитаем каждый файл и создадим соответствущий dataframe, который добавим в список l
for filename in csv_files:
    df_csv = pd.read_csv(filename, index_col="Date", header=0, usecols = ["Date","Close"])
    df_csv[os.path.splitext(os.path.basename(filename))[0]] = df_csv["Close"]
    df_csv.drop('Close',inplace=True, axis=1)
    l.append(df_csv)
#объединим все датафреймы из списка l в один - frame
frame = pd.concat(l, axis=1)
frame = frame.sort_values("Date").drop("2015-07-06",axis=0).drop("2016-03-02",axis=0)
frame
```

Out[18]:

| Date | ADBE | ADSK | AKAM | BABA | BIDU | CA | CHKP | CRM | CTSH | |
|---|---|---|---|---|---|---|---|---|---|---|
| 2015-03-02 | 73.940002 | 58.639999 | 71.050003 | 83.239998 | NaN | 32.610001 | 81.970001 | 66.809998 | 62.389999 | 5 |
| 2015-04-01 | 76.059998 | 56.830002 | 73.779999 | 81.290001 | NaN | 31.770000 | 83.480003 | 72.820000 | 58.540001 | 5 |
| 2015-05-01 | 79.089996 | 54.150002 | 76.269997 | 89.320000 | NaN | 30.450001 | 84.720001 | 72.750000 | 64.720001 | 6 |
| 2015-06-01 | 81.010002 | 50.080002 | 69.820000 | 82.269997 | NaN | 29.290001 | 79.550003 | 69.629997 | 61.090000 | 6 |
| 2015-07-01 | 81.989998 | 50.580002 | 76.709999 | 78.339996 | NaN | 29.139999 | 80.769997 | 73.300003 | 63.099998 | 2 |
| 2015-08-03 | 78.570000 | 46.750000 | 71.309998 | 66.120003 | NaN | 27.290001 | 78.010002 | 69.360001 | 62.939999 | 2 |
| 2015-09-01 | 82.220001 | 44.139999 | 69.059998 | 58.970001 | NaN | 27.299999 | 79.330002 | 69.430000 | 62.610001 | 2 |
| 2015-10-01 | 88.660004 | 55.189999 | 60.820000 | 83.830002 | NaN | 27.709999 | 84.940002 | 77.709999 | 68.110001 | 2 |
| 2015-11-02 | 91.459999 | 63.470001 | 57.610001 | 84.080002 | NaN | 28.110001 | 87.290001 | 79.690002 | 64.580002 | 2 |
| 2015-12-01 | 93.940002 | 60.930000 | 52.630001 | 81.269997 | NaN | 28.559999 | 81.379997 | 78.400002 | 60.020000 | 2 |
| 2016-01-04 | 89.129997 | 46.820000 | 45.619999 | 67.029999 | NaN | 28.730000 | 78.809998 | 68.059998 | 63.310001 | 2 |
| 2016-02-01 | 85.150002 | 51.740002 | 53.970001 | 68.809998 | NaN | 29.290001 | 83.070000 | 67.750000 | 56.980000 | 2 |
| 2016-03-01 | 93.800003 | 58.310001 | 55.570000 | 79.029999 | NaN | 30.790001 | 87.470001 | 73.830002 | 62.700001 | 2 |
