

Отчёт по лабораторной работе №3

Компьютерный практикум по статистическому анализу данных

Канева Екатерина, НФИбд-02-22

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическая часть	7
4	Выполнение лабораторной работы	8
5	Выводы	16

Список иллюстраций

4.1	Примеры с циклами (1).	8
4.2	Примеры с циклами (2).	9
4.3	Примеры с условными выражениями.	9
4.4	Примеры с функциями.	10
4.5	Примеры со сторонними библиотеками.	10
4.6	Задание 1.	11
4.7	Задание 2.	11
4.8	Задание 3.	11
4.9	Задание 4.	12
4.10	Задание 5.	12
4.11	Задание 6.	12
4.12	Задание 7(1).	13
4.13	Задание 7(2).	13
4.14	Задание 8(1).	14
4.15	Задание 8(2).	14
4.16	Задание 9.	14
4.17	Задание 10.	15
4.18	Задание 11.	15

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

- Используя Jupyter Lab, повторить примеры.
- Выполнить задания для самостоятельной работы.

3 Теоретическая часть

Julia - высокоуровневый свободный язык программирования с динамической типизацией, созданный для математических вычислений. 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.

4 Выполнение лабораторной работы

Сначала я выполнила примеры с циклами while и for (рис. 4.1, 4.2):

```
[1]: n = 0
    while n < 10
        n += 1
        println(n)
    end
1
2
3
4
5
6
7
8
9
10

[3]: myfriends = ["Ted", "Robyn", "Barney", "Lily", "Marshall"]
    i = 1
    while i <= length(myfriends)
        friend = myfriends[i]
        println("Hi $friend, it's great to see you!")
        i += 1
    end
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!

[5]: for n in 1:2:10
    println(n)
end
1
3
5
7
9

[7]: myfriends = ["Ted", "Robyn", "Barney", "Lily", "Marshall"]
    for friend in myfriends
        println("Hi $friend, it's great to see you!")
    end
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!
```

Рис. 4.1: Примеры с циклами (1).


```
[9]: m, n = 5, 5
A = fill(0, (m, n))

for i in 1:m
    for j in 1:n
        A[i, j] = i + j
    end
end

A

[9]: 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

[13]: B = fill(0, (m, n))
for i in 1:m, j in 1:n
    B[i, j] = i + j
end

B

[13]: 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

[15]: C = [i + j for i in 1:m, j in 1:n]
C

[15]: 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
```

Рис. 4.2: Примеры с циклами (2).

Потом я выполнила примеры с условными выражениями (рис. 4.3):

```
[17]: N = 12

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

Fizz

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

[22]: 10
```

Рис. 4.3: Примеры с условными выражениями.

Потом я выполнила примеры с функциями (рис. 4.4):

```

[25]: function f(x)
      x^2
      end

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

[27]: f(42)

[27]: 1764

[29]: f2(x) = x^2

[29]: f2(42)

[29]: 1764

[31]: f3 = x -> x^2

[31]: #3 (generic function with 1 method)

[33]: v = [3, 5, 2]
      sort(v)
      v

[33]: 3-element Vector{Int64}:
      3
      5
      2

[35]: sort!(v)
      v

[35]: 3-element Vector{Int64}:
      2
      3
      5

[37]: f(x) = x^2
      map(f, [1, 2, 3])

[37]: 3-element Vector{Int64}:
      1
      4
      9

[39]: f(x) = x^2
      broadcast(f, [1, 2, 3])

[39]: 3-element Vector{Int64}:
      1
      4
      9

```

Рис. 4.4: Примеры с функциями.

