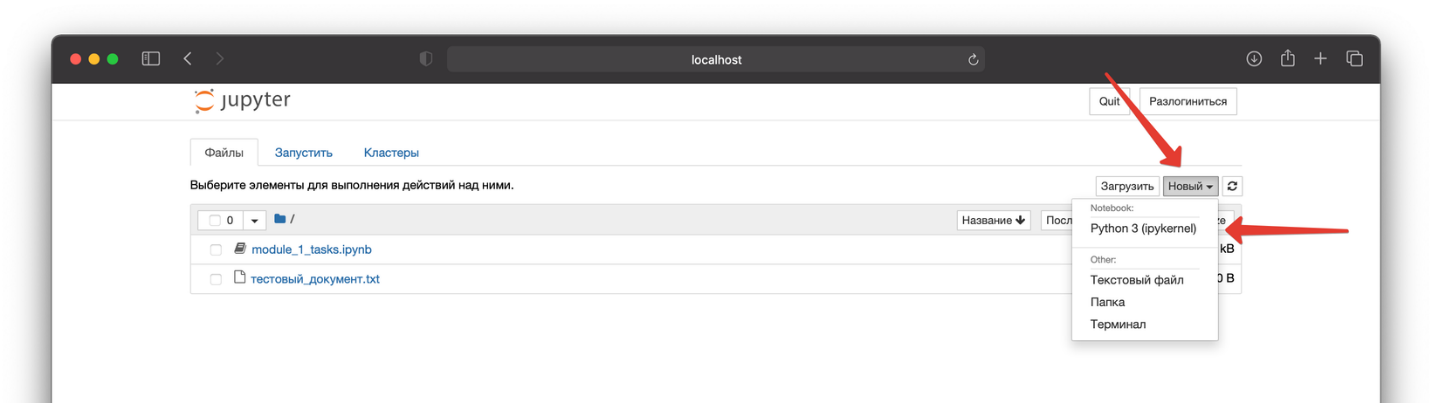
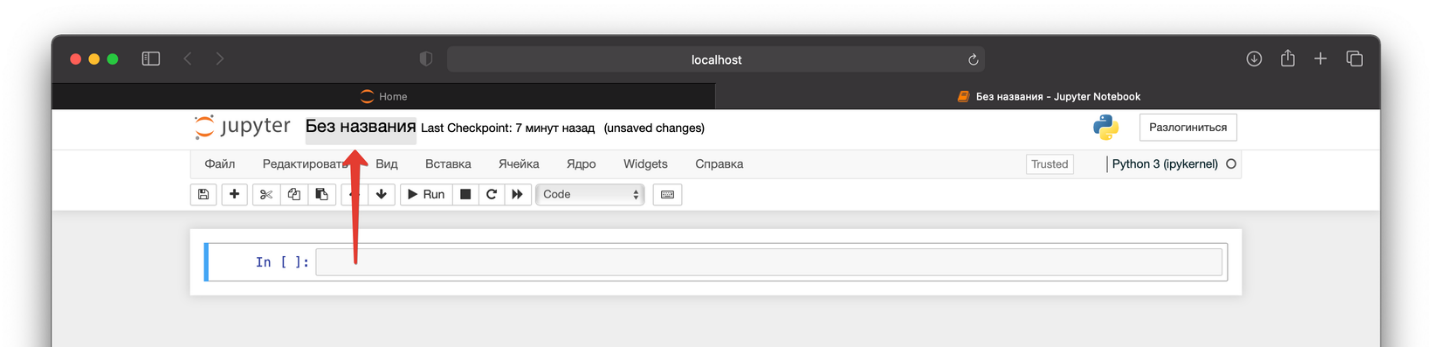
# Часть 1: Блокноты Jupyter

Объекты, с которыми мы работаем в среде Jupyter Notebook, называются **блокнотами**.