| 2016-04-01 | 94.220001 | 59.820000 | 50.990002 | 76.940002 | 194.300003 | 29.660000 | 82.870003 | 75.800003 | 58.369999 | 2 |
| 2016-05-02 | 99.470001 | 58.270000 | 54.580002 | 82.000000 | 178.539993 | 32.320000 | 84.970001 | 83.709999 | 61.439999 | 2 |
| 2016-06-01 | 95.790001 | 54.139999 | 55.930000 | 79.529999 | 165.149994 | 32.830002 | 79.680000 | 79.410004 | 57.240002 | 2 |
| 2016-07-01 | 97.860001 | 59.450001 | 50.529999 | 82.480003 | 159.600006 | 34.650002 | 76.879997 | 81.800003 | 57.490002 | 3 |
| 2016-08-01 | 102.309998 | 67.400002 | 54.900002 | 97.190002 | 171.070007 | 33.910000 | 76.739998 | 79.419998 | 57.439999 | 3 |
| 2016-09-01 | 108.540001 | 72.330002 | 52.990002 | 105.790001 | 182.070007 | 33.080002 | 77.610001 | 71.330002 | 47.709999 | 3 |
| 2016-10-03 | 107.510002 | 72.279999 | 69.470001 | 101.690002 | 176.860001 | 30.740000 | 84.559998 | 75.160004 | 51.349998 | 2 |
| 2016-11-01 | 102.809998 | 72.610001 | 66.699997 | 94.019997 | 166.949997 | 31.959999 | 82.330002 | 72.000000 | 55.080002 | 2 |
| 2016-12-01 | 102.949997 | 74.010002 | 66.680000 | 87.809998 | 164.410004 | 31.770000 | 84.459999 | 68.459999 | 56.029999 | 2 |
| 2017-01-03 | 113.379997 | 81.339996 | 68.589996 | 101.309998 | 175.070007 | 31.270000 | 98.769997 | 79.099998 | 52.590000 | 3 |
| 2017-02-01 | 118.339996 | 86.300003 | 62.599998 | 102.900002 | 174.130005 | 32.270000 | 98.910004 | 81.349998 | 59.270000 | 3 |

24 rows × 23 columns

## Пункт 2

Рассчитаем корреляцию данного DF

In [19]:
```
corr_matrix = frame.corr()
corr_matrix
```

Out[19]:

| | ADBE | ADSK | AKAM | BABA | BIDU | CA | CHKP | CRM | CTSH | E |
|---|---|---|---|---|---|---|---|---|---|---|
| **ADBE** | 1.000000 | 0.853387 | -0.351316 | 0.714708 | 0.032627 | 0.470015 | 0.472383 | 0.534918 | -0.639420 | -0.438 |
| **ADSK** | 0.853387 | 1.000000 | -0.018722 | 0.873751 | 0.016605 | 0.527015 | 0.625147 | 0.387552 | -0.617152 | -0.053 |
| **AKAM** | -0.351316 | -0.018722 | 1.000000 | 0.087536 | -0.215394 | -0.161597 | 0.183522 | -0.332294 | 0.121645 | 0.562 |
| **BABA** | 0.714708 | 0.873751 | 0.087536 | 1.000000 | 0.056031 | 0.594163 | 0.429195 | 0.387174 | -0.589198 | 0.198 |
| **BIDU** | 0.032627 | 0.016605 | -0.215394 | 0.056031 | 1.000000 | -0.652263 | 0.159641 | -0.046956 | -0.110644 | -0.190 |
