

Лабораторная работы №3

Управляющие структуры

Кузнецова С. В.

9 октября 2025

Российский университет дружбы народов, Москва, Россия

Информация

- Кузнецова София Вадимовна
- Российский университет дружбы народов

Цель лабораторной работы

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

Выполнение лабораторной работы

Циклы while и for

```
[1]: # пока n<10 прибавить к n единицу и распечатать значение:  
n = 0  
while n < 10  
    n += 1  
    println(n)  
end  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
  
[5]: # Демонстрация использования while при работе со строковыми элементами массива  
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!
```

Рис. 1: Примеры использования цикла while

Циклы while и for

```
[7]: for n in 1:2:10
      println(n)
end
```

```
1
3
5
7
9
```

```
[11]: # Рассмотренные выше примеры, но с использованием цикла for:
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. 2: Примеры использования цикла for

Циклы while и for

```
[13]: # 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  
A
```

```
[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]: # 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  
B
```

```
[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
```

```
[17]: # Еще одна реализация этого же примера:  
C = [i + j for i in 1:m, j in 1:n]  
C
```

```
[17]: 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. 3: Пример использования цикла for для создания двумерного массива

Условные выражения

```
[19]: # используем `&&` для реализации операции "AND"  
# операция % вычисляет остаток от деления  
  
N = 100  
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  
  
Buzz
```

```
[21]: # Пример использования тернарного оператора:  
x = 5  
y = 10  
  
(x > y) ? x : y
```

```
[21]: 10
```

Рис. 4: Примеры использования условного выражения

ФУНКЦИИ

```
[35]: # Первый требует ключевых слов function и end:  
function sayhi(name)  
    println("Hi $name, it's great to see you!")  
end  
# функция возведения в квадрат:  
function f(x)  
    x^2  
end  
# Вызов функции осуществляется по её имени с указанием аргументов, например:  
sayhi("C-3PO")  
f(42)
```

Hi C-3PO, it's great to see you!

```
[35]: 1764
```

```
[33]: # В качестве альтернативы, можно объявить любую из выше определённых функций в одной  
# строке:  
|  
sayhi2(name) = println("Hi $name, it's great to see you!")  
f2(x) = x^2  
  
sayhi("C-3PO")  
f(42)
```

Hi C-3PO, it's great to see you!

```
[33]: 1764
```

```
[31]: # Наконец, можно объявить выше определённые функции как «анонимные»:  
sayhi3 = name -> println("Hi $name, it's great to see you!")  
f3 = x -> x^2  
  
sayhi("C-3PO")  
f(42)
```

Hi C-3PO, it's great to see you!

```
[31]: 1764
```

Рис. 5: Примеры способов написания функции

ФУНКЦИИ

```
[37]: # задаём массив v:  
v = [3, 5, 2]  
sort(v)  
v
```

```
[37]: 3-element Vector{Int64}:  
      3  
      5  
      2
```

```
[39]: sort!(v)  
v
```

```
[39]: 3-element Vector{Int64}:  
      2  
      3  
      5
```

Рис. 6: Сравнение результатов вывода

ФУНКЦИИ

```
[45]: # В Julia функция map является функцией высшего порядка, которая принимает функцию  
# в качестве одного из своих входных аргументов и применяет эту функцию к каждому  
# элементу структуры данных, которая ей передаётся также в качестве аргумента:
```

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

```
[45]: 3-element Vector{Int64}:  
1  
4  
9
```

```
[47]: # Функция broadcast – ещё одна функция высшего порядка в Julia, представляющая собой  
# обобщение функции map. Функция broadcast() будет пытаться привести все объекты  
# к общему измерению, map() будет напрямую применять данную функцию поэлементно.  
f(x) = x^2  
broadcast(f, [1, 2, 3])
```

```
[47]: 3-element Vector{Int64}:  
1  
4  
9
```

Рис. 7: Примеры использования функций map() и broadcast()

