Департамент образования и науки города Москвы Государственное автономное образовательное учреждение высшего образования города Москвы «Московский городской педагогический университет» Институт цифрового образования Департамент информатики, управления и технологий

ДИСЦИПЛИНА:

Инструменты для хранения и обработки больших данных

Практическая работа №4.2

Тема:

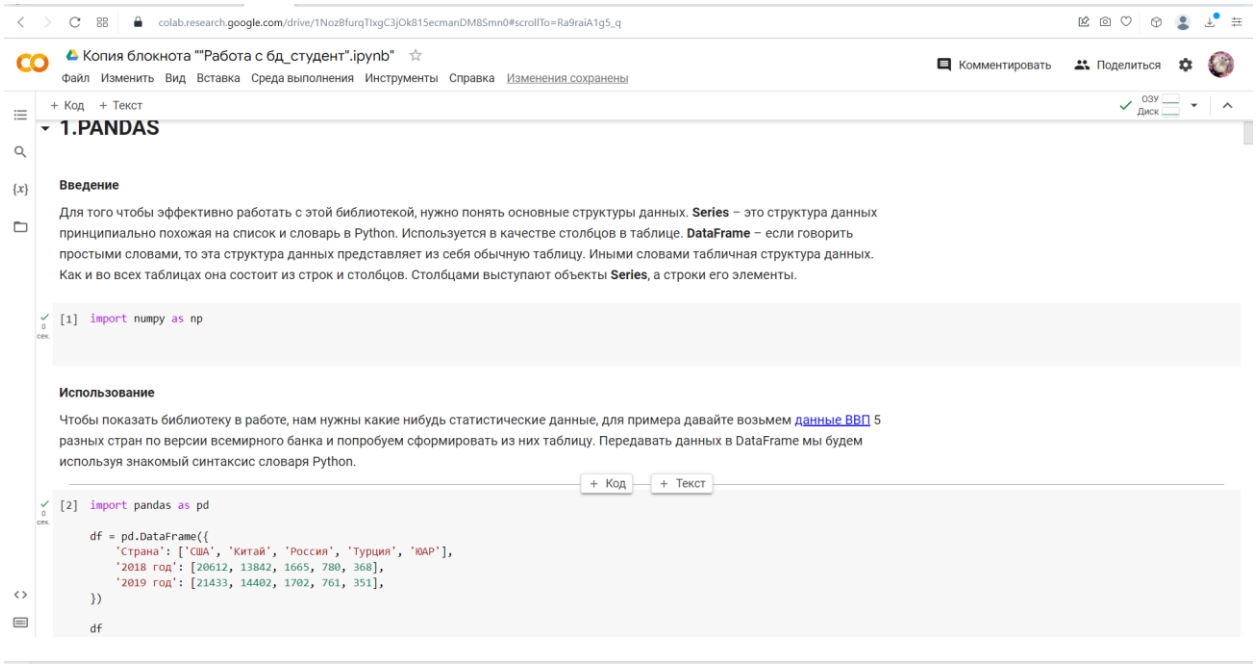
Marketing Analytics

Выполнила: Шепелева Е. В., группа: АДЭУ-201

Преподаватель: Т. М. Босенко

Москва

2022



[2]

0 сек.

	Страна	2018 год	2019 год
0	США	20612	21433
1	Китай	13842	14402
2	Россия	1665	1702
3	Турция	780	761
4	ЮАР	368	351

Объект **DataFrame** имеет два индекса по столбцам и строкам. Если индекс по строкам не указан вручную, то pandas задает его автоматически.

Индексы

Назначать индексы объекту **DataFrame** можно при его создании или в процессе работы с ним.

0 сек.

```
import pandas as pd

df = pd.DataFrame({
    'Страна': ['США', 'Китай', 'Россия', 'Турция', 'ЮАР'],
    '2018 год': [20612, 13842, 1665, 780, 368],
    '2019 год': [21433, 14402, 1702, 761, 351],
}, index=['US', 'CN', 'RU', 'TR', 'ZA']

)

df
```

	Страна	2018 год	2019 год
US	США	20612	21433
CN	Китай	13842	14402
RU	Россия	1665	1702
TR	Турция	780	761
ZA	ЮАР	368	351

Вызывая метод **DataFrame** мы передали ему аргумент **index** со списком именованных индексов.

