

# **Отчет по лабораторной работе №3**

**Компьютерный практикум по статистическому анализу данных**

Амуничников Антон Игоревич

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>5</b>	<b>Выводы</b>	<b>18</b>

## Список иллюстраций

4.1	Выполнение примеров с циклами . . . . .	8
4.2	Выполнение примеров с условными выражениями . . . . .	9
4.3	Выполнение примеров с функциями . . . . .	9
4.4	Выполнение примеров со сторонними библиотеками . . . . .	10
4.5	Задание №1 . . . . .	10
4.6	Задание №2 . . . . .	11
4.7	Задание №3 . . . . .	11
4.8	Задание №4 . . . . .	12
4.9	Задание №5 . . . . .	13
4.10	Задание №6 . . . . .	13
4.11	Задание №7 . . . . .	14
4.12	Задание №8. Реализация функции <code>outer()</code> . Проверка работы функции . . . . .	15
4.13	Задание №8. Проверка работы функции <code>outer()</code> . . . . .	16
4.14	Задание №9. Решение систему линейных уравнений . . . . .	16
4.15	Задание №10. Задание №11 . . . . .	17

## **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).

```
25) n = 0
while n < 10
  n+=1
  println(n)
end

1
2
3
4
5
6
7
8
9
10

26) for n in 1:2:10
  println(n)
end

myfriends = ["Ted", "Robyn", "Barney", "Lily", "Marshall"]

for friend in myfriends
  println("Hi $friend, it's great to see you!")
end

1
3
5
7
9
Hi Ted, it's great to see you!
Hi Robyn, it's great to see you!
Hi Barney, it's great to see you!
Hi Lily, it's great to see you!
Hi Marshall, it's great to see you!
```

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-3P0")
f(42)

Hi C-3P0, 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

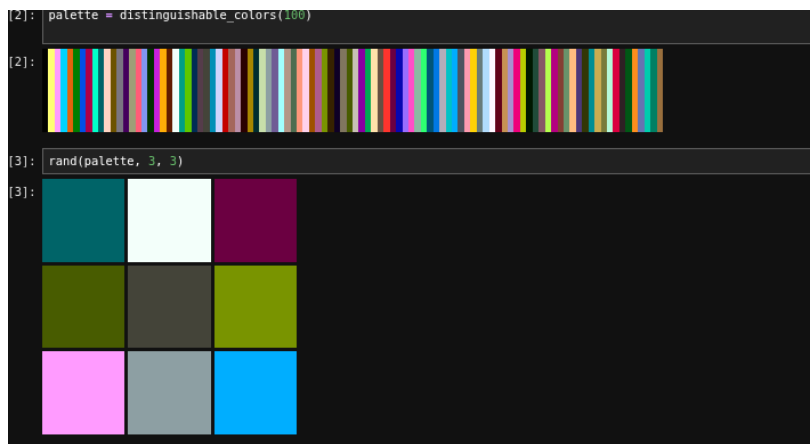


Рисунок 4.4: Выполнение примеров со сторонними библиотеками

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.

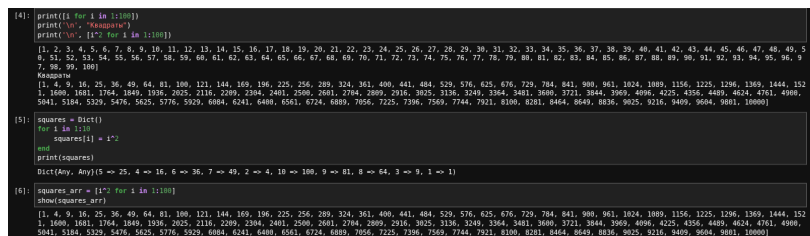


Рисунок 4.5: Задание №1

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

```

[10]: a = readline()
      a = parse(Int64, a)
      if ab == 0
        print(a)
      else
        print("nonetype")
      end
      stdin = 3
      nonetype

[9]: a = readline()
     a = parse(Int64, a)
     (a & 1 == 0) ? a : "nonetype"
     stdin = 4
[9]: 4

