Отчет по лабораторной работе №3

Компьютерный практикум по статистическому анализу данных

Амуничников Антон Игоревич

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Основная цель работы – освоить применение циклов функций и сторонних для Julia пакетов для решения задач линейной алгебры и работы с матрицами.

2 Задание

1. Используя Jupyter Lab, повторите примеры из раздела 3.2.
2. Выполните задания для самостоятельной работы (раздел 3.4)

3 Теоретическое введение

Julia – высокоуровневый свободный язык программирования с динамической типизацией, созданный для математических вычислений [**julialang**]. Эффективен также и для написания программ общего назначения. Синтаксис языка схож с синтаксисом других математических языков, однако имеет некоторые существенные отличия.

Для выполнения заданий была использована официальная документация Julia [**juliadoc**].

4 Выполнение лабораторной работы

Для начала выполним примеры из лабораторной работы, чтобы познакомиться с циклами, условными операторами, функциями и работой со сторонними библиотеками (рис. 4.1 - рис. 4.4).

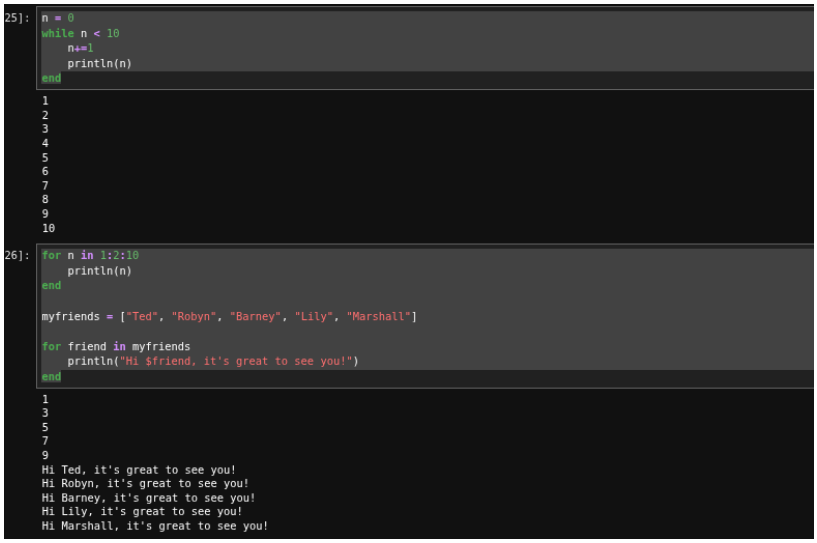

Рисунок 4.1: Выполнение примеров с циклами

```
[6]: # инициализация массива m x n из нулей:
     m, n = 5, 5
     A = fill(0, (m, n))
     # формирование массива, в котором значение каждой записи
     # является суммой индексов строки и столбца:
     for i in 1:m
         for j in 1:n
             A[i, j] = i + j
         end
     end
     print(A,"\n")
     # инициализация массива m x n из нулей:
     B = fill(0, (m, n))
     for i in 1:m, j in 1:n
         B[i, j] = i + j
     end
     print(B, "\n")

     C = [i + j for i in 1:m, j in 1:n]
     print(C, "\n")

     [2 3 4 5 6; 3 4 5 6 7; 4 5 6 7 8; 5 6 7 8 9; 6 7 8 9 10]
     [2 3 4 5 6; 3 4 5 6 7; 4 5 6 7 8; 5 6 7 8 9; 6 7 8 9 10]
     [2 3 4 5 6; 3 4 5 6 7; 4 5 6 7 8; 5 6 7 8 9; 6 7 8 9 10]

[7]: # используем `&&` для реализации операции "AND"
     # операция % вычисляет остаток от деления
     N = 15
     if (N % 3 == 0) && (N % 5 == 0)
         println("FizzBuzz")
     elseif N % 3 == 0
         println("Fizz")
     elseif N % 5 == 0
         println("Buzz")
     else
         println(N)
     end

     FizzBuzz
```