Сторонние библиотеки (пакеты) в Julia

```
[50]: # добавим и загрузим пакет Colors:  
import Pkg  
Pkg.add("Colors")  
using Colors  
# Затем создадим палитру из 100 разных цветов:  
palette = distinguishable_colors(100)  
  
rand(palette, 3, 3)
```

```
Resolving package versions...  
No Changes to `C:\Users\sofik\.julia\environments\v1.11\Project.toml`  
No Changes to `C:\Users\sofik\.julia\environments\v1.11\Manifest.toml`
```

```
[50]:
```

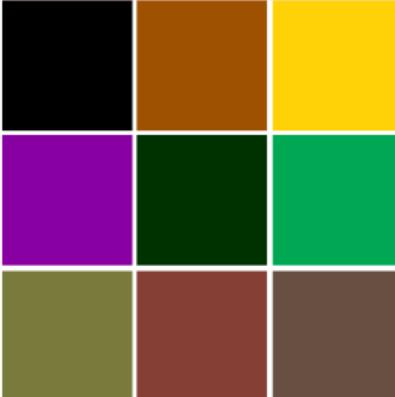


Рис. 8: Пример использования сторонних библиотек

Самостоятельная работа

```
[52]: # while

n = 1
while n <= 100
    println("$n^2 = $(n^2)")
    n += 1
end

1^2 = 1
2^2 = 4
3^2 = 9
4^2 = 16
5^2 = 25
6^2 = 36
7^2 = 49
8^2 = 64
9^2 = 81
10^2 = 100
11^2 = 121
12^2 = 144
13^2 = 169
...  ...
```

Рис. 9: Выполнение подпунктов задания №1

Самостоятельная работа

```
[56]: # for  
  
for n in 1:100  
    println("$n^2 = $(n^2)")  
end  
  
1^2 = 1  
2^2 = 4  
3^2 = 9  
4^2 = 16  
5^2 = 25  
6^2 = 36  
7^2 = 49  
8^2 = 64  
9^2 = 81  
10^2 = 100  
11^2 = 121  
12^2 = 144  
13^2 = 169  
14^2 = 196  
15^2 = 225  
16^2 = 256  
17^2 = 289  
18^2 = 324  
19^2 = 361  
20^2 = 400  
21^2 = 441  
22^2 = 484  
23^2 = 529  
...
```

Рис. 10: Выполнение подпунктов задания №1

Самостоятельная работа

```
[62]: squares = Dict()
for n in 1:100
    squares[n] = n^2
end
println(squares)
```

```
Dict{Any, Any}(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 => 12
1, 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, 29 =>
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 => 940
9)
```

Самостоятельная работа

```
[65]: squares_arr = [n^2 for n in 1:100]
println(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, 136
9, 1444, 1521, 1600, 1681, 1764, 1849, 1936, 2025, 2116, 2209, 2304, 2401, 2500, 2601, 27
04, 2809, 2916, 3025, 3136, 3249, 3364, 3481, 3600, 3721, 3844, 3969, 4096, 4225, 4356, 4
489, 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. 12: Выполнение подпунктов задания №1

Самостоятельная работа

```
[71]: # условный оператор
```

```
n = 50
if n % 2 == 0
    println(n)
else
    println("нечётное")
end
```

```
50
```

```
[73]: # тернарный оператор
```

```
println(n % 2 == 0 ? n : "нечётное")
```

```
50
```

Рис. 13: Выполнение задания №2

Самостоятельная работа

```
[77]: function add_one(x)
        return x + 1
    end
    println(add_one(10))
```

11

Рис. 14: Выполнение задания №3

Самостоятельная работа

```
[81]: # map
```

```
A = reshape(1:9, 3, 3)
B = map(x -> x + 1, A)
println(B)
```

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

```
[85]: # broadcast
```

```
B = broadcast(x -> x + 1, A)
println(B)
```

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

Рис. 15: Выполнение задания №4

Самостоятельная работа

