Санкт-Петербургский Национальный Исследовательский Университет Информационных Технологий, Механики и Оптики

**ОТЧЕТ ПО**

**ЛАБОРАТОРНОЙ РАБОТЕ №1**

Выполнил:

Абдулов И.А.

Проверил:

Мусаев А.А.

Санкт-Петербург

2022

# СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 1](#_Toc116034512)

[ВВЕДЕНИЕ 2](#_Toc116034513)

[Задание 2 3](#_Toc116034514)

[Ввод матрицы 3](#_Toc116034515)

[Транспонирование матрицы 3](#_Toc116034516)

[Умножение матриц 4](#_Toc116034517)

[Ранг матрицы 4](#_Toc116034518)

[Вывод данных 6](#_Toc116034519)

[Пример работы 7](#_Toc116034520)

[Задание 2 8](#_Toc116034521)

[Задание 3 10](#_Toc116034522)

[Обратная матрица 10](#_Toc116034523)

[Решение с numpy и сравнение времени работы 11](#_Toc116034524)

[Заключение 13](#_Toc116034525)

[Список использованных источников 14](#_Toc116034526)

# ВВЕДЕНИЕ

Данная лабораторная работа направлена на работу с матрицами в языке программирования Python.

В рамках выполнения данной лабораторной работы были поставлены данные задачи:

1. Задание 2: создать программу на языке Python, которая будет выполнять транспонирование, умножение, определение ранга матриц.
2. Задание 3: создать вышеописанную программу, используя библиотеку Numpy. Также необходимо проанализировать достоинства и недостатки использования Numpy.
3. Задание 4: Создать функцию для возведения матрицы в -1 степень и сравнить время выполнения с аналогичной из библиотеки Numpy.

# Задание 2

Задания: создать программу на языке Python, которая будет выполнять транспонирование, умножение, определение ранга матриц.

# Ввод матрицы

Для ввода матрицы с клавиатуры реализуем генератор списков.

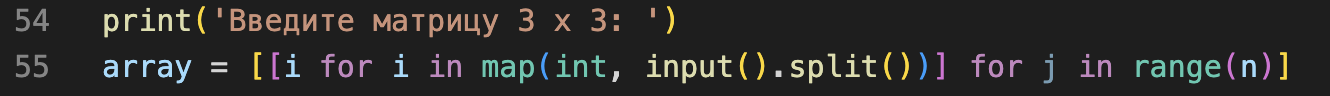


Рисунок 1 — Генератор списков в Python

## Транспонирование матрицы

Создадим функцию транспонирования матрицы transp(), которая принимает в качестве аргументов матрицу array и ее размерности n и m. В функции создается нулевая матрица transp\_array. После запускаются два цикла for, по строкам и столбцам, в которых элементу j-ой строки, i-ого столбца матрицы transp\_array присваивается элемент i-ой строки, j-ого столбца матрицы array. Функция возвращает транспонированную матрицу transp\_array.

