

Разнообразие задач в Julia: от циклов до использования библиотек

Презентация к 3 лабораторной работе

**Группа НКНбд-01-20
Мухамедияр Адиль**

Циклы

Задание(повторяем примеры из раздела)

Циклы while и for

[6]: # 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
```

[7]: 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!
```

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

```
1
3
5
7
9
```

Изучение

Циклы 2.0.

```
[9]: 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!
```

```
[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
```

```
[14]: # 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
```

```
[14]: 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}:
```

Условные выражения

```
[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
```

Условные выражения

```
[18]: # 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
```

```
[20]: # a ? b : c(означает if (a); b; else: c)
      x = 5
      y = 10
      (x > y) ? x : y
```

```
[20]: 10
```

▼ Функции

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

```
[21]: f (generic function with 1 method)
```

Функции

```
[21]: f (generic function with 1 method)
```

```
[22]: # Вызов функции  
sayhi("C-3PO")  
f(42)
```

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

```
[22]: 1764
```

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

```
[24]: 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` остаётся без изменений. Если же использовать `sort!(v)`, то отсортировано будет содержимое исходного массива `v`.

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

```
[25]: 3-element Vector{Int64}:  
 2  
 3  
 5
```

```
[26]: map(f, [1, 2, 3])
```

```
[26]: 3-element Vector{Int64}:  
 1  
 4  
 9
```

```
[27]: x -> x^3  
map(x -> x^3, [1, 2, 3])
```

```
[27]: 3-element Vector{Int64}:  
 1  
 8  
 27
```

Изучение

5
25.11.23

Функции 2.0.

```
[28]: f(x) = x^2  
broadcast(f, [1, 2, 3])  
# Задаём матрицу A:  
A = [i + 3*j for j in 0:2, i in 1:3]
```

```
[28]: 3x3 Matrix{Int64}:  
 1  2  3  
 4  5  6  
 7  8  9
```

```
[29]: # Вызываем функцию f возведения в квадрат  
f(A)
```

```
[29]: 3x3 Matrix{Int64}:  
 30  36  42  
 66  81  96  
102 126 150
```

```
[30]: B = f.(A)
```

```
[30]: 3x3 Matrix{Int64}:  
 1  4  9  
16 25 36  
49 64 81
```

Точечный синтаксис для broadcast() позволяет записать относительно сложные составные поэлементные выражения в форме, близкой к математической записи.

```
[31]: A .+ 2 .* f.(A) ./ A  
broadcast(x -> x + 2 .* f(x) / x, A)
```

```
[31]: 3x3 Matrix{Float64}:  
 3.0  6.0  9.0  
12.0 15.0 18.0  
21.0 24.0 27.0
```

▼ Сторонние библиотеки (пакеты) в Julia

```
[33]: import Pkg  
Pkg.add("Example")
```

```
Resolving package versions...  
No Changes to `C:\Users\adiks\.julia\environments\v1.9\Project.toml`  
No Changes to `C:\Users\adiks\.julia\environments\v1.9\Manifest.toml`
```

Библиотеки

```
[33]: import Pkg
      Pkg.add("Example")

      Resolving package versions...
      No Changes to 'C:\Users\adiks\.julia\environments\v1.9\Project.toml'
      No Changes to 'C:\Users\adiks\.julia\environments\v1.9\Manifest.toml'

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

      Resolving package versions...
      Installed ColorTypes ----- v0.11.4
      Installed FixedPointNumbers - v0.8.4
      Installed Reexport ----- v1.2.2
      Installed Colors ----- v0.12.10
      Updating 'C:\Users\adiks\.julia\environments\v1.9\Project.toml'
      [Sae59095] + Colors v0.12.10
      Updating 'C:\Users\adiks\.julia\environments\v1.9\Manifest.toml'
      [3da002f7] + ColorTypes v0.11.4
      [Sae59095] + Colors v0.12.10
      [53c48c17] + FixedPointNumbers v0.8.4
      [189a3867] + Reexport v1.2.2
      [37e2e46d] + LinearAlgebra
      [2f01184e] + SparseArrays
      [10745b16] + Statistics v1.9.0
      [e66e0078] + CompilerSupportLibraries_jll v1.0.5+0
      [4536629a] + OpenBLAS_jll v0.3.21+4
      [bea87d4a] + SuiteSparse_jll v5.10.1+6
      [8e850b90] + libblastrampoline_jll v5.8.0+0
      Precompiling project...
      ✓ Reexport
      ✓ CompilerSupportLibraries_jll
      ✓ FixedPointNumbers
      ✓ ColorTypes
      ✓ Colors
      5 dependencies successfully precompiled in 16 seconds. 21 already precompiled.

