

Лабораторная работа №4

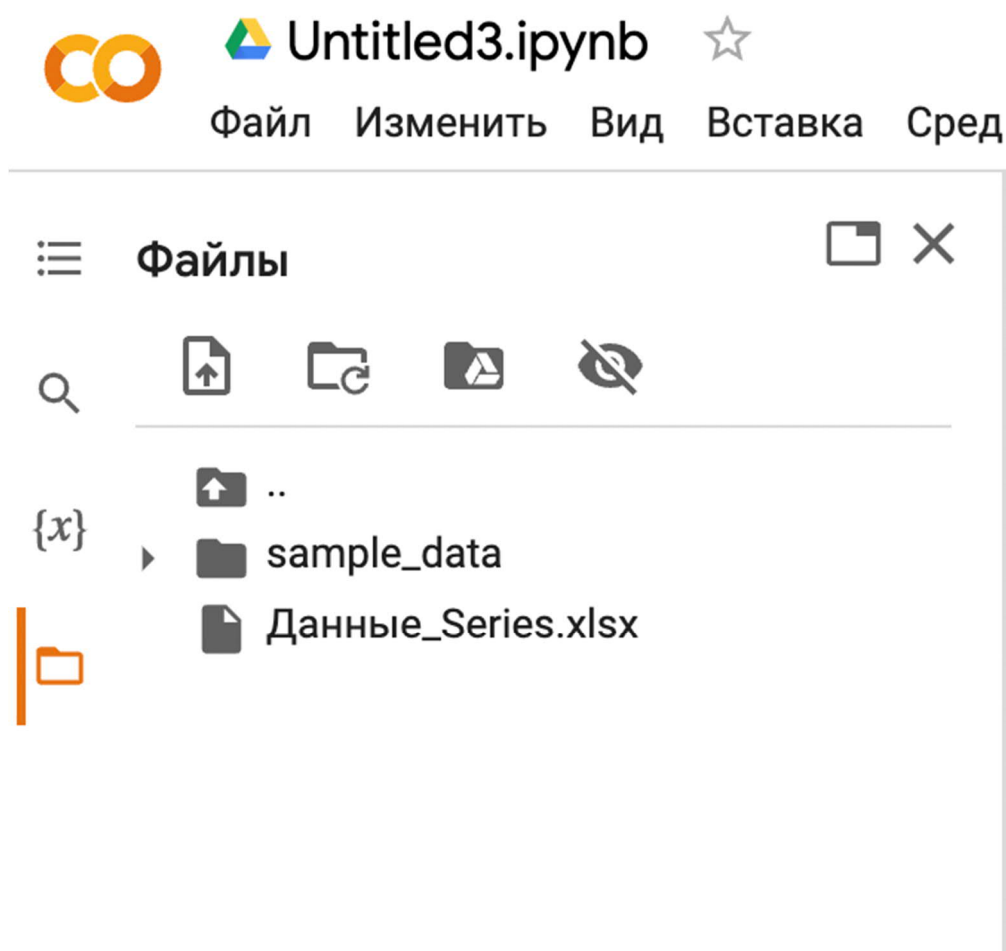
Загрузка данных

Загрузка данных типа Series

Загрузка данных в формате '.xlsx'

Для загрузки данных необходимо предусмотреть данные таким образом, чтобы заголовки располагались в отдельных ячейках. Например, в данном примере загружаются 2 показателя – год и поголовье крупного рогатого скота на конец соответствующего года (Файл «[Данные Series.xlsx](#)», Лист 1).

Для загрузки данный файл необходимо поместить в репозиторий Google Colab:



Для загрузки данных используем библиотеку «pandas». Для ее подключения используем `import` и по желанию можно указать сокращенное название для вызова функций из этой библиотеки (`pd`).

```
import pandas as pd
```

Импортируем библиотеку и загружаем файл с помощью функции «`pd.read_excel`»:

```
df= pd.read_excel('Данные_Series.xlsx', sheet_name='Лист1', header=0)
```

В данной функции прописываем название файла. Если он находится в другой директории, то необходимо прописать полный путь к нему. `sheet_name` – лист на котором расположены исходные данные. `header=0` – чтобы автоматически 1 строка исходных данных была использована как заголовки при загрузке. Посмотрим результаты загрузки:

```
import pandas as pd
df= pd.read_excel('Данные_Series.xlsx', sheet_name='Лист1', header=0)
print(df)
```