```
[87]: # Определение матрицы  
A = [1 1 3; 5 2 6; -2 -1 -3]
```

```
# Вычисление A в 3 степени  
println(map(x -> x^3, A))
```

```
[1 1 27; 125 8 216; -8 -1 -27]
```

```
[89]: # Замена третьего столбца матрицы A на сумму второго и третьего столбцов
```

```
A[:, 3] = A[:, 2] + A[:, 3]  
println(A)
```

```
[1 1 4; 5 2 8; -2 -1 -4]
```

Рис. 16: Выполнение задания №5

Самостоятельная работа

```
[93]: # Создание матрицы B  
B = repeat([10 -10 10], 11, 1)
```

```
[93]: 11x3 Matrix{Int64}:  
10  -10   10  
10  -10   10  
10  -10   10  
10  -10   10  
10  -10   10  
10  -10   10  
10  -10   10  
10  -10   10  
10  -10   10  
10  -10   10  
10  -10   10
```

```
[95]: # Вычисление матрицы C как B' * B  
C = B' * B  
println(C)
```

```
[1100 -1100 1100; -1100 1100 -1100; 1100 -1100 1100]
```

Рис. 17: Выполнение задания №6

Самостоятельная работа

```
[107]: # ---- Параметры ----
n = 6
Z = zeros(Int, n, n)

# ---- функция show_matrix (база "Matrix(...)" ----
function show_matrix(title, M)
    println("Matrix $title:")
    for i in 1:size(M,1)
        println(join(M[i, :], " "))
    end
    println()
end

# показать квадратную матрицу Z
show_matrix("Z (квадратная матрица)", Z)

# ---- Z1: единицы на соседних диагоналях (sub- и super-диагонали) ----
Z1 = zeros(Int, n, n)
for i in 1:n, j in 1:n
    if abs(i - j) == 1
        Z1[i, j] = 1
    end
end
show_matrix("Z1", Z1)

# ---- Z2: единицы, где |i-j| == 0 или 2 (гладкая диагональ и диагонали со смещением ±2) ----
Z2 = zeros(Int, n, n)
for i in 1:n, j in 1:n
    if abs(i - j) == 0 || abs(i - j) == 2
        Z2[i, j] = 1
    end
end
show_matrix("Z2", Z2)

# ---- Z3: единицы на побочной диагонали и на диагоналях, отстоящих на 2 (т.е. i+j = 5,7,9 для n=6) ----
Z3 = zeros(Int, n, n)
for i in 1:n, j in 1:n
    # для обесц. в условие будем abs(i + j - (n+1)) == 0 или 2
    if abs(i + j - (n + 1)) == 0 || abs(i + j - (n + 1)) == 2
        Z3[i, j] = 1
    end
end
show_matrix("Z3", Z3)

# ---- Z4: "антидоминант" = 1, если |i-j| чётно ----
Z4 = zeros(Int, n, n)
for i in 1:n, j in 1:n
    if iseven(i + j)
        Z4[i, j] = 1
    end
end
show_matrix("Z4", Z4)
```

Рис. 18: Выполнение задания №7

Самостоятельная работа

Матрица Z (нулевая матрица) =

```
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
```

Матрица Z1 =

```
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
```

Матрица Z2 =

```
1 0 1 0 0  
0 1 0 1 0  
1 0 1 0 1  
0 1 0 1 0  
0 0 1 0 1  
0 0 0 1 0
```

Матрица Z3 =

```
0 0 0 1 0  
0 0 1 0 1  
0 1 0 1 0  
1 0 1 0 1  
0 1 0 1 0  
1 0 1 0 0
```

Матрица Z4 =

```
1 0 1 0 1  
0 1 0 1 0  
1 0 1 0 1  
0 1 0 1 0  
1 0 1 0 1  
0 1 0 1 0
```

Рис. 19: Выполнение задания №7

Самостоятельная работа

```
[109]: function outer(x, y, operation)
        return [operation(xi, yj) for xi in x, yj in y]
end

[109]: outer (generic function with 1 method)
```

Рис. 20: Выполнение подпунктов задания №8

Самостоятельная работа