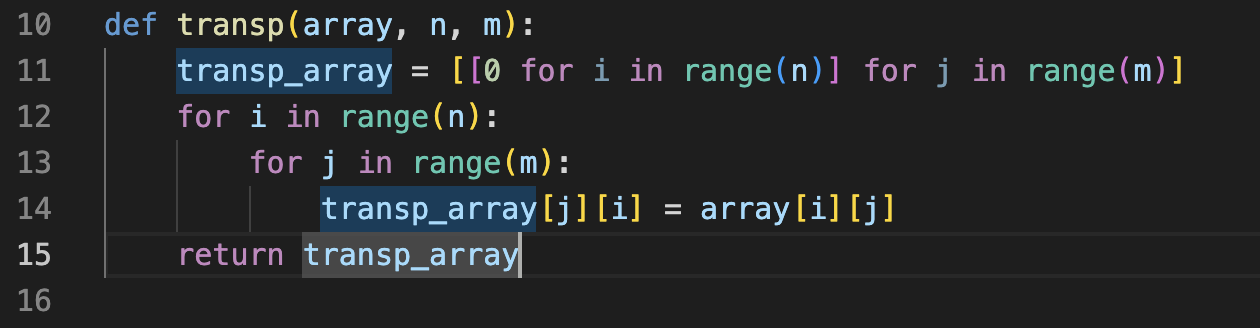


Рисунок 2 — Транспонирование матрицы

## Умножение матриц

Следующая функция mult() позволяет перемножить заданные ей в качестве аргументов матрицы arr1 и arr2. При умножении матриц количество столбцов матрицы arr1 должно совпадать с количеством строк матрицы arr2, поэтому в начале мы делаем проверку этого условия умножения матриц. Если матрицы подходят, то создаем нулевую матрицу mult\_array. После делаем вложенные циклы, в которых для элемента mult\_array[i][j] конечной матрицы, вычислим сумму произведений элементов строки i первой матрицы arr1 и элементов столбца j второй матрицы arr2, по строке и столбцу идет цикл с переменной k. После завершения работы циклов возвращаем полученную матрицу mult\_array.

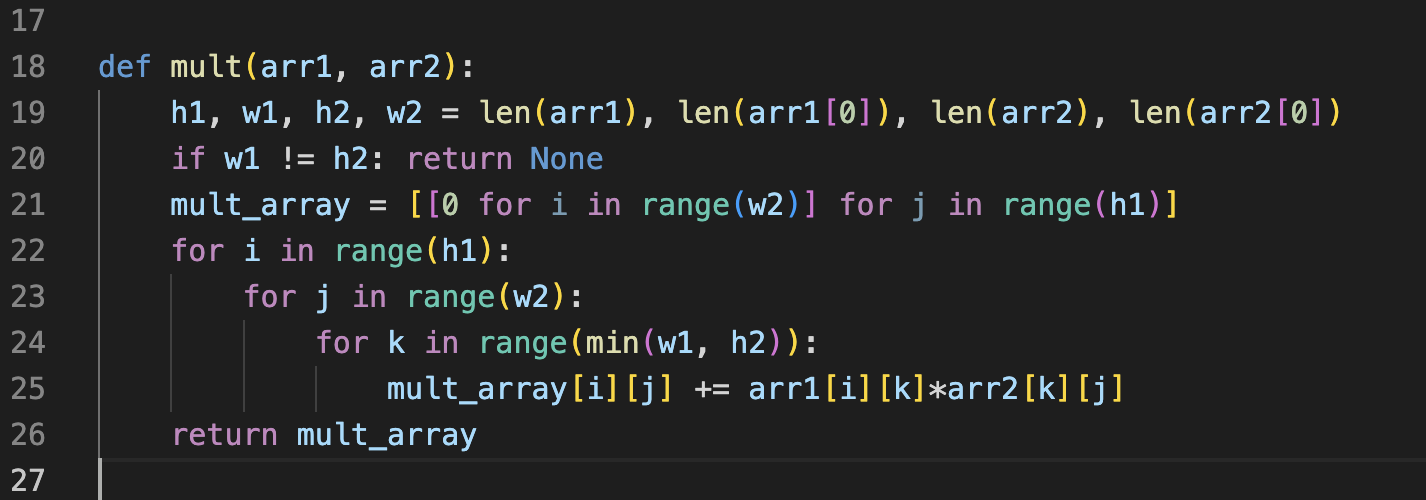


Рисунок 3 — Умножение матриц

## Ранг матрицы

Для нахождения ранга матрицы будем использовать метод окаймляющих миноров.