| **CA** | 0.470015 | 0.527015 | -0.161597 | 0.594163 | -0.652263 | 1.000000 | 0.019126 | 0.335133 | -0.556713 | 0.176 |
| **CHKP** | 0.472383 | 0.625147 | 0.183522 | 0.429195 | 0.159641 | 0.019126 | 1.000000 | 0.349589 | -0.044107 | 0.010 |
| **CRM** | 0.534918 | 0.387552 | -0.332294 | 0.387174 | -0.046956 | 0.335133 | 0.349589 | 1.000000 | -0.032611 | -0.264 |
| **CTSH** | -0.639420 | -0.617152 | 0.121645 | -0.589198 | -0.110644 | -0.556713 | -0.044107 | -0.032611 | 1.000000 | 0.125 |
| **EBAY** | -0.438998 | -0.053116 | 0.562100 | 0.198884 | -0.190022 | 0.176644 | 0.010033 | -0.264918 | 0.125003 | 1.000 |
| **FB** | 0.945377 | 0.710084 | -0.546147 | 0.566607 | 0.160983 | 0.471026 | 0.315623 | 0.518872 | -0.618488 | -0.591 |
| **GOOGL** | 0.882737 | 0.655533 | -0.567284 | 0.452358 | -0.203350 | 0.258513 | 0.317104 | 0.490075 | -0.457821 | -0.714 |
| **INFY** | -0.644775 | -0.322194 | 0.377319 | -0.168569 | 0.378583 | 0.085652 | -0.083422 | -0.280892 | 0.361764 | 0.703 |
| **INTU** | 0.805904 | 0.832712 | -0.034693 | 0.800199 | -0.461108 | 0.718292 | 0.509061 | 0.492148 | -0.546619 | -0.029 |
| **MSFT** | 0.918658 | 0.829889 | -0.341101 | 0.661599 | -0.318064 | 0.403205 | 0.496578 | 0.453572 | -0.555838 | -0.406 |
| **NTES** | 0.945812 | 0.880215 | -0.169440 | 0.806564 | -0.228890 | 0.525692 | 0.460174 | 0.484870 | -0.665497 | -0.226 |
| **NVDA** | 0.887010 | 0.892613 | -0.054151 | 0.699008 | -0.335238 | 0.517295 | 0.504884 | 0.286524 | -0.667971 | -0.249 |
| **PYPL** | 0.815057 | 0.798648 | 0.015839 | 0.783197 | 0.389159 | 0.563574 | 0.459489 | 0.148632 | -0.666766 | 0.451 |
| **RHT** | 0.295120 | 0.422972 | 0.123779 | 0.556812 | 0.226945 | 0.065133 | 0.310665 | 0.493769 | 0.017098 | 0.161 |
| **SAP** | 0.903788 | 0.874067 | -0.365017 | 0.805775 | -0.055597 | 0.609354 | 0.430469 | 0.484253 | -0.625570 | -0.217 |
| **TWTR** | -0.744648 | -0.374832 | 0.634183 | -0.139319 | -0.027082 | -0.177609 | -0.190570 | -0.437294 | 0.392820 | 0.801 |
| **VMW** | 0.007322 | 0.317948 | 0.827930 | 0.411332 | -0.287513 | 0.216074 | 0.220590 | -0.103050 | -0.179236 | 0.587 |
| **YHOO** | 0.394537 | 0.690366 | 0.337245 | 0.845418 | 0.017444 | 0.719633 | 0.386979 | 0.231736 | -0.445205 | 0.543 |

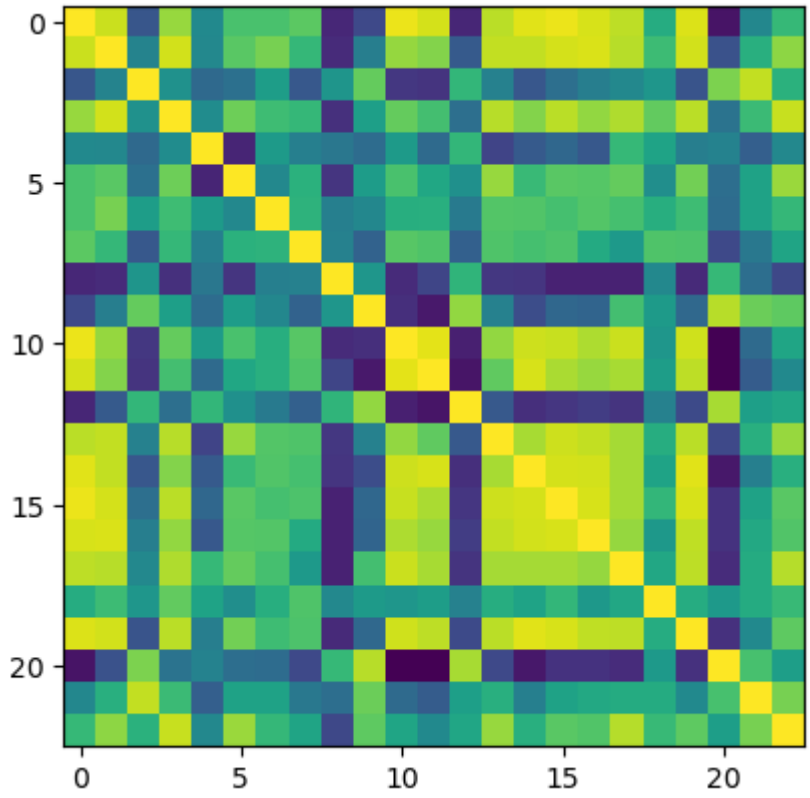
23 rows × 23 columns

## Пункт 3

Отобразим корреляционную матрицу в виде диаграммы.