```
[111]: # Матрица A1: сложение элементов
A1 = outer(0:4, 0:4, +)

# Матрица A2: возведение в степень
function safe_pow(x, y)
    x == 0 && y == 0 ? 0 : x^y
end

# Матрица A2: с пропуском первого элемента в каждой строке
A2 = [j == 1 ? i : safe_pow(i, j) for i in 0:4, j in 1:5]

# Матрица A3: циклический сдвиг по модулю 5
A3 = outer(0:4, 0:4, (x,y) -> mod(x + y, 5))

# Матрица A4: циклический сдвиг по модулю 10
A4 = outer(0:9, 0:9, (x,y) -> mod(x + y, 10))

# Матрица A5: разница по модулю 9
A5 = outer(0:8, 0:8, (x,y) -> mod(x - y, 9))

# Функция для вывода матриц
function print_matrix(name, mat)
    println("\nМатрица $name:")
    for row in eachrow(mat)
        println(row)
    end
end

# Печать всех матриц
print_matrix("A1", A1)
print_matrix("A2", A2)
print_matrix("A3", A3)
print_matrix("A4", A4)
print_matrix("A5", A5)
```

Рис. 21: Выполнение подпунктов задания №8

Самостоятельная работа

Матрица A1:

```
[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]
```

Матрица A2:

```
[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]
```

Матрица A3:

```
[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]
```

Матрица A4:

```
[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]
```

Матрица A5:

```
[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. 22: Выполнение подпунктов задания №8

Самостоятельная работа

```
[113]: 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]
b = [7, -1, -3, 5, 17]

# Решение системы
x = A \ b
println(x)

[-1.999999999999987, 2.999999999999996, 4.999999999999998, 2.000000000000001, -4.0]
```

Рис. 23: Выполнение задания №9

Самостоятельная работа

```
[131]: function print_matrix(name, mat)
    println("\nМатрица $name:")
    for row in eachrow(mat)
        println("[ " * join(row, ", ") * "]")
    end
end

M = rand(1:10, 6, 10)

print_matrix("M", M)
```

Матрица M:

```
[9, 7, 5, 6, 2, 5, 6, 1, 10, 6]
[10, 9, 5, 1, 9, 9, 1, 2, 2, 6]
[7, 1, 7, 3, 10, 6, 2, 6, 2, 2]
[6, 5, 9, 7, 10, 4, 2, 1, 6, 8]
[1, 7, 3, 3, 4, 1, 7, 8, 4, 5]
[10, 8, 3, 7, 6, 1, 2, 1, 1, 8]
```

Самостоятельная работа

```
[135]: N = 4  
greater_than_N = sum(M .> N, dims=2)  
println(greater_than_N)  
  
[8; 6; 5; 7; 4; 5;;]
```

```
[138]: M_value = 7  
rows_with_M_twice = findall(x -> count==(M_value), x) == 2, eachrow(M))  
println(rows_with_M_twice)  
  
[3, 5]
```

```
[143]: K = 75  
col_pairs = []  
for i in 1:size(M, 2)-1  
    for j in i+1:size(M, 2)  
        if sum(M[:,i] .* M[:,j]) > K  
            push!(col_pairs, (i, j))  
        end  
    end  
end  
println(col_pairs)
```



```
Any[(1, 2), (1, 5), (1, 10), (2, 5), (5, 10)]
```

Рис. 25: Выполнение подпунктов задания №10

Самостоятельная работа

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

21679980

```
[147]: sum2 = sum(i^4 * (3 + i * j) for i in 1:20 for j in 1:5)  
println(sum2)
```

195839490

Рис. 26: Выполнение задания №11

Вывод

Вывод

- В ходе выполнения лабораторной работы было освоено применение циклов функций и сторонних для Julia пакетов для решения задач линейной алгебры и работы с матрицами.

Список литературы. Библиография

Список литературы. Библиография

[1] Julia Documentation: <https://docs.julialang.org/en/v1/>

Спасибо за внимание!