Функция array\_rang принимает матрицу arr. В начале создается нулевая матрица 3 x 3 из алгебраических дополнений элементов матрицы arr. Далее выполняются два цикла, которые для элемента матрицы arr[x][y] определяют его дополнительный минор 2-го порядка, элементы минора записываются в матрицу ar. Имея элементы определителя, вычислим его. Детерминант матрицы 2-го порядка считается по формуле:



Рисунок 4 — Формула вычисления определителя матрицы 2 x 2

Далее считаем алгебраическое дополнение элемента arr[x][y] по формуле:

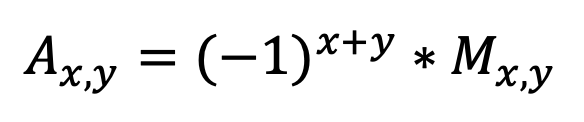


Рисунок 5 — Вычисления алгебраического дополнения элемента

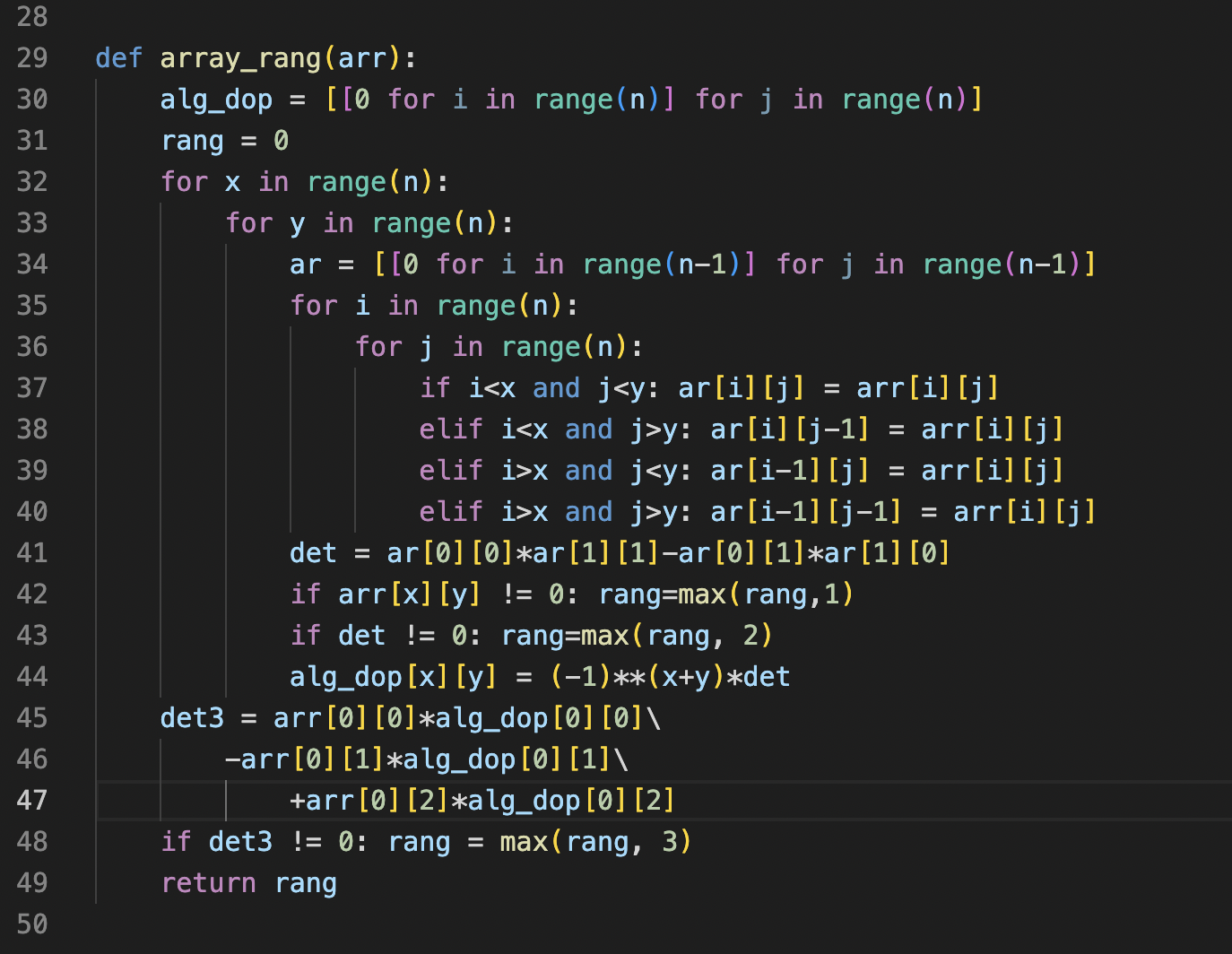


Рисунок 6 — Функция вычисления ранга матрицы

Вычислим определитель всей матрицы. Если он не равен 0, то ранг матрицы, который находится в переменной rang, считаем равным 3. Иначе, в предыдущем цикле мы проверяли определители 1-го и 2-го порядков, тем самым увеличивая значение переменной ранг.

