

Лабораторная работа 3

Управляющие структуры

Ланцова Яна Игоревна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
3.1	Задание 1	9
3.2	Задание 2	10
3.3	Задание 3 и 4	10
3.4	Задание 5	11
3.5	Задание 6 и 7	11
3.6	Задание 8	13
3.7	Задание 9 и 10	14
3.8	Задание 11	15
4	Выводы	16

Список иллюстраций

3.1	Выполнение примеров с циклами	7
3.2	Выполнение примеров с циклами	7
3.3	Выполнение примеров с условными выражениями	8
3.4	Импорт сторонних библиотек	8
3.5	Выполнение примеров со сторонними библиотеками	8
3.6	Задание №1	9
3.7	Задание №1	9
3.8	Задание №1	10
3.9	Задание №2	10
3.10	Задание №3 и 4	11
3.11	Задание №5	11
3.12	Задание №6 и 7	12
3.13	Задание №7	12
3.14	Задание №7	12
3.15	Задание №7	13
3.16	Задание №8	13
3.17	Задание №8	14
3.18	Задание №8	14
3.19	Задание №9 и 10	15
3.20	Задание №11	15

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 3.2.
2. Выполните задания для самостоятельной работы.

3 Выполнение лабораторной работы

Для начала выполним примеры из лабораторной работы, чтобы познакомиться с циклами, условными операторами, функциями и работой со сторонними библиотеками (рис. 3.1-3.5).

```
[3]: # цикл
n = 3
while n < 10
    n += 1
    println(n)
end

4
5
6
7
8
9
10

[4]: myfriends = ["Ted", "Barney", "Collins", "Lily", "Marshall", "Lesya"];
i = 1
while i <= length(myfriends)
    friend = myfriends[i]
    println("Hi $friend, it's nice to see you!")
    i += 1
end

Hi Ted, it's nice to see you!
Hi Barney, it's nice to see you!
Hi Collins, it's nice to see you!
Hi Lily, it's nice to see you!
Hi Marshall, it's nice to see you!
Hi Lesya, it's nice to see you!

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

1
4
7
10
```

Рис. 3.1: Выполнение примеров с циклами

```
[9]: m, n = 3, 3
A = fill{0, (m, n)}

[9]: 3x3 Matrix{Int64}:
 0 0 0
 0 0 0
 0 0 0

[11]: for i in 1:m
    for j in 1:n
        A[i, j] = i + j
    end
end
println(A)

[2 3 4; 3 4 5; 4 5 6]

[12]: B = fill{0, (m, n)}
for i in 1:m, j in 1:n
    B[i, j] = i + j
end
println(B)

[2 3 4; 3 4 5; 4 5 6]

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

[13]: 3x3 Matrix{Int64}:
 2 3 4
 3 4 5
 4 5 6
```

Рис. 3.2: Выполнение примеров с циклами

```

[14]: N = 52
      if (N % 2 == 0) && (N % 9 == 0)
          println("Cool number!")
      elseif N % 2 == 0
          println("Oh thats a nice num!")
      elseif N % 9 == 0
          println("Mehhhh")
      else
          println(N)
      end
      Oh thats a nice num!

[15]: x = 5
      y = 2
      (x > y) ? x : y

[15]: 5

[16]: function sayhi(name)
      println("Hi $name, it's great to see you!")
      end
      sayhi("Lesya")
      Hi Lesya, it's great to see you!

[17]: f2(x) = x^2
      f2(9)

[17]: 81

[18]: f_anonymus = x -> x^2
      f_anonymus(9)

[18]: 81

```

Рис. 3.3: Выполнение примеров с условными выражениями

```

[19]: 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'
      [7876a6f0] + Example v0.5.5
      Updating '~/.julia/environments/v1.11/Manifest.toml'
      [7876a6f0] + Example v0.5.5
      Precompiling project...
      930.2 ms ./ Example
      1 dependency successfully precompiled in 2 seconds. 41 already precompiled.

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


      Resolving package versions...
      Installed Reexport - v1.2.2
      Installed Statistics - v1.11.1
      Installed ColorTypes - v0.12.1
      Installed FixedPointNumbers - v0.8.5
      Installed Colors - v0.13.1
      Updating '~/.julia/environments/v1.11/Project.toml'
      [5ae59095] + Colors v0.13.1
      Updating '~/.julia/environments/v1.11/Manifest.toml'
      [3da08217] + ColorTypes v0.12.1
      [5ae59095] + Colors v0.13.1
      [53c48c17] + FixedPointNumbers v0.8.5
      [189a3867] + Reexport v1.2.2
      [10745016] + Statistics v1.11.1
      [37e2e46d] + LinearAlgebra v1.11.0

