

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 3

дисциплина: Компьютерный практикум

по статистическому данным анализ

Студент: Доре Стевенсон Эдгар

Группа: НКН-бд-01-19

МОСКВА

2022 г.

Постановка задачи

Основная цель работы — изучить несколько структур данных, реализованных в Julia, научиться применять их и операции над ними для решения задач.

1. Используя Jupyter Lab, повторите примеры
2. Выполните задания для самостоятельной работы.

Выполнение работы

1. Повторение примеров

```
In [1]: # 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
```

```
In [2]: 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!
```

```
In [3]: 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!
```

```
In [4]: 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
```

```
Out[4]: 5x5 Array{Int64,2}:
```

```
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
```

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

```
Out[5]: 5x5 Array{Int64,2}:
 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
```

```
In [6]: C = [i + j for i in 1:m, j in 1:n]
        C
```

```
Out[6]: 5x5 Array{Int64,2}:
 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
```

```
In [7]: # используем `&&` для реализации операции "AND"
        # операция % вычисляет остаток от деления
        N=10
        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

```
In [8]: x = 5
        y = 10
        (x > y) ? x : y
```

```
Out[8]: 10
```

```
In [9]: function sayhi(name)
        println("Hi $name, it's great to see you!")
        end
        # функция возведения в квадрат:
        function f(x)
            x^2
        end
```

Out[9]: f (generic function with 1 method)

```
In [10]: sayhi("C-3PO")
         f(42)
```

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

Out[10]: 1764

```
In [11]: sayhi2(name) = println("Hi $name, it's great to see you!")
         f2(x) = x^2
```

Out[11]: f2 (generic function with 1 method)

```
In [12]: sayhi3 = name -> println("Hi $name, it's great to see you!")
         f3 = x -> x^2
```

Out[12]: #5 (generic function with 1 method)

```
In [13]: # задаём массив v:
         v = [3, 5, 2]
         sort(v)
         v
         sort!(v)
         v
```

Out[13]: 3-element Array{Int64,1}:
2
3
5

```
In [14]: map(f, [1, 2, 3])
```

Out[14]: 3-element Array{Int64,1}:
1
4
9

```
In [15]: map(x -> x^3, [1, 2, 3])
```

```
Out[15]: 3-element Array{Int64,1}:  
 1  
 8  
27
```

```
In [16]: f(x) = x^2  
broadcast(f, [1, 2, 3])
```

```
Out[16]: 3-element Array{Int64,1}:  
 1  
 4  
 9
```

```
In [17]: f.([1, 2, 3])  
# Задаём матрицу A:  
A = [i + 3*j for j in 0:2, i in 1:3]  
# Вызываем функцию f возведения в квадрат  
f(A)
```

```
Out[17]: 3x3 Array{Int64,2}:  
 30  36  42  
 66  81  96  
102 126 150
```

```
In [18]: A .+ 2 .* f.(A) ./ A  
@. A + 2 * f(A) / A  
broadcast(x -> x + 2 * f(x) / x, A)
```

```
Out[18]: 3x3 Array{Float64,2}:  
 3.0  6.0  9.0  
12.0 15.0 18.0  
21.0 24.0 27.0
```

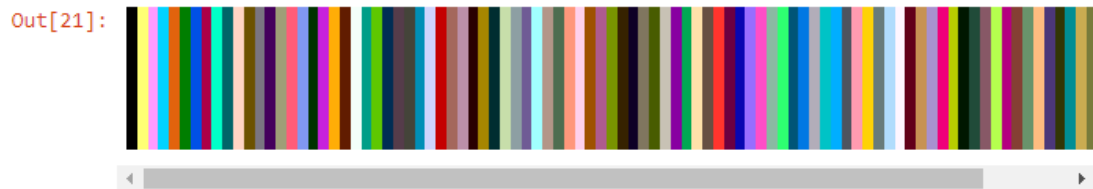
```
In [19]: import Pkg  
Pkg.add("Example")
```

```
Updating registry at `C:\Users\Admin\.julia\registries\General`  
Resolving package versions...  
No Changes to `C:\Users\Admin\.julia\environments\v1.5\Project.toml`  
No Changes to `C:\Users\Admin\.julia\environments\v1.5\Manifest.toml`
```

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

```
Resolving package versions...  
No Changes to `C:\Users\Admin\.julia\environments\v1.5\Project.toml`  
No Changes to `C:\Users\Admin\.julia\environments\v1.5\Manifest.toml`
```

```
In [21]: palette = distinguishable_colors(100)
```



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



2. Выполните задания для самостоятельной работы