	Year	Cattle
0	1883	23.6
1	1915	33.0
2	1922	26.7
3	1927	37.6
4	1929	30.4
..
91	2017	18.3
92	2018	18.2
93	2019	18.1
94	2020	18.0
95	2021	17.6

[96 rows x 2 columns]

Тип Series

Объект типа `Series` предполагает работу с одним столбцом. Преобразуем загруженные данные в формат `Series`. Для этого в `pandas` используется одноименная функция. Столбец с поголовьем КРС будет использоваться как значения, а год как индекс (именованное обозначение переменной):

```
ds = pd.Series(df['Cattle'])
```

```
ds= ds.rename(df['Year'])
```

```
print(type(ds))
```

```
print(ds)
```

Для вывода первых значений используется функция `head()`, для последних значений – `tail()`. По умолчанию выводится 5 значений. Если указать количество элементов `n`, то выведутся `n` указанных элементов.



```
ds.head(10)
```

```
1883    23.6
1915    33.0
1922    26.7
1927    37.6
1929    30.4
1930    25.5
1931    23.4
1932    21.4
1933    21.8
1934    25.3
```

```
Name: Cattle, dtype: float64
```

Для обращения к наблюдениям используются квадратные скобки. Так, для обращения к 5-му элементу указываем соответствующий индекс (индексация с 0), если нумерация индексов по умолчанию.

В нашем случае индексы — года. Поэтому для обращения к значению определенного года необходимо указать год:



```
ds[1915]
```

```
33.0
```

Для обращения к диапазону элементов используются индексы по умолчанию.

```
ds[0:6]
```

1883	23.6
1915	33.0
1922	26.7
1927	37.6
1929	30.4
1930	25.5

Name: Cattle, dtype: float64

Для вывода значений, удовлетворяющих определённым условиям, можно задавать условия в квадратных скобках. Так например, для вывода элементов выше 55 запись будет иметь вид:

```
ds[ds>55]
```

1974	56.5
1975	57.6
1976	56.9
1977	58.0
1978	58.5
1979	58.6
1980	58.1
1981	58.1
1982	58.6
1983	59.6
1984	60.0
1985	59.6
1986	60.5
1987	59.8
1988	59.3
1989	58.8
1990	57.0

Name: Cattle, dtype: float64

Для обращения только к значениям или только индексам используются функции `values` и `index` соответственно.

Математические операции будут выполняться с каждым элементом.

Так, если есть необходимость перевода в другую единицу измерения, например из миллионов голов в тыс голов можно сделать следующую запись:

```
dg=ds*1000
dg

1883      23600.0
1915      33000.0
1922      26700.0
1927      37600.0
1929      30400.0
...
2017      18300.0
2018      18200.0
2019      18100.0
2020      18000.0
2021      17600.0
Name: Cattle, Length: 96, dtype: float64
```

Для вывода основных статистических характеристик могут применяться следующие функции pandas:

- Минимальное значение:

```
ds.min()
```

- Максимальное значение:

```
ds.max()
```

- Среднее значение:

```
ds.mean()
```

- Медиана:

```
ds.median()
```

- Среднеквадратическое отклонение:

```
ds.std()
```

- Дисперсия:

```
ds.var()
```

- Отсортировать значения

```
ds.sort_values
```

Все доступные методы работы с Series можно найти в официальной документации по пакету [pandas](#)

Задания для самостоятельной тренировки

1. Загрузить данные с листа 2 в python о валовой продукции животноводства в 2021 году по регионам страны. Преобразовать данные в тип Series (индексы – регионы).
2. Вывести значение, которое соответствует Московской области
3. Вывести 10 первых значение показателя.
4. Вывести 12 последний значение показателя.
5. Найти минимальное и максимальное значение по совокупности.
6. Отсортировать данные по возрастанию.
7. Отсортировать данные по убыванию.
8. Рассчитать среднюю по совокупности. Вывести все значения, выше среднего уровня.
9. Определить медиану.
10. Рассчитать дисперсию и среднеквадратическое отклонение.
11. Определить коэффициент вариации (отношение среднеквадратического отклонения к средней по совокупность *100%).

Загрузка данных типа Data Frame

Загрузка данных в формате '.xlsx'

Для загрузки данных необходимо предусмотреть данные таким образом, чтобы заголовки располагались в отдельных ячейках. Например, в данном примере загружаются данные о поголовье скота и птицы в хозяйствах всех категорий на 01.01.2023 года (тыс. гол.) (Файл «[Данные DataFrame.xlsx](#)»).

