

Отчет по лабораторной работе №3

Управляющие структуры

Легиньких Галина Андреевна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Вывод	22

Список иллюстраций

3.1	while	7
3.2	for вналогично while	8
3.3	for массивы	8
3.4	Условные выражения	9
3.5	Функции 1-ый способ	9
3.6	Функции 2-ой способ	9
3.7	Функции 3-ий способ	10
3.8	sort as !sort	10
3.9	map	10
3.10	broadcast	11
3.11	Загрузка пакетов	12
3.12	Использование пакетов	12
3.13	Задание 1.1 while	13
3.14	Задание 1.1 for	13
3.15	Задание 1.2	14
3.16	Задание 1.3	14
3.17	Задание 2	15
3.18	Задание 3	15
3.19	Задание 4	15
3.20	Задание 5	16
3.21	Задание 6	16
3.22	Задание 7.0	16
3.23	Задание 7.1	17
3.24	Задание 7.2	17
3.25	Задание 7.3	18
3.26	Задание 7.4	18
3.27	Задание 8.1 - 8.2	19
3.28	Задание 8.3 - 8.4	19
3.29	Задание 8.5	20
3.30	Задание 9	20
3.31	Задание 10.1 - 10.2	20
3.32	Задание 10.3	21
3.33	Задание 11	21

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

1. Используя Jupyter Lab, повторите примеры из раздела 2.2.
2. Выполните задания для самостоятельной работы (раздел 2.4).

3 Выполнение лабораторной работы

1. Для начала я изучила видеоматериал к лекции и повторила примеры из раздела 3.2. Сначала это были циклы `while` и `for`. Для различных операций, связанных с перебором индексируемых элементов структур данных, традиционно используются циклы `while` и `for`. (рис. 3.1) (рис. 3.2) (рис. 3.3)

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

[4]: 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. 3.1: `while`

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

[10]: 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!

[12]: # инициализация массива m x n из нулей:
      m, n = 5, 5
      A = fill{0, (m, n)}
```

Рис. 3.2: for вналогично while

```
[12]: # инициализация массива m x n из нулей:
      m, n = 5, 5
      A = fill{0, (m, n)}

[12]: 5x5 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

[14]: # 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

[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

[ ]: # 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

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

Рис. 3.3: for массивы

2. Условные выражения. Довольно часто при решении задач требуется проверить выполнение тех или иных условий. Для этого используют условные выра-

жения. Повторила синтаксис условных выражений с тернарными операторами. (рис. 3.4)

```
[24]: # используем `&&` для реализации операции "AND"
      # операция % вычисляет остаток от деления
      N = 20
      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

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

[26]: 10
```

Рис. 3.4: Условные выражения

3. Далее перешла к функциям. Julia дает нам несколько разных способов написать функцию. Первый требует ключевых слов `function` и `end`. (рис. 3.5)

```
[28]: function sayhi(name)
      println("Hi $name, it's great to see you!")
      end
      sayhi("C-3P0")

      Hi C-3P0, it's great to see you!

[30]: # функция возведения в квадрат:
      function f(x)
          x^2
      end
      f(42)

[30]: 1764
```

Рис. 3.5: Функции 1-ый способ

В качестве альтернативы, можно объявить любую из выше определённых функций в одной строке. (рис. 3.6)

```
[44]: sayhi2(name) = println("Hi $name, it's great to see you!")
      sayhi2("C-3P0")

      Hi C-3P0, it's great to see you!

[46]: f2(x) = x^2
      f2(42)

[46]: 1764
```

Рис. 3.6: Функции 2-ой способ

Наконец, можно объявить выше определённые функции как “анонимные”. (рис. 3.7)

```
[48]: sayhi3 = name -> println("Hi $name, it's great to see you!")
      sayhi3("C-3P0")
      Hi C-3P0, it's great to see you!

[50]: f3 = x -> x^2
      f3(42)

[50]: 1764
```

Рис. 3.7: Функции 3-ий способ

По соглашению в Julia функции, сопровождаемые восклицательным знаком, изменяют свое содержимое, а функции без восклицательного знака не делают этого. (рис. 3.8)

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

[60]: 3-element Vector{Int64}:
      3
      5
      2

[56]: sort!(v)
      v

[56]: 3-element Vector{Int64}:
      2
      3
      5
```

Рис. 3.8: sort as !sort

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. (рис. 3.9)

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

[62]: 3-element Vector{Int64}:
      1
      4
      9

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

[64]: 3-element Vector{Int64}:
      1
      4
      9
```