Функция array\_rang возвращает переменную rang, которая равна максимальному порядку минора матрицы, который не равен 0.

## Вывод данных

Для вывода матрицы в приемлемом виде была создана функция, которая выводит строки в столбик.

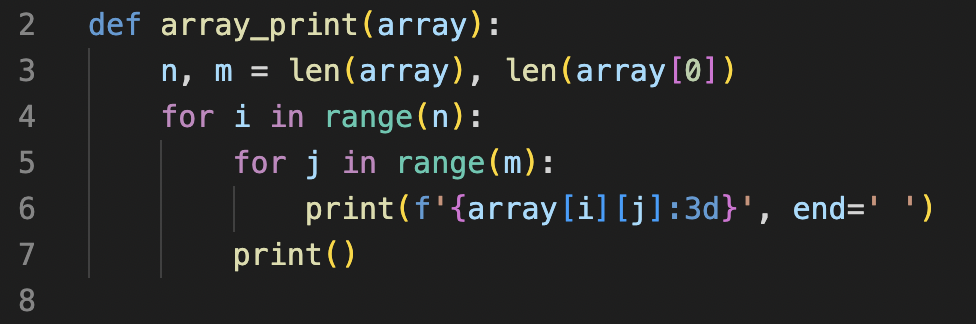


Рисунок 7 — Функция вывода матрицы

## Пример работы

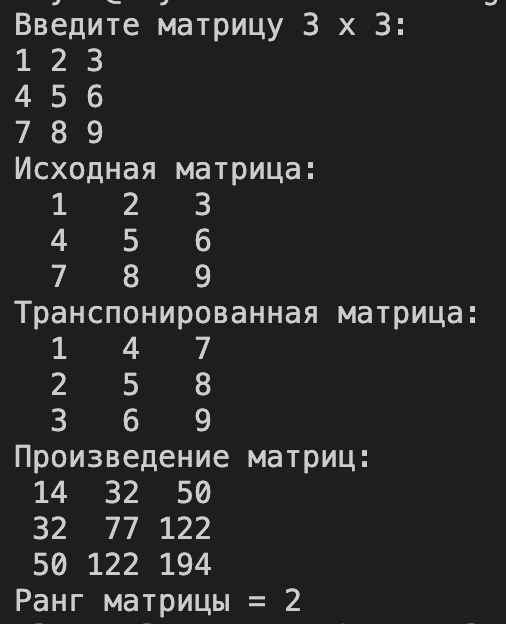


Рисунок 8 — Пример 1.1

# Задание 2

Задание: выполнить задание 1 с использованием библиотеки numpy. Проанализировать достоинства и недостатки использования numpy.

В начале импортируем numpy. Реализуем три метода библиотеки, которые транспонируют, умножают матрицы и ищут их ранг.

