0.1 Front matter

title: “Лабораторная работа 3” author: “Петрушов Дмитрий, 1032212287”

0.2 Generic otions

lang: ru-RU toc-title: “Содержание”

0.3 Bibliography

bibliography: bib/cite.bib csl: pandoc/csl/gost-r-7-0-5-2008-numeric.csl

0.4 Pdf output format

toc: true # Table of contents toc-depth: 2 lof: true # List of figures lot: true # List of tables fontsize: 12pt linestretch: 1.5 papersize: a4 documentclass: scrreprt ## I18n polyglossia polyglossia-lang: name: russian options: - spelling=modern - babelshorthands=true polyglossia-otherlangs: name: english ## I18n babel babel-lang: russian babel-otherlangs: english ## Fonts mainfont: PT Serif romanfont: PT Serif sansfont: PT Sans monofont: PT Mono mainfontoptions: Ligatures=TeX romanfontoptions: Ligatures=TeX sansfontoptions: Ligatures=TeX,Scale=MatchLowercase monofontoptions: Scale=MatchLowercase,Scale=0.9 ## Biblatex biblatex: true biblio-style: “gost-numeric” biblatexoptions: - parentracker=true - backend=biber - hyperref=auto - language=auto - autolang=other* - citestyle=gost-numeric ## Pandoc-crossref LaTeX customization figureTitle: “Рис.” tableTitle: “Таблица” listingTitle: “Листинг” lofTitle: “Список иллюстраций” lotTitle: “Список таблиц” lolTitle: “Листинги” ## Misc options indent: true header-includes: -

- **keep figures where there are in the text**
  - # keep figures where there are in the text —# Цель работы

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

1 Выполнение лабораторной работы

1.1 Циклы while и for

Для различных операций, связанных с перебором индексируемых элементов структур данных, традиционно используются циклы while и for.

Синтаксис while

```
while <условие>
    <тело цикла>
end
```

Примеры использования цикла while (рис. [??] - рис. [??]):

```
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
```

```
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!
```

Такие же результаты можно получить при использовании цикла for.

Синтаксис for

```
for <переменная> in <диапазон>
    <тело цикла>
end
```

Примеры использования цикла for (рис. [1]):

```
for n in 1:2:10
    println(n)
end
myfriends = ["Ted", "Robyn", "Barney", "Lily", "Marshall"]
for friend in myfriends
    println("Hi $friend, it's great to see you!")
end
```

```
1
3
5
7
9
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!
```

Figure 1: Примеры использования цикла for

Пример использования цикла for для создания двумерного массива, в котором значение каждой записи является суммой индексов строки и столбца (рис. [2]):

```
m, n = 5, 5
A = fill(0, (m,n))
for i  in 1:m
    for j in 1:n
        A[i,j] =i+j
    end
end
```

```
B = fill(0, (m, n))
for i in 1:m, j in 1:n
    B[i, j] = i + j
end
```

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

```
5×5 Matrix{Int64}:
 2  3  4  5   6
 3  4  5  6   7
 4  5  6  7   8
```

Figure 2: Пример использования цикла for для создания двумерного массива

1.2 Условные выражения

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

Синтаксис условных выражений с ключевым словом:

```
if <условие 1>
    <действие 1>
elseif <условие 2>
    <действие 2>
else
    <действие 3>
end
```

Примеры использования условного выражения (рис. [??]):

```
1]: 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
```

Примеры использования условного выражения (рис. [3]):

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

12]: 10
```

Figure 3: Примеры использования условного выражения

1.3 Функции

Julia дает нам несколько разных способов написать функцию.

Примеры способов написания функции (рис. [4]):

```
]: 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)
```

Figure 4: Примеры способов написания функции

По соглашению в Julia функции, сопровождаемые восклицательным знаком, изменяют свое содержимое, а функции без восклицательного знака не делают этого (рис. [5]):

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

```
: 3-element Vector{Int64}:
   2
   3
   5
```

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

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

Figure 5: Сравнение результатов вывода

(рис. [6]):

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

[17]: 3-element Vector{Int64}:
        1
        8
       27
```

Figure 6: Сравнение результатов вывода

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.

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.

Примеры использования функций map() и broadcast() (рис. [7]):

```
: f(x) = x^2
  broadcast(f, [1, 2, 3])

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

Figure 7: Примеры использования функций map() и broadcast()

1.4 Сторонние библиотеки (пакеты) в Julia

Julia имеет более 2000 зарегистрированных пакетов, что делает их огромной частью экосистемы Julia. Есть вызовы функций первого класса для других языков, обеспечивающие интерфейсы сторонних функций. Можно вызвать функции из Python или R, например, с помощью PyCall или Rcall.

С перечнем доступных в Julia пакетов можно ознакомиться на страницах следующих ресурсов: - https://julialang.org/packages/ - https://juliahub.com/ui/Home - https://juliaobserver.com/ - https://github.com/svaksha/Julia.jl

При первом использовании пакета в вашей текущей установке Julia вам необходимо использовать менеджер пакетов, чтобы явно его добавить:

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

При каждом новом использовании Julia (например, в начале нового сеанса в REPL или открытии блокнота в первый раз) нужно загрузить пакет, используя ключевое слово using:

Например, добавим и загрузим пакет Colors:

```
Pkg.add("Colors")
using Colors
```

Затем создадим палитру из 100 разных цветов:

```
palette = distinguishable_colors(100)
```

А затем определим матрицу 3 × 3 с элементами в форме случайного цвета из палитры, используя функцию rand:

```
rand(palette, 3, 3)
```

1.5 Самостоятельная работа

Выполнение задания №1 (рис. [8] - рис. [11]):

```
for i in 1:100
    println(i)
end
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
```

Figure 8: Выполнение подпунктов задания №1

```
i = 1
while i <= 100
    println(i^2)
    i+=1
end
```

```
1
4
9
16
25
36
49
64
81
100
121
144
169
196
225
256
289
324
361
400
441
484
529
576
625
676
729
784
841
```

Figure 9: Выполнение подпунктов задания №1

```
squares= Dict{Int64, Int64}()
for i in 1:100
    push!(squares, i=> i^2)
end
pairs(squares)
```

```
Dict{Int64, Int64} 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
  86 => 7396
  91 => 8281
  62 => 3844
  58 => 3364
  52 => 2704
  12 => 144
  28 => 784
```

Figure 10: Выполнение подпунктов задания №1

```
i=1
while i<=100
    append!(squares_arr,i^2)
    i+=1
end
squares_arr
```

```
300-element Vector{Any}:
     1
     4
     9
    16
    25
    36
    49
    64
    81
   100
   121
   144
   169
```

Figure 11: Выполнение подпунктов задания №1

Выполнение задания №2 (рис. [12]):

```
N=10
if N % 2 == 0
    println("четное")
else
    println("нечетное")
end
```

```
четное
```

```
N=10
(N % 2 == 0) ? println(" четное") : println(" нечетное")
```

```
 четное
```

Figure 12: Выполнение задания №2

Выполнение задания №3 (рис. [13]):

```
function add_one(x)
    x+1
end
add_one(2)
```

```
3
```

Figure 13: Выполнение задания №3

Выполнение задания №4 (рис. [14]):

```
x= fill(1, 3 * 3)
q=collect(0:(length(x)-1))
x=reshape(map(+,x,q),(3,3))
```

```
3×3 Matrix{Int64}:
 1  4  7
 2  5  8
 3  6  9
```

Figure 14: Выполнение задания №4

Выполнение задания №5 (рис. [15]):

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

```
]: 3×3 Matrix{Int64}:
    1   1   3
    5   2   6
   -2  -1  -3
```

```
]: A^3
```

```
]: 3×3 Matrix{Int64}:
   0  0  0
   0  0  0
   0  0  0
```

```
]: for i in 7:1:9
       A[i]+= A[i-3]
   end
   A
```

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

Figure 15: Выполнение задания №5

Выполнение задания №6 (рис. [16]):

```
B = Array{Int32, 2}(undef,15,3)
for i in 1:15
    B[i,1] = 10
    B[i,2] = -10
    B[i,3] = 10
end
B
```

```
15×3 Matrix{Int32}:
 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
```

```
C = (B')*B
C
```

```
3×3 Matrix{Int32}:
  1500  -1500   1500
 -1500   1500  -1500
  1500  -1500   1500
```

Figure 16: Выполнение задания №6

Выполнение задания №7 (рис. [17] - рис. [19]):

```
z = zeros(Int64,6,6)
```

```
6×6 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  0  0  0  0  0
 0  0  0  0  0  0
```

```
e= ones(Int64,6,6)
```

```
6×6 Matrix{Int64}:
 1  1  1  1  1  1
 1  1  1  1  1  1
 1  1  1  1  1  1
 1  1  1  1  1  1
 1  1  1  1  1  1
 1  1  1  1  1  1
```

```
z1=z
for i in 1:6
    if i != 1
        z1[i, i-1] = e[i,i-1]
    end
    if i != 6
        z1[i,i+1] = e[i,i+1]
    end
end
z1
```

```
6×6 Matrix{Int32}:
 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
```

Figure 17: Выполнение задания №7

```
z2 =z
for i in 1:1:6
    z2[i,i]= 1
    if (i+2 <=6)
        z2[i,i+2]=e[i,i+2]
    end
    if (i-2 >=1)
        z2[i,i-2]=e[i,i-2]
    end
end
z2
```

```
6×6 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
```

```
z3 = z
for i in 1:1:6
    if (9-i <=6)
        z3[i,9-i]=e[i,9-i]
    end
    if (5-i>=1)
        z3[i,5-i]=e[i,5-i]
    end
end
z3
```

```
6×6 Matrix{Int64}:
 1  0  1  1  0  0
 0  1  1  1  0  0
 1  1  1  0  1  1
 1  1  0  1  1  1
 0  0  1  1  1  0
 0  0  1  1  0  1
```

Figure 18: Выполнение задания №7

```
z4 = z
for i in 1:6
    z4[i,i]=1
    if(i+2 <=6) z4[i,i+2] = e[i,i+2] end
    if(i-2 >=1) z4[i,i-2] = e[i,i-2] end
    if(i+4 <=6) z4[i,i+4] = e[i,i+4] end
    if(i-4 >=1) z4[i,i-4] = e[i,i-4] end
end
z4
```

```
6×6 Matrix{Int64}:
 1  0  1  1  1  0
 0  1  1  1  0  1
 1  1  1  0  1  1
 1  1  0  1  1  1
 1  0  1  1  1  0
 0  1  1  1  0  1
```

Figure 19: Выполнение задания №7

2 Вывод

В ходе выполнения лабораторной работы было освоено применение циклов функций и сторонних для Julia пакетов для решения задач линейной алгебры и работы с матрицами.

Список литературы