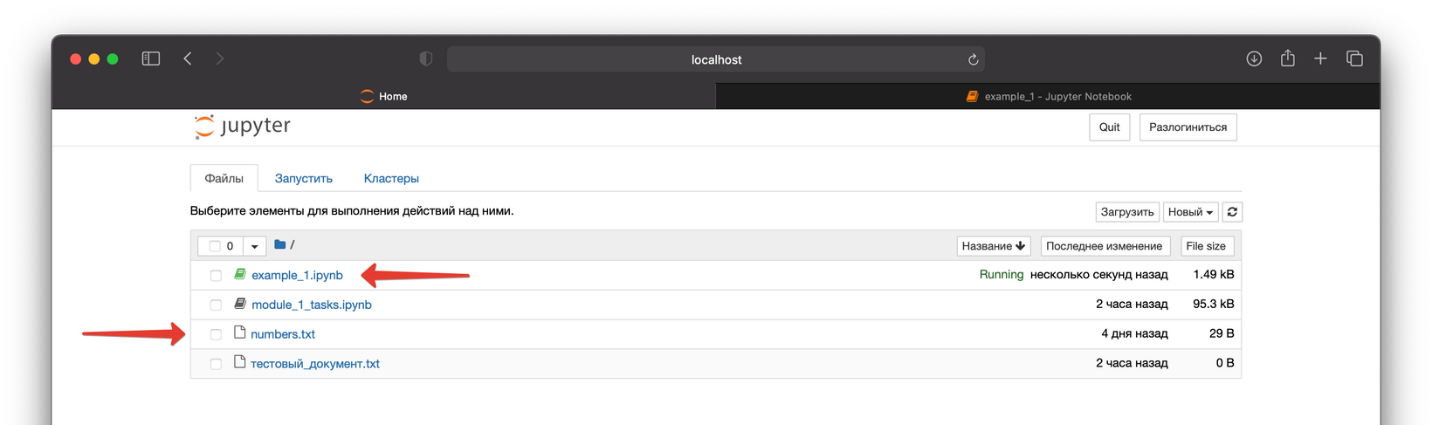
Чтобы создать новый блокнот в рабочей папке, нужно нажать на кнопку «Новый» и выбрать вариант «Python 3». Это соответствует тому, что вы создаёте блокнот для языка программирования Python (бывают блокноты и для других языков программирования):



По умолчанию блокнот создаётся с названием «Без названия». Лучше переименовать такой блокнот, чтобы не запутаться. Для этого нужно нажать на название блокнота и ввести актуальное название (например, example\_1):



Каждый новый просматриваемый блокнот открывается в новой вкладке браузера. Давайте вернёмся на первоначальную вкладку браузера и убедимся, что новый блокнот создался в нашей рабочей папке:



Видим, что в рабочей папке действительно появился новый файл example\_1.ipynb — это как раз наш блокнот.

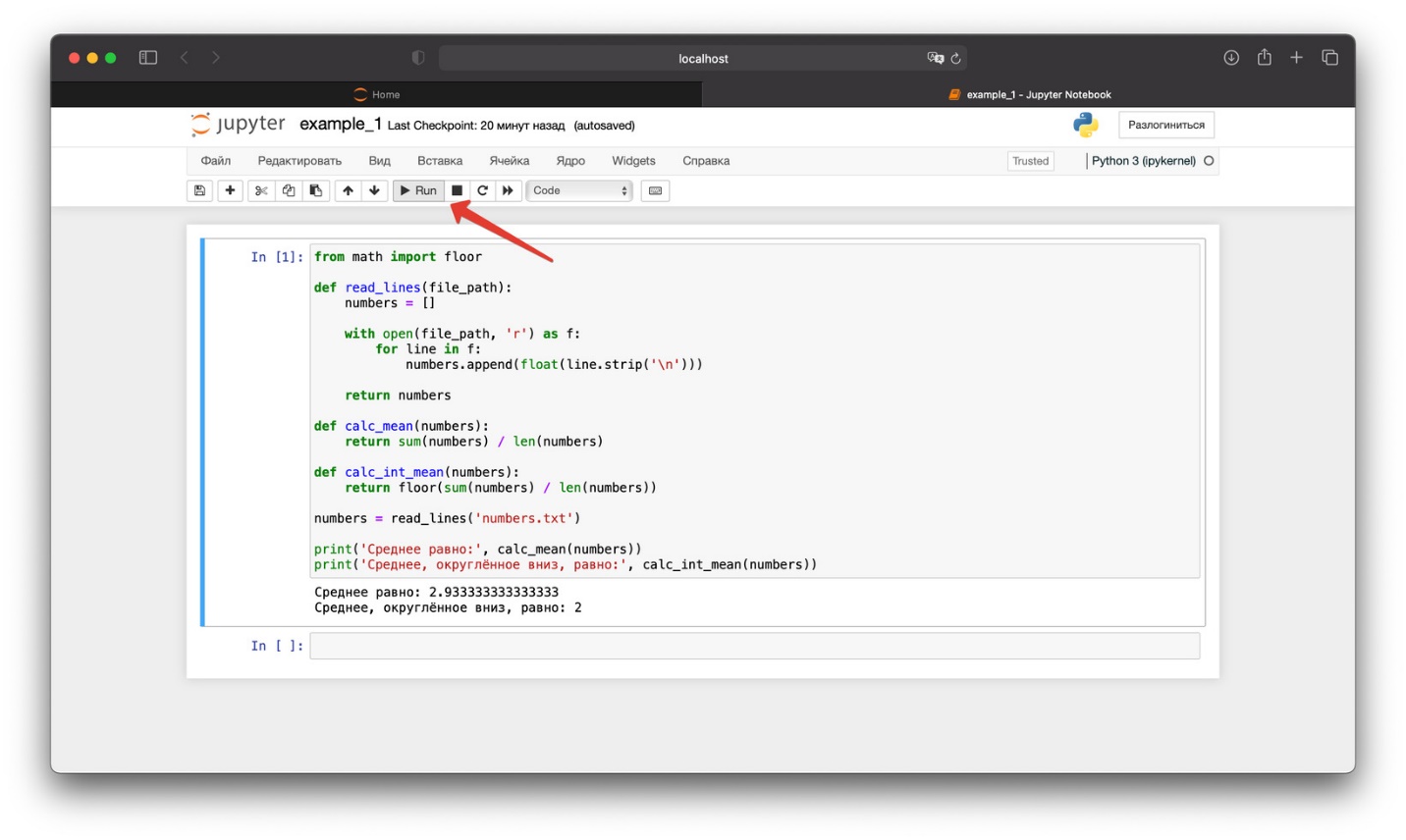
Важно, что любому блокноту, который находится в нашей рабочей папке, будут доступны (например, для чтения) все файлы, которые находятся в этой папке. В частности, внутри нашего блокнота example\_1 мы будем активно использовать файл numbers.txt, который тоже лежит в нашей рабочей папке.