Рисунок 4.2: Выполнение примеров с условными выражениями

```
[8]: x = 5
     y = 10
     (x > y) ? x : y

[8]: 10

[9]: function sayhi(name)
         println("Hi $name, it's great to see you!")
     end
     # функция возведения в квадрат:
     function f(x)
         x^2
     end

[9]: f (generic function with 1 method)

10]: sayhi("C-3PO")
     f(42)

     Hi C-3PO, it's great to see you!
10]: 1764

[1]: import Pkg
     Pkg.add("Colors")
     using Colors

       Resolving package versions...
      No Changes to `~/.julia/environments/v1.11/Project.toml`
      No Changes to `~/.julia/environments/v1.11/Manifest.toml`
```

Рисунок 4.3: Выполнение примеров с функциями

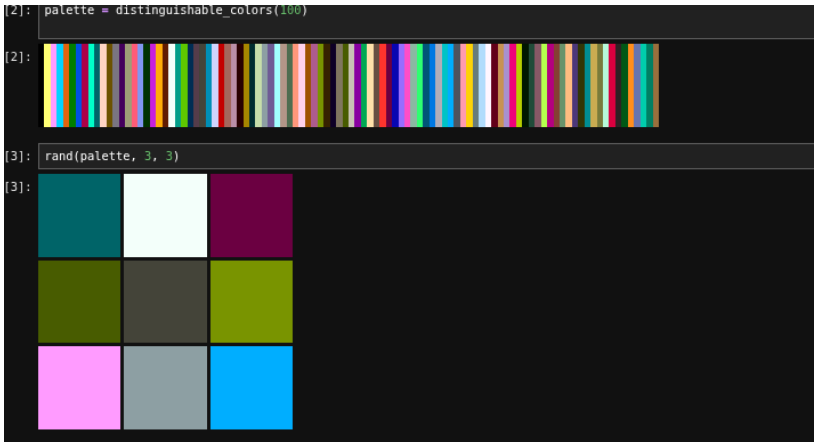

Рисунок 4.4: Выполнение примеров со сторонними библиотеками

Теперь перейдем к выполнению заданий для самостоятельной работы. Используя циклы while и for (рис. 4.5):

- выведем на экран целые числа от 1 до 100 и напечатаем их квадраты;
- создадим словарь squares, который будет содержать целые числа в качестве ключей и квадраты в качестве их пар-значений;
- создадим массив squares_arr, содержащий квадраты всех чисел от 1 до 100.

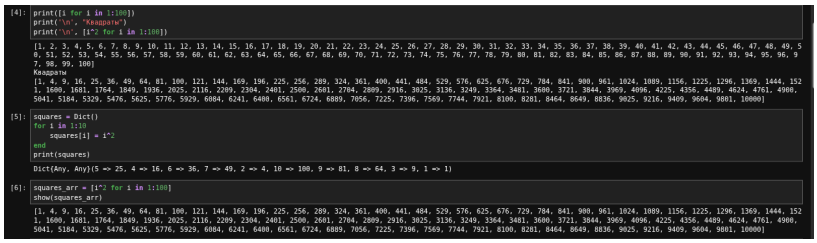

Рисунок 4.5: Задание №1

Напишем условный оператор, который печатает число, если число чётное, и строку «нечётное», если число нечётное. Перепишем код, используя тернарный оператор (рис. 4.6).

Рисунок 4.6: Задание №2

Напишем функцию add_one, которая добавляет 1 к своему входу (рис. 4.7).

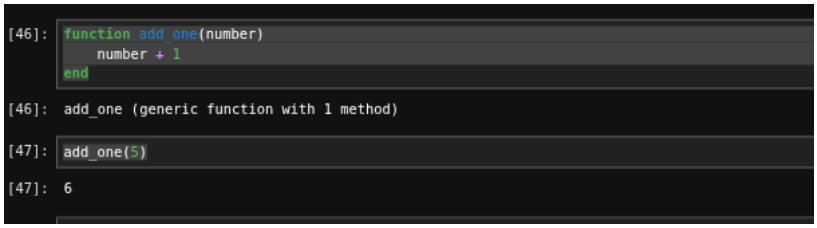

Рисунок 4.7: Задание №3

Используем map() или broadcast() для задания матрицы A, каждый элемент которой увеличивается на единицу по сравнению с предыдущим. (рис. 4.8)