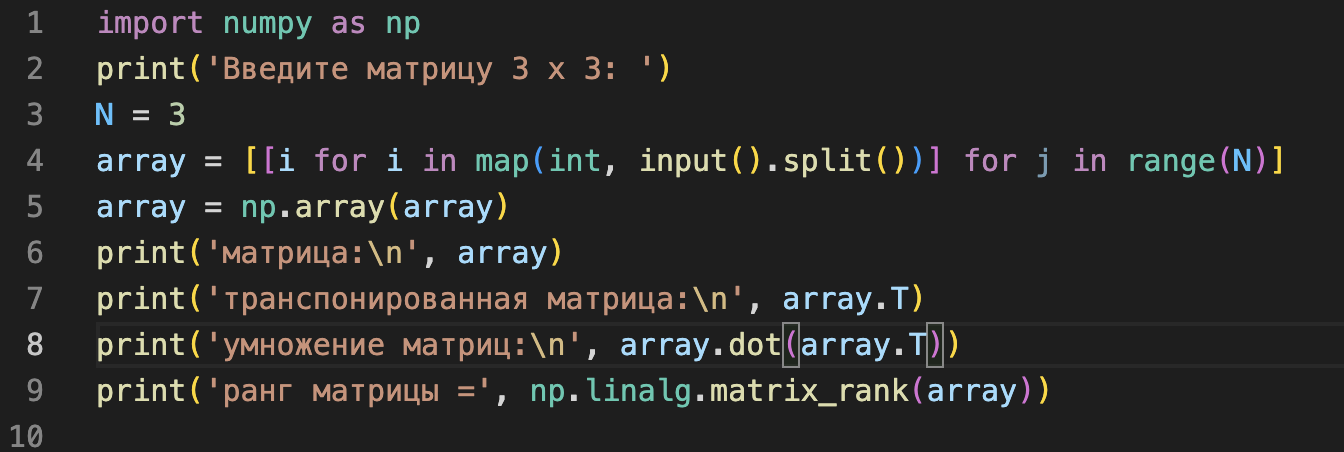


Рисунок 9 — Использование numpy

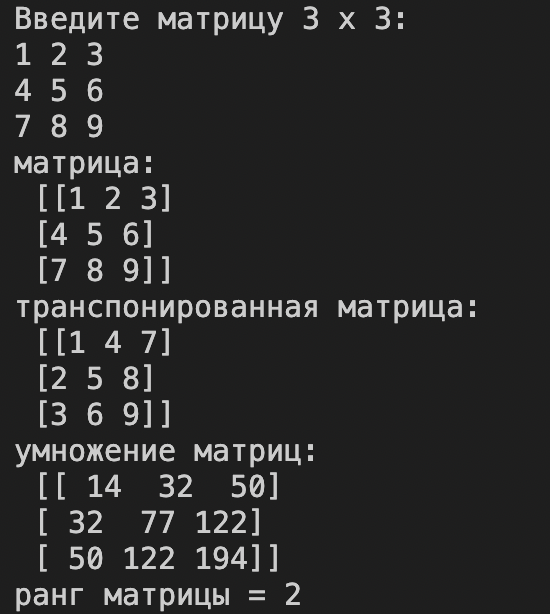


Рисунок 10 — Пример 2.1

**Достоинства и недостатки numpy:**

+ Огромный код, на который тратиться много времени и сил можно реализовать одним методом numpy. При этом скорость работы остается высокой.

- Сильно зависит от функций и библиотек, которые не относятся к Python.

# Задание 3

Задание: реализовать свою функцию для возведения матрицы A размерности 3x3 в степень -1, где входными переменными функции будут элементы матрицы А, вводимые вручную. С помощью библиотеки timeit сравнить быстродействие функции с ее аналогом из библиотеки numpy.

## Обратная матрица

Для вычисления обратной матрицы я использую следующую формулу:

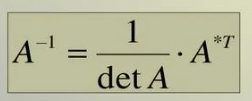


Рисунок 11 — Формула обратной матрицы

Реализуем функцию обратной матрицы array\_inv(), которая принимает уже транспонированную матрицу. Сначала она считает определитель матрицы 3-его порядка. Проверяет, чтобы определитель не равнялся 0. Переменная arr\_adj является союзной матрицей для исходной. Ее мы вычислили, используя определение:

Алгебраические дополнения равны минорам умноженным на *(-1)* в степени суммы номера строки и столбца элемента матрицы.