Фильтрация данных

Pandas позволяет производить фильтрацию вывода по индексам и столбцам. Так же можно комбинировать индексы и колонки, использовать слайсы и логические выражения.

По столбцу

Обращение к столбцам в pandas реализовано стандартным образом, так как будто вы обращаетесь к ключу словаря, или же к методу объекта. В моем случае обращение как к методу объекта невозможно, я выбрал кириллическое название столбца, а работает только с латиницей

```
[4] df["Страна"]
```

✓ 0 сек.

```
[4]  US       США
     CN     Китай
     RU    Россия
     TR    Турция
     ZA       ЮАР
     Name: Страна, dtype: object
```

По строковому индексу

Для обращения к строковым индекса существуют два метода

1. **loc** – для доступа по именованному индексу
2. **iloc** – для доступа по числовому индексу

✓ 0 сек.

```
[5]  df.loc["RU"]
```

```
Страна      Россия
2018 год      1665
2019 год      1702
Name: RU, dtype: object
```

Обращение к именованному индексу **RU**

✓ 0 сек.

```
[6]  df.iloc[0]
```

```
Страна        США
2018 год    20612
2019 год    21433
Name: US, dtype: object
```

Обращение к числовому индексу

По срезами

Объект **DataFrame** поддерживает использование срезов.

```
[7] df[2:]
```

	Страна	2018 год	2019 год
RU	Россия	1665	1702
TR	Турция	780	761
ZA	ЮАР	368	351

Отобразим все строки начиная с 3.

С использованием условий

Мы так же можем использовать логику в фильтрации данных. Давайте отобразить странны, в которых ВВП на душу населения в 2019 году был больше 761$

```
df[df["2019 год"] > 761]['Страна']
```

```
US       США
CN     Китай
RU    Россия
Name: Страна, dtype: object
```

Работа с столбцами

Вы можете создавать, удалять и переименовывать ваши столбцы в любой момент времени.

Переименование

Для переименования столбца существует метод rename Давайте переименуем наши столбцы с указанием года.

```
[9] df.rename(columns={'2018 год': '2018', '2019 год': '2019'})
```

	Страна	2018	2019
US	США	20612	21433
CN	Китай	13842	14402
RU	Россия	1665	1702
TR	Турция	780	761
ZA	ЮАР	368	351

Метод **rename** на вход принимает обычный словарь, ключ который является текущем названием столбца, а значение – новым. За один раз мы можем переименовать сколько угодно столбцов, главное не забывайте разделять элементы словаря запятой.

Важно: результат выполнение метода rename возвращает новый измененный объект **DataFrame**, поэтому переназначь основной экземпляр **DataFrame**.

Создание

Создадим новую колонку “Рост” и наполним ее значениями высчитанными из разницы 2019 к 2018 году.

```
[10] dfr=df.rename(columns={'2018 год': '2018', '2019 год': '2019'})
```

```
[11] dfr["Рост"] = dfr['2019'] - dfr['2018']
     dfr
```

	Страна	2018	2019	Рост
US	США	20612	21433	821
CN	Китай	13842	14402	560
RU	Россия	1665	1702	37
TR	Турция	780	761	-19
ZA	ЮАР	368	351	-17

Удаление

Для удаления столбца существует метод **drop**, так же необходимо передать в аргумент **axis** значение **index** или **columns**.

```
dfr.drop(["Рост"], axis="columns")
```

	Страна	2018	2019
US	США	20612	21433
CN	Китай	13842	14402
RU	Россия	1665	1702

Важно: результат выполнение метода **drop** возвращает новый измененный объект **DataFrame**, поэтому не забудьте переназначить **DataFrame**.

```
dfr
```

	Страна	2018	2019	Рост
US	США	20612	21433	821
CN	Китай	13842	14402	560
RU	Россия	1665	1702	37
TR	Турция	780	761	-19
ZA	ЮАР	368	351	-17

```
[14] dfr2=dfr.drop(["Рост"], axis="columns")
     dfr2
```

	Страна	2018	2019
US	США	20612	21433
CN	Китай	13842	14402
RU	Россия	1665	1702
TR	Турция	780	761
ZA	ЮАР	368	351

