

Лабораторная работа 2

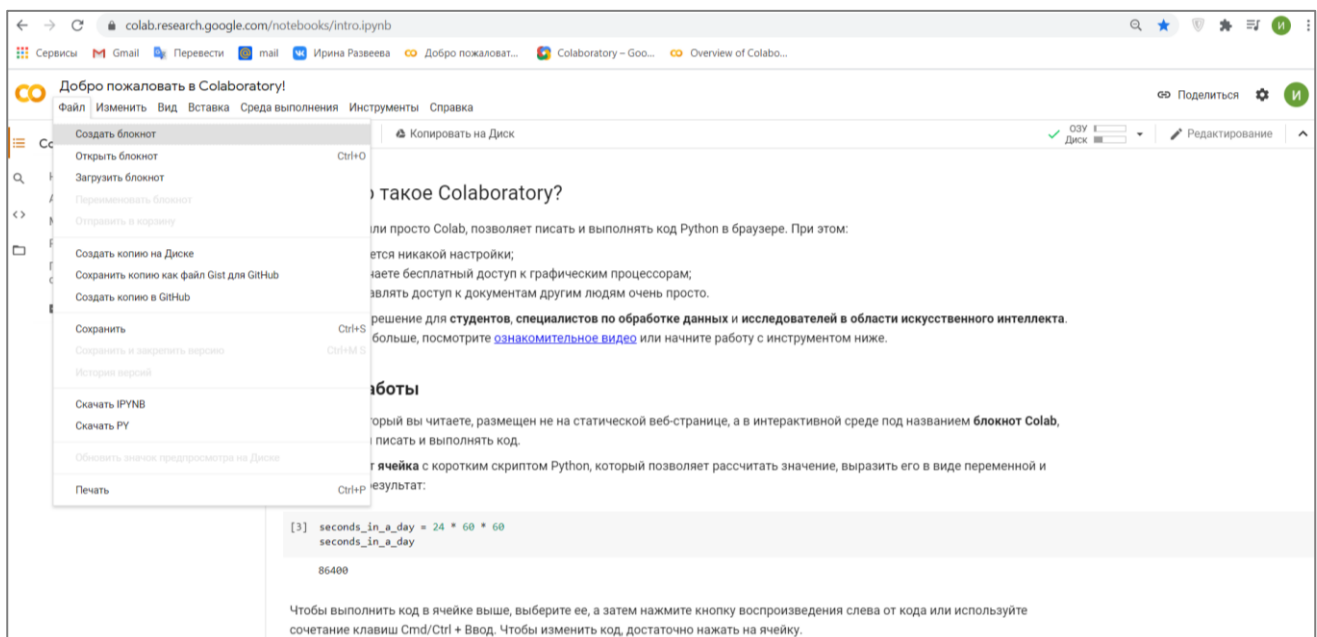
Обзор библиотек в Google Colaboratory

Цель работы: ознакомиться с основными библиотеками (NumPy и Pandas),
с которыми возможно работать в Google Colaboratory.

1. Откройте платформу по ссылке:

<https://colab.research.google.com/notebooks/intro.ipynb>

2. Перейдя во вкладку Файл выбрать «Создать блокнот».



3. Перед нами пустая рабочая область.

Документ, который вы читаете, размещен не на статической веб-странице, а в интерактивной среде под названием блокнот Colab, позволяющей писать и выполнять код.



Рассмотрим основные библиотеки, набирающие популярность, для анализа данных с помощью средств Python.

NUMPY

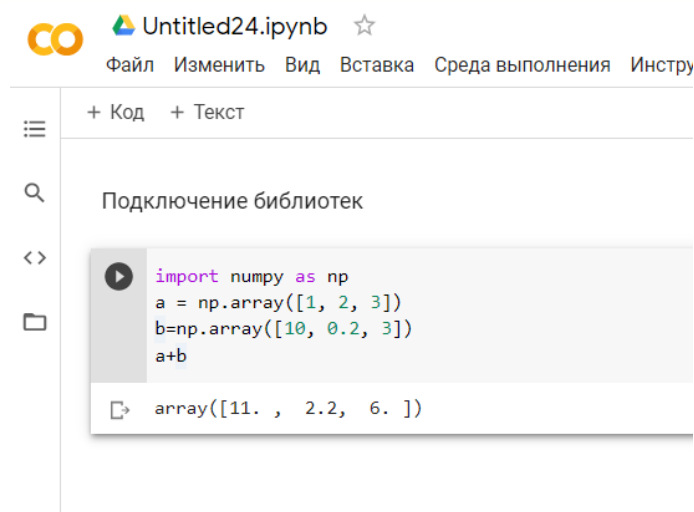
NumPy — это библиотека языка Python, добавляющая поддержку больших многомерных массивов и матриц, вместе с большой библиотекой высокоуровневых (и очень быстрых) математических функций для операций с этими массивами.