Рис. 3.9: map

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. (рис. 3.10)

```
[66]: broadcast(f, [1, 2, 3])

[66]: 3-element Vector{Int64}:
      1
      4
      9

[68]: # Задаём матрицу A:
      A = [i + 3*j for j in 0:2, i in 1:3]

[68]: 3x3 Matrix{Int64}:
      1  2  3
      4  5  6
      7  8  9

[70]: f(A)

[70]: 3x3 Matrix{Int64}:
      30  36  42
      66  81  96
     102 126 150

[72]: B = f.(A)

[72]: 3x3 Matrix{Int64}:
      1  4  9
     16 25 36
     49 64 81

[74]: A .+ 2 .* f.(A) ./ A

[74]: 3x3 Matrix{Float64}:
      3.0  6.0  9.0
     12.0 15.0 18.0
     21.0 24.0 27.0

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

[76]: 3x3 Matrix{Float64}:
      3.0  6.0  9.0
     12.0 15.0 18.0
     21.0 24.0 27.0
```

Рис. 3.10: `broadcast`

4. Julia имеет более 2000 зарегистрированных пакетов, что делает их огромной частью экосистемы Julia. Научилась загружать пакеты. (рис. 3.11)

```
[79]: import Pkg

[83]: Pkg.add("Example")

Resolving package versions...
No Changes to `C:\Users\galin\.julia\environments\v1.10\Project.toml`
No Changes to `C:\Users\galin\.julia\environments\v1.10\Manifest.toml`

[85]: Pkg.add("Colors")


Resolving package versions...
Installed FixedPointNumbers v0.8.5
Installed Reexport v1.2.2
Installed Colors v0.13.0
Installed ColorTypes v0.12.0
Updating `C:\Users\galin\.julia\environments\v1.10\Project.toml`
[5ae59095] + Colors v0.13.0
Updating `C:\Users\galin\.julia\environments\v1.10\Manifest.toml`
[3da002f7] + ColorTypes v0.12.0
[5ae59095] + Colors v0.13.0
[53c48c17] + FixedPointNumbers v0.8.5
[189a3867] + Reexport v1.2.2
[37e2e46d] + LinearAlgebra
[2f01184e] + SparseArrays v1.10.0
[10745b16] + Statistics v1.10.0
[e66e0078] + CompilerSupportLibraries_jll v1.1.1+0
[4536629a] + OpenBLAS_jll v0.3.23+4
[bea87d4a] + SuiteSparse_jll v7.2.1+1
```

Рис. 3.11: Загрузка пакетов

И использовать их. (рис. 3.12)

```
[87]: using Colors

[89]: palette = distinguishable_colors(100)

[89]: 

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

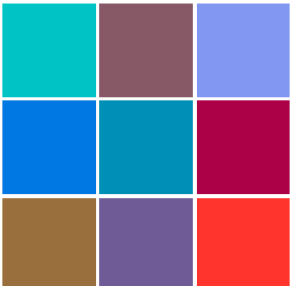
[91]: 
```

Рис. 3.12: Использование пакетов

5. Перешла к заданиям для самостоятельной работы. Дублировать задания не буду, они слишком большие. Нумерация соответствует нумерации в файле задания.

- Задание 1.1 с помощью while (рис. 3.13)

```
[97]: # Задание 1.1
println("Числа и их квадраты:")
i = 1
while i <= 100
    println("Число: $i, Квадрат: $(i^2)")
    i += 1
end
```

Числа и их квадраты:
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

Рис. 3.13: Задание 1.1 while

- Задание 1.1 с помощью for (рис. 3.14)

```
[99]: for i in 1:100
    println("Число: $i, Квадрат: $(i^2)")
end
```

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

Рис. 3.14: Задание 1.1 for

- Задание 1.2 (рис. 3.15)

```
[114]: # Задание 1.2
squares = Dict()
for i in 1:100
    squares[i] = i^2
end
squares
```

```
[114]: Dict{Any, Any} with 100 entries:
 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
```

Рис. 3.15: Задание 1.2

- Задание 1.3 (рис. 3.16)

```
[116]: # Задание 1.3
squares_arr = [i^2 for i in 1:100]
```

```
[116]: 100-element Vector{Int64}:
 1
 4
 9
16
25
36
49
64
81
100
121
144
169
 ⋮
7921
8100
8281
```