Потом я выполнила примеры со сторонними библиотеками (рис. 4.5):


```

[46]: import Pkg


[48]: Pkg.add("Colors")
      using Colors

      Resolving package versions...
      Installed ColorTypes v0.12.1
      Installed Reexport v1.2.2
      Installed FixedPointNumbers v0.8.5
      Installed Statistics v1.11.1
      Installed Colors v0.13.1
      Updating "C:\Users\ekaneva\.julia\environments\v1.11\Project.toml"
      [5ae59095] + Colors v0.13.1
      Updating "C:\Users\ekaneva\.julia\environments\v1.11\Manifest.toml"
      [3da082f7] + ColorTypes v0.12.1
      [5ae59095] + Colors v0.13.1
      [53c48c17] + FixedPointNumbers v0.8.5
      [189a3867] + Reexport v1.2.2
      [10745b16] + Statistics v1.11.1
      [37e2e46d] + LinearAlgebra v1.11.0
      [e6e0078] + CompilerSupportLibraries_jll v1.1.1+0
      [4536629a] + OpenBLAS_jll v0.3.27+1
      [8a850b90] + libblastrampoline_jll v5.11.0+0

[50]: palette = distinguishable_colors(100)

[50]: 

[54]: rand(palette, 3, 3)

[54]: 

```

Рис. 4.5: Примеры со сторонними библиотеками.

Далее я приступила к выполнению заданий для самостоятельной работы.

1. Используя циклы while и for.

- 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.

```
[13]: print([i for i in 1:100], '\n')
      print([i**2 for i in 1:100])

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100]
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900, 961, 1024, 1089, 1156, 1225, 1296, 1369, 1444, 1521, 1600, 1681, 1764, 1849, 1936, 2025, 2116, 2209, 2304, 2401, 2500, 2601, 2704, 2809, 2916, 3025, 3136, 3249, 3364, 3481, 3600, 3721, 3844, 3969, 4096, 4225, 4356, 4489, 4624, 4761, 4900, 5041, 5184, 5329, 5476, 5625, 5776, 5929, 6084, 6241, 6400, 6561, 6724, 6889, 7056, 7225, 7396, 7569, 7744, 7921, 8100, 8281, 8464, 8649, 8836, 9025, 9216, 9409, 9604, 9801, 10000]

[15]: squares = Dict{<type>{i => i^2 for i in 1:100}}
      print(squares)

Dict{<type>{5 => 25, 56 => 3136, 35 => 1225, 55 => 3025, 60 => 3600, 30 => 900, 32 => 1024, 6 => 36, 67 => 4489, 45 => 2025, 73 => 5329, 64 => 4096, 90 => 8100, 4 => 16, 13 => 169, 54 => 2916, 63 => 3969, 86 => 7396, 91 => 8281, 62 => 3844, 58 => 3364, 52 => 2704, 12 => 144, 28 => 784, 75 => 5625, 23 => 529, 92 => 8464, 41 => 1681, 43 => 1849, 11 => 121, 36 => 1296, 68 => 4624, 69 => 4761, 98 => 9604, 82 => 6724, 85 => 7225, 39 => 1521, 84 => 7056, 77 => 5929, 7 => 49, 25 => 625, 95 => 9025, 71 => 5041, 66 => 4356, 76 => 5776, 34 => 1156, 50 => 2500, 59 => 3481, 93 => 8649, 2 => 4, 10 => 100, 18 => 324, 26 => 676, 27 => 729, 42 => 1764, 87 => 7569, 100 => 10000, 79 => 6241, 16 => 256, 20 => 400, 81 => 6561, 19 => 361, 49 => 2401, 44 => 1936, 9 => 81, 31 => 961, 74 => 5476, 61 => 3721, 25 => 625, 841, 94 => 8836, 46 => 2116, 57 => 3249, 70 => 4900, 21 => 441, 38 => 1444, 88 => 7744, 78 => 6084, 72 => 5184, 24 => 576, 8 => 64, 17 => 289, 37 => 1369, 1 => 1, 53 => 2809, 22 => 484, 47 => 2209, 83 => 6889, 99 => 9801, 89 => 7921, 14 => 196, 3 => 9, 80 => 6400, 96 => 9216, 51 => 2601, 33 => 1089, 40 => 1600, 48 => 2304, 15 => 225, 65 => 4225, 97 => 9409}

[17]: squares_arr = [i^2 for i in 1:100]
      print(squares_arr)

[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900, 961, 1024, 1089, 1156, 1225, 1296, 1369, 1444, 1521, 1600, 1681, 1764, 1849, 1936, 2025, 2116, 2209, 2304, 2401, 2500, 2601, 2704, 2809, 2916, 3025, 3136, 3249, 3364, 3481, 3600, 3721, 3844, 3969, 4096, 4225, 4356, 4489, 4624, 4761, 4900, 5041, 5184, 5329, 5476, 5625, 5776, 5929, 6084, 6241, 6400, 6561, 6724, 6889, 7056, 7225, 7396, 7569, 7744, 7921, 8100, 8281, 8464, 8649, 8836, 9025, 9216, 9409, 9604, 9801, 10000]
```