Ячейки

Каждый блокнот состоит из **ячеек**. Ячейки в первую очередь используются для того, чтобы писать в них код.

Запуск

Можно использовать блокнот с одной ячейкой ровно так же, как вы бы использовали скрипт на языке Python. То есть можно вставить код программы в ячейку и исполнить его. Для того чтобы исполнить код, который написан в ячейке, нужно нажать на кнопку «Run» (или «Запуск»).

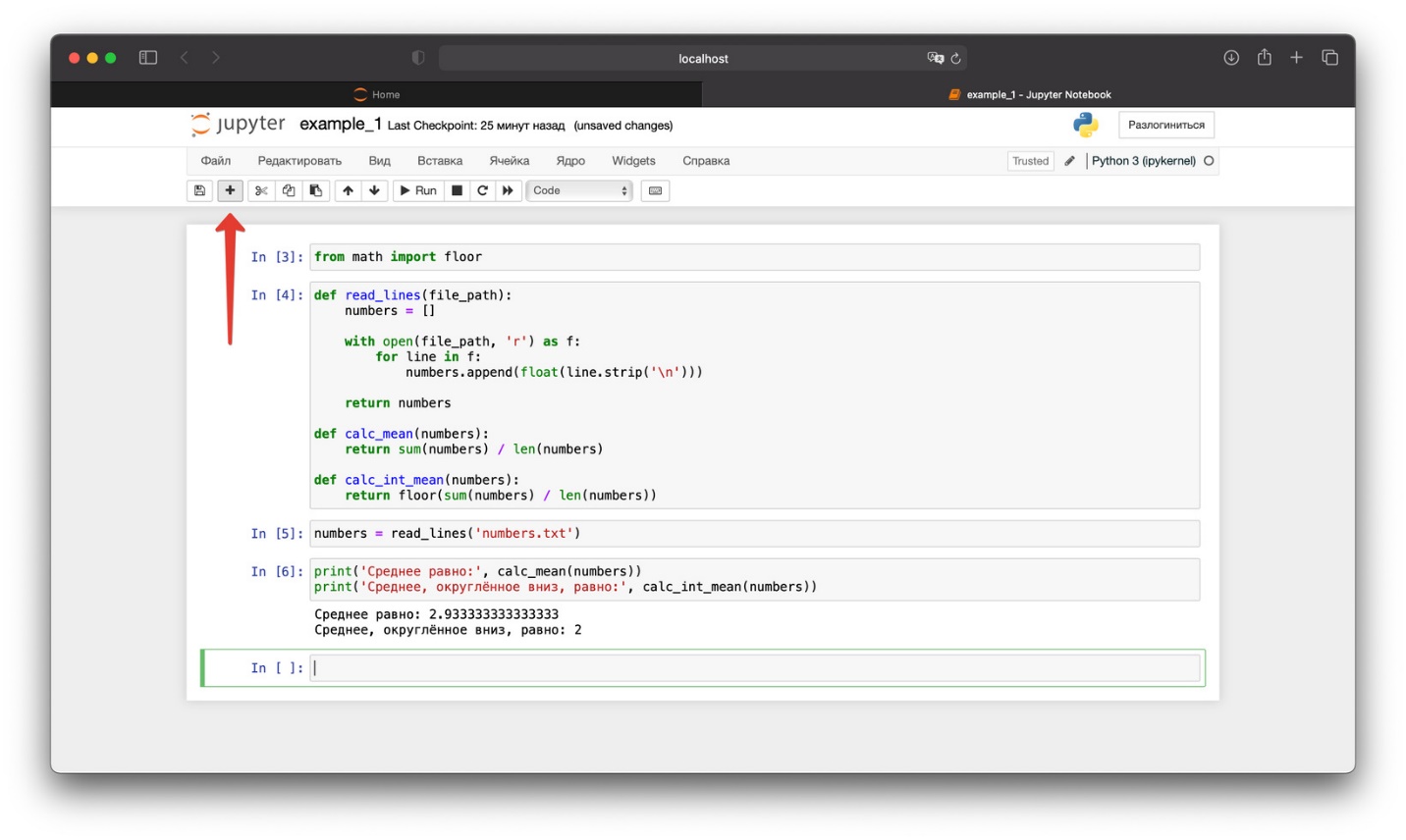


Также для запуска ячейки с кодом можно нажать на ячейку и воспользоваться горячим сочетанием клавиш Shift+Enter. Данное действие эквивалентно запуску ячейки по отдельной кнопке.