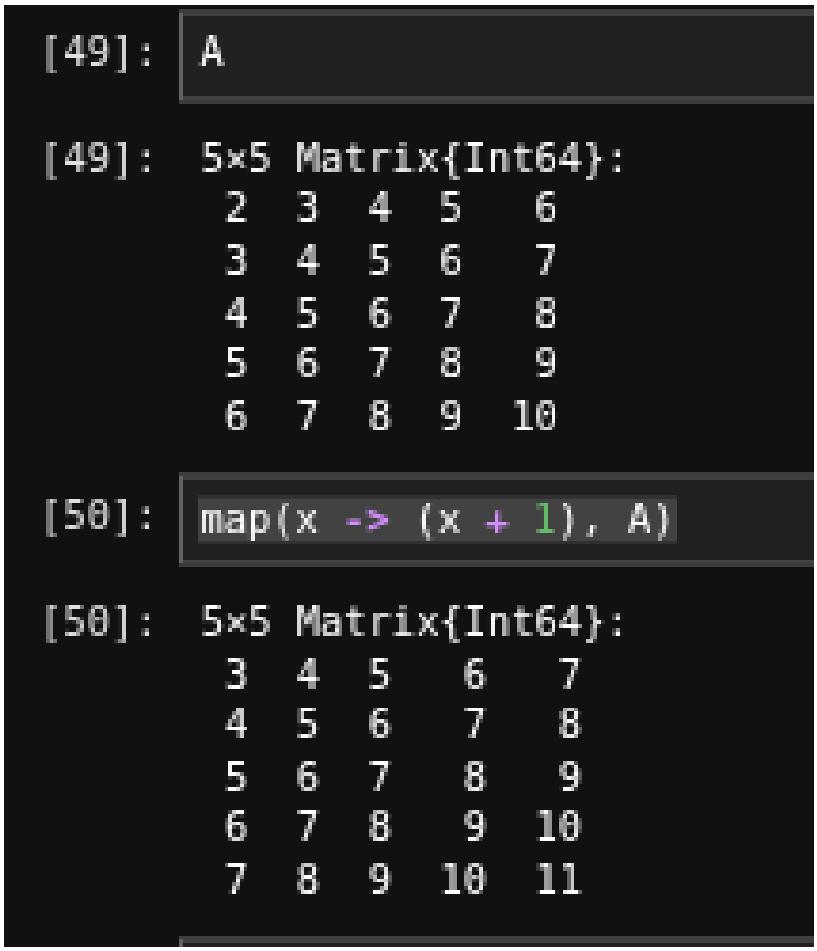

Рисунок 4.8: Задание №4

Зададим матрицу A. Найдем A^3. Заменим третий столбец матрицы A на сумму второго и третьего столбцов (рис. 4.9).

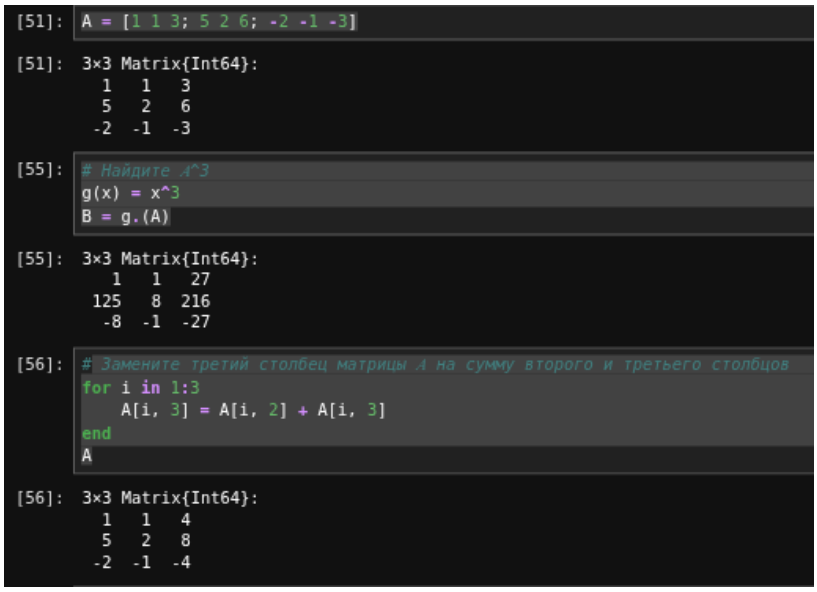

Рисунок 4.9: Задание №5

Создадим матрицу B с элементами $B_{i1} = 10$, $B_{i2} = -10$, $B_{i3} = 10$, $i = 1, 2, \dots, 15$. Вычислим матрицу $C = B^T B$ (рис. 4.10).