In [20]:
```
plt.imshow(corr_matrix)
```

Out[20]:
```
<matplotlib.image.AxesImage at 0x233499a9400>
```




## Пункт 4

Определим:

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

в соотношении с акциями компании Facebook (FB)

In [21]:
```
so = corr_matrix["FB"].sort_values(kind="quicksort")
so1 = corr_matrix["FB"].sort_values(kind="quicksort", key=abs)
print("Максимальная положительная корреляция с компанией "+so.index[-2]+ ' и составляет '+str
print("Максимальная отрицательная корреляция с компанией "+so.index[0]+ ' и составляет '+str(
print("Минимальная корреляция с компанией "+so1.index[0]+ ' и составляет '+str(so1[0]))
```

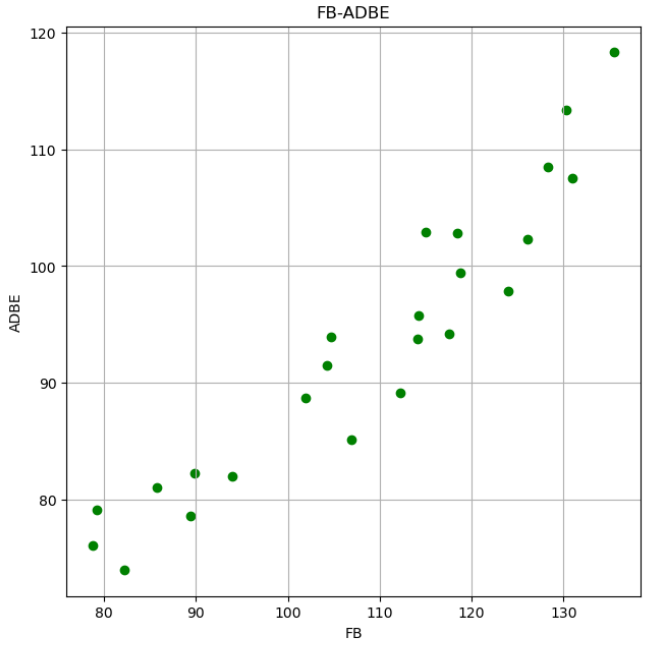
```
Максимальная положительная корреляция с компанией ADBE и составляет 0.9453771902371849
Максимальная отрицательная корреляция с компанией TWTR и составляет -0.8425772000601955
Минимальная корреляция с компанией RHT и составляет 0.12702014393032882
```

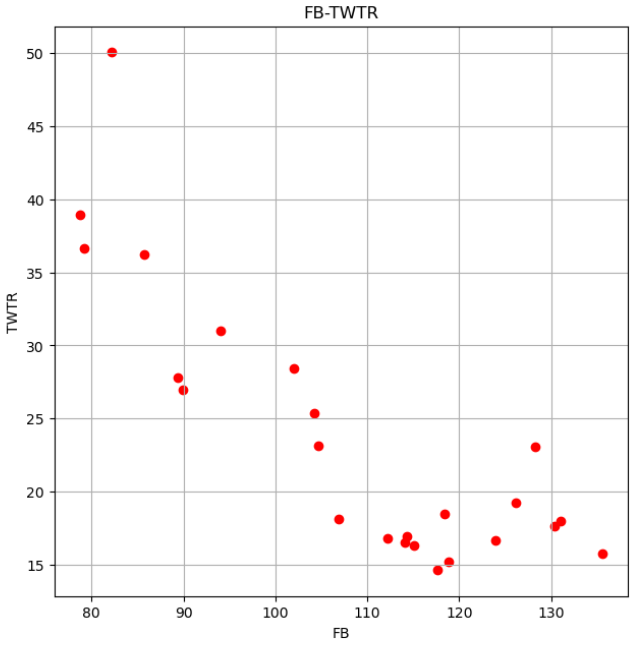
## Пункт 5

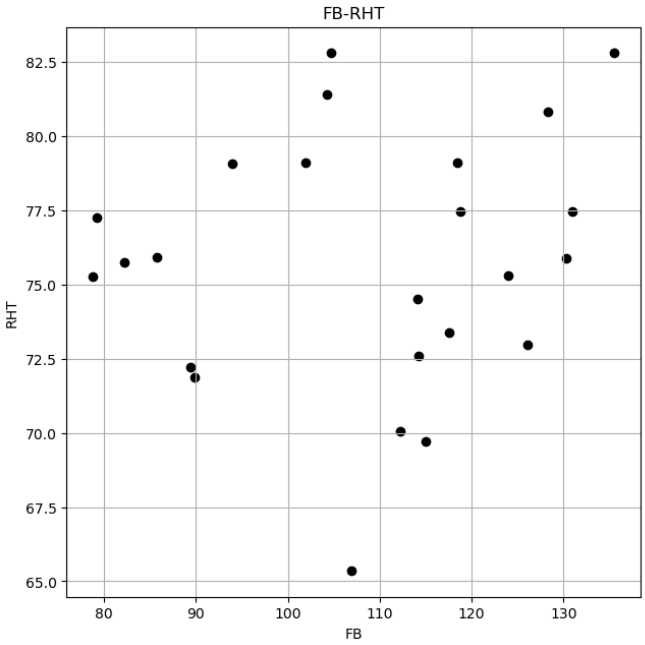
Построим диаграммы разброса

In [22]:
```
plt.figure(figsize=[16, 16])

plt.subplot(2,2,1)
plt.title("FB-ADBE")
plt.scatter(frame['FB'], frame[so.index[-2]], color = 'green')
plt.xlabel("FB")
plt.ylabel("ADBE")
plt.grid(True)


plt.subplot(2,2,2)
plt.title("FB-TWTR")
plt.scatter(frame['FB'], frame[so.index[0]], color = 'red')
```

```
plt.xlabel("FB")
plt.ylabel("TWTR")
plt.grid(True)


plt.subplot(2,2,3)
plt.title("FB-RHT")
plt.scatter(frame['FB'], frame[so1.index[0]], color = 'black')
plt.xlabel("FB")
plt.ylabel("RHT")
plt.grid(True)



plt.show()
```




## Пункт 6

Найдем среднее значение для каждого месяца

In [23]:
```
frame['Mean value'] = frame.mean(axis=1)
frame
```

Out[23]:

| Date | ADBE | ADSK | AKAM | BABA | BIDU | CA | CHKP | CRM | CTSH | |
|---|---|---|---|---|---|---|---|---|---|---|