```
In [23]: #выведите на экран целые числа от 1 до 100 и напечатайте их квадраты;  
for n in 1:100  
    println(n, " ", n^2)  
end
```

```
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  
39 1521  
40 1600  
41 1681  
42 1764  
43 1849  
44 1936  
45 2025  
46 2116  
47 2209  
48 2304  
49 2401  
50 2500  
51 2601  
52 2704  
53 2809  
54 2916  
55 3025  
56 3136  
57 3249  
58 3364  
59 3481  
60 3600  
61 3721  
62 3844  
63 3969  
64 4096  
65 4225  
66 4356  
67 4489  
68 4624  
69 4761  
70 4900  
71 5041  
72 5184  
73 5329  
74 5476  
75 5625  
76 5776  
77 5929  
78 6084  
79 6241  
80 6400  
81 6561  
82 6724  
83 6889  
84 7056  
85 7225  
86 7396  
87 7569  
88 7744  
89 7921  
90 8100  
91 8281  
92 8464  
93 8649  
94 8836  
95 9025  
96 9216  
97 9409  
98 9604  
99 9801  
100 10000
```

```
squares = Dict{Int{1},Int{1}}()  
for key in sort(collect(keys(squares)))  
    println("$key => ${squares[key]}")  
end
```

```
1 => 1  
2 => 4  
3 => 9  
4 => 16  
5 => 25  
6 => 36  
7 => 49  
8 => 64  
9 => 81  
10 => 100
```

```
In [3]: # - создайте массив squares_arr, содержащий квадраты всех чисел от 1 до 100  
squares_arr = [i^2 for i=1:100]
```

```
Out[3]: 100-element Array{Int64,1}:
```

```
1  
4  
9  
16  
25  
36  
49  
64  
81  
100  
121  
144  
169  
:  
7921  
8100  
8281
```



```
jupyter lab03-m Last Checkpoint: 01/31/2023 (autosaved) Logout
File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.5.0 C

N=6
if (N%2==0)
    println(N)
else
    println("Нечетное")
end

6

In [5]: #Перепишите код, используя тернарный оператор
(N%2==0) ? println(N) : println("Нечетное")

6

In [6]: #3 Напишите функцию add_one, которая добавляет 1 к своему входу.
add_one(x) = x+1

Out[6]: add_one (generic function with 1 method)

In [7]: add_one(4)

Out[7]: 5

In [8]: #Используйте map() или broadcast() для задания матрицы A, каждый элемент которой
        #увеличивается на единицу по сравнению с предыдущим
m, n = 5, 5
A = fill(0, (m,n))
map(x-> add_one(x), A)

Out[8]: 5x5 Array{Int64,2}:
 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

In [10]: # 5. Задайте матрицу A следующего вида:
A = [1 1 3; 5 2 6; -2 -1 -3]
```

In [31]: A^3

Out[31]: 3x3 Array{Int64,2}:
0 0 0
0 0 0
0 0 0

In [32]: $A[:,3]=A[:,1]+A[:,2]$
A

Out[32]: 3x3 Array{Int64,2}:
1 1 2
5 2 7
-2 -1 -3

In [33]: #Создайте матрицу B
B=repeat([10 -10 10], 15)

Out[33]: 15x3 Array{Int64,2}:
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

In [34]: $C=B'*B$

Out[34]: 3x3 Array{Int64,2}:
1500 -1500 1500
-1500 1500 -1500
1500 -1500 1500

```
In [35]: #создайте матрицы Z повторением элементов
Z=zeros(6,6)
E=ones(6,6)
Z1=zeros(6,6)
Z2=zeros(6,6)
Z3=zeros(6,6)
Z4=zeros(6,6)
for i in 1:6
    for j in 1:6
        if(i==j-1) || (i==j+1)
            Z1[i,j]=1
        end
        if(i==j) || (i==j+2) || (i==j-2)
            Z2[i,j]=1
        end
        if(i==5-j) || (i==7-j) || (i==9-j)
            Z3[i,j]=1
        end
        if(i+j)%2==0
            Z4[i,j]=1
        end
    end
end
```

```
In [36]: Z1
```

```
Out[36]: 6x6 Array{Float64,2}:
 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
```

```
In [37]: Z2
```

```
Out[37]: 6x6 Array{Float64,2}:
 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
```

In [38]: Z3

```
Out[38]: 6x6 Array{Float64,2}:
 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
```

In [39]: Z4

```
Out[39]: 6x6 Array{Float64,2}:
 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
```

