

Лабораторная работа №3. Компьютерный практикум по статистическому анализу данных

Леснухин Даниил Дмитриевич
Российский университет дружбы народов
Москва

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.
2. Выполните задания для самостоятельной работы (раздел 3.4)

Теоретическое введение

Julia – высокоуровневый свободный язык программирования с динамической типизацией, созданный для математических вычислений [[@julialang](#)]. Эффективен также и для написания программ общего назначения. Синтаксис языка схож с синтаксисом других математических языков, однако имеет некоторые существенные отличия.

Для выполнения заданий была использована официальная документация Julia [[@juliadoc](#)].

Выполнение лабораторной работы

Для начала выполним примеры из лабораторной работы, чтобы познакомиться с циклами, условными операторами, функциями и работой со сторонними библиотеками .

Теперь перейдем к выполнению заданий для самостоятельной работы.

Используя циклы `while` и `for`

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

Напишем функцию `add_one`, которая добавляет 1 к своему входу .

```
Циклы while и for

[1] 0
    0 000.
      n = 0
      while n < 10
      n += 1
      println(n)
      end

*** 1
    2
    3
    4
    5
    6
    7
    8
    9
   10

[3] 0
    0 000.
      i = 1
      myfriends = ["Ted", "Robyn", "Barney", "Lily", "Marshall"]
      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!

[4] 0
    0 000.
      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!

[5] 0
    0 000.
      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
      A

5x5 Matrix{Int64}:
```

Рис. 1: Выполнение примеров с циклами

Условные выражения

```
[9] 0 ✓ свк.
# используем '&&' для реализации операции "AND"
# операция % вычисляет остаток от деления
N = 99
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
```

▼ Fizz

```
[10] 0 ① свк.
if a
b
else
c
end
```

▼ *** UndefinedVarError: `a` not defined in `Main`
Suggestion: check for spelling errors or missing imports.

Stacktrace:
[1] top-level scope
@ In[10]:1

Далее: [Объяснить ошибку](#)

Рис. 2: Выполнение примеров с условными выражениями

Функции

```
[11] 0 ✓ свк.
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)

```
[12] 0 ✓ свк.
A = [1 + 3*j for j in 0:2, i in 1:3]
```

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

Рис. 3: Выполнение примеров с функциями

```
Сторонние библиотеки (пакеты) в Julia

[13]
✓ 43
DBCL

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`
[7876af07] + Example v0.5.5
Updating `~/.julia/environments/v1.11/Manifest.toml`
[7876af07] + Example v0.5.5
Precompiling project...
5409.9 ms ✓ Example
1 dependency successfully precompiled in 18 seconds. 468 already precompiled.


[14]
✓ 11
DBCL

Pkg.add("Colors")
using Colors

***
Resolving package versions...
Updating `~/.julia/environments/v1.11/Project.toml`
[5ae59095] + Colors v0.13.1
No Changes to `~/.julia/environments/v1.11/Manifest.toml`

[15]
✓ 2
DBCL

palette = distinguishable_colors(100)

***


[19]
✓ 0
DBCL

rand(palette, 3, 3)