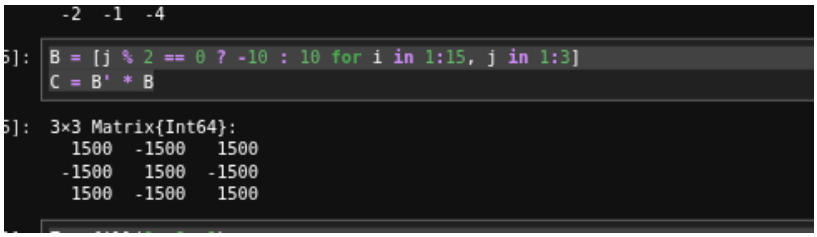

Рисунок 4.10: Задание №6

Создадим матрицу Z размерности 6×6, все элементы которой равны нулю, и матрицу E, все элементы которой равны 1. Используя цикл while или for и закономерности расположения элементов, создадим следующие матрицы размерности 6×6 (рис. 4.11).

Рисунок 4.11: Задание №7

Напишем свою функцию, аналогичную функции outer() языка R. Функция должна иметь следующий интерфейс: `outer(x,y,operation)` (рис. 4.12), (рис. 4.13).

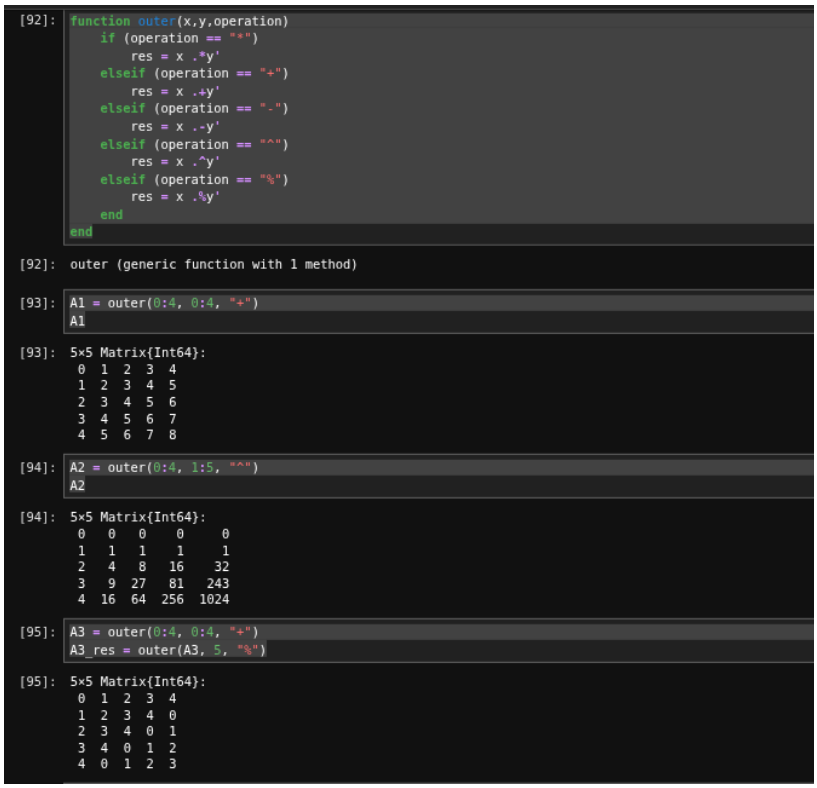

Рисунок 4.12: Задание №8. Реализация функции outer(). Проверка работы функции

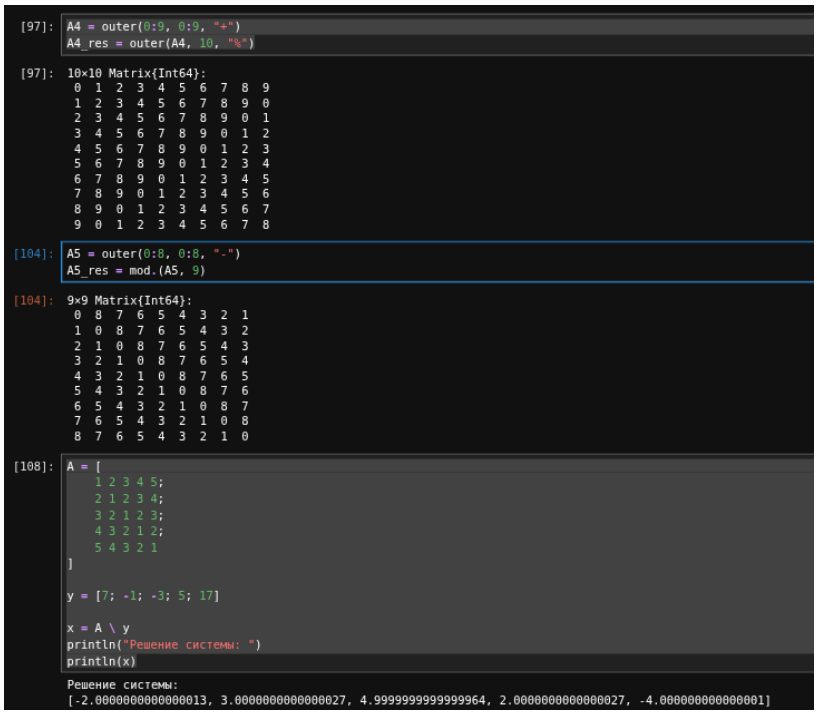