```

Рисунок 4.6: Задание №2

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

```

[46]: function add_one(number)
      number + 1
      end

[46]: add_one (generic function with 1 method)

[47]: add_one(5)

[47]: 6

```

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)

```
[49]: A
[49]: 5x5 Matrix{Int64}:
      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
[50]: map(x -> (x + 1), A)
[50]: 5x5 Matrix{Int64}:
      3  4  5  6  7
      4  5  6  7  8
      5  6  7  8  9
      6  7  8  9 10
      7  8  9 10 11
```

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).

```

[51]: A = [1 1 3; 5 2 6; -2 -1 -3]

[51]: 3x3 Matrix{Int64}:
      1  1  3
      5  2  6
     -2 -1 -3

[55]: # Найдите A^3
      g(x) = x^3
      B = g.(A)

[55]: 3x3 Matrix{Int64}:
      1  1  27
     125  8 216
     -8 -1 -27

[56]: # Замените третий столбец матрицы A на сумму второго и третьего столбцов
      for i in 1:3
          A[i, 3] = A[i, 2] + A[i, 3]
      end
      A

[56]: 3x3 Matrix{Int64}:
      1  1  4
      5  2  8
     -2 -1 -4

```

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).

```

      -2 -1 -4
5]: B = [j % 2 == 0 ? -10 : 10 for i in 1:15, j in 1:3]
      C = B' * B

5]: 3x3 Matrix{Int64}:
     1500 -1500  1500
    -1500  1500 -1500
     1500 -1500  1500

```

Рисунок 4.10: Задание №6

Создадим матрицу  $Z$  размерности  $6 \times 6$ , 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  $6 \times 6$  (рис. 4.11).

```

[81]: Z = fill(0, 6, 6)
      E = fill(1, 6, 6)

      Z1 = copy(Z)
      for i in 1:6, j in 1:6
          if abs(i - j) == 1
              Z1[i, j] = 1
          end
      end
      Z1

[81]: 6×6 Matrix{Int64}:
      0 1 0 0 0 0
      1 0 1 0 0 0
      0 1 0 1 0 0
      0 0 1 0 1 0
      0 0 0 1 0 1
      0 0 0 0 1 0

[83]: Z2 = copy(Z)
      for i in 1:6, j in 1:6
          if i == j || abs(i - j) == 2
              Z2[i, j] = 1
          end
      end
      Z2

[83]: 6×6 Matrix{Int64}:
      1 0 1 0 0 0
      0 1 0 1 0 0
      1 0 1 0 1 0
      0 1 0 1 0 1
      0 0 1 0 1 0
      0 0 0 1 0 1

[89]: Z3 = copy(Z)
      for i in 1:6, j in 1:6
          if (5 - i + 1) == j || abs(6 - i + 1 - j) == 2
              Z3[i, j] = 1
          end
      end
      Z3

[89]: 6×6 Matrix{Int64}:
      0 0 0 1 0 1
      0 0 1 0 1 0
      0 1 0 1 0 1
      1 0 1 0 1 0
      0 1 0 1 0 0
      1 0 1 0 0 0

[90]: Z4 = copy(Z)
      for i in 1:6, j in 1:6
          if (i + j) % 2 == 0
              Z4[i, j] = 1
          end
      end
      Z4

[90]: 6×6 Matrix{Int64}:
      1 0 1 0 1 0
      0 1 0 1 0 1
      1 0 1 0 1 0
      0 1 0 1 0 1
      1 0 1 0 1 0
      0 1 0 1 0 1

```

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).

```

[92]: function outer(x,y,operation)
      if (operation == "**")
          res = x .*y'
      elseif (operation == "+")
          res = x .+y'
      elseif (operation == "-")
          res = x .-y'
      elseif (operation == "^")
          res = x .^y'
      elseif (operation == "%")
          res = x .%y'
      end
      end

[92]: outer (generic function with 1 method)

[93]: A1 = outer(0:4, 0:4, "+")
      A1

[93]: 5×5 Matrix{Int64}:
      0 1 2 3 4
      1 2 3 4 5
      2 3 4 5 6
      3 4 5 6 7
      4 5 6 7 8

[94]: A2 = outer(0:4, 1:5, "^")
      A2

[94]: 5×5 Matrix{Int64}:
      0 0 0 0 0
      1 1 1 1 1
      2 4 8 16 32
      3 9 27 81 243
      4 16 64 256 1024

[95]: A3 = outer(0:4, 0:4, "+")
      A3_res = outer(A3, 5, "%")

[95]: 5×5 Matrix{Int64}:
      0 1 2 3 4
      1 2 3 4 0
      2 3 4 0 1
      3 4 0 1 2
      4 0 1 2 3