Для простоты можно использовать приведенную ниже схему знаков миноров

знаки миноров

Рисунок 12 — Схема знаков миноров

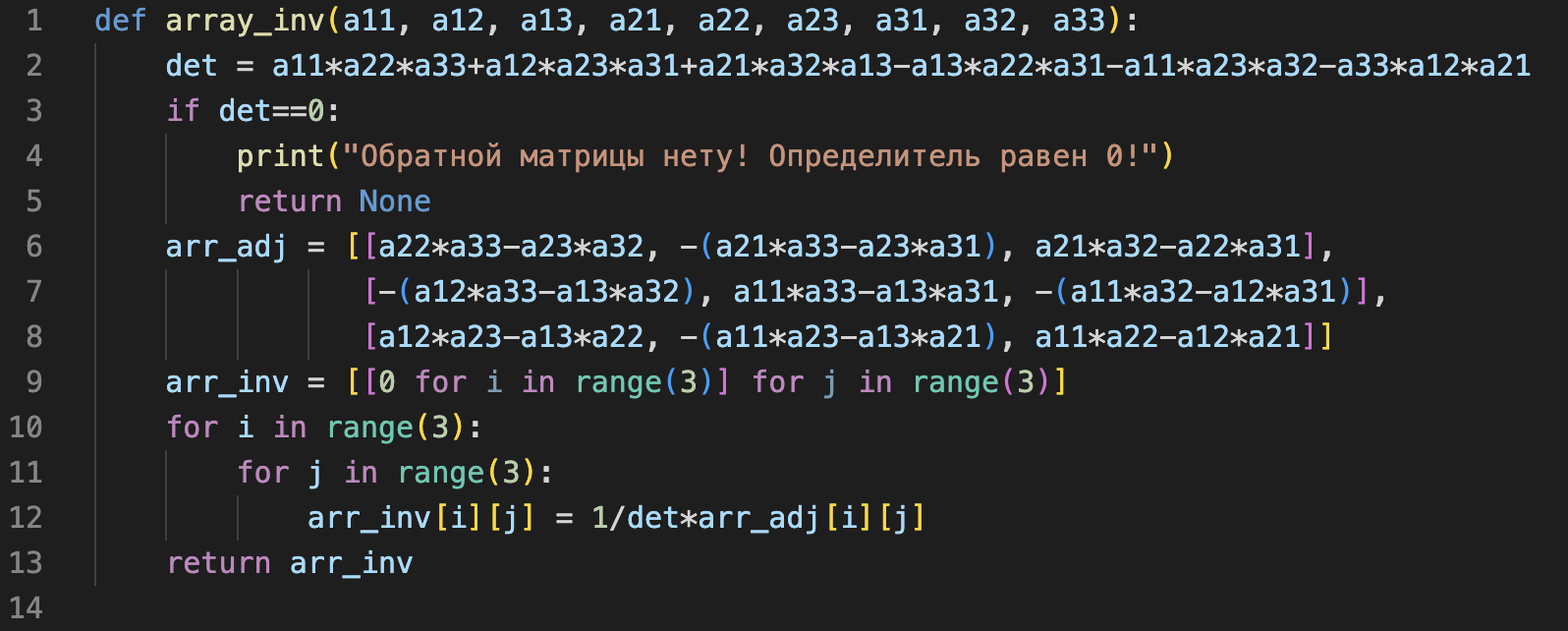


Рисунок 13 — Функция нахождения обратной матрицы

В конце работы функция возвращает обратную матрицу, вычисленную по формуле с Рисунка 11.

## Решение с numpy и сравнение времени работы

Теперь решим задачу с помощью numpy. Для этой задачи в numpy есть метод np.linalg.inv(A), которая считает обратную матрицу.

Теперь засечем время. Используем библиотеку timeit и введем переменные, в которых будет время старта и финиша работы. start принимает время начала отсчета, end — время конца отчета. После находим их разность и получаем результат в долях секунды. Выводим результаты работы программы и метода библиотеки numpy.