Из таблицы MS Excel

За загрузку данных из **excel** таблицы отвечает метод **read_excel**

```
[18] from google.colab import files
     uploaded = files.upload()
```

```
Выбрать файлы data-632...-04-10.xlsx
```

- **data-6322-2023-04-10.xlsx**(application/vnd.openxmlformats-officedocument.spreadsheetml.sheet) - 116845 bytes, last modified: 10.04.2023 - 100% done

```
Saving data-6322-2023-04-10.xlsx to data-6322-2023-04-10.xlsx
```

```
[19] !ls
```

```
data-6322-2023-04-10.xlsx  data-6322-2023-04-10.xml  sample_data
```

```
[20] data_xlsx = pd.read_excel("data-6322-2023-04-10.xlsx")
```

```
/usr/local/lib/python3.10/dist-packages/openpyxl/styles/stylesheet.py:226: UserWarning: Workbook contains no default style, apply openpyxl's default
  warn("Workbook contains no default style, apply openpyxl's default")
```

```
[21] data_xlsx
```

	ID	FullName	INN	OGRN	AccreditationAuthority	Education	CertificateNumber	CertificateIssueDate	Validity	CertificateFormSeries
0	Код	Полное официальное наименование	ИНН	ОГРН	Наименование аккредитационного органа	Образовательные программы	Номер свидетельства	Дата выдачи свидетельства	Срок действия	Серия бланка свидетельства об аккредитации
1	41	Государственное бюджетное общеобразовательное ...	7701375995	5137746011035	Департамент образования и науки города Москвы	Education:начальное общее образование\n\nEduca...	003991	11.12.2015	бессрочно	77A01

	ID	FullName	INN	OGRN	AccreditationAuthority	Education	CertificateNumber	CertificateIssueDate	Validity	CertificateFormSeries
0	Код	Полное официальное наименование	ИНН	ОГРН	Наименование аккредитационного органа	Образовательные программы	Номер свидетельства	Дата выдачи свидетельства	Срок действия	Серия бланка свидетельства об аккредитации
1	41	Государственное бюджетное общеобразовательное ...	7701375995	5137746011035	Департамент образования и науки города Москвы	Education:начальное общее образование\n\nEduca...	003991	11.12.2015	бессрочно	77A01
2	42	Государственное бюджетное общеобразовательное ...	7708071876	1027700388363	Департамент образования и науки города Москвы	Education:начальное общее образование\n\nEduca...	005069	31.03.2023	бессрочно	77A01
3	43	Государственное бюджетное общеобразовательное ...	7704118139	1027700587672	Департамент образования и науки города Москвы	Education:основное общее образование\n\nEducat...	004773	16.04.2018	бессрочно	77A01
4	44	Государственное бюджетное общеобразовательное ...	7720325492	5157746151921	Департамент образования и науки города Москвы	Education:начальное общее образование\n\nEduca...	004148	12.02.2016	бессрочно	77A01
...	...	...	...	...	...	...	...	...	...	...
901	1733	Государственное бюджетное профессиональное обр...	7716079082	1037739236215	Департамент образования и науки города Москвы	Education:среднее профессиональное образование...	005059	02.02.2023	02.02.2024	77A01
902	1734	ОБЩЕОБРАЗОВАТЕЛЬНОЕ ЧАСТНОЕ УЧРЕЖДЕНИЕ СРЕДНЯЯ...	9715391013	1207700379006	Департамент образования и науки города Москвы	Education:начальное общее образование\n\nEduca...	005068	24.03.2023	бессрочно	77A01
903	1735	АВТОНОМНАЯ НЕКОММЕРЧЕСКАЯ ОБЩЕОБАЗОВАТЕЛЬНАЯ О...	9719003670	1207700168411	Департамент образования и науки города Москвы	Education:начальное общее образование\n\n	005070	03.04.2023	бессрочно	77A01

```
data_xlsx.to_excel("country.xlsx",encoding='cp1251')
```

```
/usr/local/lib/python3.10/dist-packages/pandas/util/_decorators.py:211: FutureWarning: the 'encoding' keyword is deprecated and will be removed in a future version. Please take steps t
  return func(*args, **kwargs)
```

Сохранение данных

Так же как и в импорте API поддерживает множество форматов для экспорта данных. Воспользуемся данными о ВВП для демонстрации работы.