Создание

Большое преимущество блокнотов Jupyter заключается в том, что в них можно создавать много ячеек. В отдельные ячейки принято помещать отдельные кусочки логики программы. Новую ячейку можно создать, воспользовавшись кнопкой «+» в верхнем левом углу экрана.

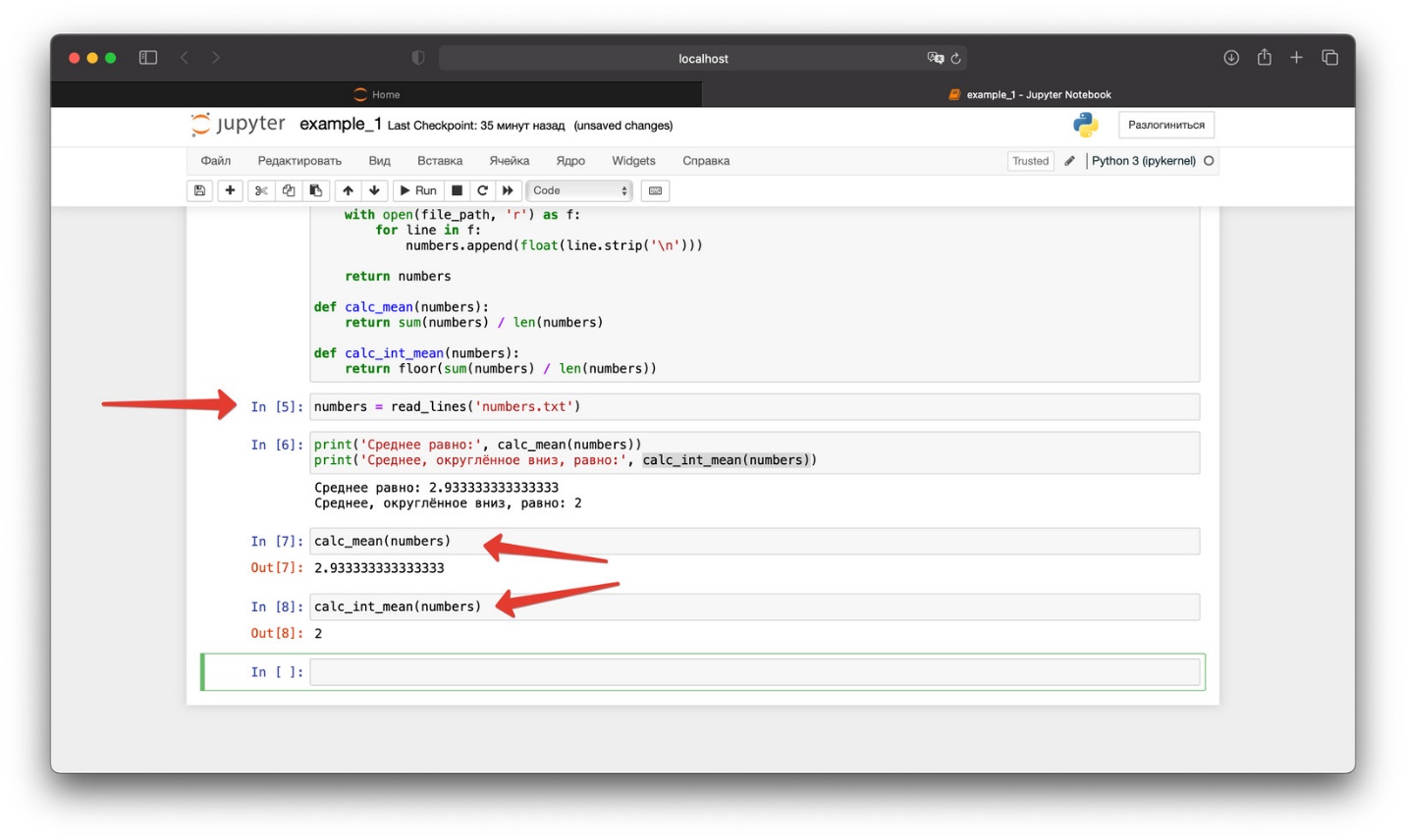
В случае нашего примера логично вынести в отдельную ячейку импорты, в отдельную — функции, а также перенести в отдельные ячейки чтение информации из файла numbers.txt и вывод результата работы функций на экран. Получится следующий блокнот:



Переиспользование вычислений

Главной особенностью того, что программу можно представить в виде системы ячеек, является возможность переиспользовать вычисления из отдельных ячеек.

Если в какой-то ячейке происходит запись значения в переменную, то значение переменной сохраняется в общую память блокнота и становится доступно всем другим ячейкам. Например, в нашем случае любая ячейка может использовать переменную numbers даже при том, что в самих этих ячейках переменная numbers не определена:



Это очень удобно в случае, если вычисление в какой-то ячейке занимает очень много времени и при этом в дальнейшем будет часто переиспользоваться.

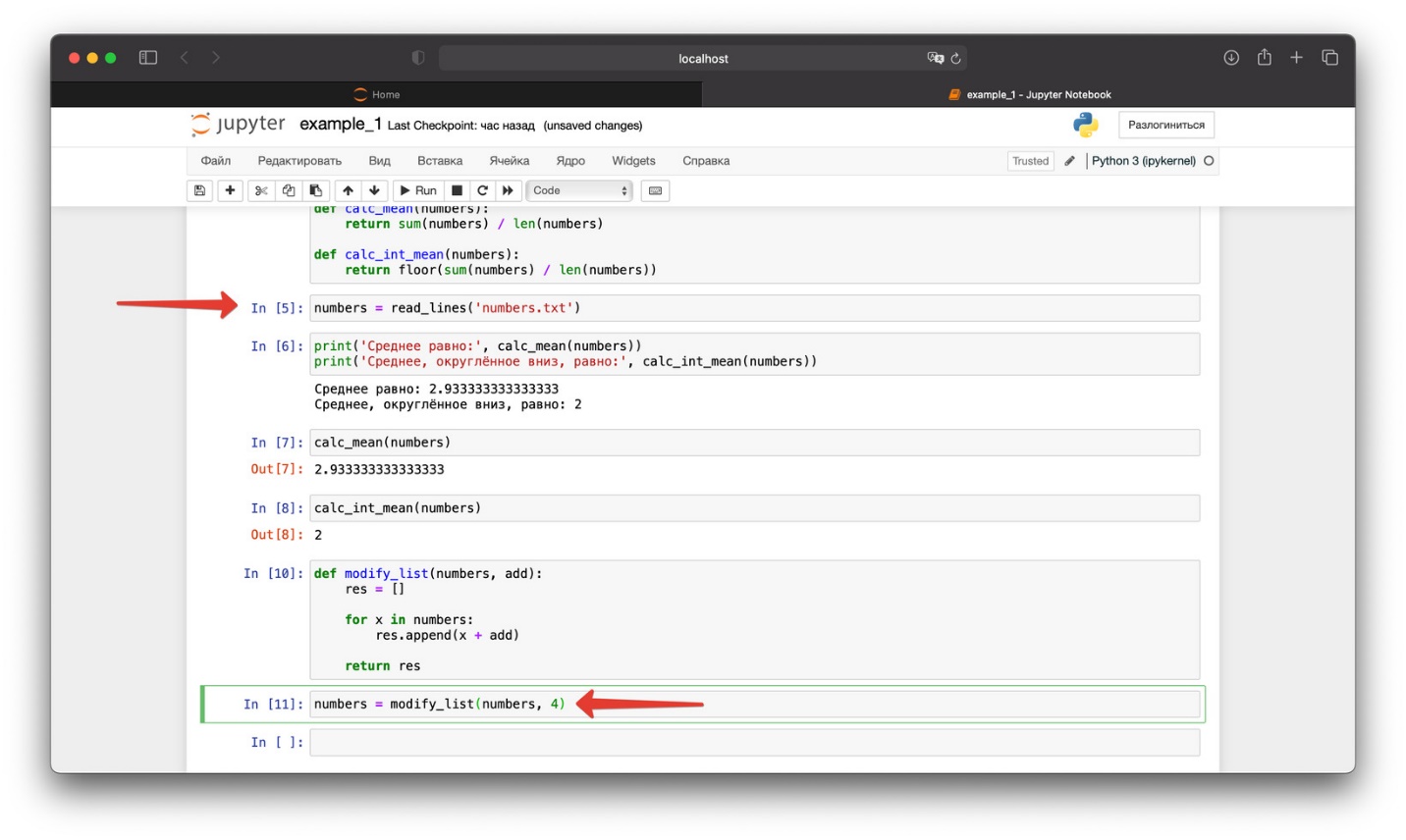
Например, в нашем случае чтение файла numbers.txt могло бы занимать какое-то продолжительное время (например, если бы файл был очень большим). Поэтому разумно вынести его в отдельную ячейку.