Подключим библиотеку numpy. И решим с ее помощью ряд задач.

Пример 1.

Задать вектора *a* и *b*, найти их сумму.

```
import numpy as np
a = np.array([1, 2, 3])
b=np.array([10, 0.2, 3])
a+b
```

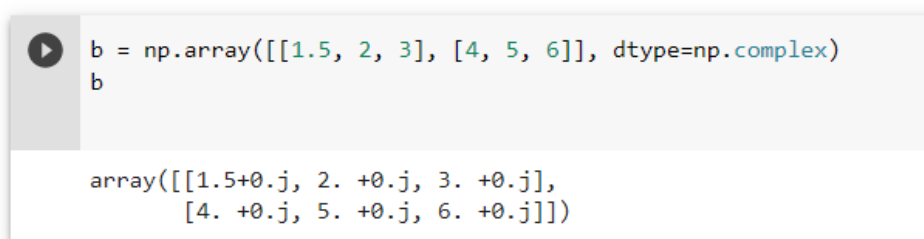


Пример 2.

Массив *b* преобразовать в массив комплексных чисел.

```
b = np.array([[1.5, 2, 3], [4, 5, 6]], dtype=np.complex)
b
```

В данном примере мы переопределяем тип массива в момент создания.



Пример 3.

Создать числовую последовательность от 10 до 30 с шагом 5.

```
np.arange(10, 30, 5)
```

```
np.arange(10, 30, 5)  
array([10, 15, 20, 25])
```

Пример 4.

Создать массив из 15 элементов. Развернуть его (первый становится последним).

```
Z = np.arange(15)  
Z = Z[::-1]  
Z
```

```
Z = np.arange(15)  
Z = Z[::-1]  
Z  
array([14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0])
```

Пример 5.

Создать массив 10x10 со случайными значениями, найти минимум и максимум.

```
Z = np.random.random((10,10))  
Zmin, Zmax = Z.min(), Z.max()  
Zmin, Zmax
```

```
Z = np.random.random((10,10))  
Zmin, Zmax = Z.min(), Z.max()  
Zmin, Zmax  
(0.003735980452566845, 0.9952152099115769)
```

Пример 6.

Создать случайный вектор размера 30 и найти среднее значение всех элементов.

```
Z = np.random.random(30)  
m = Z.mean()  
m  
0.5805939976899881
```

PANDAS

Pandas - это высокоуровневая Python библиотека для анализа данных, построена она поверх более низкоуровневой библиотеки NumPy (написана на Си), что является большим плюсом в производительности. В экосистеме Python, Pandas является наиболее продвинутой и быстроразвивающейся библиотекой для обработки и анализа данных.

Чтобы эффективно работать с pandas, необходимо освоить самые главные структуры данных библиотеки: DataFrame и Series. Без понимания что они из себя представляют, невозможно в дальнейшем проводить качественный анализ.

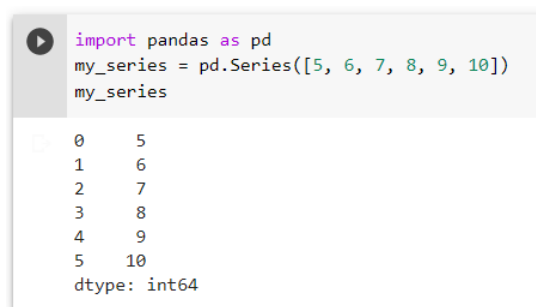
Series

Структура/объект Series представляет из себя объект, похожий на одномерный массив (питоновский список, например), но отличительной его чертой является наличие ассоциированных меток, т.н. индексов, вдоль каждого элемента из списка. Такая особенность превращает его в ассоциативный массив или словарь в Python.

Пример 7.

