

# Лабораторная работа № 2

Управляющие структуры

---

Шияпова Д.И.

10 октября 2025

Российский университет дружбы народов, Москва, Россия

- Шияпова Дарина Илдаровна
- Студентка
- Российский университет дружбы народов
- 1132226458@pfur.ru



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

# Выполнение лабораторной работы

```
[ ] # 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
```

```
[ ] 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
```

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

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

```
1  
3  
5  
7  
9
```

## Выполнение лабораторной работы

```
N = 33
# используем `&&` для реализации операции "AND"
# операция % вычисляет остаток от деления
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

```
x = 5
y = 10
(x > y) ? x : y
```

→ 10

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

# Выполнение лабораторной работы

```
import Pkg
Pkg.add("Example")

Updating registry at `~/.julia/registries/General.toml`
Resolving package versions...
Installed Example - v0.5.5
Updating `~/.julia/environments/v1.11/Project.toml`
[7876af07] + Example v0.5.5
Updating `~/.julia/environments/v1.11/Manifest.toml`
[7876af07] + Example v0.5.5
Precompiling project...
3987.0 ms ✓ Example
1 dependency successfully precompiled in 14 seconds. 487 already precompiled.

Pkg.add("Colors")
using Colors

palette = distinguishable_colors(100)

Resolving package versions...
Updating `~/.julia/environments/v1.11/Project.toml`
[5ae59095] + Colors v0.13.1
No Changes to `~/.julia/environments/v1.11/Manifest.toml`
```




Рис. 3: Выполнение примеров со сторонними библиотеками

# Выполнение лабораторной работы

```
# Задание 1. Использование циклов while и for

# Вывод чисел от 1 до 100 и их квадратов с помощью цикла while
println("Числа и их квадраты (while цикл):")
i = 1
while i <= 100
    println("Число: $i, Квадрат: $(i^2)")
    i += 1
end

println("\n" * "="^50 * "\n")
```

Числа и их квадраты (while цикл):

Число: 1, Квадрат: 1  
Число: 2, Квадрат: 4  
Число: 3, Квадрат: 9  
Число: 4, Квадрат: 16  
Число: 5, Квадрат: 25  
Число: 6, Квадрат: 36  
Число: 7, Квадрат: 49  
Число: 8, Квадрат: 64  
Число: 9, Квадрат: 81  
Число: 10, Квадрат: 100  
Число: 11, Квадрат: 121  
Число: 12, Квадрат: 144  
Число: 13, Квадрат: 169  
Число: 14, Квадрат: 196  
Число: 15, Квадрат: 225  
Число: 16, Квадрат: 256  
Число: 17, Квадрат: 289  
Число: 18, Квадрат: 324  
Число: 19, Квадрат: 361

```
# Задание 1 Вывод чисел от 1 до 100 и их квадратов с помощью цикла for
println("Числа и их квадраты (for цикл):")
for j in 1:100
    println("Число: $j, Квадрат: $(j^2)")
end

println("\n" * "="^50 * "\n")
```

Рис. 5: Задание №1



# Выполнение лабораторной работы

```
# Задание 1 Создание словаря squares
println("Создание словаря squares:")
squares = Dict{Int, Int}{}
for num in 1:100
    squares[num] = num^2
end

# Вывод первых 10 элементов словаря для проверки
println("Первые 10 элементов словаря squares:")
for (key, value) in collect(squares)[1:10]
    println("$key => $value")
end
```

```
→ Создание словаря squares:
Первые 10 элементов словаря squares:
5 => 25
56 => 3136
35 => 1225
55 => 3025
60 => 3600
30 => 900
32 => 1024
6 => 36
67 => 4489
45 => 2025
```

```
# Задание 1 Создание массива squares_arr
println("Создание массива squares_arr:")
squares_arr = Int64[]
for num in 1:100
    push!(squares_arr, num^2)
end

# Вывод первых 10 элементов массива для проверки
println("Первые 10 элементов массива squares_arr:")
println(squares_arr[1:10])
```

```
→ Создание массива squares_arr:
Первые 10 элементов массива squares_arr:
```



# Выполнение лабораторной работы

▶ # Задание 2 Условные операторы для проверки чётности

```
# Стандартный условный оператор if-else
println("Проверка чётности (if-else):")
for num in 1:10
    if num % 2 == 0
        println(num)
    else
        println("нечётное")
    end
end
```

↕ Проверка чётности (if-else):