Рис. 4.6: Задание 1.

2. Напишем условный оператор, который печатает число, если число чётное, и строку «нечётное», если число нечётное. Перепишем код, используя тернарный оператор (рис. 4.7):

```
[13]: a = parse{Int64, readline()}

      if a % 2 == 0
      print(a)
      else
      print("нечётное")
      end

      stdin> 5
      нечётное

[15]: a = parse{Int64, readline()}

      a % 2 == 0 ? print(a) : print("нечётное")

      stdin> 5
      нечётное
```

Рис. 4.7: Задание 2.

3. Напишем функцию add_one, которая добавляет 1 к своему входу (рис. 4.8):

```
[17]: function add_one(x)
      x + 1
      end

      add_one(8.9)

[17]: 9.9
```

Рис. 4.8: Задание 3.

4. Используем `map()` для задания матрицы A, каждый элемент которой увеличивается на единицу по сравнению с предыдущим. (рис. 4.9).

```
[23]: a = [1 2 3; 4 5 6; 7 8 9]
      A = map(add_one, a)

[23]: 3x3 Matrix{Int64}:
      2 3 4
      5 6 7
      8 9 10
```

Рис. 4.9: Задание 4.

5. Зададим матрицу A. Найдем A^3 . Заменяем третий столбец матрицы A на сумму второго и третьего столбцов (рис. 4.10):

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

[25]: 3x3 Matrix{Int64}:
      1 1 3
      5 2 6
      -2 -1 -3

[27]: A^3

[27]: 3x3 Matrix{Int64}:
      0 0 0
      0 0 0
      0 0 0

[29]: A[:, 3] = A[:, 3] + A[:, 2]
      A

[29]: 3x3 Matrix{Int64}:
      1 1 4
      5 2 8
      -2 -1 -4
```

Рис. 4.10: Задание 5.

6. Создадим матрицу B с элементами $B_{i1} = 10$, $B_{i2} = -10$, $B_{i3} = 10$, $i = \overline{1, 15}$. Вычислим матрицу $C = B^T B$. (рис. 4.11).

```
[39]: B = fill{10, (15, 3)}

      for i in 1:15
      B[i, 2] = -10
      end

[43]: C = B' * B

[43]: 3x3 Matrix{Int64}:
      1500 -1500 1500
      -1500 1500 -1500
      1500 -1500 1500
```

Рис. 4.11: Задание 6.

7. Создадим матрицу Z размерности 6x6, все элементы которой равны нулю, и матрицу E, все элементы которой равны 1. Используя цикл `while` или `for` и закономерности расположения элементов, создадим новые матрицы размерности 6x6 (рис. 4.12-4.13):

```

[153]: Z = fill(0, (6, 6))

[153]: 6x6 Matrix{Int64}:
 0 0 0 0 0
 0 0 0 0 0
 0 0 0 0 0
 0 0 0 0 0
 0 0 0 0 0
 0 0 0 0 0

[155]: E = fill(1, (6, 6))

[155]: 6x6 Matrix{Int64}:
 1 1 1 1 1
 1 1 1 1 1
 1 1 1 1 1
 1 1 1 1 1
 1 1 1 1 1
 1 1 1 1 1

[60]: Z1 = fill(0, (6, 6))

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

Z1

[60]: 6x6 Matrix{Int64}:
 0 1 0 0 0
 1 0 1 0 0
 0 1 0 1 0
 0 0 1 0 1
 0 0 0 1 0
 0 0 0 0 1

[62]: Z2 = fill(0, (6, 6))

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

Z2

```

Рис. 4.12: Задание 7(1).