Рис. 3.16: Задание 1.3

- Задание 2 (рис. 3.17)

```
[118]: # Задание 2.1
function check_number(num)
    if num%2==0
        println(num)
    else
        println("нечётное")
    end
end
check_number(2)

2

[120]: check_number(5)

нечётное

[122]: # Задание 2.2
function check_number_ternary(num)
    println(num%2==0 ? num : "нечётное")
end
check_number_ternary(2)

2

[124]: check_number_ternary(5)

нечётное
```

Рис. 3.17: Задание 2

- Задание 3 (рис. 3.18)

```
[128]: # Задание 3
add_one(x) = x + 1
add_one(2)

[128]: 3

[130]: add_one(3)

[130]: 4

[136]: add_one_2 = x -> x + 1
add_one_2(2)

[136]: 3
```

Рис. 3.18: Задание 3

- Задание 4 (рис. 3.19)

```
[150]: # Задание 4
m, n = 3, 3
A = reshape(1:9, m, n)
A = map(add_one, A)

[150]: 3x3 Matrix{Int64}:
 2  5  8
 3  6  9
 4  7 10

[156]: # broadcast()
A .+= 1

[156]: 3x3 Matrix{Int64}:
 5  8 11
 6  9 12
 7 10 13
```

Рис. 3.19: Задание 4

- Задание 5 (рис. 3.20)

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

[164]: 3x3 Matrix{Int64}:
 1  1  3
 5  2  6
-2 -1 -3

[170]: A3 = map(x -> x^3, A)

[170]: 3x3 Matrix{Int64}:
 1  1  27
125  8  216
-8 -1 -27

[172]: A[:,3] .+= A[:,2] .+ A[:,3]

[172]: 3-element view(::Matrix{Int64}, :, 3) with eltype Int64:
 4
 8
-4
```

Рис. 3.20: Задание 5

- Задание 6 (рис. 3.21)

```
[178]: # Задание 6
B = repeat([10 -10 10], 15, 1)
C = B'*B

[178]: 3x3 Matrix{Int64}:
1500 -1500 1500
-1500 1500 -1500
1500 -1500 1500
```

Рис. 3.21: Задание 6

- Задание 7.0 (рис. 3.22)

```
[186]: # Задание 7
Z = zeros(6,6)

[186]: 6x6 Matrix{Float64}:
 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  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
 0.0  0.0  0.0  0.0  0.0  0.0

[188]: E = ones(6,6)

[188]: 6x6 Matrix{Float64}:
 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  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
 1.0  1.0  1.0  1.0  1.0  1.0
```

Рис. 3.22: Задание 7.0

- Задание 7.1 (рис. 3.23)

```
[192]: Z1 = Z
      for i in 1:6
        if i > 1
          Z1[i,i-1] = 1
        end
        if i < 6
          Z1[i,i+1] = 1
        end
      end
      Z1

[192]: 6x6 Matrix{Float64}:
 0.0  1.0  0.0  0.0  0.0  0.0
 1.0  0.0  1.0  0.0  0.0  0.0
 0.0  1.0  0.0  1.0  0.0  0.0
 0.0  0.0  1.0  0.0  1.0  0.0
 0.0  0.0  0.0  1.0  0.0  1.0
 0.0  0.0  0.0  0.0  1.0  0.0
```

Рис. 3.23: Задание 7.1

- Задание 7.2 (рис. 3.24)

```
[212]: Z2 = Z
      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
      Z2

[212]: 6x6 Matrix{Float64}:
 1.0  0.0  1.0  0.0  0.0  0.0
 0.0  1.0  0.0  1.0  0.0  0.0
 1.0  0.0  1.0  0.0  1.0  0.0
 0.0  1.0  0.0  1.0  0.0  1.0
 0.0  0.0  1.0  0.0  1.0  0.0
 0.0  0.0  0.0  1.0  0.0  1.0
```

Рис. 3.24: Задание 7.2

- Задание 7.3 (рис. 3.25)