По аналогии, файл необходимо поместить в репозиторий Google Colab, импортировать библиотеку pandas и загрузить данные:

```
import pandas as pd
```

```
df= pd.read_excel('Данные_DataFrame.xlsx', sheet_name='Лист1',  
header=0)
```

В данной функции прописываем название файла. Если он находится в другой директории, то необходимо прописать полный путь к нему. sheet_name – лист на котором расположены исходные данные. header=0) – чтобы автоматически 1 строка исходных данных была использована как заголовки при загрузке.

```
import pandas as pd
df = pd.read_excel('Данные_DataFrame.xlsx', sheet_name='Лист1', header=0)
df
```

	Субъект РФ	Крупный рогатый скот	Коровы	Свиньи	Овцы и козы	Птица
0	Белгородская область	235.30	92.30	4378.5	59.20	47590.0
1	Брянская область	554.90	222.10	1002.1	22.40	17290.5
2	Владимирская область	133.90	56.50	1.7	24.40	3207.0
3	Воронежская область	489.00	171.70	1966.1	176.80	12289.8
4	Ивановская область	59.40	26.00	84.7	23.20	2930.2
...
79	Амурская область	64.10	30.10	22.5	11.00	2197.1
80	Магаданская область	3.70	1.70	2.1	0.40	85.5
81	Сахалинская область	28.70	12.80	41.6	4.20	639.3
82	Еврейская автономная область	6.30	2.70	1.3	3.50	73.4
83	Чукотский автономный округ	0.02	0.01	0.1	0.01	31.9

84 rows x 6 columns

Тип Data Frame

Для объектов типа DataFrame применимы функции head() и tail() так же как и для типа Series.

Для обращения к столбцам таблицы могут быть использованы различные пути:

1. Указать название столбца в квадратных скобках:

```
df['Коровы']
```

0	92.30
1	222.10
2	56.50
3	171.70
4	26.00
...	...
79	30.10
80	1.70
81	12.80
82	2.70
83	0.01

Name: Коровы, Length: 84, dtype: float64

2. Прописав название столбца после точки:

```
df.Коровы
```

0	92.30
1	222.10
2	56.50
3	171.70
4	26.00
...	
79	30.10
80	1.70
81	12.80
82	2.70
83	0.01

Name: Коровы, Length: 84, dtype: float64

3. С помощью функции `iloc`. Для этого на вход необходимо подать список, состоящий из индексов строк и столбцов для обращения. Так, для того чтобы вывести столбец с поголовьем коров (индекс 2) по всем регионам, необходимо на первом месте в списке подать «:», чтобы выбрать все строки, а на второй позиции подать индекс нужного столбца «2»:

```
df.iloc[:,2]
```

0	92.30
1	222.10
2	56.50
3	171.70
4	26.00
...	
79	30.10
80	1.70
81	12.80
82	2.70
83	0.01

Name: Коровы, Length: 84, dtype: float64

С помощью данной функции можно обращаться к отдельным элементам таблицы по индексам. Так, чтобы выбрать поголовье коров во Воронежской области (индекс строки 3):

```
df.iloc[3,2]
```

171.7

Чтобы обратиться к показателям конкретного региона, можно выбрать нужный индекс строки и указать, что данные нужны по всем столбцам:

```
df.iloc[3,:]
```

Субъект РФ	Воронежская область
Крупный рогатый скот	489.0
Коровы	171.7
Свиньи	1966.1
Овцы и козы	176.8
Птица	12289.8

Name: 3, dtype: object

Аналогично можно выбирать диапазон строк или столбцов для вывода:

```
df.iloc[0:3,0:4]
```

	Субъект РФ	Крупный рогатый скот	Коровы	Свиньи
0	Белгородская область	235.3	92.3	4378.5
1	Брянская область	554.9	222.1	1002.1
2	Владимирская область	133.9	56.5	1.7

Для обращения к элементам, удовлетворяющим определённым условиям:

1. Можно использовать квадратные скобки.