[35]: palette = distinguishable_colors(100)

[35]: 

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

[41]: 
```

Изучение

7
25.11.23

Задача №1

Самостоятельная работа

Задача 1

[42]:

```
# 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
20, 400
21, 441
22, 484
23, 529
24, 576
25, 625
26, 676
27, 729
28, 784
29, 841
30, 900
31, 961
32, 1024
33, 1089
34, 1156
35, 1225
36, 1296
37, 1369
38, 1444
```

Самостоятельная
работа

8
25.11.23

Задачи №2-3

Задача 2

```
[45]: # Функция для проверки четности числа и вывода результата
function check_even_odd(n)
    if n % 2 == 0
        println(n)
    else
        println("нечётное")
    end
    # Тернарный оператор
    n % 2 == 0 ? println(n) : println("нечётное")
end

# Пример использования функции check_even_odd с числом 4 и 5
check_even_odd(4)
check_even_odd(5)
```

```
4
4
нечётное
нечётное
```

Задача 3

```
[48]: # Функция add_one
add_one = x -> x + 1

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

```
[48]: 6
```

Самостоятельная
работа

9
25.11.23

Задачи №4-6

Задача 4

```
[49]: # Создание матрицы 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

[49]: 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

```
[50]: # Инициализация матрицы A
A = [1 1 3; 5 2 6; -2 -1 -3]

# Вычисление A^3
A_cubed = A^3

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

[50]: 3-element Vector{Int64}:
 4
 8
 -4
```

Задача 6

```
[52]: # 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

[52]: 3x3 Matrix{Int64}:
1500 -1500 1500
-1500 1500 -1500
1500 -1500 1500
```

Самостоятельная
работа

10
25.11.23

Задача №7

Задача 7

```
[56]: # 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 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 1 0 0 0; 0 0 1 0 0 0; 0 1 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]
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; 0 1 0 1 0 1]
```

Самостоятельная
работа

11
25.11.23

Задачи №8-9

Задача 8

```
[63]: 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
```

```
[63]: 3x3 Matrix{Float64}:
      4.0  5.0  6.0
      8.0 10.0 12.0
      12.0 15.0 18.0
```

Задача 9

```
[64]: 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
```

```
[64]: 5-element Vector{Float64}:
 -2.0000000000000036
  3.0000000000000058
  4.999999999999998
  1.9999999999999991
 -3.9999999999999999
```

Самостоятельная
работа

12
25.11.23

Задача №10

Задача 10

[73]:

```
using Random

# Создание матрицы M размером 6x10 со случайными элементами от 1 до 10
M = rand(1:10, 6, 10)

# N задано как 4
N = 4

# Подсчет количества элементов в каждой строке, больших N
count_greater_than_N = [sum(row .> N) for row in eachrow(M)]

# M_value задано как 7
M_value = 7

# Определение строк, где M_value встречается ровно 2 раза
rows_with_M_value_twice = [sum(row .== M_value) == 2 for row in eachrow(M)]

# K задано как 75
K = 75

# Определение пар 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:

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

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

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

Пары столбцов, сумма элементов которых больше K (75): Any[(1, 4), (1, 6), (1, 8), (1, 10), (4, 6), (4, 8), (4, 10), (6, 8), (6, 10), (8, 10)]

Самостоятельная
работа

13
25.11.23

Задача №11

Задача 11

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

```
[77]: 639215.2833333334
```

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

```
[78]: 89912.02146097137
```

Заключение

“

Спасибо за внимание!

”