```
нечётное
2
нечётное
4
нечётное
6
нечётное
8
нечётное
10
```

```
# Задание 2 Тернарный оператор
println("Проверка чётности (тернарный оператор):")
for num in 1:10
    result = num % 2 == 0 ? num : "нечётное"
    println(result)
end
```

↕ Проверка чётности (тернарный оператор):

```
нечётное
2
нечётное
4
нечётное
6
нечётное
8
нечётное
```

```
# Задание 3 Функция add_one

function add_one(x)
    return x + 1
end
```

⇒ add\_one (generic function with 1 method)

```
x = add_one(6)
x
```

⇒ 7

Рис. 8: Задание №3

```
# Задание 4
n = 4 # строки
m = 3 # столбцы

# Создаем линейный диапазон и преобразуем в матрицу
A1 = reshape(1:(n*m), n, m)
println("Исходная матрица A1:")
println(A1)

# Добавляем 1 к каждому элементу с помощью broadcast
A1_plus_one = A1 .+ 1
println("\nМатрица A1 + 1:")
println(A1_plus_one)
```

⇒ Исходная матрица A1:  
[1 5 9; 2 6 10; 3 7 11; 4 8 12]

Матрица A1 + 1:  
[2 6 10; 3 7 11; 4 8 12; 5 9 13]

Рис. 9: Задание №4

## Выполнение лабораторной работы

```
# Задание 5 Матрица A
A = [1 1 3; 5 2 6; -2 -1 -3]

println("Исходная матрица A:")
println(A)

# A^3
A_3 = A^3
println("\nA^3:")
println(A_3)

# Замена третьего столбца
A[:, 3] = A[:, 2] + A[:, 3]
println("\nМатрица A после замены третьего столбца:")
println(A)
```

↔ Исходная матрица A:  
[1 1 3; 5 2 6; -2 -1 -3]

A^3:  
[0 0 0; 0 0 0; 0 0 0]

Матрица A после замены третьего столбца:  
[1 1 4; 5 2 8; -2 -1 -4]

# Выполнение лабораторной работы

```
# Задание 6 Создаем матрицу B (15*3)
B = [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
      10 -10 10
      10 -10 10
      10 -10 10
      10 -10 10]

println("Матрица B (15*3):")
println(B)
println("Размер: ", size(B))

# Вычисляем C = B^T B
C = B' * B

println("\nМатрица C = B^T B (3*3):")
println(C)
```

Матрица B (15\*3):  
[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; 10 -10 10; 10 -10 10; 10 -10 10; 10 -10 10]  
Размер: (15, 3)

Матрица C = B^T B (3\*3):  
[1500 -1500 1500; -1500 1500 -1500; 1500 -1500 1500]

Рис. 11: Задание №6

# Выполнение лабораторной работы

```
# Задание 7

# Матрица Z1
Z1 = zeros(int, 6, 6)
for i in 1:5
    Z1[i, i+1] = 1
    Z1[i+1, i] = 1
end

# Матрица Z2
Z2 = zeros(int, 6, 6)
for i in 1:6
    for j in 1:6
        if abs(i - j) == 0 || abs(i - j) == 2
            Z2[i, j] = 1
        end
    end
end

# Матрица Z3
Z3 = [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]

# Матрица Z4
Z4 = [(i + j) % 2 == 0 ? 1 : 0 for i in 1:6, j in 1:6]

println("Z1:")
println(Z1)
println("\nZ2:")
println(Z2)
println("\nZ3:")
println(Z3)
println("\nZ4:")
println(Z4)
```

```
Z1:
[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]
```

```
Z2:
[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]
```

```
Z3:
```



# Выполнение лабораторной работы

```
# задание 8 Функция outer
function outer(x, y, operation)
    result = similar(x, length(x), length(y))
    for i in 1:length(x)
        for j in 1:length(y)
            result[i, j] = operation(x[i], y[j])
        end
    end
    return result
end
```

```
# Матрица A1
A1 = outer(0:4, 0:4, +)
```

```
# Матрица A2
A2 = outer(1:4, 1:5, ^)
```

```
# Матрица A3
A3 = outer(0:4, 0:4, (a,b) -> (a + b) % 5)
```

```
# Матрица A4
A4 = outer(0:9, 0:9, (a,b) -> (a + b) % 10)
```

```
# Матрица A5
A5 = outer(0:8, 0:8, (a,b) -> (a - b + 9) % 9)
```

```
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
```

# Выполнение лабораторной работы

```
# Матрица A1
A1 = outer(0:4, 0:4, +)