```


Рис. 3.4: Импорт сторонних библиотек

```

[21]: palette = distinguishable_colors(100)

[21]: 

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

[22]: 

```

Рис. 3.5: Выполнение примеров со сторонними библиотеками

Теперь перейдем к выполнению заданий для самостоятельной работы.

3.1 Задание 1

Используя циклы while и for.

- выведем на экран целые числа от 1 до 100 и напечатаем их квадраты (рис. 3.6)

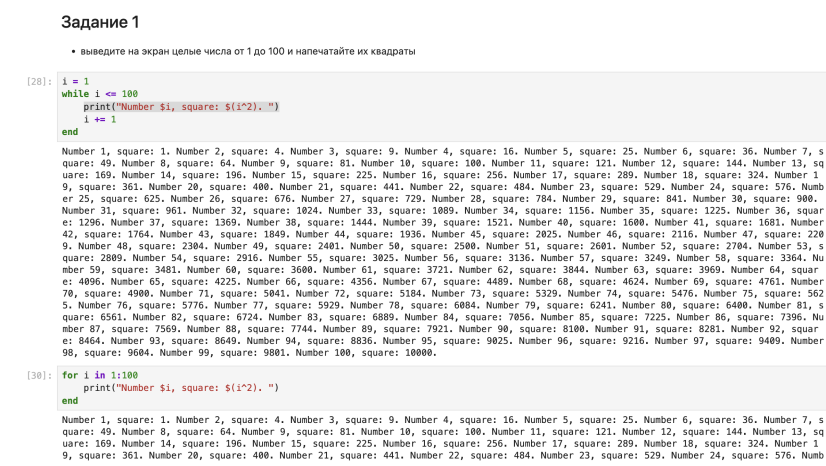


Рис. 3.6: Задание №1

- создадим словарь squares, который будет содержать целые числа в качестве ключей и квадраты в качестве их пар-значений (рис. 3.7)

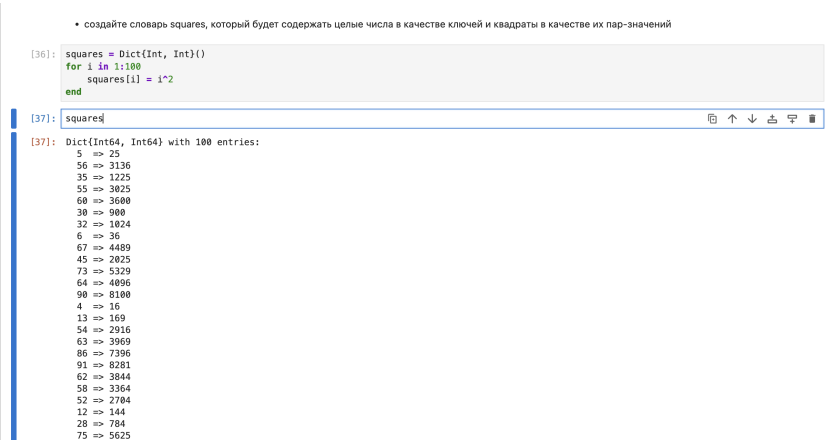


Рис. 3.7: Задание №1

- создадим массив squares_arr, содержащий квадраты всех чисел от 1 до 100 (рис. 3.8)

```
• создайте массив squares_arr, содержащий квадраты всех чисел от 1 до 100

[38]: squares_arr = [i**2 for i in 1:100]

[39]: 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
```

Рис. 3.8: Задание №1

3.2 Задание 2

Напишем условный оператор, который печатает число, если число чётное, и строку «нечётное», если число нечётное. Перепишем код, используя тернарный оператор (рис. 3.9).

```
Задание 2

Напишите условный оператор, который печатает число, если число чётное, и строку «нечётное», если число нечётное. Перепишите код, используя тернарный оператор.

[41]: numbers = [1, 2, 3, 4, 5, 6]
      for num in numbers
          if num%2 == 0
              println(num)
          else
              println("Нечетное")
          end
      end
      Нечетное
      2
      Нечетное
      4
      Нечетное
      6

[43]: for num in numbers
      println(num%2==0 ? num : "Нечетное")
      end
      Нечетное
      2
      Нечетное
      4
      Нечетное
      6
```