Рисунок 4.13: Задание №8. Проверка работы функции outer()

Решим систему линейных уравнений с 5 неизвестными (рис. 4.14).

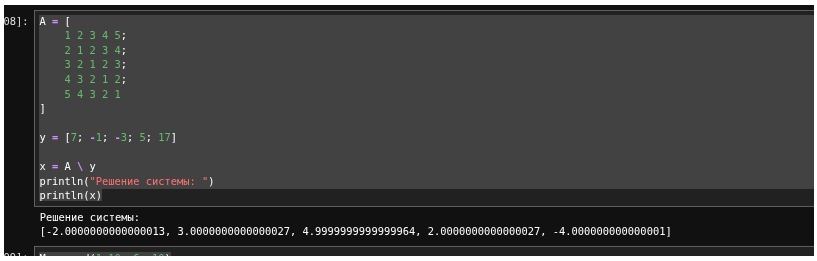

Рисунок 4.14: Задание №9. Решение систему линейных уравнений

В 10 задании произведем анализ количества элементов матрицы, удовлетворяющих необходимым условиям. Вычислим выражения (рис. 4.15).

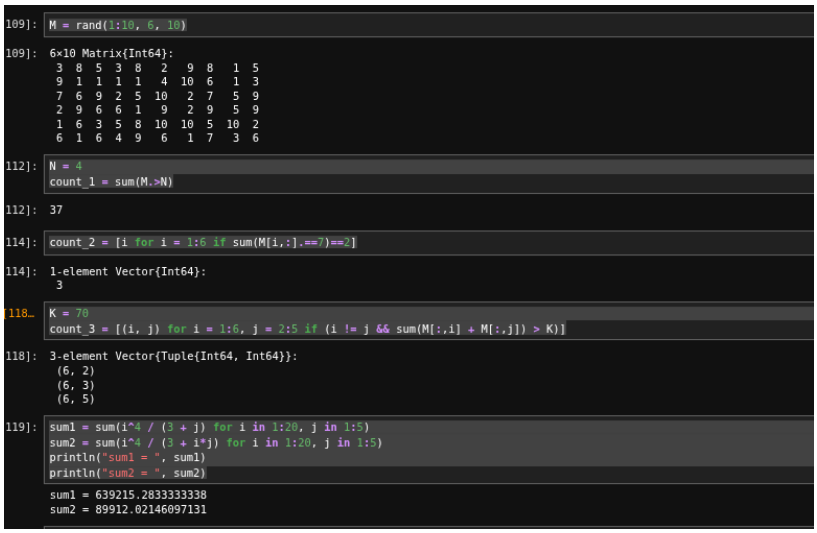

Рисунок 4.15: Задание №10. Задание №11

5 Выводы

В результате выполнения данной лабораторной работы я освоил применение циклов функций и сторонних для Julia пакетов для решение задач линейной алегбры и работы с матрицами.