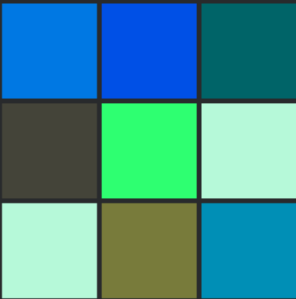
***

```

Рис. 4: Выполнение примеров со сторонними библиотеками

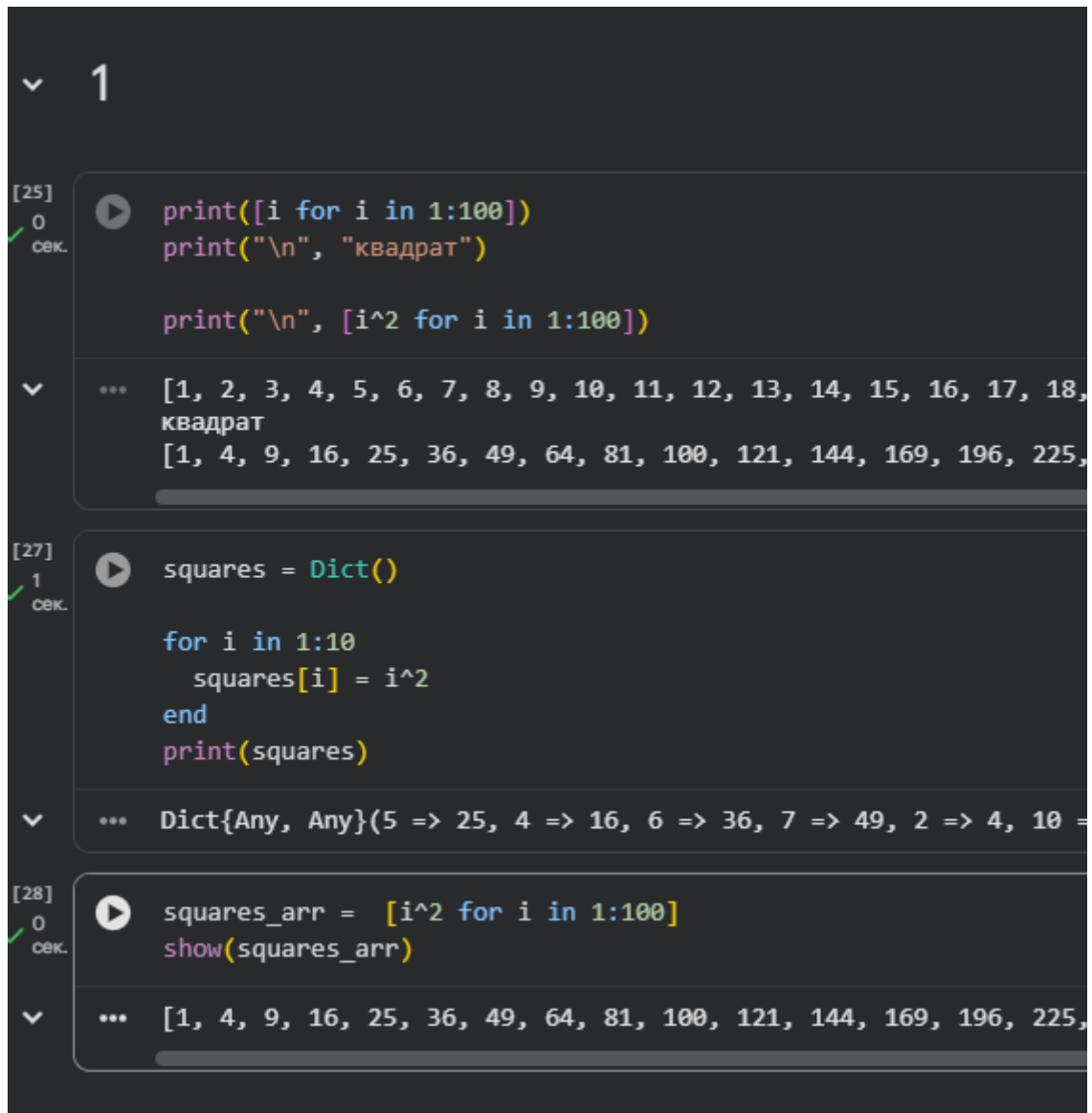


Рис. 5: Задание №1

2

[30] 1 сек.

```
a = readline()

a = parse(Int64, a)
if a % 2 == 0
    println(a)
else
    print("Нечетное")
end
```

... stdin> 2
2

Рис. 6: Задание №2

3

[32] 0 сек.

```
function add_one(num)
    num += 1
end
```

add_one (generic function with 1 method)

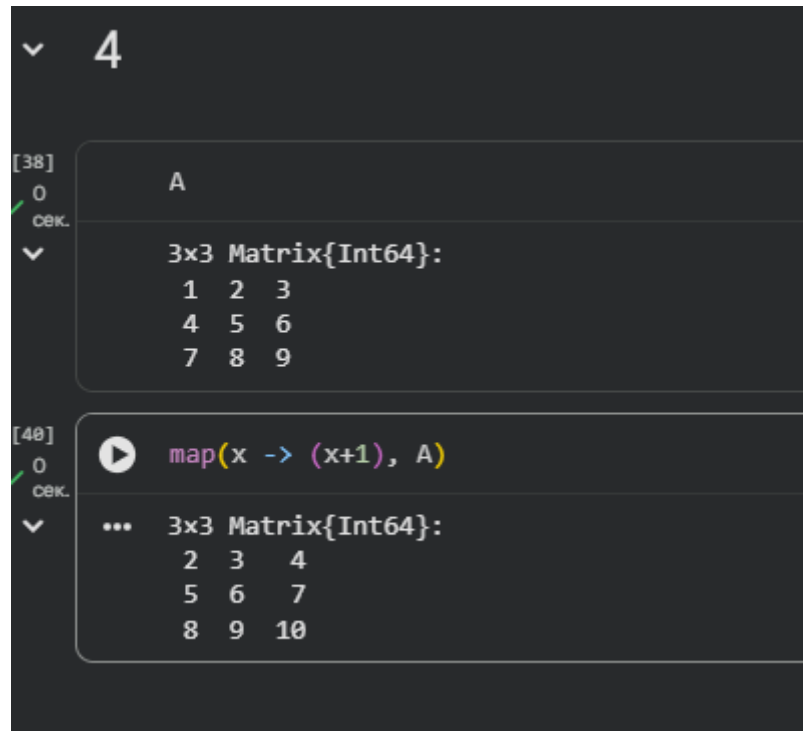
[33] 0 сек.