Рис. 3.9: Задание №2

3.3 Задание 3 и 4

Напишем функцию add_one, которая добавляет 1 к своему входу. В следующем задании используем map() или broadcast() для задания матрицы A, каждый элемент которой увеличивается на единицу по сравнению с предыдущим. (рис.

3.10)

```
Задание 3
Напишите функцию add_one, которая добавляет 1 к своему входу

[46]: function add_one(x)
      x = x + 1
      end

[46]: add_one (generic function with 1 method)
[47]: add_one(9)
[47]: 10

Задание 4
Используйте map() или broadcast() для задания матрицы A, каждый элемент которой увеличивается на единицу по сравнению с предыдущим

[48]: matrix = [1 2 3; 4 5 6; 7 8 9]
[48]: 3×3 Matrix{Int64}:
      1 2 3
      4 5 6
      7 8 9

[50]: matrix_plus_one = broadcast(x -> x + 1, matrix)
[50]: 3×3 Matrix{Int64}:
      2 3 4
      5 6 7
      8 9 10

[51]: matrix_plus_plus_one = map(x -> x + 1, matrix)
[51]: 3×3 Matrix{Int64}:
```

Рис. 3.10: Задание №3 и 4

3.4 Задание 5

Зададим матрицу A. Найдем A^3 . Заменим третий столбец матрицы A на сумму второго и третьего столбцов (рис. 3.11).

```
Задание 5
Задайте матрицу A и найдите
•  $A^3$ 
• Замените третий столбец матрицы A на сумму второго и третьего столбцов.

[52]: A = [1 1 3; 5 2 6; -2 -1 -3];
      A_cubed = A^3

[52]: 3×3 Matrix{Int64}:
      0 0 0
      0 0 0
      0 0 0

[54]: A_modified = copy(A);
      A_modified[:, 3] = A[:, 2] + A[:, 3];
      A_modified

[54]: 3×3 Matrix{Int64}:
      1 1 4
      5 2 8
      -2 -1 -4
```

Рис. 3.11: Задание №5

3.5 Задание 6 и 7

Создайте матрицу B с элементами с элементами $B_{i1} = 10$, $B_{i2} = -10$, $B_{i3} = 10$, $i = 1, 2, \dots$. Вычислить матрицу $C = B^T B$ (рис. 3.12). В следующем задании создадим матрицу Z размерности 6×6 , все элементы которой равны нулю, и матрицу E,

все элементы которой равны 1. Используя цикл while или for и закономерности, нужно повторить матрицы (рис. 3.13 - 3.15):

```

Задание 6

[58]: B = fill(0, (15, 3));

[57]: for i in 1:15
      B[i, 1] = 10
      B[i, 2] = -10
      B[i, 3] = 10
    end

[59]: C = transpose(B) * B

[59]: 3x3 Matrix{Int64}:
 0  0  0
 0  0  0
 0  0  0

Задание 7

Создайте матрицу Z размерности 6 x 6, все элементы которой равны нулю, и матрицу E, все элементы которой равны 1. Используя цикл while или for и закономерности расположения элементов, создайте матрицы размерности 6 x 6

[70]: Z = fill(0, (6, 6));
      E = fill(1, (6, 6));

```

Рис. 3.12: Задание №6 и 7

```

[84]: Z1 = fill(0, (6, 6))
      for i in 1:6
        for j in 1:6
          if abs(i - j) == 1
            Z1[i, j] = 1
          end
        end
      end
      Z1

[84]: 6x6 Matrix{Int64}:
 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

[85]: Z2 = fill(0, (6, 6))
      for i in 1:6
        for j in 1:6
          if i == j
            Z2[i, j] = 1
          elseif abs(i - j) == 2
            Z2[i, j] = 1
          end
        end
      end
      Z2

[85]: 6x6 Matrix{Int64}:
 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