| 2015-03-02 | 73.940002 | 58.639999 | 71.050003 | 83.239998 | NaN | 32.610001 | 81.970001 | 66.809998 | 62.389999 | 5 |
| 2015-04-01 | 76.059998 | 56.830002 | 73.779999 | 81.290001 | NaN | 31.770000 | 83.480003 | 72.820000 | 58.540001 | 5 |
| 2015-05-01 | 79.089996 | 54.150002 | 76.269997 | 89.320000 | NaN | 30.450001 | 84.720001 | 72.750000 | 64.720001 | 6 |
| 2015-06-01 | 81.010002 | 50.080002 | 69.820000 | 82.269997 | NaN | 29.290001 | 79.550003 | 69.629997 | 61.090000 | 6 |
| 2015-07-01 | 81.989998 | 50.580002 | 76.709999 | 78.339996 | NaN | 29.139999 | 80.769997 | 73.300003 | 63.099998 | 2 |
| 2015-08-03 | 78.570000 | 46.750000 | 71.309998 | 66.120003 | NaN | 27.290001 | 78.010002 | 69.360001 | 62.939999 | 2 |
| 2015-09-01 | 82.220001 | 44.139999 | 69.059998 | 58.970001 | NaN | 27.299999 | 79.330002 | 69.430000 | 62.610001 | 2 |
| 2015-10-01 | 88.660004 | 55.189999 | 60.820000 | 83.830002 | NaN | 27.709999 | 84.940002 | 77.709999 | 68.110001 | 2 |
| 2015-11-02 | 91.459999 | 63.470001 | 57.610001 | 84.080002 | NaN | 28.110001 | 87.290001 | 79.690002 | 64.580002 | 2 |
| 2015-12-01 | 93.940002 | 60.930000 | 52.630001 | 81.269997 | NaN | 28.559999 | 81.379997 | 78.400002 | 60.020000 | 2 |
| 2016-01-04 | 89.129997 | 46.820000 | 45.619999 | 67.029999 | NaN | 28.730000 | 78.809998 | 68.059998 | 63.310001 | 2 |
| 2016-02-01 | 85.150002 | 51.740002 | 53.970001 | 68.809998 | NaN | 29.290001 | 83.070000 | 67.750000 | 56.980000 | 2 |
| 2016-03-01 | 93.800003 | 58.310001 | 55.570000 | 79.029999 | NaN | 30.790001 | 87.470001 | 73.830002 | 62.700001 | 2 |
| 2016-04-01 | 94.220001 | 59.820000 | 50.990002 | 76.940002 | 194.300003 | 29.660000 | 82.870003 | 75.800003 | 58.369999 | 2 |
| 2016-05-02 | 99.470001 | 58.270000 | 54.580002 | 82.000000 | 178.539993 | 32.320000 | 84.970001 | 83.709999 | 61.439999 | 2 |
| 2016-06-01 | 95.790001 | 54.139999 | 55.930000 | 79.529999 | 165.149994 | 32.830002 | 79.680000 | 79.410004 | 57.240002 | 2 |