В этом случае в любой другой ячейке при обработке значений из файла numbers мы не тратим времени на его чтение, используя уже хранящееся в памяти значение.

Порядок вычислений

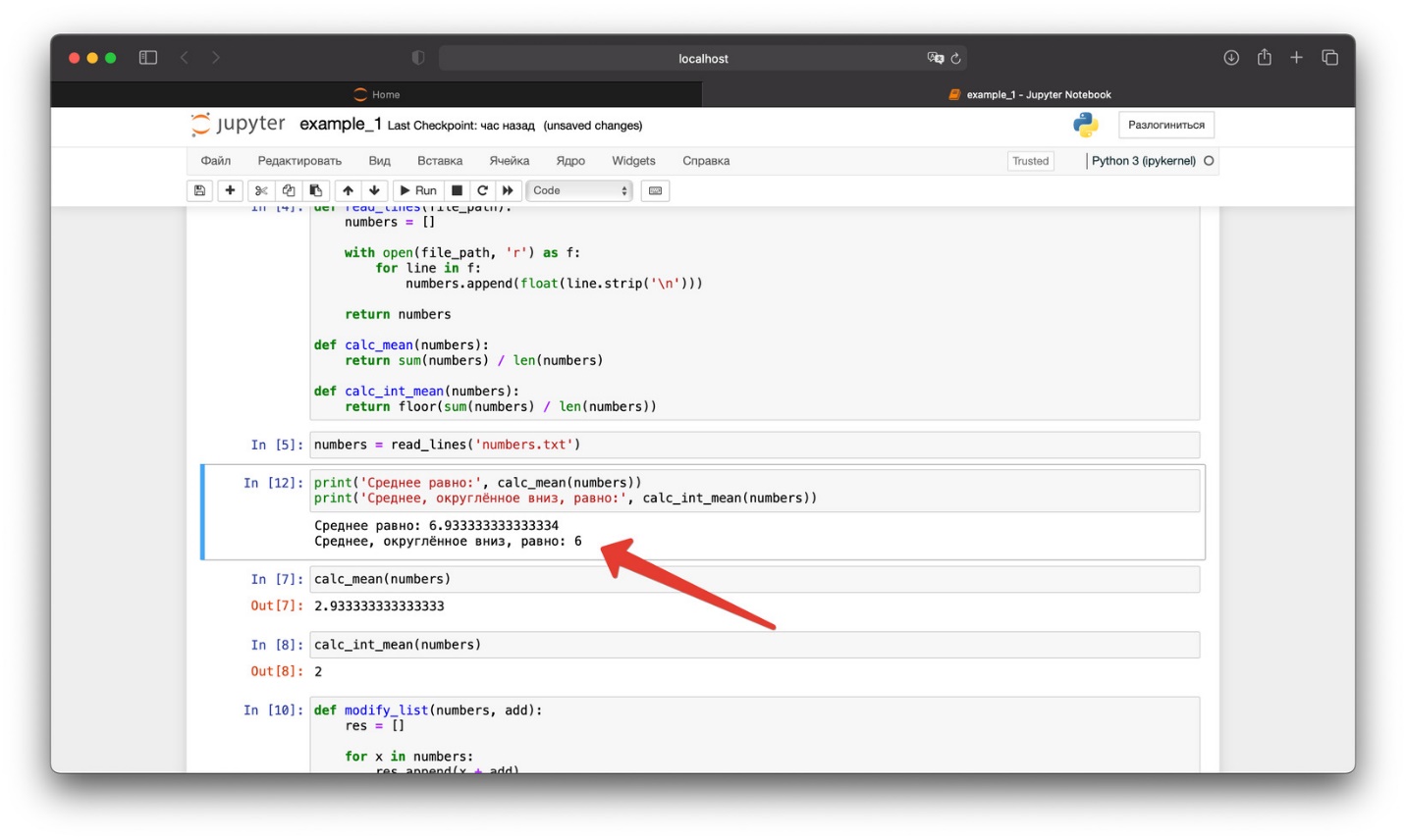
Ещё одной важной особенностью работы с ячейками является то, что порядок их исполнения задаётся порядком, в котором они вычисляются, а не порядком их следования.