```

Рис. 3.13: Задание №7

```

[91]: Z3 = fill(0, (6, 6));
      for i in 1:6
        for j in 1:6
          if i == 1 && (j == 4 || j == 6)
            Z3[i, j] = 1
          elseif i == 2 && (j == 3 || j == 5)
            Z3[i, j] = 1
          elseif i == 3 && (j == 2 || j == 4 || j == 6)
            Z3[i, j] = 1
          elseif i == 4 && (j == 1 || j == 3 || j == 5)
            Z3[i, j] = 1
          elseif i == 5 && (j == 2 || j == 4)
            Z3[i, j] = 1
          elseif i == 6 && (j == 1 || j == 3)
            Z3[i, j] = 1
          end
        end
      end
      Z3

[91]: 6x6 Matrix{Int64}:
 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

```

Рис. 3.14: Задание №7

```

[90]: Z4 = fill(0, (6, 6));
      for i in 1:6
        for j in 1:6
          if (i + j) % 2 == 0
            Z4[i, j] = 1
          end
        end
      end
      Z4

[90]: 6x6 Matrix{Int64}:
      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
      0  1  0  1  0  1

```

Рис. 3.15: Задание №7

3.6 Задание 8

Напишем свою функцию, аналогичную функции `outer()` языка R. Функция должна иметь следующий интерфейс: `outer(x, y, operation)` (рис. 3.16-3.18).

Задание 8

```

[90]: function outer(x, y, operation)
      """
      Аналог функции outer() из R
      x, y - векторы или матрицы
      operation - функция, применяемая к каждой паре элементов
      """
      x_vec = vec(x)
      y_vec = vec(y)

      m = length(x_vec)
      n = length(y_vec)

      result = Matrix{Any}(undef, m, n)

      for i in 1:m
        for j in 1:n
          result[i, j] = operation(x_vec[i], y_vec[j])
        end
      end

      return result
    end

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

[95]: A = [1, 2, 3];
      B = [4, 5, 6];

[96]: result1 = outer(A, B, *)

[96]: 3x3 Matrix{Any}:
      4  5  6
      8 10 12
      12 15 18

```

Рис. 3.16: Задание №8

Я люблю жирмаксить

```

[97]: result2 = outer(A, B, +)

[97]: 3x3 Matrix{Any}:
 5 6 7
 6 7 8
 7 8 9

[100]: A1 = outer(0:4, 0:4, +)

[100]: 5x5 Matrix{Any}:
 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

[102]: A2 = outer(0:4, 1:5, ~)

[102]: 5x5 Matrix{Any}:
 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

[104]: A3 = outer(0:4, 0:4, (i,j) -> (i + j) % 5)

[104]: 5x5 Matrix{Any}:
 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

```

Рис. 3.17: Задание №8

```

[105]: A4 = outer(0:9, 0:9, (i,j) -> (i + j) % 10)

[105]: 10x10 Matrix{Any}:
 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

[109]: A5 = outer(0:8, 0:8, (i,j) -> 1 - j < 0 ? 7 : 1 - j + 9 : 1 - j)

[109]: 9x9 Matrix{Any}:
 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. 3.18: Задание №8

3.7 Задание 9 и 10

Решим систему линейных уравнений с 5 неизвестными в 9 задании. В 10 задании произведем анализ количества элементов матрицы, удовлетворяющих необходимым условиям (рис. 3.19).

Задание 9

```
[122]: 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;
print(x)
[-2.0000000000000013, 3.0000000000000027, 4.9999999999999964, 2.0000000000000027, -4.000000000000001]
```

Задание 10

```
[115]: M = rand(1:10, 6, 10);
N = 4;
K = 75;
count_N = sum(M.>N)

[115]: 39

[117]: count_M = [1 for i=1:6 if sum(M[i, :])==2]

[117]: 2-element Vector{Int64}:
 1
 3

[120]: col = [(i, j) for i=1:6, j=2:5 if (i==j && sum(M[:,i] + M[:,j])>K)]

[120]: 3-element Vector{Tuple{Int64, Int64}}:
 (4, 3)
```

Рис. 3.19: Задание №9 и 10

3.8 Задание 11

Вычислим выражения (рис. 3.20).

Задание 11

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

[123]: 639215.2833333338

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

[125]: 89912.02146097131
```

Рис. 3.20: Задание №11

4 Выводы

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.