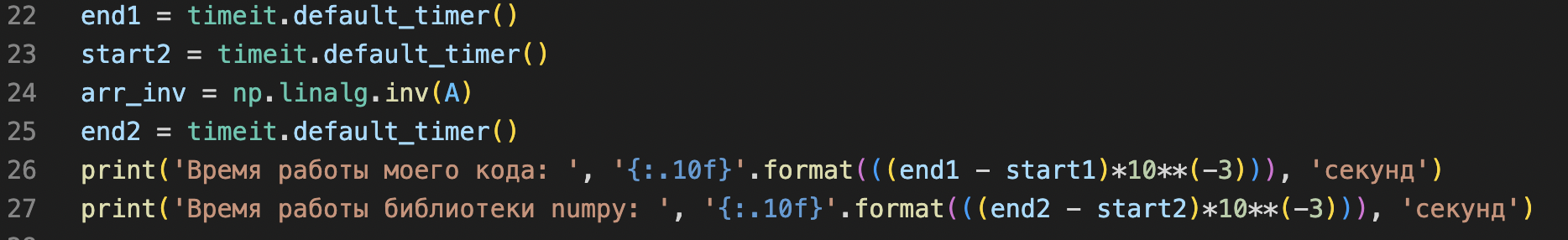


Рисунок 14 — Вычисление времени работы

Сравним время работы.

Ниже представлено два примера. Если проанализировать их, то можно понять, что время работы моего кода лучше на матрице 3 x 3, в среднем быстрее в 4-5 раз.

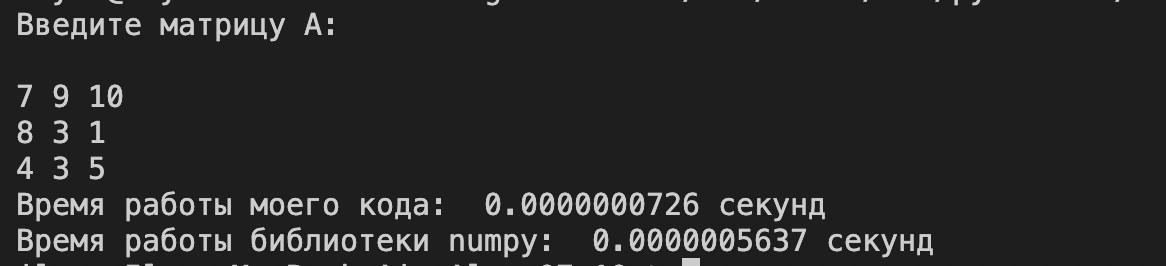


Рисунок 15 — Пример 3.1(Разница в 7 раз)

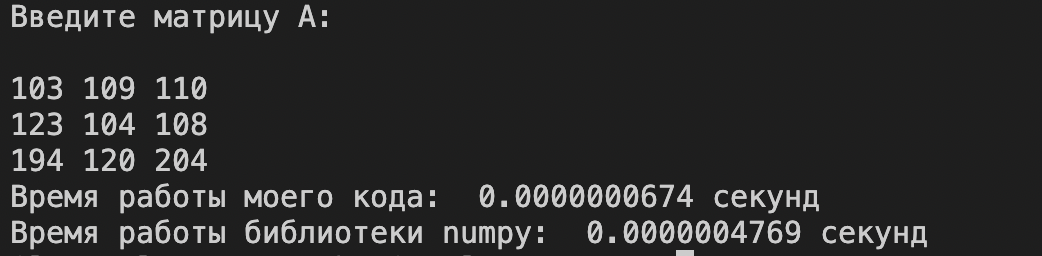


Рисунок 16 — Пример 3.2 (Разница в 8 раз)

# Заключение

В ходе работы была изучена работа в Python, в частности с матрицами. Также я ознакомился с библиотекой numpy. Я написал функции для транспонирования умножения матриц, для нахождения их ранга и обратной матрицы. Также выполнил те же задания с использованием numpy и сравнил два кода. Скажу, что numpy определенно выигрывает и в удобстве, и в быстроте.

Ссылка на GitHub: <https://github.com/estle/lab1ADs>

# Список использованных источников

1 Pythonist: офиц. сайт.URL: https://pythonist.ru/kak-transponirovat-matriczu-v-python/ (Дата обращения 06.10.2022).