Например, ячейка, которая расположена в блокноте ниже другой ячейки, может повлиять на результат её вычисления в будущем. В нашем примере с помощью функции modify\_list мы меняем список numbers:



Это приводит к тому, что значение переменной numbers меняется в общей памяти блокнота. Из-за этого любая ячейка, которая использует значение переменной и вычисляется после того, как мы изменили это значение, будет работать с обновлённой версией numbers.

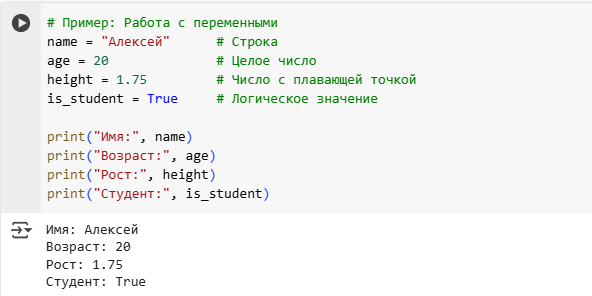
В частности, если теперь мы перезапустим вычисление ячейки, которая рассчитывает среднее значение для списка numbers и которую мы уже когда-то запускали, среднее изменится. Ровно потому, что теперь в numbers представлены другие значения.



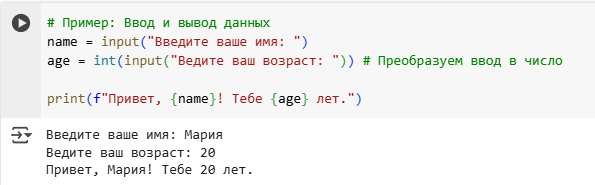
То есть при работе с ячейками важно помнить, что порядок их выполнения совпадает с порядком, в котором вы их запускаете, а не с их порядком следования в блокноте. И запускать ячейки можно в абсолютно любом порядке, не обязательно сверху–вниз.

# Часть 2: Основы Python

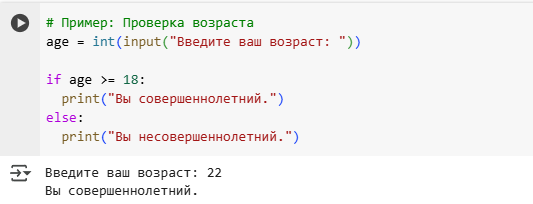
1. **Переменные и типы данных**



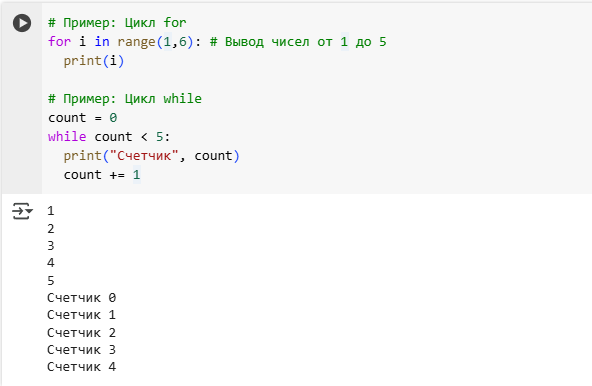
1. **Ввод и вывод данных**



1. **Условные операторы**



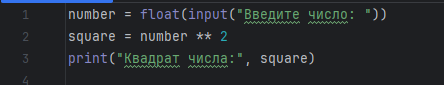
1. **Циклы**



**Практическое задание для Части 2**

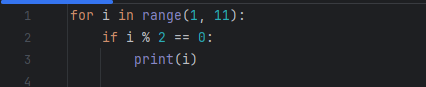
# Напишите программу, которая запрашивает у пользователя число и выводит его квадрат.