В таблицу CSV

За запись данных в таблицу **CSV** отвечает метод **to_csv**

```
import pandas as pd

df = pd.DataFrame({
    'Страна': ['США', 'Китай', 'Россия', 'Турция', 'ЮАР'],
    '2018 год': [20612, 13842, 1665, 780, 368],
    '2019 год': [21433, 14402, 1702, 761, 351],
}, index=['US', 'CN', 'RU', 'TR', 'ZA']

)

df.to_csv("country.csv",encoding='cp1251')
```

Скачивание файлов в локальную файловую систему Метод *files.download *активирует скачивание файла из браузера на локальный компьютер.

```
[24] from google.colab import files

     files.download('country.csv')
```

В таблицу MS Excel

За запись данных в таблицу **MS Excel** отвечает метод **to_excel**

```
[25] import pandas as pd

     df = pd.DataFrame({
         'Страна': ['США', 'Китай', 'Россия', 'Турция', 'ЮАР'],
         '2018 год': [20612, 13842, 1665, 780, 368],
         '2019 год': [21433, 14402, 1702, 761, 351],
     }, index=['US', 'CN', 'RU', 'TR', 'ZA']

     )

     df.to_excel("country.xlsx",encoding='cp1251')
```

```
/usr/local/lib/python3.10/dist-packages/pandas/util/_decorators.py:211: FutureWarning: the 'encoding' keyword is deprecated and will be removed in a future version. Please take steps
  return func(*args, **kwargs)
```

```
from google.colab import files

files.download('country.xlsx')
```

Визуализация данных

Визуализация это большая часть работы в анализе и обработке данных. Не будем сильно углубляться и рассмотрим простой пример визуализации наших данных.

Установка библиотеки **matplotlib**

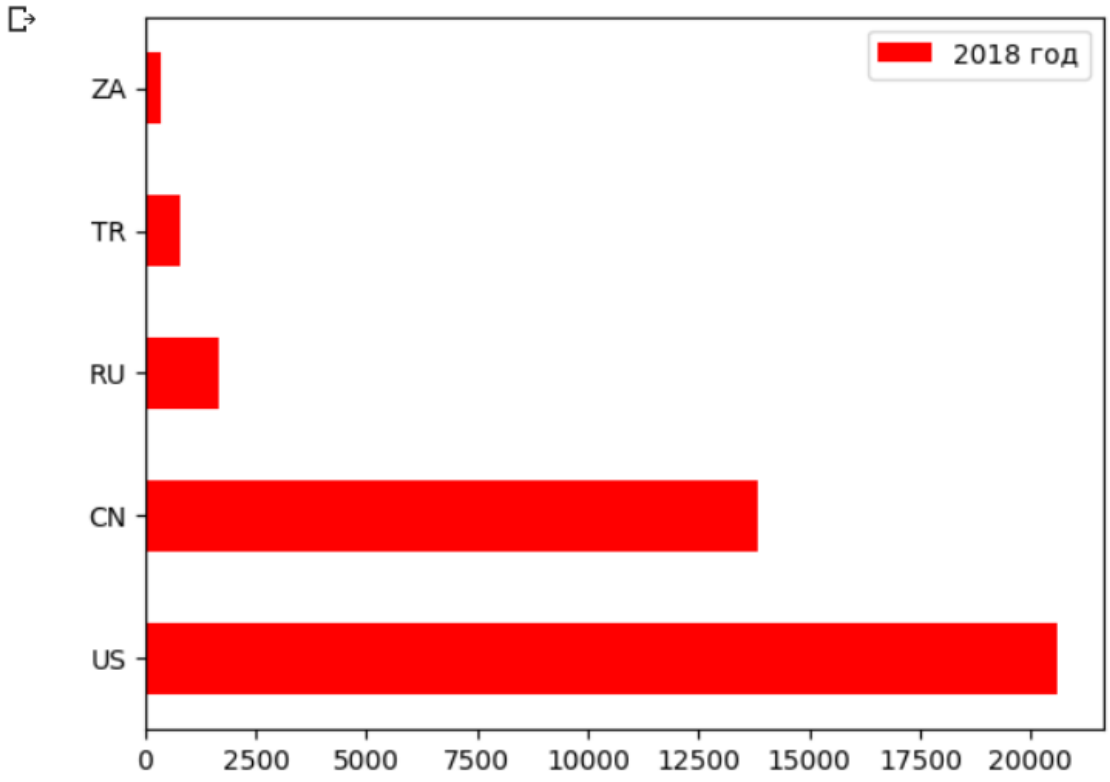
Для рисования графиков нам понадобится эта библиотека

```
[27] import matplotlib.pyplot as plt
```