# Матрица A2
A2 = outer(1:4, 1:5, ^)

# Матрица A3
A3 = outer(0:4, 0:4, (a,b) -> (a + b) % 5)

# Матрица A4
A4 = outer(0:9, 0:9, (a,b) -> (a + b) % 10)

# Матрица A5
A5 = outer(0:8, 0:8, (a,b) -> (a - b + 9) % 9)
```

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

A5

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

# Выполнение лабораторной работы

```
# Задние 9 Решение системы линейных уравнений
```

```
# Матрица коэффициентов A
```

```
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  
]
```

```
# Вектор правой части
```

```
y = [7, -1, -3, 5, 17]
```

```
# Решение системы Ax = y
```

```
x = A \ y
```

```
println("Решение системы:")
```

```
for i in 1:length(x)
```

```
    println("x$i = ", round(x[i], digits=6))
```

```
end
```

```
# Проверка
```

```
println("\nПроверка A*x:")
```

```
println(round(A * x, digits=6))
```

```
println("Должно быть: $y")
```



```
Решение системы:
```

```
x1 = -2.0
```

```
x2 = 3.0
```

```
x3 = 5.0
```

```
x4 = 2.0
```

```
x5 = -4.0
```

```
Проверка A*x:
```

```
[7.0, -1.0, -3.0, 5.0, 17.0]
```

```
Должно быть: [7, -1, -3, 5, 17]
```

# Выполнение лабораторной работы

```
# Задание 10
# Создание матрицы M размерности 6x10 со случайными целыми числами от 1 до 10
M = rand(1:10, 6, 10)

println("Матрица M:")
println(M)

# Параметры
N = 4
K = 75
target_M = 7

# Число элементов в каждой строке, которые больше N
count_greater_than_M = [sum(row .> N) for row in eachrow(M)]
println("\n1. Число элементов > $N в каждой строке:")
for (i, count) in enumerate(count_greater_than_M)
    println("Строка $i: $count элементов")
end

# Строки, где число target_M встречается ровно 2 раза
rows_with_exactly_two_M = []
for (i, row) in enumerate(eachrow(M))
    if sum(row .== target_M) == 2
        push!(rows_with_exactly_two_M, i)
    end
end
println("\n2. Строки, где число $target_M встречается ровно 2 раза:")
println(rows_with_exactly_two_M)

# Пары столбцов, сумма элементов которых больше K
column_pairs = []
n_cols = size(M, 2)

for i in 1:n_cols
    for j in (i+1):n_cols
        if sum(M[:, i] + M[:, j]) > K
            push!(column_pairs, (i, j))
        end
    end
end

println("\n3. Пары столбцов с суммой элементов > $K:")
for pair in column_pairs
    col1_sum = sum(M[:, pair[1]])
    col2_sum = sum(M[:, pair[2]])
    total_sum = col1_sum + col2_sum
    println("Столбцы $pair[1] и $pair[2]: $col1_sum + $col2_sum = $total_sum")
end
```

Матрица M:  
[7 5 6 1 7 6 7 2 2 8; 7 4 10 7 6 5 10 5 8 3; 1 4 4 6 6 6 5 1 10 4; 7 4 7 9 3 1 1 9 6 3; 1 1 5 5 7 10 3 3 3 10; 10 6 9 9 10 9 2 3 6 8]

1. Число элементов > 4 в каждой строке:

Строка 1: 7 элементов

Строка 2: 8 элементов

Строка 3: 5 элементов

Строка 4: 5 элементов

Строка 5: 5 элементов

Строка 6: 8 элементов

2. Строки, где число 7 встречается ровно 2 раза:

Any[2, 4]

3. Пары столбцов с суммой элементов > 75:


Столбцы 3 и 4: 41 + 37 = 78

```
# Задание 11
# Первая сумма:
sum1 = 0.0
for i in 1:20
    for j in 1:5
        sum1 += i^4 / (3 + j)
    end
end

println("Результат: ", round(sum1, digits=6))

# Вторая сумма:
sum2 = 0.0
for i in 1:20
    for j in 1:5
        sum2 += i^4 / (3 + i * j)
    end
end

println("Результат: ", round(sum2, digits=6))
```

 Результат: 639215.283333  
Результат: 89912.021461

Напишите программный код или [сгенерируйте](#) его с помощью искусственного интеллекта.

В результате выполнения данной лабораторной работы я освоила применение циклов функций и сторонних для Julia пакетов для решения задач линейной алгебры и работы с матрицами.

научилась применять их и операции над ними для решения задач.