Создать объект Series, вывести на экран.

```
import pandas as pd
my_series = pd.Series([5, 6, 7, 8, 9, 10])
my_series
```



```
import pandas as pd
my_series = pd.Series([5, 6, 7, 8, 9, 10])
my_series
```

0	5
1	6
2	7
3	8
4	9
5	10

dtype: int64

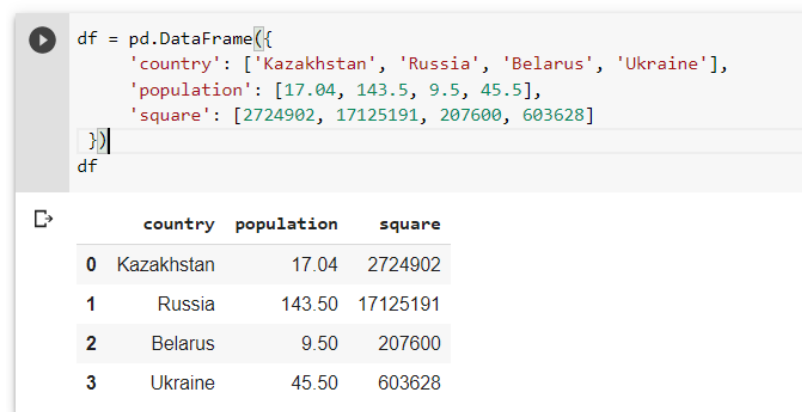
DataFrame

Объект DataFrame лучше всего представлять себе в виде обычной таблицы и это правильно, ведь DataFrame является табличной структурой данных. В любой таблице всегда присутствуют строки и столбцы. Столбцами в объекте DataFrame

выступают объекты Series, строки которых являются их непосредственными элементами.

DataFrame проще всего сконструировать на примере питоновского словаря:

```
df = pd.DataFrame({
    'country': ['Kazakhstan', 'Russia', 'Belarus', 'Ukraine'],
    'population': [17.04, 143.5, 9.5, 45.5],
    'square': [2724902, 17125191, 207600, 603628]
})
df
```

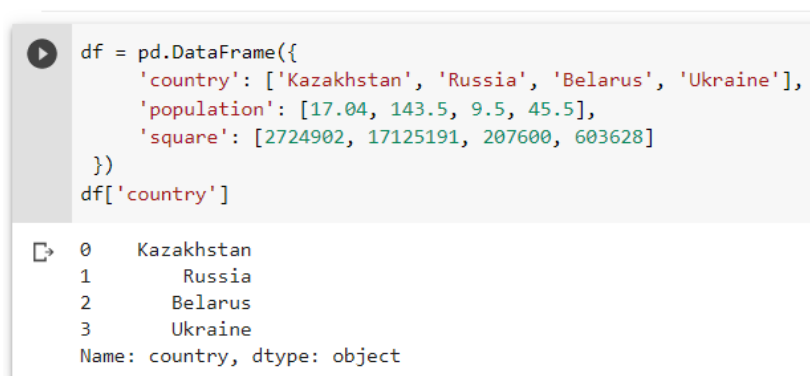


```
df = pd.DataFrame({
    'country': ['Kazakhstan', 'Russia', 'Belarus', 'Ukraine'],
    'population': [17.04, 143.5, 9.5, 45.5],
    'square': [2724902, 17125191, 207600, 603628]
})
df
```

	country	population	square
0	Kazakhstan	17.04	2724902
1	Russia	143.50	17125191
2	Belarus	9.50	207600
3	Ukraine	45.50	603628

Чтобы убедиться, что столбец в DataFrame это Series, извлекаем любой:

```
df = pd.DataFrame({
    'country': ['Kazakhstan', 'Russia', 'Belarus', 'Ukraine'],
    'population': [17.04, 143.5, 9.5, 45.5],
    'square': [2724902, 17125191, 207600, 603628]
})
df['country']
```



```
df = pd.DataFrame({
    'country': ['Kazakhstan', 'Russia', 'Belarus', 'Ukraine'],
    'population': [17.04, 143.5, 9.5, 45.5],
    'square': [2724902, 17125191, 207600, 603628]
})
df['country']
```

0	Kazakhstan
1	Russia
2	Belarus
3	Ukraine

Name: country, dtype: object

Самостоятельная работа:

1. Ознакомиться с математическими операциями над объектом Series.
2. Создать таблицу из 3 столбцов, 5 строк.