```
[292]: Z3 = Z = zeros(6,6)
for i in 1:6
    for j in 1:6
        # Условие для постановки единиц на основе расположения в матрице
        if (i == 1 && (j == 4 || j == 6)) ||
            (i == 2 && (j == 3 || j == 5)) ||
            (i == 3 && (j == 2 || j == 4 || j == 6)) ||
            (i == 4 && (j == 1 || j == 3 || j == 5)) ||
            (i == 5 && (j == 2 || j == 4)) ||
            (i == 6 && (j == 1 || j == 3))
            Z3[i, j] = 1
        end
    end
end
Z3
```

```
[292]: 6x6 Matrix{Float64}:
0.0  0.0  0.0  1.0  0.0  1.0
0.0  0.0  1.0  0.0  1.0  0.0
0.0  1.0  0.0  1.0  0.0  1.0
1.0  0.0  1.0  0.0  1.0  0.0
0.0  1.0  0.0  1.0  0.0  0.0
1.0  0.0  1.0  0.0  0.0  0.0
```

Рис. 3.25: Задание 7.3

- Задание 7.4 (рис. 3.26)

```
[298]: Z4 = zeros(6,6)
for i in 1:6
    for j in 1:6
        if mod(i+j,2)==0
            Z4[i,j]=1
        end
    end
end
Z4
```

```
[298]: 6x6 Matrix{Float64}:
1.0  0.0  1.0  0.0  1.0  0.0
0.0  1.0  0.0  1.0  0.0  1.0
1.0  0.0  1.0  0.0  1.0  0.0
0.0  1.0  0.0  1.0  0.0  1.0
1.0  0.0  1.0  0.0  1.0  0.0
0.0  1.0  0.0  1.0  0.0  1.0
```

Рис. 3.26: Задание 7.4

- Задание 8.1 - 8.2 (рис. 3.27)

```
[300]: # Задание 8
function outer(x,y,operation)
    m, n = length(x), length(y)
    result = zeros(m,n)
    for i in 1:m
        for j in 1:n
            result[i,j] = operation(x[i],y[j])
        end
    end
end
return result
end

[300]: outer (generic function with 1 method)

[308]: A1 = outer(0:4, 0:4, +)
A1
5x5 Matrix{Float64}:
 0.0  1.0  2.0  3.0  4.0
 1.0  2.0  3.0  4.0  5.0
 2.0  3.0  4.0  5.0  6.0
 3.0  4.0  5.0  6.0  7.0
 4.0  5.0  6.0  7.0  8.0

[318]: A2 = outer(0:4, 1:5, ^)
A2
5x5 Matrix{Float64}:
 0.0  0.0  0.0  0.0  0.0
 1.0  1.0  1.0  1.0  1.0
 2.0  4.0  8.0  16.0  32.0
 3.0  9.0  27.0  81.0  243.0
 4.0  16.0  64.0  256.0  1024.0
```

Рис. 3.27: Задание 8.1 - 8.2

- Задание 8.3 - 8.4 (рис. 3.28)

```
[322]: A3 = outer(0:4, 0:4, (a,b)->mod(a+b,5))
A3
5x5 Matrix{Float64}:
 0.0  1.0  2.0  3.0  4.0
 1.0  2.0  3.0  4.0  0.0
 2.0  3.0  4.0  0.0  1.0
 3.0  4.0  0.0  1.0  2.0
 4.0  0.0  1.0  2.0  3.0

[324]: A4 = outer(0:9, 0:9, (a,b)->mod(a+b,10))
A4
10x10 Matrix{Float64}:
 0.0  1.0  2.0  3.0  4.0  5.0  6.0  7.0  8.0  9.0
 1.0  2.0  3.0  4.0  5.0  6.0  7.0  8.0  9.0  0.0
 2.0  3.0  4.0  5.0  6.0  7.0  8.0  9.0  0.0  1.0
 3.0  4.0  5.0  6.0  7.0  8.0  9.0  0.0  1.0  2.0
 4.0  5.0  6.0  7.0  8.0  9.0  0.0  1.0  2.0  3.0
 5.0  6.0  7.0  8.0  9.0  0.0  1.0  2.0  3.0  4.0
 6.0  7.0  8.0  9.0  0.0  1.0  2.0  3.0  4.0  5.0
 7.0  8.0  9.0  0.0  1.0  2.0  3.0  4.0  5.0  6.0
 8.0  9.0  0.0  1.0  2.0  3.0  4.0  5.0  6.0  7.0
 9.0  0.0  1.0  2.0  3.0  4.0  5.0  6.0  7.0  8.0
```

Рис. 3.28: Задание 8.3 - 8.4

- Задание 8.5 (рис. 3.29)