Так, чтобы вывести регионы с численностью коров выше 200 тыс. гол мы внутри скобок обращаемся к столбцу и задаем условие:

```
[43] df[df.Коровы>200]
```

	Субъект РФ	Крупный рогатый скот	Коровы	Свиньи	Овцы и козы	Птица
1	Брянская область	554.9	222.1	1002.1	22.4	17290.5
29	Республика Калмыкия	298.2	212.8	7.7	1523.5	109.1
31	Краснодарский край	566.0	220.7	666.8	222.7	23030.2
34	Ростовская область	628.2	300.4	331.9	938.4	17015.5
36	Республика Дагестан	939.1	473.2	0.5	4696.1	4009.4
43	Республика Башкортостан	856.6	361.4	571.0	582.5	11098.7
46	Республика Татарстан (Татарстан)	913.5	328.2	468.5	300.8	17465.9
52	Оренбургская область	534.2	234.5	263.9	295.9	7281.3
66	Алтайский край	620.1	259.7	318.8	150.7	9047.5

2. Аналогично можно использовать функцию «where»:

```
df.where(df.Коровы>200, df)
```

	Субъект РФ	Крупный рогатый скот	Коровы	Свиньи	Овцы и козы	Птица
0	Белгородская область	235.30	92.30	4378.5	59.20	47590.0
1	Брянская область	554.90	222.10	1002.1	22.40	17290.5
2	Владимирская область	133.90	56.50	1.7	24.40	3207.0
3	Воронежская область	489.00	171.70	1966.1	176.80	12289.8
4	Ивановская область	59.40	26.00	84.7	23.20	2930.2
...
79	Амурская область	64.10	30.10	22.5	11.00	2197.1
80	Магаданская область	3.70	1.70	2.1	0.40	85.5
81	Сахалинская область	28.70	12.80	41.6	4.20	639.3
82	Еврейская автономная область	6.30	2.70	1.3	3.50	73.4
83	Чукотский автономный округ	0.02	0.01	0.1	0.01	31.9

84 rows x 6 columns

Математические операции будут выполняться с каждым элементом, так же как и для типа Series.

Для удаления элементов используется функция "pop", а для того, чтобы исходная таблица данных не изменилась создадим новую таблицу df2 через специальную команду copy. Далее удалим столбец с поголовьем крупного рогатого скота:

```
df2=df.copy()
```

```
df2.pop('Крупный рогатый скот')
```

Для расчета основных статистических показателей можно использовать функцию describe:

	<code>df.describe()</code>					
	Крупный рогатый скот	Коровы	Свиньи	Овцы и козы	Птица	
count	84.000000	84.000000	82.000000	84.000000	84.000000	
mean	208.152619	91.896548	337.807805	246.857976	6557.15119	
std	203.964294	89.272247	620.286294	591.435462	7963.74965	
min	0.020000	0.010000	0.010000	0.010000	0.10000	
25%	63.275000	29.925000	11.200000	21.550000	972.55000	
50%	167.600000	71.350000	163.400000	59.450000	3903.65000	
75%	265.425000	128.825000	341.500000	183.600000	9158.30000	
max	939.100000	473.200000	4378.500000	4696.100000	47590.00000	

Также статистические функции, которые использовались для работы с данными типа "Series" доступны и для DataFrame.

При применении этих функций ко всем столбцам таблицы необходимо иметь ввиду, что для текстовых столбцов не все показатели могут быть рассчитаны. Так, средняя не считается, поэтому выводится предупреждение.

Все доступные методы работы с DataFrame можно найти в официальной документации по пакету [pandas](https://pandas.pydata.org/pandas-docs/stable/10min.html).

Задания для самостоятельной тренировки

1. Загрузить данные листа «Лист2» о производство основных продуктов животноводства в хозяйствах всех категорий (тыс.т) в динамике файла «Данные_DataFrame.xlsx» в python (проверить тип – DataFrame). Для более удобного обращения к столбцам можно переименовать столбцы:
`df.rename(index={0: "Имя1", 1: "Имя2", 2: "Имя3"})`
2. Вывести первые 10 строк исходной таблицы. Вывести последние 5 строк таблицы.
3. Вывести все значения с 2010 года.
4. Вывести только те показатели таблицы, для которых производство молока превысило 30 млн т.
5. Вывести 12-ое значение столбца «Мед».
6. Вывести с 10 по 25 значения столбца «Яйца, млрд шт.».
7. Найти минимальное и максимальное значения по каждому показателю.
8. Рассчитать среднюю по каждому показателю. Вывести все значения, выше среднего уровня по производству меда.
9. Рассчитать основные статистические показатели для каждого столбца.
10. Удалить из таблицы 1 любой столбец на свой выбор