

# Лабораторная работа № 3

## Управляющие структуры

Оразгелдиев Язгелди

2025-10-11

# Содержание I

## Информация

# Информация

► Оразгелдиев Язгелди

# Докладчик

- ▶ Оразгелдиев Язгелди
- ▶ 1032225075@pfur.ru

# Докладчик

- ▶ Оразгелдиев Язгелди
- ▶ 1032225075@pfur.ru
- ▶ <https://github.com/YazgeldiOrazgeldiyev>

## Цель работы

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

## Задание

1. Используя Jupyter Lab, повторите примеры из раздела 3.2.



## Задание

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

# Содержание исследования

Циклы while и for

```
[1]: # пока n<10 прибавлять к n единицу и рассчитывать значение:
n = 0
while n < 10
    n += 1
    println(n)
end

1
2
3
4
5
6
7
8
9
10

[2]: 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!

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

Рисунок 1: Циклы while и for

# Содержание исследования

**Условные выражения**

```
# используем 'div' 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

# a ? b : c (эквивалент if (a); b; else; c)
x = 5
y = 10
(x > y) ? x : y

10
```

Рисунок 2: Условные выражения

# Содержание исследования

**Функции**

```
function sayHi(name)
    println("Hi $name, it's great to see you!")
end

# функция возведения в квадрат:
function f(x)
    x^2
end

f (generic function with 1 method)

# Вызов функции
sayHi("C-3PO")
f(12)

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

# создать массив v:
v = [3, 5, 2]
sort(v)
v

3-element Vector{Int64}:
 3
 5
 2

Функция sort(v) возвращает отсортированный массив, который содержит те же элементы, что и массив v, но исходный массив v остался без изменений. Если же исползвать sort!(v), то отсортировано будет содержимое исходного массива v.

sort!(v)
v
```

Рисунок 3: Функции

## Содержание исследования

[illegible]

### Рисунок 4: Colors

# Содержание исследования

## Задание для самостоятельной работы

### Самостоятельная работа

#### Задача 1

```
# Часть 1: Вывод целых чисел от 1 до 100 и их квадратов
for i in 1:100
    println("$i, $(i^2)")
end

# Часть 2: Создание словаря с квадратами чисел
squares = Dict{i => i^2 for i in 1:100}

# Часть 3: Создание массива с квадратами чисел
squares_arr = [i^2 for i in 1:100]
```

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

Рисунок 5: задание 1

# Содержание исследования

```
100-element Vector{Int64}:
```

```
1  
4  
9  
16  
25  
36  
49  
64  
81  
100  
121  
144  
169  
:  
7921  
8100  
8281  
8464  
8649  
8836  
9025  
9216  
9409  
9604  
9801  
10000
```

Рисунок 6: Задание 1

# Содержание исследования

```
Задача 2

# Функция для проверки четности числа и вывода результата
function check_even_odd(n)
    if n % 2 == 0
        println(n)
    else
        println("нечётное")
    end
end

# Пример использования функции
n % 2 == 0 ? println(n) : println("нечётное")

end

# Пример использования функции check_even_odd с числами 4 и 5
check_even_odd(4)
check_even_odd(5)

4
4
нечётное
нечётное

Задача 3

# Функция add_one
add_one = x -> x + 1

# Пример использования функции add_one с числом 5
add_one_result = add_one(5)

647
```

Рисунок 7: Задание 2 и 3



# Содержание исследования

## Задача 4

```
# Создаем матрицу A размером 5x5, где каждый элемент увеличивается на 1
A = reshape(1:25, 5, 5)

# Использование map или broadcast для увеличения каждого элемента на 1
A_map = map(x -> x + 1, A)
A_broadcast = A .+ 1
```

```
5x5 Matrix{Int64}:
 2  7 12 17 22
 3  8 13 18 23
 4  9 14 19 24
 5 10 15 20 25
 6 11 16 21 26
```

## Задача 5

```
# Инициализация матрицы A
A = [1 1 1; 1 2 6; -2 -1 -3]

# Вычисление A^3
A_cubed = A^3

# Замена второго столбца на сумму второго и третьего столбцов
A[:, 2] = A[:, 2] + A[:, 3]
```

```
3-element Vector{Int64}:
 4
 8
-4
```

Рисунок 8: Задание 4 и 5

# Содержание исследования

## Задача 6

```
: # Создание матрицы B размером 15x3
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]

# Вычисление матрицы C = B^T * B
C = transpose(B) * B

3x3 Matrix{Int64}:
 1500  -1500   1500
-1500   1500  -1500
 1500  -1500   1500
```

Рисунок 9: Задание 6

# Содержание исследования

**Задача 7**