```
[348]: A5 = outer(0:8, 0:8, (a,b)->mod(-abs(a-b),9))
A5

[348]: 9x9 Matrix{Float64}:
 0.0  8.0  7.0  6.0  5.0  4.0  3.0  2.0  1.0
 8.0  0.0  8.0  7.0  6.0  5.0  4.0  3.0  2.0
 7.0  8.0  0.0  8.0  7.0  6.0  5.0  4.0  3.0
 6.0  7.0  8.0  0.0  8.0  7.0  6.0  5.0  4.0
 5.0  6.0  7.0  8.0  0.0  8.0  7.0  6.0  5.0
 4.0  5.0  6.0  7.0  8.0  0.0  8.0  7.0  6.0
 3.0  4.0  5.0  6.0  7.0  8.0  0.0  8.0  7.0
 2.0  3.0  4.0  5.0  6.0  7.0  8.0  0.0  8.0
 1.0  2.0  3.0  4.0  5.0  6.0  7.0  8.0  0.0
```

Рис. 3.29: Задание 8.5

- Задание 9 (рис. 3.30)

```
[356]: # Задание 9
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]
x = A \ y
println("Решение системы уравнений: ", x)

Решение системы уравнений: [-1.9999999999999987, 2.9999999999999996, 4.999999999999998, 2.0000000000000001, -4.0]
```

Рис. 3.30: Задание 9

- Задание 10.1 - 10.2 (рис. 3.31)

```
[393]: # Задание 10
M = rand(1:10, 6, 10)

[393]: 6x10 Matrix{Int64}:
 3  5  4  10  7  2  6  7  9  4
 2 10 10  9  5  6  2  5  9  5
 3  2  2  4  6  8  1  5 10  9
 1  5 10 10  9  3  4  8  6  6
 1  2  7  1  3  1  2  2  5  4
 4  1  5  5  4 10  5  9  3  6

[395]: N = 4
count_N = 0
for i in 1:6
    for j in 1:10
        if M[i,j]>N
            count_N +=1
        end
    end
end
count_N

[395]: 34

[397]: B = 10
for i in 1:6
    count_B = 0
    for j in 1:10
        if M[i,j]==B
            count_B +=1
        end
    end
    if count_B == 2
        println("Строка с числом B, которое встречается ровно 2 раза: ", i)
    end
end

Строка с числом B, которое встречается ровно 2 раза: 2
Строка с числом B, которое встречается ровно 2 раза: 4
```

Рис. 3.31: Задание 10.1 - 10.2

- Задание 10.3 (рис. 3.32)

```
[405]: K = 75
for i in 1:9
    for j in 2:10
        if sum(M[:, i]) + sum(M[:, j]) > K
            println("Пара столбцов матрицы M, сумма элементов которой больше K: (", i, ", ", j, ")")
        end
    end
end
```

Пара столбцов матрицы M, сумма элементов которой больше K: (3,3)
Пара столбцов матрицы M, сумма элементов которой больше K: (3,4)
Пара столбцов матрицы M, сумма элементов которой больше K: (3,9)
Пара столбцов матрицы M, сумма элементов которой больше K: (4,3)
Пара столбцов матрицы M, сумма элементов которой больше K: (4,4)
Пара столбцов матрицы M, сумма элементов которой больше K: (4,9)
Пара столбцов матрицы M, сумма элементов которой больше K: (5,9)
Пара столбцов матрицы M, сумма элементов которой больше K: (8,9)
Пара столбцов матрицы M, сумма элементов которой больше K: (9,3)
Пара столбцов матрицы M, сумма элементов которой больше K: (9,4)
Пара столбцов матрицы M, сумма элементов которой больше K: (9,5)
Пара столбцов матрицы M, сумма элементов которой больше K: (9,8)
Пара столбцов матрицы M, сумма элементов которой больше K: (9,9)
Пара столбцов матрицы M, сумма элементов которой больше K: (9,10)

Рис. 3.32: Задание 10.3

- Задание 11 (рис. 3.33)

```
[409]: # Задание 11
sum1 = sum(i^4 / (3 + j) for i in 1:20 for j in 1:5)
sum1
```

[409]: 639215.2833333334

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

[411]: 89912.02146097136

[]:

Рис. 3.33: Задание 11

4 Вывод

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