```

[62]: 6x6 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

[66]: Z3 = fill(0, (6, 6))

for i in 1:6
    for j in 1:6
        if abs(i - 7 + j) == 2 || i == 7 - j
            Z3[i, j] = 1
        end
    end
end

Z3

[66]: 6x6 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

[68]: Z4 = fill(0, (6, 6))

for i in 1:6
    for j in 1:6
        if abs(i - j) == 2 || i == j || abs(i - j) == 4
            Z4[i, j] = 1
        end
    end
end

Z4

[68]: 6x6 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

```

Рис. 4.13: Задание 7(2).

8. В языке R есть функция `outer()`. Напишем свою функцию, аналогичную функции `outer()` языка R. Функция будет иметь следующий интерфейс: `outer(x,y,operation)`. Используя написанную функцию `outer()`, создадим новые матрицы (рис. 4.14-4.15):

```
[62]: 6x6 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

[66]: Z3 = fill{0, (6, 6)}

      for i in 1:6
      for j in 1:6
      if abs(i - 7 + j) == 2 || i == 7 - j
      Z3[i, j] = 1
      end
      end
      end
      Z3

[66]: 6x6 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

[68]: Z4 = fill{0, (6, 6)}

      for i in 1:6
      for j in 1:6
      if abs(i - j) == 2 || i == j || abs(i - j) == 4
      Z4[i, j] = 1
      end
      end
      end
      Z4

[68]: 6x6 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
```

Рис. 4.14: Задание 8(1).

```
[84]: A4 = outer{0:9, 0:9, (i, j) -> (i + j) % 10}

[84]: 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

[88]: A5 = outer{0:8, 0:8, (i, j) -> (i - j + 9) % 9}

[88]: 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
```

Рис. 4.15: Задание 8(2).

9. Решим систему линейных уравнений с 5 неизвестными, рассмотрев соответствующее матричное уравнение $Ax = y$ (рис. 4.10):

```
[110]: A = fill{0, (5, 5)}

      for i in 1:5
      for j in 1:5
      A[i, j] = abs(i - j) + 1
      end
      end
      A

[110]: 5x5 Matrix{Int64}:
      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

[112]: y = [7, -1, -3, 5, 17];

[116]: print(A \ y)

[-2.00000000000000036, 3.00000000000000058, 4.9999999999999998, 1.9999999999999991, -3.9999999999999999]
```

Рис. 4.16: Задание 9.

10. Создадим матрицу М размерности 6x10, элементами которой являются целые числа, выбранные случайным образом с повторениями из совокупности 1-10. Найдём число элементов в каждой строке матрицы М, которые больше числа N (например, N = 4), определим, в каких строках матрицы М число М (например, M = 7) встречается ровно 2 раза, определим все пары столбцов матрицы М, сумма элементов которых больше K (например, K = 75) (рис. 4.10):

```
[123]: M = rand(1:10, 6, 10)

N = 4
Me = 7
K = 75;

[125]: print(sum(M.>N))
37

[133]: print([i for i in 1:6 if sum(M[i,:]>=7) == 2])
[1]

[143]: print([(i, j) for i in 1:6, j in 1:5 if (i != j && sum(M[i,i] + M[i,j])>K)])
[(1, 2), (3, 2), (4, 2), (5, 2), (6, 2), (1, 3), (2, 3), (4, 3), (5, 3), (6, 3), (2, 4), (3, 4), (2, 5), (3, 5)]
```

Рис. 4.17: Задание 10.

11. Вычислим суммы (рис. 4.10):

```
[145]: sum1 = sum(i^4 / (3 + j) for i in 1:20, j in 1:5)

[145]: 639215.2833333338

[147]: sum1 = sum(i^4 / (3 + j*i) for i in 1:20, j in 1:5)

[147]: 89912.02146097131
```

Рис. 4.18: Задание 11.

5 Выводы

Освоила применение циклов функций и сторонних для Julia пакетов для решения задач линейной алгебры и работы с матрицами.