Создание графиков

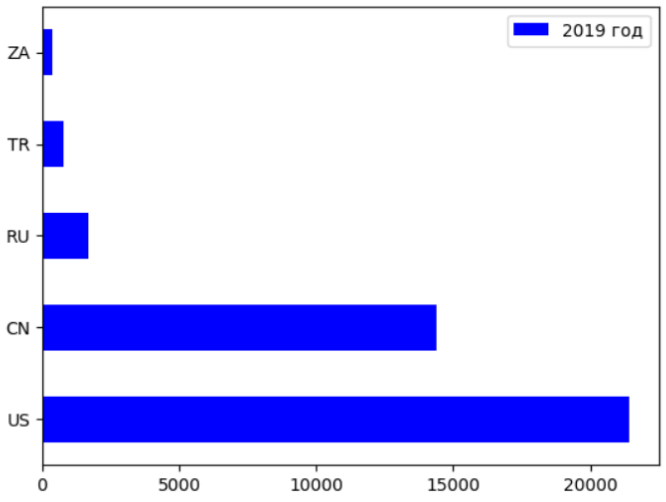
Самый просто способ сгенерировать график, это передать обработчику данные для одной из координат, для второй он возьмет информацию из индекса.

```
df.plot(kind='barh', y='2018 год', color='red')
plt.show()
```




```
[29] df.plot(kind='barh', y='2019 год', color='blue')
     plt.show()
```

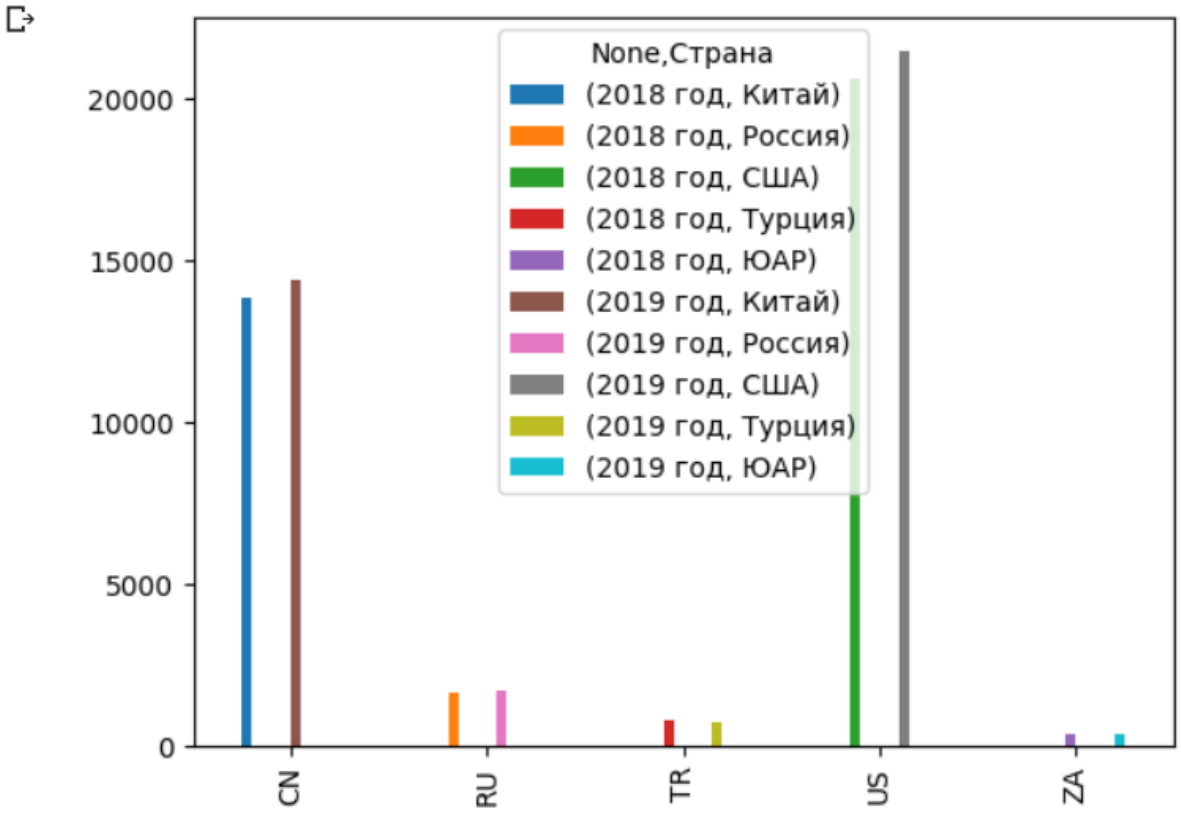

```
[29] df.plot(kind='barh', y='2019 год', color='blue')
     plt.show()
```