```
# Создание матриц Z, E размером 6x6
Z = zeros(int, 6, 6)
E = ones(int, 6, 6)

# Создание матриц Z1, Z2, Z3, Z4
Z1, Z2, Z3, Z4 = copy(Z), copy(Z), copy(Z), copy(Z)

# Заполнение матриц Z1 и Z2
for i in 1:6
    for j in 1:6
        if i == j || i == 7 - j
            Z1[i, j] = 1 - (i + j) % 2
            Z2[i, j] = (i + j) % 2
        end
    end
end

# Заполнение матриц Z3 и Z4
for i in 1:6
    for j in 1:6
        if (i + j) % 2 == 1
            Z3[i, j] = 1
            Z4[7 - i, 7 - j] = 1
        end
    end
end

println("Z1 = ", Z1)
println("Z2 = ", Z2)
println("Z3 = ", Z3)
println("Z4 = ", Z4)

Z1 = [1 0 0 0 0 0; 0 1 0 0 0 0; 0 0 1 0 0 0; 0 0 0 1 0 0; 0 0 0 0 1 0; 0 0 0 0 0 1]
Z2 = [0 0 0 0 0 1; 0 0 0 0 1 0; 0 0 0 1 0 0; 0 0 1 0 0 0; 0 1 0 0 0 0; 1 0 0 0 0 0]
Z3 = [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; 1 0 1 0 1 0]
Z4 = [0 1 0 1 0 1; 1 0 1 0 1 0; 0 1 0 1 0 1; 1 0 1 0 1 0]
```

Рисунок 10: Задание 7

# Содержание исследования

## Задача 8

```
function custom_outer(x, y, operation)
    result = zeros(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

# Пример использования функции custom_outer
x = [1, 2, 3]
y = [4, 5, 6]
result = custom_outer(x, y, (a, b) -> a * b) # Пример с умножением
```

```
3x3 Matrix{Float64}:
 4.0  5.0  6.0
 8.0 10.0 12.0
12.0 15.0 18.0
```

## Задача 9

```
using LinearAlgebra

# Определение матрицы коэффициентов A и вектора свободных членов y
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]

# Решение системы линейных уравнений
x = A \ y
```

```
5-element Vector{Float64}:
-2.0000000000000036
 3.0000000000000058
 4.999999999999998
 1.9999999999999991
-3.9999999999999999
```

Рисунок 11: Задание 8 и 9

# Содержание исследования

## Задача 10

```
using Random

# Создаем матрицу M размером NxM со случайными элементами от 1 до 10
M = rand(1:10, 6, 10)

# N задано конст 4
N = 4
# Подсчитываем количество элементов в каждой строке, больше N
count_greater_than_N = [sum(row .> N) for row in eachrow(M)]

# M_value задано конст 7
M_value = 7
# Определяем строки, где M_value встречается ровно 2 раза
rows_with_M_value_twice = [sum(row .== M_value) == 2 for row in eachrow(M)]

# K задано конст 75
K = 75
# Находим пары столбцов, сумма элементов которых больше K
column_pairs_sum_greater_than_K = []
for i in 1:size(M, 2)
    for j in (i + 1):size(M, 2)
        if sum(M[:, i] + M[:, j]) > K
            push!(column_pairs_sum_greater_than_K, (i, j))
        end
    end
end

# Вывод результатов
println("Матрица M:\n", M)
println("Количество элементов в каждой строке больше N (" , N, "): ", count_greater_than_N)
println("Строки, где значение ", M_value, " встречается ровно 2 раза: ", rows_with_M_value_twice)
println("Пары столбцов, сумма элементов которых больше K (" , K, "): ", column_pairs_sum_greater_than_K)
```

Матрица M:

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

Количество элементов в каждой строке больше N (4): [9, 4, 6, 5, 7, 10]

Строки, где значение 7 встречается ровно 2 раза: [Bool[1, 0, 0, 1, 0, 1]]

Пары столбцов, сумма элементов которых больше K (75): Any{Tuple{Int64, Int64}}: (1, 4), (1, 5), (1, 9), (2, 4), (2, 5), (2, 6), (2, 7), (2, 9), (4, 5), (4, 9)]

Рисунок 12: Задание 10

# Содержание исследования

## Задача 11

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

639215.2833333334

```
# Вычисление второй суммы  
sum2 = sum([i^4 / (3 + i * j) for i in 1:20, j in 1:5])
```

89912.02146097137

Рисунок 13: Задание 11

# Результаты

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