```
add_one(5)
```

... 6

Рис. 7: Задание №3

Используем `map()` или `broadcast()` для задания матрицы A , каждый элемент которой увеличивается на единицу по сравнению с предыдущим.



```
4

[38] 0 сек.
A
3x3 Matrix{Int64}:
 1  2  3
 4  5  6
 7  8  9

[40] 0 сек.
map(x -> (x+1), A)
... 3x3 Matrix{Int64}:
 2  3  4
 5  6  7
 8  9 10
```

Рис. 8: Задание №4

Зададим матрицу A . Найдем A^3 . Заменяем третий столбец матрицы A на сумму второго и третьего столбцов.

Создадим матрицу B с элементами $B_{i1} = 10$, $B_{i2} = -10$, $B_{i3} = 10$, $i = 1, 2, \dots, 15$. Вычислим матрицу $C = B^T B$.

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 расположения элементов, создадим следующие матрицы размерности 6×6 .


В 10 задании произведем анализ количества элементов матрицы, удовлетворяющих необходимым условиям. Вычислим выражения ([рис. @fig-015]).

Выводы


В результате выполнения данной лабораторной работы я освоил применение циклов функций и сторонних для Julia пакетов для решение задач линейной алгебры и работы с матрицами.

5


Новый раздел

[41] 0 сек.
✓  `A = [1 1 3; 5 2 6; -2 -1 -3]`

▼ ... 3×3 Matrix{Int64}:
1 1 3
5 2 6
-2 -1 -3

[42] 0 сек.
✓  `g(x) = x^3`
`B = g.(A)`

▼ ... 3×3 Matrix{Int64}:
1 1 27
125 8 216
-8 -1 -27

[43] 0 сек.
✓  `for i in 1:3`
`A[i, 3] = A[i, 2] + A[i, 3]`
`end`
`A`

▼ ... 3×3 Matrix{Int64}:
1 1 4
5 2 8
-2 -1 -4

Рис. 9: Задание №5

✓ 6

```
49] B = [j % 2 == 0 ? -10 : 10 for i in 1:15, j in 1:3]
0   C = B' * B
сек.
```

✓ ... 3×3 Matrix{Int64}:
1500 -1500 1500
-1500 1500 -1500
1500 -1500 1500

Рис. 10: Задание №6

```
Z = fill(0, 6, 6)
E = fill(1, 6, 6)

Z1 = copy(Z)
for i in 1:6, j in 1:6
    if abs(i-j)==1
        Z1[i, j] = 1
    end
end
Z1

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

▶ Z2 = copy(Z)
for i in 1:6, j in 1:6
    if i == j || abs(i-j)==2
        Z2[i, j] = 1
    end
end
Z2

... 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. 11: Задание №7

```

Z3 = copy(Z)
for i in 1:6, j in 1:6
    if (6 - i + 1) == j || abs(6 - i + 1 - j) == 2
        Z3[i, j] = 1
    end
end
Z3

*** 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

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

*** 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. 12: Задание №8. Реализация функции `outer()`. Проверка работы функции

11

[83]
✓ 0 сек.

▶

```
sum1 = sum(i^4 / (3 + j) for i in 1:20, j in 1:5)
sum2 = sum(i^4 / (3 + i*j) for i in 1:20, j in 1:5)
print("\n", sum1)
print("\n", sum2)
```

▼

...
639215.2833333338
89912.02146097131

Рис. 13: Задание №10. Задание №11