Объединение данных на одном графике

У нас есть отдельный график для 2018 и 2019 года, но как их объединить в одной диаграмме? Очень просто, нужно использовать метод pivot из библиотеки **pandas**.

```
df.pivot(columns="Страна").plot(kind='bar')
plt.show()
```



Автоматизация выбора данных (парсинг)

```
[31] import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
     from unicodedata import normalize
```

ИЗВЛЕЧЕНИЕ ТАБЛИЦ

Вызовем функцию read_html, передав аргументом ссылку на страницу.

```
[32] # Список стран по ВВП (номинал)
     tables = pd.read_html(
         'https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_%D1%81%D1%82%D1%80%D0%B0%D0%BD_%D0%BF%D0%BE_%D0%92%D0%92%D0%9F_(%D0%BD%D0%BE%D0%BC%D0%B8%D0%BD%D0%B0%D0%BB)',
         match='Страна')
```

```
[33] len(tables)
```

```
4
```

```
df = tables[1]
df
```

	№	Страна	2018	2019
0	1	США	20612	21433
1	2	Китай	13842	14402
2	3	Япония	4952	5080
3	4	Германия	3966	3862
4	5	Индия	2713	2869
...	...	...	...	...
188	183	Маршалловы Острова	22	23
189	184	Кирибати	2	2
190	185	Науру	12	12
191	186	Тувалу	5	5
192	NaN	Всего в мире	85690	87552

193 rows × 4 columns

```
[35] df1 = tables[0]
     df1
```

	0	1	2	3	4
0	Список МВФ[1] № Страна 2018 2019 1 США 20612 ...	0	Список ВБ[3] № Страна 2018 2019 1 США 20580 2...	0.0	Список ООН[4] № Страна 2018 1 США 20580 2 Ки...
1	№	Страна	2018	2019.0	NaN
2	1	США	20612	21433.0	NaN
3	2	Китай	13842	14402.0	NaN
4	3	Япония	4952	5080.0	NaN
...	...	...	...	...	...
621	193	Науру	018	NaN	NaN
622	194	Кирибати	018	NaN	NaN
623	—	Монтсеррат (Великобритания)	006	NaN	NaN
624	195	Тувалу	004	NaN	NaN
625	NaN	Всего в мире	75130	NaN	NaN

626 rows × 5 columns

ОБРАБАТЫВАЕМ ТАБЛИЦЫ

В первую очередь избавимся от лишнего столбца, вызвав метод drop.

```
[36] df.drop(('№'), axis=1, inplace=True)
```

```
df
```

	Страна	2018	2019
0	США	20612	21433
1	Китай	13842	14402
2	Япония	4952	5080
3	Германия	3966	3862
4	Индия	2713	2869
...	...	...	...
188	Маршалловы Острова	22	23
189	Кирибати	2	2
190	Науру	12	12
191	Тувалу	5	5
192	Всего в мире	85690	87552