## Посмотреть решение



# Напишите программу, которая выводит все четные числа от 1 до 10.

## Посмотреть решение

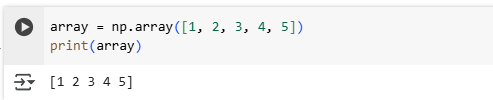


# Часть 3: Знакомство с NumPy

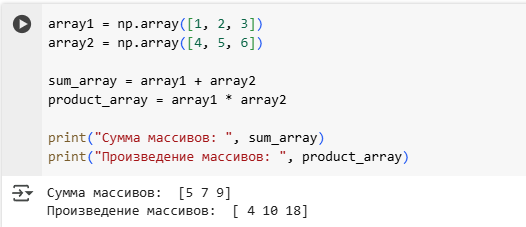
**1. Импорт библиотеки**



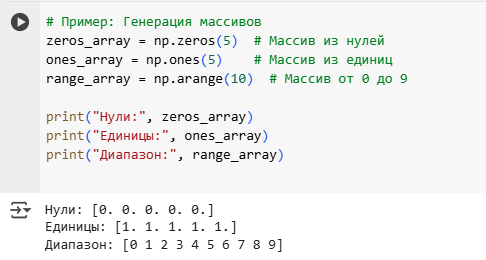
**2. Создание массивов**



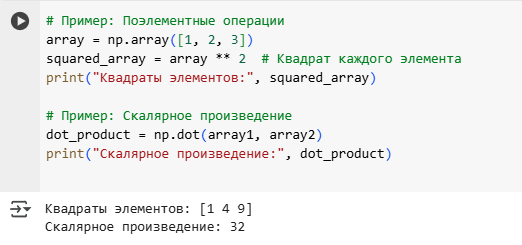
**3. Основные операции с массивами**



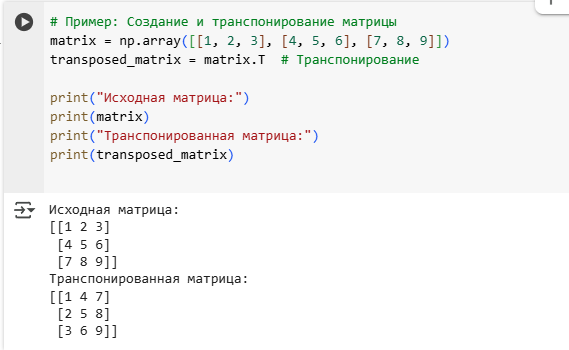
**4. Генерация массивов**



**5. Математические операции**



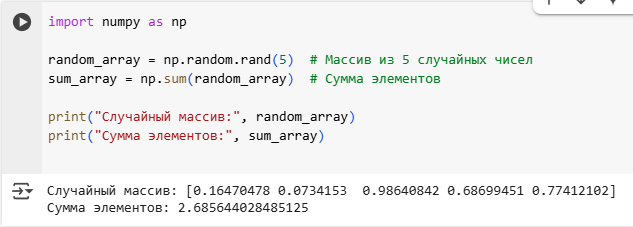
**6. Работа с матрицами**



**Практическое задание для Части 3**

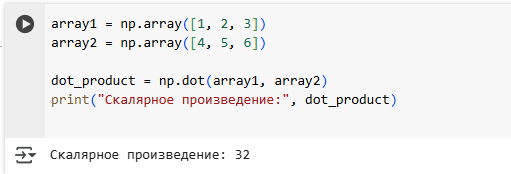
# Создайте массив из 5 случайных чисел и вычислите их сумму.

## Посмотреть решение



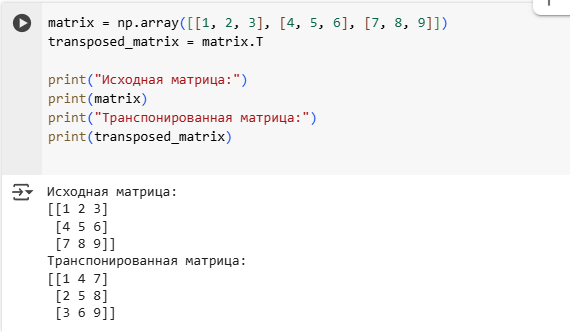
# Создайте два массива и вычислите их скалярное произведение.

## Посмотреть решение



# Создайте матрицу 3x3 и выведите её транспонированную версию.

## Посмотреть решение



# Итоговый пример для закрепления