| 2016-07-01 | 97.860001 | 59.450001 | 50.529999 | 82.480003 | 159.600006 | 34.650002 | 76.879997 | 81.800003 | 57.490002 | 3 |
| 2016-08-01 | 102.309998 | 67.400002 | 54.900002 | 97.190002 | 171.070007 | 33.910000 | 76.739998 | 79.419998 | 57.439999 | 3 |
| 2016-09-01 | 108.540001 | 72.330002 | 52.990002 | 105.790001 | 182.070007 | 33.080002 | 77.610001 | 71.330002 | 47.709999 | 3 |
| 2016-10-03 | 107.510002 | 72.279999 | 69.470001 | 101.690002 | 176.860001 | 30.740000 | 84.559998 | 75.160004 | 51.349998 | 2 |
| 2016-11-01 | 102.809998 | 72.610001 | 66.699997 | 94.019997 | 166.949997 | 31.959999 | 82.330002 | 72.000000 | 55.080002 | 2 |
| 2016-12-01 | 102.949997 | 74.010002 | 66.680000 | 87.809998 | 164.410004 | 31.770000 | 84.459999 | 68.459999 | 56.029999 | 2 |
| 2017-01-03 | 113.379997 | 81.339996 | 68.589996 | 101.309998 | 175.070007 | 31.270000 | 98.769997 | 79.099998 | 52.590000 | 3 |

| | ADBE | ADSK | AKAM | BABA | BIDU | CA | CHKP | CRM | CTSH |
|---|---|---|---|---|---|---|---|---|---|
| **Date** | | | | | | | | | |
| **2017-02-01** | 118.339996 | 86.300003 | 62.599998 | 102.900002 | 174.130005 | 32.270000 | 98.910004 | 81.349998 | 59.270000 |

## Пункт 7

Построим график для упомянутых компания и среднего значения по таблице выше

In [24]:
```
plt.figure(figsize=[12, 6])

plt.title('Timelapse statistics')
plt.plot(frame.index, frame['FB'] , "-o", label="Facebook")
plt.plot(frame.index, frame['RHT'] , "-o", label="RHT - none")
plt.plot(frame.index, frame['ADBE'] , "-o", label="ADBE - max")
plt.plot(frame.index, frame['TWTR'] , "-o", label="Twitter - min")
plt.plot(frame.index, frame['Mean value'] , "-o", label="Mean")
plt.grid(True)
plt.legend(loc=2, title="Legend", fontsize=12)
plt.xlabel("Date")
plt.xticks(fontsize=8, rotation=90)
plt.ylabel("Price", fontsize=12)

plt.show()
```