193 rows × 3 columns

```
print(df.to_string())
```

```
135                    Нигер  129  129
136                 Никарагуа  130  125
137                   Намибия  136  125
138         Республика Конго   134  125
139                  Молдавия  113  120
140    Экваториальная Гвинея   136  118
141                      Чад   110  110
142                   Руанда   963  101
143                    Гаити   966  870
144                 Киргизия   827  846
145               Таджикистан   752  812
146        Республика Косово    795  797
147                   Малави   691  767
148                Мавритания   705   76
149                 Мальдивы   532  576
150                     Того   536  546
151                    Фиджи   554  541
152                 Барбадос   509  521
153                   Гайана   479  517
154                Черногория   551   50
155               Южный Судан   466  493
156                  Эсватини   471  459
157              Сьерра-Леоне   409  421
158                  Суринам   347   37
159                  Джибути   301  335
160                  Либерия   326  318
161                  Бурунди   319  311
162       Аруба (Нидерланды)   282  289
163                   Бутан    251   25
164                   Лесото   247  242
165                      ЦАР   228  228
166                Сент-Люсия   207  212
167                  Эритрея   201  198
168                Кабо-Верде   197  198
```

Кроме того, следует убрать источники, заключённые в квадратные скобки. Для этого мы воспользуемся методом replace, указав регулярное выражение и **regex=True**. Теперь таблица выглядит более приемлемо.

```
[39] df.replace({'\[[0-9]+\]': ''}, regex=True, inplace=True)
```

```
print(df.to_string())
```

```
9                    Канада   1710   1750
10                 Бразилия   1885   1839
11         Республика Корея   1725   1647
12                 Испания   1420   1394
13               Австралия   1421   1387
14                 Мексика   1222   1258
15               Индонезия   1042   1120
16             Нидерланды    914    907
17       Саудовская Аравия    787    793
18                 Турция    780    761
19               Швейцария    706    705
20                 Тайвань    608    611
21                  Польша    587    592
22                    Иран    435    583
23                 Таиланд    506    544
24                  Швеция    555    530
25                 Бельгия    543    529
26                 Нигерия    398    448
27                 Австрия    456    446
28               Аргентина    517    444
29                     ОАЭ    422    421
30                Норвегия    434    403
31                Ирландия    387    398
32                 Израиль    370    394
33               Филиппины    347    377
34                Сингапур    373    372
35           Гонконг (КНР)    362    366
```

```
df
```

	Страна	2018	2019
0	США	20612	21433
1	Китай	13842	14402
2	Япония	4952	5080
3	Германия	3966	3862
4	Индия	2713	2869
...	...	...	...
188	Маршалловы Острова	22	23
189	Кирибати	2	2
190	Науру	12	12
191	Тувалу	5	5
192	Всего в мире	85690	87552

193 rows × 3 columns

Теперь отбросим нижний результирующий уровень

```
[42] df.drop(df.index[len(df)-1])
```

[42]

	Страна	2018	2019
0	США	20612	21433
1	Китай	13842	14402
2	Япония	4952	5080
3	Германия	3966	3862
4	Индия	2713	2869
...	...	...	...
187	Палау	29	28
188	Маршалловы Острова	22	23
189	Кирибати	2	2
190	Науру	12	12
191	Тувалу	5	5

192 rows × 3 columns

```
df.to_excel("countryALL.xlsx",encoding='cp1251')
```

```
/usr/local/lib/python3.10/dist-packages/pandas/util/_decorators.py:211: FutureWarning: the 'encoding' keyword is deprecated and will be removed in a future version. Please take steps t
  return func(*args, **kwargs)
```

```
[44] from google.colab import files

     files.download('countryALL.xlsx')
```

Чтобы изменить содержимое ячейки, дважды нажмите на нее (или выберите "Ввод")

```
[48]  !pip install PyMySQL
      !pip install mysql-connector-python
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: PyMySQL in /usr/local/lib/python3.10/dist-packages (1.0.3)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: mysql-connector-python in /usr/local/lib/python3.10/dist-packages (8.0.33)
Requirement already satisfied: protobuf<=3.20.3,>=3.11.0 in /usr/local/lib/python3.10/dist-packages (from mysql-connector-python) (3.20.3)
```

```
[50] !pip install sqlalchemy
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: sqlalchemy in /usr/local/lib/python3.10/dist-packages (2.0.10)
Requirement already satisfied: greenlet!=0.4.17 in /usr/local/lib/python3.10/dist-packages (from sqlalchemy) (2.0.2)
Requirement already satisfied: typing-extensions>=4.2.0 in /usr/local/lib/python3.10/dist-packages (from sqlalchemy) (4.5.0)
```

```
[56] import sqlalchemy
```

Ссылка на colab: https://colab.research.google.com/drive/1NozBfurqTIxgC3jOk815ecmanDM8Smn0?usp=sharing