```
In [40]: function outer(X, Y, op)
           if (ndims(X)==1)
               X=reshape(X, (size(X,1), size(X,2)))
           end
           if (ndims(Y)==1)
               Y=reshape(Y, (size(Y,1), size(Y,2)))
           end
           Z=zeros(size(X)[1], size(Y)[2])
           for i in 1:size(X)[1], j in 1:size(Y)[2], k in size(X)[2]
               Z[i, j] = op(X[i,k], Y[k,j])
           end
           return Z
       end
```

Out[40]: outer (generic function with 1 method)

In [41]: M1=[i for i in 0:4]

```
Out[41]: 5-element Array{Int64,1}:
 0
 1
 2
 3
 4
```

```
In [42]: outer(M1, M1', +)
```

```
Out[42]: 5x5 Array{Float64,2}:
 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
```

```
In [43]: M2=[i for i in 1:5]
outer(M1, M2', ^)
```

```
Out[43]: 5x5 Array{Float64,2}:
 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
```

```
In [44]: outer(M1, M1', +).%5
```

```
Out[44]: 5x5 Array{Float64,2}:
 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
```

```
In [45]: outer([i for i in 0:9], [i for i in 0:9]', +).%10
```

```
Out[45]: 10x10 Array{Float64,2}:
 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
```

```
In [46]: outer([i for i in 0:8], [i for i in -9:-1]', -).%9
```

```
Out[46]: 9x9 Array{Float64,2}:
 0.0  8.0  7.0  6.0  5.0  4.0  3.0  2.0  1.0
 1.0  0.0  8.0  7.0  6.0  5.0  4.0  3.0  2.0
 2.0  1.0  0.0  8.0  7.0  6.0  5.0  4.0  3.0
 3.0  2.0  1.0  0.0  8.0  7.0  6.0  5.0  4.0
 4.0  3.0  2.0  1.0  0.0  8.0  7.0  6.0  5.0
 5.0  4.0  3.0  2.0  1.0  0.0  8.0  7.0  6.0
 6.0  5.0  4.0  3.0  2.0  1.0  0.0  8.0  7.0
 7.0  6.0  5.0  4.0  3.0  2.0  1.0  0.0  8.0
 8.0  7.0  6.0  5.0  4.0  3.0  2.0  1.0  0.0
```

```
In [47]: #решить систему
A=outer([i for i in 1:5], [i for i in 1:5]',-)
A=abs.(A)
A=A.+1
y=[7, -1, -3, 5, 17]
x=inv(A)*y
```

```
Out[47]: 5-element Array{Float64,1}:
 -2.0
  3.0
 4.999999999999999
 1.9999999999999998
 -3.999999999999999
```

```
In [51]: N=4
for i in 1:6
    c=0
    for j in 1:10
        if M[i,j]>N
            c=c+1
        end
    end
    println(" line ", i, "contains ", c, " elements > ", N)
end
```

```
line 1contains 6 elements > 4
line 2contains 3 elements > 4
line 3contains 6 elements > 4
line 4contains 5 elements > 4
line 5contains 6 elements > 4
line 6contains 7 elements > 4
```

```
In [52]: N=4
for i in 1:6
    c=0
    for j in 1:10
        if M[i,j]==N
            c=c+1
        end
    end
    if c==2
        println(" строка ", i, "содержит число ", N, " ровно 2 раза ")
    end
end
```

```
строка 3содержит число4 ровно 2 раза
```

```
In [57]: K=75
sum=0
for j in 1:10
    for k in (j+1):10
        for i in 1:6
            sum=sum + M[i,j]+M[i,k]
        end
        if sum > K
            println(" Сумма столбцов ", j, " ", k, " = ", sum)
        end
        sum =0
    end
end
```

```
Сумма столбцов 2 6 = 76
Сумма столбцов 2 7 = 82
Сумма столбцов 2 9 = 77
Сумма столбцов 6 7 = 76
Сумма столбцов 7 9 = 77
```

```
In [60]: # Вычислить сумму
```

```
In [52]: #Вычислите суммы
Arr=[(i^4)/(3+j) for j in 1:5, i in 1:20]
sum = 0
for i in 1:5
    for j in 1:20
        sum = sum + Arr[i,j]
    end
end
sum
```

```
Out[52]: 639215.2833333338
```

```
In [53]: Arr=[(i^4)/(3+i*j) for j in 1:5, i in 1:20]
sum = 0
for i in 1:5
    for j in 1:20
        sum = sum + Arr[i,j]
    end
end
sum
```

```
Out[53]: 89912.02146097131
```

Выводы

Получал навыки работы с матрицами и функциями в Julia