```

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

```

[97]: A4 = outer(0:9, 0:9, "+")
      A4_res = outer(A4, 10, "%")

[97]: 10x10 Matrix{Int64}:
      0 1 2 3 4 5 6 7 8 9
      1 2 3 4 5 6 7 8 9 0
      2 3 4 5 6 7 8 9 0 1
      3 4 5 6 7 8 9 0 1 2
      4 5 6 7 8 9 0 1 2 3
      5 6 7 8 9 0 1 2 3 4
      6 7 8 9 0 1 2 3 4 5
      7 8 9 0 1 2 3 4 5 6
      8 9 0 1 2 3 4 5 6 7
      9 0 1 2 3 4 5 6 7 8

[104]: A5 = outer(0:8, 0:8, "-")
      A5_res = mod.(A5, 9)

[104]: 9x9 Matrix{Int64}:
      0 8 7 6 5 4 3 2 1
      1 0 8 7 6 5 4 3 2
      2 1 0 8 7 6 5 4 3
      3 2 1 0 8 7 6 5 4
      4 3 2 1 0 8 7 6 5
      5 4 3 2 1 0 8 7 6
      6 5 4 3 2 1 0 8 7
      7 6 5 4 3 2 1 0 8
      8 7 6 5 4 3 2 1 0

[108]: A = [
      1 2 3 4 5;
      2 1 2 3 4;
      3 2 1 2 3;
      4 3 2 1 2;
      5 4 3 2 1
      ]

      y = [7; -1; -3; 5; 17]

      x = A \ y
      println("Решение системы: ")
      println(x)

      Решение системы:
      [-2.0000000000000013, 3.0000000000000027, 4.9999999999999964, 2.0000000000000027, -4.000000000000001]

```

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

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

```

[98]: A = [
      1 2 3 4 5;
      2 1 2 3 4;
      3 2 1 2 3;
      4 3 2 1 2;
      5 4 3 2 1
      ]

      y = [7; -1; -3; 5; 17]

      x = A \ y
      println("Решение системы: ")
      println(x)

      Решение системы:
      [-2.0000000000000013, 3.0000000000000027, 4.9999999999999964, 2.0000000000000027, -4.000000000000001]

```

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).



```

109]: M = rand(1:10, 6, 10)
109]: 6x10 Matrix{Int64}:
      3  8  5  3  8  2  9  8  1  5
      9  1  1  1  1  4  10  6  1  3
      7  6  9  2  5  10  2  7  5  9
      2  9  6  6  1  9  2  9  5  9
      1  6  3  5  8  10  10  5  10  2
      6  1  6  4  9  6  1  7  3  6

112]: N = 4
      count_1 = sum(M.>N)

112]: 37

114]: count_2 = [i for i = 1:6 if sum(M[i,:].==7)==2]

114]: 1-element Vector{Int64}:
      3

118_: K = 70
      count_3 = [(i, j) for i = 1:6, j = 2:5 if (i != j && sum(M[:,i] + M[:,j]) > K)]

118]: 3-element Vector{Tuple{Int64, Int64}}:
      (6, 2)
      (6, 3)
      (6, 5)

119]: sum1 = sum(i^4 / (3 + j) for i in 1:20, j in 1:5)
      sum2 = sum(i^4 / (3 + i*j) for i in 1:20, j in 1:5)
      println("sum1 = ", sum1)
      println("sum2 = ", sum2)

      sum1 = 639215.2833333338
      sum2 = 89912.62146097131

```

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 алгебры и работы с матрицами.